Manual

TE2000 TC3 HMI Engineering

TwinCAT

Version: 1.6
Date: 2020-01-22
Order No.: TE2000

BECKHOFF

Table of contents

1 Foreword ... 9
1.1 Notes on the documentation ... 9
1.2 Safety instructions ... 10
2 Overview ... 11
2.1 Product description ... 11
2.2 Scenarios ... 11
2.3 Architecture ... 15
2.3.1 Engineering ... 15
2.3.2 Client/Server ... 15
2.4 Business model ... 16
3 Installation ... 17
3.1 System requirements ... 17
3.2 Version overview ... 17
3.3 Installation ... 18
3.4 Licensing ... 21
4 Quick start ... 27
4.1 HMI Engineering interface ... 27
4.2 Adding controls ... 29
4.3 Adding events ... 30
4.4 Adding an internal variable ... 31
4.5 Creating a data binding ... 31
4.6 Navigation concept ... 34
4.6.1 Creating navigation ... 34
4.7 Change the language ... 34
4.7.1 Adding a new language ... 34
4.7.2 Manipulating translations in the Localization editor ... 36
4.8 User Control ... 37
4.8.1 Creating a User Control ... 37
5 Development environment ... 42
5.1 Solution Explorer ... 42
5.1.1 Project properties ... 43
5.2 Designer ... 46
5.2.1 Graphic editor: ... 47
5.2.2 HTML editor ... 48
5.3 Properties ... 49
5.4 Actions and Conditions Editor ... 50
5.4.1 General ... 51
5.4.2 Controls ... 57
5.4.3 Functions ... 58
5.4.4 Action Templates ... 59
5.5 Document Outline ... 62
5.6 LiveView ... 62
5.7 TwinCAT HMI Configuration ... 65

5.7.1 Server symbols ... 67
5.7.2 Internal symbols ... 73
5.7.3 Localizations ... 73
5.7.4 Data types ... 75
5.7.5 Parameter ... 83
5.7.6 Controls ... 84
5.7.7 Global Events ... 85
5.7.8 Functions ... 86
5.7.9 Files ... 87
5.7.10 Users and user groups ... 88
5.7.11 Client certificates ... 88
5.7.12 Action Templates ... 89
5.7.13 Recipe management ... 89
5.7.14 Permissions Management ... 90
5.8 Localization editor ... 90
5.9 Symbols and Bindings ... 91
5.9.1 Create binding ... 92
5.9.2 Create function binding ... 94
5.9.3 Configure binding ... 95
5.10 Functions ... 97
5.10.1 Binding ... 97
5.10.2 Data Conversion ... 99
5.10.3 Formatting ... 103
5.10.4 Locale ... 105
5.10.5 Recipe ... 106
5.10.6 Theme ... 108
5.10.7 UserManagement ... 109
5.11 Publishing ... 110
5.11.1 Autostart Server ... 114
5.11.2 Autostart Client ... 115
5.12 Server configuration ... 117
5.13 Package Management ... 118
5.13.1 Creating a package ... 119
5.13.2 Installing a package ... 123
5.13.3 Uninstalling a package ... 125
5.13.4 Updating a package ... 127
5.14 Gallery Explorer ... 127
5.15 TwinCAT HMI Toolbar ... 128
5.16 TwinCAT HMI Items ... 129

6 Controls ... 132
6.1 General configuration ... 134
6.1.1 Positioning and calculation of the size of controls ... 134
6.1.2 Color editor ... 136
6.1.3 Transformation editor ... 138
6.1.4 Box Shadow Editor ... 140
6.2 Beckhoff Controls ... 141

6.2.1 Bar Chart ... 142
6.2.2 Button ... 167
6.2.3 Checkbox ... 179
6.2.4 Combobox ... 185
6.2.5 Datagrid ... 203
6.2.6 Ellipse ... 229
6.2.7 Event Grid ... 235
6.2.8 IFrame ... 246
6.2.9 Image ... 251
6.2.10 Keyboard ... 258
6.2.11 Line ... 272
6.2.12 Line Chart ... 281
6.2.13 Linear Gauge ... 306
6.2.14 Polygon ... 329
6.2.15 Radial Gauge ... 337
6.2.16 Rectangle ... 359
6.2.17 Textblock ... 367
6.2.18 Textbox ... 376
6.2.19 Toggle Button ... 386
6.2.20 Trend Line Chart ... 393
6.2.21 Video ... 421
6.3 System Controls ... 433
6.3.1 Control ... 433
6.3.2 Container Control ... 459
6.3.3 Container ... 465
6.3.4 Content ... 470
6.3.5 Grid ... 475
6.3.6 HTML host ... 486
6.3.7 Partial ... 490
6.3.8 Region ... 496
6.3.9 User Control ... 504
6.3.10 User Control Host ... 509
6.3.11 View ... 515
6.4 User controls ... 520
6.5 Framework controls ... 520
6.6 Control rights ... 520

7 User management ... 521
7.1 Users and user groups ... 521
7.1.1 Create new user ... 521
7.1.2 Changing user properties ... 523
7.1.3 Activate auto-login user ... 523
7.1.4 Deleting a user ... 524
7.1.5 Creating a new user group ... 524
7.1.6 Changing user group properties ... 525
7.1.7 Deleting a user group ... 526
7.1.8 System users and system user groups ... 526

7.2 Permissions system ... 527
7.2.1 Control level ... 527
7.2.2 Symbol level ... 530
7.2.3 File level ... 532
7.3 Access management in the client ... 533
7.4 Login page ... 534
7.4.1 Add login page ... 535
7.4.2 Switch login page ... 536
7.4.3 Edit login page ... 537
7.5 User management in the client ... 540
7.6 Client certificates ... 540
7.6.1 Creating a Client Certificate ... 541
7.6.2 Assigning a Client Certificate ... 542
7.6.3 Activating Client Certificates ... 543
7.6.4 Installing the Client certificate ... 544

8 Event system ... 552
8.1 TcHmiEventlogger extension ... 552
8.2 Event Grid Control ... 553

9 Internationalization ... 557
9.1 Change the language ... 557
9.2 Unit conversion ... 557
9.3 Localization ... 558

10 Themes ... 559
10.1 Introduction ... 559
10.2 Theme editor ... 561
10.2.1 Class theme ... 562
10.2.2 Control theme ... 565
10.2.3 CSS theme ... 566
10.3 Theme switching ... 568
10.4 Concept ... 570
10.4.1 Attribute levels ... 572
10.4.2 Element levels ... 572

11 Historical data ... 574
11.1 TcHmiSqliteHistorize extension ... 574
11.2 Trend Line Chart Control ... 576

12 Recipe management ... 581
12.1 Management in the engineering ... 582
12.1.1 Create folder ... 582
12.1.2 Create new recipe type ... 583
12.1.3 Changing the recipe type ... 583
12.1.4 Recipe Type Editor ... 584
12.1.5 Deleting a recipe type ... 589
12.1.6 Creating a new recipe ... 590
12.1.7 Changing the recipe ... 591
12.1.8 Recipe Editor ... 591

12.1.9 Delete recipe ... 593
12.2 Management in the client ... 593
12.2.1 Example ... 594

13 Protocols ... 600
13.1 ADS ... 600
13.1.1 Access by IndexGroup and Offset ... 601
13.2 OPC UA Client ... 601
13.2.1 Add extension ... 602
13.2.2 Configure connection ... 603
13.2.3 Use data points ... 604

14 Server ... 606
14.1 Server-to-server communication ... 606
14.1.1 Configure connection ... 606
14.1.2 Use data points ... 609
14.2 Extensions ... 610
14.2.1 First steps ... 610
14.3 API Reference ... 613
14.3.1 TcHmiSrv.DynamicSymbols Namespace ... 613
14.3.2 TcHmiSrv.Management Namespace ... 694
14.3.3 TcHmiSrv Namespace ... 701
14.4 Appendix ... 1109
14.4.1 TcHmiAds return codes ... 1109

15 Framework ... 1111
15.1 API reference ... 1111
15.1.1 TcHmi ... 1112
15.1.2 tchmi_base64decode ... 2104
15.1.3 tchmi_base64encode ... 2104
15.1.4 tchmi_clone_object ... 2105
15.1.5 tchmi_equal ... 2106
15.1.6 tchmi_path ... 2106
15.1.7 tchmi_create_guid ... 2107
15.1.8 any ... 2107
15.1.9 Array ... 2107
15.1.10 boolean ... 2107
15.1.11 JQuery ... 2108
15.1.12 Date ... 2108
15.1.13 number ... 2108
15.1.14 null ... 2108
15.1.15 object ... 2108
15.1.16 string ... 2108
15.1.17 undefined ... 2108
15.1.18 TcHmiCharting.LineGraph.Point ... 2109
15.1.19 TcHmiCharting.IAxis.Position ... 2109
15.2 Scripting languages ... 2109
15.2.1 JavaScript ... 2110

15.3 Controls ... 2123
15.3.1 Properties ... 2123
15.3.2 Introduction ... 2163
15.3.3 First steps ... 2164
15.3.4 Development environment ... 2167

16 Samples ... 2169
16.1 Controls ... 2169
16.1.1 Barchart ... 2169
16.1.2 Combobox ... 2169
16.1.3 Datagrid ... 2169
16.1.4 Grid ... 2169
16.1.5 Event Grid ... 2169
16.1.6 Linechart ... 2169
16.1.7 Trend Line Chart ... 2169
16.2 Recipe management ... 2169

17 FAQ ... 2170
17.1 General ... 2170
17.2 Server ... 2170
17.3 Client ... 2172
17.4 Communication ... 2172
17.5 Framework ... 2173

18 Appendix ... 2175
18.1 Schema ... 2175
18.1.1 General ... 2175
18.1.2 framework ... 2192
18.1.3 server ... 2268
18.2 Return Codes ... 2282
18.2.1 ADS Return Codes ... 2282
18.3 Troubleshooting ... 2285
18.3.1 Windows error reporting ... 2286
18.3.2 Client Console ... 2286
18.3.3 Engineering server ... 2286

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.
The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.
Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

Tip or pointer

This symbol indicates information that contributes to better understanding.

2 Overview

2.1 Product description

TwinCAT HMI is a tool for creating HMIs (Human Machine Interfaces). Like TwinCAT, it is integrated directly into Visual Studio. A graphical WYSIWYG editor is used for the configuration, i.e. no programming is required.

Controls from the toolbox can be arranged on the user interface and linked with real-time variables, e.g. from the PLC or from a C++ module. User Controls can be used to graphically create and configure specific controls. JavaScript developers can develop their own controls. During development, the functionality can be tested with live data from the real-time system. Modifications are also possible in this live mode.

The logic of the HMI can either be implemented on the client side in JavaScript or as a server extension. A server extension allows you to develop extensions in C#.

Because the HMI is web-based and responsive - based on HTML5 and JavaScript - the resolution and orientation adapt automatically. This allows the same page to be used for different display sizes, aspect ratios and orientations.

2.2 Scenarios

The HMI can be integrated into a wide range of scenarios.

In a network, different controllers can exchange their data with one or more clients. The clients can run on local PCs or Panel PCs, or on mobile devices such as tablets or smartphones.

Local-Client

Multi-client

An HMI can be configured for different resolutions and can be called up by several devices simultaneously.

Multi-runtime

The TwinCAT HMI can connect several real-time systems. TwinCAT runtimes can be programmed in PLC languages, C/C++ or MATLAB®/Simulink®.

Multi-protocol

The TwinCAT HMI server can communicate with all TwinCAT runtimes via the ADS extension. Third-party controllers can be connected via the OPC UA extension. Additional protocols can also be integrated via the Server Extensions SDK.

Multi-server

The TwinCAT HMI server can communicate with other TwinCAT HMI servers. This enables so-called server-to-server communication. Server-to-server communication offers the possibility of exchanging any data points among the servers. This enables, for example, higher-level systems to be created.

2.3 Architecture

2.3.1 Engineering

The TwinCAT HMI integrates into the Visual Studio® development environment from Microsoft. TwinCAT3 provides the free Visual Shell, which can also be used with the TwinCAT HMI. This allows you to develop the PLC and the HMI in one tool. TwinCAT HMI Creator is an editor for graphical development of the HMI. The TwinCAT HMI framework comes with predefined controls. Communication with the TwinCAT HMI Server takes place via secure network connections. The TwinCAT HMI Server has a modular structure. HMI Server Extensions provide additional functions. Furthermore, you can develop your own customer extensions to execute non-real-time-relevant business logics centrally in the server. The TwinCAT ADS protocol (Automation Device Specification) is supported, which enables communication with all TwinCAT devices.

2.3.2 Client/Server

The TwinCAT HMI is essentially based on a client-server architecture. No additional software installation is required for the TwinCAT HMI Client. It is an HTML5-capable web browser that does not require any additional extensions. The browser connects to the TwinCAT HMI server, which provides the HMI in a virtual directory (WWW root). The HMI is passed to the client via HTTP(S) protocol. The HMI project consists of content such as pages and images and the TwinCAT HMI framework, which is based on JavaScript and dynamizes the web-based HMI. Among other things, the framework provides the controls and establishes a secure web socket connection (WS(S)) to the server in order to be able to communicate with it.

The TwinCAT HMI server has a modular structure. Additional functionalities can be provided via server extensions, such as the messaging system or other protocols. The server supports the TwinCAT ADS protocol and can therefore communicate with all TwinCAT devices. Third-party systems can be connected via OPC UA extensions.

2.4 Business model

The business model of the TwinCAT HMI is based on that of TwinCAT 3 XAE. All products can be unlocked via a 7-day trial license, which can be renewed at any time. Only the TwinCAT HMI server is licensed. No data points are counted. Only the simultaneous client connections (e.g. browser) and the number of active targets (controllers) are taken into account. The TwinCAT HMI Server includes a client and a target connection as standard. Additional TF20x0 packages are available for further connections. Further TF2x00 server extensions can also be licensed optionally.

3 Installation

3.1 System requirements

TE2000 TC3 HMI Engineering:

Technical data	Version 1.10
Min. TwinCAT version	3.1.4022.0
Min. TwinCAT level	TC1000 TC3 | ADS
Supported Visual Studio versions	2013 Version >= 12.0.40629 with Update 5* (Shell, Community, Professional) 2015 Version >= 14.0.25431 with Update 3* (Community, Professional) 2017 Version >= 15.0 (Community, Professional) with ASP.NET** TwinCAT XAE Shell (TcXaeShell)
Operating system	Windows 7/8/10 (32-/64-bit)
.NET Framework version	>= 4.7.2
.NET standard version	>= 2.0

* Visual Studio updates can be found via the menu item "Extras/Extensions and Updates".

**ASP.NET can be added through the Visual Studio Setup process.

TwinCAT HMI Client:

Technical data	Version 1.10
Supported browser versions	>= Google Chrome 58 >= Mozilla Firefox 53 >= Microsoft Edge 40
Supported mobile browsers	>= Google Chrome for Android 58 >= Apple Safari for iOS 9

Framework Control and Server Extension development:

Technical data	Version 1.10
Supported Visual Studio versions	2013 Version >= 12.0.40629 with Update 5 (Community, Professional) 2015 Version >= 14.0.25431 with Update 3 (Community, Professional) 2017 Version >= 15.0 (Community, Professional) with ASP.NET

For the development of Framework Controls and Server Extensions a full version of Visual Studio is required. The Visual Studio Shell is not sufficient.

3.2 Version overview

Version 1.10

Version number	Name

1.10.1336.x	Version 1.10 Release 5
1.10.1171.x	Version 1.10 Release 4
1.10.1018.x	Version 1.10 Release 3
1.10.765.x	Version 1.10 Release 2
1.10.683.x	Version 1.10 Release 1

Version 1.8

Version number	Name
1.8.861.0	Version 1.8 Release 4
1.8.737.0	Version 1.8 Release 3
1.8.670.0	Version 1.8 Release 2
1.8.616.0	Version 1.8 Release 1

3.3 Installation

The TwinCAT 3 function for Windows-based operating systems is installed step-by-step.

1. Close all running Visual Studio instances so that the setup can integrate into the installed Visual Studio environments.
2. Double-click the downloaded file *TE2000-HMI-Engineering*.
   Start the installation under Windows with "Run As Admin" by right-clicking the setup files and selecting the corresponding option in the context menu.
3. Select the language to guide you through the installation.

4. Click on **Next** and then accept the end user agreement.

5. Enter your user data.

6. Start the installation and click **Install**.

7. If not all Visual Studio instances are closed, the setup will pause.
8. Please close all Visual Studio instances and click **Retry**.

9. Select **Finish** to exit the setup.

⇨ The installation is complete and the TE2000 HMI Engineering is integrated into the Visual Studio environment with its own project templates.

3.4 Licensing

If no TF2000 HMI server license is available on the Engineering system when a TwinCAT HMI project is first created, a 7-day test license can be generated. The suggested character string must be typed and confirmed with **OK**.

The trial license can be renewed at any time after it has expired.

Licensing a full version

1. Start TwinCAT XAE
2. Open an existing TwinCAT 3 project or create a new project
3. In the **Solution Explorer**, navigate to the entry **SYSTEM/License**

4. Open the **Manage Licenses** tab and add a **Runtime License** for your product (in this screenshot **TF2000: TC3 HMI Server**).

5. **Optional:** If you wish to add the license for a remote device, you must first connect to this device via the TwinCAT XAE toolbar

6. Open the **Order Information** tab
   The **System ID** and **HW Platform** fields cannot be changed. They describe the platform to be licensed.
   In general, a TwinCAT 3 license is linked to two parameters:
   The **System ID** identifies your device unambiguously.
   The **HW Platform** is an indicator for the performance of the device.

7. Optionally you can enter your own order number and a comment

8. On the **Order Information** tab, click the **Generate License Request File...** button to generate a license request file that is validated by a Beckhoff license server (if you do not know your **Beckhoff License ID**, contact your Beckhoff sales representative).
9. After you have saved the License Request File, the system asks whether the file should be sent to the Beckhoff license server by email:

10. If you confirm the dialog with **Yes**, your default email client opens and creates a new email for tclicense@beckhoff.com containing the license request file.
11. Send this activation request to Beckhoff

The License Response File is sent to the same email address that sent the License Request File.

12. Shortly afterwards, you will receive a license file from the Beckhoff license server.
    Import the file via the **Activate License Response File...** button to activate the product

13. Select the **License Response File** you received in your folder system

14. The License Response File is imported and all licenses contained in it are activated. All affected demo licenses are removed

15. Restart TwinCAT to activate the license

The license file is automatically copied to your local hard disk under ...\TwinCAT\3.1\Target\License.

4 Quick start

4.1 HMI Engineering interface

1 Solution Explorer

The Solution Explorer displays the HMI project tree. It is possible to add and manage various project types in one Solution. For example, it is possible to add a TwinCAT 3 project to the same Solution as a TwinCAT HMI project.

2 Toolbox

The Toolbox contains all the available visualization elements, known as controls. These can be added to the graphic editor of the designer □ by drag-and-drop. A predefined set of controls is supplied with the TwinCAT HMI.

3 Properties

The Properties window displays various properties depending on the object that is selected. The selected object could be a visualization control in the graphic editor of the Designer □ or the Document Outline respectively, or it could be an object in the Solution Explorer □.

If you have selected a visualization control there are three different views between which you can switch using the following buttons:

The left button (wrench) displays the general view with the various properties of this control. The central button (lightning bolt) displays all available events and the right button (lock) is for viewing the access rights to this control.

4 Document Outline (Standard VS tool window)

The Document Outline displays an overview of all the controls that have been added to the Designer □. As in the graphic editor, individual elements can be selected directly in the Document Outline, for example in order to change their properties in the Properties window. Controls can also be moved in the Document Outline, for example into or out of a container.

5 Designer

The Designer is the front-end development tool and is divided into two editors – a graphic editor in which controls can be added from the Toolbox □ by drag-and-drop and then changed in the Properties window □, and a code editor. The code is generated automatically in the background and can be directly and synchronously edited. In the top right-hand corner there are two buttons □ for switching between these two editors. The three buttons in the bottom right-hand corner □ are used to display both editors at the same time and to select the arrangement of the windows.

There are various quick-access buttons □ on the right-hand side. With these buttons you can open the Document Outline □, the TwinCAT HMI Configuration □, the TwinCAT HMI Gallery Explorer □ and the TwinCAT HMI LiveView □.

6 TwinCAT HMI Configuration

The configuration is the central TwinCAT HMI tool window, where you can access all available types of symbols (e.g. server symbols, internal symbols and localizations) and the User Management settings.

7 TwinCAT HMI Gallery Explorer

The Gallery Explorer contains predefined graphics that can be added directly to the graphic editor or to the Solution tree by drag-and drop.

8 TwinCAT HMI LiveView

In the LiveView the visualization can be tested directly online without having to download the changes first. It can be started as an additional tool window inside Visual Studio (left-hand tab) or in a web browser by URL link (right-hand tab).

4.2 Adding controls

1. Open the Designer.
2. Switch to the Toolbox.

3. Drag and drop a control into the graphic editor.

4.3 Adding events

All available events can be found in the Event view of the Properties window □, divided into four categories.

- All user-specific events (e.g. change of a symbol) are listed in the "Custom" category.
- All events generated by the system are listed in the "Framework" category.
- All events that interact with the operator are listed in the "Operator" category.
- All events that only this control has at its disposal are listed in the "Control" category.

You can add a logical operation to each event that will be executed automatically if the event occurs. To add a logical operation, select one of the events and click on the pencil button:

The "Actions and Conditions [▶ 50]" dialog opens, a graphic editor for implementing your logical operation. You can also double-click on the control to implement a standard event. In the case of a button, for example, the standard event would be "is pressed".

The editor consists of a left-hand and a right-hand part. In the left-hand part, all predefined actions are available in a tree view. The right-hand part is a graphic editor into which the actions and functions can be dragged and dropped.

There are four different categories in the Toolbox:

- General: General actions and conditions that are not directly assigned to a control
- Controls: All controls that are available within the scope. All properties are listed underneath each control.
- Controls (Out of scope): These controls are not in the current scope.
- Functions: Various functions that can also be used within the editor

In case a special logical operation is to be used several times you can build your own templates, which can then be found later in the Toolbox under the category "Action templates [▶ 59]". You can manipulate the Action Templates with the menu bar of the Toolbox, e.g. add, edit, delete, copy and paste. The Action Templates dialog has the same basic functions as the "Actions and Conditions" editor, plus a name box and an area for editing the template parameters.

4.4 Adding an internal variable

1. Open the configuration window [▶ 65].
2. Switch to the "Internal Symbols [▶ 73]" tab.
3. Click on the **New Symbol** button , double-click on **create new internal symbol** or drag **create new internal symbol** onto the desired destination.
4. Enter the data type, a symbol name and a default value.

4.5 Creating a data binding

There are three different ways to create a data binding, i.e. mapping a symbol to an entry in the Properties.

Create a data binding via "Select Binding Source"

1. Select the control whose property you wish to map.
2. Open the Properties and select the correct entry.

3. The first way is to use the dialog **Select Binding Source**. To do this, click on the button to the right of the property of the entry.

4. then select **Create data binding**

⇨ The window **Select Binding Source** opens.

In the dialog **Select Binding Source** you will find all types of available symbols on the various tabs:

Server symbols: all symbols that are available on the server under the ADS entry, e.g. the PLC variables.

Internal symbols: all internal symbols that have been created in the Configuration [▶ 67] (see: Adding an internal variable [▶ 31])

Localizations: Localization symbols, i.e. the localized text entries that you have created in the Localization editor [▶ 73].

Mapped symbols: Server symbols that are already mapped and are available in the engineering (e.g. Data Binding).

Controls: Each control has different properties whose values can be set or read.

Creating a data binding by drag-and-drop.

You can drag and drop a symbol from the Configuration □ into the Properties input box:

Creating a data binding for already mapped symbols

If the symbol is already mapped, you can enter the name of the symbol directly in the Properties field. In this case an autocomplete function is available.

4.6 Navigation concept

The desktop view is the start page for the application on which you can create your visualization. A good method of increasing the tidiness is to split the visualization into various parts. To do this, divide it into various content objects. These content objects can be displayed in different areas of the desktop view. For example, the visualization could be divided into three areas in the desktop view: a status bar □, a content area □ and a navigation bar area □. Various content objects can be displayed in the content area during the runtime. The contents of the status bar area and the contents of the navigation bar area can also be changed.

4.6.1 Creating navigation

1. Switch to the Solution Explorer.

2. Add the desired number of content objects:

2.1 Right-click on the TwinCAT HMI project entry or on one of the folders.

2.2 Select **Add** and **New Item** and select **Content**.

3. Open the view or the content where you wish to add the navigation.

4. Add a region and buttons in order to switch between the contents.

5. Set the **Target Content** attribute in order to define the start content.

6. Add a **SetTargetContent** action to the **.onPressed** event of all buttons and set the corresponding content.

4.7 Change the language

Localized texts are defined in the TwinCAT HMI by key/value pairs. Several translations can be added for each key. Each language can be edited in the Localization editor.

4.7.1 Adding a new language

To add a new language, create a new localization object.

1. Right-click on the Localization folder.

2. Select **Add** / **New Item**

3. Select **Localization** and click on **Add**

⇨ The TwinCAT HMI Wizard starts

4. Select a language and click on **Create**

⇨ The Localization editor starts

4.7.2 Manipulating translations in the Localization editor

The Localization editor has a column for the key and another for the translation. You can display all available languages in the same editor. To do this, right-click on the column header and select one of the languages in the drop-down menu.

In the upper area of the editor there is a toolbar that offers the following options:

-

Export or import data to or from an Excel file
- Delete entry
-

Drop-down menu default language
- Search or delete search

You can create a new entry in the red and gray shaded row with a double click.

You can switch between different languages with the LoadLocal action in the dialog "Actions and Conditions [▶ 50]". For example, you could use two buttons to switch between two languages. The ".onPressed" event of the buttons would be used to call the SetLocal action and to change the corresponding language.

4.8 User Control

It is possible in a User Control to swap out part of the visualization to a new control in order to be able to reuse it more easily. This control can be designed as a general module with the help of transfer parameters.

4.8.1 Creating a User Control

A User Control is an object type of its own.

1. Right-click on the TwinCAT HMI project or on a folder

2. Select **Add / New Item** and **UserControl**.

3. The User Control editor appears. It is very similar to the Designer, except that there is additionally a parameter editor (P button) on the right-hand side for quick access.

4. Drag a control out of the toolbox into the User Control editor
5. Click on **P** to open the Parameter editor.
6. Enter the name of the parameter in the dialog **Edit/Define Parameters**. The "Display Name" is generated automatically after the parameter name has been entered.
7. In the **Data type** column, select the data type from a drop-down menu. If the data type is a structure, select the corresponding structure type in the **Options** column.

8. In the **Bindable** column, define whether the parameter may be mapped to a symbol later on.

⇨ You can now use the newly created parameters inside the User Control with their mapping properties (see Creating a data binding [▶ 31]).

As a simple example, the illustration below shows the properties of an added User Control whose size is 100 x 130 pixels.

Two text blocks have been added to this User Control, each bound to a parameter of the type String (see □).

The User Controls that were added to the project are now available via the Toolbox; you can add them to the Designer by drag-and-drop. The parameters have been added to the properties of the User Control and are displayed in the Properties window □. If they were created as bindable it is also possible to map them to a symbol.

Document Outline
ViewDesktop
ViewDesktopBeckhoff
MyUserControl_1
Follow control selection
MyUserControl.usercontrol
Desktop.view*
BECKHOFF
Text1
Text2
100 %
Toolbox
Search Toolbox
Checkbox
Combobox
Datagrid
Ellipse
Frame
Image
Line
Linear Gauge
Line Chart
Polygon
Radial Gauge
Recipe Brow...
Rectangle
Textblock
Textbox
Toggle Butt...
Video
HMI System
Pointer
Container
Grid
HTML Host
Region
HMI UserControls
Pointer
MyUserCont...
General
Properties
Identifier
MyUserControl_1
Type
UserControlHost
Height
Left
Top
Right
Bottom
Maximum Width
Maximum Height
Minimum Width
Minimum Height
Z-Index
Opacity
Visibility
Transform
(Collection)
Grid Column Index
Grid Row Index
Border
Width
Style
Radius
Background Image
Image
Padding
Width
Height
Horizontal Alignment
Vertical Alignment
Parameter
Text1
Text2
TwinCAT HMI Configuration
Search:
Server Symbols
All Symbols
Mapped Symbols
Internal Symbols
Localizations
Controls
Files
All files
Content files
Views
User Controls
Users and Groups
TwinCAT HMI Configuration
TwinCAT HMI Gallery Explorer
Error List
Output
Find Results 1

5 Development environment

The HMI engineering environment consists of the following windows:

- Solution Explorer [▶ 42]
- Document Outline [▶ 62]
- Designer [▶ 46]
- HMI LiveView [▶ 62]
- Toolbox [▶ 132]
- Properties [▶ 49]
- HMI Configuration [▶ 65]
- Gallery Explorer [▶ 127]
- Server Configuration [▶ 117]

5.1 Solution Explorer

The Solution Explorer manages all available projects in a Solution. After creating a new TwinCAT HMI project the following project structure is constructed:

1. **TcHmiProject:** The project node offers access to the Project Settings [▶ 43].
2. **Server:** The server node contains the Server Settings [▶ 117] and lists the loaded server extensions.
3. **Framework:** The framework node enables the referencing of Framework Controls [▶ 2123].
4. **Properties:** Lists the internal configuration files of the project.
   - **Default.tpl:** Template file used for the generation of the HTML code.
   - **tchmi.framework.Schema.tpl.json:** Contains the type information for the framework.
   - **tchmi.project.Schema.tpl.json:** Contains the type information for the project**.**
   - **tchmiconfig.tpl.json:** Contains the configuration of the project.
   - **tchmimanifest.json:** Manifest file of the project.
   - **tchmipublish.config.json:** Contains the publish profile [▶ 110] of the project.
5. **References:** Offers the option to reference TwinCAT HMI packages [▶ 118].
6. **Imports:** Folder for contents such as images or videos that are added dynamically (e.g. by drag-and-drop via the Gallery Explorer).
7. **Themes:** The folder contains all themes [▶ 559] that have been added to the project.
8. **Fonts:** This folder contains fonts that apply irrespective of the theme.
9. **Images:** Folder for images.
10. **Localization:** This folder contains the language files required for the project.
11. **Desktop.view:** The start page that is displayed when starting the HMI.
12. **FavIcon.ico:** The icon displayed in the title bar of the browser.

5.1.1 Project properties

Following the selection of the HMI project node the properties of the HMI project are displayed in the Properties window.

Explanations:

- **Creation:** Date and time when the setup of the TwinCAT HMI was created.
- **Product Version:** Version no. of the TwinCAT HMI.
- **Device IP:** Local IP address.
- **TcHmiSrv's Port:** Port number of the server on which a client can reach the server via HTTP(S).

- **WebSocket Router's Port:** WebSocket port number required for the WebSocket connection between an HMI client and the server instance.
- **Default Viewport Height:** Default value for the height of a Viewport when a new HMI page is added. Various resolutions of the HMI can be simulated via the Viewport.
- **Default Viewport Width:** Default value for the width of a Viewport when a new HMI page is added. Various resolutions of the HMI can be simulated via the Viewport.
- **Interval:** Global update rate [ms] for refreshing the HMI server symbols.
- **Subscription Mode:** Update mode between server and client.
  - Change: The client is only informed of changes to the symbol values.
  - Poll: The client receives the symbol value cyclically, even if no change has taken place.
- **Theme Name:** Name of the active theme (in engineering and of the initially active theme in the client).
- **Timeout:** WebSocket timeout for communication between server and client.
- **Title:** Title of the HMI, which is displayed in the browser (tab).
- **Verbose Level:**
  - 0 = no messages
  - 1 = error messages only
  - 2 = error messages and warnings
  - 3 = error messages, warnings and infos
  - 4 = error messages, warnings, infos and debug messages
- **Version:** Version number of this project.
- **Company Name:** Optional name of the company providing or using the HMI.
- **Company's Website:** URL of the company providing or using the HMI.
- **Login Page:** Login page of the HMI, which is displayed when authentication is active.
- **Scale Mode:** Scaling mode of the HMI. This setting only applies if the Viewport is configured with absolute values (pixel specifications).
  - ScaleToFit: The width or height of the HMI is scaled to the current size of the browser window without a scrollbar appearing.
  - ScaleToWidth: The height is scaled accordingly to the width (while maintaining the aspect ratio with scrollbar).
  - ScaleToHeight: The width is scaled accordingly to the height (while maintaining the aspect ratio with scrollbar).
  - ScaleToFill: The height and width are stretched to the maximum width and height of the browser window (the aspect ratio is not maintained).
- **Startup View:** The start page (View) of the HMI, which is loaded when opening the HMI in the browser.
- **Framework Binaries:** Local directory from which the components of the framework are provided.
- **Server Binaries:** Local directory from which the components of the server are provided.
- **LOG_DESIGNER_MODE_COM_MESSAGES:** Defines whether the communication between Visual Studio and the Designer is to be displayed.
- **LOG_TCHMISERVER_MESSAGES:** Defines whether the communication of the TcHmi server is to be displayed.
- **Project File:** The name of the project file.
- **Project Folder:** Absolute storage location of the project.
- **Default Language:** Default language that is loaded for the HMI project. This setting can be overwritten by the settings per user or switched in the client.
- **Time Format Locale:** Time format of the currently set time zone that is loaded for the HMI project. This setting can be overwritten by the setting per user.
- **Time Zone:** Time zone that is loaded for the HMI project. This setting can be overwritten by the setting per user.
- **Viewport (meta attribute):** Meta attributes that define the scaling behavior and adaptability of the HMI.

5.2 Designer

The Designer is the front-end development tool of the HMI. It is divided into two editors – a graphic editor and a code editor. The code is generated automatically behind the graphic editor and can be directly edited.

In the top right-hand corner there are three buttons for switching between these two editors or - if both are displayed - swapping the views.

The three buttons in the bottom right-hand corner are used to display both editors at the same time and to select the arrangement of the windows (horizontal or vertical).

There are various quick-access buttons on the right-hand side. With these buttons you can open the Document Outline window, the TwinCAT HMI Configuration window, the TwinCAT HMI Gallery Explorer G and the TwinCAT HMI LiveView L.

5.2.1 Graphic editor:

You can design your interface in the graphic editor of the Designer. To do this, drag controls from the Toolbox and drop them into the editor. The automatically assigned names and sizes of the controls are displayed. You can set all properties in the Properties window.

If controls overlap in the editor, you can define which should be in the foreground and which in the background with a right mouse click. (Note: this changes the attribute Z-Index [▶ 454])

Control selection

A control can be selected by mouse click. This control is additionally selected if the Ctrl key on the keyboard is pressed simultaneously.

It is also possible to mark several controls with a selection frame. You can draw this frame by left-clicking next to all the controls and then moving the mouse. If the Ctrl key is pressed, the frame is also drawn within a container, otherwise the container would be moved.

All controls that *touch* the rectangular frame are selected after releasing the mouse button. If the Alt key is pressed, then only those controls that are *completely enclosed* are selected. Here too, it is possible to select additional controls using the Ctrl key.

Positioning aid

With the control trap you can align controls alongside others and lock them in position.

This function can be deactivated with the **Alt key**.

If you hold the **Shift key** down when moving a control, it will only be moved horizontally or vertically.

If you hold the **Shift key** down when enlarging a control at the corner, the aspect ratio will be kept constant.

If you press the **Ctrl key** you can copy one or more controls – even into another container.

5.2.2 HTML editor

In the HTML editor you can edit directly and check the code generated. Designer and HTML editor work synchronously.

5.3 Properties

The Properties window displays various properties depending on the object that is selected. The selected object can be a control in the graphic editor or an object in the Solution Explorer.

The name and symbol of the control are displayed in the **Identifier** field. If you have selected a control there are three different views between which you can switch with the following buttons:

The left-hand button (wrench) displays the general view with the various properties of this control:

The central button (lightning bolt) displays all available events:

The right-hand button (lock) displays the access rights [▶ 520] to this control:

5.4 Actions and Conditions Editor

The Actions and Conditions Editor allows you to configure control events. To open the Actions and Conditions Editor, select a control and click on the events in the Properties window.

Clicking the Edit icon (pen) opens the Actions and Conditions Editor for the event associated with the selected control.

The actions are listed in the left part of the editor. The right part is a graphic editor into which the actions can be moved by dragging and dropping.

The toolbox is divided into different categories:

- General [▶ 51]: General actions and conditions that are not directly assigned to a control.
- Selected Control: Properties of the selected control that can be changed during runtime.
- Controls [▶ 57] (in scope): All available controls on the same HMI side as the control. The properties of the control that can be changed during runtime are listed below the control.
- Controls (Out of scope): The controls that are not on the same HMI side as the control.
- Functions [▶ 58]: Various functions that can also be used within the editor.
- Action Templates [▶ 59]: Defined templates for actions that can be reused.

You can perform the following actions in the Actions and Conditions Editor:

Add an Action Template

Edit an Action Template

Delete an Action Template

Copy an Action Template

Expand/collapse folder

5.4.1 General

The general actions can be used independently of a control.

5.4.1.1 Condition

A condition is an If statement within the Actions and Conditions Editor.

Explanations:

1 Condition: Condition of the If statement. A Binding [▶ 92] to a symbol can be created.

2 Comparison operator: Comparison operator of the If statement. The selection is made via the combo box (`==`, `!=`, `>=`, `>`, `<=`, `<`).

3 Comparison value: Comparison value of the If statement. A Binding [▶ 92] to a symbol can be created.

4 Add a condition: Click the button to add another condition to the If statement.

5 Logical operator: Logical operator between the conditions. The selection is made via a combo box (`AND`, `OR`).

6 Remove condition: Click the button to remove a previously added condition.

7 Then branch: Then branch of the If statement that is executed if the condition is met.

8 Or branch: Or branch of the If statement that is executed if the condition is not met.

5.4.1.2 SwitchCase

A SwitchCase statement refers to a case distinction within the Actions and Conditions Editor.

Explanations:

1 Condition: Condition of the case distinction. A Binding [▶ 92] to a symbol can be created.

2 Add case: Click the button to add another case to the case distinction.

3 Case: Individual case that is compared with the condition. A Binding [▶ 92] to a symbol can be created.

4 Then branch: This branch is executed if the case meets the condition.

5 Remove case: Click the button to remove a previously added case.

6 Default branch: This branch is executed if no case meets the condition.

5.4.1.3 WriteToSymbol

The WriteToSymbol action writes any value to a symbol (server symbol [▶ 67], internal symbol [▶ 73]) or to a control [▶ 132] property.

Explanations:

1 Symbol: Symbol or control property to be written to. A binding [▶ 92] is required.

2 Value: Value to be written to the symbol. A binding to another symbol or a control property can be created.

5.4.1.4 JavaScript

A JavaScript action consists of JavaScript code and enables the execution of simple JavaScript functions.

5.4.1.5 CreateBinding

The CreateBinding action creates a binding [▶ 92] to a control property at runtime.

Explanations:

1 Target: The control property for which the binding is to be created.

2 Expression: Expression (symbol [▶ 67], control property [▶ 132] or localized text [▶ 73]) to be bound to the target.

> As of version 1.10.1336.0, this functionality is taken over by the createBinding [▶ 98] function.

5.4.1.6 RemoveBinding

The RemoveBinding action removes a binding from a control property at runtime.

Explanations:

1 Target: Control property from which the binding [▶ 92] is to be removed.

> From version 1.10.1336.0, this functionality is taken over by the removeBinding [▶ 98] function.

5.4.1.7 Comment

The comment action inserts a comment in the Actions and Conditions Editor. The comment is used for information or as a structuring aid with more complex logics and is not interpreted during runtime.

5.4.1.8 Feedback from actions

All actions in the Actions and Conditions Editor can provide feedback on the success of the action. A "Success" and an "Error" branch can be displayed for each action by right-clicking on it.

Explanations:

- The "Success" branch is called if the action was executed without errors. For JavaScript code containing a context object [▶ 2112], the context object must return "Success".
- The "Error" branch is called if an exception occurs during execution of the action. For JavaScript code that contains a context object, the "Error" branch is called if an exception or an "Error" message of the context object occurs.

- With an asynchronous logic, the function "Wait for completion [▶ 2112]" waits for the action to be executed. With a synchronous logic, this function has no effect.

5.4.2 Controls

The Actions and Conditions Editor allows you to change all control properties at runtime (similar to the static configuration in the Properties window [▶ 49]). In addition to the properties of the selected control, the properties of all controls instantiated within all HMI pages can be changed.

The control properties of the instantiated controls are listed in the left column of the Actions and Conditions Editor. Drag & drop can be used to add the properties to the right part of the editor. Drag & drop automatically adds the control property to a WriteToSymbol action.

The following example shows setting the maximum value of a linear gauge [▶ 306] by pressing a button (onPressed event [▶ 458]).

5.4.3 Functions

Various functions are available graphically in the Actions and Conditions Editor. In addition, user-specific functions [▶ 2113] are listed there according to their category. The functions can also be used in the Properties window [▶ 49].

The functions are listed in the left column of the Actions and Conditions Editor. Drag & drop can be used to add the functions to the right part of the editor. Not all functions are useful in all contexts.

5.4.4 Action Templates

Action Templates allow you to reuse multiple actions that are often used together in the project. Reuse is made possible by passing on so-called template parameters. Action Templates are used like an action in the Actions and Conditions Editor. The template parameters are also transferred to the Action Template there.

All the functions of the Actions and Conditions Editor are available within the Action Template.

5.4.4.1 Editor

The Action Template Editor can be opened via the Actions and Conditions Editor or the Configuration window [▶ 65]. The editor allows you to create and edit Action Templates. The editor is structured like the Actions and Conditions Editor and contains the following extensions:

Explanations:

1 Explorer: The Explorer offers the same categories as the Actions and Conditions Editor.

2 Template name: Name of the Actions Template displayed under Action Templates in the Actions and Conditions Editor.

3 Template description: Description of the Action Template that is displayed during mouseover over the Action Template in the Actions and Conditions Editor.

4 Action editor: Graphic editor that records the Actions and Conditions using drag & drop. The functionality is the same as the Actions and Conditions Editor.

5 Parameter name: Name of a Template Parameter that is passed to the Action Template.

6 Parameter data type: Data type of the Template Parameter.

7 Parameter description: Description of the Template Parameter that is displayed during mouseover over the parameter in the Actions and Conditions Editor.

8 Parameter buttons:

- **New**: Adds a new Template Parameter.

- **Edit**: Edits an existing Template Parameter.
- **Remove**: Removes an existing Template Parameter.
- **Up**: Moves an existing Template Parameter up one position.
- **Down**: Moves an existing Template Parameter down one position.

5.4.4.2 Example

The following description shows the creation of an Action Template as an example. The example describes an Action Template for switching any Boolean value.

1. In the Action and Conditions Editor, click "Add Action Template". The Action Template Editor [▶ 59] opens.
2. Enter a name and a description for the Action Template.

3. Add a Template Parameter of type "Boolean" to the Action Template.

4. Drag a condition [▶ 52] into the editor area and implement the logic for switching the Boolean value using WriteToSymbol actions [▶ 53]. You can insert the Template Parameter into the editor area using drag & drop.

5. The Action Template can now be used as a normal action. In the Actions and Conditions Editor, drag the Action Template into the editor area and link an icon of type "Boolean" to the template parameter.

5.5 Document Outline

The Document Outline shows an overview of all controls that are located on the view or content page currently open in the Designer. Individual elements can be selected directly in the Document Outline, for example in order to change their properties in the Properties window.

- By clicking on you can hide individual controls so that they are no longer displayed in the Designer. The controls are still displayed in the LiveView or in the browser. The controls are made visible again by clicking on .
- You can lock controls to prevent them being shifted by clicking on . Clicking on unlocks the controls so that they can be shifted again.
- The lightning bolt icon indicates that events are configured for this control. A click on the icon opens the Properties window and displays the events.
- If controls are superimposed on one another in the Designer, thus making them inaccessible via the graphic editor, they can be selected directly via the Document Outline.
- It is also possible to influence the order of the controls by drag-and-drop. (See detail page Positioning of controls [▶ 134]). In addition, you can shift the controls into containers or grids or remove them from a container or grid.

5.6 LiveView

In the LiveView the current configuration of the HMI can be tested directly online with values from the PLC without the changes first having to be published on a server. The LiveView can be opened directly as a tool window in Visual Studio. The LiveView synchronizes itself directly with the Designer without you first having to save the changes. If, for example, you shift a control in the Designer, it will be shifted automatically in the LiveView.

The LiveView toolbar offers various options:

1. HMI project for which the LiveView is displayed. If there are several HMI projects in the Solution, you can switch between the various projects.
2. Refresh button: Refreshes the LiveView when changes have been made in the Designer that couldn't be automatically refreshed.
3. Developer Tools button: Opens the Developer Tools [▶ 2286] of the browser engine. These can be used for debugging purposes:

4. Full-screen mode: This button opens the LiveView in full-screen mode. The full-screen mode can be exited again by pressing the Esc key.
5. Zoom-Out: Zooms out of the LiveView.
6. Zoom In: Zooms into the LiveView.
7. Reset Zoom: Resets the zoom to the normal view.
8. QR code display: Displays a QR code for the LiveView in order to open the LiveView on another device (e.g. a Smartphone). So that the LiveView can be opened via an external device you have to set the binding to a network card in the Visual Studio settings. The settings can be found under: Menu bar/Tools/Options/TwinCAT/HMI Environment/Binding Device

9. Opens the LiveView in an available browser. If several browsers are installed on the operating system you can choose between the browsers.

5.7 TwinCAT HMI Configuration

The **Configuration** window is the central configuration window in the TwinCAT HMI. It offers an overview of all elements in the project and enables the configuration of these elements. Furthermore, the user management and all data types can be configured via the window.

The **Configuration** window contains a toolbar (1), a search bar (2), a project element explorer (3) and a display area (4).

Depending on the node selected, various buttons are available in the toolbar in the right-hand explorer:

Refresh: This button refreshes the Configuration window.

Create Internal Symbol: This button adds a new internal symbol to the project.

Create Localized Symbol: This button adds a new localized key to the language tables in the project.

Create Server Symbol: This button adds a new symbol to the server.

Map Symbol: This button maps a server symbol.

Unmap Symbol: This button removes a mapping from a symbol.

Add User: This button adds a new user to the user management.

Add Group: This button adds a new user group to the user management.

Collapse All: This button collapses all expanded folders.

Automap Symbols: This button activates the automatic mapping of symbols. If the button is activated the symbol is automatically mapped in the background. If the button is deactivated the Dialog Map symbol appears; a mapping for the symbol can be assigned in this dialog.

Show advanced Items: This button shows all configuration options and is intended for power users.

Toggle all On-/Offline: This button switches the state of all symbols between online and offline. No communication takes place with the server in the offline state.

Show live values for mapped symbols: This button shows live values from the PLC for mapped symbols in the Configuration window.

Edit permissions: This button opens the permissions management for the project elements.

The **Configuration** window can interact with the Properties window [▶ 49] by drag-and-drop.

5.7.1 Server symbols

Server Symbols lists all symbols that are saved in the TwinCAT HMI server.

5.7.1.1 All Symbols

All Symbols lists all symbols of the extensions available in the server. The user-specific server extensions are also listed here.

Map Symbol: This button maps a server symbol.

Show advanced Items: This button shows all configuration options and is intended for power users.

Show Advance Properties

If **Show Advanced Items** is activated, symbols and further extensions are available to power users.

Create Mapping

Server symbols can be mapped with a right-click on the symbol or by clicking on the **Map Symbol** button.

A name can be assigned for the mapping in the dialog **Map new Symbol**.

5.7.1.2 Mapped symbols

Mapped symbols lists all symbols that have been mapped in the server.

Create Server Symbol: This button adds a new symbol to the server.

Unmap Symbol: This button removes a mapping from a symbol.

Toggle all On-/Offline: This button switches the state of all symbols between online and offline. No communication takes place with the server in the offline state.

Show live values for mapped symbols: This button shows live values from the PLC for mapped symbols in the Configuration window.

Create Server Symbol

Server symbols can be created centrally in the server under **Mapped Symbols**. They can also be used offline without a PLC. Clicking on the button **Create Server Symbol** opens a dialog for adding server symbols.

1. **Data type:** Data type of the new symbol.
2. **Symbol Name:** Name of the new symbol.
3. **Default Value:** Default value of the symbol that is assigned when starting the server.

Server symbols that were created offline can be bound with a different symbol (e.g. from the PLC) later on. To do this, click on **Map to other symbol** in the context menu and select the symbol via the window **Select Binding Source**.

5.7.1.3 Historized Symbols

Historized Symbols lists all symbols that are historized by the server.

Create Historized Symbol

A historized symbol is created via the mapped symbols. To do this, click on **Historize Settings** in the context menu for a mapped symbol:

The configuration for the historization of the symbol takes place in the dialog **Historize Settings**:

1. **Recording Enabled**: This setting activates the historization of the symbol.
2. **Max Entries:** Maximum number of entries in the ring buffer of the historized symbol.
3. **Interval:** Interval at which the data records are written to the ring buffer.

A historized symbol is identified by the following icon after the symbol name:

5.7.2 Internal symbols

Internal symbols are independent of the server and are valid within a browser instance. The internal symbols thus enable variables to be saved individually for each client.

Create Internal Symbol: This button adds a new internal symbol to the project.

Create Internal Symbol

Clicking on the button **Create Internal Symbol** or double-clicking on **<create new internal symbol>** opens a dialog for adding internal symbols:

1. **Data type**: Data type of the internal symbol. The selection is made via the combobox.
2. **Symbol Name**: Name of the internal symbol.
3. **Default Value**: Default value of the internal symbol that is set when loading the client.
4. **Persist**: Defines whether the symbol is saved persistently in the client. A persistent internal symbol is available at the next start of the HMI.
5. **Read-only**: Defines whether the symbol may only be read or whether read and write access is allowed.

5.7.3 Localizations

Localizations list all available localization texts [▶ 557]. You can bind the entries with the control properties by drag-and-drop.

Create Localized Symbol: This button adds a new localized key to the language tables in the project.

Create Localized Symbol

The dialog **Create new localized symbol** is opened by clicking on the button **Create Localized Symbol**, double-clicking on **<create new localized symbol>** and dragging and dropping the entry **<create new localized symbol>** onto a control property.

1. **Key**: Key of the localized text. The key may only be assigned once.
2. **Language**: All available languages.
3. **Value**: Text of the key in the respective language.

5.7.4 Data types

Data types list all available data types of the TwinCAT HMI and the current project.

- **Server**: All server data types.
- **Project**: User-specific project data types that were created by the Scheme Generator [▶ 75].
- **Framework**: All framework data types (including data types from framework controls).
- **General**: Base data types.

5.7.4.1 Creating a data type

A new user-specific project data type can be created on the **Data types** tab. To do this, right-click inside the window and select **Create new project datatype…** from the context menu.

In the subsequent dialog you select the basis on which the new data type is to be created.

5.7.4.1.1 Array

Create an array-based type creates a new project data type based on an array. The subsequent dialog configures the array:

1. **Choose Base Type**: Base data type on which the array is based.
2. **Array Size**: Size of the array. A dynamic array is generated by entering no value.

Clicking on **Create Type** opens the Scheme Generator [▶ 81].

5.7.4.1.2 Object

Create a new object type creates a new project data type based on an object. Properties can be added to the object in the subsequent dialog:

1. **Name**: Name of the object property.
2. **Data type**: Data type of the property.

Clicking on **Create Type** opens the Scheme Generator [▶ 81].

5.7.4.1.3 Remote data type

Create a new datatype based on a remote type creates a new project data type on the basis of an existing data type on the server or a PLC. In the subsequent dialog the remote data type can be selected on whose basis the new data type is to be created:

For example, function blocks from a PLC can be selected in the dialog. The data type can be created on this basis.

Clicking on **Create Type** opens the Scheme Generator [▶ 81].

5.7.4.1.4 Expand data type

Create new datatype based on an existing one creates a new project data type based on any data type. This function is used to expand an existing data type. The subsequent dialog enables the selection of the data type via a combobox:

All data types including the project data types are available in the combobox.

Clicking on **Create Type** opens the Scheme Generator [▶ 81].

5.7.4.1.5 Schema Generator

The Schema Generator is called at the completion of every process of creating a data type. In addition, it is possible to start directly with the Schema Generator if **Start from scratch** is selected.

1. **Data Type Explorer**: The Data Type Explorer displays the structure of the data type in a tree view.
2. **Name**: Name of the data type.
3. **Title**: Title of the data type.
4. **Description**: Description of the data type.
5. **Base Type**: Base data type of the data type.
   - **Scheme Type**: JSON base data types.
   - **Existing Type**: Arbitrary TwinCAT HMI data type.
   - **Multiple**: Arbitrary data types with restrictions [▶ 83]: AnyOf, AllOf or OneOf (see JSON documentation).
   - **Not Set**: Do not set a data type. This necessitates a manual restriction [▶ 83] of the data type.
6. **Define Subtypes**: Enables the definition of subtypes within the schema. A subtype is a user-specific data type within the user-specific data type.
7. **Additional Properties**: Adds optional properties to the data type. Optional properties enable the addition of further properties that are not required, but are validated (see JSON documentation).
8. **Required Properties**: Defines all properties that are required. One property is entered per row. The data type is not valid if it is lacking a required property.
9. **Subschema Preview**: Provides a preview of the generated schema of the selected node in the Data Type Explorer.

5.7.4.1.5.1 Restricting a data type

If "Multiple" or "Not Set" is selected as the base data type, additional fields are shown in the Properties area of the Schema Editor. The additional fields enable the restriction of the data type.

1. **Array Limits**: Defines a minimum and maximum array length.
2. **Number Limits**: Defines minimum and maximum limits for a numerical value.
3. **String Limits**: Defines a minimum and maximum length for a string.
4. **String Format**: Defines the format of a string.
5. **Enum Options**: Defines an enumeration and its entries. One entry is made per row.

5.7.5 Parameter

Parameter lists all the transfer parameters of a User Control. The Parameter category is visible only if a User Control has the focus.

5.7.6 Controls

Controls lists all available controls (instanced controls) and offers the option of inserting the properties of the controls into other editors by drag-and-drop.

5.7.7 Global Events

Global Events lists all events of the TwinCAT HMI that can neither be assigned to any control nor to any HMI page.

Global Events:

- **onInitialized**: The event is triggered as soon as the system has been initialized and the main control (View) is loaded. Other controls may not be loaded or visible at this time.
- **onLocaleChanged**: The event is triggered as soon as a language has been loaded.
- **onThemeDataChanged**: The event is triggered as soon as all JSON and CSS files have been loaded following a change of theme. Furthermore the event is triggered as soon as a new font has been loaded.
- **onUserInGroupChanged**: The event is triggered if the group membership of the current user is now known to the system or has changed.

5.7.8 Functions

The functions list all functions available in the project. These include the functions provided by [▶ 97] TwinCAT HMI and the user-specific functions [▶ 2113].

5.7.9 Files

Files lists the files used in the project, categorized according to the project folders.

5.7.9.1 All Files

Lists all available files of the TwinCAT HMI project.

5.7.9.2 Content Files

Lists all content files (pages) of the project.

5.7.9.3 Views

Lists all views of the project. A view is the parent element of an HMI. The StartView can be set in the Project properties [▶ 43].

5.7.9.4 User Controls

Lists all User Controls in the project.

5.7.10 Users and user groups

Lists all available users and user groups. Explanations of the configuration can be found in User Management [▶ 521].

5.7.11 Client certificates

The client certificates list all client certificates that are stored in the project. The creation and use of the client certificates are described in the user management [▶ 540].

5.7.12 Action Templates

Action Templates lists all Action Templates [▶ 59] used in the project.

5.7.13 Recipe management

The recipe management [▶ 581] contains all recipes and recipe types that have been configured for the project. With a double click on a recipe or a recipe type the corresponding editors [▶ 584] of the recipe management are opened.

5.7.14 Permissions Management

The permissions management enables the configuration of the rights [▶ 521] for all project elements.

Edit permissions: This button opens the permissions management for the project elements.

Further information on rights can be found in the chapter Rights system [▶ 527].

5.8 Localization editor

You can open the Localization editor in the Solution Explorer by double-clicking on the respective language file.

The Localization editor enables the editing of language files.

1. Removes a selected entry (key and language entry) from the language table.

2. Export or import the Excel table or CSV.
3. Defines the default language in the HMI project.
4. Search for keys or language entries.
5. The key must be unique and is linked with a control in the Designer at a later date. The language entry of the currently active language is then automatically displayed.
6. The language entry contains the text for the key.

Adding a new language entry

To create a new localization entry, click in the last row of the table on **create new localized symbol**. A unique key must exist for each entry.

Further languages can be displayed via the context menu of the table.

It is not possible to edit the additionally displayed language in this table. To do that you have to open the respective table for the language.

In the Designer you can link the language key directly to the property of a control as a symbol. The language symbols are marked by an "L". When hovering with the mouse pointer over a language symbol, the corresponding language entries are displayed.

5.9 Symbols and Bindings

A binding links any symbol to a property [▶ 49] of a control. A binding automatically updates the property of the control when the value of the symbol changes.

5.9.1 Create binding

Under Properties [▶ 49] you can configure a binding by clicking the □ symbol.

The Select Binding Source dialog displays all available symbols, categorized by symbol type. Server symbols must be mapped before they are bound to a control property. The mapping is created by clicking the **Map Symbol** or automatically by clicking **OK**.

The mapped server symbols are available under **Mapped symbols** after the mapping has been created.

Besides internal variables and localization symbols, properties of other controls can also be bound (e.g. the current text of a text box [▶ 376]).

Alternatively, the symbols in input fields can be bound directly to a property of a control using auto-completion.

Available from version 1.10.

5.9.2 Create function binding

In addition to variable bindings, a function binding can be created. A function binding can be created for all control properties, if the JavaScript functions return a suitable value for the property of the control. A function binding can be used for conversions [▶ 2116], for example.

A function binding is created with the **Create function binding** command.

An active function binding is identified by **fx** highlighted in blue.

The transfer parameters of the function are transferred directly in the Properties window.

In addition, it is possible to implement custom JavaScript functions and create a function binding for them. Further information can be found here [▶ 2113].

5.9.3 Configure binding

A binding created for a property of a control can be configured via the Edit Binding dialog. The dialog can be opened by right-clicking or double-clicking on the bound symbol.

The Edit Binding dialog offers various configuration options, depending on the symbol type:

1 Binding mode: Binding mode determines whether the binding acts in one direction or in both directions.

- **One-way binding** (read-only): The value of the symbol is read.
- **Two-way binding** (read and write access): The value of the symbol is read and the symbol is written when a specific event is triggered.

2 Binding event: The binding event can only be optionally configured for TwoWay bindings. It determines at which event the symbol is written to (e.g. onValueChanged [▶ 327] for a linear gauge [▶ 306]). If no event is selected for a TwoWay binding, the value is automatically written back if the value changes.

- **Event**: An event of the current control for which the symbol is to be written to.

3 Interval: The interval specifies the time within which the binding is to be queried or updated. The interval can only be set for server symbols.

- **Time (ms):** Time in milliseconds.

4 Timeout: The timeout defines the time after which a read or write access to the symbol leads to an error if no response is sent from the server. The timeout can only be set for server symbols.

- **Time (ms)**: Time in milliseconds.

5 Subscription Mode: Update mode between server and client.

- **Use project default**: The global setting is used, which is defined in the project properties. [▶ 43]
- **Change:** The client is only informed of changes to the symbol value.
- **Poll:** The client receives the symbol value cyclically, even if no value change has taken place.

6 Parallel: This property defines whether a command is to be executed by the server in parallel or in a queue. The decision whether processing is to take place in parallel or in a queue depends on the respective application. By default, all commands are executed in a queue. This property can only be set for server symbols.

- **Parallel (disabled)**: All commands to the server are queued and processed one after the other. Execution may take a little longer if there are several commands in the queue.
- **Parallel (enabled)**: The commands are sent directly to the server without being queued. The commands are executed directly. With this option, it can happen that the execution order of commands sent in short succession is switched.

Available from version 1.10.

5.10 Functions

Functions enable various logical operations to be executed on the client side. The functions can be used in the Properties window [▶ 49], in the Actions and Conditions Editor [▶ 50] or in the Code Behind [▶ 2109].

In addition to the functions supplied, it is possible to implement your own logical operations in JavaScript [▶ 2113].

5.10.1 Binding

5.10.1.1 CheckBinding

The CheckBinding function checks whether a binding to a control property exists. Refer also to the description page for the corresponding API function [▶ 1877].

Parameter

Name	Type	Description
control	Control [▶ 2216]	The control containing the property to be checked.
propertyName	String [▶ 2108]	The name of the control property to be checked.

Return value

Type	Description
Boolean [▶ 2107]	true: A binding to the control property exists. false: A binding to the control property does not exist.

Note: Available from version 1.10.

5.10.1.2 CheckBindingEx

The CheckBindingEx function checks whether a binding of a certain symbol to a control property exists.

Parameter

Name	Type	Description
control	Control [▶ 2216]	The control containing the property to be checked.
propertyName	String [▶ 2108]	The name of the control property to be checked.
symbol	Symbol [▶ 2036]	The symbol for which the binding to a control property is to be checked.

Return value

Type	Description
Boolean [▶ 2107]	true: A binding of the symbol to the control property exists. false: A binding of the symbol to the control property does not exist.

Note: Available since version 1.10.

5.10.1.3 CreateBinding

The CreateBinding function creates a binding on a control property. Refer also to the description page for the corresponding API function [▶ 1875].

Parameter

Name	Type	Description
control	Control [▶ 2216]	The control that is to be linked.
propertyName	String [▶ 2108]	The name of the property on the control.
symbol	Symbol [▶ 2036] expression	Binding source

Return value

Type	Description
void	no return value

Nature and source of the danger

Available as function from version 1.10.1336.10

5.10.1.4 RemoveBinding

The RemoveBinding function deletes a binding on a control property. Refer also to the description page for the corresponding API function [▶ 1877].

Parameter

Name	Type	Description
control	Control [▶ 2216]	The control that is to be changed.
propertyName	String [▶ 2108]	The name of the property on the control.

Return value

Type	Description
void	no return value

Available as function from version 1.10.1336.10

5.10.2 Data Conversion

5.10.2.1 Concatenate

The Concatenate function links several strings together.

Parameter

Name	Type	Description
value […]	String [▶ 2108]	Any number of strings that are to be linked.

Return value

Type	Description
String [▶ 2108]	A string linked from the transfer parameters.

Note: Available since version 1.8.

5.10.2.2 ConvertBooleanToEnum

The function ConvertBooleanToEnum converts a Boolean value to the value of an enumerator.

Parameter

Name	Type	Description
value	Boolean [▶ 2107]	The Boolean value to be converted. Only a binding [▶ 92] makes sense at this point.
trueValue	String [▶ 2108]	The value to be returned if the Boolean value is **true**.
falseValue	String [▶ 2108]	The value to be returned if the Boolean value is **false**.

Return value

Type	Description
String [▶ 2108]	trueValue or falseValue depending on the Boolean value.

Note: Available since version 1.8.

5.10.2.3 ConvertBooleanToVisibility

The function ConvertBooleanToVisibility converts a Boolean value to a value of the enumerator "Visibility [▶ 453]".

Parameter

Name	Type	Description
value	Boolean [▶ 2107]	The Boolean value to be converted. Only a binding [▶ 92] makes sense at this point.
falseValue	Visibility [▶ 453]	The value to be returned if the Boolean value is **false**.

Return value

Type	Description
Visibility [▶ 453]	The visibility depending on the Boolean value.

Note: Available since version 1.8.

5.10.2.4 Decrement

The Decrement function counts a numerical value in a binding downward in steps of a specified step size.

Parameter

Name	Type	Description
value	Symbol [▶ 2036]	The value to be decremented. The value must be specified as a binding [▶ 92] to a symbol.
minValue	Number [▶ 2108]	A minimum value that may not be undershot.
stepWidth	Number [▶ 2108]	The step size with which the value is to be decremented.

Return value

Type	Description
void	No return value.

Note: Available since version 1.8.

5.10.2.5 Increment

The Increment function counts a numerical value in a binding upward in steps of a specified step size.

Parameter

Name	Type	Description
value	Symbol [▶ 2036]	The value to be incremented. The value must be specified as a binding [▶ 92] to a symbol.
maxValue	Number [▶ 2108]	A maximum value that may not be exceeded.
stepWidth	Number [▶ 2108]	The step size with which the value is to be incremented.

Return value

Type	Description
void	No return value.

Note: Available since version 1.8.

5.10.2.6 GetElementByIndex

The function GetElementByIndex returns the value of a certain array index of a binding.

Parameter

Name	Type	Description
arraySymbol	Symbol [▶ 2036]	A binding to the symbol of an array.
index	Number [▶ 2108]	The index of the array for which the value is to be returned.

Return value

Type	Description
Any [▶ 2107]	The value that is stored in the position of the index in the array.

Note: Available since version 1.8.

5.10.2.7 SetElementByIndex

The function SetElementByIndex sets the value of a certain array index in a binding.

Parameter

Name	Type	Description
arraySymbol	Symbol [▶ 2036]	A binding to the symbol of an array.
index	Number [▶ 2108]	The index of the array for which the value is to be set.
value	Any [▶ 2107]	The value to be written in the position of the array.

Return value

Type	Description
void	No return value.

Note: Available since version 1.10.

5.10.2.8 ToDateString

The function ToDateString converts a date in accordance with a transferred localization into a string containing the date.

Parameter

Name	Type	Description
date	Any [▶ 2107]	Text, numbers (milliseconds since 1/1/1970) or JavaScript date object.
locale	String [▶ 2108]	The localization variable in the form of a string. If this is not specified, the currently active language [▶ 557] will be used.

Return value

Type	Description
String [▶ 2108]	The date of the input value according to the localization in the form of a string.

Note: Available since version 1.10.

5.10.2.9 ToTimeString

The function ToTimeString converts a date in accordance with a transferred localization into a string containing the time.

Parameter

Name	Type	Description
date	Any [▶ 2107]	Text, numbers (milliseconds since 1/1/1970) or JavaScript date object.
locale	String [▶ 2108]	The localization variable in the form of a string. If this is not specified, the currently active language [▶ 557] will be used.

Return value

Type	Description
String [▶ 2108]	The time of the input value according to the localization in the form of a string.

Note: Available since version 1.10.

5.10.2.10 ToDateTimeString

The function ToDateTimeString converts a date in accordance with a transferred localization into a string containing the date including the time.

Parameter

Name	Type	Description
date	Any [▶ 2107]	Text, numbers (milliseconds since 1/1/1970) or JavaScript date object.
locale	String [▶ 2108]	The localization variable in the form of a string. If this is not specified, the currently active language [▶ 557] will be used.

Return value

Type	Description
String [▶ 2108]	The date with time of the input value according to the localization in the form of a string.

Note: Available since version 1.10.

5.10.2.11 ToLowerCase

The function ToLowerCase converts all the letters in a string to lower-case letters.

Parameter

Name	Type	Description
value	String [▶ 2108]	Any string.

Return value

Type	Description
String [▶ 2108]	Input string in lower-case letters.

Note: Available since version 1.8.

5.10.2.12 ToUpperCase

The function ToUpperCase converts all the letters in a string to upper-case letters.

Parameter

Name	Type	Description
value	String [▶ 2108]	Any string.

Return value

Type	Description
String [▶ 2108]	Input string in upper-case letters.

Note: Available since version 1.8.

5.10.2.13 TrimToLength

The function TrimToLength shortens a string to a specified number of characters from left or right.

Parameter

Name	Type	Description
value	String [▶ 2108]	String to be shortened.
trimToLength	Number [▶ 2108]	Number of characters to which the string is to be shortened. Positive values shorten the string from the left, negative values from the right.

Return value

Type	Description
String [▶ 2108]	The shortened string.

Note: Available since version 1.8.

5.10.3 Formatting

5.10.3.1 FormatString

The function FormatString formats a given string by replacing placeholders in the string by the values of the symbols.

Parameter

Name	Type	Description
formatString	String [▶ 2108]	A string with placeholders.
args […]	Any [▶ 2107]	Any value used in a placeholder.

Return value

Type	Description
String [▶ 2108]	String in which the placeholders are replaced by values of the symbols.

Placeholder

Sample: {0|.1f}

Placeholder	Description
{	Placeholder is initiated.
0	Index of the parameter (optional: without an index iteration takes place in sequence).
|	Separator.
+	If a + is specified here, the string is always assigned a sign. For example "+42" for the positive number 42. Negative numbers of course always get a sign.
.X	Number of digits.
f	Type description.
}	End of the placeholder.

Type description

Character	Description
d	Signed decimal integer
i	Signed decimal integer
u	Unsigned decimal integer
o	Unsigned octal
x	Unsigned hexadecimal integer
X	Unsigned hexadecimal integer (upper case)
f	Decimal floating point
e	Scientific notation (mantissa/exponent), lower case
g	Use the shortest representation: e or f
s	String of characters
b	Binary number
t	true or false
T	Type of the argument ('number', 'string', 'null' …)
v	Primitive value of the specified argument
j	JavaScript object or array as a JSON encoded string

Available since version 1.10

Sample

Input:

```
FormatString('Hello {1|.5s}. This is a floating point number - {0|.2f}. Two zeros are attached -
{03i} (3 digits in total).', 2, 'World123', 3);
```

Output:

"Hello World. This is a floating point number – 2.00. Two zeros are attached – 003 (3 digits in total)."

5.10.3.2 GaugeValueFormatter

The function GaugeValueFormatter rounds up a numerical value to a transferred number of decimal places.

Parameter

Name	Type	Description
value	Number [▶ 2108]	Value to be rounded up.
maxDecimals	Number [▶ 2108]	Number of decimal places to which the value is to be rounded.

Return value

Type	Description
String [▶ 2108]	The rounded value in the form of a string.

Note: Available since version 1.8.

5.10.4 Locale

5.10.4.1 GetLocale

The function GetLocale returns the currently active language (with region if available) of the client. Refer also to the description page for the corresponding API function [▶ 1893].

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
String [▶ 2108]	Currently active language (e.g. "de-DE").

Note: Available since version 1.8.

5.10.4.2 GetRegisteredLocales

The function GetRegisteredLocales returns all registered languages (with region if available) of the project. Refer also to the description page for the corresponding API function [▶ 1894].

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
String [▶ 2108][]	An array with all registered languages of the project in the form of a string.

Note: Available since version 1.8.

5.10.4.3 SetLocale

The function SetLocale sets the active language in the client. Refer also to the description page for the corresponding API function [▶ 1893].

Parameter

Name	Type	Description
locale	String [▶ 2108]	A language registered in the project.

Return value

Type	Description
void	No return value.

Note: Available since version 1.8.

5.10.5 Recipe

5.10.5.1 Activate

When the Activate function is executed, the recipe defined in the parameter is activated.

Parameter

Name	Type	Description
recipeReference	String [▶ 2108]	The reference or the recipe (path plus name) that is to be activated.

Return value

Type	Description
void	No return value.

Note: Available since version 1.10.1171.142

5.10.5.2 Teach

The Teach function reads the current values of the symbols of a recipe from the target and stores them in this recipe.

Parameter

Name	Type	Description
recipeReference	String [▶ 2108]	The reference of the recipe (path plus name) in which the current values of the target are to be read and stored.

Return value

Type	Description
void	No return value.

Note: Available since version 1.10.1171.142

5.10.5.3 TeachAsNew

The TeachAsNew function reads the current values of the symbols of a defined recipe from the target and stores them in a new recipe.

Parameter

Name	Type	Description
recipeReference	String [▶ 2108]	The reference of the recipe (path plus name) whose current symbol values are to be read from the target. **Examples:** ‚MyRecipe1‘ ‚MyFolder::MyRecipe2‘
newRecipeName	String [▶ 2108]	The name of the new recipe that is to be created and in which the values are to be stored. If required, this name can already contain the folder tree in which the recipe is to be stored. Otherwise the folder tree can be specified under newRecipePath.
newRecipePath	String [▶ 2108]	The path under which the new recipe is to be stored. If this parameter is empty, the recipe is created at the highest folder level. **Example:** ‚MyFolder‘ ‚MyFolder1::MyFolder2‘

Return value

Type	Description
void	No return value.

Note: Available since version 1.10.1171.142

5.10.6 Theme

5.10.6.1 GetTheme

The function GetTheme returns the name of the active theme. Refer also to the description page for the corresponding API function [▶ 2059].

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
String [▶ 2108]	Name of the active theme.

Note: Available since version 1.10.

5.10.6.2 SetAttributeToThemeValue

The function SetAttributeToThemeValue sets an attribute of a control to the value of the active theme. In the case of a change of theme, the value of the attribute is changed in accordance with the new active theme. Refer also to the description page for the corresponding API function [▶ 2061].

Parameter

Name	Type	Description
control	Control [▶ 2216]	The control for which the attribute is to be set.
propertyName	String [▶ 2108]	Name of the attribute that is to be set to the value of the active theme.

Return value

Type	Description
void	No return value.

Note: Available since version 1.10.

5.10.6.3 SetTheme

The function SetTheme sets the active theme. Refer also to the description page for the corresponding API function [▶ 2060].

Parameter

Name	Type	Description
newTheme	String [▶ 2108]	Available theme that is to be set to active.

Return value

Type	Description
void	No return value.

Note: Available since version 1.10.

5.10.7 UserManagement

5.10.7.1 CheckAccess

The function CheckAccess checks whether the logged-in user has the right to access a control. Refer also to the description page for the corresponding API function [▶ 1836].

Parameter

Name	Type	Description
TcHmiControl	Control [▶ 2216]	The control for which the right is to be checked.
AccessRightToCheck	String [▶ 2108]	The right that is to be checked.

Return value

Type	Description
Boolean [▶ 2107]	true: The user has the right in question. false: The user does not have the right in question.

Note: Available since version 1.8.

5.10.7.2 GoToLoginPage

The GoToLoginPage function calls the login page of the server without logging out the currently logged in user. This makes it possible to change users.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value.

Available from version 1.10.1171.142.

5.10.7.3 Logout

The function Logout logs the current user out of the server. Refer also to the description page for the corresponding API function [▶ 1916].

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value.

Note: Available since version 1.8.

5.11 Publishing

So that a TwinCAT HMI project can be requested from a TE2000 HMI server, it must be transferred to the server.

1. Open the Publish dialog via the context menu or the Visual Studio menu (TwinCAT HMI / Publish to TwinCAT HMI Server).

2. When executing for the first time, create a Publish profile that you can use again later.

3. Enter the IP address or the host name as well as the port of the TwinCAT HMI server. Alternatively you can perform a broadcast search using the ... button and find all available HMI servers in the local network (the administrator password must be set on the server via the Config page). The project can be transferred by HTTP (default setting: port 1010) and HTTPS (default setting: 1020).

4. Select the server configuration that you wish to transfer to the target system.

5. Conclude the procedure by clicking on **Publish**.

⇨ The Visual Studio output window informs you about the course of the procedure.

5.11.1 Autostart Server

The TwinCAT HMI Server can optionally be started automatically when the operating system is started.

Server as process

In versions 1.8 and 1.10 [▶ 17], the TF2000 HMI Server starts as a process on the operating system.

1. Open the Start menu and copy the link from the following folder:
   C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Beckhoff\TF2000 HMI Server

2. Open the TwinCAT Autostart folder and place the link there:
   C:\TwinCAT\3.1\Target\StartUp

⇨ The server starts automatically with the next system startup.

5.11.2 Autostart Client

The client is any HTML5-capable web browser. An overview of the supported versions can be found in the system requirements [▶ 17]. The browsers can be started automatically when the operating system is started. Various options are available for this:

Systems with TwinCAT:

If the browser is located on a system on which TwinCAT is installed, the browser can be started via the TwinCAT Autostart folder. To do this, open the TwinCAT Autostart folder:

C:\TwinCAT\3.1\Target\StartUp

You can then place the link of your browser there:

Systems without TwinCAT

If the browser is located on a system which only functions as a client and on which TwinCAT is not installed, you can use the startup folder of the Windows user. To do this, copy the link of your browser into this folder. You open the startup folder by entering the command "shell:startup" under "Run":

You can then place the link of your browser there:

Set up full screen mode

Some browsers support starting in full screen mode (kiosk mode), which makes it possible to hide the underlying operating system. The kiosk mode is set differently depending on the browser.

In the following, the configuration of the kiosk mode is explained using the "Google Chrome" browser as an example. With this browser, the kiosk mode can be set via startup parameters.

1. Open the link properties.

2. In the **Target** line you can add optional parameters at the end:

In addition to the kiosk mode, further parameters can be set that load different configurations in the browser. For the browser "Google Chrome" there are the following relevant parameters:

Parameter	Description
--kiosk https://ip-of-hmi-server:1020	Start kiosk mode. The URL can load any page (in this case the HMI).
--disable-pinch	Disable zoom in browser.
--overscroll-history-navigation=0	Disable wipe navigation of the browser.
--incognito	Start incognito mode of the browser.

The parameters can be combined with each other, for example:

```
chrome.exe -kiosk https://ip-of-hmi-server:1020 -incognito --disable-pinch --overscroll-history-navigation=0
```

5.12 Server configuration

The configuration of the server and the configuration of the extensions can be managed on the configuration page of the TwinCAT HMI Server. The configuration page of the TwinCAT HMI Server can be opened by double-clicking the extension node in the Solution Explorer. Alternatively you can carry out the configuration directly on the configuration page of the server. Open it via the systray icon of the TwinCAT HMI Server by right-clicking on the "Config" entry.

The server can manage several configurations in order, for example, to configure different controllers. You can add new configurations via "Create new configuration".

The configuration, which is transferred to the remote server, can be selected during the publish process. [▶ 110] Two configurations are supplied by default:

- Default Configuration: Local configuration that is loaded from the Engineering Server (valid in LiveView [▶ 62] and when transferred to the Engineering Server).
- Remote configuration: Configuration that is transferred to the remote server when publishing.

5.13 Package Management

The TwinCAT HMI Package Management makes it possible to export a TwinCAT HMI extensibility project as a so-called TwinCAT HMI Package.

A TwinCAT HMI Package can be installed as a reference in a TwinCAT HMI project.

5.13.1 Creating a package

1. By means of a right mouse click on the uppermost node of the TwinCAT HMI extensibility project, select **Create TwinCAT HMI Package**.

2. Assign a version. (The name is not editable as it is assigned via the project name).

3. Select **Next**.

4. Select all the project files that the package is to contain.

5. Optionally, select an **Industrial Field** for the package and specify the repository for the package. The following options are available for this:
   Export to local repository: Save in C:\TwinCAT\Functions\TE2000-HMI-Engineering\References

Export to project directory: Save in the project directory
Export to specific directory: Save via a specific path (selectable via **Select target directory**)

⇨ The creation of the package is concluded by actuating **Finish**.

To provide the package with further metadata (e.g. author, description), edit the file Manifest.json in the associated TwinCAT HMI extensibility project.

5.13.2 Installing a package

1. Open the Packages window by right-clicking on References **Manage TwinCAT HMI Packages**.

2. Select the package that you wish to install. The following sources are available for this:
   Solution – Repository: Local repository (under C:\TwinCAT\Functions\TE2000-HMI-Engineering \References)

   Solution – Projects: List of packages within the Solution

Browse – Current: Option to add a package by **Browse**.

3. Select **Install** to install the package.

⇨ The package contents are now available for use in the TwinCAT HMI project – framework controls inside the Toolbox, server extension in the Server Extensions area.

The Install option is only available in case the Package does not auto-install and another version of this package is not installed.

In order to be able to save a package without an associated TwinCAT HMI extensibility project in the repository, it must be saved manually in C:\TwinCAT\Functions\TE2000-HMI-Engineering\References.

5.13.3 Uninstalling a package

1. Open the Packages window by right-clicking on References **Manage TwinCAT HMI Packages**.

2. Select the package that you wish to uninstall. The following sources are available for this:
   Solution – Repository: Local repository (under C:\TwinCAT\Functions\TE2000-HMI-Engineering \References)

Solution – Projects: List of packages within the Solution

Browse – Current: List of packages that were added by **Browse**.

3. Select **Uninstall** to uninstall the package.

⇨ The package contents are no longer available for use in the TwinCAT HMI project.

> If no updates of the package take place and the package is really to be removed, all references to the package used in the TwinCAT HMI project should be removed (Framework Control: remove all instances / Server Extension: remove all mapped symbols of the server extension).

5.13.4 Updating a package

1. Uninstall [▶ 125] the package.
2. Install [▶ 123] a different version of the package.

> Since this is an update of the package, all references to the package used in the TwinCAT HMI project should be retained after uninstalling the package (Framework Control: all instances / Server Extension: all mapped symbols of the server extension).

5.14 Gallery Explorer

In the Galley Explorer you can find predefined graphics that are supplied with the TwinCAT HMI. You can add them directly to the graphic editor or to the project tree by drag-and-drop.

5.15 TwinCAT HMI Toolbar

The TwinCAT HMI Engineering supplies its own toolbar for Visual Studio that offers fast access to important functions. You can activate the toolbar in Visual Studio under View\Toolbars.

The toolbar provides the following options:

Publish to HMI server

Edit publishing profile

Cancel publishing

Bring to front

Send to back

Bring one level forward

Send one level backward

Same height

Same width

Insert horizontal intermediate space

Insert vertical intermediate space

Swap

Align left

Align horizontally central

Align right

Align top

Align vertically central

Align bottom

Show developer tools

Manage TwinCAT HMI packages

Toggle Highlight Container visibility

Toggle LiveView's mode

5.16 TwinCAT HMI Items

Further elements can be added to a TwinCAT HMI project via the context menu in the Solution Explorer.

The following elements can be chosen:

Start point for creating individual cascading style sheets.

CodeBehind scripts (JavaScript).

Content [▶ 470], a page that can be used and switched within regions [▶ 496].

Reusable functions that are usable in the client.

A localization file for a further language.

A User Control [▶ 504] can be instanced multiple times and each has a unique ID. All User Controls are listed in the Toolbox.

A view [▶ 515] can be used as an entry page, e.g. configured as a start page and a further view.

6 Controls

A predefined set of controls is supplied in the toolbox with the TwinCAT HMI:

Properties

Each control can be equipped with various properties. The properties are set in the **Properties** under the icon .

Events

Each control can trigger different actions on the occurrence of various events. These actions are configured in the **Properties** under the icon .

Rights

The rights management [▶ 520] for each control can be found in the **Properties** under the icon .

6.1 General configuration

Various generally valid configurations or editors are described in this chapter.

6.1.1 Positioning and calculation of the size of controls

The TwinCAT HMI offers various options for the positioning and calculation of the size of controls. In general, each control is always defined by its position inside its container. In the standard settings these are the values for Left [▶ 440] and Top [▶ 440] combined with Width [▶ 441] and Height [▶ 442]. These four values are specified in the unit [▶ 2242] pixel (px). If percent (%) is specified as the unit for these values, then the size of the parent control is used as a reference. A control with 50% width is thus half as wide as its parent control.

The influence of the different attributes on the size of the control is defined with the aid of the attributes WidthMode [▶ 441] or HeightMode [▶ 442]. All controls have the *Value* and *Parent* modes. Many controls additionally offer the *Content* mode. The attributes necessary for the modes are listed in the following tables. The various separators have the following meanings:

- ‚,‘: both attributes can be used.
- ‚&‘: both attributes must be used.
- ‚|‘: one of the two attributes must be used.

Value (Standard)	Width, Height	Left, Right	Top, Bottom
Parent	-	Left & Right	Top & Bottom
Content	-	Left | Right	Top | Bottom

Value

The Value mode is configured as the default together with the aforementioned attributes Left, Top, Width and Height. However, an arbitrary combination of any two of the three possible attributes in the horizontal and vertical direction is possible in order to define the position and size of a control:

- Left [▶ 440] (including LeftUnit [▶ 440])

- Width [▶ 441] (including WidthUnit [▶ 441])
- Right [▶ 443] (including RightUnit [▶ 443])

and

- Top [▶ 440] (including TopUnit [▶ 440])
- Height [▶ 442] (including HeightUnit [▶ 442])
- Bottom [▶ 443] (including BottomUnit [▶ 443])

If values are configured for all three variables per direction, the last one in this list (i.e. Right or Bottom respectively) is discarded. If only one variable is configured per direction for a control, it **cannot** be displayed!

Parent

The attributes Width and Height are ignored in the Parent mode. The size of the control depends on the size of the parent control. In addition, the distances to the edges of the parent control must be defined by the attribute pairs Left, Right and Top, Bottom. After changing the mode from Value to Parent, therefore, all four attributes must be configured so that the control is displayed.

Content

Most container controls additionally have the Content mode. The attributes Width and Height are ignored here too. The size of the control depends on the size of the child control. In this mode, therefore, the control is just large enough so that every child control is fully visible. Here are some examples of this mode:

- If the only child control is configured with Left = 10 px and Width = 200 px, then the container control is given the width 210 px.
- If the only child control is configured with Right = 20 px and Width = 200 px, then the container control is given the width 220 px.
- If the only child control is configured with Left = 30 px and Right = 20 px and no Width, then the container control is given the width 50 px and the child control has no width.
- If the only child control has Left = 100 px and Width = 50%, then the parent element would only have a size of 100 px.
- If a child control itself is aligned exclusively to the parent control (e.g. it has Left = 10% and Width = 50%), then it is skipped in the size adjustment of the parent control. If there are otherwise no child controls, the container control has the width 0 px and thus **cannot be displayed**!

Minimum / Maximum Width / Height

In addition, a minimum and maximum size can be defined for each control in the horizontal and vertical directions. The attributes Minimum Width [▶ 452] (including Minimum Width Unit [▶ 452]), Maximum Width [▶ 451] (including Maximum Width Unit [▶ 451]) or Minimum Height [▶ 450] (including Minimum Height Unit [▶ 450]) and Maximum Height [▶ 450] (including Maximum Height Unit [▶ 451]) respectively are responsible for this.

Priority: calculated size > calculated maximum value > calculated minimum value

A set maximum value thus overwrites the calculated size, but a set minimum value overwrites the maximum value.

For example, this allows the height of the View [▶ 515] to be adapted to the contents (WidthMode=Content), but at the same time the minimum height can be set to 100% so that, for example, no area is created without a background image.

Overlapping

If two controls (partly) occupy the same area on the screen, then the order will be observed. This order can be changed by drag & drop within the Document Outline [▶ 62] or directly in the HTML source text [▶ 48].

The Z-Index [▶ 454] attribute offers additional options. This can assume negative numeric values and has the default value 0. A control with a higher numerical value in this attribute will be drawn "further forward", i.e. in case of doubt it conceals a control with a lower numerical value. It must be noted that the Z-index also has **no** global meaning, but is valid only inside a container.

Overfilling

A container control [▶ 465] groups together any child controls. The container must be large enough to accept all child controls, because any overfilling will be cut off.

6.1.2 Color editor

You can define any desired color using the color editor embedded in the Properties window. To do this, first select the color to be modified from the **Colors** category and you can then choose between the four following definition options:

Theme: The color defined by the theme will be used.

None: No color will be used.

Solid: A uniform color will be used.

Gradient: A color with a color gradient will be used.

The color editor shown in the following picture is used to define a color as "solid". On the right-hand side you will find the entries **R** (Red), **G** (Green), **B** (Blue) and **A** (Alpha). Here you can directly enter the color via the color components and the Alpha factor, which specifies the transparency. Underneath that is a further box in which you can also define the color by means of its corresponding hexadecimal color code.

A color selection editor is offered on the left-hand side. In the right-hand part of this editor you can select a color to use as the basis. In the main part you can then adjust this color again. In addition it is possible using the pipette symbol to select a color, for example from an image opened in an external tool. Underneath the color selection editor there is a further bar in which the last-used color is offered on the left-hand side and the currently selected color on the right-hand side.

If the **Gradient** tab is selected, the editor described above is extended by the setting options for a color gradient. The color gradient is initially displayed in a bar. By selecting its left or right marker you can change the color with which the gradient starts or ends. The start and end of the color gradient can be adjusted via the position of the markers. This information can additionally be defined in percent in the right box underneath. In addition there is a slider on the left under the color gradient bar which can be used to change the angle with which the gradient is displayed.

6.1.3 Transformation editor

Six different types of transformation can be defined with this dialog:

- Translate: Translate in X-, Y-, Z-direction
- Rotate: Rotate by an angle around an axis defined by a vector (X, Y, Z) – the default value of the vector is (0, 0, 1). The vector thus points out of the screen in the Z-axis direction.
- Scale: Zoom in the X, Y, Z direction, in each case by a factor
- Skew: Skewed display of the horizontal and vertical sides, in each case by an angle
- Origin: Set the origin (X, Y, Z) of the transformation relative to the position of the control
- Perspective: Set the perspective of the transformations of all child elements

Items	List of all objects added
✕	Delete the selected object
↑ ↓	Change the object order (top = left)
Add	Add a new object

Translate

Name	Description
X	Definition of the distance by which the control is to be shifted in the X direction
Y	Definition of the distance by which the control is to be shifted in the Y direction
Z	Definition of the distance by which the control is to be shifted in the Z direction

Rotate

Name	Description
Angle	Definition of an angle by which the control is to be rotated
Angle Unit	Selection of the unit of the angle: • Deg: Degrees (full circle = 360 deg) • Rad: Radians (full circle = 2*Pi*Radians) • Turn: Number of turns (full circle = 1 turn) • Grad: Gon (full circle = 400 grad)
Vector X	Definition of the X component of the vector
Vector Y	Definition of the Y component of the vector
Vector Z	Definition of the Z component of the vector

Scale

Name	Description
X	Definition of the zoom factor in the X direction
Y	Definition of the zoom factor in the Y direction
Z	Definition of the zoom factor in the Z direction

Skew

Name	Description
X Angle	Definition of the angle by which the vertical sides are to be skewed
X Angle Unit	Selection of the unit of the angle: • Deg: Degrees (full circle = 360 deg) • Rad: Radians (full circle = 2*Pi*Radians) • Turn: Number of turns (full circle = 1 turn) • Grad: Gon (full circle = 400 grad)
Y Angle	Definition of the angle by which the horizontal sides are to be skewed
Y Angle Unit	Selection of the unit of the angle: • Deg: Degrees (full circle = 360 deg) • Rad: Radians (full circle = 2*Pi*Radians) • Turn: Number of turns (full circle = 1 turn) • Grad: Gon (full circle = 400 grad)

Origin

Name	Description
X	Definition of the X coordinate of the origin of the transformations
X Unit	Selection of the unit of the coordinate: • Pixels for an absolute position • Percent for a relative position
Y	Definition of the Y coordinate of the origin of the transformations
Y Unit	Selection of the unit of the coordinate: • Pixels for an absolute position • Percent for a relative position
Z	Definition of the Z coordinate of the origin of the transformations
Z Unit	Selection of the unit of the coordinate: • Pixels for an absolute position • Percent for a relative position

Perspective

Name	Description
Distance	Definition of the distance of the perspective
Distance Unit	Selection of the unit of the coordinate: • Pixels for an absolute position • Percent for a relative position
Origin X	Definition of the X coordinate of the origin of the perspective
Origin X Unit	Selection of the unit of the coordinate: • Pixels for an absolute position • Percent for a relative position
Origin Y	Definition of the Y coordinate of the origin of the perspective
Origin Y Unit	Selection of the unit of the coordinate: • Pixels for an absolute position • Percent for a relative position

6.1.4 Box Shadow Editor

You can set a shadow for the control with this editor.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Category: Colors

Name	Description
Color	Definition of the color of the shadow

Category: General

Name	Description
blur	Definition of the blur
blur unit	Selecting the unit: • Pixels for absolute blurring
inset	Selection of whether the shadow should be drawn at the top left inside the control instead of at the bottom right outside the control
offsetX	Definition of the offset in the X direction
offsetX unit	Selecting the unit: • Pixel for an absolute offset
offsetY	Definition of the offset in the Y direction
offsetY unit	Selecting the unit: • Pixel for an absolute offset
spread	Definition of the spread of the shadow
spread unit	Selecting the unit: • Pixel for an absolute spread

6.2 Beckhoff Controls

Some display names of attributes have been renamed from version 1.10.

Name	Description
Bar Chart [▶ 142]	Bar chart that can be configured with various elements.
Button [▶ 167]	Simple button.
Checkbox [▶ 179]	Simple checkbox. A checkbox has a ToggleState parameter. This parameter has the value "Normal" if no checkmark is set and "Active" if the checkmark is set. The .onToggleStateChanged event can be used to perform actions when the state of the checkbox changes.
Combobox [▶ 185]	A combo box is a drop-down menu for selecting one of several options. Each option is defined by its ID and also has a text that is displayed in the drop-down menu, and a value.
Datagrid [▶ 203]	Table for generic display of data.
Ellipse [▶ 229]	Ellipse
Event Grid [▶ 235]	Table for displaying events.
IFrame [▶ 246]	Nested browser context. Embeds another HTML page in the current visualization.
Image [▶ 251]	Image display
Keyboard [▶ 258]	On-screen keyboard
Line [▶ 272]	Straight line
Linear Gauge [▶ 306]	Linear display
Line Chart [▶ 281]	Line chart
Polygon [▶ 329]	Closed shape formed from a set of interconnected straight lines. The last point is automatically connected to the first point.
Radial Gauge [▶ 337]	Radial display
Rectangle [▶ 359]	Rectangle
Textblock [▶ 367]	Text block for displaying text
Textbox [▶ 376]	Text box for entering text.
Toggle Button [▶ 386]	Button that can be toggled between two states.
Trend Line Chart [▶ 393]	Line chart for historicized data
Video [▶ 421]	Display area for a video.

6.2.1 Bar Chart

A BarChart represents data with rectangular bars. It can be statically filled with data or dynamically supplied with data via an array.

A description of the configuration can be found in chapter Configuration [▶ 151].

Attributes

Category: Colors

Name	Description
GridBackgroundColor [▶ 153]	Definition of the background color of the coordinate system
XAxisColor [▶ 157]	Definition of the color of the x-axis
YAxisColor [▶ 162]	Definition of the color of the y-axis
GridLineColor [▶ 153]	Definition of the color of the coordinate grid
SubgridLineColor [▶ 157]	Definition of the color of the coordinate subgrid
XLabelFontColor [▶ 158]	Definition of the font color of the x-axis labels
YLabelFontColor [▶ 162]	Definition of the font color of the y-axis labels
YAxisNameFontColor [▶ 164]	Definition of the font color of the y-axis name
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
BarGraphData [▶ 153]	Definition of the data to be displayed as graph using the 'Bar Graph Data [▶ 152]' dialog
BarGraphColors [▶ 152]	Definition of the colors of the configured bar graphs using the 'Bar Graph Color [▶ 151]' dialog
BarGraphWidth [▶ 153]	Definition of the width of the individual bars.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: XAxis

Name	Description
ShowXAxis [▶ 155]	Selecting whether the x-axis is to be displayed
XShowLabels [▶ 155]	Selecting whether the x-axis labels are to be displayed
XMainTickSteps [▶ 160]	Definition of the step size of the x-axis main subdivision
XMainTickMinValue [▶ 159]	Definition of the minimum value of the x-axis main subdivision
XMainTickMaxValue [▶ 159]	Definition of the maximum value of the x-axis main subdivision
ShowXSubTicks [▶ 155]	Selecting whether the subcoordinate points are to be displayed on the x-axis
XSubTickSteps [▶ 160]	Definition of the step size of the x-axis subdivision
XAxisAutoScaling [▶ 157]	Selecting whether the x-axis should be automatically scaled according to the data sets. 'Maintick Min Value', 'Maintick Max Value' and 'X-Axis Maintick Steps' need not be set.
XAxisDecimalPlaces [▶ 158]	Definition of the decimal places displayed for the x-axis
XAxisWidth [▶ 160]	Definition of the x-axis width
XLabelFontFamily [▶ 158]	Definition of the font using the name of the font family or the keyword
XLabelFontSize [▶ 158]	Definition of the font size for the x-axis labels
XLabelFontSizeUnit [▶ 159]	Selecting the font size unit for the x-axis labels: • Pixels for an absolute size • Percent for a relative size
XLabelFontWeight [▶ 159]	Selecting the font weight of the x-axis labels: • Normal • Bold: Bold
XAxisFormat [▶ 160]	Definition of the formatting of the x-axis labels

Category: YAxis

Name	Description
ShowYAxis [▶ 156]	Selecting whether the y-axis is to be displayed
YPosition [▶ 166]	Selecting the position of the y-axis on the coordinate system • Left: Y-axis is displayed on the left side of the coordinate system. • Right: Y-axis is displayed on the right side of the coordinate system.
YAxisAutoScaling [▶ 161]	Selecting whether the y-axis should be automatically scaled according to the data sets. 'Maintick Min Value', 'Maintick Max Value' and 'Y-Axis Maintick Steps' need not be set.
YAxisDecimalPlaces [▶ 162]	Definition of the decimal places displayed for the y-axis
YAxisUnit [▶ 166]	Definition of the unit of the y-axis
YAxisWidth [▶ 166]	Definition of the y-axis width
ShowYAxisName [▶ 156]	Selecting whether the y-axis name is to be displayed
YAxisName [▶ 164]	Definition of the y-axis name
YAxisNameFontFamily [▶ 164]	Definition of the font using the name of the font family or the keyword
YAxisNameFontSize [▶ 165]	Definition of the font size of the y-axis name
YAxisNameFontSizeUnit [▶ 165]	Selecting the unit of the font size of the y-axis name: • Pixels for an absolute size • Percent for a relative size
YAxisNameFontWeight [▶ 165]	Selecting of the font weight of the y-axis name: • Normal • Bold: Bold
YShowLabels [▶ 156]	Selecting whether the y-axis labels are to be displayed
YLabelFontFamily [▶ 162]	Definition of the font using the name of the font family or the keyword
YLabelFontSize [▶ 163]	Definition of the font size for the y-axis labels
YLabelFontSizeUnit [▶ 163]	Selecting the font size unit for the y-axis labels: • Pixels for an absolute size • Percent for a relative size
YLabelFontWeight [▶ 163]	Selecting the font weight of the y-axis labels: • Normal • Bold: Bold
YMainTickMinValue [▶ 164]	Definition of the minimum value of the y-axis main subdivision
YMainTickMaxValue [▶ 163]	Definition of the maximum value of the y-axis main subdivision
YMainTickSteps [▶ 164]	Definition of the step size of the y-axis main subdivision
ShowYSubTicks [▶ 156]	Selecting whether the subcoordinate points are to be displayed on the y-axis
YSubTickSteps [▶ 166]	Definition of the step size of the y-axis subdivision

Category: Grid

Name	Description
ShowGrid [▶ 154]	Selecting whether the coordinate grid is to be displayed
GridShowHorizontalLines [▶ 154]	Selecting whether the horizontal lines of the coordinate grid are to be displayed
GridShowVerticalLines [▶ 154]	Selecting whether the vertical lines of the coordinate grid are to be displayed
GridLineWidth [▶ 154]	Definition of the line width of the coordinate grid
SubgridShowHorizontalLines [▶ 154]	Selecting whether the horizontal lines of the coordinate subgrid are to be displayed
SubgridShowVerticalLines [▶ 155]	Selecting whether the vertical lines of the coordinate subgrid are to be displayed
SubgridLineWidth [▶ 157]	Definition of the line width of the coordinate subgrid

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Bar Chart

Available since version 1.8

See also

API reference [▶ 1249] for this control

6.2.1.1 Configuration

A bar chart can be configured in different ways:

- Via Engineering [▶ 151]
  - Array of numbers
  - Array of arrays of numbers
- Via an array from the PLC [▶ 151]
  - Array of numbers

```
aVaules : ARRAY [1..10] OF INT;
```

  - Array of arrays of numbers

```
aValues : ARRAY [1..2, 1..10] OF INT;
```

Definition via Engineering:

1. Add a bar chart to the interface.

2. Define which type of object you want to use.

Array of numbers (Array<number>):

3. Define the number of members of the array and their values for the 'Bar Graph Data' attribute of the 'Common' category using the 'Bar Graph Data [▶ 152]' dialog.

Array of arrays of numbers (Array<Array<number>>)

3. Define the number of data sets of the array for the 'Bar Graph Data' attribute of the 'Common' category using the 'Bar Graph Data [▶ 152]' dialog.

4. Define the number of members of the array and their values using the 'Bar Graph Data [▶ 152]' dialog.

Definition via an array from the PLC:

1. Add a bar chart to the interface.

2. Link the array to the 'Bar Graph Data' attribute of the 'Common' category.

Advanced Settings:

1. Define the color of the individual data sets for the 'Bar Graph Colors' attribute of the 'Common' category using the 'Bar Graph Color [▶ 151]' dialog.

2. Deactivate the 'X-Axis Autoscaling' attribute of the 'X-Axis' category.

3. Define the minimum and maximum value of the x-axis corresponding to your data set in the 'X-Axis Maintick Min Value' and 'X-Axis Maintick Max Value' attributes of the 'X-Axis' category.

4. Define the step size matching your data set for the 'X-Axis Maintick Steps' attribute of the 'X-Axis' category.

Example: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007203226236427.zip

6.2.1.2 Dialogs

6.2.1.2.1 Bar Graph Color Dialog

Use this dialog you define the color of the bar graphs. The color objects are assigned to the arrays of the data sets sequentially in case of data structure 'Array<Array<number>>'.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
Add	Add a new object

Category: Colors

Name	Description
Color	Definition of the color of the bar graph

6.2.1.2.2 Bar Graph Data Dialog

You can use this dialog to statically define the data that the bar graph is to display in Engineering. To do this, you can first choose between the structure of the data as 'Array<Array<number>> ' and 'Array<number>'. In the second step, you can then define the entries.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
Add	Add a new object

If you have selected the data structure 'Array<Array<number>>', you can first define different arrays, each representing a bar graph.

Name	Description
Array	Definition of an array of numbers representing a data set to be displayed as a bar graph

Each of these arrays has entries that you can define using the second version of the dialog. If you use the data structure 'Array<number>', this dialog is offered directly.

Name	Description
Number	Definition of a value from a data set to be displayed as a bar graph

6.2.1.3 Attributes

6.2.1.3.1 BarGraphColors

Scheme: tchmi:framework#/definitions/ChartBarGraphColorList [▶ 2205]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getBarGraphColors [▶ 1265]

Attribute setter: setBarGraphColors [▶ 1266]

Note: Available since version 1.8

6.2.1.3.2 BarGraphData

Scheme: tchmi:framework#/definitions/ChartBarGraphDataDefinitionList [▶ 2206]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getBarGraphData [▶ 1266]

Attribute setter: setBarGraphData [▶ 1267]

Note: Available since version 1.8

6.2.1.3.3 BarGraphWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getBarGraphWidth [▶ 1267]

Attribute setter: setBarGraphWidth [▶ 1268]

Note: Available since version 1.10

6.2.1.3.4 GridBackgroundColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getGridBackgroundColor [▶ 1258]

Attribute setter: setGridBackgroundColor [▶ 1259]

Note: Available since version 1.8

6.2.1.3.5 GridLineColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getGridLineColor [▶ 1261]

Attribute setter: setGridLineColor [▶ 1261]

Note: Available since version 1.8

6.2.1.3.6 GridLineWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getGridLineWidth [▶ 1281]

Attribute setter: setGridLineWidth [▶ 1282]

Note: Available since version 1.8

6.2.1.3.7 ShowGrid

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getShowGrid [▶ 1285]

Attribute setter: setShowGrid [▶ 1285]

Note: Available since version 1.8

6.2.1.3.8 GridShowHorizontalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getGridShowHorizontalLines [▶ 1284]

Attribute setter: setGridShowHorizontalLines [▶ 1284]

Note: Available since version 1.8

6.2.1.3.9 GridShowVerticalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getGridShowVerticalLines [▶ 1278]

Attribute setter: setGridShowVerticalLines [▶ 1279]

Note: Available since version 1.8

6.2.1.3.10 SubgridShowHorizontalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getSubgridShowHorizontalLines [▶ 1307]

Attribute setter: setSubgridShowHorizontalLines [▶ 1307]

Note: Available since version 1.8

6.2.1.3.11 SubgridShowVerticalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getSubgridShowVerticalLines [▶ 1306]

Attribute setter: setSubgridShowVerticalLines [▶ 1307]

Note: Available since version 1.8

6.2.1.3.12 ShowXAxis

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getShowXAxis [▶ 1268]

Attribute setter: setShowXAxis [▶ 1269]

Note: Available since version 1.8

6.2.1.3.13 XShowLabels

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXShowLabels [▶ 1269]

Attribute setter: setXShowLabels [▶ 1270]

Note: Available since version 1.8

6.2.1.3.14 ShowXSubTicks

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getShowXSubTicks [▶ 1276]

Attribute setter: setShowXSubTicks [▶ 1276]

Note: Available since version 1.8

6.2.1.3.15 ShowYAxis

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getShowYAxis [▶ 1286]

Attribute setter: setShowYAxis [▶ 1286]

Note: Available since version 1.8

6.2.1.3.16 YShowLabels

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYShowLabels [▶ 1289]

Attribute setter: setYShowLabels [▶ 1289]

Note: Available since version 1.8

6.2.1.3.17 ShowYAxisName

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getShowYAxisName [▶ 1290]

Attribute setter: setShowYAxisName [▶ 1291]

Note: Available since version 1.8

6.2.1.3.18 ShowYSubTicks

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getShowYSubTicks [▶ 1288]

Attribute setter: setShowYSubTicks [▶ 1288]

Note: Available since version 1.8

6.2.1.3.19 SubgridLineColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getSubgridLineColor [▶ 1262]

Attribute setter: setSubgridLineColor [▶ 1262]

Note: Available since version 1.8

6.2.1.3.20 SubgridLineWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getSubgridLineWidth [▶ 1308]

Attribute setter: setSubgridLineWidth [▶ 1308]

Note: Available since version 1.8

6.2.1.3.21 XAxisAutoScaling

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXAxisAutoScaling [▶ 1272]

Attribute setter: setXAxisAutoScaling [▶ 1272]

Note: Available since version 1.8

6.2.1.3.22 XAxisColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXAxisColor [▶ 1260]

Attribute setter: setXAxisColor [▶ 1260]

Note: Available since version 1.8

6.2.1.3.23 XAxisDecimalPlaces

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXAxisDecimalPlaces [▶ 1273]

Attribute setter: setXAxisDecimalPlaces [▶ 1273]

Note: Available since version 1.8

6.2.1.3.24 XLabelFontColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXLabelFontColor [▶ 1263]

Attribute setter: setXLabelFontColor [▶ 1263]

Note: Available since version 1.8

6.2.1.3.25 XLabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXLabelFontFamily [▶ 1279]

Attribute setter: setXLabelFontFamily [▶ 1280]

Note: Available since version 1.8

6.2.1.3.26 XLabelFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXLabelFontSize [▶ 1280]

Attribute setter: setXLabelFontSize [▶ 1281]

Note: Available since version 1.8

6.2.1.3.27 XLabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXLabelFontSizeUnit [▶ 1277]

Attribute setter: setXLabelFontSizeUnit [▶ 1277]

Note: Available since version 1.8

6.2.1.3.28 XLabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXLabelFontWeight [▶ 1282]

Attribute setter: setXLabelFontWeight [▶ 1283]

Note: Available since version 1.8

6.2.1.3.29 XMainTickMaxValue

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXMainTickMaxValue [▶ 1271]

Attribute setter: setXMainTickMaxValue [▶ 1271]

Note: Available since version 1.8

6.2.1.3.30 XMainTickMinValue

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXMainTickMinValue [▶ 1275]

Attribute setter: setXMainTickMinValue [▶ 1275]

Note: Available since version 1.8

6.2.1.3.31 XMainTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXMainTickSteps [▶ 1274]

Attribute setter: setXMainTickSteps [▶ 1274]

Note: Available since version 1.8

6.2.1.3.32 XSubTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXSubTickSteps [▶ 1270]

Attribute setter: setXSubTickSteps [▶ 1271]

Note: Available since version 1.8

6.2.1.3.33 XAxisWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXAxisWidth [▶ 1277]

Attribute setter: setXAxisWidth [▶ 1278]

Note: Available since version 1.8

6.2.1.3.34 XAxisFormat

Schema: tchmi:general#/definitions/Number [▶ 2186]

Definition of the formatting of the X-axis labels.

The selected function defines the X-axis labels. It is called for each displayed value of the main X-axis subdivision. To do this, it receives the index of the BarGraphData [▶ 153] attribute.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The current index of the value on the X axis.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The new value of the main X-axis subdivision. HTML is not allowed here.

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getXAxisWidth [▶ 1277]

Attribute setter: setXAxisWidth [▶ 1278]

Available since version 1.10

6.2.1.3.35 YAxisAutoScaling

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisAutoScaling [▶ 1296]

Attribute setter: setYAxisAutoScaling [▶ 1296]

Note: Available since version 1.8

6.2.1.3.36 YAxisColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisColor [▶ 1259]

Attribute setter: setYAxisColor [▶ 1259]

Note: Available since version 1.8

6.2.1.3.37 YAxisDecimalPlaces

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisDecimalPlaces [▶ 1294]

Attribute setter: setYAxisDecimalPlaces [▶ 1295]

Note: Available since version 1.8

6.2.1.3.38 YLabelFontColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYLabelFontColor [▶ 1264]

Attribute setter: setYLabelFontColor [▶ 1264]

Note: Available since version 1.8

6.2.1.3.39 YLabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYLabelFontFamily [▶ 1304]

Attribute setter: setYLabelFontFamily [▶ 1305]

Note: Available since version 1.8

6.2.1.3.40 YLabelFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYLabelFontSize [▶ 1298]

Attribute setter: setYLabelFontSize [▶ 1298]

Note: Available since version 1.8

6.2.1.3.41 YLabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYLabelFontSizeUnit [▶ 1302]

Attribute setter: setYLabelFontSizeUnit [▶ 1303]

Note: Available since version 1.8

6.2.1.3.42 YLabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYLabelFontWeight [▶ 1301]

Attribute setter: setYLabelFontWeight [▶ 1302]

Note: Available since version 1.8

6.2.1.3.43 YLabelFontWeight

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYMainTickMaxValue [▶ 1295]

Attribute setter: setYMainTickMaxValue [▶ 1295]

Note: Available since version 1.8

6.2.1.3.44 YMainTickMinValue

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYMainTickMinValue [▶ 1292]

Attribute setter: setYMainTickMinValue [▶ 1293]

Note: Available since version 1.8

6.2.1.3.45 YMainTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYMainTickSteps [▶ 1297]

Attribute setter: setYMainTickSteps [▶ 1297]

Note: Available since version 1.8

6.2.1.3.46 YAxisName

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisName [▶ 1291]

Attribute setter: setYAxisName [▶ 1292]

Note: Available since version 1.8

6.2.1.3.47 YAxisNameFontColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisNameFontColor [▶ 1265]

Attribute setter: setYAxisNameFontColor [▶ 1265]

Note: Available since version 1.8

6.2.1.3.48 YAxisNameFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisNameFontFamily [▶ 1305]

Attribute setter: setYAxisNameFontFamily [▶ 1306]

Note: Available since version 1.8

6.2.1.3.49 YAxisNameFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisNameFontSize [▶ 1303]

Attribute setter: setYAxisNameFontSize [▶ 1304]

Note: Available since version 1.8

6.2.1.3.50 YAxisNameFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisNameFontSizeUnit [▶ 1300]

Attribute setter: setYAxisNameFontSizeUnit [▶ 1300]

Note: Available since version 1.8

6.2.1.3.51 YAxisNameFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisNameFontWeight [▶ 1301]

Attribute setter: setYAxisNameFontWeight [▶ 1301]

Note: Available since version 1.8

6.2.1.3.52 YSubTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYSubTickSteps [▶ 1289]

Attribute setter: setYSubTickSteps [▶ 1290]

Note: Available since version 1.8

6.2.1.3.53 YAxisUnit

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisUnit [▶ 1293]

Attribute setter: setYAxisUnit [▶ 1294]

Note: Available since version 1.8

6.2.1.3.54 YAxisWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYAxisWidth [▶ 1299]

Attribute setter: setYAxisWidth [▶ 1299]

Note: Available since version 1.8

6.2.1.3.55 YPosition

Scheme: tchmi:framework#/definitions/ChartBarGraphPosition [▶ 2207]

Origin: Bar Chart [▶ 142]

See also

Attribute getter: getYPosition [▶ 1287]

Attribute setter: setYPosition [▶ 1287]

Note: Available since version 1.8

6.2.1.4 Themed Resources

These resources cannot be defined for each instance but only for several/all bar charts via the theme file. If you really want to change only one instance, you must create a class for it.

Note: Available since version 1.10

6.2.1.4.1 DefaultGraphColor

This is the default color used for graphs when nothing is defined in the BarGraphColors [▶ 152] attribute.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Bar Chart [▶ 142]

Note: Available since version 1.10

6.2.2 Button

The button is a simple switching element that provides visual feedback when the background is switched. By querying the various Operator Events [▶ 172] a logic can be executed in response.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Text [▶ 176]	Definition of the text displayed in the button
StateSymbol [▶ 178]	Reflects the pressed state of the button for the assigned symbol and vice versa. If the button is pressed, the symbol is set to true. If the button is released, the symbol is set to false. If the symbol is set to true independent of the button, the button is pressed. If the symbol is set to false independent of the button, the button is released. During an active user interaction, no value changes of the symbol are transferred to the button.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Icon

Name	Description
Icon [▶ 175]	Definition of an optional image in the button
IconWidth [▶ 177]	Definition of the width of the icon
IconWidthUnit [▶ 177]	Definition of the unit of the width: • Pixels for an absolute width • Percent for a relative width
IconHeight [▶ 174]	Definition of the height of the icon
IconHeightUnit [▶ 174]	Definition of the unit of height: • Pixels for an absolute height • Percent for a relative height
IconHorizontalAlignment [▶ 175]	Selecting the horizontal alignment of the icon within the control
IconVerticalAlignment [▶ 176]	Selecting the vertical alignment of the icon within the control
IconPadding [▶ 175]	Definition of the distance of the icon to the border

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Text

Name	Description
TextColor [▶ 176]	Definition of the text color
TextFontFamily [▶ 173]	Definition of the font using the name of the font family or the keyword
TextFontSize [▶ 173]	Definition of the font size
TextFontSizeUnit [▶ 173]	Definition of the font size unit: • Pixels for an absolute size • Percent for a relative size
TextFontStyle [▶ 174]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
TextFontWeight [▶ 174]	Selecting the font weight of the text: • Normal • Bold: Bold
TextPadding [▶ 176]	Definition of the distance of the text to the border
TextVerticalAlignment [▶ 177]	Selecting the vertical alignment of the text
TextHorizontalAlignment [▶ 175]	Selecting the horizontal alignment of the text
WordWrap [▶ 178]	Selecting whether the text wraps if it is too long for one row

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onStateChanged [▶ 178]	The pressed state has changed.
onStatePressed [▶ 178]	The pressed state has changed from not pressed to pressed.
onStateReleased [▶ 179]	The pressed state has changed from pressed to not pressed.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Button

Note: Available since version 1.8

See also

API reference [▶ 1309] for this control

6.2.2.1 Attributes

6.2.2.1.1 TextFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Button [▶ 167]

See also

Attribute getter: getTextFontFamily [▶ 1332]

Attribute setter: setTextFontFamily [▶ 1333]

Note: Available since version 1.8

6.2.2.1.2 TextFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Button [▶ 167]

See also

Attribute getter: getTextFontSize [▶ 1326]

Attribute setter: setTextFontSize [▶ 1326]

Note: Available since version 1.8

6.2.2.1.3 TextFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Button [▶ 167]

See also

Attribute getter: getTextFontSizeUnit [▶ 1327]

Attribute setter: setTextFontSizeUnit [▶ 1327]

Note: Available since version 1.8

6.2.2.1.4 TextFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Button [▶ 167]

See also

Attribute getter: getTextFontStyle [▶ 1330]

Attribute setter: setTextFontStyle [▶ 1331]

Note: Available since version 1.8

6.2.2.1.5 TextFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Button [▶ 167]

See also

Attribute getter: getTextFontWeight [▶ 1331]

Attribute setter: setTextFontWeight [▶ 1332]

Note: Available since version 1.8

6.2.2.1.6 IconHeight

Numerical value of the image height of the icon.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Button [▶ 167]

See also

Attribute getter: getIconHeight [▶ 1320]

Attribute setter: setIconHeight [▶ 1321]

Note: Available since version 1.8

6.2.2.1.7 IconHeightUnit

Pixels, or percent for relative positioning.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Button [▶ 167]

See also

Attribute getter: getIconHeightUnit [▶ 1321]

Attribute setter: setIconHeightUnit [▶ 1322]

Note: Available since version 1.8

6.2.2.1.8 IconHorizontalAlignment

Horizontal alignment of the icon.

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Button [▶ 167]

See also

Attribute getter: getIconHorizontalAlignment [▶ 1323]

Attribute setter: setIconHorizontalAlignment [▶ 1323]

Note: Available since version 1.8

6.2.2.1.9 TextHorizontalAlignment

Horizontal alignment of the text.

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Button [▶ 167]

See also

Attribute getter: getTextHorizontalAlignment [▶ 1325]

Attribute setter: setTextHorizontalAlignment [▶ 1325]

Note: Available since version 1.8

6.2.2.1.10 Icon

Defines an optional image in the button.

Scheme: tchmi:framework#/definitions/Path [▶ 2248]

Origin: Button [▶ 167]

See also

Attribute getter: getIcon [▶ 1317]

Attribute setter: setIcon [▶ 1318]

Note: Available since version 1.8

6.2.2.1.11 IconPadding

Defines the distance of the icon to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Button [▶ 167]

See also

Attribute getter: getIconPadding [▶ 1324]

Attribute setter: setIconPadding [▶ 1324]

Note: Available since version 1.8

6.2.2.1.12 TextPadding

Defines the distance of the text to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Button [▶ 167]

See also

Attribute getter: getTextPadding [▶ 1328]

Attribute setter: setTextPadding [▶ 1329]

Note: Available since version 1.8

6.2.2.1.13 Text

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Button [▶ 167]

See also

Attribute getter: getText [▶ 1316]

Attribute setter: setText [▶ 1317]

Note: Available since version 1.8

6.2.2.1.14 TextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Button [▶ 167]

See also

Attribute getter: getTextColor [▶ 1316]

Attribute setter: setTextColor [▶ 1316]

Note: Available since version 1.8

6.2.2.1.15 IconVerticalAlignment

Vertical alignment of the icon.

Scheme: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Button [▶ 167]

See also

Attribute getter: getIconVerticalAlignment [▶ 1322]

Attribute setter: setIconVerticalAlignment [▶ 1322]

Note: Available since version 1.8

6.2.2.1.16 TextVerticalAlignment

Vertical alignment of the text.

Scheme: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Button [▶ 167]

See also

Attribute getter: getTextVerticalAlignment [▶ 1328]

Attribute setter: setTextVerticalAlignment [▶ 1328]

Note: Available since version 1.8

6.2.2.1.17 IconWidth

Numerical value of the image width of the icon.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Button [▶ 167]

See also

Attribute getter: getIconWidth [▶ 1318]

Attribute setter: setIconWidth [▶ 1319]

Note: Available since version 1.8

6.2.2.1.18 IconWidthUnit

Pixels, or percent for relative positioning.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Button [▶ 167]

See also

Attribute getter: getIconWidthUnit [▶ 1319]

Attribute setter: setIconWidthUnit [▶ 1320]

Note: Available since version 1.8

6.2.2.1.19 WordWrap

Wraps the text if it is too long to fit in one row.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Button [▶ 167]

See also

Attribute getter: getWordWrap [▶ 1329]

Attribute setter: setWordWrap [▶ 1330]

Note: Available since version 1.8

6.2.2.1.20 StateSymbol

Reflects the pressed state of the button for the assigned symbol and vice versa.

If the button is pressed, the symbol is set to true.

If the button is released, the symbol is set to false.

If the symbol is set to true independent of the button, the button is pressed.

If the symbol is set to false independent of the button, the button is released.

During an active user interaction, no value changes of the symbol are transferred to the button.

Scheme: tchmi:general#/definitions/TcHmiButtonStateSymbol [▶ 2254]

Origin: Button [▶ 167]

See also

Attribute getter: getStateSymbol [▶ 1333]

Attribute setter: setStateSymbol [▶ 1334]

Note: Available since version 1.10

6.2.2.2 Events

6.2.2.2.1 onStateChanged

The pressed state has changed.

Origin: Button [▶ 386]

Note: Available since version 1.10

6.2.2.2.2 onStatePressed

The pressed state has changed from not pressed to pressed.

Origin: Button [▶ 386]

Note: Available since version 1.10

6.2.2.2.3 onStateReleased

The pressed state has changed from pressed to not pressed.

Origin: Button [▶ 386]

Note: Available since version 1.10

6.2.3 Checkbox

A checkbox has a ToggleState [▶ 184] parameter. This parameter has the value "Normal" if no checkmark is set and "Active" if the checkmark is set.

The .onToggleStateChanged [▶ 185] event can be used to perform actions when the state of the checkbox changes.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control
ToggleStateIconColor [▶ 185]	Definition of the color of the toggle state icon

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
StateSymbol [▶ 184]	Reflects the state of the checkbox to the assigned symbol and vice versa.
ToggleState [▶ 184]	Definition of the internal switching state of the control
ToggleGroup [▶ 184]	User-defined name for a group. Only one control in this group can be active.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Control

Name	Description
onToggleStateChanged [▶ 185]	The ToggleState value of the checkbox has changed.
onStateChanged [▶ 185]	The status of the checkbox has changed.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Checkbox

Available since version 1.8

See also

API reference [▶ 1334] for this control

6.2.3.1 Attributes

6.2.3.1.1 StateSymbol

Reflects the state of the checkbox to the assigned symbol and vice versa.

Scheme: tchmi:framework#/definitions/TcHmiCheckboxStateSymbol [▶ 2254]

Origin: Checkbox [▶ 179]

See also

Attribute getter: getStateSymbol [▶ 1342]

Attribute setter: setStateSymbol [▶ 1343]

Note: Available since version 1.10

6.2.3.1.2 ToggleState

Internal state of the control. This can be Normal or Active.

Scheme: tchmi:framework#/definitions/ToggleState [▶ 2254]

Origin: Checkbox [▶ 179]

See also

Attribute getter: getToggleState [▶ 1341]

Attribute setter: setToggleState [▶ 1342]

Note: Available since version 1.8

6.2.3.1.3 ToggleGroup

User-defined name for a group. Only one control in this group can be active.

A ToggleGroup can consist of controls of the type Toggle Button [▶ 386] and Checkbox. [▶ 179]

For a more detailed description of the functionality, see the documentation at the ToggleGroup attribute of the ToggleButton [▶ 392].

Schema: tchmi:general#/definitions/String [▶ 2188]

Origin: Checkbox [▶ 179]

See also

Attribute getter: getToggleGroup [▶ 1340]

Attribute setter: setToggleGroup [▶ 1341]

Available since version 1.10

6.2.3.1.4 ToggleStateIconColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Checkbox [▶ 179]

See also

Attribute getter: getToggleStateIconColor [▶ 1340]

Attribute setter: setToggleStateIconColor [▶ 1340]

Note: Available since version 1.8

6.2.3.2 Events

6.2.3.2.1 onToggleStateChanged

The ToggleState [▶ 184] value of the checkbox has changed.

Origin: Checkbox [▶ 179]

Note: Available since version 1.8

6.2.3.2.2 onStateChanged

The state has changed.

Origin: Checkbox [▶ 179]

Note: Available since version 1.10

6.2.4 Combobox

A combo box is a drop-down menu for selecting an option from several options. Each option is defined by its ID and also has a text that is displayed in the drop-down menu, and an assigned value.

Note that the dropdown menu for the size of the control will be added to this. This becomes relevant for the topic Overfill [▶ 134] with small containers.

A description of the configuration can be found in chapter Configuration [▶ 193].

Attributes

Category: Colors

Name	Description
TextColor [▶ 202]	Definition of the text color
DropDownTextColor [▶ 197]	Definition of the text color in the drop-down list
DropDownBackgroundColor [▶ 195]	Definition of the background color of the drop-down list
DropDownHighlightColor [▶ 196]	Definition of the highlight color of the background of a selected entry in the drop-down list
DropDownHighlightTextColor [▶ 196]	Definition of the highlight color of the text of a selected entry in the drop-down list
DropDownToggleButtonBackgroundColor [▶ 197]	Definition of the color of the button to expand the drop-down list
DropDownToggleButtonArrowColor [▶ 197]	Definition of the color of the arrow of the button to expand the drop-down list
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Data

Name	Description
SrcData [▶ 202]	Definition of data for the list entries
Text [▶ 202]	Definition of the text that is displayed if free text entry is allowed
AllowTextInput [▶ 203]	Definition as to whether free text entry is to be allowed
ComboboxText [▶ 194]	Definition of the text that is displayed if no entry is selected
SelectedIndex [▶ 201]	Definition of the selected entry via its index. The first entry in a drop-down list starts at index 0.
ContentPadding [▶ 203]	Defines the distance of the text to the border.

Category: Text

Name	Description
TextFontFamily [▶ 198]	Definition of the font using the name of the font family or the keyword
TextFontSize [▶ 198]	Definition of the font size
TextFontSizeUnit [▶ 199]	Selecting the font size unit • Pixels for an absolute size • Percent for a relative size
TextFontStyle [▶ 199]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
TextFontWeight [▶ 199]	Selecting the font weight • Normal • Bold: Bold
TextHorizontalAlignment [▶ 199]	Selecting the horizontal alignment of the text
TextVerticalAlignment [▶ 200]	Definition of the vertical alignment of the text

Category: DropDown

Name	Description
DropDownHorizontalAlignment [▶ 197]	Selecting the horizontal alignment of the text in the drop-down list
DropDownVerticalAlignment [▶ 198]	Selecting the vertical alignment of the text in the drop-down list
DataHeight [▶ 194]	Definition of the height of the entries in the drop-down menu
DataHeightUnit [▶ 194]	Selecting the unit of the height of the list items • Pixels for an absolute height • Percent for a relative height
List Height [▶ 200]	Definition of the maximum height of the drop-down list
List Height Unit [▶ 200]	Selecting the unit of the drop-down list height
DropDownFontFamily [▶ 195]	Definition of the font of the list items via the name of the font family or the keyword
DropDownFontSize [▶ 195]	Definition of the font size of the list items
DropDownFontSizeUnit [▶ 195]	Selecting the font size unit • Pixels for an absolute size • Percent for a relative size
DropDownFontStyle [▶ 196]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
DropDownFontWeight [▶ 196]	Selecting the font weight of the text entries: • Normal • Bold: Bold
DropDownVerticalAlignment [▶ 198]	Selecting the vertical alignment of the text in the drop-down list
DropDownStyle [▶ 198]	Defines the position of the drop-down menu

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onSelectionChanged [▶ 203]	An entry has been selected.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Combo box

Available since version 1.8

See also

API reference [▶ 1343] for this control

6.2.4.1 Configuration

The combo box drop-down list can be configured in three different ways:

- Via Engineering [▶ 193]
- Via an array from the PLC [▶ 193]
  - Array of a structure representing the structure of a list entry

> The structure elements must be named identically to the attributes defined on the TwinCAT HMI side. Upper and lower case is taken into account.

```
aComboboxEntries : ARRAY [1..5] OF ST_ComboboxEntry;
TYPE ST_ComboboxEntry:
STRUCT
id : INT;
text : STRING;
value : INT;
END_STRUCT
END_TYPE
```

  - Array of a simple data type

```
aTexts : ARRAY [1..5] OF STRING;
aValues : ARRAY [1..5] OF INT;
```

The selected element of the drop-down list can be queried through various Attributes [▶ 193].

Definition via Engineering:

1. Add a combo box to the interface.
2. Change the drop-down list for the 'Src Data' attribute of the 'Data' category using the 'Combobox Src Data [▶ 193]' dialog
   1. Define the required number of entries and their sequence.
   2. For each of the three entries, assign an ID, a text that is displayed in the drop-down list and the value behind this entry.

Definition via an array from the PLC:

1. Add a combo box to the interface.
2. Link the array to the 'Src Data' attribute of the 'Data' category

To display the information of the selected element in the drop-down list, you can perform the following steps:

1. Add three text blocks.
2. Link the first text block with the attribute 'TcHmiCombobox::SelectedIndex'.
3. Link the second text block with the attribute 'TcHmiCombobox::SelectedText'.
4. Link the third text block with the attribute 'TcHmiCombobox::SelectedValue'.

Sample project: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/3922137483.zip

6.2.4.2 Dialogs

6.2.4.2.1 ComboboxSrcData dialog

This dialog can be used to define the entries of a combo box in the Engineering.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Category: General

Name	Description
ID	Definition of a freely selectable ID for the list entry. This ID must be unique within the defined entries.
Text	Definition of the text of the list entry
Value	Definition of the value of a list entry

6.2.4.3 Attributes

6.2.4.3.1 DataHeight

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDataHeight [▶ 1360]

Attribute setter: setDataHeight [▶ 1360]

Note: Available since version 1.8

6.2.4.3.2 DataHeightUnit

Only pixel is allowed here.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Combo box [▶ 185]

See also

Attribute getter: getDataHeightUnit [▶ 1361]

Attribute setter: setDataHeightUnit [▶ 1361]

Note: Available since version 1.8

6.2.4.3.3 ComboboxText

The text that is displayed if no entry is selected.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Combobox [▶ 185]

See also

Attribute getter: getComboboxText [▶ 1366]

Attribute setter: setComboboxText [▶ 1366]

Note: Available since version 1.8

6.2.4.3.4 DropDownBackgroundColor

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownBackgroundColor [▶ 1352]

Attribute setter: setDropDownBackgroundColor [▶ 1353]

Note: Available since version 1.8

6.2.4.3.5 DropDownFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownFontFamily [▶ 1375]

Attribute setter: setDropDownFontFamily [▶ 1376]

Note: Available since version 1.8

6.2.4.3.6 DropDownFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownFontSize [▶ 1376]

Attribute setter: setDropDownFontSize [▶ 1377]

Note: Available since version 1.8

6.2.4.3.7 DropDownFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownFontSizeUnit [▶ 1377]

Attribute setter: setDropDownFontSizeUnit [▶ 1377]

Note: Available since version 1.8

6.2.4.3.8 DropDownFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownFontStyle [▶ 1378]

Attribute setter: setDropDownFontStyle [▶ 1378]

Note: Available since version 1.8

6.2.4.3.9 DropDownFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownFontWeight [▶ 1374]

Attribute setter: setDropDownFontWeight [▶ 1375]

Note: Available since version 1.8

6.2.4.3.10 DropDownHighlightColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownHighlightColor [▶ 1353]

Attribute setter: setDropDownHighlightColor [▶ 1354]

Note: Available since version 1.8

6.2.4.3.11 DropDownHighlightTextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownHighlightTextColor [▶ 1355]

Attribute setter: setDropDownHighlightTextColor [▶ 1356]

Note: Available since version 1.8

6.2.4.3.12 DropDownHorizontalAlignment

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownHorizontalAlignment [▶ 1373]

Attribute setter: setDropDownHorizontalAlignment [▶ 1374]

Note: Available since version 1.8

6.2.4.3.13 DropDownTextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownTextColor [▶ 1352]

Attribute setter: setDropDownTextColor [▶ 1352]

Note: Available since version 1.8

6.2.4.3.14 DropDownToggleButtonArrowColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownToggleButtonArrowColor [▶ 1356]

Attribute setter: setDropDownToggleButtonArrowColor [▶ 1357]

Note: Available since version 1.8

6.2.4.3.15 DropDownToggleButtonBackgroundColor

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownToggleButtonBackgroundColor [▶ 1354]

Attribute setter: setDropDownToggleButtonBackgroundColor [▶ 1355]

Note: Available since version 1.8

6.2.4.3.16 DropDownVerticalAlignment

Scheme: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Combobox [▶ 185]

See also

Attribute getter: getDropDownVerticalAlignment [▶ 1372]

Attribute setter: setDropDownVerticalAlignment [▶ 1373]

Note: Available since version 1.8

6.2.4.3.17 DropDownStyle

Defines the position of the drop-down menu.

- "Desktop" is the classic representation at the position of the control.
- "Mobile" is the representation in the middle of an overlay in the screen, which is separated from the background.

Schema: tchmi:framework#/definitions/DropDownStyle [▶ 2225]

Origin: Combo box [▶ 185]

See also

Attribute getter: getDropDownStyle [▶ 1357]

Attribute setter: setDropDownStyle [▶ 1358]

Note: Available since version 1.10

6.2.4.3.18 TextFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextFontFamily [▶ 1367]

Attribute setter: setTextFontFamily [▶ 1367]

Note: Available since version 1.8

6.2.4.3.19 TextFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextFontSize [▶ 1368]

Attribute setter: setTextFontSize [▶ 1368]

Note: Available since version 1.8

6.2.4.3.20 TextFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextFontSizeUnit [▶ 1369]

Attribute setter: setTextFontSizeUnit [▶ 1369]

Note: Available since version 1.8

6.2.4.3.21 TextFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextFontStyle [▶ 1370]

Attribute setter: setTextFontStyle [▶ 1370]

Note: Available since version 1.8

6.2.4.3.22 TextFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextFontWeight [▶ 1362]

Attribute setter: setTextFontWeight [▶ 1362]

Note: Available since version 1.8

6.2.4.3.23 TextHorizontalAlignment

Horizontal alignment of the text.

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextHorizontalAlignment [▶ 1371]

Attribute setter: setTextHorizontalAlignment [▶ 1371]

Note: Available since version 1.8

6.2.4.3.24 TextVerticalAlignment

Vertical alignment of the text.

Scheme: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextVerticalAlignment [▶ 1371]

Attribute setter: setTextVerticalAlignment [▶ 1372]

Note: Available since version 1.8

6.2.4.3.25 MaxListHeight

Defines the height of the combo box.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Combo box [▶ 185]

See also

Attribute getter: getMaxListHeight [▶ 1358]

Attribute setter: setMaxListHeight [▶ 1358]

Note: Available since version 1.10

6.2.4.3.26 MaxListHeightUnit

Only pixel is allowed here.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Combo box [▶ 185]

See also

Attribute getter: getMaxListHeightUnit [▶ 1359]

Attribute setter: setMaxListHeightUnit [▶ 1359]

Note: Available since version 1.10

6.2.4.3.27 SelectedId

ID [▶ 202] of the currently selected entry. This ID must be unique within the defined entries.

Scheme: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Combobox [▶ 185]

See also

Attribute getter: getSelectedId [▶ 1364]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.4.3.28 SelectedIndex

Index [▶ 202] of the currently selected entry. The first entry in a drop-down list starts at index 0.

Scheme: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Combobox [▶ 185]

See also

Attribute getter: getSelectedIndex [▶ 1363]

Attribute setter: setSelectedIndex [▶ 1364]

Note: Available since version 1.8

6.2.4.3.29 SelectedText

Display text [▶ 202] of the currently selected entry.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Combobox [▶ 185]

See also

Attribute getter: getSelectedText [▶ 1363]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.4.3.30 SelectedValue

Value [▶ 202] of the currently selected entry.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Combobox [▶ 185]

See also

Attribute getter: getSelectedValue [▶ 1363]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.4.3.31 SrcData

Data for the options menu.

A simple list of options can be compiled in the Engineering. Each option consists of these components:

- ID [▶ 201]: Custom ID
- Text [▶ 201]: Display text
- Value [▶ 201]: Value that can be passed on later within the application.

If you use Binding [▶ 91] for this parameter, then further, complex data are supported.

- Arrays can be used directly. In this way, a string array or number array can also be linked.
- Objects are supported as a key-value list. The name of the property is interpreted as display text, the value is interpreted as a value.

Scheme: tchmi:framework#/definitions/ListItemList [▶ 2241]

Origin: Combobox [▶ 185]

See also

Attribute getter: getSrcData [▶ 1365]

Attribute setter: setSrcData [▶ 1365]

Note: Available since version 1.8

6.2.4.3.32 TextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Combobox [▶ 185]

See also

Attribute getter: getTextColor [▶ 1351]

Attribute setter: setTextColor [▶ 1351]

Note: Available since version 1.8

6.2.4.3.33 Text

The text that is displayed when text input is allowed.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Combobox [▶ 185]

See also

Attribute getter: getText

Attribute setter: setText

Note: Available since version 1.8

6.2.4.3.34 AllowTextInput

Definition of whether free text entries are allowed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Combobox [▶ 185]

See also

Attribute getter: getAllowTextInput

Attribute setter: setAllowTextInput

Note: Available since version 1.8

6.2.4.3.35 ContentPadding

Defines the distance of the text to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Combobox [▶ 185]

See also

Attribute getter: getContentPadding [▶ 1381]

Attribute setter: setContentPadding [▶ 1382]

Note: Available since version 1.10

6.2.4.4 Events

6.2.4.4.1 onSelectionChanged

A new entry has been selected.

Origin: Combobox [▶ 185]

Note: Available since version 1.8

6.2.5 Datagrid

A data grid is a table. Data sets can be displayed in this table that have been defined via Engineering or dynamically via an array.

A description of the configuration can be found in chapter Configuration [▶ 211].

Attributes

Category: Colors

Name	Description
HeaderBackgroundColor [▶ 220]	Definition of the background color of the header row
HeaderTextColor [▶ 222]	Definition of the text color of the column headers
RowNumbersBackgroundColor [▶ 223]	Definition of the background color of the column for row numbering
RowNumbersTextColor [▶ 225]	Definition of the text color of the row numbering
TableBorderColor [▶ 228]	Definition of the grid color
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Data

Name	Description
Columns [▶ 227]	Definition of column options - the assignment of options to rows is based on the order in the Column Definitions dialog [▶ 212]
SrcData [▶ 228]	Definition of the data to be displayed in the DataGrid
SelectedColumnIndex [▶ 226]	Definition of the selected column using the column index
SelectedRowIndex [▶ 227]	Definition of the selected row using the row index

Category: Grid

Name	Description
RowHeight [▶ 222]	Definition of the row height
RowHeightUnit [▶ 222]	Definition of the unit of the row height: • Pixels for an absolute height • Percent for a relative height
TableBorderWidth [▶ 228]	Definition of the line width of the grid
TableBorderWidthUnit [▶ 219]	Selecting the unit of the line width: • Pixels for an absolute width • Percent for a relative width
TableBorderStyle [▶ 228]	Selecting the style of the grid lines
CellContentPadding [▶ 218]	Definition of the distance of the text to the edge of the cell
GridFontFamily [▶ 218]	Definition of the font using the name of the font family or the keyword
GridFontSize [▶ 218]	Definition of the font size of the table contents
GridFontSizeUnit [▶ 219]	Definition of the font size unit: • Pixels for an absolute size • Percent for a relative size
GridFontStyle [▶ 219]	Selecting the font style of the table contents: • Normal • Italic: italic • Oblique: slanted
GridFontWeight [▶ 219]	Selecting the font weight of the table contents: • Normal • Bold: bold
SrcDataHasChanged [▶ 228]	Indicates whether the user has changed the data.
SelectedRowValue [▶ 227]	The value of the selected row.
SelectedCellValue [▶ 226]	The value of the selected cell.

Category: Header

Name	Description

Category: Row Numbers

Name	Description
ShowRowNumbers [▶ 227]	Selecting whether the row numbering is to be displayed
RowNumbersResizable [▶ 225]	Selecting whether the width of the columns can be changed at runtime using the separators in the column header row
RowNumbersWidth [▶ 226]	Definition of the width of the row numbering column
RowNumbersWidthUnit [▶ 226]	Selecting the unit of column width: • Pixels for an absolute width • Percent for a relative width
RowNumbersCellPadding [▶ 223]	Definition of the distance of the numbering to the edge of the cell
RowNumbersFontFamily [▶ 223]	Definition of the font using the name of the font family or the keyword
RowNumbersFontSize [▶ 223]	Definition of the font size of the row numbering
RowNumbersFontSizeUnit [▶ 224]	Definition of the font size unit: • Pixels for an absolute size • Percent for a relative size
RowNumbersFontStyle [▶ 224]	Definition of the font style of the row numbering: • Normal • Italic: italic • Oblique: slanted
RowNumbersFontWeight [▶ 224]	Definition of the font weight of the row numbering: • Normal • Bold: bold
RowNumbersHorizontalAlignment [▶ 225]	Definition of horizontal alignment of row numbering
RowNumbersVerticalAlignment [▶ 225]	Definition of vertical alignment of row numbering

Category: Table Border

Name	Description

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onDataChanged [▶ 229]	The data has been changed.
onSelectedItemChanged [▶ 229]	Another entry is now selected.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Data grid

Available since version 1.8

See also

API reference [▶ 1382] for this control

6.2.5.1 Configuration

The data grid can be configured as follows to display a one-dimensional array:

1. Add a data grid to the interface.
2. Link the one-dimensional array with the attribute "SrcData [▶ 228]" of the category "Data".
3. Change the column settings for the "SrcColumns [▶ 227]" attribute in the "Data" category using the Column Definitions dialog [▶ 212].
4. Remove the default columns.
5. Add a new column of type "SimpleColumnDefinition".

6. The column can then be configured.

For an array of a structure, you can proceed as follows:

1. Add a data grid to the interface.
2. Link the desired array with the attribute "SrcData [▶ 228]" of the category "Data".
3. Change the column settings for the "SrcColumns [▶ 227]" attribute in the "Data" category using the Column Definitions dialog [▶ 212].
4. Define the desired number of columns of type "ColumnDefinitnion" and their order.
5. Adjust the settings of each column; the "Name" attribute must correspond to the name of the structure element of the array that is to be displayed in this column.
   The definition of the "Name" attribute is case-sensitive.
   For nested structures, you can use "::" as a separator to access the sub-elements of the structure, for example, "myStruct::myStructMember".

In order for changes in the data grid to be written back to the controller, the following steps must be performed:

1. Go to the Events page of the Properties window.
2. Configure the ".onDataChanged [▶ 229]" event.
3. Add a "WriteToSymbol [▶ 53]" action.
4. Link the first parameter to your array.
5. Link the second parameter with the attribute "TcHmiDatagrid::SrcData".

Sample: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007203184343435.zip

6.2.5.2 Dialogs

6.2.5.2.1 Column Definitions dialog

This dialog can be used to customize the data grid column settings.

Properties	Settings of the column object selected under 'Elements'
Name	Definition of the name of the structure item to be displayed in this column. The value must match the name of the structure member. Upper and lower case is taken into account.
Label	Definition of the column header
Control	Selection of controls to be displayed for the cell: • TextBlock [▶ 367] • TextBox [▶ 376] • CheckBox [▶ 179] • ComboBox [▶ 185] • Image [▶ 251]
Width	Definition of the column width
Resize	Selecting whether the column width should be customizable by the user
Min width	Definition of the minimum column width
Editable	Defines whether the column can be edited. This option has no effect if control is 'TextBlock' or 'Image'.
Horizontal alignment	Selecting the horizontal alignment of the column contents

Vertical alignment	Selecting the vertical alignment of the column contents
Data definitions	Definition of the ComboBox / Image content; this attribute is available once the ComboBox or Image control type has been selected.
Format	Formatting function [▶ 97] that can convert the displayed value.

Items	List of all objects added
✕	Delete the selected object
↑ ↓	Change the object order (top = left)
Add	Add a new object

6.2.5.2.1.1 Show combo box

The data grid offers the possibility to display a column as a combo box.

1. Select the **Combobox** control type.
2. Open the **Data definitions** dialog by clicking the **...** button.

In the **Data definitions** dialog you can add as many entries as you like.

1. **Text**: Text that is displayed for the combo box entry.
2. **Value**: Value used to select the combo box entry.
3. **Add**: Adds a new combo box entry.

After configuration, the combobox automatically displays the text stored for the respective value. The data structure specifies from the outside the value used to select the entry in the combo box.

The following sample is based on the default configuration of the data grid. The code can be copied under the attribute SrcData: [▶ 228]

```
[
   {
      "Test1": "Example Data",
      "Test2": 42,
      "Test3": true,
      "Test4": "42"
   },
   {
      "Test1": "Example Data",
      "Test2": 3.1415,
      "Test3": false,
      "Test4": "43"
   }
]
```

This results in the following view:

6.2.5.2.1.2 Show image

The data grid offers the possibility to display a column as an image.

1. Select the "Image" control type.
2. Open the "Data definitions" dialog by clicking the "..." button.

In the "Data definitions" dialog you can add as many entries as you like.

1. **Text**: Name of the image. This property is used for clarity and is not displayed in the data grid.
2. **Value**: Value that controls which image is displayed.
3. **Source**: Path to the image to be displayed.
4. **Width / Height**: Width and height at which the image is to be displayed in the column.
5. **Add**: Adds a new image entry.

After configuration, the column automatically displays the image that is stored for the respective value. The data structure specifies from the outside the value used to load the different images.

The following sample is based on the default configuration of the data grid. The code can be copied under the attribute SrcData: [▶ 228]

```
[
   {
      "Test1": "Example Data",
      "Test2": 42,
      "Test3": true,
      "Test4": "42"
   },
   {
      "Test1": "Example Data",
      "Test2": 3.1415,
      "Test3": false,
      "Test4": "43"
   }
]
```

This results in the following view:

6.2.5.3 Attributes

6.2.5.3.1 CellContentPadding

Defines the distance of the cells to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getCellContentPadding [▶ 1404]

Attribute setter: setCellContentPadding [▶ 1404]

Note: Available since version 1.8

6.2.5.3.2 GridFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridFontFamily [▶ 1403]

Attribute setter: setGridFontFamily [▶ 1403]

Note: Available since version 1.8

6.2.5.3.3 GridFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridFontSize [▶ 1402]

Attribute setter: setGridFontSize [▶ 1402]

Note: Available since version 1.8

6.2.5.3.4 GridFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridFontSizeUnit [▶ 1401]

Attribute setter: setGridFontSizeUnit [▶ 1401]

Note: Available since version 1.8

6.2.5.3.5 GridFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridFontStyle [▶ 1400]

Attribute setter: setGridFontStyle [▶ 1400]

Note: Available since version 1.8

6.2.5.3.6 GridFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridFontWeight [▶ 1405]

Attribute setter: setGridFontWeight [▶ 1405]

Note: Available since version 1.8

6.2.5.3.7 GridWidthUnit

Pixels, or percent for relative positioning.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridWidthUnit [▶ 1425]

Attribute setter: setGridWidthUnit [▶ 1425]

Note: Available since version 1.8

6.2.5.3.8 HeaderBackgroundColor

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderBackgroundColor [▶ 1392]

Attribute setter: setHeaderBackgroundColor [▶ 1393]

Note: Available since version 1.8

6.2.5.3.9 HeaderCellPadding

Defines the distance between the header and the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderCellPadding [▶ 1411]

Attribute setter: setHeaderCellPadding [▶ 1412]

Note: Available since version 1.8

6.2.5.3.10 HeaderFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderFontFamily [▶ 1410]

Attribute setter: setHeaderFontFamily [▶ 1411]

Note: Available since version 1.8

6.2.5.3.11 HeaderFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderFontSize [▶ 1409]

Attribute setter: setHeaderFontSize [▶ 1410]

Note: Available since version 1.8

6.2.5.3.12 HeaderFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderFontSizeUnit [▶ 1408]

Attribute setter: setHeaderFontSizeUnit [▶ 1409]

Note: Available since version 1.8

6.2.5.3.13 HeaderFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderFontStyle [▶ 1412]

Attribute setter: setHeaderFontStyle [▶ 1412]

Note: Available since version 1.8

6.2.5.3.14 HeaderFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderFontWeight [▶ 1407]

Attribute setter: setHeaderFontWeight [▶ 1408]

Note: Available since version 1.8

6.2.5.3.15 HeaderHeight

Definition of the height of the header.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderHeight [▶ 1406]

Attribute setter: setHeaderHeight [▶ 1406]

Note: Available since version 1.8

6.2.5.3.16 HeaderHeightUnit

Defines the unit of the height of the header:

- Pixels for an absolute size
- Percent for a relative size

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderHeightUnit [▶ 1406]

Attribute setter: setHeaderHeightUnit [▶ 1407]

Note: Available since version 1.8

6.2.5.3.17 HeaderTextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getHeaderTextColor [▶ 1391]

Attribute setter: setHeaderTextColor [▶ 1392]

Note: Available since version 1.8

6.2.5.3.18 RowHeight

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowHeight [▶ 1399]

Attribute setter: setRowHeight [▶ 1399]

Note: Available since version 1.8

6.2.5.3.19 RowHeightUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowHeightUnit [▶ 1400]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.5.3.20 RowNumbersBackgroundColor

Background color of the number column.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersBackgroundColor [▶ 1390]

Attribute setter: setRowNumbersBackgroundColor [▶ 1391]

Note: Available since version 1.8

6.2.5.3.21 RowNumbersCellPadding

Defines the distance of the number column to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersCellPadding [▶ 1419]

Attribute setter: setRowNumbersCellPadding [▶ 1420]

Note: Available since version 1.8

6.2.5.3.22 RowNumbersFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersFontFamily [▶ 1420]

Attribute setter: setRowNumbersFontFamily [▶ 1421]

Note: Available since version 1.8

6.2.5.3.23 RowNumbersFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersFontSize [▶ 1421]

Attribute setter: setRowNumbersFontSize [▶ 1422]

Note: Available since version 1.8

6.2.5.3.24 RowNumbersFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersFontSizeUnit [▶ 1422]

Attribute setter: setRowNumbersFontSizeUnit [▶ 1423]

Note: Available since version 1.8

6.2.5.3.25 RowNumbersFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersFontStyle [▶ 1423]

Attribute setter: setRowNumbersFontStyle [▶ 1424]

Note: Available since version 1.8

6.2.5.3.26 RowNumbersFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersFontWeight [▶ 1415]

Attribute setter: setRowNumbersFontWeight [▶ 1415]

Note: Available since version 1.8

6.2.5.3.27 RowNumbersHorizontalAlignment

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersHorizontalAlignment [▶ 1418]

Attribute setter: setRowNumbersHorizontalAlignment [▶ 1418]

Note: Available since version 1.8

6.2.5.3.28 RowNumbersResizable

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersResizable [▶ 1414]

Attribute setter: setRowNumbersResizable [▶ 1414]

Note: Available since version 1.8

6.2.5.3.29 RowNumbersTextColor

Text color of the number column.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersTextColor [▶ 1394]

Attribute setter: setRowNumbersTextColor [▶ 1395]

Note: Available since version 1.8

6.2.5.3.30 RowNumbersVerticalAlignment

Scheme: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersVerticalAlignment [▶ 1418]

Attribute setter: setRowNumbersVerticalAlignment [▶ 1419]

Note: Available since version 1.8

6.2.5.3.31 RowNumbersWidth

Width of the number column.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersWidth [▶ 1416]

Attribute setter: setRowNumbersWidth [▶ 1416]

Note: Available since version 1.8

6.2.5.3.32 RowNumbersWidthUnit

Pixels, or percent for relative positioning.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getRowNumbersWidthUnit [▶ 1417]

Attribute setter: setRowNumbersWidthUnit [▶ 1417]

Note: Available since version 1.8

6.2.5.3.33 SelectedCellValue

Scheme: tchmi:general#/definitions/Any [▶ 2175]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSelectedCellValue [▶ 1429]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.2.5.3.34 SelectedColumnIndex

The selected column is addressed by index.

Scheme: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSelectedColumnIndex [▶ 1398]

Attribute setter: setSelectedColumnIndex [▶ 1398]

Note: Available since version 1.8

6.2.5.3.35 SelectedRowIndex

The selected row is addressed by index.

Scheme: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSelectedRowIndex [▶ 1397]

Attribute setter: setSelectedRowIndex [▶ 1397]

Note: Available since version 1.8

6.2.5.3.36 SelectedRowValue

Scheme: tchmi:general#/definitions/Any [▶ 2175]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSelectedRowValue [▶ 1429]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.2.5.3.37 ShowRowNumbers

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getShowRowNumbers [▶ 1413]

Attribute setter: setShowRowNumbers [▶ 1413]

Note: Available since version 1.8

6.2.5.3.38 SrcColumn

Scheme: tchmi:framework#/definitions/DGColumnDefinitionList [▶ 2218]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSrcColumn [▶ 1395]

Attribute setter: setSrcColumn [▶ 1395]

Note: Available since version 1.8

6.2.5.3.39 SrcData

Scheme: tchmi:general#/definitions/Array [▶ 2175]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSrcData [▶ 1396]

Attribute setter: setSrcData [▶ 1396]

Note: Available since version 1.8

6.2.5.3.40 SrcDataHasChanged

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getSrcDataHasChanged [▶ 1428]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.2.5.3.41 TableBorderColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getTableBorderColor [▶ 1393]

Attribute setter: setTableBorderColor [▶ 1394]

Note: Available since version 1.8

6.2.5.3.42 TableBorderStyle

Scheme: tchmi:framework#/definitions/BorderStyle [▶ 2196]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getGridStyle [▶ 1426]

Attribute setter: setGridStyle [▶ 1426]

Note: Available since version 1.8

6.2.5.3.43 TableBorderWidth

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Datagrid [▶ 203]

See also

Attribute getter: getTableBorderWidth [▶ 1424]

Attribute setter: setTableBorderWidth [▶ 1424]

Note: Available since version 1.8

6.2.5.4 Events

6.2.5.4.1 onDataChanged

The data [▶ 228] has been changed.

Origin: Datagrid [▶ 203]

Note: Available since version 1.8

6.2.5.4.2 onSelectedItemChanged

Another entry is now selected.

Origin: Datagrid [▶ 203]

Note: Available since version 1.8

6.2.6 Ellipse

An ellipse control is an ellipse object.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Appearance

Name	Description
FillColor [▶ 234]	Definition of the fill color of the control
StrokeColor [▶ 234]	Definition of the line color of the control
StrokeStyle [▶ 234]	Selecting the style of the ellipse line
StrokeThickness [▶ 235]	Definition of the line width of the ellipse
StrokeThicknessUnit [▶ 235]	Selecting the unit of the line width: • Pixels for an absolute width

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Ellipse

Note: Available since version 1.8

See also

API reference [▶ 1429] for this control

6.2.6.1 Attributes

6.2.6.1.1 FillColor

Defines the fill color of the control.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Ellipse [▶ 229]

See also

Attribute getter: getFillColor [▶ 1434]

Attribute setter: setFillColor [▶ 1435]

Note: Available since version 1.8

6.2.6.1.2 StrokeColor

Defines the stroke color of the control.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Ellipse [▶ 229]

See also

Attribute getter: getStrokeColor [▶ 1435]

Attribute setter: setStrokeColor [▶ 1436]

Note: Available since version 1.8

6.2.6.1.3 StrokeStyle

Definition of the stroke style.

Comma or space-separated list of numbers that specify the length of strokes and spaces.

If an odd number is specified, the list is repeated once. Example: "5, 3, 2" is equivalent to "5, 3, 2, 5, 3, 2".

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Ellipse [▶ 229]

See also

Attribute getter: getStrokeStyle [▶ 1438]

Attribute setter: setStrokeStyle [▶ 1438]

Note: Available since version 1.8

6.2.6.1.4 StrokeThickness

Defines the stroke thickness of the control in pixels.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Ellipse [▶ 229]

See also

Attribute getter: getStrokeThickness [▶ 1436]

Attribute setter: setStrokeThickness [▶ 1437]

Note: Available since version 1.8

6.2.6.1.5 StrokeThicknessUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Ellipse [▶ 229]

See also

Attribute getter: getStrokeThicknessUnit [▶ 1437]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.7 Event Grid

The Event Grid is provided for the tabular display of alarms and messages.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Filter [▶ 243]	Defines which events are queried by the server
Columns [▶ 243]	The columns to be displayed in the EventGrid
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Menubar

Name	Description
MenuBarPosition [▶ 244]	Position of the menu bar
ShowMenuBar [▶ 244]	Whether the menu bar is to be displayed

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.
configure [▶ 244]	Determines whether the column configuration in the event grid is customizable.
viewDetails [▶ 244]	Determines whether the display of event details can be enabled in the event grid.

Inheritance hierarchy

Control [▶ 433]

Event Grid

Available since version 1.10

See also

API reference [▶ 1439] for this control

6.2.7.1 Configuration

The TwinCAT 3 EventLogger can be used to send events. The display of the events can be defined via the filters [▶ 240]. The columns [▶ 242] can be configured via the Properties. Further information about the EventGrid can be found in the section Event System [▶ 552].

Sample: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007204494484107.zip

6.2.7.2 Dialogs

6.2.7.2.1 Filter

You can configure which events are to be displayed via the filter editor. The filters are defined via the properties of the Event Grid.

Structure of the Filter Editor

1. **Add clause**: Add a condition
2. **Delete clause**: Delete a condition
3. **And/Or**: Logical combination of conditions
4. **Path**: Filter options:

- **domain**: Server extension which delivered the alarm (e.g. TcHmiEventLogger).
- **sourceDomain**: Server extension that triggered the alarm. The sourceDomain may differ from the domain if a server extension forwards the alarms (e.g. TcHmiSqliteLogger).
- **name**: Name of the event.
- **text:** Text of the event.
- **Id**: ID of the event.
- **timeRaised**: Time at which the event was triggered.
- **timeCleared**: Time at which an alarm was reset.
- **timeConfirmed**: Time at which an alarm was confirmed.
- **type**: Type of event (alarm or message).
- **severity**: Severity of the event (prohibitions, info, warning, error, critical error).
- **alarmState**: State of an alarm (triggered, confirmed, deleted, confirmed and deleted).
- **confirmationState**: State of acknowledgement of an alarm (not required, not supported, waiting for acknowledgement, acknowledged, reset).
- **"params**::": The parameter object of an event can be accessed in the text field of the path via the keyword "params::" (without quotation marks). The parameter object contains optional parameters that can differ depending on the event source. For example, the TwinCAT Eventlogger provides additional information about the event origin in the PLC. The parameters are accessed in camelCase, e.g.: params::sourceName

5. **Operator**: Comparison operator of the condition
6. **Value**: Value of the condition

Grouping of conditions

Several conditions can be grouped and linked if more than one condition is selected. The conditions are selected by clicking the rows with the CTRL key pressed.

1. **Group clause**: Grouping of several conditions. The brackets link them optically.
2. **Ungroup clause**: Cancel existing groupings.

6.2.7.2.2 Columns

The columns to be displayed in the Event Grid can be configured in the columns editor. The columns can also be configured in the client at runtime.

An entry is added for each column. The column name is specified via the combo box, where you can select a column from all available columns [▶ 243].

The Columns Editor consists of the following areas:

1. **Type**: The type of column indicates whether it is a normal "Column" or a "CustomColumn". The selection of the column type can be made via the lower combo box (1). The parameters of the parameter object of an event can be displayed in a custom column.
2. **ColumnName**: The name of the column to be displayed.
   For a column of type "Column", the names can be selected via the combo box.
   For a column of type "CustomColumn" the parameters are displayed with the keyword "params::" (without quotation marks) and the name of the parameter in camelCase.
   Using the example of the EventLogger, the "SourceName" can be displayed in a column, for example:

The parameter object can differ for different event sources. If an event does not have the parameter in the parameter object configured in the column, an empty entry is displayed in the corresponding cell.

3. **Width:** The width of the column. If no value is specified, the width is set automatically.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
Add	Add a new object

6.2.7.3 Attributes

6.2.7.3.1 Filter

Defines which events are queried by the server.

Scheme: tchmi:server#/definitions/eventFilter [▶ 2273]

Origin: Event Grid [▶ 235]

See also

Attribute getter: getFilter [▶ 1445]

Attribute setter: setFilter [▶ 1445]

Note: Available since version 1.10

6.2.7.3.2 Columns

The columns to be displayed in the Event Grid.

Scheme: tchmi:framework#/definitions/EventGridColumnList [▶ 2226]

Origin: Event Grid [▶ 235]

See also

Attribute getter: getColumns [▶ 1445]

Attribute setter: setColumns [▶ 1446]

Note: Available since version 1.10

6.2.7.3.3 MenuBarPosition

The position of the menu bar.

Scheme: tchmi:framework#/definitions/EventGridMenuBarPosition [▶ 2226]

Origin: Event Grid [▶ 235]

See also

Attribute getter: getMenuBarPosition [▶ 1446]

Attribute setter: setMenuBarPosition [▶ 1447]

Note: Available since version 1.10

6.2.7.3.4 ShowMenuBar

Whether the menu bar is to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Event Grid [▶ 235]

See also

Attribute getter: getShowMenuBar [▶ 1447]

Attribute setter: setShowMenuBar [▶ 1448]

Note: Available since version 1.10

6.2.7.4 Permissions

6.2.7.4.1 configure

Determines whether the column configuration in the event grid is customizable.

Origin: Event Grid [▶ 235]

See also

See also User management at control level [▶ 527].

Note: Available since version 1.10

6.2.7.4.2 viewDetails

Determines whether the display of event details can be enabled in the event grid.

Origin: Event Grid [▶ 235]

See also

See also User management at control level [▶ 527].

Note: Available since version 1.10

6.2.7.5 Menu bar

The Event Grid has a menu bar that provides filter and configuration options at runtime in the client. The menu bar can be shown and hidden and placed at the edge of the Event Grid:

1. **ShowMenuBar**: Show or hide menu bar.
2. **MenuBarPosition**: Position of the menu bar.

The menu bar has the following buttons:

1. Show or hide all alarms.
2. Show or hide all messages.
3. Show or hide verbose events.
4. Show or hide information events.
5. Show or hide warnings.
6. Show or hide errors.
7. Show or hide critical errors.
8. Call the column configuration. The column configuration only applies to the client.
9. Confirm alarm.
10. Confirm all unconfirmed alarms.

6.2.8 IFrame

An IFrame is a nested browser context. It can be used to include another HTML page in the visualization or to display a PDF document. The attribute 'Src [▶ 249]' is used to define the HTML page or the PDF.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Src [▶ 251]	Definition of the URL of the web page to be embedded or the path of a PDF to be displayed Example of URL: https://www.beckhoff.com Example of PDF: <Path within the TC HMI project>/Test.pdf#page=10
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Control

Name	Description
onLoad [▶ 251]	The IFrame was loaded successfully.
onError [▶ 251]	The IFrame was not loaded successfully.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

IFrame

Note: Available since version 1.8

See also

API reference [▶ 1449] for this control

6.2.8.1 Attributes

6.2.8.1.1 Src

The URL of the website to embed.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: IFrame [▶ 246]

See also

Attribute getter: getSrc [▶ 1454]

Attribute setter: setSrc [▶ 1455]

Note: Available since version 1.8

6.2.8.2 Events

6.2.8.2.1 onError

The IFrame was not loaded successfully.

Origin: IFrame [▶ 246]

Note: Available since version 1.8

6.2.8.2.2 onLoad

The IFrame was loaded successfully.

Origin: IFrame [▶ 246]

Note: Available since version 1.8

6.2.9 Image

An image is an image display in which image files from the project path can be displayed. The supported image formats are defined by the browser used.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 257]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side. The 'Width' attribute is ignored in all calculations! • Content: The width is calculated from the height. The aspect ratio of the image is maintained. If 'Width Mode' and 'Height Mode' are set to Content, the image is drawn in its original size.
HeightMode [▶ 256]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side. The 'Height' attribute is ignored in all calculations!

Name	Description
	• Content: The height is calculated from the width while maintaining the aspect ratio of the image. If 'Width Mode' and 'Height Mode' are set to Content, the image is drawn in its original size.
OriginalWidth [▶ 258]	Original width of the image file.
OriginalHeight [▶ 257]	Original height of the image file.

Category: Common

Name	Description
Src [▶ 256]	Definition of the image path within the TC HMI project
Alt [▶ 256]	Definition of an alternative text that is displayed if the image cannot or should not be loaded
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Control

Name	Description
onLoad [▶ 258]	The image was loaded successfully.
onError [▶ 258]	The image was not loaded successfully.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Image

Note: Available since version 1.8

See also

API reference [▶ 1455] for this control

6.2.9.1 Attributes

6.2.9.1.1 Src

The address of the image. Images on other servers are also possible.

Scheme: tchmi:framework#/definitions/Path [▶ 2248]

Origin: Image [▶ 251]

See also

Attribute getter: getSrc [▶ 1462]

Attribute setter: setSrc [▶ 1463]

Note: Available since version 1.8

6.2.9.1.2 Alt

Alternative text that is displayed if the image cannot or should not be loaded.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Image [▶ 251]

See also

Attribute getter: getAlt [▶ 1463]

Attribute setter: setAlt [▶ 1464]

Note: Available since version 1.8

6.2.9.1.3 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.

- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!
- **Content:** The height is calculated from the width while maintaining the aspect ratio of the image. If Width Mode [▶ 257] and *Height Mode* are set to Content, the image is drawn in its original size.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Scheme: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Image [▶ 251]

See also

Attribute getter: getHeightMode [▶ 1460]

Attribute setter: setHeightMode [▶ 1461]

Note: Available since version 1.8

6.2.9.1.4 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!
- **Content:** The width is calculated from the height while maintaining the aspect ratio of the image. If *Width Mode* and Height Mode [▶ 256] are set to Content, the image is drawn in its original size.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Scheme: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Image [▶ 251]

See also

Attribute getter: getWidthMode [▶ 1461]

Attribute setter: setWidthMode [▶ 1462]

Note: Available since version 1.8

6.2.9.1.5 OriginalHeight

Original height of the image file.

Returns 0 if no image is available.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Image [▶ 251]

See also

Attribute getter: getOriginalHeight [▶ 1464]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.9.1.6 OriginalWidth

Original width of the image file.

Returns 0 if no image is available.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Image [▶ 251]

See also

Attribute getter: getOriginalWidth [▶ 1465]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.9.2 Events

6.2.9.2.1 onError

The image was not loaded successfully.

Origin: Image [▶ 251]

Note: Available since version 1.8

6.2.9.2.2 onLoad

The image was loaded successfully.

Origin: Image [▶ 251]

Note: Available since version 1.8

6.2.10 Keyboard

A keyboard is an on-screen keyboard that can be used for input without a hardware keyboard. The 'Layout [▶ 262]' attribute can be used to select different layouts. Various German and American layouts are available, as well as a numpad and a calculator.

In the layouts whose names end with '_compact', a window in which additional keys are offered can be called up by pressing a few keys for a prolonged period. Example: Prolonged pressing of the 's' key opens a window with the additional key 'ß'.

Attributes

Category: Colors

Name	Description
LabelColor [▶ 268]	Definition of the text color of the keys
LabelStrokeColor [▶ 268]	Definition of the outline color of the graphic key characters
LabelFillColor [▶ 268]	Definition of the fill color of the graphic key characters
KeyBackgroundColor [▶ 267]	Definition of the background color of the keys
AdditionalKeyBackdropColor [▶ 265]	Definition of the background color of the window in which the additional characters for the compact layouts are displayed
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
KeyboardPadding [▶ 268]	Definition of the inner distance of the keypad to the outer sides of the keyboard
AdditionalKeyBackdropPadding [▶ 265]	Definition of the inner distance of the additional keys to the outer sides of the window in which they are displayed
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
LayoutFile [▶ 269]	Activates a layout from the project.
Layout [▶ 269]	Selection of the keyboard layout
ScaleMode [▶ 270]	Selecting the scaling of the keys
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Behavior

Name	Description
FixedTarget [▶ 265]	Definition of a control using its name, to which all input is sent regardless of the focus The name of the control must be entered as a simple string. The control must not be linked directly.
ReleaseStickyKeysManually [▶ 270]	Selecting whether the Shift key must be deactivated manually after it has been used together with another key
LongpressDelay [▶ 269]	Definition of the time in milliseconds that a key must be pressed for a long keystroke to be detected.
RepetitionDelay [▶ 270]	Definition of the time in milliseconds that a key must be pressed for the input to be repeated.
PressedKeys [▶ 271]	

Category: Labels

Name	Description
LabelStrokeThickness [▶ 267]	Definition of the stroke thickness of the graphic key characters
LabelFontFamily [▶ 265]	Definition of the font using the name of the font family or the keyword
LabelFontSize [▶ 266]	Definition of the font size of the key text
LabelFontSizeUnit [▶ 266]	Selecting the unit of the text size: • Pixels for an absolute size • Percent for a relative size
LabelFontStyle [▶ 266]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
LabelFontWeight [▶ 266]	Selecting the font weight: • Normal • Bold: bold

Events

Category: Control

Name	Description
onKeyPressStarted [▶ 271]	Interaction with a key was started.
onKeyPressFinished [▶ 271]	Interaction with a key was finished.
onKeyPressCanceled [▶ 271]	Interaction with a key was cancelled.
onKeyPressEnded [▶ 271]	Interaction with a key was canceled or finished.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Keyboard

Note: Available since version 1.8

See also

API reference [▶ 1465] for this control

6.2.10.1 Attributes

6.2.10.1.1 AdditionalKeyBackdropColor

The background color of the additional characters.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getAdditionalKeyBackdropColor [▶ 1475]

Attribute setter: setAdditionalKeyBackdropColor [▶ 1475]

Note: Available since version 1.8

6.2.10.1.2 AdditionalKeyBackdropPadding

Inner distance of the additional keys.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getAdditionalKeyBackdropPadding [▶ 1478]

Attribute setter: setAdditionalKeyBackdropPadding [▶ 1479]

Note: Available since version 1.8

6.2.10.1.3 FixedTarget

The name of the control to which the characters are sent, regardless of the focus.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getFixedTarget [▶ 1482]

Attribute setter: setFixedTarget [▶ 1482]

Note: Available since version 1.8

6.2.10.1.4 LabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelFontFamily [▶ 1489]

Attribute setter: setLabelFontFamily [▶ 1490]

Note: Available since version 1.8

6.2.10.1.5 LabelFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelFontSize [▶ 1487]

Attribute setter: setLabelFontSize [▶ 1488]

Note: Available since version 1.8

6.2.10.1.6 LabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelFontSizeUnit [▶ 1488]

Attribute setter: setLabelFontSizeUnit [▶ 1489]

Note: Available since version 1.8

6.2.10.1.7 LabelFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelFontStyle [▶ 1490]

Attribute setter: setLabelFontStyle [▶ 1491]

Note: Available since version 1.8

6.2.10.1.8 LabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelFontWeight [▶ 1487]

Attribute setter: setLabelFontWeight [▶ 1487]

Note: Available since version 1.8

6.2.10.1.9 LabelStrokeThickness

Definition of the stroke thickness of the graphic key characters

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelStrokeThickness [▶ 1486]

Attribute setter: setLabelStrokeThickness [▶ 1486]

Note: Available since version 1.8

6.2.10.1.10 KeyBackgroundColor

The background color of the keys.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getKeyBackgroundColor [▶ 1474]

Attribute setter: setKeyBackgroundColor [▶ 1474]

Note: Available since version 1.8

6.2.10.1.11 KeyPadding

Inner distance in the individual keys.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getKeyPadding [▶ 1477]

Attribute setter: setKeyPadding [▶ 1478]

Note: Available since version 1.8

6.2.10.1.12 KeyboardPadding

Inner distance of the whole keyboard.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getKeyboardPadding [▶ 1476]

Attribute setter: setKeyboardPadding [▶ 1477]

Note: Available since version 1.8

6.2.10.1.13 LabelColor

The color of the texts.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelColor [▶ 1473]

Attribute setter: setLabelColor [▶ 1473]

Note: Available since version 1.8

6.2.10.1.14 LabelFillColor

Definition of the fill color of the graphic key characters

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelFillColor [▶ 1475]

Attribute setter: setLabelFillColor [▶ 1476]

Note: Available since version 1.8

6.2.10.1.15 LabelStrokeColor

Definition of the outline color of the graphic key characters.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLabelStrokeColor [▶ 1472]

Attribute setter: setLabelStrokeColor [▶ 1472]

Note: Available since version 1.8

6.2.10.1.16 LayoutFile

Activates a layout from the project. Overwrites the setting in the Layout [▶ 269] attribute.

Schema: tchmi:framework#/definitions/Path [▶ 2248]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLayoutFile [▶ 1479]

Attribute setter: setLayoutFile [▶ 1480]

Note: Available since version 1.10

6.2.10.1.17 Layout

Defines the layout of the keyboard. Can be overwritten via LayoutFile. [▶ 269]

- EN_US-QWERTY_standard: American layout
- EN_US-QWERTY_full: American layout with numpad
- EN_US-QWERTY_compact: Compact American layout
- DE_DE-QWERTZ_standard: German layout
- DE_DE-QWERTZ_full: German layout with numpad
- DE_DE-QWERTZ_compact: Compact German layout
- PINpad: Layout of a PIN entry
- Numpad: Layout of a numpad
- Calculator: Layout of a calculator

Schema: tchmi:framework#/definitions/ControlKeyboardLayouts [▶ 2227]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLayout [▶ 1480]

Attribute setter: setLayout [▶ 1481]

Note: Available since version 1.8

Was LayoutPath before version 1.10

6.2.10.1.18 LongpressDelay

Definition of the time in milliseconds that a key must be pressed in order for a long keystroke to be detected and a window with the additional keys to be displayed. This function is only available in the '_compact' layouts and for certain keys.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getLongpressDelay [▶ 1484]

Attribute setter: setLongpressDelay [▶ 1484]

Note: Available since version 1.8

6.2.10.1.19 ReleaseStickyKeysManually

Selecting whether the Shift key must be deactivated manually after it has been used together with another key.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getReleaseStickyKeysManually [▶ 1483]

Attribute setter: setReleaseStickyKeysManually [▶ 1483]

Note: Available since version 1.8

6.2.10.1.20 RepetitionDelay

Definition of the time in milliseconds that a key must be pressed for the input to be repeated. This function is only available for certain keys, e.g. the Delete key.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getRepetitionDelay [▶ 1485]

Attribute setter: setRepetitionDelay [▶ 1485]

Note: Available since version 1.8

6.2.10.1.21 ScaleMode

Selecting the scaling of the keys:

- None: No scaling of the keys
- ScaleToFill: The entire control is completely filled in. The proportions of the keys are not taken into account.
- ScaleToFit: The keys are displayed as large as possible without exceeding the edges of the control. The proportions of the keys are maintained.
- ScaleToFitWidth: The keys are displayed across the entire width of the control, regardless of whether they exceed the limits of the control.
  The proportions of the keys are maintained.
- ScaletoFitHeight: The keys are displayed across the entire height of the control, regardless of whether they exceed the limits of the control.
  The proportions of the keys are maintained.

Scheme: tchmi:framework#/definitions/ScaleMode [▶ 2251]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getScaleMode [▶ 1481]

Attribute setter: setScaleMode [▶ 1481]

Note: Available since version 1.8

6.2.10.1.22 PressedKeys

Scheme: tchmi:framework#/definitions/KeyList [▶ 2238]

Origin: Keyboard [▶ 258]

See also

Attribute getter: getPressedKeys [▶ 1491]

Note: Available since version 1.8

6.2.10.2 Events

6.2.10.2.1 onKeyPressStarted

The user has pressed any key on the keyboard.

Origin: Keyboard [▶ 258]

Note: Available since version 1.8

6.2.10.2.2 onKeyPressCanceled

The user has canceled the interaction with any key on the keyboard. This can be triggered, for example, by moving to the side.

Origin: Keyboard [▶ 258]

Note: Available since version 1.8

6.2.10.2.3 onKeyPressFinished

The user has completed the interaction with any key on the keyboard. In other words, the user has released a key that was previously pressed.

Origin: Keyboard [▶ 258]

Note: Available since version 1.8

6.2.10.2.4 onKeyPressEnded

The user has canceled or finished the interaction with any key on the keyboard.

Origin: Keyboard [▶ 258]

Note: Available since version 1.8

6.2.11 Line

A line control represents a straight line.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
X1 [▶ 279]	Definition of the x-axis coordinate of the starting point of the line
X1Unit [▶ 279]	Selecting the unit of the x-axis coordinate of the starting point: • Pixels for absolute positioning
X2 [▶ 280]	Definition of the x-axis coordinate of the end point of the line
X2Unit [▶ 280]	Selecting the unit of the x-axis coordinate of the end point: • Pixels for absolute positioning
Y1 [▶ 280]	Definition of the y-axis coordinate of the starting point of the line
Y1Unit [▶ 280]	Selecting the unit of the y-axis coordinate of the starting point: • Pixels for absolute positioning
Y2 [▶ 281]	Definition of the y-axis coordinate of the end point of the line
Y2Unit [▶ 281]	Selecting the unit of the y-axis coordinate of the end point: • Pixels for absolute positioning
ScaleMode [▶ 278]	Selecting the scaling of the line: • None: No scaling of the line • ScaleToFill: The line is displayed on the entire size of the control. The proportions of the line are not taken into account. • ScaleToFit: The line is displayed as large as possible without exceeding the edges of the control. The proportions of the line are maintained. • ScaleToFitWidth: The line is displayed over the entire width of the control, regardless of whether it exceeds the limits of the control. The proportions of the line are maintained. • ScaleToFitHeight: The line is displayed over the entire height of the control, regardless of whether it exceeds the limits of the control. The proportions of the line are maintained.
StrokeColor [▶ 278]	Definition of the line color
StrokeStyle [▶ 278]	Line style definition – comma and space separated list of numbers, which specify the length of the line and distances. Example: 10, 5, 20, 30 corresponds to a sequence of line length 10, distance 5, line length 20, distance 30. This sequence is repeated over the entire length of the line. If an odd number is specified, the list is repeated once. Example: 20, 5, 10 becomes 20, 5, 10, 20, 5, 10
StrokeThickness [▶ 279]	Definition of the line width
StrokeThicknessUnit [▶ 279]	Selecting the unit of the line width: • Pixels for an absolute width
IsEnabled [▶ 439]	Selection of whether a control should be activated

Name	Description
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Shape

Name	Description

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Line

Note: Available since version 1.8

See also

API reference [▶ 1492] for this control

6.2.11.1 Attributes

6.2.11.1.1 ScaleMode

Scales the content to the control size.

Scheme: tchmi:framework#/definitions/ScaleMode [▶ 2251]

Origin: Line [▶ 272]

See also

Attribute getter: getScaleMode [▶ 1504]

Attribute setter: setScaleMode [▶ 1504]

Note: Available since version 1.8

6.2.11.1.2 StrokeColor

Defines the stroke color of the control.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line [▶ 272]

See also

Attribute getter: getStrokeColor [▶ 1498]

Attribute setter: setStrokeColor [▶ 1498]

Note: Available since version 1.8

6.2.11.1.3 StrokeStyle

Definition of the stroke style. Comma or space-separated list of numbers that specify the length of strokes and spaces. If an odd number is specified, the list is repeated once. Example: "5, 3, 2" is equivalent to "5, 3, 2, 5, 3, 2".

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Line [▶ 272]

See also

Attribute getter: getStrokeStyle [▶ 1506]

Attribute setter: setStrokeStyle [▶ 1507]

Note: Available since version 1.8

6.2.11.1.4 StrokeThickness

Defines the stroke thickness of the control in pixels.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line [▶ 272]

See also

Attribute getter: getStrokeThickness [▶ 1505]

Attribute setter: setStrokeThickness [▶ 1505]

Note: Available since version 1.8

6.2.11.1.5 StrokeThicknessUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Line [▶ 272]

See also

Attribute getter: getStrokeThicknessUnit [▶ 1506]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.11.1.6 X1

x-coordinate of the first point.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line [▶ 272]

See also

Attribute getter: getX1 [▶ 1500]

Attribute setter: setX1 [▶ 1500]

Note: Available since version 1.8

6.2.11.1.7 X1Unit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Line [▶ 272]

See also

Attribute getter: getX1Unit [▶ 1501]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.11.1.8 X2

x-coordinate of the second point.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line [▶ 272]

See also

Attribute getter: getX2 [▶ 1501]

Attribute setter: setX2 [▶ 1502]

Note: Available since version 1.8

6.2.11.1.9 X2Unit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Line [▶ 272]

See also

Attribute getter: getX2Unit [▶ 1502]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.11.1.10 Y1

y-coordinate of the first point.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line [▶ 272]

See also

Attribute getter: getY1 [▶ 1499]

Attribute setter: setY1 [▶ 1499]

Note: Available since version 1.8

6.2.11.1.11 Y1Unit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Line [▶ 272]

See also

Attribute getter: getY1Unit [▶ 1500]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.11.1.12 Y2

y-coordinate of the second point.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line [▶ 272]

See also

Attribute getter: getY2 [▶ 1503]

Attribute setter: setY2 [▶ 1503]

Note: Available since version 1.8

6.2.11.1.13 Y2Unit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Line [▶ 272]

See also

Attribute getter: getY2Unit [▶ 1504]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.12 Line Chart

A LineChart is a diagram in which curves can be displayed. It can be statically filled with data or dynamically supplied with data via an array.

A description of the configuration can be found in chapter Configuration [▶ 289].

Attributes

Category: Colors

Name	Description
XAxisColor [▶ 298]	Definition of the color of the x-axis
GridBackgroundColor [▶ 291]	Definition of the background color of the coordinate system
GridLineColor [▶ 291]	Definition of the color of the coordinate grid
SubgridLineColor [▶ 294]	Definition of the color of the coordinate subgrid
SectionsBackgroundColor [▶ 292]	Definition of the background color of the control
XLabelFontColor [▶ 301]	Definition of the font color of the x-axis labels
XAxisNameFontColor [▶ 299]	Definition of the font color of the x-axis name
TooltipFontColor [▶ 295]	Definition of the font color of the tooltip
TooltipBackgroundColor [▶ 295]	Definition of the background color of the tooltip
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
LineGraphData [▶ 292]	Definition of the data to be displayed as graph(s)
LineGraphDescriptions [▶ 292]	Definition of graph descriptions for each of the graphs defined by the array
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: XAxis

Name	Description
ShowXAxis [▶ 293]	Selecting whether the x-axis is to be displayed
ShowXAxisLabels [▶ 303]	Selecting whether the x-axis labels are to be displayed
ShowXAxisName [▶ 293]	Selecting whether the x-axis name is to be displayed
ShowXSubTicks [▶ 293]	Selecting whether the subcoordinate points are to be displayed on the x-axis
XAxisAutoScaling [▶ 297]	Selecting whether the x-axis should be automatically scaled according to the data sets. 'Maintick Min Value', 'Maintick Max Value' and 'X-Axis Maintick Steps' need not be set.
XAxisDecimalPlaces [▶ 298]	Definition of the decimal places displayed for the x-axis
XAxisName [▶ 299]	Definition of the x-axis name
XAxisNameFontFamily [▶ 300]	Definition of the font using the name of the font family or the keyword
XAxisNameFontSize [▶ 300]	Definition of the font size of the x-axis name
XAxisNameFontSizeUnit [▶ 300]	Selecting the unit of the font size of the x-axis name: • Pixels for an absolute size • Percent for a relative size
XAxisNameFontWeight [▶ 301]	Selecting of the font weight of the x-axis name: • Normal • Bold: bold
XAxisUnit [▶ 301]	Definition of the unit of the x-axis
XAxisWidth [▶ 301]	Definition of the x-axis width
XLabelFontFamily [▶ 302]	Definition of the font using the name of the font family or the keyword
XLabelFontSize [▶ 302]	Definition of the font size for the x-axis labels
XLabelFontSizeUnit [▶ 302]	Selecting the font size unit for the x-axis labels: • Pixels for an absolute size • Percent for a relative size
XLabelFontWeight [▶ 302]	Selecting the font weight of the x-axis labels: • Normal • Bold: bold
XMainTickMaxValue [▶ 303]	Definition of the maximum value of the x-axis main subdivision
XMainTickMinValue [▶ 303]	Definition of the minimum value of the x-axis main subdivision
XMainTickSteps [▶ 303]	Definition of the step size of the x-axis main subdivision
XSubTickSteps [▶ 304]	Definition of the step size of the x-axis subdivision
XAxisFormat [▶ 298]	Definition of the formatting of the x-axis labels

Category: Grid

Name	Description
ShowGrid [▶ 292]	Selecting whether the coordinate grid is to be displayed
GridLineWidth [▶ 291]	Definition of the line width of the coordinate grid
GridShowHorizontalLines [▶ 291]	Selecting whether the horizontal lines of the coordinate grid are to be displayed
GridShowVerticalLines [▶ 292]	Selecting whether the vertical lines of the coordinate grid are to be displayed

Category: YAxis

Name	Description
YAxis [▶ 304]	Definition of one or more y-axes
ShowYSubTicks [▶ 294]	Selecting whether the subcoordinate points are to be displayed on the y-axis
YAxisWidth [▶ 304]	Definition of the y-axis width
YLabelFontFamily [▶ 304]	Definition of the font using the name of the font family or the keyword
YLabelFontSize [▶ 305]	Definition of the font size for the y-axis labels
YLabelFontSizeUnit [▶ 305]	Selecting the font size unit for the y-axis labels: • Pixels for an absolute size • Percent for a relative size
YLabelFontWeight [▶ 305]	Selecting the font weight of the y-axis labels: • Normal • Bold: bold
YMainTickSteps [▶ 305]	Definition of the step size of the y-axis main subdivision
YSubTickSteps [▶ 306]	Definition of the step size of the y-axis subdivision

Category: Subgrid

Name	Description
SubgridLineWidth [▶ 294]	Definition of the line width of the coordinate subgrid
SubgridShowHorizontalLines [▶ 294]	Selecting whether the horizontal lines of the coordinate subgrid are to be displayed
SubgridShowVerticalLines [▶ 295]	Selecting whether the vertical lines of the coordinate subgrid are to be displayed
ShowTooltip [▶ 408]	Selecting whether the tooltips are to be displayed
TooltipFormat [▶ 412]	• Definition of the formatting of the tooltips.
TooltipFontFamily [▶ 411]	Definition of the font using the name of the font family or the keyword
TooltipFontSize [▶ 411]	Definition of the font size
TooltipFontWeight [▶ 412]	Selecting the font weight: • Normal • Bold: Bold

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Line Chart

Note: Available since version 1.8

See also

API reference [▶ 1557] for this control

6.2.12.1 Configuration

A LineChart can be configured as follows:

1. Define the y-axes of the coordinate system using the"YAxis Definitions [▶ 290]" dialog

1.1. Define the desired number of y-axes and their sequence.

1.2. Adjust the settings of the y-axis/axes if necessary.

2. Link the array containing the data points to the attribute "Line Graph Data" of the category "Common".

> The array must have two dimensions. This also applies if only one graph is stored in this array. In the latter case the array should be declared as array[1..1, 1..100] OF ST_Point.

```
aPoints : ARRAY[1..2, 1..100] OF ST_Point;
TYPE ST_Point :
    STRUCT
        x : INT;
        y : INT;
    END_STRUCT
END_TYPE
```

> The structure elements must be named identically to the attributes defined on the TwinCAT HMI side. Upper and lower case is taken into account.

3. Add a graph description for each of the graphs contained in the array to the "Line Graph Descriptions" attribute of the "Common" category using the "LineGraphDescription Definition [▶ 289]" editor.

Example: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007203177442059.zip

6.2.12.2 Dialogs

6.2.12.2.1 LineGraphDescription definition dialog

This dialog can be used to change the settings of the graphs for the individual subarrays to be displayed in the LineChart.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Properties	Settings for the y-axis assignment selected under "Elements"
Y-axis Id	Selecting the y-axis to which the data set is to be connected
Line width	Definition of the line width of the graph
Point dot	Selecting whether the individual data points are to be marked
Point dot radius	Definition of the radius of the data points
Point dot stroke width	Definition of the width of the dotted line

6.2.12.2.2 YAxis definition dialog

This dialog can be used to add and parameterize the different y-axes.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Properties	Settings of the y-axis definition selected under "Elements"
ID	Definition of a unique ID of the y-axis
Position	Position of the y-axis on the graph • Left side of the graph • Right side of the graph
Main-tick-min-value	Definition of the beginning of the displayed value range
Main-tick-max-value	Definition of the end of the displayed value range
Auto scaling	Selecting whether the value range should be automatically adjusted according to the values to be displayed in the array
Decimal places	Definition of decimal places
Show labels	Selecting whether the label of the y-axis is to be displayed
Show axis name	Selecting whether the axis name should be displayed next to the axis in the graph
Axis name	Definition of the axis name, which can optionally be displayed next to the axis
Axis name font-family	Definition of the font
Axis name font-size	Definition of the font size
Axis name font-weight	Selecting the font weight • Bold • Normal
Unit	Definition of the unit

6.2.12.3 Attributes

6.2.12.3.1 GridBackgroundColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getGridBackgroundColor [▶ 1567]

Attribute setter: setGridBackgroundColor [▶ 1568]

Note: Available since version 1.8

6.2.12.3.2 GridLineColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getGridLineColor [▶ 1569]

Attribute setter: setGridLineColor [▶ 1569]

Note: Available since version 1.8

6.2.12.3.3 GridLineWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getGridLineWidth [▶ 1591]

Attribute setter: setGridLineWidth [▶ 1592]

Note: Available since version 1.8

6.2.12.3.4 GridShowHorizontalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getGridShowHorizontalLines [▶ 1589]

Attribute setter: setGridShowHorizontalLines [▶ 1590]

Note: Available since version 1.8

6.2.12.3.5 GridShowVerticalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getGridShowVerticalLines [▶ 1587]

Attribute setter: setGridShowVerticalLines [▶ 1588]

Note: Available since version 1.8

6.2.12.3.6 LineGraphData

Scheme: tchmi:framework#/definitions/ChartLineGraphDataDefinitionList [▶ 2207]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getLineGraphData [▶ 1575]

Attribute setter: setLineGraphData [▶ 1575]

Note: Available since version 1.8

6.2.12.3.7 LineGraphDescriptions

Scheme: tchmi:framework#/definitions/ChartLineGraphDescriptionDefinitionList [▶ 2207]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getLineGraphDescriptions [▶ 1575]

Attribute setter: setLineGraphDescriptions [▶ 1576]

Note: Available since version 1.8

6.2.12.3.8 SectionsBackgroundColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getGridBackgroundColor [▶ 1567]

Attribute setter: setGridBackgroundColor [▶ 1568]

Note: Available since version 1.10

6.2.12.3.9 ShowGrid

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getShowGrid [▶ 1599]

Attribute setter: setShowGrid [▶ 1600]

Note: Available since version 1.8

6.2.12.3.10 ShowTooltip

Selecting whether the tooltips should be displayed in stop mode.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getShowTooltip [▶ 1613]

Attribute setter: setShowTooltip [▶ 1613]

Note: Available since version 1.10

6.2.12.3.11 ShowXAxis

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getShowXAxis [▶ 1576]

Attribute setter: setShowXAxis [▶ 1577]

Note: Available since version 1.8

6.2.12.3.12 ShowXAxisName

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getShowXAxisName [▶ 1582]

Attribute setter: setShowXAxisName [▶ 1582]

Note: Available since version 1.8

6.2.12.3.13 ShowXSubTicks

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getShowXSubTicks [▶ 1578]

Attribute setter: setShowXSubTicks [▶ 1579]

Note: Available since version 1.8

6.2.12.3.14 ShowYSubTicks

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getShowYSubTicks [▶ 1603]

Attribute setter: setShowYSubTicks [▶ 1604]

Note: Available since version 1.8

6.2.12.3.15 SubgridLineColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getSubgridLineColor [▶ 1570]

Attribute setter: setSubgridLineColor [▶ 1570]

Note: Available since version 1.8

6.2.12.3.16 SubgridLineWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getSubgridLineWidth [▶ 1611]

Attribute setter: setSubgridLineWidth [▶ 1611]

Note: Available since version 1.8

6.2.12.3.17 SubgridShowHorizontalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getSubgridShowHorizontalLines [▶ 1610]

Attribute setter: setSubgridShowHorizontalLines [▶ 1610]

Note: Available since version 1.8

6.2.12.3.18 SubgridShowVerticalLines

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getSubgridShowVerticalLines [▶ 1609]

Attribute setter: setSubgridShowVerticalLines [▶ 1609]

Note: Available since version 1.8

6.2.12.3.19 TooltipBackgroundColor

Definition of the background color of the coordinate system.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipBackgroundColor [▶ 1574]

Attribute setter: setTooltipBackgroundColor [▶ 1574]

Note: Available since version 1.8

6.2.12.3.20 TooltipFontColor

Definition of the font color of the tooltip.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipFontColor [▶ 1573]

Attribute setter: setTooltipFontColor [▶ 1573]

Note: Available since version 1.8

6.2.12.3.21 TooltipFontFamily

Definition of the font using the name of the font family or the keyword.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipFontFamily [▶ 1615]

Attribute setter: setTooltipFontFamily [▶ 1615]

Note: Available since version 1.10

6.2.12.3.22 TooltipFontSize

Definition of the font size.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipFontSize [▶ 1616]

Attribute setter: setTooltipFontSize [▶ 1616]

Note: Available since version 1.10

6.2.12.3.23 TooltipFontSizeUnit

Selecting the unit of the font size in the tooltips (pixel or percent).

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipFontSizeUnit [▶ 1617]

Attribute setter: setTooltipFontSizeUnit [▶ 1617]

Note: Available since version 1.10

6.2.12.3.24 TooltipFontWeight

Selecting the font weight for the tooltips (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipFontWeight [▶ 1618]

Attribute setter: setTooltipFontWeight [▶ 1618]

Note: Available since version 1.10

6.2.12.3.25 TooltipFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

Definition of the formatting of the tooltips.

This is called when the tooltip is displayed.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(valueX: number, valueY: number): string;
```

Parameter

Name	Type	Description
valueX	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The numerical value for the x-coordinate.
valueY	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The numerical value for the y-coordinate.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The tooltip to be displayed. HTML is allowed here.

Origin: Line Chart [▶ 281]

See also

Attribute getter: getTooltipFormat [▶ 1614]

Attribute setter: setTooltipFormat [▶ 1614]

Note: Available since version 1.10

6.2.12.3.26 XAxisAutoScaling

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisAutoScaling [▶ 1584]

Attribute setter: setXAxisAutoScaling [▶ 1584]

Note: Available since version 1.8

6.2.12.3.27 XAxisColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisColor [▶ 1568]

Attribute setter: setXAxisColor [▶ 1569]

Note: Available since version 1.8

6.2.12.3.28 XAxisDecimalPlaces

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisDecimalPlaces [▶ 1586]

Attribute setter: setXAxisDecimalPlaces [▶ 1586]

Note: Available since version 1.8

6.2.12.3.29 XAxisFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

The formatting function for the labelling texts of the x-axis.

The selected function defines the X-axis labels. It is called for each displayed value of the main X-axis subdivision. It is assigned the respective numerical value.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The current numerical value of the x-axis.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The new value of the main X-axis subdivision. HTML is not allowed here.

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisFormat [▶ 1599]

Attribute setter: setXAxisFormat [▶ 1599]

Note: Available since version 1.10

6.2.12.3.30 XAxisName

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisName [▶ 1583]

Attribute setter: setXAxisName [▶ 1583]

Note: Available since version 1.8

6.2.12.3.31 XAxisNameFontColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisNameFontColor [▶ 1571]

Attribute setter: setXAxisNameFontColor [▶ 1571]

Note: Available since version 1.8

6.2.12.3.32 XAxisNameFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisNameFontFamily [▶ 1590]

Attribute setter: setXAxisNameFontFamily [▶ 1591]

Note: Available since version 1.8

6.2.12.3.33 XAxisNameFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisNameFontSize [▶ 1593]

Attribute setter: setXAxisNameFontSize [▶ 1593]

Note: Available since version 1.8

6.2.12.3.34 XAxisNameFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisNameFontSizeUnit [▶ 1594]

Attribute setter: setXAxisNameFontSizeUnit [▶ 1594]

Note: Available since version 1.8

6.2.12.3.35 XAxisNameFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisNameFontWeight [▶ 1595]

Attribute setter: setXAxisNameFontWeight [▶ 1595]

Note: Available since version 1.8

6.2.12.3.36 XAxisUnit

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisUnit [▶ 1585]

Attribute setter: setXAxisUnit [▶ 1585]

Note: Available since version 1.8

6.2.12.3.37 XAxisWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXAxisWidth [▶ 1588]

Attribute setter: setXAxisWidth [▶ 1589]

Note: Available since version 1.8

6.2.12.3.38 XLabelFontColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXLabelFontColor [▶ 1572]

Attribute setter: setXLabelFontColor [▶ 1572]

Note: Available since version 1.8

6.2.12.3.39 XLabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXLabelFontFamily [▶ 1592]

Attribute setter: setXLabelFontFamily [▶ 1593]

Note: Available since version 1.8

6.2.12.3.40 XLabelFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXLabelFontSize [▶ 1596]

Attribute setter: setXLabelFontSize [▶ 1596]

Note: Available since version 1.8

6.2.12.3.41 XLabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXLabelFontSizeUnit [▶ 1597]

Attribute setter: setXLabelFontSizeUnit [▶ 1597]

Note: Available since version 1.8

6.2.12.3.42 XLabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXLabelFontWeight [▶ 1598]

Attribute setter: setXLabelFontWeight [▶ 1598]

Note: Available since version 1.8

6.2.12.3.43 XMainTickMaxValue

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXMainTickMaxValue [▶ 1581]

Attribute setter: setXMainTickMaxValue [▶ 1581]

Note: Available since version 1.8

6.2.12.3.44 XMainTickMinValue

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXMainTickMinValue [▶ 1580]

Attribute setter: setXMainTickMinValue [▶ 1581]

Note: Available since version 1.8

6.2.12.3.45 XMainTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXMainTickSteps [▶ 1587]

Attribute setter: setXMainTickSteps [▶ 1587]

Note: Available since version 1.8

6.2.12.3.46 XShowLabels

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXShowLabels [▶ 1577]

Attribute setter: setXShowLabels [▶ 1578]

Note: Available since version 1.8

6.2.12.3.47 XSubTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getXSubTickSteps [▶ 1579]

Attribute setter: setXSubTickSteps [▶ 1580]

Note: Available since version 1.8

6.2.12.3.48 YAxis

Scheme: tchmi:framework#/definitions/ChartYAxisDefinitionList [▶ 2212]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYAxis [▶ 1600]

Attribute setter: setYAxis [▶ 1601]

Note: Available since version 1.8

6.2.12.3.49 YAxisWidth

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYAxisWidth [▶ 1607]

Attribute setter: setYAxisWidth [▶ 1607]

Note: Available since version 1.8

6.2.12.3.50 YLabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYLabelFontFamily [▶ 1608]

Attribute setter: setYLabelFontFamily [▶ 1608]

Note: Available since version 1.8

6.2.12.3.51 YLabelFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYLabelFontSize [▶ 1604]

Attribute setter: setYLabelFontSize [▶ 1605]

Note: Available since version 1.8

6.2.12.3.52 YLabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYLabelFontSizeUnit [▶ 1605]

Attribute setter: setYLabelFontSizeUnit [▶ 1605]

Note: Available since version 1.8

6.2.12.3.53 YLabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYLabelFontWeight [▶ 1606]

Attribute setter: setYLabelFontWeight [▶ 1606]

Note: Available since version 1.8

6.2.12.3.54 YMainTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYMainTickSteps [▶ 1601]

Attribute setter: setYMainTickSteps [▶ 1602]

Note: Available since version 1.8

6.2.12.3.55 YSubTickSteps

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Line Chart [▶ 281]

See also

Attribute getter: getYSubTickSteps [▶ 1602]

Attribute setter: setYSubTickSteps [▶ 1603]

Note: Available since version 1.8

6.2.13 Linear Gauge

A linear gauge is a linear display in which a value can be displayed by filling in a bar. A start and end value can be defined for this bar. The attribute 'Editable [▶ 314]' can be used to display a slider through which the operator can change the display.

Attributes

Category: Colors

Name	Description
ProgressBackgroundColor [▶ 320]	Definition of the background color of the bar
ProgressForegroundColor [▶ 320]	Definition of the bar color
ValueColor [▶ 324]	Definition of the text color of the display of the current value
UnitColor [▶ 322]	Definition of the unit color
TickColor [▶ 322]	Definition of the color of the bar division
LabelColor [▶ 315]	Definition of the text color of the scaling values
Knob Color [▶ 315]	Definition of the fill color of the slider
KnobArrowsColor [▶ 315]	Definition of the outline and arrow color of the slider
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Orientation [▶ 319]	Selecting the orientation: • Horizontal: The linear display with the scaling is aligned horizontally. • Vertical: The linear display with the scaling is aligned vertically.
Padding [▶ 319]	Definition of an additional distance between the control border and the bar for the four sides
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value

Name	Description
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered.

Name	Description
	• Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Value [▶ 324]	Definition of the value to be displayed
Editable [▶ 314]	Selecting whether the slider should be displayed and therefore whether the linear gauge is editable
BaseAnimationTime [▶ 314]	Definition of the basic animation time in milliseconds. It describes the time of the animation from minimum to maximum value.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Value

Name	Description
MinValue [▶ 319]	Defines the smallest value that the control element should graphically display
MaxValue [▶ 318]	Defines the largest value that the control element should graphically display
Step [▶ 321]	Definition of the increment for changing the value via the slider
ShowValueText [▶ 321]	Selecting whether the current value is to be displayed
Range [▶ 320]	Definition of the value range of the color scales
StartFromZero [▶ 321]	Selecting whether the minimum value should be zero regardless of the 'Minimum Value' attribute. This setting has no effect if the set minimum value is equal to or greater than zero.
ValueFontFamily [▶ 324]	Definition of the font using the name of the font family or the keyword
ValueFontSize [▶ 325]	Definition of the font size
ValueFontSizeUnit [▶ 325]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
ValueFontStyle [▶ 325]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
ValueFontWeight [▶ 325]	Selecting the font weight: • Normal • Bold: bold
ValueFormat [▶ 326]	Definition of a formatting function for the value
ClickAnywheretoEdit [▶ 314]	Selecting whether the value can be changed by clicking anywhere in the control

Category: Labels

Name	Description
ShowLabels [▶ 320]	Selecting whether the scaling values are to be displayed
Label Range [▶ 318]	Definition of the jump distance between the individual scaling values
Label Position [▶ 318]	Selecting the position of the scaling to the bar: The following options are available for horizontal alignment: • Top • Bottom The following options are available for vertical alignment: • Left • Right
LabelFontFamily [▶ 315]	Definition of the font using the name of the font family or the keyword
LabelFontSize [▶ 316]	Definition of the font size of the scaling
LabelFontSizeUnit [▶ 316]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
LabelFontStyle [▶ 316]	Selecting the font style of the scaling: • Normal • Italic: italic • Oblique: slanted
LabelFontWeight [▶ 317]	Selecting the font weight of the scaling: • Normal • Bold: bold
LabelFormat [▶ 317]	Definition of a formatting function for the scaling

Category: Ticks

Name	Description
ShowTicks [▶ 321]	Selecting whether the scaling subdivision is to be displayed
TickDefinition [▶ 322]	Definition of scale division

Category: Unit

Name	Description
Unit [▶ 314]	Definition of the unit or description of the value
UnitOrientation [▶ 319]	Selecting the unit orientation: • Theme • Horizontal • Vertical
UnitFontSize [▶ 323]	Definition of the font size of the unit
UnitFontSizeUnit [▶ 323]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
UnitFontStyle [▶ 323]	Selecting the font style of the unit: • Normal • Italic: italic • Oblique: slanted
UnitFontWeight [▶ 323]	Selecting the font weight of the unit: • Normal • Bold: bold

Events

Category: Control

Name	Description
onValueChanged [▶ 327]	The value has been changed.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onUserInteractionFinished [▶ 327]	The user has finished the interaction with the control.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Linear Gauge

Note: Available since version 1.8

See also

API reference [▶ 1507] for this control

6.2.13.1 Attributes

6.2.13.1.1 BaseAnimationTime

The basic animation time in milliseconds. It describes the time of the animation from minimum to maximum value.

Scheme: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getBaseAnimationTime [▶ 1555]

Attribute setter: setBaseAnimationTime [▶ 1555]

Note: Available since version 1.8

6.2.13.1.2 ClickAnywhereToEdit

To change the value, you can click anywhere in the control.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getClickAnywhereToEdit [▶ 1536]

Attribute setter: setClickAnywhereToEdit [▶ 1537]

Note: Available since version 1.8

6.2.13.1.3 Unit

The unit or description of the display value.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnit [▶ 1549]

Attribute setter: setUnit [▶ 1549]

Note: Available since version 1.8

6.2.13.1.4 Editable

Determines whether the slider should be displayed.

The appearance of the slider can be changed via Themed Resource [▶ 327] of the control.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getEditable [▶ 1525]

Attribute setter: setEditable [▶ 1526]

Note: Available since version 1.8

6.2.13.1.5 KnobArrowsColor

Outline and arrow color of the slider.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getKnobArrowsColor [▶ 1517]

Attribute setter: setKnobArrowsColor [▶ 1517]

Note: Available since version 1.8

6.2.13.1.6 KnobColor

Fill color of the slider.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getKnobColor [▶ 1520]

Attribute setter: setKnobColor [▶ 1520]

Note: Available since version 1.8

6.2.13.1.7 LabelColor

The color of the texts for the scale divisions.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelColor [▶ 1521]

Attribute setter: setLabelColor [▶ 1521]

Note: Available since version 1.8

6.2.13.1.8 LabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelFontFamily [▶ 1542]

Attribute setter: setLabelFontFamily [▶ 1542]

Note: Available since version 1.8

6.2.13.1.9 LabelFontSize

The font size.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelFontSize [▶ 1543]

Attribute setter: setLabelFontSize [▶ 1543]

Note: Available since version 1.8

6.2.13.1.10 LabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelFontSizeUnit [▶ 1544]

Attribute setter: setLabelFontSizeUnit [▶ 1544]

Note: Available since version 1.8

6.2.13.1.11 LabelFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelFontStyle [▶ 1545]

Attribute setter: setLabelFontStyle [▶ 1545]

Note: Available since version 1.8

6.2.13.1.12 LabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelFontWeight [▶ 1541]

Attribute setter: setLabelFontWeight [▶ 1542]

Note: Available since version 1.8

6.2.13.1.13 LabelFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

The formatting function for the label texts.

It is called for each value in the label.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The numerical value for the label.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The text to be displayed. HTML is not allowed here.

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelFormat [▶ 1546]

Attribute setter: setLabelFormat [▶ 1546]

Note: Available since version 1.8

6.2.13.1.14 LabelPosition

The position of the label texts and scale division relative to the display.

Scheme: tchmi:framework#/definitions/LinearGaugeLabelPosition [▶ 2239]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelPosition [▶ 1540]

Attribute setter: setLabelPosition [▶ 1541]

Note: Available since version 1.8

6.2.13.1.15 LabelRange

The value range between the label texts.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getLabelRange [▶ 1539]

Attribute setter: setLabelRange [▶ 1540]

Note: Available since version 1.8

6.2.13.1.16 MaxValue

The largest value that the control element should display graphically.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getMaxValue [▶ 1526]

Attribute setter: setMaxValue [▶ 1527]

Note: Available since version 1.8

6.2.13.1.17 MinValue

The smallest value that the control should display graphically.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getMinValue [▶ 1527]

Attribute setter: setMinValue [▶ 1528]

Note: Available since version 1.8

6.2.13.1.18 Orientation

The orientation of the display. Horizontal and vertical orientation are supported.

Scheme: tchmi:framework#/definitions/LinearGaugeOrientation [▶ 2239]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getOrientation [▶ 1523]

Attribute setter: setOrientation [▶ 1523]

Note: Available since version 1.8

6.2.13.1.19 UnitOrientatin

The orientation of unit.

Scheme: tchmi:framework#/definitions/LinearGaugeUnitOrientation [▶ 2240]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitOrientation [▶ 1549]

Attribute setter: setUnitOrientation [▶ 1550]

Note: Available since version 1.8

6.2.13.1.20 Padding

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getPadding [▶ 1523]

Attribute setter: setPadding [▶ 1524]

Note: Available since version 1.8

6.2.13.1.21 ProgressBackgroundColor

The color to the right of the value.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getProgressBackgroundColor [▶ 1515]

Attribute setter: setProgressBackgroundColor [▶ 1516]

Note: Available since version 1.8

6.2.13.1.22 ProgressForegroundColor

The color to the left of the value.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getProgressForegroundColor [▶ 1516]

Attribute setter: setProgressForegroundColor [▶ 1517]

Note: Available since version 1.8

6.2.13.1.23 Range

The value range of the color scales.

Scheme: tchmi:framework#/definitions/GaugeRangeDefinitionList [▶ 2237]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getRange [▶ 1530]

Attribute setter: setRange [▶ 1530]

Note: Available since version 1.8

6.2.13.1.24 ShowLabels

Label texts are either displayed or not.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getShowLabels [▶ 1538]

Attribute setter: setShowLabels [▶ 1539]

Note: Available since version 1.8

6.2.13.1.25 ShowTicks

Defines whether main axis markers are to be drawn.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getShowTicks [▶ 1547]

Attribute setter: setShowTicks [▶ 1547]

Note: Available since version 1.8

6.2.13.1.26 ShowValueText

Either displays the value or not.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getShowValueText [▶ 1529]

Attribute setter: setShowValueText [▶ 1529]

Note: Available since version 1.8

6.2.13.1.27 StartFromZero

Starts from zero or at the minimum value. This has no effect if the minimum value is equal to or greater than zero.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getStartFromZero [▶ 1531]

Attribute setter: setStartFromZero [▶ 1531]

Note: Available since version 1.8

6.2.13.1.28 Step

Defines the increment with which the value can be incremented/decremented.

Scheme: tchmi:framework#/definitions/PositiveNumber [▶ 2248]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getStep [▶ 1528]

Attribute setter: setStep [▶ 1529]

Note: Available since version 1.8

6.2.13.1.29 TickColor

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getTickColor [▶ 1518]

Attribute setter: setTickColor [▶ 1518]

Note: Available since version 1.8

6.2.13.1.30 TickDefinition

Describes the scale divisions.

Scheme: tchmi:framework#/definitions/GaugeTickDefinition [▶ 2237]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getTickDefinition [▶ 1548]

Attribute setter: setTickDefinition [▶ 1548]

Note: Available since version 1.8

6.2.13.1.31 UnitColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitColor [▶ 1522]

Attribute setter: setUnitColor [▶ 1522]

Note: Available since version 1.8

6.2.13.1.32 UnitFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitFontFamily [▶ 1553]

Attribute setter: setUnitFontFamily [▶ 1554]

Note: Available since version 1.8

6.2.13.1.33 UnitFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitFontSize [▶ 1550]

Attribute setter: setUnitFontSize [▶ 1551]

Note: Available since version 1.8

6.2.13.1.34 UnitFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitFontSizeUnit [▶ 1554]

Attribute setter: setUnitFontSizeUnit [▶ 1554]

Note: Available since version 1.8

6.2.13.1.35 UnitFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitFontStyle [▶ 1551]

Attribute setter: setUnitFontStyle [▶ 1552]

Note: Available since version 1.8

6.2.13.1.36 UnitFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getUnitFontWeight [▶ 1552]

Attribute setter: setUnitFontWeight [▶ 1553]

Note: Available since version 1.8

6.2.13.1.37 Value

The value of the display.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValue [▶ 1524]

Attribute setter: setValue [▶ 1525]

Note: Available since version 1.8

6.2.13.1.38 ValueColor

The color of the display value.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueColor [▶ 1519]

Attribute setter: setValueColor [▶ 1519]

Note: Available since version 1.8

6.2.13.1.39 ValueFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueFontFamily [▶ 1536]

Attribute setter: setValueFontFamily [▶ 1536]

Note: Available since version 1.8

6.2.13.1.40 ValueFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueFontSize [▶ 1535]

Attribute setter: setValueFontSize [▶ 1535]

Note: Available since version 1.8

6.2.13.1.41 ValueFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueFontSizeUnit [▶ 1534]

Attribute setter: setValueFontSizeUnit [▶ 1534]

Note: Available since version 1.8

6.2.13.1.42 ValueFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueFontStyle [▶ 1537]

Attribute setter: setValueFontStyle [▶ 1538]

Note: Available since version 1.8

6.2.13.1.43 ValueFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueFontWeight [▶ 1532]

Attribute setter: setValueFontWeight [▶ 1532]

Note: Available since version 1.8

6.2.13.1.44 ValueFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

The formatting function for the display value.

It is called for each displayed value.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The current numerical value.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The text to be displayed. HTML is not allowed here.

Origin: Linear Gauge [▶ 306]

See also

Attribute getter: getValueFormat [▶ 1533]

Attribute setter: setValueFormat [▶ 1533]

Note: Available since version 1.8

6.2.13.2 Events

6.2.13.2.1 onUserInteractionFinished

The user has finished the interaction [▶ 314] with the control.

Origin: Linear Gauge [▶ 306]

Note: Available since version 1.8

6.2.13.2.2 onValueChanged

The value [▶ 324] has been changed.

Origin: Linear Gauge [▶ 306]

Note: Available since version 1.8

6.2.13.3 Themed Resources

These resources cannot be defined per instance but only for several/all linear gauges via the theme file. If you really want to change only one instance, you must create a class for it.

Note: Available since version 1.10

6.2.13.3.1 KnobConfigHorizontal

A different graphic can be selected for the display of the slider.

This definition only applies to gauges that are horizontally aligned via the attribute Orientation [▶ 319]. If the control is positioned vertically, the resource KnobConfigVertical [▶ 328] is used.

The path of the image file and the display size of the image must be specified. Furthermore, the offset of the "anchor point" can be defined. This allows the image to be positioned correctly with respect to the value bar.

Note that an SVG file opens its own viewport and therefore the aspect ratio cannot normally be changed.

Partially transparent images in PNG and SVG format are also supported. For pixel graphics (PNG, JPG) please note that the attributes KnobColor [▶ 315] and KnobArrowsColor [▶ 315] have no effect. If an SVG is used, the color definition can take effect if the SVG is prepared accordingly.

When loading, the following character substitution is performed in the SVG:

- `{KnobColorStroke}` is replaced by "`stroke:`" followed by KnobColor [▶ 315] as CSS color if KnobColor [▶ 315] is set, otherwise removed
- `{KnobColorFill}` is replaced by "`fill:`" followed by KnobColor [▶ 315] as CSS color if KnobColor [▶ 315] is set, otherwise removed
- `{KnobColorGradientDefinition}` is replaced by a matching SVG `<linearGradient>` element as string if KnobColor [▶ 315] is a LinearGradientColor [▶ 2085], otherwise removed
- `{KnobArrowsColorStroke}` is replaced by "`stroke:`" followed by KnobArrowsColor [▶ 315] as CSS color if KnobArrowsColor [▶ 315] is set, otherwise removed
- `{KnobArrowsColorFill}` is replaced by "`fill:`" followed by KnobArrowsColor [▶ 315] as CSS color if KnobArrowsColor [▶ 315] is set, otherwise removed
- `{KnobArrowsColorGradientDefinition}` is replaced by a matching SVG `<linearGradient>` element as string if KnobArrowsColor [▶ 315] is a LinearGradientColor [▶ 2085], otherwise removed

Example SVG

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="40" height="40">
    <defs>{KnobColorGradientDefinition}</defs>
    <circle r="18px" cx="20px" cy="20px"
        style="stroke-width:3px;{KnobColorFill}{KnobArrowsColorStroke}"></circle>
    <path d="M 18 11 L 16 11 L 7 20 L 16 29 L 18 29 L 18 27 L 11 20 L 18 13 Z"
        style="stroke:none;{KnobArrowsColorFill}"></path>
    <path d="M 22 13 L 24 11 L 33 20 L 24 29 L 22 29 L 22 27 L 29 20 L 22 13 Z"
    style="stroke:none;{KnobArrowsColorFill}"></path>
</svg>
```

Scheme: tchmi:framework#/definitions/KnobDefinition [▶ 2239]

Origin: Linear Gauge [▶ 306]

Note: Available since version 1.10

6.2.13.3.2 KnobConfigVertical

A different graphic can be selected for the display of the slider.

This definition only applies to gauges that are vertically aligned via the attribute Orientation [▶ 319]. If the control element is positioned horizontally, the resource KnobConfigHorizontal [▶ 327] is used.

The path of the image file and the display size of the image must be specified. Furthermore, the offset of the "anchor point" can be defined. This allows the image to be positioned correctly with respect to the value bar.

Note that an SVG file opens its own viewport and therefore the aspect ratio cannot normally be changed.

Partially transparent images in PNG and SVG format are also supported. For pixel graphics (PNG, JPG) please note that the attributes KnobColor [▶ 315] and KnobArrowsColor [▶ 315] have no effect. If an SVG is used, the color definition can take effect if the SVG is prepared accordingly.

When loading, the following character substitution is performed in the SVG:

- `{KnobColorStroke}` is replaced by "`stroke:`" followed by KnobColor [▶ 315] as CSS color if KnobColor [▶ 315] is set, otherwise removed
- `{KnobColorFill}` is replaced by "`fill:`" followed by KnobColor [▶ 315] as CSS color if KnobColor [▶ 315] is set, otherwise removed
- `{KnobColorGradientDefinition}` is replaced by a matching SVG `<linearGradient>` element as string if KnobColor [▶ 315] is a LinearGradientColor [▶ 2085], otherwise removed
- `{KnobArrowsColorStroke}` is replaced by "`stroke:`" followed by KnobArrowsColor [▶ 315] as CSS color if KnobArrowsColor [▶ 315] is set, otherwise removed
- `{KnobArrowsColorFill}` is replaced by "`fill:`" followed by KnobArrowsColor [▶ 315] as CSS color if KnobArrowsColor [▶ 315] is set, otherwise removed
- `{KnobArrowsColorGradientDefinition}` is replaced by a matching SVG `<linearGradient>` element as string if KnobArrowsColor [▶ 315] is a LinearGradientColor [▶ 2085], otherwise removed

Example SVG

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="40" height="40">
    <defs>{KnobColorGradientDefinition}</defs>
    <circle r="18px" cx="20px" cy="20px"
        style="stroke-width:3px;{KnobColorFill}{KnobArrowsColorStroke}"></circle>
    <path d="M 18 11 L 16 11 L 7 20 L 16 29 L 18 29 L 18 27 L 11 20 L 18 13 Z"
        style="stroke:none;{KnobArrowsColorFill}"></path>
    <path d="M 22 13 L 24 11 L 33 20 L 24 29 L 22 29 L 22 27 L 29 20 L 22 13 Z"
    style="stroke:none;{KnobArrowsColorFill}"></path>
</svg>
```

Scheme: tchmi:framework#/definitions/KnobDefinition [▶ 2239]

Origin: Linear Gauge [▶ 306]

Note: Available since version 1.10

6.2.14 Polygon

Closed shape formed from a set of interconnected points. The last point is automatically connected to the first point.

Attributes

Category: Colors

Name	Description
FillColor [▶ 335]	Definition of the fill color of the control
StrokeColor [▶ 336]	Definition of the line color of the control
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Points [▶ 335]	Definition of the points that describe the polygon. Each point is defined by an x-coordinate and a y-coordinate.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Shape

Name	Description
StrokeThickness [▶ 336]	Definition of the line width
StrokeThicknessUnit [▶ 337]	Selecting the unit of the line width: • Pixels for an absolute width
StrokeStyle [▶ 336]	Definition of the line style
FillMode [▶ 335]	Selecting the fill areas: • NonZero: All areas that lie within the connecting lines are filled in. • EvenOdd: If a polygon is drawn so that the resulting surfaces overlap, this section is filled in for an odd number of overlapping surfaces and not filled in for an even number.
ScaleMode [▶ 335]	Selecting the polygon scaling: • None: The polygon is not scaled. The points that do not lie within the control are truncated. • ScaleToFill: The polygon is displayed on the entire size of the control. The proportions of the polygon are not taken into account. • ScaleToFit: The polygon is displayed as large as possible without exceeding the edges of the control. The proportions of the polygon are maintained. • ScaleToFitWidth: The polygon is displayed over the entire width of the control, regardless of whether it exceeds the limits of the control. The proportions of the polygon are maintained. • ScaleToFitHeight: The polygon is displayed over the entire height of the control, regardless of whether it exceeds the limits of the control. The proportions of the polygon are maintained.

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Polygon

Note: Available since version 1.8

See also

API reference [▶ 1618] for this control

6.2.14.1 Attributes

6.2.14.1.1 FillColor

Defines the fill color of the control.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Polygon [▶ 329]

See also

Attribute getter: getFillColor [▶ 1625]

Attribute setter: setFillColor [▶ 1625]

Note: Available since version 1.8

6.2.14.1.2 FillMode

Defines which areas are inside or outside. This is only required for self-overlapping polygons.

Scheme: tchmi:framework#/definitions/FillMode [▶ 2227]

Origin: Polygon [▶ 329]

See also

Attribute getter: getFillMode [▶ 1631]

Attribute setter: setFillMode [▶ 1631]

Note: Available since version 1.8

6.2.14.1.3 Points

Each point is defined by an x-coordinate and a y-coordinate.

It is common practice (but not necessary) to put a comma between the x- and y-coordinates and a space between the points, as in this example: `100,10 250,150 200,110`

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Polygon [▶ 329]

See also

Attribute getter: getPoints [▶ 1626]

Attribute setter: setPoints [▶ 1627]

Note: Available since version 1.8

6.2.14.1.4 ScaleMode

Scales the content to the control size.

Scheme: tchmi:framework#/definitions/ScaleMode [▶ 2251]

Origin: Polygon [▶ 329]

See also

Attribute getter: getScaleMode [▶ 1630]

Attribute setter: setScaleMode [▶ 1630]

Note: Available since version 1.8

6.2.14.1.5 StrokeColor

Defines the stroke color of the control.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Polygon [▶ 329]

See also

Attribute getter: getStrokeColor [▶ 1626]

Attribute setter: setStrokeColor [▶ 1626]

Note: Available since version 1.8

6.2.14.1.6 StrokeStyle

Definition of the stroke style.

Comma or space-separated list of numbers that specify the length of strokes and spaces.

If an odd number is specified, the list is repeated once. Example: "`5, 3, 2`" is equivalent to "`5, 3, 2, 5, 3, 2`".

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Polygon [▶ 329]

See also

Attribute getter: getStrokeStyle [▶ 1627]

Attribute setter: setStrokeStyle [▶ 1628]

Note: Available since version 1.8

6.2.14.1.7 StrokeThickness

Defines the border width of the control in pixels.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Polygon [▶ 329]

See also

Attribute getter: getStrokeThickness [▶ 1628]

Attribute setter: setStrokeThickness [▶ 1629]

Note: Available since version 1.8

6.2.14.1.8 StrokeThicknessUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Polygon [▶ 329]

See also

Attribute getter: getStrokeThicknessUnit [▶ 1629]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.15 Radial Gauge

A radial gauge is a radial display in which a value can be displayed by filling in a bar. A start and end value can be defined for this bar. The attribute 'Editable [▶ 345]' can be used to display a slider through which the operator can change the display.

Attributes

Category: Colors

Name	Description
ProgressBackgroundColor [▶ 350]	Definition of the background color of the bar
ProgressForegroundColor [▶ 351]	Definition of the bar color
GaugeBackground [▶ 346]	Definition of the color of the round background
ValueColor [▶ 355]	Definition of the text color of the display of the current value
LabelColor [▶ 347]	Definition of the text color of the scaling values
UnitColor [▶ 354]	Definition of the unit color
TickColor [▶ 353]	Definition of the scaling color
KnobColor [▶ 346]	Definition of the fill color of the slider
KnobArrowsColor [▶ 346]	Definition of the outline and arrow color of the slider
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
StartPosition [▶ 352]	Definition of the start angle of the display
EndPosition [▶ 346]	Definition of the end angle of the display
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value

Name	Description
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Value [▶ 355]	Definition of the value to be displayed
Editable [▶ 345]	Selecting whether the slider should be displayed and therefore whether the linear gauge is editable
BaseAnimationTime [▶ 345]	Definition of the basic animation time in milliseconds. It describes the time of the animation from minimum to maximum value.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Value

Name	Description
MinValue [▶ 350]	Defines the smallest value that the control element should graphically display
MaxValue [▶ 350]	Defines the largest value that the control element should graphically display
Step [▶ 352]	Definition of the increment for changing the value via the slider
Range [▶ 351]	Definition of the value range of the color scales
ShowValueText [▶ 352]	Selecting whether the current value is to be displayed
StartFromZero [▶ 352]	Selecting whether the minimum value is to be zero irrespective of the 'Minimum Value' attribute.
ValueFontFamily [▶ 356]	Definition of the font using the name of the font family or the keyword
ValueFontSize [▶ 356]	Definition of the font size
ValueFontSizeUnit [▶ 356]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
ValueFontStyle [▶ 356]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
ValueFontWeight [▶ 357]	Selecting the font weight: • Normal • Bold: bold
ValueFormat [▶ 357]	Definition of a formatting function for the value
ClickAnywhereToEdit [▶ 345]	Selecting whether the value can be changed by clicking anywhere in the control

Category: Labels

Name	Description
ShowLabels [▶ 351]	Selecting whether the scaling values are to be displayed
LabelPosition [▶ 349]	Selecting the position of the scaling to the bar: • Inside • Outside
LabelRange [▶ 350]	Definition of the jump distance between the individual scaling values
LabelFormat [▶ 348]	Definition of a formatting function for the scaling
LabelFontFamily [▶ 347]	Definition of the font using the name of the font family or the keyword
LabelFontSize [▶ 347]	Definition of the font size of the scaling
LabelFontSizeUnit [▶ 348]	Selecting the unit: • Pixels for an absolute size • Percent for a relative size
LabelFontStyle [▶ 348]	Selecting the font style of the scaling: • Normal • Italic: italic • Oblique: slanted
LabelFontWeight [▶ 348]	Selecting the font weight of the scaling: • Normal • Bold: bold

Category: Ticks

Name	Description
ShowTicks [▶ 351]	Selecting whether the scaling subdivision is to be displayed
TickDefinition [▶ 353]	Definition of scale divisions
SubTickPosition [▶ 353]	Selecting the position of the auxiliary scaling to the bar: • Inside • Outside

Category: Unit

Name	Description
Unit [▶ 345]	Definition of the unit or description of the value
UnitFontFamily [▶ 354]	Definition of the font using the name of the font family or the keyword
UnitFontSize [▶ 354]	Definition of the font size of the unit
UnitFontSizeUnit [▶ 354]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
UnitFontStyle [▶ 355]	Selecting the font style of the label: • Normal • Italic: italic • Oblique: slanted
UnitFontWeight [▶ 355]	Selecting the font weight: • Normal • Bold: bold

Events

Category: Control

Name	Description
onValueChanged [▶ 358]	The value has been changed.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onUserInteractionFinished [▶ 358]	The user has finished the interaction with the control.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Radial Gauge

Note: Available since version 1.8

See also

API reference [▶ 1632] for this control

6.2.15.1 Attributes

6.2.15.1.1 BaseAnimationTime

The basic animation time in milliseconds. It describes the time of the animation from minimum to maximum value.

Scheme: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getBaseAnimationTime [▶ 1680]

Attribute setter: setBaseAnimationTime [▶ 1681]

Note: Available since version 1.8

6.2.15.1.2 ClickAnywhereToEdit

To change the value, you can click anywhere in the control.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getClickAnywhereToEdit [▶ 1658]

Attribute setter: setClickAnywhereToEdit [▶ 1658]

Note: Available since version 1.8

6.2.15.1.3 Unit

The unit or description of the display value.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnit [▶ 1675]

Attribute setter: setUnit [▶ 1675]

Note: Available since version 1.8

6.2.15.1.4 Editable

Determines whether the slider should be displayed.

The appearance of the slider can be changed via Themed Resource of the control.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getEditable [▶ 1650]

Attribute setter: setEditable [▶ 1651]

Note: Available since version 1.8

6.2.15.1.5 EndPosition

The end angle of the display.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getEndPosition [▶ 1648]

Attribute setter: setEndPosition [▶ 1648]

Note: Available since version 1.8

6.2.15.1.6 GaugeBackgroundColor

The color of the round background.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getGaugeBackgroundColor [▶ 1641]

Attribute setter: setGaugeBackgroundColor [▶ 1642]

Note: Available since version 1.8

6.2.15.1.7 KnobArrowsColor

Outline and arrow color of the slider.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getKnobArrowsColor [▶ 1642]

Attribute setter: setKnobArrowsColor [▶ 1643]

Note: Available since version 1.8

6.2.15.1.8 KnobColor

Fill color of the slider.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getKnobColor [▶ 1643]

Attribute setter: setKnobColor [▶ 1644]

Note: Available since version 1.8

6.2.15.1.9 LabelColor

The color of the texts for the scale divisions.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelColor [▶ 1645]

Attribute setter: setLabelColor [▶ 1645]

Note: Available since version 1.8

6.2.15.1.10 LabelFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelFontFamily [▶ 1667]

Attribute setter: setLabelFontFamily [▶ 1668]

Note: Available since version 1.8

6.2.15.1.11 LabelFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelFontSize [▶ 1668]

Attribute setter: setLabelFontSize [▶ 1668]

Note: Available since version 1.8

6.2.15.1.12 LabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelFontSizeUnit [▶ 1669]

Attribute setter: setLabelFontSizeUnit [▶ 1669]

Note: Available since version 1.8

6.2.15.1.13 LabelFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelFontStyle [▶ 1670]

Attribute setter: setLabelFontStyle [▶ 1670]

Note: Available since version 1.8

6.2.15.1.14 LabelFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelFontWeight [▶ 1671]

Attribute setter: setLabelFontWeight [▶ 1671]

Note: Available since version 1.8

6.2.15.1.15 LabelFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

The formatting function for the label texts.

It is called for each value in the label.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The numerical value for the label.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The text to be displayed. HTML is not allowed here.

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelFormat [▶ 1666]

Attribute setter: setLabelFormat [▶ 1667]

Note: Available since version 1.8

6.2.15.1.16 LabelPosition

The position of the label texts (inside or outside).

Scheme: tchmi:framework#/definitions/RadialGaugeLabelPosition [▶ 2248]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelPosition [▶ 1664]

Attribute setter: setLabelPosition [▶ 1665]

Note: Available since version 1.8

6.2.15.1.17 LabelRange

The value range between the label texts.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getLabelRange [▶ 1665]

Attribute setter: setLabelRange [▶ 1666]

Note: Available since version 1.8

6.2.15.1.18 MaxValue

The largest value that the control element should display graphically.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getMaxValue [▶ 1652]

Attribute setter: setMaxValue [▶ 1653]

Note: Available since version 1.8

6.2.15.1.19 MinValue

The smallest value that the control should display graphically.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getMinValue [▶ 1651]

Attribute setter: setMinValue [▶ 1652]

Note: Available since version 1.8

6.2.15.1.20 ProgressBackgroundColor

The color to the right of the value.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getProgressBackgroundColor [▶ 1640]

Attribute setter: setProgressBackgroundColor [▶ 1641]

Note: Available since version 1.8

6.2.15.1.21 ProgressForegroundColor

The color to the left of the value.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getProgressForegroundColor [▶ 1639]

Attribute setter: setProgressForegroundColor [▶ 1640]

Note: Available since version 1.8

6.2.15.1.22 Range

The value range of the color scales.

Scheme: tchmi:framework#/definitions/GaugeRangeDefinitionList [▶ 2237]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getRange [▶ 1654]

Attribute setter: setRange [▶ 1655]

Note: Available since version 1.8

6.2.15.1.23 ShowLabels

Label texts are either displayed or not.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getShowLabels [▶ 1663]

Attribute setter: setShowLabels [▶ 1664]

Note: Available since version 1.8

6.2.15.1.24 ShowTicks

Defines whether main axis markers are to be drawn.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getShowTicks [▶ 1672]

Attribute setter: setShowTicks [▶ 1672]

Note: Available since version 1.8

6.2.15.1.25 ShowValueText

Either displays the number value or not.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getShowValueText [▶ 1655]

Attribute setter: setShowValueText [▶ 1656]

Note: Available since version 1.8

6.2.15.1.26 StartPosition

The start angle of the display.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getStartPosition [▶ 1649]

Attribute setter: setStartPosition [▶ 1649]

Note: Available since version 1.8

6.2.15.1.27 StartFromZero

Starts from zero or at the minimum value. This has no effect if the minimum value is equal to or greater than zero.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getStartFromZero [▶ 1656]

Attribute setter: setStartFromZero [▶ 1656]

Note: Available since version 1.8

6.2.15.1.28 Step

Defines the increment with which the value can be incremented/decremented.

Scheme: tchmi:framework#/definitions/PositiveNumber [▶ 2248]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getStep [▶ 1653]

Attribute setter: setStep [▶ 1654]

Note: Available since version 1.8

6.2.15.1.29 SubTickPosition

The position (inside or outside) of the secondary to the primary scale divisions.

Scheme: tchmi:framework#/definitions/RadialGaugeLabelPosition [▶ 2248]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getSubTickPosition [▶ 1673]

Attribute setter: setSubTickPosition [▶ 1673]

Note: Available since version 1.8

6.2.15.1.30 TickColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getTickColor [▶ 1647]

Attribute setter: setTickColor [▶ 1647]

Note: Available since version 1.8

6.2.15.1.31 TickDefinition

Describes the scale divisions.

Scheme: tchmi:framework#/definitions/GaugeTickDefinition [▶ 2237]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getTickDefinition [▶ 1674]

Attribute setter: setTickDefinition [▶ 1674]

Note: Available since version 1.8

6.2.15.1.32 UnitColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnitColor [▶ 1646]

Attribute setter: setUnitColor [▶ 1646]

Note: Available since version 1.8

6.2.15.1.33 UnitFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnitFontFamily [▶ 1676]

Attribute setter: setUnitFontFamily [▶ 1677]

Note: Available since version 1.8

6.2.15.1.34 UnitFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnitFontSize [▶ 1675]

Attribute setter: setUnitFontSize [▶ 1676]

Note: Available since version 1.8

6.2.15.1.35 UnitFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnitFontSizeUnit [▶ 1679]

Attribute setter: setUnitFontSizeUnit [▶ 1680]

Note: Available since version 1.8

6.2.15.1.36 UnitFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnitFontStyle [▶ 1677]

Attribute setter: setUnitFontStyle [▶ 1678]

Note: Available since version 1.8

6.2.15.1.37 UnitFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getUnitFontWeight [▶ 1678]

Attribute setter: setUnitFontWeight [▶ 1679]

Note: Available since version 1.8

6.2.15.1.38 Value

The value of the display.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValue [▶ 1650]

Attribute setter: setValue [▶ 1650]

Note: Available since version 1.8

6.2.15.1.39 ValueColor

The color of the display value.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueColor [▶ 1644]

Attribute setter: setValueColor [▶ 1644]

Note: Available since version 1.8

6.2.15.1.40 ValueFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueFontFamily [▶ 1659]

Attribute setter: setValueFontFamily [▶ 1659]

Note: Available since version 1.8

6.2.15.1.41 ValueFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueFontSize [▶ 1660]

Attribute setter: setValueFontSize [▶ 1660]

Note: Available since version 1.8

6.2.15.1.42 ValueFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueFontSizeUnit [▶ 1661]

Attribute setter: setValueFontSizeUnit [▶ 1661]

Note: Available since version 1.8

6.2.15.1.43 ValueFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueFontStyle [▶ 1657]

Attribute setter: setValueFontStyle [▶ 1657]

Note: Available since version 1.8

6.2.15.1.44 ValueFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueFontWeight [▶ 1662]

Attribute setter: setValueFontWeight [▶ 1662]

Note: Available since version 1.8

6.2.15.1.45 ValueFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

The formatting function for the display value.

It is called for each displayed value.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The current numerical value.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The text to be displayed. HTML is not allowed here.

Origin: Radial Gauge [▶ 337]

See also

Attribute getter: getValueFormat [▶ 1662]

Attribute setter: setValueFormat [▶ 1663]

Note: Available since version 1.8

6.2.15.2 Events

6.2.15.2.1 onUserInteractionFinished

The user has finished the interaction [▶ 345] with the control.

Origin: Radial Gauge [▶ 337]

Note: Available since version 1.8

6.2.15.2.2 onValueChanged

The value [▶ 355] has been changed.

Origin: Radial Gauge [▶ 337]

Note: Available since version 1.8

6.2.15.3 Themed Resources

These resources cannot be defined per instance but only for several/all radial gauges via the theme file. If you really want to change only one instance, you must create a class for it.

Note: Available since version 1.10

6.2.15.3.1 KnobConfig

A different graphic can be selected for the display of the slider.

The path of the image file and the display size of the image must be specified. Furthermore, the offset of the "anchor point" can be defined. This allows the image to be positioned correctly with respect to the value bar.

Note that an SVG file opens its own viewport and therefore the aspect ratio cannot normally be changed.

Partially transparent images in PNG and SVG format are also supported. For pixel graphics (PNG, JPG) please note that the attributes KnobColor [▶ 346] and KnobArrowsColor [▶ 346] have no effect. If an SVG is used, the color definition can take effect if the SVG is prepared accordingly.

When loading, the following character substitution is performed in the SVG:

- `{KnobColorStroke}` is replaced by "`stroke:`" followed by KnobColor [▶ 346] as CSS color if KnobColor [▶ 346] is set, otherwise removed
- `{KnobColorFill}` is replaced by "`fill:`" followed by KnobColor [▶ 346] as CSS color if KnobColor [▶ 346] is set, otherwise removed
- `{KnobColorGradientDefinition}` is replaced by a matching SVG `<linearGradient>` element as string if KnobColor [▶ 346] is a LinearGradientColor [▶ 2085], otherwise removed
- `{KnobArrowsColorStroke}` is replaced by "`stroke:`" followed by KnobArrowsColor [▶ 315] as CSS color if KnobArrowsColor [▶ 346] is set, otherwise removed
- `{KnobArrowsColorFill}` is replaced by "`fill:`" followed by KnobArrowsColor [▶ 315] as CSS color if KnobArrowsColor [▶ 346] is set, otherwise removed
- `{KnobArrowsColorGradientDefinition}` is replaced by a matching SVG `<linearGradient>` element as string if KnobArrowsColor [▶ 346] is a LinearGradientColor [▶ 2085], otherwise removed

Example SVG

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="40" height="40">
    <defs>{KnobColorGradientDefinition}</defs>
    <circle r="18px" cx="20px" cy="20px"
        style="stroke-width:3px;{KnobColorFill}{KnobArrowsColorStroke}"></circle>
    <path d="M 18 11 L 16 11 L 7 20 L 16 29 L 18 29 L 18 27 L 11 20 L 18 13 Z"
        style="stroke:none;{KnobArrowsColorFill}"></path>
    <path d="M 22 13 L 24 11 L 33 20 L 24 29 L 22 29 L 22 27 L 29 20 L 22 13 Z"
    style="stroke:none;{KnobArrowsColorFill}"></path>
</svg>
```

Scheme: tchmi:framework#/definitions/KnobDefinition [▶ 2239]

Origin: Radial Gauge [▶ 337]

Note: Available since version 1.10

6.2.16 Rectangle

A rectangle is a rectangular object.

Attributes

Category: Colors

Name	Description
FillColor [▶ 366]	Definition of the fill color of the control
StrokeColor [▶ 366]	Definition of the line color
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Shape

Name	Description
StrokeThickness [▶ 367]	Definition of the line width
StrokeThicknessUnit [▶ 367]	Selecting the unit of the line width: • Pixels for an absolute width
StrokeStyle [▶ 366]	Definition of the line style
RadiusX [▶ 365]	Definition of the x-axis radius of the ellipse used to round the corners of the rectangle
RadiusXUnit [▶ 365]	Selecting the unit of the x-axis radius: • Pixel for an absolute radius
RadiusY [▶ 365]	Definition of the y-axis radius of the ellipse used to round the corners of the rectangle
RadiusYUnit [▶ 365]	Selecting the unit of the y-axis radius: • Pixel for an absolute radius

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Rectangle

Note: Available since version 1.8

See also

API reference [▶ 1682] for this control

6.2.16.1 Attributes

6.2.16.1.1 RadiusX

Definition of the x-axis radius of the ellipse used to round the corners of the rectangle.

If a value for RadiusX is specified but not RadiusY (or vice versa), this value applies to RadiusX and RadiusY.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getRadiusX [▶ 1692]

Attribute setter: setRadiusX [▶ 1692]

Note: Available since version 1.8

6.2.16.1.2 RadiusXUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getRadiusXUnit [▶ 1694]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.16.1.3 RadiusY

Definition of the y-axis radius of the ellipse used to round the corners of the rectangle.

If a value for RadiusX is specified but not RadiusY (or vice versa), this value applies to RadiusX and RadiusY.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getRadiusY [▶ 1693]

Attribute setter: setRadiusY [▶ 1694]

Note: Available since version 1.8

6.2.16.1.4 RadiusYUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getRadiusYUnit [▶ 1693]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.16.1.5 FillColor

Defines the fill color of the control.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getFillColor [▶ 1688]

Attribute setter: setFillColor [▶ 1688]

Note: Available since version 1.8

6.2.16.1.6 StrokeColor

Defines the stroke color of the control.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getStrokeColor [▶ 1689]

Attribute setter: setStrokeColor [▶ 1689]

Note: Available since version 1.8

6.2.16.1.7 StrokeStyle

Definition of the stroke style. Comma or space-separated list of numbers that specify the length of strokes and spaces. If an odd number is specified, the list is repeated once. Example: "5, 3, 2" is equivalent to "5, 3, 2, 5, 3, 2".

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getStrokeStyle [▶ 1691]

Attribute setter: setStrokeStyle [▶ 1691]

Note: Available since version 1.8

6.2.16.1.8 StrokeThickness

Defines the stroke thickness of the control in pixels.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getStrokeThickness [▶ 1689]

Attribute setter: setStrokeThickness [▶ 1690]

Note: Available since version 1.8

6.2.16.1.9 StrokeThicknessUnit

Pixel unit. At this point, only pixels can be used as a unit.

Scheme: tchmi:framework#/definitions/PixelUnit [▶ 2248]

Origin: Rectangle [▶ 359]

See also

Attribute getter: getStrokeThicknessUnit [▶ 1690]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.17 Textblock

A text block is a control in which text can be displayed.

Attributes

Category: Colors

Name	Description
TextColor [▶ 375]	Definition of the text color
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The 'Width' attribute is ignored in all calculations!
HeightMode [▶ 374]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the upper and lower side of the parent element. The 'Height' attribute is ignored in all calculations! • Content: The height is defined by the length of the text. The 'Height' attribute is ignored in all calculations!

Category: Common

Name	Description
Text [▶ 375]	Definition of the text
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Text

Name	Description
TextHorizontalAlignment [▶ 375]	Selecting the horizontal alignment of the text
TextVerticalAlignment [▶ 375]	Selecting the vertical alignment of the text
ContentPadding [▶ 373]	Definition of the distance between the text and the control border
WordWrap [▶ 376]	Selecting whether the text should be wrapped if it is too long to fit in one row
TextFontFamily [▶ 373]	Definition of the font using the name of the font family or the keyword
TextFontSize [▶ 373]	Definition of the font size
TextFontSizeUnit [▶ 373]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
TextFontStyle [▶ 374]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
TextFontWeight [▶ 374]	Selecting the font weight: • Normal • Bold: bold

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Text block

Note: Available since version 1.8

See also

API reference [▶ 1694] for this control

6.2.17.1 Attributes

6.2.17.1.1 ContentPadding

Defines the distance of the text to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Textblock [▶ 367]

See also

Attribute getter: getContentPadding [▶ 1706]

Attribute setter: setContentPadding [▶ 1707]

Note: Available since version 1.8

6.2.17.1.2 TextFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextFontFamily [▶ 1710]

Attribute setter: setTextFontFamily [▶ 1711]

Note: Available since version 1.8

6.2.17.1.3 TextFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextFontSize [▶ 1704]

Attribute setter: setTextFontSize [▶ 1704]

Note: Available since version 1.8

6.2.17.1.4 TextFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextFontSizeUnit [▶ 1705]

Attribute setter: setTextFontSizeUnit [▶ 1705]

Note: Available since version 1.8

6.2.17.1.5 TextFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextFontStyle [▶ 1708]

Attribute setter: setTextFontStyle [▶ 1709]

Note: Available since version 1.8

6.2.17.1.6 TextFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextFontWeight [▶ 1709]

Attribute setter: setTextFontWeight [▶ 1710]

Note: Available since version 1.8

6.2.17.1.7 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!
- **Content:** The height is defined by the text length of the control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Scheme: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Textblock [▶ 367]

See also

Attribute getter: getHeightMode [▶ 1701]

Attribute setter: setHeightMode [▶ 1701]

Note: Available since version 1.8

6.2.17.1.8 TextHorizontalAlignment

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextHorizontalAlignment [▶ 1706]

Attribute setter: setTextHorizontalAlignment [▶ 1706]

Note: Available since version 1.8

6.2.17.1.9 TextVerticalAlignment

Scheme: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextVerticalAlignment [▶ 1703]

Attribute setter: setTextVerticalAlignment [▶ 1703]

Note: Available since version 1.8

6.2.17.1.10 Text

Contents of the text block.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Textblock [▶ 367]

See also

Attribute getter: getText [▶ 1702]

Attribute setter: setText [▶ 1702]

Note: Available since version 1.8

6.2.17.1.11 TextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Textblock [▶ 367]

See also

Attribute getter: getTextColor [▶ 1700]

Attribute setter: setTextColor [▶ 1700]

Note: Available since version 1.8

6.2.17.1.12 WordWrap

Wraps the text if it is too long to fit in one row.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Textblock [▶ 367]

See also

Attribute getter: getWordWrap [▶ 1707]

Attribute setter: setWordWrap [▶ 1708]

Note: Available since version 1.8

6.2.18 Textbox

A text box is a control in which text can be displayed and entered.

Attributes

Category: Colors

Name	Description
TextColor [▶ 384]	Definition of the text color
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Text [▶ 384]	Definition of the text
AutoFocusOut [▶ 385]	Determines whether the control element automatically loses focus when the user confirms or ends his input by pressing Enter or Escape.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Text

Name	Description
TextHorizontalAlignment [▶ 383]	Selecting the alignment of the text
TextVerticalAlignment [▶ 383]	Selecting the vertical alignment of the text
TextFontFamily [▶ 382]	Definition of the font using the name of the font family or the keyword
TextFontSize [▶ 382]	Definition of the font size
TextFontSizeUnit [▶ 382]	Selecting the font size unit: • Pixels for an absolute size • Percent for a relative size
TextFontStyle [▶ 383]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
TextFontWeight [▶ 383]	Selecting the font weight: • Normal • Bold: bold
WordWrap [▶ 385]	Selecting whether the text should wrap if it is too long to fit in one row
ContentPadding [▶ 382]	Definition of the distance between the text and the control border
Multiline [▶ 384]	Selecting whether a multi-line text should be allowed
Placeholder [▶ 384]	Definition of a placeholder, which is displayed if no text is entered

Events

Category: Control

Name	Description
onTextChanged [▶ 386]	The text has been changed.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onUserInteractionFinished [▶ 386]	The user has finished the interaction with the control.
onFocusIn [▶ 385]	The control now has the focus.
onFocusOut [▶ 385]	The control has lost focus.
onUserInteractionCanceled [▶ 386]	The user has terminated the interaction with the control element.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Text box

Note: Available since version 1.8

See also

API reference [▶ 1711] for this control

6.2.18.1 Attributes

6.2.18.1.1 ContentPadding

Defines the distance of the text to the border.

Scheme: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Textbox [▶ 376]

See also

Attribute getter: getContentPadding [▶ 1725]

Attribute setter: setContentPadding [▶ 1725]

Note: Available since version 1.8

6.2.18.1.2 TextFontFamily

Comma-separated list of fonts as font family or keyword: 'serif', 'sans-serif', 'monospace'.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextFontFamily [▶ 1728]

Attribute setter: setTextFontFamily [▶ 1729]

Note: Available since version 1.8

6.2.18.1.3 TextFontSize

The font size. If the percent is specified as the unit, this is relative to the font size of the parent element.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextFontSize [▶ 1721]

Attribute setter: setTextFontSize [▶ 1721]

Note: Available since version 1.8

6.2.18.1.4 TextFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextFontSizeUnit [▶ 1722]

Attribute setter: setTextFontSizeUnit [▶ 1722]

Note: Available since version 1.8

6.2.18.1.5 TextFontStyle

Font style (normal, italic: oblique with special characters, oblique)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextFontStyle [▶ 1726]

Attribute setter: setTextFontStyle [▶ 1727]

Note: Available since version 1.8

6.2.18.1.6 TextFontWeight

Font weight (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextFontWeight [▶ 1727]

Attribute setter: setTextFontWeight [▶ 1728]

Note: Available since version 1.8

6.2.18.1.7 TextHorizontalAlignment

Scheme: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextHorizontalAlignment [▶ 1720]

Attribute setter: setTextHorizontalAlignment [▶ 1720]

Note: Available since version 1.8

6.2.18.1.8 TextVerticalAlignment

Schema: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Text box [▶ 376]

See also

Attribute getter: getTextVerticalAlignment [▶ 1719]

Attribute setter: setTextVerticalAlignment [▶ 1719]

Note: Available since version 1.8

Not applicable from version 1.10.1018.48

6.2.18.1.9 Multiline

Determines whether multi-line text should be allowed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Textbox [▶ 376]

See also

Attribute getter: getMultiline [▶ 1725]

Attribute setter: setMultiline [▶ 1726]

Note: Available since version 1.8

6.2.18.1.10 Placeholder

Placeholder which is displayed if no text is entered.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Textbox [▶ 376]

See also

Attribute getter: getPlaceholder [▶ 1723]

Attribute setter: setPlaceholder [▶ 1723]

Note: Available since version 1.8

6.2.18.1.11 Text

Contents of the text box.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Textbox [▶ 376]

See also

Attribute getter: getText [▶ 1718]

Attribute setter: setText [▶ 1719]

Note: Available since version 1.8

6.2.18.1.12 TextColor

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Textbox [▶ 376]

See also

Attribute getter: getTextColor [▶ 1717]

Attribute setter: setTextColor [▶ 1718]

Note: Available since version 1.8

6.2.18.1.13 WordWrap

Wraps the text if it is too long to fit in one row.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Textbox [▶ 376]

See also

Attribute getter: getWordWrap [▶ 1724]

Attribute setter: setWordWrap [▶ 1724]

Note: Available since version 1.8

6.2.18.1.14 AutoFocusOut

Determines whether the control element automatically loses focus when the user confirms or ends his input by pressing Enter or Escape.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Textbox [▶ 376]

See also

Attribute getter: getAutoFocusOut [▶ 1729]

Attribute setter: setAutoFocusOut [▶ 1730]

Note: Available since version 1.10

6.2.18.2 Events

6.2.18.2.1 onFocusIn

The control now has the focus.

Origin: Textbox [▶ 376]

Note: Available since version 1.8

6.2.18.2.2 onFocusOut

The control has lost focus.

Origin: Textbox [▶ 376]

Note: Available since version 1.8

6.2.18.2.3 onTextChanged

The text has been changed.

Origin: Textbox [▶ 376]

Note: Available since version 1.8

6.2.18.2.4 onUserInteractionFinished

The user has finished the interaction with the control.

Origin: Textbox [▶ 376]

Note: Available since version 1.8

6.2.18.2.5 onUserInteractionCanceled

The user has terminated the interaction with the control element.

Origin: Textbox [▶ 376]

Note: Available since version 1.10

6.2.19 Toggle Button

The toggle button is a switching element that can be switched between two states. The 'Toggle Group' attribute can be used to interconnect different toggle buttons. When a new toggle button is activated, the previously activated button is automatically deactivated.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Text [▶ 176]	Definition of the text displayed in the button
StateSymbol [▶ 178]	Reflects the pressed state of the button for the assigned symbol and vice versa. If the button is pressed, the symbol is set to true. If the button is released, the symbol is set to false. If the symbol is set to true independent of the button, the button is pressed. If the symbol is set to false independent of the button, the button is released. During an active user interaction, no value changes of the symbol are transferred to the button.
ToggleGroup [▶ 392]	User-defined name for a group. Only one control in this group can be active.
ToggleState [▶ 392]	Internal state of the control. This can be Normal or Active.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Icon

Name	Description
Icon [▶ 175]	Definition of an optional image in the button
IconWidth [▶ 177]	Definition of the width of the icon
IconWidthUnit [▶ 177]	Definition of the unit of the width: • Pixels for an absolute width • Percent for a relative width
IconHeight [▶ 174]	Definition of the height of the icon
IconHeightUnit [▶ 174]	Definition of the unit of height: • Pixels for an absolute height • Percent for a relative height
IconHorizontalAlignment [▶ 175]	Selecting the horizontal alignment of the icon within the control
IconVerticalAlignment [▶ 176]	Selecting the vertical alignment of the icon within the control
IconPadding [▶ 175]	Definition of the distance of the icon to the border

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Text

Name	Description
TextColor [▶ 176]	Definition of the text color
TextFontFamily [▶ 173]	Definition of the font using the name of the font family or the keyword
TextFontSize [▶ 173]	Definition of the font size
TextFontSizeUnit [▶ 173]	Definition of the font size unit: • Pixels for an absolute size • Percent for a relative size
TextFontStyle [▶ 174]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
TextFontWeight [▶ 174]	Selecting the font weight of the text: • Normal • Bold: Bold
TextPadding [▶ 176]	Definition of the distance of the text to the border
TextVerticalAlignment [▶ 177]	Selecting the vertical alignment of the text
TextHorizontalAlignment [▶ 175]	Selecting the horizontal alignment of the text
WordWrap [▶ 178]	Selecting whether the text wraps if it is too long for one row

Events

Category: Control

Name	Description
onToggleStateChanged [▶ 393]	The ToggleState value of the button has changed.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onToggleStateChanged [▶ 393]	The ToggleState value of the button has changed.
onStateChanged [▶ 178]	The pressed state has changed.
onStatePressed [▶ 178]	The pressed state has changed from not pressed to pressed.
onStateReleased [▶ 179]	The pressed state has changed from pressed to not pressed.
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Button [▶ 167]

Toggle Button

See also

API reference [▶ 1730] for this control

6.2.19.1 Attributes

6.2.19.1.1 ToggleGroup

User-defined name for a group. Only one control in this group can be active. If another button is pressed with this ToggleGroup, all other buttons are deactivated.

From version 1.10 a ToggleGroup can consist of controls of the type Toggle Button [▶ 386] and Checkbox. [▶ 179]

Please note that this ToggleGroup is global. So if several ToggleButtons are used in one UserControl [▶ 520] and this UserControl is used several times, only one ToggleButton is active. If one ToggleButton should be active per UserControl in this constellation, the attribute of the buttons can be set via one UserControl parameter [▶ 37] per instance of the UserControl. A binding from the UserControlParameter to all ToggleButtons must then be set manually within the UserControl.

Schema: tchmi:general#/definitions/String [▶ 2188]

Origin: Toggle Button [▶ 386]

See also

Attribute getter: getToggleGroup [▶ 1738]

Attribute setter: setToggleGroup [▶ 1738]

Note: Available since version 1.8

6.2.19.1.2 ToggleState

Internal state of the control. This can be Normal or Active.

Scheme: tchmi:framework#/definitions/ToggleState [▶ 2254]

Origin: Toggle Button [▶ 386]

See also

Attribute getter: getToggleState [▶ 1738]

Attribute setter: setToggleState [▶ 1739]

Note: Available since version 1.10

6.2.19.2 Events

6.2.19.2.1 onToggleStateChanged

The ToggleState [▶ 392] value of the button has changed.

Origin: Toggle Button [▶ 386]

Note: Available since version 1.8

6.2.20 Trend Line Chart

A TrendLineChart is a trend element in which curves of historicized symbols can be displayed.

A description of the configuration can be found in chapter Configuration [▶ 400].

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control
GridBackgroundColor [▶ 403]	Definition of the background color of the coordinate system
GridLineColor [▶ 403]	Definition of the color of the coordinate grid
SubgridLineColor [▶ 410]	Definition of the color of the coordinate subgrid
XLabelFontColor [▶ 417]	Definition of the font color of the x-axis labels
XAxisColor [▶ 413]	Definition of the color of the x-axis
XAxisNameFontColor [▶ 415]	Definition of the font color of the X-axis name
MenuBarLabelFontColor [▶ 405]	Definition of the font color of the menu bar
SectionsBackgroundColor [▶ 407]	Definition of the background color of the control
TooltipFontColor [▶ 411]	Definition of the color of the tooltip
TooltipBackgroundColor [▶ 411]	Definition of the background color of the tooltip

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Start [▶ 409]	Definition of the start time at which the display should begin
End [▶ 403]	Definition of the end time at which the display should end
LineGraphDescriptions [▶ 405]	Definition of the historicized variables to be displayed in the graph and their method of display
Interval [▶ 404]	Definition of the interval at which the data are updated
MouseMode [▶ 407]	Selection of the mouse mode that is active after stopping: • ZoomX: By drawing a rectangle, an area can be defined in the X direction which is to be zoomed in on. • ZoomXY: By drawing a rectangle, an area can be defined in the X and Y direction which is to be zoomed in on. • PanX: The observed area can be shifted in the X direction in the display area of the graph by drag-and-drop. • PanXY: The observed area can be shifted in the X and Y direction in the display area of the graph by drag-and-drop.
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: XAxis

Name	Description
ShowXAxis [▶ 408]	Selecting whether the X-axis is to be displayed
XShowLabels [▶ 418]	Selecting whether the X-axis labels are to be displayed
XMainTickSteps [▶ 418]	Definition of the number of main X-axis subdivisions
ShowXSubTicks [▶ 409]	Selecting whether the auxiliary X-axis subdivisions are to be displayed
XSubTickSteps [▶ 418]	Definition of the number of auxiliary X-axis subdivisions
ShowXAxisName [▶ 408]	Selecting whether the X-axis name is to be displayed
XAxisName [▶ 415]	Definition of the X-axis name
XLabelFontFamily [▶ 417]	Definition of the font using the name of the font family or the keyword
XLabelFontSize [▶ 417]	Definition of the font size for the X-axis labels
XLabelFontWeight [▶ 418]	Selecting the font weight of the X-axis labels: • Normal • Bold: bold
XAxisWidth [▶ 416]	Definition of the X-axis width
XAxisNameFontFamily [▶ 415]	Definition of the font using the name of the font family or the keyword
XAxisNameFontSize [▶ 415]	Definition of the font size of the X-axis name
XAxisNameFontWeight [▶ 416]	Selecting of the font weight of the X-axis name: • Normal • Bold: bold
XAxisFormat [▶ 414]	Definition of the formatting of the X-axis labels

Category: YAxis

Name	Description
YAxis [▶ 419]	Definition of one or more Y-axes
ShowYSubTicks [▶ 409]	Selecting whether the auxiliary subdivisions on the Y-axis are to be displayed
YAxisWidth [▶ 419]	Definition of the Y-axis width
YLabelFontFamily [▶ 419]	Definition of the font using the name of the font family or the keyword
YLabelFontSize [▶ 419]	Definition of the font size for the Y-axis labels
YLabelFontSizeUnit [▶ 420]	Selecting the font size unit for the Y-axis labels: • Pixels for an absolute size • Percent for a relative size
YLabelFontWeight [▶ 420]	Selecting the font weight of the Y-axis labels: • Normal • Bold: bold
YMainTickSteps [▶ 420]	Definition of the number of main Y-axis subdivisions
YSubTickSteps [▶ 421]	Definition of the number of auxiliary Y-axis subdivisions

Category: Grid

Name	Description
ShowGrid [▶ 407]	Selecting whether the coordinate grid is to be displayed
GridLineWidth [▶ 404]	Definition of the line width of the coordinate grid
GridShowHorizontalLines [▶ 404]	Selecting whether the horizontal lines of the coordinate grid are to be displayed
GridShowVerticalLines [▶ 404]	Selecting whether the vertical lines of the coordinate grid are to be displayed

Category: Subgrid

Name	Description
SubridLineWidth [▶ 410]	Definition of the line width of the coordinate subgrid system
SubgridShowHorizontalLines [▶ 410]	Selecting whether the horizontal lines of the coordinate subgrid are to be displayed
SubgridShowVerticalLines [▶ 410]	Selecting whether the vertical lines of the coordinate subgrid are to be displayed

Category: Menubar

Name	Description
ShowMenuBar [▶ 408]	Selection of whether the menu bar is to be displayed
MenuBarPosition [▶ 406]	Selection of the position of the menu bar: • Bottom: Underside • Top: Top side
MenuBarLabelFontFamily [▶ 405]	Definition of the font using the name of the font family or the keyword
MenuBarLabelFontSize [▶ 405]	Definition of the font size
MenuBarLabelFontStyle [▶ 406]	Selecting the font style: • Normal • Italic: italic • Oblique: slanted
MenuBarLabelFontWeight [▶ 406]	Selecting the font weight: • Normal • Bold: Bold

Category: Tooltip

Name	Description
ShowTooltip [▶ 408]	Selecting whether the tooltips should be displayed in stop mode
TooltipFormat [▶ 412]	• Definition of the formatting of the tooltips.
TooltipFontFamily [▶ 411]	Definition of the font using the name of the font family or the keyword
TooltipFontSize [▶ 411]	Definition of the font size
TooltipFontWeight [▶ 412]	Selecting the font weight: • Normal • Bold: Bold

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Trend line chart

See also

API reference [▶ 1739] for this control

Note: Available since version 1.10

6.2.20.1 Configuration

A TrendLineChart can be configured as follows:

1. Define the Y-axes of the coordinate system with the help of the dialog "YAxis Definitions" of the attribute "YAxis".

1.1. Define the desired number of Y-axes to which the symbols in the following are to be assigned.

1.2. Adjust the settings of the Y-axis/axes if necessary.

2. Define the variables that are to be displayed in the trend element with the help of the dialog "LineGraphDescriptions" of the attribute "LineGraphDescriptions" of the category "Common".

2.1. Add a Line Graph Description for each of the variables.

2.2. Select the appropriate variable in each description under "Symbol".

> It is only possible to select variables for which historization has been activated beforehand.

2.3. Under "Y-Axis ID", select the Y-axis to which the symbol is to be assigned. If you don't select a Y-axis, an automatically scaling Y-axis bearing the name of the symbol will be generated.

3. Adapt the start and end point of the period to be observed.

Example: Last ten minutes in the past:

3.1. Set the value of the attribute "Start" of the category "Common" to "PT10M".

4. Adapt the X-axis scaling by setting the attribute "XMainTickSteps" of the category "X-Axis" to "5".

5. Activate the attribute "ShowMenubar" of the category "Menubar".

6.2.20.2 Dialogs

6.2.20.2.1 LineGraphDescription dialog

The settings of the graphs and their assignment to the individual symbols that are to be displayed in the TrendLineChart can be changed with this dialog.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Properties	Settings for the Y-axis assignment selected under "Elements"
Symbol	Selection of the historicized symbol
Y-Axis Id	Selection of the Y-axis with which the symbol is to be connected
Line width	Definition of the line width of the graph
Point dot	Selecting whether the individual data points are to be marked
Point dot radius	Definition of the radius of the data points
Point dot stroke width	Definition of the width of the dotted line
Point dot in stopmode	Selection of whether the individual data points are to be drawn when the display is stopped in the trend element.

6.2.20.2.2 YAxis Definition Dialog

This dialog can be used to add and parameterize the different Y-axes.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Properties	Settings of the Y-axis definition selected under "Elements"
Id	Definition of a unique ID of the Y-axis
Position	Position of the Y-axis on the graph • Left side of the graph • Right side of the graph
Main-tick-min-value	Definition of the beginning of the displayed value range
Main-tick-max-value	Definition of the end of the displayed value range
Auto scaling	Selecting whether the value range should be automatically adjusted according to the values to be displayed in the array
Decimal places	Definition of decimal places
Show labels	Selecting whether the label of the Y-axis is to be displayed
Show axis name	Selecting whether the axis name should be displayed next to the axis in the graph
Axis name	Definition of the axis name, which can optionally be displayed next to the axis
Axis name font family	Definition of the font
Axis name font size	Definition of the font size
Axis name font weight	Selecting the font weight • Bold • Normal
Unit	Definition of the unit

6.2.20.3 Attributes

6.2.20.3.1 ActualEndTime

Actual end time (read-only).

Scheme: tchmi:general#/definitions/DateTime [▶ 2178]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getActualEndTime [▶ 1760]

Note: Available since version 1.10

6.2.20.3.2 ActualStartTime

Actual start point (read-only).

Scheme: tchmi:general#/definitions/DateTime [▶ 2178]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getActualStartTime [▶ 1759]

Note: Available since version 1.10

6.2.20.3.3 End

Definition of the end time at which the display should end.

Scheme: tchmi:framework#/definitions/TrendLineChartEnd [▶ 2263]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getEnd [▶ 1759]

Attribute setter: setEnd [▶ 1760]

Note: Available since version 1.10

6.2.20.3.4 GridBackgroundColor

Definition of the background color of the coordinate system.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getGridBackgroundColor [▶ 1750]

Attribute setter: setGridBackgroundColor [▶ 1750]

Note: Available since version 1.10

6.2.20.3.5 GridLineColor

Definition of the color of the coordinate grid.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getGridLineColor [▶ 1751]

Attribute setter: setGridLineColor [▶ 1751]

Note: Available since version 1.10

6.2.20.3.6 GridLineWidth

Definition of the line width of the coordinate grid.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getGridLineWidth [▶ 1790]

Attribute setter: setGridLineWidth [▶ 1791]

Note: Available since version 1.10

6.2.20.3.7 GridShowHorizontalLines

Selecting whether the horizontal lines of the coordinate grid are to be displayed. The number of horizontal grid lines is defined by the attribute YMainTickSteps [▶ 420].

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getGridShowHorizontalLines [▶ 1789]

Attribute setter: setGridShowHorizontalLines [▶ 1790]

Note: Available since version 1.10

6.2.20.3.8 GridShowVerticalLines

Selecting whether the vertical lines of the coordinate grid are to be displayed. The number of vertical grid lines is defined by the attribute XMainTickSteps [▶ 418].

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getGridShowVerticalLines [▶ 1788]

Attribute setter: setGridShowVerticalLines [▶ 1789]

Note: Available since version 1.10

6.2.20.3.9 Interval

Definition of the interval at which the data are updated.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getInterval [▶ 1762]

Attribute setter: setInterval [▶ 1762]

Note: Available since version 1.10

6.2.20.3.10 LineGraphDescription

Definition of the historicized variables to be displayed in the graph and their method of display.

Scheme: tchmi:framework#/definitions/TrendLineChartLineGraphDescriptionDefinitionList [▶ 2263]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getLineGraphDescription [▶ 1761]

Attribute setter: setLineGraphDescription [▶ 1761]

Note: Available since version 1.10

6.2.20.3.11 MenuBarLabelFontColor

Definition of the font color of the menu bar.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarLabelFontColor [▶ 1754]

Attribute setter: setMenuBarLabelFontColor [▶ 1755]

Note: Available since version 1.10

6.2.20.3.12 MenuBarLabelFontFamily

Definition of the font using the name of the font family or the keyword.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarLabelFontFamily [▶ 1796]

Attribute setter: setMenuBarLabelFontFamily [▶ 1796]

Note: Available since version 1.10

6.2.20.3.13 MenuBarLabelFontSize

Definition of the font size.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarLabelFontSize [▶ 1797]

Attribute setter: setMenuBarLabelFontSize [▶ 1797]

Note: Available since version 1.10

6.2.20.3.14 MenuBarLabelFontSizeUnit

Selection of the unit of the font size in the menu bar (pixel or percent).

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarLabelFontSizeUnit [▶ 1798]

Attribute setter: setMenuBarLabelFontSizeUnit [▶ 1798]

Note: Available since version 1.10

6.2.20.3.15 MenuBarLabelFontStyle

Font style (normal, italics: inclined with special characters, oblique: inclined)

Scheme: tchmi:framework#/definitions/FontStyle [▶ 2228]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarLabelFontStyle [▶ 1800]

Attribute setter: setMenuBarLabelFontStyle [▶ 1800]

Note: Available since version 1.10

6.2.20.3.16 MenuBarLabelFontWeight

Selection of the font weight for the menu bar (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarLabelFontWeight [▶ 1799]

Attribute setter: setMenuBarLabelFontWeight [▶ 1799]

Note: Available since version 1.10

6.2.20.3.17 MenuBarPosition

Selection of the position of the menu bar (Bottom, Top).

Scheme: tchmi:framework#/definitions/ChartMenuBarPosition [▶ 2209]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMenuBarPosition [▶ 1795]

Attribute setter: setMenuBarPosition [▶ 1795]

Note: Available since version 1.10

6.2.20.3.18 MouseMode

Selection of the mouse mode that is active after stopping.

- ZoomX: By drawing a rectangle, an area can be defined in the X direction which is to be zoomed in on.
- ZoomXY: By drawing a rectangle, an area can be defined in the X and Y direction which is to be zoomed in on.
- PanX: The observed area can be shifted in the X direction in the display area of the graph by drag-and-drop.
- PanXY: The observed area can be shifted in the X and Y direction in the display area of the graph by drag-and-drop.

Scheme: tchmi:framework#/definitions/ChartMouseMode [▶ 2209]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getMouseMode [▶ 1763]

Attribute setter: setMouseMode [▶ 1763]

Note: Available since version 1.10

6.2.20.3.19 SectionsBackgroundColor

Definition of the background color of the control.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getSectionsBackgroundColor [▶ 1757]

Attribute setter: setSectionsBackgroundColor [▶ 1758]

Note: Available since version 1.10

6.2.20.3.20 ShowGrid

Selecting whether the coordinate grid is to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowGrid [▶ 1787]

Attribute setter: setShowGrid [▶ 1788]

Note: Available since version 1.10

6.2.20.3.21 ShowMenubar

Selection of whether the menu bar is to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowMenubar [▶ 1794]

Attribute setter: setShowMenubar [▶ 1794]

Note: Available since version 1.10

6.2.20.3.22 ShowTooltip

Selecting whether the tooltips should be displayed in stop mode.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowTooltip [▶ 1801]

Attribute setter: setShowTooltip [▶ 1801]

Note: Available since version 1.10

6.2.20.3.23 ShowXAxis

Selecting whether the X-axis is to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowXAxis [▶ 1763]

Attribute setter: setShowXAxis [▶ 1764]

Note: Available since version 1.10

6.2.20.3.24 ShowXAxisName

Selecting whether the X-axis name is to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowXAxisName [▶ 1773]

Attribute setter: setShowXAxisName [▶ 1773]

Note: Available since version 1.10

6.2.20.3.25 ShowXSubTicks

Selecting whether the auxiliary X-axis subdivisions are to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowXSubTicks [▶ 1766]

Attribute setter: setShowXSubTicks [▶ 1767]

Note: Available since version 1.10

6.2.20.3.26 ShowYSubTicks

Selecting whether the auxiliary subdivisions on the Y-axis are to be displayed.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getShowYSubTicks [▶ 1781]

Attribute setter: setShowYSubTicks [▶ 1781]

Note: Available since version 1.10

6.2.20.3.27 Start

Definition of the start time at which the display should begin.

Scheme: tchmi:framework#/definitions/TrendLineChartStart [▶ 2264]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getStart [▶ 1758]

Attribute setter: setStart [▶ 1758]

Note: Available since version 1.10

6.2.20.3.28 SubgridLineColor

Definition of the color of the coordinate subgrid.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getSubgridLineColor [▶ 1752]

Attribute setter: setSubgridlineColor [▶ 1752]

Note: Available since version 1.10

6.2.20.3.29 SubgridLineWidth

Definition of the line width of the coordinate subgrid.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getSubgridLineWidth [▶ 1793]

Attribute setter: setSubgridLineWidth [▶ 1793]

Note: Available since version 1.10

6.2.20.3.30 SubgridShowHorizontalLines

Selecting whether the horizontal lines of the coordinate subgrid are to be displayed. The number of horizontal auxiliary coordinate lines is defined by the attribute YSubTickSteps.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getSubgridShowHorizontalLines [▶ 1792]

Attribute setter: setSubgridShowHorizontalLines [▶ 1793]

Note: Available since version 1.10

6.2.20.3.31 SubgridShowVerticalLines

Selecting whether the vertical lines of the coordinate subgrid are to be displayed. The number of vertical auxiliary coordinate lines is defined by the attribute XSubTickSteps [▶ 418].

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getSubgridShowVerticalLines [▶ 1791]

Attribute setter: setSubgridShowVerticalLines [▶ 1792]

Note: Available since version 1.10

6.2.20.3.32 TooltipBackgroundColor

Definition of the background color of the coordinate system.

Scheme: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipBackgroundColor [▶ 1756]

Attribute setter: setTooltipBackgroundColor [▶ 1757]

Note: Available since version 1.10

6.2.20.3.33 TooltipFontColor

Definition of the font color of the tooltip.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipFontColor [▶ 1755]

Attribute setter: setTooltipFontColor [▶ 1756]

Note: Available since version 1.10

6.2.20.3.34 TooltipFontFamily

Definition of the font using the name of the font family or the keyword.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipFontFamily [▶ 1802]

Attribute setter: setTooltipFontFamily [▶ 1803]

Note: Available since version 1.10

6.2.20.3.35 TooltipFontSize

Definition of the font size.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipFontSize [▶ 1803]

Attribute setter: setTooltipFontSize [▶ 1804]

Note: Available since version 1.10

6.2.20.3.36 TooltipFontSizeUnit

Selecting the unit of the font size in the tooltips (pixel or percent).

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipFontSizeUnit [▶ 1804]

Attribute setter: setTooltipFontSizeUnit [▶ 1805]

Note: Available since version 1.10

6.2.20.3.37 TooltipFontWeight

Selecting the font weight for the tooltips (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipFontWeight [▶ 1805]

Attribute setter: setTooltipFontWeight [▶ 1806]

Note: Available since version 1.10

6.2.20.3.38 TooltipFormat

Definition of the formatting of the tooltips.

Schema: tchmi:framework#/definitions/Function [▶ 2229]

The selected function defines the tooltip display. It is called as soon as the mouse cursor is positioned on a point and/or an error. In addition it contains the current time stamp of the mouse, which is specified in milliseconds, a list of points and a list of errors. The respective list is empty if there are no points or errors at this position.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(actTime: number, points: TcHmiTrendLineChart.TooltipInformationObject[], errors:
TcHmiTrendLineChart.TooltipErrorInformationObject[]): string;
```

Parameter

Name	Type	Description
actTime	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The current time stamp of the mouse position in milliseconds since 01/01/1970.
points	TcHmiTrendLineChart.TooltipInformationObject [▶ 1809][] tchmi:framework#/definitions/TrendLineChartTooltipInformationList [▶ 2265]	The recorded points located at the mouse position (within the point radius).
errors	TcHmiTrendLineChart.TooltipErrorInformationObject [▶ 1809][] tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationList [▶ 2264]	The recorded errors located at the mouse position.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The tooltip to be displayed. HTML is allowed here.

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getTooltipFormat [▶ 1801]

Attribute setter: setTooltipFormat [▶ 1802]

Note: Available since version 1.10

6.2.20.3.39 XAxisColor

Definition of the color of the X-axis.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisColor [▶ 1749]

Attribute setter: setXAxisColor [▶ 1749]

Note: Available since version 1.10

6.2.20.3.40 XAxisFormat

Schema: tchmi:framework#/definitions/Function [▶ 2229]

Definition of the formatting of the X-axis labels.

The selected function defines the X-axis labels. It is called for each displayed value of the main X-axis subdivision. In addition it contains the respective time stamp, which is specified in milliseconds.

A function must fulfill the following signature in order to be offered in the engineering. The parameters mentioned here are set by the control. Further parameters are possible and can be specified per instance in the engineering.

Example: The value for the Linear Gauge is set by the control. The further parameter "maxDecimals" is specified in Engineering for each instance.

[Function]

```
exampleFunction(value: number): string;
```

Parameter

Name	Type	Description
value	number [▶ 2108] tchmi:general#/definitions/Number [▶ 2186]	The current numerical value of the X-axis in milliseconds since 01/01/1970.

Return value

Type	Description
string [▶ 2108] tchmi:general#/definitions/String [▶ 2188]	The new value of the main X-axis subdivision. HTML is not allowed here.

Origin: Trend line chart [▶ 393]

See also

Attribute getter: getXAxisFormat [▶ 1778]

Attribute setter: setXAxisFormat [▶ 1779]

Note: Available since version 1.10

6.2.20.3.41 XAxisName

Definition of the X-axis name.

Scheme: tchmi:general#/definitions/String [▶ 2188]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisName [▶ 1774]

Attribute setter: setXAxisName [▶ 1774]

Note: Available since version 1.10

6.2.20.3.42 XAxisNameFontColor

Definition of the font color of the X-axis name.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisNameFontColor [▶ 1752]

Attribute setter: setXAxisNameFontColor [▶ 1753]

Note: Available since version 1.10

6.2.20.3.43 XAxisNameFontFamily

Definition of the font using the name of the font family or the keyword.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisNameFontFamily [▶ 1775]

Attribute setter: setXAxisNameFontFamily [▶ 1775]

Note: Available since version 1.10

6.2.20.3.44 XAxisNameFontSize

Definition of the font size of the X-axis name.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisNameFontSize [▶ 1775]

Attribute setter: setXAxisNameFontSize [▶ 1776]

Note: Available since version 1.10

6.2.20.3.45 XAxisNameFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisNameFontSizeUnit [▶ 1776]

Attribute setter: setXAxisNameFontSizeUnit [▶ 1777]

Note: Available since version 1.10

6.2.20.3.46 XAxisNameFontWeight

Selection of the font weight of the X-axis name (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisNameFontWeight [▶ 1777]

Attribute setter: setXAxisNameFontWeight [▶ 1778]

Note: Available since version 1.10

6.2.20.3.47 XAxisWidth

Definition of the axis width.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXAxisWidth [▶ 1772]

Attribute setter: setXAxisWidth [▶ 1772]

Note: Available since version 1.10

6.2.20.3.48 XLabelFontColor

Definition of the color of the X-axis labels.

Scheme: tchmi:framework#/definitions/SolidColor [▶ 2253]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXLabelFontColor [▶ 1753]

Attribute setter: setXLabelFontColor [▶ 1754]

Note: Available since version 1.10

6.2.20.3.49 XLabelFontFamily

Definition of the font using the name of the font family or the keyword.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXLabelFontFamily [▶ 1768]

Attribute setter: setXLabelFontFamily [▶ 1769]

Note: Available since version 1.10

6.2.20.3.50 XLabelFontSize

Definition of the font size for the X-axis labels.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXLabelFontSize [▶ 1769]

Attribute setter: setXLabelFontSize [▶ 1769]

Note: Available since version 1.10

6.2.20.3.51 XLabelFontSizeUnit

Pixels, or percent for relative sizes.

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXLabelFontSizeUnit [▶ 1770]

Attribute setter: setXLabelFontSizeUnit [▶ 1770]

Note: Available since version 1.10

6.2.20.3.52 XLabelFontWeight

Selection of the font weight of the X-axis labels (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXLabelFontWeight [▶ 1771]

Attribute setter: setXLabelFontWeight [▶ 1771]

Note: Available since version 1.10

6.2.20.3.53 XMainTickSteps

Definition of the number of main X-axis subdivisions.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXMainTickSteps [▶ 1765]

Attribute setter: setXMainTickSteps [▶ 1766]

Note: Available since version 1.10

6.2.20.3.54 XShowLabels

Selecting whether the X-axis labels are to be displayed

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXShowLabels [▶ 1764]

Attribute setter: setXShowLabels [▶ 1765]

Note: Available since version 1.10

6.2.20.3.55 XSubTickSteps

Definition of the number of auxiliary X-axis subdivisions.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getXSubTickSteps [▶ 1767]

Attribute setter: setXSubTickSteps [▶ 1768]

Note: Available since version 1.10

6.2.20.3.56 YAxis

Definition of one or more Y-axes.

Scheme: tchmi:framework#/definitions/ChartYAxisDefinitionList [▶ 2212]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYAxis [▶ 1779]

Attribute setter: setYAxis [▶ 1780]

Note: Available since version 1.10

6.2.20.3.57 YAxisWidth

Definition of the Y-axis width.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYAxisWidth [▶ 1787]

Attribute setter: setYAxisWidth [▶ 1787]

Note: Available since version 1.10

6.2.20.3.58 YLabelFontFamily

Definition of the font using the name of the font family or the keyword.

Scheme: tchmi:framework#/definitions/FontFamily [▶ 2227]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYLabelFontFamily [▶ 1783]

Attribute setter: setYLabelFontFamily [▶ 1783]

Note: Available since version 1.10

6.2.20.3.59 YLabelFontSize

Definition of the font size for the Y-axis labels.

Scheme: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYLabelFontSize [▶ 1784]

Attribute setter: setYLabelFontSize [▶ 1784]

Note: Available since version 1.10

6.2.20.3.60 YLabelFontSizeUnit

Selection of the unit of the font size of the Y-axis labels (pixel or percent).

Scheme: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYLabelFontSizeUnit [▶ 1785]

Attribute setter: setYLabelFontSizeUnit [▶ 1785]

Note: Available since version 1.10

6.2.20.3.61 YLabelFontWeight

Selection of the font weight of the Y-axis labels (normal, bold).

Scheme: tchmi:framework#/definitions/FontWeight [▶ 2228]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYLabelFontWeight [▶ 1786]

Attribute setter: setYLabelFontWeight [▶ 1786]

Note: Available since version 1.10

6.2.20.3.62 YMainTickSteps

Definition of the number of main Y-axis subdivisions.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYMainTickSteps [▶ 1780]

Attribute setter: setYMainTickSteps [▶ 1781]

Note: Available since version 1.10

6.2.20.3.63 YSubTickSteps

Definition of the number of auxiliary Y-axis subdivisions.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Trend Line Chart [▶ 393]

See also

Attribute getter: getYSubTickSteps [▶ 1782]

Attribute setter: setYSubTickSteps [▶ 1782]

Note: Available since version 1.10

6.2.20.4 Themed Resources

These resources cannot be defined per instance, but only for several/all trend line charts via the theme file [▶ 565]. If you really want to change only one instance, you must create a class [▶ 562] for it.

Note: Available since version 1.10

6.2.20.4.1 YAxisGenerated

This is the axis definition that is used for Y-axes if nothing is defined in the attribute YAxis [▶ 419].

Scheme: tchmi:framework#/definitions/ChartYAxisDefinition [▶ 2209]

Origin: Trend Line Chart [▶ 393]

Note: Available since version 1.10

6.2.21 Video

Video can be played with the video control. The supported video formats are defined by the browser used.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 427]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the upper and lower sides of the parent element. The 'Width' attribute is ignored in all calculations! • Content: The width is calculated from the height while maintaining the aspect ratio of the video. If the attributes 'Width Mode' and 'Height Mode' are set to Content, the video is drawn in its original size. The 'Width' attribute is ignored in all calculations!
HeightMode [▶ 427]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the upper and lower side of the parent element. The 'Height' attribute is ignored in all calculations!

Name	Description
	• Content: The height is calculated from the width while maintaining the aspect ratio of the video. If the attributes 'Width Mode [▶ 423]' and 'Height Mode' are set to Content, the video is drawn in its original size. The 'Height' attribute is ignored in all calculations!
OriginalWidth [▶ 430]	Plays back the original width of the video or 0 if it is not available.
OriginalHeight [▶ 430]	Plays back the original height of the video or 0 if it is not available.

Category: Common

Name	Description
SrcList [▶ 428]	Definition of a list of videos that can be played back
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Category: Video

Name	Description
Volume [▶ 428]	Definition of the volume of the video
Mute [▶ 428]	Selecting whether the sound is to be switched off
Playbackrate [▶ 429]	Definition of a playback rate. The normal playback speed is multiplied by this value to obtain the current rate.
CurrentTime [▶ 429]	Definition of the current playback time of the video
Duration [▶ 430]	Plays the length of the video in seconds or 0 if it is not available.
isEnded [▶ 429]	Returns a Boolean value indicating whether playback has finished or not.
isPlaying [▶ 429]	Returns a Boolean value indicating whether the video is currently playing or not.

Category: Player

Name	Description
Controls [▶ 430]	Selecting whether the integrated control elements are to be displayed
Autoplay [▶ 431]	Selecting whether the video should be started automatically
Loop [▶ 431]	Selecting whether the video is to be repeated automatically on a permanent basis
Poster [▶ 431]	Definition of a startup image to be displayed until the operator starts or rewinds the video. If nothing is specified here, nothing is displayed until the first image of the video has been loaded.

Events

Category: Control

Name	Description
onLoadedData [▶ 432]	The video was loaded successfully.
onError [▶ 432]	The video was not loaded successfully.
onVolumeChange [▶ 432]	The volume of the video has been changed.
onDurationChange [▶ 432]	The duration of the video has changed.
onIsEndedChange [▶ 432]	The video was terminated.
onIsPlayingChange [▶ 432]	The video has changed its playback status.
onTimeUpdate [▶ 432]	The playback position has been changed.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Video

See also

API reference [▶ 1812] for this control

6.2.21.1 Attributes

6.2.21.1.1 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The Height attribute is ignored in all calculations!
- **Content:** The height is calculated from the width while maintaining the aspect ratio of the video. If Width Mode [▶ 427] and *Height Mode* are set to Content, the video is drawn in its original size.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Scheme: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Video [▶ 421]

See also

Attribute getter: getHeightMode [▶ 1818]

Attribute setter: setHeightMode [▶ 1818]

Note: Available since version 1.8

6.2.21.1.2 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!
- **Content:** The width is calculated from the height while maintaining the aspect ratio of the video. If *Width Mode* and Height Mode [▶ 427] are set to Content, the video is drawn in its original size.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Scheme: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Video [▶ 421]

See also

Attribute getter: getWidthMode [▶ 1819]

Attribute setter: setWidthMode [▶ 1819]

Note: Available since version 1.8

6.2.21.1.3 SrcList

A list of videos that the browser should play.

The videos are **not** played one after the other, but the browser tries to play each video until it finds a format that it supports.

If the format type is not set, the browser tries to detect the code itself.

You can check the current status of browser support on the external website http://caniuse.com/.

Scheme: tchmi:framework#/definitions/VideoSourceList [▶ 2267]

Origin: Video [▶ 421]

See also

Attribute getter: getSrcList [▶ 1820]

Attribute setter: setSrcList [▶ 1820]

Note: Available since version 1.8

6.2.21.1.4 Volume

Volume of the video. The number 0 sets the video to mute while 1 means full volume.

Scheme: tchmi:framework#/definitions/Fraction [▶ 2229]

Origin: Video [▶ 421]

See also

Attribute getter: getVolume [▶ 1821]

Attribute setter: setVolume [▶ 1821]

Note: Available since version 1.8

6.2.21.1.5 Mute

Determines whether the sound is switched off.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Video [▶ 421]

See also

Attribute getter: getMute [▶ 1822]

Attribute setter: setMute [▶ 1822]

Note: Available since version 1.8

6.2.21.1.6 Playbackrate

The normal playback speed is multiplied by this value to obtain the current rate. Thus, the value 1.0 represents the normal speed.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Video [▶ 421]

See also

Attribute getter: getPlaybackrate [▶ 1823]

Attribute setter: setPlaybackrate [▶ 1823]

Note: Available since version 1.8

6.2.21.1.7 isEnded

Returns a Boolean value indicating whether playback has finished or not.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Video [▶ 421]

See also

Attribute getter: getIsEnded [▶ 1824]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.21.1.8 isPlaying

Returns a Boolean value indicating whether the video is currently playing or not.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Video [▶ 421]

See also

Attribute getter: getIsPlaying [▶ 1824]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.21.1.9 CurrentTime

The current playback time of the video.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Video [▶ 421]

See also

Attribute getter: getCurrentTime [▶ 1825]

Attribute setter: setCurrentTime [▶ 1825]

Note: Available since version 1.8

6.2.21.1.10 Duration

Plays the length of the video in seconds or 0 if it is not available.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Video [▶ 421]

See also

Attribute getter: getDuration [▶ 1825]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.21.1.11 OriginalWidth

Plays back the original width of the video or 0 if it is not available.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Video [▶ 421]

See also

Attribute getter: getOriginalWidth [▶ 1826]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.21.1.12 OriginalHeight

Plays back the original height of the video or 0 if it is not available.

Scheme: tchmi:general#/definitions/Number [▶ 2186]

Origin: Video [▶ 421]

See also

Attribute getter: getOriginalHeight [▶ 1826]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.2.21.1.13 Controls

Determines whether control elements are to be displayed. The user can control playback, adjust the volume and pause/resume the video.

> Some browsers do not allow automatic [▶ 431] or application-based [▶ 1830] start of a video. You should therefore carry out intensive tests to determine whether it makes sense to deactivate the control elements.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Video [▶ 421]

See also

Attribute getter: getControls [▶ 1827]

Attribute setter: setControls [▶ 1827]

Note: Available since version 1.8

6.2.21.1.14 Autoplay

Specifies whether the video is to be started directly.

Caution: In some browsers this is not allowed by default. In part this is possible again when the sound is switched off [▶ 428].

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Video [▶ 421]

See also

Attribute getter: getAutoplay [▶ 1828]

Attribute setter: setAutoplay [▶ 1828]

Note: Available since version 1.8

6.2.21.1.15 Loop

When the video is finished, it starts again from the beginning.

Scheme: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Video [▶ 421]

See also

Attribute getter: getLoop [▶ 1829]

Attribute setter: setLoop [▶ 1829]

Note: Available since version 1.8

6.2.21.1.16 Poster

A URL representing an individual image that is displayed until the user plays or winds the video.

If this attribute is not set, nothing is displayed until the first video frame is available. From this point in time, this first frame is used.

Schema: tchmi:general#/definitions/String [▶ 2188]

Origin: Video [▶ 421]

See also

Attribute getter: getPoster [▶ 1830]

Attribute setter: setPoster [▶ 1830]

Note: Available since version 1.8

6.2.21.2 Events

6.2.21.2.1 onError

The video was not loaded successfully.

Origin: Video [▶ 421]

Note: Available since version 1.8

6.2.21.2.2 onLoadedData

The video was loaded successfully.

Origin: Video [▶ 421]

Note: Available since version 1.8

6.2.21.2.3 onVolumeChange

The volume [▶ 428] of the video has been changed.

Origin: Video [▶ 421]

Note: Available since version 1.8

6.2.21.2.4 onDurationChange

The duration of the video [▶ 430] has changed.

Origin: Video [▶ 421]

Note: Available since version 1.8

6.2.21.2.5 onIsEndedChange

The video was terminated [▶ 429].

Origin: Video [▶ 421]

Note: Available since version 1.8

6.2.21.2.6 onIsPlayingChange

The video has changed its playback status [▶ 429].

Origin: Video [▶ 421]

Note: Available since version 1.8

6.2.21.2.7 onTimeUpdate

The playback position [▶ 429] has been changed.

Origin: Video [▶ 421]

Note: Available since version 1.8

6.3 System Controls

Some display names of attributes have been renamed from version 1.10.

Name	Description
Control [▶ 433]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. Base class of all controls.
Container Control [▶ 459]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. Base class for all controls that can contain controls.
Container [▶ 465]	Container for other controls.
Content [▶ 470]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. A reusable container that can be loaded in a region.
Grid [▶ 475]	Layout grid for controls that can react fluidly and responsively.
HTML Host [▶ 486]	Container for HTML content. Can be edited in the HTML editor of the Designer.
Partial [▶ 490]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. A base class for reusable control containers.
Region [▶ 496]	Container for content controls. (created via "Add new file" in Solution View).
User Control [▶ 504]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. A reusable collection of controls.
User Control Host [▶ 509]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. Container for user controls.
View [▶ 515]	No instance of this class can be created. This page serves only as a list of the attributes that this class provides. Top-level control for a TcHmi visualization.

6.3.1 Control

No instance of this class can be created. This page serves only as a list of the attributes that this class provides. It is the basic class of all controls.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control
IsEnabled [▶ 439]	Selection of whether a control should be activated

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control

See also

API reference [▶ 1118] for this control

6.3.1.1 Attributes

6.3.1.1.1 Id

Name of the control.

Schema: tchmi:framework#/definitions/Id [▶ 2238]

Origin: Control [▶ 433]

See also

Attribute getter: getId [▶ 1123]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.3.1.1.2 IsEnabled

Activation status of a control. Only active controls react to mouse and touch.

Schema: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Control [▶ 433]

See also

Attribute getter: getIsEnabled [▶ 1124]

Attribute setter: setIsEnabled [▶ 1124]

Note: Available since version 1.8

6.3.1.1.3 ClassNames

Definition of the class(es) of the control. These classes can be addressed within the theme.

Schema: tchmi:framework#/definitions/ClassNameList [▶ 2213]

Origin: Control [▶ 433]

See also

Attribute getter: getClassNames [▶ 1126]

Attribute setter: setClassNames [▶ 1127]

Note: Available since version 1.10

6.3.1.1.4 IsAttached

Indicates the status regarding whether the control is currently logically integrated inside the DOM.

Schema: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Control [▶ 433]

See also

Attribute getter: getIsAttached [▶ 1127]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.3.1.1.5 Left

Definition of the distance from the left side to the left side of the surrounding control. This value may be empty if Width [▶ 441] and Right [▶ 443] are filled out.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getLeft [▶ 1128]

Attribute setter: setLeft [▶ 1128]

Note: Available since version 1.8

6.3.1.1.6 LeftUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getLeftUnit [▶ 1129]

Attribute setter: setLeftUnit [▶ 1129]

Note: Available since version 1.8

6.3.1.1.7 Top

Definition of the distance from the top side to the top side of the surrounding control. This value may be empty if Height [▶ 442] and Bottom [▶ 443] are filled out.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getTop [▶ 1130]

Attribute setter: setTop [▶ 1130]

Note: Available since version 1.8

6.3.1.1.8 TopUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getTopUnit [▶ 1131]

Attribute setter: setTopUnit [▶ 1131]

Note: Available since version 1.8

6.3.1.1.9 Width

External width of the control. Left [▶ 440] and Right [▶ 443]are used to calculate the size if this value is empty. This value is only interpreted if the attribute Width Mode [▶ 441] has the value Value.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getWidth [▶ 1133]

Attribute setter: setWidth [▶ 1134]

Note: Available since version 1.8

6.3.1.1.10 WidthUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getWidthUnit [▶ 1134]

Attribute setter: setWidthUnit [▶ 1135]

Note: Available since version 1.8

6.3.1.1.11 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeMode [▶ 2252]

Origin: Control [▶ 433]

See also

Attribute getter: getWidthMode [▶ 1136]

Attribute setter: setWidthMode [▶ 1137]

Note: Available since version 1.8

6.3.1.1.12 Height

External height of the control. Top [▶ 440] and Bottom [▶ 443] attributes are used to calculate the size if this value is empty. This value is only interpreted if the attribute Height Mode [▶ 442] has the value Value.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getHeight [▶ 1132]

Attribute setter: setHeight [▶ 1132]

Note: Available since version 1.8

6.3.1.1.13 HeightUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getHeightUnit [▶ 1132]

Attribute setter: setHeightUnit [▶ 1133]

6.3.1.1.14 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeMode [▶ 2252]

Origin: Control [▶ 433]

See also

Attribute getter: getHeightMode [▶ 1135]

Attribute setter: setHeightMode [▶ 1136]

Note: Available since version 1.8

6.3.1.1.15 Right

Definition of the distance from the right side to the right side of the surrounding control. This value is only used if one of the two attributes Left [▶ 440] and Width [▶ 441]is not used.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRight [▶ 1139]

Attribute setter: setRight [▶ 1139]

Note: Available since version 1.8

6.3.1.1.16 RightUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getRightUnit [▶ 1140]

Attribute setter: setRightUnit [▶ 1140]

Note: Available since version 1.8

6.3.1.1.17 Bottom

Definition of the distance from the bottom side to the bottom side of the surrounding control. This value is only used if one of the two attributes Top [▶ 440] and Height [▶ 442] is not used.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getBottom [▶ 1137]

Attribute setter: setBottom [▶ 1138]

Note: Available since version 1.8

6.3.1.1.18 BottomUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getBottomUnit [▶ 1138]

Attribute setter: setBottomUnit [▶ 1138]

Note: Available since version 1.8

6.3.1.1.19 RenderedLeft

Returns the calculated distance in pixels between the left side and the left side of the higher-level control. This value always returns a value, irrespective of the positioning of a control.

This value very probably has rounding inaccuracies.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRenderedLeft [▶ 1148]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.3.1.1.20 RenderedTop

Returns the calculated distance in pixels between the top side and the top side of the higher-level control. This value always returns a value, irrespective of the positioning of a control.

This value very probably has rounding inaccuracies.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRenderedTop [▶ 1149]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.3.1.1.21 RenderedBottom

Returns the calculated distance in pixels between the bottom side and the bottom side of the higher-level control. This value always returns a value, irrespective of the positioning of a control.

This value very probably has rounding inaccuracies.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRenderedBottom [▶ 1149]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.3.1.1.22 RenderedRight

Returns the calculated distance in pixels between the right side and the right side of the higher-level control. This value always returns a value, irrespective of the positioning of a control.

This value very probably has rounding inaccuracies.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRenderedRight [▶ 1149]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.3.1.1.23 RenderedWidth

Returns the calculated width of the control in pixels. This value always returns a value, irrespective of the positioning of a control.

This value very probably has rounding inaccuracies.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRenderedWidth [▶ 1150]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.3.1.1.24 RenderedHeight

Returns the calculated height of the control in pixels. This value always returns a value, irrespective of the positioning of a control.

This value very probably has rounding inaccuracies.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getRenderedHeight [▶ 1150]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.10

6.3.1.1.25 GridColumnIndex

Definition of the column of a grid [▶ 475] in which the control is to be displayed.

The attribute has no effect if the control is not in such a container.

Schema: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Control [▶ 433]

See also

Attribute getter: getGridColumnIndex [▶ 1126]

Attribute setter: setGridColumnIndex [▶ 1126]

Note: Available since version 1.8

6.3.1.1.26 GridRowIndex

Definition of the row of a grid [▶ 475] in which the control is to be displayed.

The attribute has no effect if the control is not in such a container.

Schema: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Control [▶ 433]

See also

Attribute getter: getGridRowIndex [▶ 1125]

Attribute setter: setGridRowIndex [▶ 1125]

Note: Available since version 1.8

6.3.1.1.27 BorderColor

Border color

A border is only displayed if Border Color, Border Width [▶ 447] and Border Style [▶ 447] have been set.

Schema: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Control [▶ 433]

See also

Attribute getter: getBorderColor [▶ 1151]

Attribute setter: setBorderColor [▶ 1151]

Note: Available since version 1.8

Note: Only SolidColor is possible before TcHmi version 1.10.

6.3.1.1.28 BorderWidth

Definition of the border width of the four sides of the control.

A border is only displayed if Border Color [▶ 446], Border Width and Border Style [▶ 447] have been set.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/BorderWidth [▶ 2197]

Origin: Control [▶ 433]

See also

Attribute getter: getBorderWidth [▶ 1152]

Attribute setter: setBorderWidth [▶ 1152]

Note: Available since version 1.8

6.3.1.1.29 BorderStyle

Selection of the border style of the four sides of the control. The possible settings are none, solid line, dotted line or dashed line

A border is only displayed if Border Color, Border Width [▶ 447] and Border Style have been set.

Schema: tchmi:framework#/definitions/BorderStyle [▶ 2196]

Origin: Control [▶ 433]

See also

Attribute getter: getBorderStyle [▶ 1153]

Attribute setter: setBorderStyle [▶ 1153]

Note: Available since version 1.8

6.3.1.1.30 BorderRadius

Definition of the radius of the four corners of a border to round them off.

Schema: tchmi:framework#/definitions/BorderRadius [▶ 2194]

Origin: Control [▶ 433]

See also

Attribute getter: getBorderRadius [▶ 1154]

Attribute setter: setBorderRadius [▶ 1154]

Note: Available since version 1.8

6.3.1.1.31 BackgroundColor

Definition of the background color. This is displayed if no background image or a transparent background image is set.

Schema: tchmi:framework#/definitions/Color [▶ 2213]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundColor [▶ 1154]

Attribute setter: setBackgroundColor [▶ 1155]

Note: Available since version 1.8

6.3.1.1.32 BackgroundImage

An image that is above the background color.

Schema: tchmi:framework#/definitions/Path [▶ 2248]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImage [▶ 1155]

Attribute setter: setBackgroundImage [▶ 1156]

Note: Available since version 1.8

6.3.1.1.33 BackgroundImageWidth

Numerical value of the width of the background image.

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImageWidth [▶ 1157]

Attribute setter: setBackgroundImageWidth [▶ 1158]

Note: Available since version 1.8

6.3.1.1.34 BackgroundImageWidthUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImageWidthUnit [▶ 1158]

Attribute setter: setBackgroundImageWidthUnit [▶ 1159]

Note: Available since version 1.8

6.3.1.1.35 BackgroundImageHeight

Numerical value of the height of the background image.

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImageHeight [▶ 1159]

Attribute setter: setBackgroundImageHeight [▶ 1160]

Note: Available since version 1.8

6.3.1.1.36 BackgroundImageHeightUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImageHeightUnit [▶ 1156]

Attribute setter: setBackgroundImageHeightUnit [▶ 1157]

Note: Available since version 1.8

6.3.1.1.37 BackgroundImagePadding

Definition of an additional distance between the border of the control and background image for the four sides.

Schema: tchmi:framework#/definitions/Padding [▶ 2246]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImagePadding [▶ 1162]

Attribute setter: setBackgroundImagePadding [▶ 1162]

Note: Available since version 1.8

6.3.1.1.38 BackgroundImageHorizontalAlignment

Definition of the alignment of the background image within the control

Schema: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImageHorizontalAlignment [▶ 1160]

Attribute setter: setBackgroundImageHorizontalAlignment [▶ 1160]

Note: Available since version 1.8

6.3.1.1.39 BackgroundImageVerticalAlignment

Definition of the alignment of the background image within the control

Schema: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Origin: Control [▶ 433]

See also

Attribute getter: getBackgroundImageVerticalAlignment [▶ 1161]

Attribute setter: setBackgroundImageVerticalAlignment [▶ 1161]

Note: Available since version 1.8

6.3.1.1.40 MinHeight

Definition of the minimum height if the control size is defined as dynamic.

Maximum Height [▶ 450] overwrites the calculated height, but *Minimum Height* overwrites *Maximum Height*.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getMinHeight [▶ 1143]

Attribute setter: setMinHeight [▶ 1143]

Note: Available since version 1.8

6.3.1.1.41 MinHeightUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getMinHeightUnit [▶ 1144]

Attribute setter: setMinHeightUnit [▶ 1144]

Note: Available since version 1.8

6.3.1.1.42 MaxHeight

Definition of the maximum height if the control size is defined as dynamic.

Maximum Height overwrites the calculated height, but Minimum Height [▶ 450] overwrites *Maximum Height*.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getMaxHeight [▶ 1146]

Attribute setter: setMaxHeight [▶ 1147]

Note: Available since version 1.8

6.3.1.1.43 MaxHeightUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getMaxHeightUnit [▶ 1147]

Attribute setter: setMaxHeightUnit [▶ 1148]

Note: Available since version 1.8

6.3.1.1.44 MaxWidth

Definition of the maximum width if the control size is defined as dynamic.

Maximum Width overwrites the calculated width, but Minimum Width [▶ 452] overwrites *Maximum Width*.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getMaxWidth [▶ 1144]

Attribute setter: setMaxWidth [▶ 1145]

Note: Available since version 1.8

6.3.1.1.45 MaxWidthUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getMaxWidthUnit [▶ 1145]

Attribute setter: setMaxWidthUnit [▶ 1146]

Note: Available since version 1.8

6.3.1.1.46 MinWidth

Definition of the minimum width if the control size is defined as dynamic.

Maximum Width overwrites the calculated width, but Minimum Width [▶ 451] overwrites Maximum Width.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/MeasurementValue [▶ 2243]

Origin: Control [▶ 433]

See also

Attribute getter: getMinWidth [▶ 1141]

Attribute setter: setMinWidth [▶ 1141]

Note: Available since version 1.8

6.3.1.1.47 MinWidthUnit

Pixels, or percent for relative positioning.

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Origin: Control [▶ 433]

See also

Attribute getter: getMinWidthUnit [▶ 1142]

Attribute setter: setMinWidthUnit [▶ 1142]

Note: Available since version 1.8

6.3.1.1.48 Opacity

Transparency of the entire control.

The number 0 sets the control to fully transparent, while 1 means fully visible.

Schema: tchmi:framework#/definitions/Fraction [▶ 2229]

Origin: Control [▶ 433]

See also

Attribute getter: getOpacity [▶ 1164]

Attribute setter: setOpacity [▶ 1164]

Note: Available since version 1.8

6.3.1.1.49 Tooltip

Tool tip for the control.

This is displayed for a short time when the mouse is held above a control.

Schema: tchmi:general#/definitions/String [▶ 2188]

Origin: Control [▶ 433]

See also

Attribute getter: getTooltip [▶ 1166]

Attribute setter: setTooltip [▶ 1167]

Note: Available since version 1.8

6.3.1.1.50 Transform

Transforms a control.

The control can be rotated, scaled, translated or skewed.

Furthermore, the origin of the transformation and the perspective can be set once.

Schema: tchmi:framework#/definitions/TransformList [▶ 2255]

Origin: Control [▶ 433]

See also

Attribute getter: getTransform [▶ 1165]

Attribute setter: setTransform [▶ 1165]

Note: Available since version 1.8

6.3.1.1.51 Type

Type name of the control.

Schema: tchmi:framework#/definitions/ControlType [▶ 2217]

Origin: Control [▶ 433]

See also

Attribute getter: getType [▶ 1168]

The attribute can only be read and therefore has no setter.

Note: Available since version 1.8

6.3.1.1.52 Visibility

Visibility of a control.

- The default value **Visible** displays the control directly.
- If the setting is **Hide**, the control continues to react to the mouse or to touches. It is also involved in the size calculations with fluid designs.
- If **Collapsed** is used, the control doesn't use up any space, not does it receive any mouse interaction or touches.

Schema: tchmi:framework#/definitions/Visibility [▶ 2268]

Origin: Control [▶ 433]

See also

Attribute getter: getVisibility [▶ 1163]

Attribute setter: setVisibility [▶ 1163]

Note: Available since version 1.8

6.3.1.1.53 ZIndex

Definition of the position on the Z-axis.

The following applies: The higher the entry, the more the control is in the foreground. A control with a larger Z index hides a control with a smaller Z index.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:general#/definitions/UDINT [▶ 2190]

Origin: Control [▶ 433]

See also

Attribute getter: getZindex [▶ 1166]

Attribute setter: setZindex [▶ 1166]

Note: Available since version 1.8

6.3.1.1.54 BoxShadow

Definition of a box shadow.

Schema: tchmi:framework#/definitions/BoxShadowList [▶ 2197]

Origin: Control [▶ 433]

See also

Attribute getter: getBoxShadow [▶ 1167]

Attribute setter: setBoxShadow [▶ 1168]

Note: Available since version 1.8

6.3.1.1.55 AccessConfig

Access rights for the Control. For details see also Authorization System/Control Level [▶ 527].

Schema: tchmi:framework#/definitions/AccessConfig [▶ 2192]

Origin: Control [▶ 433]

See also

Attribute getter: getAccessConfig [▶ 1169]

Attribute setter: setAccessConfig [▶ 1169]

Note: Available since version 1.8

6.3.1.2 Events

6.3.1.2.1 onAttached

The control was added to the display.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.2 onDetached

The control was removed from the display.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.3 onInitialized

The control was initialized. Not all of the values of the binding are available yet.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.4 onMouseClick

The left mouse button was clicked over the control and released.

This event is also triggered by interaction with a touch screen. This can be prevented by enabling 'preventDefault' [▶ 50] on the corresponding touch event.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.5 onMouseDoubleClick

The left mouse button was clicked twice over the control and released.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.6 onMouseDown

A mouse button was depressed over the control.

This event is also triggered by interaction with a touch screen. This can be prevented by enabling 'preventDefault' [▶ 50] on the corresponding touch event.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.7 onMouseDownLeft

The left mouse button was depressed over the control.

This event is also triggered by interaction with a touch screen. This can be prevented by enabling 'preventDefault' [▶ 50] on the corresponding touch event.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.8 onMouseDownRight

The right mouse button was depressed over the control.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.9 onMouseEnter

The mouse pointer was moved over the control.

onMouseLeave becomes active on leaving the control.

As opposed to onMouseOver [▶ 457], this event is called only once on reaching the control, regardless of the child control.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.10 onMouseLeave

The mouse pointer has left the control.

onMouseEnter [▶ 456] is called on reaching the control.

As opposed to onMouseOut [▶ 456], this event is called only once on leaving the control, regardless of the child control.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.11 onMouseMove

The mouse was moved over the control.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.12 onMouseOut

The mouse pointer has left the control or one of its child controls.

The event onMouseLeave [▶ 456] observes only the control itself.

onMouseOver [▶ 457] becomes active on reaching the control or one of its children.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.13 onMouseOver

The mouse pointer was moved over the control or one of its children. The event onMouseEnter [▶ 456] observes only the control itself.

onMouseOut [▶ 456] becomes active on leaving the control or one of its children.

Origin: Control [▶ 433]

6.3.1.2.14 onMouseRightClick

The right mouse button was clicked over the control and released.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.15 onMouseUp

A mouse button was released over the control.

This event is also triggered by interaction with a touch screen. This can be prevented by enabling 'preventDefault' [▶ 50] on the corresponding touch event.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.16 onMouseUpLeft

The left mouse button was released over the control.

This event is also triggered by interaction with a touch screen. This can be prevented by enabling 'preventDefault' [▶ 50] on the corresponding touch event.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.17 onMouseUpRight

The right mouse button was released over the control.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.18 onMouseWheel

The mouse wheel was moved over the control.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.19 onMoved

The control was moved in relation to the browser window.

Note: Learn how controls can be positioned [▶ 134].

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.20 onPressed

A mouse click was executed or a touch screen was briefly touched.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.21 onResized

The size of the control was changed.

Note: Learn how the size of controls can be defined [▶ 134].

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.22 onTouchCancel

An interaction with a touch screen was aborted.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.23 onTouchEnd

A finger/stylus was lifted off a touch screen.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.24 onTouchMove

A finger/stylus was moved on a touch screen.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.2.25 onTouchStart

A finger/stylus was placed against a touch screen.

Origin: Control [▶ 433]

Note: Available since version 1.8

6.3.1.3 Permissions

6.3.1.3.1 observe

Determines whether the control is generally observable.

If this right is missing, the control is not displayed.

Origin: Control [▶ 433]

See also

See also User management at control level [▶ 527].

Note: Available since version 1.8

6.3.1.3.2 operate

Determines whether the control is generally operable.

If the main function of a control is the interaction (e.g. button [▶ 167]), a light-colored overlay is drawn if the right is missing.

Origin: Control [▶ 433]

See also

See also User management at control level [▶ 527].

Note: Available since version 1.8

6.3.2 Container Control

No instance of this class can be created. This page serves only as a list of the attributes that this class provides. It is the basic class for all controls that can contain controls.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 464]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations! • Content: The height is defined by the width of the content. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 464]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the height of the content. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control
IsEnabled [▶ 464]	Selection of whether the control is enabled

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Container Control

See also

API reference [▶ 1174] for this control

6.3.2.1 Attributes

6.3.2.1.1 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Container Control [▶ 459]

See also

Attribute getter: getHeightMode [▶ 1179]

Attribute setter: setHeightMode [▶ 1180]

Note: Available since version 1.8

6.3.2.1.2 IsEnabled

Activation status of a control. This overwrites the status of the child control. Only active controls react to mouse and touch

Schema: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Container Control [▶ 459]

See also

Attribute getter: getIsEnabled [▶ 1181]

Attribute setter: setIsEnabled [▶ 1182]

Note: Available since version 1.8

6.3.2.1.3 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Container Control [▶ 459]

See also

Attribute getter: getWidthMode [▶ 1180]

Attribute setter: setWidthMode [▶ 1181]

Note: Available since version 1.8

6.3.3 Container

The container control offers the option of grouping other controls. All controls that are added to a container are defined relative to the position of the container and are shifted automatically with the container. This control can also be used for the subdivision and arrangement of a control, for example a view or a content.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 464]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations! • Content: The height is defined by the width of the content. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 464]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the height of the content. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 464]	Selection of whether the control is enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Container Control [▶ 459]

Container

See also

API reference [▶ 1184] for this control

6.3.4 Content

A content control is an independent container object that can be loaded in a region [▶ 496]. This control can be used, for example, to subdivide operating concepts into various contents. It is created under the TcHmi project tree with the context menu **Add new Item**…

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 464]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations! • Content: The height is defined by the width of the content. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 464]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the height of the content. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 464]	Selection of whether the control is enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Container Control [▶ 459]

Partial [▶ 490]

Content

See also

API reference [▶ 1189] for this control

6.3.5 Grid

The grid is a layout grid that can be used for the development of fluid and responsive designs. With the help of the grid, parts of the page or the entire page can be divided into various cells. On the basis of the cells you can then design the page by adding controls to the various cells and defining the behavior of the cells when changing the browser size. The grid offers a great many different configuration options. You can find the description of an example configuration in the chapter Configuration [▶ 480].

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 486]	Selection of the method of calculating the width
HeightMode [▶ 485]	Selection of the method of calculating the height

Category: Common

Name	Description
ColumnOptions [▶ 485]	Definition of the columns
RowOptions [▶ 485]	Definition of the rows
CellOptions [▶ 484]	Definition of the cells
CellWrap [▶ 484]	Selection of whether the cells should wrap within the row
IsEnabled [▶ 464]	Selection of whether the control is enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

  Container Control [▶ 459]

Grid

See also

API reference [▶ 1194] for this control

6.3.5.1 Configuration

A grid control can be configured in different ways. An example configuration of the Desktop.view together with a grid control is explained below. Prior to this you should read the chapter on Positioning and calculation of the size of controls [▶ 134] as the basis for this example.

1. Create a new project.

2. Open the Desktop.view.

The Desktop.view has a fixed size as a default setting. If the browser size changes, therefore, the Desktop.view and its contents do not adapt themselves. In the following steps the view will be adjusted so that it changes its size dynamically with the browser size.

3. Select the view.

4. Change the "Width" attribute to 100 %.

With this setting the size of the view adapts itself automatically to the size of the browser. The viewport of the graphic editor can now be used to simulate a desired browser size and test the effect on the visualization.

5. Add a grid control to the interface.

6. Change the following attributes: Left = 20 px, Top = 100 px, Right = 20 px.

7. Select the "Parent" mode for the "Width Mode" attribute.

With these settings the size of the grid control adapts itself to the size of the parent element accordingly. In this case the parent element is the Desktop.view.

8. Insert a second column with the help of the "Column Options [▶ 483]" dialog.

9. Insert a text block in each column by drag and drop.

10. Change the position of both controls to: Left = 120, Top = 140

With these settings the right-hand cell will be shifted automatically to the row below if there is insufficient space in the browser to display both side by side. You can test this function by changing the width of the integrated LiveView [▶ 62] window or the browser instance. So far the grid has not adapted itself automatically in height. The consequence of this is that only half of each of the two text blocks is displayed if the right-hand cell is displayed in the second row.

11. Select the "Content" mode in the "Height Mode" attribute of the grid control.

12. Also select the "Content" mode in the "Height Mode" attribute of the view.

With these settings not only will the grid be enlarged according to its contents, but also the view. In this way you can ensure that all contents are displayed so as to be reachable if the right-hand cell jumps to the row below. In order to be able to distinguish better between the two cells, their background colors will be set in the following.

13. Insert two cells with the help of the "Cell Options [▶ 481]" dialog.

14. Change the first cell option to: Background Color = #FFCA0000, Column Index = 0, Row Index = 0

15. Change the second cell option to: Background Color = #FFFFFF00, Column Index = 1, Row Index = 0

At the moment, both cells close and thus the entire row are bordering the lowermost control. In the row settings, for example, a minimum row height can be specified.

16. Change the "Row Min Height" attribute of the row option to 300 px with the help of the "Row Options [▶ 483]" dialog.

If the browser height is larger than the grid, a white border will be displayed at the moment. To make sure that the visu completely fills the browser with each browser size, the following step must be carried out.

17. Change the "Minimum Height" attribute of the view to 100 %.

In summary, you have created your first visualization with the help of the grid control according to the responsive design approach. This visualization fills the entire browser regardless of the browser size. If insufficient width is available, the content of the second column is automatically displayed in the row below. If on the other hand the height is insufficient, it will be expanded automatically according to the content to be displayed.

Sample project: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/4010038539.zip

6.3.5.2 Dialogs

6.3.5.2.1 Cell Options Dialog

You can change the cell settings with this dialog.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control
BorderColor [▶ 446]	Definition of the border color of the control
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides

Category: Border

Name	Description
Border Width	Definition of the border width of the four sides of the cell
Border Style	Selection of the border style of the four sides of the cell

Category: Layout

Name	Description
Grid Column Index	Definition of the column of a grid that is to be defined by the cell option
Grid Row Index	Definition of the row of a grid that is to be defined by the cell option
Padding	Definition of an additional distance between the cell border and the controls added to the cell for the four sides

6.3.5.2.2 Column Options Dialog

You can change the number of columns and their settings with this dialog.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Category: Layout

Name	Description
Column Width	Definition of the column width
WidthMode [▶ 486]	Selection of the method of calculating the width
HeightMode [▶ 485]	Selection of the method of calculating the height
Column Max Width	Definition of the maximum column width
Column Min Width	Definition of the minimum column width
Overflow	Selection of whether controls that have been added to the row but placed outside the row area should be displayed

6.3.5.2.3 Row Options Dialog

You can change the number of rows and their settings with this dialog.

Items	List of all objects added
	Delete the selected object
	Change the object order (top = left)
	Add a new object

Category: Layout

Name	Description
Row Height	Definition of the row height
HeightMode [▶ 485]	Selection of the method of calculating the height
Row Max Height	Definition of the maximum row height
Row Min Height	Definition of the minimum row height
Overflow	Selection of whether controls that have been added to the row but placed outside the row area should be displayed

6.3.5.3 Attributes

6.3.5.3.1 CellOptions

Definition for the cells.

Here you can define a specific feature for each cell. In each definition a row and column index defines which cell is to be addressed.

You can define:

- background color (identical to Control Background Color [▶ 447])
- border color (identical to Control Border Color [▶ 446])
- background image (identical to Control Background Image [▶ 448])
- width of the background image (identical to Control Background Image Width [▶ 448])
- height of the background image (identical to Control Background Image Height [▶ 449])
- horizontal alignment of the background image (identical to Control Background Image [▶ 449])
- vertical alignment of the background image (identical to Control Background Image [▶ 450])
- border width (identical to Control Border Width [▶ 447])
- border style (identical to Control Border Style [▶ 447])
- padding (identical to Control Internal Distance [▶ 449])

Schema: tchmi:framework#/definitions/CellOptions [▶ 2199]

Origin: Grid [▶ 475]

See also

Attribute getter: getCellOptions [▶ 1203]

Attribute setter: setCellOptions [▶ 1204]

Note: Available since version 1.8

6.3.5.3.2 CellWrap

Determines whether the cells should wrap within the row.

Schema: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: Grid [▶ 475]

See also

Attribute getter: getCellWrap [▶ 1204]

Attribute setter: setCellWrap [▶ 1205]

Note: Available since version 1.8

6.3.5.3.3 ColumnOptions

Definition for the columns and thus their number.

You can define:

- Width (similar to Control Width [▶ 441])
- Width Mode (similar to Control Width Mode [▶ 441])
- Minimum Width (identical to Control Minimum Width [▶ 452])
- Maximum Width (identical to Control Maximum Width [▶ 451])
- Overflow determines whether a child control is truncated if it is larger than the cell.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/ColumnOptions [▶ 2214]

Origin: Grid [▶ 475]

See also

Attribute getter: getColumnOptions [▶ 1201]

Attribute setter: setColumnOptions [▶ 1202]

Note: Available since version 1.8

6.3.5.3.4 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Grid [▶ 475]

See also

Attribute getter: getHeightMode [▶ 1200]

Attribute setter: setHeightMode [▶ 1200]

Note: Available since version 1.8

6.3.5.3.5 RowOptions

Definition for the rows and thus their number.

You can define:

- Height (similar to Control Height [▶ 442])

- Height Mode (similar to Control Height Mode [▶ 442])
- Minimum Height (identical to Control Minimum Height [▶ 450])
- Maximum Height (identical to Control Maximum Height [▶ 450])
- Overflow determines whether a child control is truncated if it is larger than the cell.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/RowOptions [▶ 2249]

Origin: Grid [▶ 475]

See also

Attribute getter: getRowOptions [▶ 1202]

Attribute setter: setRowOptions [▶ 1203]

Note: Available since version 1.8

6.3.5.3.6 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Grid [▶ 475]

See also

Attribute getter: getWidthMode [▶ 1201]

Attribute setter: setWidthMode [▶ 1201]

Note: Available since version 1.8

6.3.6 HTML host

The HTML host is a container for HTML contents. This control can be edited in the HTML editor of the Designer.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 441]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 442]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

HTML host

See also

API reference [▶ 1208] for this control

6.3.7 Partial

No instance of this class can be created. This page serves only as a list of the attributes that this class provides. It is the basic class for reusable control containers.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 464]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations! • Content: The height is defined by the width of the content. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 464]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the height of the content. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 464]	Selection of whether the control is enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Container Control [▶ 459]

Partial

See also

API reference [▶ 1213] for this control

6.3.8 Region

The region is a container for content controls. The content to be displayed can be defined using the "Target Content" attribute.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 502]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side. The "Width" attribute is ignored in all calculations! • Content: The width is defined by the children of the content control. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 502]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the children of the content control. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 439]	Selection of whether a control should be activated
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control
TargetContent [▶ 502]	Definition of the path to the content control that is to be displayed
ScaleMode [▶ 503]	Selection of the scaling of the content that is displayed in the region: • None: No scaling of the content • ScaleToFill: The content is displayed over the entire size of the control. The proportions are thereby ignored. • ScaleToFit: The content is displayed as large as possible without extending beyond the edges of the control. The proportions are retained. • ScaleToFitWidth: The content is displayed over the entire width of the control. The proportions are retained. • ScaleToFitHeight: The content is displayed over the entire height of the control. The proportions are retained.
Scrolling [▶ 503]	Selection of the scrolling behavior of the content: • No • Yes • Auto A browser has to be used to test the scrolling.

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Control

Name	Description
onTargetContentReplaced [▶ 503]	The content control was replaced.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Region

See also

API reference [▶ 1218] for this control

6.3.8.1 Attributes

6.3.8.1.1 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the content control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Region [▶ 496]

See also

Attribute getter: getWidthMode [▶ 1224]

Attribute setter: setWidthMode [▶ 1225]

Note: Available since version 1.8

6.3.8.1.2 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!
- **Content:** The height is defined by the children of the content control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: Region [▶ 496]

See also

Attribute getter: getHeightMode [▶ 1223]

Attribute setter: setHeightMode [▶ 1224]

Note: Available since version 1.8

6.3.8.1.3 TargetContent

Path to the content [▶ 470] control.

Schema: tchmi:framework#/definitions/ContentPath [▶ 2216]

Origin: Region [▶ 496]

See also

Attribute getter: getTargetContent [▶ 1225]

Attribute setter: setTargetContent [▶ 1226]

Note: Available since version 1.8

6.3.8.1.4 ScaleMode

Defines the scaling of the control.

Possible values:

- None (Standard)
- ScaleToFit
- ScaleToFitWidth
- ScaleToFitHeight
- ScaleToFill

Schema: tchmi:framework#/definitions/ScaleMode [▶ 2251]

Origin: Region [▶ 496]

See also

Attribute getter: getScaleMode [▶ 1226]

Attribute setter: setScaleMode [▶ 1227]

Note: Available since version 1.8

6.3.8.1.5 Scrolling

Defines whether the content can be scrolled.

Possible values:

- No (Standard)
- Yes
- Auto

Schema: tchmi:framework#/definitions/ScrollMode [▶ 2252]

Origin: Region [▶ 496]

See also

Attribute getter: getScrolling [▶ 1227]

Attribute setter: setScrolling [▶ 1228]

Note: Available since version 1.8

6.3.8.2 Events

6.3.8.2.1 onTargetContentReplaced

The content control was replaced. The new control is not yet displayed at this point in time.

Origin: Region [▶ 496]

Note: Available since version 1.8

6.3.9 User Control

A user control is a reusable collection of controls which, after creation, is available in the toolbox like a standard control. The control is created under the TC HMI project tree with the context menu **Add new Item…**

When adding a user control, a user control host is created that has the user control as a child element. Therefore, every change to an attribute (for example the background color) acts from the outside on the user control host and from the inside on the user control.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 464]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations! • Content: The height is defined by the width of the content. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 464]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the height of the content. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 464]	Selection of whether the control is enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Container Control [▶ 459]

Partial [▶ 490]

User Control

See also

API reference [▶ 1228] for this control

6.3.10 User Control Host

The user control host is an administration container for user controls [▶ 509]. It is created automatically when using a user control.

Changes since version 1.10

Form version 1.10, the attribute TargetUserControl is mandatory at the time of compilation and can no longer be changed at runtime.
A missing attribute TargetUserControl and errors in the configuration of the UserControl defined in TargetUserControl now lead to compilation errors that can indicate an error in the configuration.

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 514]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side. The "Width" attribute is ignored in all calculations! • Content: The width is defined by the user control that is displayed in the user control host. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 514]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the user control that is displayed in the user control host. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
TargetUserControl [▶ 515]	Definition of the path to the user control that is to be displayed
IsEnabled [▶ 515]	Selection of whether the control should be enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Control

Name	Description
onTargetUserControlReplaced [▶ 515]	The user control was replaced.

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

User Control Host

See also

API reference [▶ 1233] for this control

6.3.10.1 Attributes

6.3.10.1.1 HeightMode

Selection of the method of calculating the height. The available options are:

- **Value:** The height is defined by the height entered.
- **Parent:** The height is defined by the distance to the top and bottom side. The *Height* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the user control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: User Control Host [▶ 509]

See also

Attribute getter: getHeightMode [▶ 1239]

Attribute setter: setHeightMode [▶ 1239]

Note: Available since version 1.8

6.3.10.1.2 WidthMode

Selection of the method of calculating the width. The available options are:

- **Value:** The width is defined by the width entered.
- **Parent:** The width is defined by the distance to the left and right side. The *Width* attribute is ignored in all calculations!
- **Content:** The width is defined by the children of the user control.

Note: Learn how controls can be positioned and their size defined [▶ 134].

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Origin: User Control Host [▶ 509]

See also

Attribute getter: getWidthMode [▶ 1240]

Attribute setter: setWidthMode [▶ 1240]

Note: Available since version 1.8

6.3.10.1.3 IsEnabled

Activation status of a control. This overwrites the status of the user control [▶ 504]. Only active controls react to mouse and touch

Schema: tchmi:general#/definitions/Boolean [▶ 2176]

Origin: User Control Host [▶ 509]

See also

Attribute getter: getIsEnabled [▶ 1242]

Attribute setter: setIsEnabled [▶ 1242]

Note: Available since version 1.8

6.3.10.1.4 TargetUserControl

Path to the user control [▶ 504].

Schema: tchmi:framework#/definitions/UserControlPath [▶ 2266]

Origin: User Control Host [▶ 509]

See also

Attribute getter: getTargetUserControl [▶ 1241]

Attribute setter: setTargetUserControl [▶ 1241]

Note: Available since version 1.8

ReadOnly from version 1.10

6.3.10.2 Events

6.3.10.2.1 onTargetUserControlReplaced

The user control was replaced. The new control is not yet displayed at this point in time.

Origin: User Control Host [▶ 509]

Note: Available since version 1.8

6.3.11 View

A view is the top-level control of a visualization. The control is created under the TC HMI project tree with the context menu **Add new Item…**

Attributes

Category: Colors

Name	Description
BackgroundColor [▶ 447]	Definition of the background color
BorderColor [▶ 446]	Definition of the border color of the control

Category: Layout

Name	Description
Left [▶ 440]	Definition of the distance from the left side to the left side of the surrounding control
LeftUnit [▶ 440]	Definition of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Top [▶ 440]	Definition of the distance from the top side to the top side of the surrounding control
TopUnit [▶ 440]	Selection of the unit of the distance to the top side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Right [▶ 443]	Definition of the distance from the right side to the right side of the surrounding control
RightUnit [▶ 443]	Selection of the unit of the distance to the right side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Bottom [▶ 443]	Definition of the distance from the bottom side to the bottom side of the surrounding control
BottomUnit [▶ 443]	Selection of the unit of the distance to the bottom side of the surrounding control: • Pixels for absolute positioning • Percent for relative positioning
Width [▶ 441]	Definition of the external width of the control
WidthUnit [▶ 441]	Selection of the unit of the width: • Pixels for an absolute width • Percent for a relative width
Height [▶ 442]	Definition of the external height of the control
HeightUnit [▶ 442]	Selection of the unit of the height: • Pixels for an absolute height • Percent for a relative height
MaxWidth [▶ 451]	Definition of the maximum width if the control size is defined as dynamic
MaxWidthUnit [▶ 451]	Selection of the unit of the maximum width: • Pixels for an absolute value • Percent for a relative value
MinWidth [▶ 452]	Definition of the minimum width if the control size is defined as dynamic
MinWidthUnit [▶ 452]	Selection of the unit of the minimum width: • Pixels for an absolute value • Percent for a relative value
MaxHeight [▶ 450]	Definition of the maximum height if the control size is defined as dynamic
MaxHeightUnit [▶ 451]	Selection of the unit of the maximum height: • Pixels for an absolute value • Percent for a relative value
MinHeight [▶ 450]	Definition of the minimum height if the control size is defined as dynamic

Name	Description
MinHeightUnit [▶ 450]	Selection of the unit of the minimum height: • Pixels for an absolute value • Percent for a relative value
RenderedLeft [▶ 444]	Calculated distance in pixels from the left side to the left side of the surrounding control
RenderedTop [▶ 444]	Calculated distance in pixels from the top side to the top side of the surrounding control
RenderedRight [▶ 445]	Calculated distance in pixels from the right side to the right side of the surrounding control
RenderedBottom [▶ 444]	Calculated distance in pixels from the bottom side to the bottom side of the surrounding control
RenderedWidth [▶ 445]	Calculated width of the control in pixels
RenderedHeight [▶ 445]	Calculated height of the control in pixels
GridRowIndex [▶ 446]	Definition of the row of a grid in which the control is to be displayed
GridColumnIndex [▶ 446]	Definition of the column of a grid in which the control is to be displayed
Opacity [▶ 452]	Definition of the transparency of the entire control
Visibility [▶ 453]	Selection of the visibility of a control: • Visible: Control is visible • Hidden: Control is invisible, but is drawn – controls behind it cannot be operated • Collapsed: Control is invisible and is not drawn – controls behind it can be operated
ZIndex [▶ 454]	Definition of the position on the Z-axis
Transform [▶ 453]	Definition of a transformation of a control: • Translate: Translate in X-, Y-, Z-direction • Rotate: Rotate by an angle • Scale: Zoom by a factor • Skew: Sides shown skewed by an angle • Origin: Set the origin of the transformation relative to the position of the control • Perspective: Set the perspective of the transformations of all child elements
BoxShadow [▶ 454]	Definition of a shadow
WidthMode [▶ 464]	Selection of the method of calculating the width: • Value: The width is defined by the width entered. • Parent: The width is defined by the distance to the left and right side of the parent element. The "Width" attribute is ignored in all calculations! • Content: The height is defined by the width of the content. The "Width" attribute is ignored in all calculations!
HeightMode [▶ 464]	Selection of the method of calculating the height: • Value: The height is defined by the height entered. • Parent: The height is defined by the distance to the top and bottom side of the parent element. The "Height" attribute is ignored in all calculations!

Name	Description
	• Content: The height is defined by the height of the content. The "Height" attribute is ignored in all calculations!

Category: Common

Name	Description
IsEnabled [▶ 464]	Selection of whether the control is enabled
Id [▶ 438]	Definition of the name of the control
Type [▶ 453]	Type name of the control.
ClassNames [▶ 439]	Definition of control classes
IsAttached [▶ 439]	If a control is logically assigned in the DOM.
ToolTip [▶ 452]	Definition of the tool tip for the control

Category: Border

Name	Description
BorderRadius [▶ 447]	Definition of the radius of the four corners to round them off
BorderStyle [▶ 447]	Selection of the border style of the four sides of the control
BorderWidth [▶ 447]	Definition of the border width of the four sides of the control

Category: Background Image

Name	Description
BackgroundImage [▶ 448]	Definition of an image that is located above the background color
BackgroundImageHeight [▶ 449]	Definition of the height of the background image
BackgroundImageHeightUnit [▶ 449]	Selection of the unit of the height of the background image: • Pixels for an absolute height • Percent for a relative height
BackgroundImageWidth [▶ 448]	Definition of the width of the background image
BackgroundImageWidthUnit [▶ 448]	Selection of the unit of the width of the background image: • Pixels for an absolute width • Percent for a relative width
BackgroundImagePadding [▶ 449]	Definition of an additional distance between the border of the control and background image for the four sides
BackgroundImageVerticalAlignment [▶ 450]	Definition of the vertical alignment of the background image within the control
BackgroundImageHorizontalAlignment [▶ 449]	Definition of the horizontal alignment of the background image within the control

Events

Category: Framework

Name	Description
onAttached [▶ 455]	The control was added to the display.
onDetached [▶ 455]	The control was removed from the display.
onInitialized [▶ 455]	The control was initialized.
onMoved [▶ 458]	The control was moved in relation to the browser window.
onResized [▶ 458]	The size of the control was changed.

Category: Operator

Name	Description
onPressed [▶ 458]	A mouse click was executed or a touch screen was briefly touched.
onMouseClick [▶ 455]	The left mouse button was clicked over the control and released.
onMouseDoubleClick [▶ 455]	The left mouse button was clicked twice over the control and released.
onMouseRightClick [▶ 457]	The right mouse button was clicked over the control and released.
onMouseDown [▶ 455]	A mouse button was depressed over the control.
onMouseEnter [▶ 456]	The mouse pointer was moved over the control.
onMouseLeave [▶ 456]	The mouse pointer has left the control.
onMouseUp [▶ 457]	A mouse button was released over the control.
onMouseDownLeft [▶ 456]	The left mouse button was depressed over the control.
onMouseDownRight [▶ 456]	The right mouse button was depressed over the control.
onMouseMove [▶ 456]	The mouse was moved over the control.
onMouseOver [▶ 457]	The mouse pointer was moved over the control or one of its children.
onMouseOut [▶ 456]	The mouse pointer has left the control or one of its child controls.
onMouseUpLeft [▶ 457]	The left mouse button was released over the control.
onMouseUpRight [▶ 457]	The right mouse button was released over the control.
onMouseWheel [▶ 457]	The mouse wheel was moved over the control.
onTouchStart [▶ 458]	A finger/stylus was placed against a touch screen.
onTouchMove [▶ 458]	A finger/stylus was moved on a touch screen.
onTouchEnd [▶ 458]	A finger/stylus was lifted off a touch screen.
onTouchCancel [▶ 458]	An interaction with a touch screen was aborted.

Access rights

Name	Description
observe [▶ 459]	Determines whether the control is generally observable.
operate [▶ 459]	Determines whether the control is generally operable.

Inheritance hierarchy

Control [▶ 433]

Container Control [▶ 459]

Partial [▶ 490]

View

See also

API reference [▶ 1243] for this control

6.4 User controls

A user control enables the re-use of existing controls in a separate control. Existing controls include the system controls, the Beckhoff controls and the framework controls. The controls can be placed in the user control by drag & drop.

Further information on the controls can be found in the chapter Creating a user control [▶ 37].

6.5 Framework controls

User-specific controls [▶ 2123] can be developed in JavaScript in order to integrate controls or libraries from third-party suppliers or to map applicative requirements.

6.6 Control rights

Enables or prevents the editing or viewing of the respective control by the user groups configured. For more information, see the chapter User Management [▶ 521].

7 User management

User management in TwinCAT HMI is an authorization system for controlling user group-specific access at control, symbol or file level.

The use of the user management is optional and only works with active authentication.

> Since user management only works when authentication is active, the configured functions are not available in live view.

7.1 Users and user groups

The TwinCAT HMI User Management is based on a user group concept. The user groups are managed centrally in the TwinCAT HMI Server (TcHmiSrv). The actual users, who may be members of one or more user groups, are made available as a data source by the server extension TcHmiUsermanagement. The creation and editing of users and user groups takes place within the development environment via the TwinCAT HMI Configuration window under the category **Users and Groups**, within which the individual **Users** and **Groups** are listed.

7.1.1 Create new user

1. Select **Create new user** with a right mouse click on **Users**, or the **Create User** function with a left mouse click.

2. Configure the user via the following properties in the **Create / edit user** dialog.

- **User name:** Unique name of the user.
- **Password:** Password of the new user for the login. Empty passwords are not allowed.
- **Confirm Password:** Confirm password.
- **Authenticate by client certificate:** Selection of a client certificate, [▶ 540] if available, which is assigned to the user. This makes it possible for a user to log on using a client certificate.
- **Auth Extension:** Server extension that manages the authentication of the user. By default, the TcHmiUserManagement extension is selected. The developer can develop his own authentication extension to provide his own user management.
- **Member of the following groups:** Group membership of the user. A selection from the existing groups is possible. A user can be a member of several groups. By default, the user is a member of the system group "__SystemUsers".
- **Language:** Language of the user. The language can be taken from the project properties or the browser default or can be defined individually.

- **Time zone:** Time zone in which the user is located. By assigning the time zone, it is possible to localize the times displayed on the HMI. The time zone can be taken from the project properties or the browser default or can be defined individually.
- **Time format locale:** Time format to be displayed to the user. By assigning the time format, it is possible to format the times displayed on the HMI. The time format can be taken from the project properties or the browser default or can be defined individually.
- **Auto-Logout Time:** Time of inactivity of the user after which the user is automatically logged out on the HMI (client side) without any user interaction.

Confirm your settings with **OK**.

The settings "Authenticate by client certificate", "Auth Extension", "Time zone" and "Time format local" are available as of version 1.10.1336.0

7.1.2 Changing user properties

1. Double-click on the user whose properties you wish to change or right-click on **Edit user**.

2. Make the desired changes (see Create new user [▶ 521]).

7.1.3 Activate auto-login user

Enabling the auto-login function allows a specific user to log in automatically without authentication. This skips the login page and logs in a user with preconfigured rights. If the auto-login function is activated, the auto-login user cannot be logged out. As soon as another user is logged out, the auto-login user is logged in automatically.

The Auto Login function is activated in the TwinCAT HMI Configuration window under the **Users** item. To do this, the checkbox for a user must be set. The auto login function can only be activated for one user and not for several users at the same time.

Available since version 1.10.1171.142

See also

GoToLoginPage [▶ 109]

7.1.4 Deleting a user

1. Select **Delete User** by right-clicking on the user to be deleted.

7.1.5 Creating a new user group

1. Select **Add new group** by right-clicking on **Groups**, or the **Create new group** function with a left mouse click.

2. Configure the user group with the following properties in the associated dialog:

- **Group name:** Unique name of the group.
- **Symbol Access:** Standard access (None, Read, Write, Read/Write) to symbols.
- **File Access:** Standard access (None, Read, Write, Read/Write) to files.
- **Status:** Group availability (Enabled, Disabled).
- **Members:** Group members (selection of the existing users).

Confirm your settings with **OK**.

7.1.6 Changing user group properties

1. Call the dialog by double-clicking on the user group whose properties you wish to change, or by a right mouse click on it via **Edit group**.

2. Make the desired changes (see Creating a new user group [▶ 524]).

7.1.7 Deleting a user group

1. Select **Delete Group** by right-clicking on the user group to be deleted.

7.1.8 System users and system user groups

The following users and user groups exist as standard in the system and cannot be deleted:

System user

__SystemAdministrator: System administrator and member of the group "__Systemadministrators".

__SystemGuest: Guest who is automatically active if no user is logged in and is a member of the group "__SystemGuests".

System user groups

__SystemAdministrators: System administrator group with unrestricted rights and the following default settings:

- Symbol Access: Read/Write
- File Access: Read/Write
- Enabled: True
- Members: __SystemAdministrator

__SystemGuests: Guest group with restricted rights, which allow only the login function, and the following default settings:

- Symbol Access: None
- File Access: None
- Enabled: True
- Members: __SystemGuest

__SystemUsers: User group with restricted rights that allow access to all system server symbols required at runtime. The user group has the following default settings:

- Symbol Access: None
- File Access: Read/Write
- Enabled: True
- Members: Every user except __SystemAdministrator or __SystemGuest

By default, all new users are members of the group __SystemUsers as a minimum. This behavior can be changed on the server configuration page under "TcHmiSrv/Security" under the name "Default Usergroup".

The system users and system user groups are visible and configurable only in Advanced Mode. Do not change any settings in the system user groups.

7.2 Permissions system

You can issue user group-specific permissions at three levels in the permissions system of the TwinCAT HMI User Management.

1. The control level [▶ 527] represents the client-side permissions system of the TwinCAT HMI Framework. This is a manipulation of the user interface through the hiding of controls or the deactivation of their functions, depending on which group(s) the logged-in user belongs to.
2. The symbol level [▶ 530] controls the user group-specific access to server symbols (mapped symbols).
3. The file level [▶ 532] enables the user group-specific access to the files made available by the TwinCAT HMI Server.

While level one is purely a manipulation of the user interface on the client side, the second and third levels offer real data security on the server side.

In order to be able to configure the rights of the system user groups at the individual levels, you have to activate Advanced Mode in the TwinCAT HMI Configuration window.

7.2.1 Control level

The control level represents the client-side permissions system. A control has defined access methods that can be permitted or not permitted for the different user groups. Each control has at least the default access methods "Observe" and "Operate".

"Observe" describes the general observability of the control and thus its visibility to the user. If this access method is not permitted for the user group of the logged-in user, the control is not visible. This also applies to all child controls of the control, if there are any.

"Operate" represents the general operability of the control. If this access method is not permitted for the user group of the logged-in user, then the control is displayed as deactivated and no control events that could be initiated by the user (operator events) are initiated.

The access methods are configured in the Properties window (Configuring control access [▶ 527]).

The access methods of a user control can be extended via special access methods, so-called virtual permissions (Defining and using virtual permissions of a user control [▶ 529]).

7.2.1.1 Configuring control access

1. Select the control in the designer.
2. Open the **Show Permissions** tab in the Properties.

3. Configure the desired permissions of the individual user groups; the following options are available for this:
   Allow: The access method is permitted for the user group.

Disallow: The access method is not permitted for the user group.

Inherit: The permission of the user group is derived from the higher-level parent control (container control: View, Content, User Control, Container, Region or User Control Host).

Special cases

Access methods of a view

Operate and **Observe** are permitted for a view as standard (default = Allow). Since the view –as the entry point to the HMI application –is always the highest container control, the **Inherit** option is not available.

User is a member of several user groups

The following rules apply if the user is a member of several user groups:

- If **Allow** is configured for at least one user group, the access method is permitted for the user (**Allow**).

- If **Disallow** is configured for at least one user group and **Allow** is not configured for any user group, the access method is not permitted for the user (**Disallow**).

- If **Inherit** is configured for all user groups, the permission is derived from the higher-level container control (**Inherit**).

7.2.1.2 Defining and using virtual permissions of a user control

Defining virtual permissions

1. Open the parameter editor of the user control.

2. Under **Virtual Permissions**, add the desired additional user control access methods. The following properties can be configured here:
   Name: Unique name of the virtual permission
   Description: Description used as a tooltip

Additional properties are displayed if you activate the **Details** checkbox:
Internal Name: Internal name used in HTML code (corresponds by default to "Display Name").
Display Name: Name displayed to the user in the engineering (corresponds by default to "Name").
Visible: Visibility of the virtual permission for the user in the engineering.
Default Value (Internal): Default return value of the virtual permission if the query of the current permissions situation does not produce a return value.

3. Confirm your input with **OK**.

Using virtual permissions

4. Select a control inside the user control in the designer.
5. Assign a virtual permission for the desired access method to the selected control, which is to represent a special access method of the user control to the outside.

The desired access method is now no longer configurable inside the user control. However, it is available for configuration through the assignment of the virtual permissions as an additional access method of the user control host when using the user control in the designer.

For the purpose of reusability of the user control, direct configuration of the access methods inside a user control should be dispensed with. The independence of project-specific user groups is achieved with this approach. Accordingly, the default access methods "Observe" and "Operate" should be configured either with "Inherit" or as special access methods via the use of virtual permissions. Special access methods of controls within a user control should always be fed to the outside via the use of virtual permissions because the user control host only has the default access methods "Observe" and "Operate" as standard.

7.2.2 Symbol level

The symbol level represents the first part of the server-side permissions system. General and user group-specific access methods, which apply when authentication is active, can be configured for every mapped server symbol (Mapped Symbols). The configuration is done via the Permissions Management editor, which can be called via the TwinCAT HMI Configuration window (Configuring symbol access [▶ 531]).

7.2.2.1 Configuring symbol access

1. Select the **Symbol Permissions** tab in the Permission Management editor.

2. Configure the following access methods for a mapped server symbol:
   Symbol Access: Generally permitted symbol access method that is valid for each user group (default value: Read/Write).

   Group Symbol Access: Permitted default access method of a user group for all symbols (see also Creating a new user group [▶ 524]).

   Specific Group Symbol Access: Permitted user group-specific access method for the symbol that is initially specified by the combination of **Symbol Access** and **Group Symbol Access,** whose configurations mutually restrict each other.

However, it is also possible to overwrite this combination so that **Symbol Access** and **Group Symbol Access** are invalid for this symbol for this user group.

	Symbol Access			Group1		
	Read	Write		Read	Write	Group
			Default:	✓	✗	
Symbol	☐	☐	Toggle all:	☐	☐	☐
Symbol1	✓	✗		✓	✓	✗

3. Confirm your configuration with **OK**

Special cases

User is a member of several user groups

If a user is a member of several user groups, the combination of permitted access methods of the user groups applies, which in this case mutually expand one another.

> The permitted access methods for system server symbols can be configured in Advanced Mode.

7.2.3 File level

The file level represents the second part of the server-side permissions system. General and user group-specific access methods, which apply when authentication is active, can be configured for every file and every folder added to the TwinCAT HMI Server via the publishing. The configuration is done via the Permissions Management editor, which can be called via the TwinCAT HMI Configuration (Configuring file access [▶ 532]).

7.2.3.1 Configuring file access

1. Select the **File and Folder Permissions** tab in the Permission Management editor.

2. Configure the following access methods for a file (or a folder):
   File Access: Generally permitted file access method that is valid for each user group (default value: Read/Write).

	File Access			Group1		
	Read	Write		Read	Write	Group
			Default:	✗	✗	
File/Folder	☐	☐	Toggle all:	☐	☐	☐
/MyImage.png	✓	✓		✗	✗	✓

Group File Access: Permitted default access method of a user group for all files (see also Creating a new user group [▶ 524]).

	File Access			Group1		
	Read	Write		Read	Write	Group
			Default:	✗	✗	
File/Folder	☐	☐	Toggle all:	☐	☐	☐
/MyImage.png	✓	✓		✗	✗	✓

Specific Group File Access: Permitted user group-specific access method for the file that is initially specified by the combination of **FileAccess** and **Group File Access**, whose configurations mutually restrict each other.

	File Access			Group1		
	Read	Write		Read	Write	Group
			Default:	✗	✗	
File/Folder	☐	☐	Toggle all:	☐	☐	☐
/MyImage.png	✓	✓		✗	✗	✓

However, it is also possible to overwrite this combination so that **File Access** and **Group File Access** are invalid for this file for this user group.

	File Access			Group1		
	Read	Write		Read	Write	Group
			Default:	✗	✗	
File/Folder	☐	☐	Toggle all:	☐	☐	☐
/MyImage.png	✗	✗		✓	✓	✗

3. Confirm your configuration with **OK**

Special cases

User is a member of several user groups

If a user is a member of several user groups, the combination of permitted access methods of the user groups applies, which in this case mutually expand one another.

7.3 Access management in the client

The system provides functions [▶ 97] for managing access in the client. The functions can be used to log out a user and to check the user's current access. The functions are available in the development environment in the Actions and Conditions Editor [▶ 50].

Check Access

Function [▶ 109] for querying whether a certain control access method is permitted for the logged-in user or not.

GoToLoginPage

Function [▶ 109] to open the login page without logging out the currently logged in user. This allows a user change to be performed without a previous logout of the user.

Logout

Function [▶ 109] for logging out the logged in user from the TwinCAT HMI server. The function loads the login page after the logout.

7.4 Login page

If authentication is active, the login page (login dialog) is displayed before the start page is opened and allows users to log in. Start page of the HMI loads only after successful authentication.

The login page can be replaced by a user-specific login dialog. This can be added to any TwinCAT HMI project if required. The login page is edited in HTML, CSS and JavaScript. It is not possible to edit a login page via the TwinCAT HMI Designer, as the framework is not yet loaded at the time of user login.

Available since version 1.10.1336.0

7.4.1 Add login page

A TwinCAT HMI project can contain several login pages. Only one login page can be active [▶ 536] per project.

1. Right-click **Add** anywhere in the project, and then click **New Item**.

2. Select the **LoginPage** type and assign a name.

⇨ The login page can now be used in the project.

7.4.2 Switch login page

Several login pages can be managed within one project. Only one login page can be active per project. The switching of the active login page takes place in the project properties.

To do this, select the project node in the Solution Explorer and open the Properties window [▶ 43]:

The "Default" entry displays the standard login page of the TwinCAT HMI.

7.4.3 Edit login page

The creation of a login page takes place within an HTML file with HTML, CSS and JavaScript. The template contained in the TwinCAT HMI represents an exemplary login page and can be adapted and modified as required. The comments in the template serve as support.

A login page is an HTML page with the following structure:

- **Head tag**: Here you can specify the title of the login page, which is displayed in the browser window.
- **Style tags**: Here the appearance of the login page in CSS is described.
- **Body tag**: The structure of the login page in HTML is described here.
- **Script tags**: Here the logic of the login page is implemented in JavaScript.

7.4.3.1 Login

The login command enables a user to log in to the TwinCAT HMI server and is sent to the server as a WebSocket request in JSON format. If the authentication data is valid, the server allows the login and calls the start page of the HMI.
The WebSocket request must be structured as follows:

```
var message = {
   "requestType": "ReadWrite",
      "commands": [
         {
            "commandOptions": ["SendErrorMessage", "SendWriteValue"],
            "symbol": "Login",
            "writeValue":
            {
               "userName": username,
               "password": password,
               "persistent": stayLoggedIn
            }
         }
      ]
   };
```

In the template you will find this WebSocket request in the function "Login" in line 350.

The basic functionality of the requests to the server contained in the template should not be changed in order to ensure proper functioning of the login page.

7.4.3.2 Include graphics

Graphics such as a company logo can also be used within a user-specific login page. The use of inline graphics in Base64 format in the HTML file is recommended. The use of graphic resources outside the HMTL file is also possible, but requires additional configuration in the server.

inline graphics

Within an HTML file, graphics can be integrated as Base64 strings. The Base64 string can be used directly in HTML code or CSS code. With the user-specific login page, this has the advantage that no user rights have to be configured for the "__SystemGuests [▶ 526]" group in the server.

An HTML image tag with a Base64 string as source is structured as follows:

```
<img src=”data:[mime type];[charset],[base64]”>
```

The "MIME-Type" specifies the original file format, the "Charset" the character encoding and "base64" the data encoding. As an example of an SVG graphic:

```
<img src="…[rest of base64]"/>
```

The graphics can be encoded into a Base64 string via a Base64 Image Encoder. Encoding can be performed using an online tool, for example.

A favicon can also be integrated into the HTML page as an inline graphic.

External resources

The integration of external graphics into the user-specific login page is possible if they are located within the HMI project. First the graphic must be added to the project. The graphic can then be referenced within the HTML file, for example within an HTML image tag:

```
<img src="manual_pressed.svg" />
```

Since no user is logged in to the HMI server at the time the login page is loaded, the rights of the "__SystemGuests" group apply at this point in time. Therefore, this group must be granted explicit access to the graphics to be used within the login page. Open the configuration page of the HMI server in the category "General".

Open the configuration of the rights of the "_SystemGuests" group under "Usergroups".

Click "Add" under "Files".

Add the graphic with read/write permissions.

You can then use the graphic on the login page.

7.5 User management in the client

In addition to the configuration in the TwinCAT HMI Configuration window, the management (dynamic creation) of users is also possible during runtime in the client. For example, it is possible to add new users to the server at runtime, edit existing users (change password, change group membership) or remove them.

It is not possible to manage (dynamically create) groups at runtime. The groups must first be created in engineering and the rights to the corresponding controls configured. If the rights of a user are to be changed, the group membership of the user can be changed.

The administration at runtime in the client is made possible by the Framework API [▶ 1938]. This can be used in the JavaScript code. [▶ 2110]

Available since version 1.10.1018.48

7.6 Client certificates

The client certificates allow the authentication to be restricted so that only certain clients that have the certificate can access the server and thus the HMI. All other clients that do not have the certificate will not have access to the server. A so-called client certificate, which can be generated and signed by TwinCAT HMI in the engineering department, must be installed in the respective client. The server-side part of the certificate is automatically stored as the remote terminal in the TwinCAT HMI server during generation. A client certificate can optionally be assigned to a user so that the user is automatically logged in via the client certificate.

The use of client certificates is optional.

The use of authentication via client certificates requires encrypted communication via HTTPS.

Available since version 1.10.1336.0

7.6.1 Creating a Client Certificate

The client certificates can be created and managed in the TwinCAT HMI Configuration window. [▶ 65]

1. Open the TwinCAT HMI Configuration window and select the "Client certificates" tab.

2. Double-click on the **<create new client certificate>** entry or right-click on the "Client certificates" tab and select "**Create new client certificate...**".

3. In the dialog that appears, you can define the properties of the client certificate.

4. The name of the client certificate is required. All other parameters can be optionally specified for signing the certificate.

7.6.2 Assigning a Client Certificate

A client certificate can optionally be assigned to one or more users. This enables a user to log in automatically using the client certificate.

1. Create [▶ 521] a new user or edit an existing user.
2. In the following dialog, the client certificate is selected for the user.

A client certificate can be assigned to multiple users. A maximum of one client certificate can be assigned to a user.

7.6.3 Activating Client Certificates

Authentication via client certificates must be activated in the TwinCAT HMI server. You can make this setting directly in the configuration of the engineering server and use a project publish [▶ 110] to transfer it to a server on the target system. Alternatively, you can make the setting as required directly on the target system at a server.

1. Open the configuration page of the TwinCAT HMI server.

   Engineering server:
   Open the "Server" node within the TwinCAT HMI project in Visual Studio. Then open the "Extensions" node and double-click on the "TcHmiSrv" entry:

   Server on the target system (TF2000 HMI Server):
   Right-click on the SysTray icon of the TwinCAT HMI server:

   Open the configuration page via the context menu by clicking on the entry "Config":

2. On the configuration page, click on the entry "TcHmiSrv" (1) and then go to the "Security" tab (2). There you will find the entry "Only client certificates allowed to authenticate" (3).

3. Enable authentication using client certificates by setting the option to "true" (3). Confirm the action with the button "Accept" (4).
4. Authentication using client certificates is active. If you have made the setting on the engineering server, you can transfer [▶ 110] the project to a server on the target system.
5. The client certificates must be installed [▶ 544] in the clients that are to gain access to the system.

7.6.4 Installing the Client certificate

The client certificates can be installed in the clients that are to gain access to the system. First, the certificate must be exported as a PFX file.

1. Open the TwinCAT HMI Configuration window [▶ 65] and select the category "Client certificates".

2. Right-click on the desired certificate and select the entry "Export PFX (client part) ...":

The exported certificate can then be installed on the target systems that are to authenticate themselves using the certificate.

Installing the Client Certificate in a Browser

The following steps show the integration of a client certificate into a browser using the example of the "Google Chrome" browser. Integration is performed in the same way with other browsers.

1. Copy the exported certificate to the target system.
2. Open the browser on the target system.
3. Open the settings of your browser.
4. Open the advanced settings and select "Manage certificates".
5. Select the "Import" item:

6. Click "Next" in the following wizard:

7. Then click on "Browse" and select the certificate:

You may have to switch to the PFX format in the "Browse" dialog:

8. After selecting the certificate, click "Next".

9. The following dialog can be confirmed without entering a password with the default settings by clicking on "Next":

10. Select the area "Personnel" and click on "Next":

11. Confirm the import by clicking on "Finish":

Open Client

Start the browser and open the HMI via the IP address of the server. When you access the server using a client certificate for the first time, the dialog that offers the selection of a client certificate is opened in the browser. Select your certificate there:

Access to the HMI is then granted. If the client certificate has been assigned to [▶ 542] a user, this user is logged in automatically and the HMI is loaded. Otherwise you have to authenticate yourself with username and password.

Note Communication with the server must be encrypted using HTTPS if you are using a client certificate.

Error handling

If you have already worked with client certificates before, it may happen that the browser skips the query for a client certificate and access to the HMI is not granted. In this case, you can clear the browser cache and restart the browser. The browser should then query the client certificate and access the HMI.

8 Event system

The central event system is integrated in the TwinCAT HMI server. It is possible to manage events from the HMI server and its extensions. In addition, the TcHmiEventlogger [▶ 552] extension can be connected with the TwinCAT 3 event logger from local or remote real-time systems.

The EventGrid Control [▶ 553] can be configured flexibly and used to display current or historical events.

Available since version 1.10

8.1 TcHmiEventlogger extension

When creating a new TwinCAT HMI project the TcHmiEventLogger extension is also loaded and is located in the project under the server node with the Extensions [▶ 117].

Configuration

In the TwinCAT HMI server configurations you can configure the event logger with which the Engineering HMI Server (Publish Configuration: Default) or the Remote HMI Server (Publish Configuration: Remote) communicates. By default the extension connects to the local TwinCAT 3 event logger. Remote target systems can also be entered if the ADS route is configured.

Licensing

One target license is counted per event logger. If an ADS connection is also configured with the same NetId, only one target license is counted.

8.2 Event Grid Control

The Event Grid Control is a control for the tabular display of alarms and messages. It automatically displays the alarms and messages of the target systems addressed in the event logger extension. Alarms can be confirmed directly in the control.

The Event Grid Control is located in the Toolbox under the category **Beckhoff**, from where you can insert it into an HMI page by drag-and-drop.

Apart from the tabular display of the alarms and messages, the control offers various configuration options in the engineering and during the runtime in the browser.

Detailed view

A detailed view can be displayed for each message and alarm in the Event Grid. The detailed view is opened by selecting a row in the Event Grid and double-clicking on the row. The detailed view provides additional information about the event.

Filter

Filters enable the display of alarms and messages according to certain criteria. It is possible, for example, to display only active alarms or to filter according to a certain time period. The filters can be configured under Filter [▶ 240] in the control's properties.

You can switch between different filters at runtime by calling a WriteToSymbol [▶ 53] for the filter of the Event Grid. Alternatively, you can use various filter settings via the control's menu bar [▶ 245].

Columns

The configuration of the columns defines which columns are displayed in the Event Grid Control. By default the type (alarm or message), the severity, the timestamp of the input and the associated text are displayed. Further columns such as the time of the confirmation can be configured in the control's properties [▶ 242]. The settings apply to all clients.

There is an option at runtime to change the columns displayed per client. To do this, open the column editor from the menu bar [▶ 245]. The column editor automatically opens a popup with an overlay and positions itself in the center of the view:

Confirming an alarm

An alarm can be confirmed by selecting the row containing the alarm and clicking on the button **Confirm alarm** (single check mark) in the menu bar [▶ 245].

If several alarms are active, there is an option to confirm all alarms simultaneously. To do this, click on the button **Confirm all alarms** (two check marks) in the menu bar [▶ 245]. A popup with an overlay then opens automatically and displays all active alarms. At this point the operator has the option to abort the dialog or to confirm the active alarms.

Permissions

Within the context of an authorization system it may be useful to allow only certain operators to access the Event Grid. In addition, individual operators may be assigned rights to confirm the events. Rights are configured in TwinCAT HMI at two levels (see User Management [▶ 521]).

At control level the configuration right (configuration at runtime) and the right to a detail view can be granted [▶ 244] explicitly in addition to the standard rights for the Event Grid.

The right to confirm the alarms can be granted at symbol level. To do this, open the rights management [▶ 530] in the expert view.

Further configuration options

Further configuration options for the Event Grid Control can be found here [▶ 235].

Example

An example of the use of the event logger with the Event Grid: https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007204494484107.zip
The example shows two different configuration options for the Event Grid and the associated PLC project, in which example alarms and messages can be generated.

9 Internationalization

Internationalization in TwinCAT HMI enables the management of languages and units and their switching in the client. Furthermore, the administration of different time zones and time formats is supported.

9.1 Change the language

Language switching in TwinCAT HMI is implemented with keys, which are managed in language tables (localization files). Any texts in different languages can be stored in the language table for a key.

In the Designer, [▶ 46] the keys are linked directly to the respective control properties. [▶ 49] The texts behind the respective keys are resolved in the Designer directly in the currently active language.

Set active language

The active language can be set in engineering as follows:

- **Project properties:** The default language for the project can be set in the project properties. [▶ 43]
- **User Configuration**: The language per user can be set in the user configuration. [▶ 521] This setting overrides the default language set in the project properties.

Manage language tables

The language tables (localization files) of a TwinCAT HMI project are managed in the Localization editor. [▶ 90] Language tables can be added to the project as a new TwinCAT HMI item. [▶ 129]

Switch language in client

The active language can be switched at runtime in the client, regardless of which language settings have been specified in the project properties and for the logged in user. The switchover at runtime is realized with the function "SetLocale [▶ 106]", which can be used in the Actions & Conditions editor. [▶ 50]

9.2 Unit conversion

Unit switching in TwinCAT HMI enables the switching of the displayed units and the conversion of the corresponding values depending on the language or a specific event.

Conversion of values

The conversion of the units is solved by functions [▶ 97]. A function is linked to the Control property in [▶ 94] the properties window and returns the converted value. As a transfer parameter, the function receives the output value that is present in a certain unit.

Within the function, the developer can convert [▶ 2118] the corresponding value (e.g. from a symbol in the PLC) into the desired unit. The respective unit can be added to the converted value directly within the function as a string or alternatively stored as a language key.

Switching units

The units can be switched depending on the language or any other event. To do this, another parameter is passed to the function that contains the unit into which the value is to be converted. As soon as this parameter changes, the function is automatically called again and the value is converted and returned accordingly.

For example, pass an internal symbol [▶ 73] to the function that contains the unit into which the value is to be converted. The unit can be written anywhere in the internal symbol. Possible scenarios are e.g:

- "onLocaleChanged [▶ 85]": This global event is triggered as soon as a language is loaded. At this event the unit can be written into the internal symbol according to the active language [▶ 105].

- Events: Depending on any event, [▶ 30] a unit can be written into the internal symbol. The event can be triggered, for example, if a value in the PLC changes or the user has pressed a button on the user interface.

9.3 Localization

Localization in TwinCAT HMI enables the management of different time zones and time formats. The server is based on the UTC time. Some controls (for example, the Event Grid and the Trend Line Chart) require a timestamp. To display the times on the client according to their region (with time offset), the time zone and time format can be specified in the project properties or per user.

Setting in the project properties

The setting of the time zone and time format can be set for the project in the project properties [▶ 43].

Setting in the user properties

The time zone and time format settings can be set per user [▶ 521]. This setting overrides the setting in the project properties when user authentication is active.

Available since version 1.10.1336.0

10 Themes

The theme system allows you to switch the design of all elements. The theme can be switched during runtime per client or globally in the project properties. Only one theme can be active at a time (similar to language switching [▶ 557]). A theme can change the appearance of a control (e.g. Top [▶ 440], Left [▶ 440], Width [▶ 441], Height [▶ 442], BackgroundColor [▶ 447] etc.). Besides the appearance of the control, all control attributes [▶ 132] can be changed.

A theme can be created via the integrated theme editor or directly as a CSS file.

Available since version 1.10

10.1 Introduction

In a TwinCAT HMI project, the various themes are located under the Themes project node:

The theme node contains all themes defined for the project. For each theme there is a .theme file, which contains the JSON definition of a theme and can be opened with the theme editor.

The default theme can be viewed and switched under the general project properties (click on the project node and open the Properties window).

For a new project, the base theme is included in the project and set as the default theme. The base theme includes the font setting, which is included at project level regardless of the theme (see Fonts). In addition, each TwinCAT HMI Control has a base theme, which describes the design of the control if this is not explicitly overwritten by the developer in Engineering or by another theme.

Add a new theme by right-clicking on the **Themes** project node under **New\Add New Item**. In the dialog select the type **Theme** and click **Add**.

The new theme contains the .theme file, which can be opened via the theme editor [▶ 561].

Under a theme, right-click **Add Item** to add more items.

You have a choice between a Cascading Style Sheets file and a CSS control theme file (see CSS theme [▶ 566]).

10.2 Theme editor

Open the theme editor by double-clicking on a .theme file in a theme. The following graphic shows the theme editor, which is open for the basic theme as an example.

The theme editor is divided into the following areas:

1 Status information (Attribute Values for Control Type Button):

- **Default theme:** Here you can set the default theme (as in the project properties).
- **View:** Determines which attributes are displayed. In the standard view, only those attributes and controls are displayed that are usually changed by the user. The Advanced view displays all controls that can be changed and their attributes.

2 Theme Control Types and Classes: In this area you can select whether a Class theme [▶ 562] or a Control theme [▶ 565] should be edited.

3 Themeable Attributes: In this area you select for which attribute a property is to be defined, depending on the respective theme.
Only those attributes are displayed that can be changed via the theme.

4 Attribute Properties: In this area, the property for the respective attribute is set. Any number of properties can be set for each Class theme and each Control theme.

10.2.1 Class theme

A class theme defines properties that apply to all controls assigned to this class, regardless of the control type. A control class is defined at project level for each theme in the theme editor.

The following steps are required to create a class theme:

1. Open a theme in the theme editor, right-click on **Control Classes** and select **Create new Control Class**:

2. Added control classes that do not yet contain definitions are initially grayed out in all themes:

3. You can define any number of attribute properties for a control class. All control attributes are available:

4. The control class must be assigned to a control for the properties to apply to the control. Open the Properties window for a control and select the entry **ClassNames** under **Common**:

5. In the dialog, select the control classes that you want to add to the control. To do this, select the required control class and press the arrow. Confirm the dialog with **OK**:

⇨ The control class is now assigned to the control, and the properties of the control class are used.

> If you add several classes to a control and each class describes the same attributes, the properties of the last class in the list apply (analogous to the class selectors in CSS).

To display the attribute properties of the class theme, the attribute must not be explicitly set at the control. The control colors must be set to Theme.

10.2.2 Control theme

A control theme defines the properties for all instances of a control. The control theme is defined for each theme in the theme editor [▶ 561] at project level.

The following steps are required to create a Control theme:

1. First, the control type must be selected in the theme editor. In the upper screenshot the control type **Button** was selected.
2. After selecting the control type, select the attribute from the respective changeable attributes.
3. Drag and drop the attribute onto the large area for the attribute properties in the center of the editor.
4. You can define any number of additional attributes for the control type.

Control types that contains defined attribute properties are shown in **bold** font. To change the attribute properties for a control type at a later stage, select the respective control type again. The existing configurations are then automatically displayed in the center of the theme editor. A property that has already been configured can be deleted by selecting it and then pressing the Delete key.

> To display the attribute properties of the Control theme, the attribute must not be explicitly set at the control. The control colors must be set to Theme.

10.2.3 CSS theme

Beside the control theme and the class theme there is the possibility to add [▶ 559] any number of Cascading Style Sheets files to a theme. A distinction is made between a normal CSS file and a CSS control theme file.

CSS files can also be added at project level, independent of a theme. These files define general properties such as the inclusion of fonts.

Cascading Style Sheets file

A Cascading Style Sheets file allows the definition of any CSS properties for all elements contained in the project (standard controls, framework controls etc.). In addition, specific CSS classes can be overwritten by a control. All CSS selectors and CSS properties are available within a CSS file.

If a class theme [▶ 562] is defined in the theme editor, it can be addressed in the CSS code via the class name with the prefix `tchmi-class-`. Example:

```
tchmi-class-myclassthemeclass
```

CSS control theme file

A CSS control theme file can be used to replace specific CSS properties of the base theme. This is made possible by importing all CSS properties of the control into the CSS control theme file of the project.

In the next window you have to select the control for which you want to create a CSS control theme file.

In this dialog you can select several controls for which a CSS control theme is to be generated. You must also select from which theme the CSS control theme should be derived. If you select the Base theme, all properties defined in the Base theme of the control are displayed.

Some controls contain further TwinCAT HMI Controls, e.g. the Event Grid. This is indicated in the dialog by a note (upside down exclamation mark) after the control. If you want to overwrite all properties, you must also include the CSS files of the child controls. For more detailed information about the dependencies between the individual controls, see the documentation for each control.

The following example shows a CSS control theme file for the control TcHmiButton [▶ 167]:

```
/** Styles for the theme: Base */

/* Style for the main element */
.tchmi-button {
   /* color gradient for default view */
   background-image: linear-gradient(135deg, #eff1f3, #aeb9c2);
   /* default color for button text */
```

```
   color: #4794da;
   /* default box shadow */
   box-shadow: 0px 0px 5px 0px rgba(0,0,0,0.6);
}

/* class down will be set/unset in the control on mouse/touch down */
.tchmi-button.down {
   /* another color gradient */
   background-image: linear-gradient(135deg, #aeb9c2, #eff1f3);
   color: #000000;
   box-shadow: inset 0px 0px 5px 0px rgba(0,0,0,0.6);
}
```

10.3 Theme switching

You can switch the entire theme in the project properties, the theme editor or during runtime using a function. Controls allow you to switch between different theme classes within a theme at runtime. Individual control attributes can be assigned a theme property at runtime if they have been overwritten previously.

Switching in the properties

The default theme can be viewed and switched under the general project properties (click on the project node and open the Properties window).

The theme editor [▶ 561] features the **Default Theme** property in the header bar at the top, irrespective of the theme. There you can set the default theme in the same way as in the project properties.

Switching at runtime

The theme can be switched during runtime in the browser for each client [▶ 15]. To do this, an action must be configured, e.g. on the **.onPressed** event of a button (so that the theme is switched when a specific button is pressed). In the **Actions and Conditions** editor, the **Theme** folder contains the **SetTheme** function under functions. Use drag & drop to add this function under the actions and enter the name of the theme by selecting it in the combo box.

Switching classes

In certain cases it is necessary to switch not the entire theme, but only the properties of a group of elements or a particular control instance during runtime. This is possible if different theme classes [▶ 562] are defined for the control. During runtime the attribute ClassNames [▶ 439] of the control is then set again. To do this, an action must be configured at runtime, as for theme switching.

Switching from control attributes to a theme

A control attribute, which is set by an attribute definition directly at the control (Level 1 [▶ 570]), can be set to the value of the currently active theme during runtime with the SetAttributeToThemeValue [▶ 108] function. The value of the attribute is then changed according to the active theme each time the theme is switched.

This function is used if individual control attributes are to be overwritten and then reset to the value of a theme.

10.4 Concept

Theming in TwinCAT HMI distinguishes between control-based themes and class-based themes. A control theme [▶ 565] specifies properties that apply to all instances of the respective control type. A class theme [▶ 562] specifies properties that only apply to controls to which this class is assigned. You can assign several classes to a control.

The theme properties can be set at different levels. Cascading Style Sheets use a comparable approach in the sphere of web development, for example. The levels determine which property applies to an element if different properties are defined for an element. The properties of the higher levels are overwritten by the properties of the lower levels, if these are defined. The lowest level is level 1, which overwrites all properties of the higher levels.

The theme system has the following levels:

Level 1: Attribute level [▶ 572]

This level describes the appearance using attribute definitions at the control. In analogy to the CSS world, this level would be a "style" attribute that is defined directly at the element.

Levels 2-4: attribute levels [▶ 572]

Like level 1, these levels describe the appearance using attribute definitions at the control. In analogy to the CSS world, these levels would be external CSS files.

Level 5: attribute levels [▶ 572]

Like levels 1-4, this level describes the appearance using attribute definitions at the control. If nothing is specified above levels 1-4, this defaultValueInternal value applies. In the analogy to the CSS world these levels would be comparable to the font color black, for example, which applies even if nothing else has been defined.

Levels 6-9: element levels [▶ 572]

These levels describe the appearance of a control using Cascading Style Sheets files if the appearance of a control cannot be defined using an attribute or has not been defined using an attribute.

10.4.1 Attribute levels

The attribute levels describe the appearance of the controls using attribute definitions.

Level 1: Attribute definition at the control at project level

The attribute definition on a control instance overwrites all properties defined by a theme. The attribute definition at the control [▶ 132] is specified in the Properties window or directly in the HTML code. The attribute definition at the control follows the usual procedure for overwriting the default properties of the base theme if no other theme is defined.

This option has been available since version 1.8.

Level 2: Attribute definition in a class at project level

The attribute definition in a class [▶ 562] is specified in the theme editor of the respective theme at project level. The class is added to a control as a property so that it automatically adopts the attribute definitions of the class. The attribute definition in a class overwrites the attribute definition for a control type if both definitions contain the same attribute.

Level 3: Attribute definition per control type at project level

The attribute definitions in a control type [▶ 565] are specified in the theme editor of the respective theme at project level. The attribute definition in a control type applies to all instances of the respective control in the project and is only overwritten by level 1 or level 2.

Level 4: Attribute definition per control type at control level

The attribute definition per control type at control level [▶ 2141] is specified directly in the directory of the control and is defined by the control developer. The definition is the same as in level 2 within a .theme file. For the standard controls, this property cannot be changed at the control level. The attribute definition per control type at control level is therefore only available for framework control developers.

Level 5: Attribute definition through DefaultValueInternal at control level

The DefaultValueInternal [▶ 2134] defines the properties of an attribute independent of the active theme and is used if the attribute is not overwritten by a theme at the higher levels.

In addition to the attribute definitions, it is possible to configure so-called ThemedResources at levels 2 to 4. The ThemedResources are control properties that cannot be configured via the Properties window and can only be changed via the theme system. An example of this are the knob definitions [▶ 327] at the LinearGauge. The definition [▶ 2134] of ThemedResources is handled by the control developer.

10.4.2 Element levels

The element levels describe the appearance of the controls using Cascading Style Sheets files.

Level 6: Cascading Style Sheets per theme at project level

At project level it is possible to add, in addition to the .theme file of the theme editor, any number of CSS files to a theme in the project (see Introduction [▶ 559]).

> If a control is described in the theme editor and also in a CSS file within the theme, the definitions within the theme editor (lower level) apply.

Level 7: Cascading Style Sheets at project level

At the project level, any number of Cascading Style Sheets files can be defined, independent of a theme. This option has been available since **version 1.8** under the name CSS-Behind file. The definitions within the CSS file at the project level overwrite the CSS definitions at the control level and apply regardless of the theme.

Any CSS-Behind files at project level, which were already added to the project with **version 1.8**, can still be used. When using different themes, it usually makes sense to assign this file to a specific theme.

Level 8: Cascading Style Sheets per theme at control level

Any number of Cascading Style Sheets files can be added to a theme at the control level. The CSS definitions at control level describe the normal layout of all elements of a control. These properties apply if no properties of the control are overwritten by a theme at the higher levels. Each control should implement at least one theme called Base.

Level 9: Cascading Style Sheets at control level

At the control level, you can add any number of general Cascading Style Sheets files that describe the properties of elements, regardless of the theme.

11 Historical data

Historical data are data from earlier times that are stored in a database.

Arbitrary server symbols of the TwinCAT HMI (e.g. numerical values from the PLC) can be historized by the TcHmiSqliteHistorize extension. The extension saves the data in a ring buffer in an SQlite database. The interval at which the data are saved in the database can be defined for each symbol. The data are saved in the database only as long as the server is running. No historical data are written to the database if the server is terminated.

The historical data are displayed in an included control – the Trend Line Chart. The control enables the display of various historical data at the same time.

Available since version 1.10

11.1 TcHmiSqliteHistorize extension

When creating a new TwinCAT HMI project the TcHmiSqliteHistorize extension is also loaded and is located in the project under the server node with the Extensions [▶ 117].

Configuration

The extension configuration page is opened with a double-click on the extension in the Solution. Alternatively this can also be called via the server configuration page on the **TcHmiSqliteHistorize** tab.

On the configuration page you have the option to create new historized symbols and to define the storage location of the database and all historized symbols. By default the historized symbols are stored on the local hard drive. Alternatively they can be saved directly in the computer's RAM.

NOTE

Loss of data

If saved in the RAM, the data will be lost in the event of the PC being (unexpectedly) shut down.

Adding a historized symbol

In order for a symbol to be used as a historized symbol, it must be mapped [▶ 69]. A historized symbol can be added in two different ways:

Configuration window (recommended):

A historized symbol can be added in the TwinCAT HMI Configuration [▶ 65] window, where a dialog is available for adding the symbols. Instructions for this can be found here [▶ 71].

Extension configuration:

A new historized symbol can be added via the **Add** button on the configuration page of the TcHmiSqliteHistorize extension. A dialog then opens for adding the symbol.

1. **Linked Server Symbol**: Mapped server symbol to be historized. All mapped server symbols are displayed in the combobox.

2. **Interval Time**: Interval at which the symbol is saved in the database.
3. **Maximum number of Entries**: Maximum number of entries in the ring memory of the database. On the basis of the interval time and the maximum number of entries you can calculate the time period for which the data can be stored. If the time period is to be extended you can extend the interval time or increase the maximum number of entries.
4. **Recording enabled**: Defines whether the symbol is to be historized or whether historization is to be deactivated.
5. **Row limit for filter**: Maximum number of data records that are filtered by the TcHmiSqliteHistorize extension and forwarded to the trendline chart.

11.2 Trend Line Chart Control

The Trend Line Chart Control [▶ 393] enables the display of the historized symbols. The Trend Line Chart Control can display several historized symbols simultaneously. In addition, time periods from the past can be displayed and the data they contain enlarged and observed in detail.

The Trend Line Chart Control is located in the Toolbox under the category **Beckhoff**, from where you can insert it into an HMI page by drag-and-drop.

The control offers various configuration options in the engineering and during the runtime in the browser.

Detailed view

If the live view in the Trend Line Chart has been paused, you can view individual data points with the aid of the zoom functions. Hovering with the mouse over a data point will bring up a display of detailed information on the time and data point.

Configuring axes

If you wish to display a historized symbol in the Trend Line Chart you must first carry out a Y-axis configuration [▶ 400] for the symbol. Select the control and open the **Properties** window, where you select the property **Y Axis** under the category **Y Axis**. An editor then opens that enables the configuration of a Y-axis as well as the addition of further axes. Configure the axis [▶ 402] according to the settings that you need for the historized symbol (e.g. min. and max value on the Y-axis).

Then confirm the dialog with **OK** and open the descriptions of the chart [▶ 401] (LineGraphDescriptions) in the category **Common** in the **Properties** window, where the configured Y-axes are linked with the historized symbols. Add a new description and select the historized symbol as well as the associated Y-axis. The available historized symbols or Y-axes respectively are displayed in the comboboxes.

Confirm the dialog with **OK**. You can then start the LiveView or publish the project on a server and view the historized symbol in the chart.

Configuring the interval time

You can configure the update rate of the control in the Trend Line Chart Control. The update rate can be found under the entry **Interval** in the category **Common**. The update rate is specified in milliseconds.

If you have a historized symbol with an update rate of 100 ms, you should also set the update rate of the control to 100 ms, as otherwise the control will be updated, for example, at 1-second intervals, even though your data are already available every 100 ms.

Configuring the time period

The start and end times of the historized data displayed can be configured in the control's Properties window. The times can be specified as time interval, point in time or "First or Latest" respectively.

In addition, it is possible to set the start and end times of the historized data displayed in the browser for each client (see menu bar).

Menu bar

The Trend Line Chart Control has a menu bar that can optionally be displayed under the properties of the control (**ShowMenuBar** in the category **MenuBar**). The menu bar enables the analysis of the data at runtime in the browser:

1. **Start time**: Start time of the historized data displayed (**First** for the beginning of the recording).
2. **Select time interval**: Select the time interval for the start time (e.g. two minutes before the last value).
3. **Select point in time:** Select point in time for the start time (e.g. 6/16/2018 at 5:53 pm).
4. **End time:** End time of the historized data displayed (**Latest** for the last recorded value).
5. **Select time interval**: Select the time interval for the end time.
6. **Select point in time:** Select the point in time for the end time.
7. **Start live view**: The live view can be started after pausing the live view.
8. **Pause live view**: The live view can be paused in order to view data points in greater detail.
9. **Shift data on the X-axis**: The data displayed can be shifted on the X-axis if the live view has been paused.
10. **Shift data on the X and Y-axes**: The data displayed can be shifted on all axes if the live view has been paused.
11. **Zoom X-axis**: When this button has been activated you can zoom in on the X-axis with the aid of the mouse.
12. **Zoom X-Y-axis**: When this button has been activated you can zoom in on both axis equally with the aid of the mouse.
13. **Reset Zoom**: This button resets all zoom settings.

Further configuration options

Further configuration options for the Trend Line Chart Control can be found here [▶ 393].

Example

An example of the use of the historized symbols in interaction with the Trend Line Chart Control: Example [▶ 2169].
The example shows two historized symbols (the associated TwinCAT project is included) and the associated configuration in the Trend Line Chart Control.

12 Recipe management

The recipe management enables the activation of symbol values that are managed in recipes. Furthermore, online values from the development environment and from the visualization clients can be saved by the target system in a recipe.

Recipe types and recipes

A fundamental distinction is made between recipe types and recipes. Comparable with a class in the object orientation, a recipe type is a general description of a set of symbols. On the basis of this description you can create various recipes, comparable with an instance of a class. Specific values for the symbols defined in the recipe type can be saved in a recipe.

The definition of colors as color codes is used as an example. In this example the recipe type Color describes the general structure of a color with its red, green and blue components. Three color recipes for yellow, light gray and navy are created on the basis of this description. The color components defined for these colors are each saved in the three different recipes.

Inheritance of symbol configurations

If, by default, certain recipe types are to have a basic set of symbols managed in a central place, then these recipe types can inherit [▶ 586]from one or more recipe types. As a result they contain the symbol entries configured in these parent recipe types. These entries are automatically updated in case of changes being made in the parent recipe types.

Reuse of recipes in other recipes

In order to reuse [▶ 587] already defined recipes in other recipes, a recipe type may possess one or more recipe types as members, comparable with standard symbol entries. To do this you can drag-and-drop the recipe type of the recipe to be reused from the TC HMI Configuration window into the Recipe Editor [▶ 591] in the basic recipe type and then enter the recipe to be reused into the recipes derived from it.

Available since version 1.10.1171.142

See also

Actions and Conditions Editor [▶ 50]

Recipe [▶ 106]

12.1 Management in the engineering

The management of recipe types and recipes is integrated in the TwinCAT HMI configuration window [▶ 65] under **Recipe Management**, where all available recipes are listed under the entry **Recipes** and all recipe types under **Recipe Types**. Both the recipes and the recipe types can be managed there in a self-made folder structure.

12.1.1 Create folder

Folders can be created for the structuring of recipe types and recipes. The folders can be nested in one another as required. The folders represent the relative path of a recipe.

Create recipe type folder:

Click on **Create new Recipe Type Folder…**. in the configuration window.

Adding a recipe folder:

Click on **Create new Recipe Folder….** in the configuration window

You can then insert recipe types, recipes or further folders in the folders.

Existing folders can be deleted with a right click on **Delete** and renamed by right-clicking on **Rename**.

12.1.2 Create new recipe type

1. In the TwinCAT HMI configuration window, right-click on **Recipe Types** and select **Create new Recipe Type** in the context menu or click on the **Create new Recipe Type** button in the toolbar.

2. Enter a name for the new recipe type.
3. Confirm the dialog with **OK**.

⇨ You have now created a new recipe type.

The Recipe Type Editor [▶ 584] is opened with a double click on the recipe type or by right-clicking on **Edit**.

12.1.3 Changing the recipe type

In the TwinCAT HMI configuration window, right-click on the recipe type to be changed and select **Edit** in the context menu or double-click on this recipe type to open the Recipe Type Editor [▶ 584].

12.1.4 Recipe Type Editor

A recipe type can be configured in the Recipe Type Editor. The editor has a toolbar in the upper area, a configurable list of server symbols in the central area and general settings for the entire recipe type in the lower area.

New symbols can be added directly to the recipe type by drag-and-drop on the TwinCAT HMI configuration window. Alternatively, they can be added to the toolbar via the **+** button.

Symbol list

Server symbols that are to be added to a recipe type can either be placed in the symbol list by drag-and-drop from the TwinCAT HMI configuration window or selected in the symbol selection window, which is accessible via the toolbar [▶ 592]. The symbol list has different columns.

Display Name	The name to be used for the symbol entry in the recipes derived from the recipe type is defined here.
Symbol	This entry is automatically set by drag-and-drop.
Default	Define a default value here that is adopted automatically when creating a new recipe. **Note:** A value must be entered here so that the recipe type can be saved. The **Read Default Values from Target** function in the toolbar can be used to adopt the current values of all symbols from the PLC into this field.
Min	If necessary, define a minimum value for this symbol here that can be used in the recipes derived from the recipe type. **Info:** This field is available only with numerical values and enumerations.
Max	If necessary, define a maximum value for this symbol here that can be used in the recipes derived from the recipe type. **Info:** This field is available only with numerical values and enumerations.
Unit	If necessary, enter a unit for the symbol here.
Enabled	Activate or deactivate individual symbols for the recipe type here. **Note:** This setting is automatically adopted for the recipes derived from the recipe type. On activating one of these recipes, the values of the deactivated symbol entries are not written to the corresponding server symbols.
Comment	If necessary, enter a comment for the symbol entry here.

You can mark several symbol entries in the symbol list at the same time by keeping the Ctrl key pressed while selecting the individual entries. This allows you, for example, to delete several entries at the same time with the Delete key or the X button.

Other recipe types can also be added by drag-and-drop in the symbol list for the reuse of recipes in other recipes [▶ 587].

Toolbar

You can execute various functions via the toolbar in the upper area of the editor.

	Delete selected rows	The selected entries in the symbol list are deleted from the recipe type.
	Create new group	A new group in which the symbol entries can be summarized is created in the symbol list.
	Read default values from target	The default values are read from the target and entered for all symbol entries.
	Read default values of the selected rows from the target	The default values are read from the target and entered only for the selected symbol entries.
	Add symbols to this recipe type	The symbol selection window is opened, where you can select the desired symbols and add them to the recipe type as an alternative to drag-and-drop.

General settings

In the lower part of the editor you can configure generally valid settings for the entire recipe type.

Inherited From	Select one or more recipe types from which the currently opened recipe type should inherit [▶ 586].
Enabled	Activate or deactivate the entire recipe type here. **Note:** The recipes derived from the recipe type are deactivated automatically.

Comment	If necessary, enter a comment for the entire recipe type here.

12.1.4.1 Inheritance of recipe types

A recipe type can inherit from one or more recipe types. In the process, all symbols of the basic recipe type are adopted into the derived recipe type (comparable with inheritance in object-oriented programming). Inheritance hierarchies can be constructed as desired through the support of inheritance. Changes in the basic recipe type are automatically adopted into the derived recipe types.

1. Create a new recipe type.
2. Select the basic recipe type in the lower status bar under "Inherited From:" via the "…" button:

3. In the subsequent window you have the possibility to select the basic recipe type:

Multiple selection is possible by keeping the "Ctrl" key pressed.
Note: A recipe type cannot inherit from itself.

4. The symbols are inherited by the derived recipe type. The symbols are displayed grayed out since a change is only possible in the basic recipe type.

12.1.4.2 Reuse of recipe types

A recipe type can contain further recipe types as members. This enables several recipe types to be reused in a recipe type. When creating a recipe, other recipes (instead of a value as in the case of the symbols) are stored for the individual recipe type members.

1. Create a new recipe type.

2. Add a recipe type from the configuration window to the new recipe type by drag-and-drop.

3. The recipe type is now contained as a member. The recipe type can subsequently be replaced by another recipe type by clicking on the **...** button behind the symbol name.

12.1.5 Deleting a recipe type

1. In the TwinCAT HMI configuration window, right-click on the recipe type to be deleted and select **Delete** in the context menu.

2. Confirm the dialog with **OK** if you wish to delete the selected recipe type(s) including all recipes derived from it/them.

12.1.6 Creating a new recipe

1. In the TwinCAT HMI configuration window, right-click on **Recipes** and select **Create new Recipe** in the context menu or click on the **Create new Recipe** button in the toolbar.

2. Enter a name for the new recipe in the **Create new Recipe** dialog.

3. Using the **…** button, select one of the existing recipes on which the new recipe is to be based.
4. Confirm the dialog with **OK**.

The Recipe Editor [▶ 591] is opened by double-clicking on the recipe or right-clicking on **Edit**.

12.1.7 Changing the recipe

1. In the TwinCAT HMI configuration window, right-click on the recipe to be changed and select Edit in the context menu or double-click on this recipe to open the Recipe Editor [▶ 591].

12.1.8 Recipe Editor

A recipe can be managed in the Recipe Editor. The editor has a toolbar in the upper area, a symbol list in the central area and general information in the lower area.

Symbol list

All symbol entries with the information defined in the recipe type are shown in the symbol list. Here you can change the values of the individual entries, save them together with the recipe and manage them via the toolbar. [▶ 592] You can mark several symbol entries at the same time by keeping the Ctrl key pressed while selecting the individual entries.

Name	The name of the symbol entry is shown here.
Value	Define a value for the symbol entry here. **Note:** When creating a new recipe the default value is automatically adopted from the recipe type.
Min	The lowest possible value for the symbol is shown here.
Max	The highest possible value for the symbol is shown here.
Unit	The unit of the value is shown here.
Comment	The comment for a symbol entry is shown here.

Toolbar

You can execute various functions via the toolbar in the upper area of the editor.

	Activate recipe	The values saved in the recipe are written to the symbols.
	Read values from target	The current values of all symbols are read from the target and written to the recipe.
	Read values for the selected symbols from target	The current values of the selected symbols are read from the target and written to the recipe.
	Show the online values for the symbols	The current values of the symbols are displayed in an extra column.
	Import recipes	The recipes selected in the dialog are imported into the server.
	Export recipe	The opened recipe is exported with a name that is to be defined.

General Information

General information about the recipe is displayed in the lower part of the editor.

Based on Recipe Type	The recipe type on which the recipe is based is displayed here.

12.1.9 Delete recipe

1. In the TwinCAT HMI configuration window, right-click on the recipe to be deleted and select **Delete** in the context menu.

12.2 Management in the client

The recipe management can also be used in the client at runtime. Basic functionalities are mapped via the Actions and Conditions Editor [▶ 50], so that preconfigured recipes can simply be activated or newly learned [▶ 106] at runtime. The Framework API [▶ 1956] offers the option to run all the functions, which are also available in the engineering, in the client by means of Code Behind. The recipe management example shows the use of the Framework API at runtime.

Authorization concept

If user authentication is active, [▶ 521] you must explicitly enable access to the recipe configuration for the desired user groups. Access is not granted by default. Open the authorization system for the symbols [▶ 530] in the extended view [▶ 65] and search for the term "TcHmiRecipeManagement.Config". You can assign the corresponding authorizations on this symbol.

12.2.1 Example

The recipe management example shows the use of the recipe management [▶ 581] at runtime. It thereby extends the functions available in the Actions and Conditions Editor [▶ 50] by the use of the Framework API. The example serves as the point of entry to the recipe management and can be extended as desired. In parallel with the example project, observe the chapter "Management in the engineering [▶ 582]".

The example consists of a TwinCAT HMI project and a TwinCAT project with example symbols in the PLC. The example contains a total of three symbols, which are used by two different recipe types. Symbol 1 and Symbol 2 are the symbols from the PLC and Symbol 3 is an offline symbol [▶ 69]. The offline symbol illustrates that symbols from other sources (OPC UA, use-specific server extension, etc.) can also be used as the ADS extension.

The example can be downloaded here [▶ 2169].

User interface

The example has a user interface in the client that can execute various functions of the recipe management at runtime. These are described in more detail below.

1. Recipe selection: All available recipe types are displayed in the combo box. If there is a recipe inside a folder, the relative path to the recipe is displayed. The combo box is used to select the recipe that is used as the basis for the other buttons and fields in this example.

2. Teach and activate recipe: The recipe currently selected in the combo box can be edited in this area.

- Teach selected recipe: The currently selected recipe is taught again. This means that the current online values are adopted from the PLC and saved in the recipe.
- Teach as new recipe: A new recipe is created with the current online values from the PLC on the basis of the currently selected recipe. A new name for the recipe must be entered in the text box.
- Activate selected recipe: The currently selected recipe is activated. This means that the values stored in the recipe are written to the corresponding symbols in the PLC.
- Delete selected recipe: The currently selected recipe is deleted.

3. Upload and download recipe: This area enables recipes to be downloaded and uploaded. The file browser of the respective browser that enables the selection of the storage location or the selection of the recipe thereby opens.

The upload and download functions are not available in the LiveView in Visual Studio.

4. Display and edit recipe: This area enables the display and editing of the recipe members (symbols) of the currently selected recipe.

- Datagrid: The currently selected recipe is loaded and the individual members are displayed in the data grid. The "New value" column has two different functions. Firstly, the column is used to receive new values for the individual members of the recipe, which are subsequently saved by clicking on the **Update selected recipe** button. Secondly, the column is used to display the current online values of the members for comparison with the stored values by clicking on the **Read values from target** button.
- **Update selected recipe**: This button saves the recipe with the new values specified in the "New value" column.
- **Read values from target**: This button reads out the current online values of the recipe members and displays them in the "New value" column.

5. Online recipe values: The current online values of the three symbols Value 1, Value 2 and Value 3 are displayed in this area.

12.2.1.1 Logic in the background

The example extends the "Activate [▶ 106]", "Teach [▶ 106]" and "TeachAsNew [▶ 107]" functions available in the Action and Conditions Editor by further user-specific functions that call the Framework API [▶ 1956] of the recipe management in the background. These functions can also be reused in other projects if they are imported accordingly.

The functions can be found in the HMI project in the **RecipeFunctions** folder.

Since all functions are called asynchronously [▶ 2112], they are additionally given a context object [▶ 2119]as a transfer parameter. This provides information about the success or failure of the function.

ConvertDatagridToRecipe

The ConvertDatagridToRecipe function reads the current contents of the data grid and saves them in the transferred recipe. This function is executed by clicking on the **Update selected recipe** button.

Parameter:

Name	Type	Description
datagridControl	tchmi:framework#/definitions/Control [▶ 2216]	The TcHmiDatagrid, in which the recipe members are to be displayed.
recipeReference	tchmi:framework#/definitions/RecipeReference [▶ 2249]	The reference to the recipe that is to be described. The reference consists of the recipe path (with folder structure).

Return value:

Type	Description
void	No return value.

ConvertRecipeToDatagrid

The ConvertRecipeToDatagrid function displays all members of the transferred recipe in a data grid. The function is transferred to the SrcData attribute of the data grid when changing the selection in the combo box in order to load the newly selected recipe.

Parameter:

Name	Type	Description
recipeReference	tchmi:framework#/definitions/RecipeReference [▶ 2249]	The reference to the recipe that is to be displayed. The reference consists of the recipe path (with folder structure).

Return value:

Type	Description
void	No return value.

The data set for the data grid is transferred to the context object.

DeleteRecipe

The DeleteRecipe function deletes a transferred recipe.

Parameter:

Name	Type	Description
recipeReference	tchmi:framework#/definitions/RecipeReference [▶ 2249]	The reference to the recipe that is to be deleted. The reference consists of the recipe path (with folder structure).

Return value:

Type	Description
void	No return value.

DownloadRecipe

The Download function enables the download of a transferred recipe as a JSON file. This can then be imported using the UploadRecipe function.

Parameter:

Name	Type	Description
recipeReference	tchmi:framework#/definitions/RecipeReference [▶ 2249]	The reference to the recipe that is to be downloaded. The reference consists of the recipe path (with folder structure).

Return value:

Type	Description
void	No return value.

The upload and download functions are not available in the LiveView in Visual Studio.

ListRecipeAsReferenceArray

The ListRecipeAsReferenceArray function creates an array of all available recipes that is needed for the selection in the combo box. The function is used on the SrcData attribute of the combo box.

Parameter:

Name	Type	Description
recipeList	tchmi:server#/definitions/folderRecipe [▶ 2276]	The reference to all recipes. The list of all recipes is the "recipeList" symbol in the RecipeManagement extension.

Return value:

Type	Description
void	No return value.

The data set for the combo box is transferred to the context object.

UpdateDatagridFromTarget

The UpdateDatagridFromTarget function lists the current online values of the recipe members of the transferred recipe in the "New value" column in the data grid. The function is executed by clicking on the **Read values from target** button.

Parameter:

Name	Type	Description
datagridControl	tchmi:framework#/definitions/Control [▶ 2216]	The TcHmiDatagrid, in which the online values are to be displayed in the "New value" column.
recipeReference	tchmi:framework#/definitions/RecipeReference [▶ 2249]	The reference to the recipe that is to be displayed in the data grid. The reference consists of the recipe path (with folder structure).

Return value:

Type	Description
void	No return value.

UploadRecipe

The UploadRecipe function loads a recipe from a JSON file into the recipe management. Recipes that were exported using the DownloadRecipe function can be imported with this function.

Parameter:

Name	Type	Description
-	-	-

The recipe in the form of the JSON file is selected directly via the file browser in the browser.

Return value:

Type	Description
void	No return value.

The upload and download functions are not available in the LiveView in Visual Studio.

12.2.1.2 Extension with filter function

The sample can be extended by a filter function that filters the list of recipes based on the recipe name.

You can download the extension of the sample here [▶ 2169].

BECKHOFF
Recipe Management Sample
Filter:
Recipe1
Selected recipe:
Recipe1
Teach selected recipe
Activate selected recipe
Download recipe
New recipe name:
NewRecipe
Teach as new recipe
Delete selected recipe
Upload recipe
Member name
Saved value
New value
Value1
111
111
Value2
222
222
Update selected recipe
Read values from target
Online values:
Value 1: 0
Value 2: 0
Value 3: 0

13 Protocols

13.1 ADS

The ADS server extension displays the configured runtime of the selected publish configuration. You can add further devices.

All locally configured routes are displayed with their AmsNetId when adding a runtime. A runtime is added via the AmsNetId.

13.1.1 Access by IndexGroup and Offset

It makes most sense to access PLC variables by symbol name in order to detect changes in the symbolism (e.g. due to an OnlineChange). Small controllers (e.g. BC/BX) that have no symbolism can be addressed directly via IndexGroup and Offset. Via the configured ADS runtime you can add a new symbol that fetches the values by IndexGroup and Offset (hex values are converted to the corresponding decimal values).

13.2 OPC UA Client

The OPC UA Client Extension enables communication between the TwinCAT HMI Server and an OPC UA Server. The TwinCAT HMI Server connects as a client to the OPC UA Server. This can be used, for example, to connect third-party controllers that do not support the ADS protocol to the TwinCAT HMI. The data exchange is bidirectional. The data points of the OPC UA Server are automatically browsed by the OPC UA Client Extension.

To use the OPC UA Client Extension the license TF2110 is required. The license allows the connection to OPC UA servers and the use of a namespace. One TwinCAT HMI Target license (TF2050 - TF2090) per namespace is required for the use of additional namespaces.

Available since version 1.10.1336.0

13.2.1 Add extension

The OPC UA Client Extension is not loaded by default and must be added later to a TwinCAT HMI project. The extension is added via TwinCAT HMI Package Management [▶ 118]. When TwinCAT HMI is installed, the extension is automatically stored in the local repository and is therefore listed directly in the Package Manager.

Open the Package Manager and install the extension [▶ 123].

13.2.2 Configure connection

The connection to an OPC UA Server is configured on the configuration page of the extension.

Open the TwinCAT HMI Server configuration page [▶ 117] and navigate to the **TcHmiOpcUa** tab:

1. **advancedSettings**: Under the advanced settings you can create profiles for the display of the browsed variables.
2. **connectionSettings**: You can create profiles for establishing a connection under Connection Settings.
3. **namespaces:** The connection to an OPC UA server is configured under the namespaces. The configuration profiles can be selected for each connection.
4. **securitySettings:** Under the security settings you can create profiles for the OPA UA Security.

Connecting to an OPC UA Server

1. Open the TwinCAT HMI Server configuration page and navigate to the **TcHmiOpcUa** tab.
2. Open the point **namespaces** and add a new entry by clicking on the button **+ Add**.
3. Configure the connection to the OPC UA Server:

Name: Freely selectable name under which the OPC UA Server is listed in TwinCAT HMI.
advanced: Profile of the advanced settings.
connection: Profile of the connection settings.
endpoint: Endpoint URL of the OPC UA Server.
security: Profile of the security settings.

⇨ You can then use [▶ 604] the symbols of the OPC UA Server.

13.2.3 Use data points

After the connection to the OPC UA Server has been configured, the symbols of the OPC UA Server can be accessed. Access is via the TwinCAT HMI Configuration window [▶ 65]. There the symbols can be used in the [▶ 67] same way as with the other extensions.

The OPC UA Server is listed below the extension "TcHmiOpcUa" with the configured name. [▶ 603]

TwinCAT HMI Configuration
Enter a search term...
Server Symbols
All Symbols
Mapped Symbols
Historized Symbols
Internal Symbols
Localizations
Data types
Controls
Global Events
Functions
Files
All files
Content files
Views
User Controls
Users and Groups
Users
Groups
Client Certificates
Action Templates
Recipe Management
Recipes
Recipe Types
Name
Datatype
Mapped to
ADS object
TcHmiOpcUa
Root
basic #/definitions/UA.ID.basic.85.TCHMI_GENERIC_TYP
TcOpcUaServer #/definitions/UA.ID.TcOpcUaServer.85.TCHMI_GENERIC_TY
Configuration #/definitions/UA.ID.TcOpcUaServer.16.TCHMI_GENERIC_TY
DeviceSet #/definitions/UA.ID.TcOpcUaServer.5001.TCHMI_GENERIC_
PLC1 #/definitions/UA.ID.TcOpcUaServer.PLC1.TCHMI_GENERIC_
DeviceManual G String
DeviceRevision G String
DeviceState #/definitions/UA.ID.TcOpcUaServer.20737.TCHMI_GENERIC
GVL #/definitions/UA.ID.TcOpcUaServer.GVL.TCHMI_GENERIC_T
GVL_dynamic #/definitions/UA.ID.TcOpcUaServer.GVL_dynamic.TCHMI_G
GVL_static #/definitions/UA.ID.TcOpcUaServer.GVL_static.TCHMI_GEN
HardwareRevision G String
MAIN #/definitions/UA.ID.TcOpcUaServer.MAIN.TCHMI_GENERIC
Manufacturer S UA.LocalizedText
Model S UA.LocalizedText
Programs #/definitions/UA.ID.TcOpcUaServer.PLC1.Programs.TCHMI_
RevisionCounter G INT32
SerialNumber G String
SoftwareRevision G String
Tasks #/definitions/UA.ID.TcOpcUaServer.PLC1.Tasks.TCHMI_GEN
Server #/definitions/UA.ID.TcOpcUaServer.2253.TCHMI_GENERIC_
TwinCAT HMI Configuration
Logged Events
Output
Error List

14 Server

14.1 Server-to-server communication

Server-to-server communication enables communication between TwinCAT HMI servers via WebSockets. Using server-to-server communication, any data points can be exchanged between the servers. The prerequisite for this is that the symbols are mapped on the target server. [▶ 67]

Server-to-server communication can be used in different scenarios. For example, it is possible for a server to tap the data points of any number of subordinate servers. It is also possible for one server to pass the data points on to another server, which in turn passes the data on to the next server.

Server-to-server communication requires a client license (TF2010 - TF2040) on the server to which another server wants to connect. In the scenario shown above, TwinCAT HMI Server 1, TwinCAT HMI Server 2 and TwinCAT HMI Server n require a client license if the higher-level TwinCAT HMI Server connects to them.

Available since version 1.10.1336.0

14.1.1 Configure connection

Server-to-server communication is set up on the configuration page [▶ 117] of the TwinCAT HMI server.

Add Remote Server

1. Open the configuration page of the server and navigate to the subpage **TcHmiSrv** → **Webserver**:

2. Under **Remote Server,** you can add a server that you want to connect to. Click on the entry **+ Add**.

Configuring the Remote Server

The connection to the remote server can be configured as follows:

1. **Domain**: The domain can be chosen freely. The remote server is listed in TwinCAT HMI under this name.
2. **Remote URL**: URL of the remote server. The complete address of the remote server must be specified here, including the protocol to be used. Using encrypted communication requires adding the certificate [▶ 608] of the remote server.
3. **Username (optional)**: The user name is required if authentication [▶ 521] is active on the remote server. Depending on the configured symbol rights [▶ 530], the different users have different access rights to the symbols. It is possible to log another user on to the remote server at runtime [▶ 608].
   If authentication is disabled on the remote server, the user does not need to enter anything.
4. **Password (optional)**: Password of the user logged on to the remote server.

5. **Timeout (optional)**: WebSocket timeout for communication with the remote server.
6. **Enabled:** Enables or disables the connection to the remote server.
7. **Add**: Adds the connection to the remote server.

14.1.1.1 Add remote server certificate

Communication with the remote server can be encrypted. In this case, the SSL certificate of the remote server must be exported and imported to the server that wants to connect to the remote server.

1. Export the certificate of the remote server. Open the configuration page of the remote server and click on "Export SSL Certificate".
2. Open the configuration page of the server you want to connect to the remote server.
3. Navigate under "TcHmiSrv" to the tab "Webserver" and click on the entry "Allowed certificates for remote servers".

4. Add a new certificate by clicking on the entry "+ Add".
5. Select the certificate of the remote server you want to import.

6. Confirm the dialog by clicking on "Add".

14.1.1.2 Changing the Configuration at Runtime

The configuration of the remote server can be changed at runtime. This makes it possible, for example, to log on to the remote server with another user and thus obtain different access rights to the symbols of the remote server. Access at runtime is via the Config symbol of the TwinCAT HMI server. Under the Config symbol you will find the configuration of the remote server, which contains the same options as the configuration page of the server.

REMOTESERVERS	object	TcHmiSrv.Config::REMOTESERVERS
[<key>]	object (map)	TcHmiSrv.Config::REMOTESERVERS[<key>]
REMOTE_CONNECT_TIMEOUT	string	TcHmiSrv.Config::REMOTESERVERS[<key>]::REMOTE_CONNECT_TIMEOUT
REMOTE_ENABLED	boolean	TcHmiSrv.Config::REMOTESERVERS[<key>]::REMOTE_ENABLED
REMOTE_PASSWORD	string	TcHmiSrv.Config::REMOTESERVERS[<key>]::REMOTE_PASSWORD
REMOTE_URL	string	TcHmiSrv.Config::REMOTESERVERS[<key>]::REMOTE_URL
REMOTE_USERNAME	string	TcHmiSrv.Config::REMOTESERVERS[<key>]::REMOTE_USERNAME

When mapping the symbols [▶ 67], the domain must be specified (e.g. MyRemoteServer). The symbols can then be described and read in the Actions and Conditions editor [▶ 53].

14.1.2 Use data points

After the connection has been configured, the mapped symbols of the remote server can be accessed. Access is via the TwinCAT HMI Configuration window [▶ 65] via the domain of the remote server. There the symbols can be used in the same way as with the other extensions [▶ 92].

The symbols can be mapped via the TwinCAT HMI Configuration window. Activating the extended view [▶ 65] (button "...") displays the extended mapped symbols of the remote server.

14.2 Extensions

The TwinCAT HMI Server has a modular structure and encapsulates individual functionalities in server extensions. This enables a flexible expandability of the server. This can be used, for example, to integrate business logic centrally in the server and to protect know-how. Possible application scenarios are, for example, the connection to a user-specific database or the implementation of communication protocols.

The development of server extensions is done in C#. The development takes place in a separate project type within the Visual Studio. A corresponding project template is available for this purpose.

The execution of a server extension on the TwinCAT HMI server (TF2000) requires a license for the extension SDK (TF2200).

The development of a server extension requires a full version [▶ 17] of Visual Studio.

14.2.1 First steps

In addition to the TwinCAT HMI project, add another Server Extension project to your solution:

In the following dialog you can choose which server version should be used for development:

If the project was compiled successfully, the extension can be loaded from the server. Open the context menu in the TwinCAT HMI project under Server/Extensions:

Activate the respective extension to be loaded from the Engineering Server:

The server tries to load the extension. The state of the loaded extension is symbolized in the project node Server/Extensions:

If a problem has occurred during loading, the error list or log from the server can help with error analysis.

After loading the extension, the extension symbols are available in the TwinCAT HMI Config window:

14.3 API Reference

14.3.1 TcHmiSrv.DynamicSymbols Namespace

Classes

	Class	Description
	Definition [▶ 621]	Contains the additional information for a definition that is required by GetSchema(Command, Definition) [▶ 677] and GetDefinitions(Command, IEnumerable.Definition.) [▶ 674].
	Symbol [▶ 646]	Contains the additional information for a symbol that is required by ListSymbols(Command, IEnumerable.Symbol.) [▶ 681].
	TcHmiResolutionException [▶ 655]	Represents errors that occur during resolution.
	TcHmiResolver [▶ 660]	**Obsolete.** Provides methods to resolve sub-objects of several types by a given Path.
	TcHmiResolver.T. [▶ 664]	Provides basic functions to resolve objects of type T by a given Path.
	TcHmiSchemaBuilder.TType. [▶ 666]	Provides basic functions to create instances of class Value that represent a schema builded of type TType.
	TcHmiSchemaBuilder.TType, TInfo. [▶ 667]	Provides basic functions to create instances of class Value that represent a schema builded of type TType with additional information of type TInfo.
	TcHmiSchemaGenerator [▶ 668]	Provides methods to get the schema of a particular type and store it in an instance of type Value.
	TcHmiSerializer [▶ 683]	Provides methods to serialize between instances of type Object and Value.
	TcHmiUniversalResolver [▶ 691]	Provides methods to resolve sub-objects of several types by a given Path.

Structures

	Structure	Description
	BuildInfo [▶ 614]	Contains additional information that is used in the Definition [▶ 621] class to pass additional information.
	SchemaKey.T. [▶ 644]	Represents a key of a schema.

Interfaces

	Interface	Description
	IDefinition.TType. [▶ 632]	Contains the basic information for a definition that is required by GetSchema.TType.(Command, IDefinition.TType., TcHmiSchemaBuilder.TType.) [▶ 678] and GetDefinitions.TType.(Command, IEnumerable.IDefinition.TType.., TcHmiSchemaBuilder.TType.) [▶ 675].
	IEditable [▶ 634]	Defines a method that will be called by CreateObject(Value, Type, Boolean, Boolean, ICollection.String., Boolean, UInt32, Boolean) [▶ 687] and EditObject(Object., String, Value, Boolean, Boolean, ICollection.String., Boolean, UInt32, Boolean) [▶ 688] to set a value to a specific member.
	IEditableCollection [▶ 636]	Defines methods that will be called by EditObject(Object., String, Value, Boolean, Boolean, ICollection.String., Boolean, UInt32, Boolean) [▶ 688] to get and set values in the inherited collection by index.
	IExtendedDefinition.TType, TInfo. [▶ 637]	Provides an interface to pass additional definition information to GetSchema.TType, TInfo.(Command, IExtendedDefinition.TType, TInfo., TcHmiSchemaBuilder.TType, TInfo.) [▶ 678] and GetDefinitions.TType, TInfo.(Command, IEnumerable.IExtendedDefinition.TType, TInfo.., TcHmiSchemaBuilder.TType, TInfo.) [▶ 675].
	IExtendedSymbol.TType, TInfo. [▶ 639]	Provides an interface to pass additional symbol information to ListSymbols.TType, TInfo.(Command, IEnumerable.IExtendedSymbol.TType, TInfo.., TcHmiSchemaBuilder.TType, TInfo.) [▶ 682].
	ISerializable [▶ 641]	Defines a method that will be called by CreateObject(Value, Type, Boolean, Boolean, ICollection.String., Boolean, UInt32, Boolean) [▶ 687] and SerializeObject(Object, IncludeMembers, BindingFlags, ICollection.String., Boolean, UInt32, Boolean) [▶ 690] to get the value of a specific member.
	ISymbol.TType. [▶ 642]	Contains the basic information for a symbol that is required by ListSymbols.TType.(Command, IEnumerable.ISymbol.TType.., TcHmiSchemaBuilder.TType.) [▶ 683].

Enumerations

	Enumeration	Description
	IncludeMembers [▶ 640]	Specifies flags that control which members are included in serialization and schema generation.
	JsonSchemaStringFormat [▶ 643]	Represents the string formats specified in the JSON Schema specification.

14.3.1.1 BuildInfo Structure

Contains additional information that is used in the Definition [▶ 621] class to pass additional information.

Namespace: TcHmiSrv.DynamicSymbols [▶ 613]
Assembly: TcHmiSrvExtNet.DynamicSymbols (in TcHmiSrvExtNet.DynamicSymbols.dll) Version: 1.10.683.0 (1.10.683.0)

Syntax

C#

```
public struct BuildInfo
```

The BuildInfo type exposes the following members.

Constructors

	Name	Description
	BuildInfo [▶ 616]	Initializes a new instace of the BuildInfo class.

Properties

	Name	Description
	BindingAttr [▶ 617]	A bitmask comprised of one or more 'BindingFlags' that specify how the search is conducted.
	IgnoreElements [▶ 617]	A collection of elements that should be ignored. To ignore elements in the write value use the prefix 'writeValue'.
	MaxDepth [▶ 617]	The maximum schema depth.
	Members [▶ 618]	A bitmask comprised of one or more IncludeMembers [▶ 640] that specify which members should be included.

Methods

	Name	Description
	Equals [▶ 619]	Determines whether the specified BuildInfo is equal to the current BuildInfo. (Overrides ValueType.Equals(Object).)
	GetHashCode [▶ 619]	Serves as a hash function for a particular type. (Overrides ValueType.GetHashCode..)
	GetType	Gets the Type of the current instance. (Inherited from Object.)
	ToString	Returns the fully qualified type name of this instance. (Inherited from ValueType.)

Operators

	Name	Description
	Equality [▶ 620]	Determines whether the specified BuildInfo is equal to the current BuildInfo.
	Inequality [▶ 620]	Determines whether the specified BuildInfo is inequal to the current BuildInfo.

Reference

TcHmiSrv.DynamicSymbols Namespace [▶ 613]

14.3.1.1.1 BuildInfo Constructor

Initializes a new instace of the BuildInfo [▶ 614] class.

Namespace: TcHmiSrv.DynamicSymbols [▶ 613]
Assembly: TcHmiSrvExtNet.DynamicSymbols (in TcHmiSrvExtNet.DynamicSymbols.dll) Version: 1.10.683.0 (1.10.683.0)

Syntax

C#

```
public BuildInfo(
    IncludeMembers members,
    BindingFlags bindingAttr,
    ICollection<string> ignoreElements,
    uint maxDepth
)
```

Parameters

members	Type: TcHmiSrv.DynamicSymbols.IncludeMembers [▶ 640] A bitmask comprised of one or more IncludeMembers [▶ 640] that specify which members should be included.
bindingAttr	Type: System.Reflection.BindingFlags A bitmask comprised of one or more 'BindingFlags' that specify how the search is conducted.
ignoreElements	Type: System.Collections.Generic.ICollection.String. A collection of elements that should be ignored. To ignore elements in the write value use the prefix 'writeValue'.
maxDepth	Type: System.UInt32 The maximum schema depth.

Reference

BuildInfo Structure [▶ 614]

TcHmiSrv.DynamicSymbols Namespace [▶ 613]

14.3.1.1.2 BuildInfo Properties

The BuildInfo [▶ 614] type exposes the following members.

Properties

	Name	Description
	BindingAttr [▶ 617]	A bitmask comprised of one or more 'BindingFlags' that specify how the search is conducted.
	IgnoreElements [▶ 617]	A collection of elements that should be ignored. To ignore elements in the write value use the prefix 'writeValue'.
	MaxDepth [▶ 617]	The maximum schema depth.
	Members [▶ 618]	A bitmask comprised of one or more IncludeMembers [▶ 640] that specify which members should be included.

Reference

BuildInfo Structure [▶ 614]

TcHmiSrv.DynamicSymbols Namespace [▶ 613]

15.1.1.2.1.2. getIsDestroyed
108

[Function]

```
public getIsDestroyed(): boolean;
```

This function returns a true value if the control was destroyed. This should be used within a callback of an asynchronous communication (XmlHttpRequest) to check if the control has been destroyed in the meantime.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	Value indicating whether the control was destroyed.

See also

Origin: TcHmiControl [▶ 433]

Note: Available since version 1.8

15.1.1.2.1.2. getAttributeDefaultValueInternal
109

[Function]

```
protected getAttributeDefaultValueInternal(propertyName: string): any;
```

Returns the default value of the attribute within a control. This is defined in Description.json.

Parameter

Name	Type	Description
propertyName	string [▶ 2108]	Name of the property

Return value

Type	Description
any [▶ 2107]	The default value from Description.json of the control.

See also

Origin: TcHmiControl [▶ 433]

Note: Available since version 1.10

15.1.1.2.1.2. __children
110

[Property]

```
protected __children: TcHmi.Controls.System.baseTcHmiControl[];
```

This property stores an array of all child controls of this control. This is the purely logical relationship and is managed by the system.

If a control is derived from the ContainerControl [▶ 1174], this information can be viewed publicly using thegetChildren [▶ 1183] function.

Definition

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118][]	List of all child controls. If there are no children, this is an empty array.

See also

Origin: TcHmiControl [▶ 433]

Note: Available since version 1.8

Sample - Javascript

```
console.log('We have ' + myControl.__children.length + ' child controls.');
var subControl = TcHmi.ControlFactory.create('tchmi-button', 'newbutton', myControl);
if(subControl !== undefined) {
    // The child control is not visible (not attached to the DOM), but it is our child
    console.log('Now we have one control more: ' + myControl.__children.length);
    // append control to ourself
    myControl.getElement().append(subControl.getElement());
}
```

15.1.1.2.1.3 TcHmiContainerControl

[Class]

No instance of this class can be created. This page serves only as a list of the attributes that this class provides.

Base class for all controls that can contain controls.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1180]	TcHmiContainerControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1181]	TcHmiContainerControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1179]	TcHmiContainerControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1180]	TcHmiContainerControl	Setter for the HeightMode attribute.

Name	Origin	Description
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

Container Control

See also

Reference [▶ 459] of this control

Note: Available since version 1.8

15.1.1.2.1.3. getHeightMode 1

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 464] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1180]

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. setHeightMode
2

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 464] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1179]

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. getWidthMode
3

[Function]

```
public getWidthMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the WidthMode [▶ 464] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setWidthMode [▶ 1181]

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. setWidthMode
4

[Function]

```
public setWidthMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the WidthMode [▶ 464] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWidthMode [▶ 1180]

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. getIsEnabled
5

[Function]

```
public getIsEnabled(): boolean;
```

This function returns the currently set value for the IsEnabled [▶ 464] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setIsEnabled [▶ 1182]

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. setIsEnabled
6

[Function]

```
public setIsEnabled(newValue: boolean | null): void;
```

This function sets a new value for the IsEnabled [▶ 464] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIsEnabled [▶ 1181]

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. addChild
7

[Function]

```
public addChild(co: TcHmi.Controls.System.baseTcHmiControl, pos: number | null): void;
```

Adds a child control to this control. This only affects the purely logical relationship. The child control is not added in the DOM and is therefore not automatically visible.

Parameter

Name	Type	Description
co	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Control to be added.
pos	number [▶ 2108], null [▶ 2108]	Sets the new position within the child list. If this is null or negative, the new child is added to the end.

Return value

Type	Description
void	No return value.

See also

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. removeChild 8

[Function]

```
public removeChild(co: TcHmi.Controls.System.baseTcHmiControl): void;
```

Removes a child control from the control. This only affects the purely logical relationship. The child control is not removed from the DOM and is therefore not automatically invisible.

Parameter

Name	Type	Description
co	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Control to be removed.

Return value

Type	Description
void	No return value.

See also

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.3. getChildren 9

[Function]

```
public getChildren(): TcHmi.Controls.System.baseTcHmiControl[];
```

Returns an array of all child controls of this control. This is the purely logical relationship.

Parameter

Name	Type	Description
-	- [▶ 1118]	-

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118][]	List of all child controls. If there are no children, an empty array is returned.

See also

Origin: TcHmiContainerControl [▶ 459]

Note: Available since version 1.8

15.1.1.2.1.4 TcHmiContainer

[Class]

Container for other controls.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1180]	TcHmiContainerControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1181]	TcHmiContainerControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1179]	TcHmiContainerControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1180]	TcHmiContainerControl	Setter for the HeightMode attribute.

Name	Origin	Description
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

Container Control [▶ 1174]

Container

See also

Reference [▶ 465] of this control

Note: Available since version 1.8

15.1.1.2.1.5 TcHmiContent

[Class]

No instance of this class can be created. This page serves only as a list of the attributes that this class provides.

A reusable container that can be loaded in a region.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1180]	TcHmiContainerControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1181]	TcHmiContainerControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1179]	TcHmiContainerControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1180]	TcHmiContainerControl	Setter for the HeightMode attribute.

Name	Origin	Description
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

Container Control [▶ 1174]

Partial [▶ 1213]

Content

See also

Reference [▶ 470] of this control

Note: Available since version 1.8

15.1.1.2.1.6 TcHmiGrid

[Class]

Layout grid for controls that can react fluidly and responsively.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1201]	TcHmiGrid	Getter for the WidthMode attribute.
setWidthMode [▶ 1201]	TcHmiGrid	Setter for the WidthMode attribute.
getHeightMode [▶ 1200]	TcHmiGrid	Getter for the HeightMode attribute.
setHeightMode [▶ 1200]	TcHmiGrid	Setter for the HeightMode attribute.

Name	Origin	Description
getRowOptions [▶ 1202]	TcHmiGrid	Getter for the RowOptions attribute.
setRowOptions [▶ 1203]	TcHmiGrid	Setter for the RowOptions attribute.
getColumnOptions [▶ 1201]	TcHmiGrid	Getter for the ColumnOptions attribute.
setColumnOptions [▶ 1202]	TcHmiGrid	Setter for the ColumnOptions attribute.
getCellOptions [▶ 1203]	TcHmiGrid	Getter for the CellOptions attribute.
setCellOptions [▶ 1204]	TcHmiGrid	Setter for the CellOptions attribute.
getCellWrap [▶ 1204]	TcHmiGrid	Getter for the CellWrap attribute.
setCellWrap [▶ 1205]	TcHmiGrid	Setter for the CellWrap attribute.
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.

Name	Origin	Description
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

  Container Control [▶ 1174]

    Grid

See also

Reference [▶ 475] of this control

Note: Available since version 1.8

15.1.1.2.1.6. getHeightMode 1

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 485] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1200]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. setHeightMode 2

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 485] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1200]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. getWidthMode 3

[Function]

```
public getWidthMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the WidthMode [▶ 486] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setWidthMode [▶ 1201]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. setWidthMode 4

[Function]

```
public setWidthMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the WidthMode [▶ 486] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWidthMode [▶ 1201]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. getColumnOptions 5

[Function]

```
public getColumnOptions(): TcHmi.Controls.System.TcHmiGrid.IColumnOptions[];
```

This function returns the currently set value for the ColumnOptions [▶ 485] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.System.TcHmiGrid.IColumnOptions [▶ 1206][]	The current value of the attribute.

See also

Attribute setter: setColumnOptions [▶ 1202]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. setColumnOptions

[Function]

```
public setColumnOptions(newValue: TcHmi.Controls.System.TcHmiGrid.IColumnOptions[] | null): void;
```

This function sets a new value for the ColumnOptions [▶ 485] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.System.TcHmiGrid.IColumnOptions [▶ 1206][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getColumnOptions [▶ 1201]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. getRowOptions

[Function]

```
public getRowOptions(): TcHmi.Controls.System.TcHmiGrid.IRowOptions[];
```

This function returns the currently set value for the RowOptions [▶ 485] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.System.TcHmiGrid.IRowOptions [▶ 1207][]	The current value of the attribute.

See also

Attribute setter: setRowOptions [▶ 1203]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. setRowOptions
8

[Function]

```
public setRowOptions(newValue: TcHmi.Controls.System.TcHmiGrid.IRowOptions[] | null): void;
```

This function sets a new value for the RowOptions [▶ 485] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.System.TcHmiGrid.IRowOptions [▶ 1207][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowOptions [▶ 1202]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. getCellOptions
9

[Function]

```
public getCellOptions(): TcHmi.Controls.System.TcHmiGrid.ICellOptions[];
```

This function returns the currently set value for the CellOptions [▶ 484] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.System.TcHmiGrid.ICellOptions [▶ 1205][]	The current value of the attribute.

See also

Attribute setter: setCellOptions [▶ 1204]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. setCellOptions 10

[Function]

```
public setCellOptions(newValue: TcHmi.Controls.System.TcHmiGrid.ICellOptions[] | null): void;
```

This function sets a new value for the CellOptions [▶ 484] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.System.TcHmiGrid.ICellOptions [▶ 1205][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getCellOptions [▶ 1203]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. getCellWrap 11

[Function]

```
public getCellWrap(): boolean;
```

This function returns the currently set value for the CellWrap [▶ 484] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setCellWrap [▶ 1205]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. setCellWrap
12

[Function]

```
public setCellWrap(newValue: boolean | null): void;
```

This function sets a new value for the CellWrap [▶ 484] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getCellWrap [▶ 1204]

Origin: TcHmiGrid [▶ 475]

Note: Available since version 1.8

15.1.1.2.1.6. ICellOptions
13

[Interface]

```
export interface ICellOptions {
    rowIndex: number;
    columnIndex: number;
    borderWidth?: TcHmi.FourSidedCss;
    borderStyle?: TcHmi.BorderStyle;
    borderColor?: TcHmi.SolidColor;
    padding?: TcHmi.FourSidedCss;
    backgroundColor?: TcHmi.Color;
    backgroundImage?: string;
    backgroundImagePadding?: TcHmi.FourSidedCss;
    backgroundImageWidth?: number;
    backgroundImageWidthUnit?: TcHmi.DimensionUnit;
    backgroundImageHeight?: number;
    backgroundImageHeightUnit?: TcHmi.DimensionUnit;
    backgroundImageHorizontalAlignment?: TcHmi.HorizontalAlignment;
    backgroundImageVerticalAlignment?: TcHmi.VerticalAlignment;
}
```

Properties

Name	Type	Description
rowIndex	number [▶ 2108]	Index of the row for this definition
columnIndex	number [▶ 2108]	Index of the column for this definition
borderWidth [optional]	TcHmi.FourSidedCss [▶ 2088]	Border width
borderStyle [optional]	TcHmi.BorderStyle [▶ 2081]	Border style
borderColor [optional]	TcHmi.SolidColor [▶ 2084]	Border color
padding [optional]	TcHmi.FourSidedCss [▶ 2088]	Border inside distance
backgroundColor [optional]	TcHmi.Color [▶ 2084]	Background color
backgroundImage [optional]	string [▶ 2108]	Background image
backgroundImagePadding [optional]	TcHmi.FourSidedCss [▶ 2088]	Inner distance of the background image
backgroundImageWidth [optional]	number [▶ 2108]	Numerical value of the width of the background image
backgroundImageWidthUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the width of the background image
backgroundImageHeight [optional]	number [▶ 2108]	Numerical value of the height of the background image
backgroundImageHeightUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the height of the background image
backgroundImageHorizontalAlignment [optional]	TcHmi.HorizontalAlignment [▶ 2089]	Horizontal alignment of the background image
backgroundImageVerticalAlignment [optional]	TcHmi.VerticalAlignment [▶ 2089]	Vertical alignment of the background image

See also

Scheme: tchmi:framework#/definitions/CellOptions [▶ 2199]

15.1.1.2.1.6. IColumnOptions 14

[Interface]

```
export interface IColumnOptions {
    width: number | null;
    widthUnit: TcHmi.DimensionUnit | 'factor';
    widthMode: 'Value' | 'Content';
    minWidth: number | null;
    minWidthUnit: TcHmi.DimensionUnit;
    maxWidth: number | null;
    maxWidthUnit: TcHmi.DimensionUnit;
    overflow: boolean;
}
```

Properties

Name	Type	Description
width	number [▶ 2108], null [▶ 2108]	Numerical value of the width of this column
widthUnit	TcHmi.DimensionUnit [▶ 2086] | 'factor'	Unit of the width of this column. This can also be 'factor' at this point
widthMode	'Value' | 'Content'	Only value and content is allowed here.
minWidth	number [▶ 2108], null [▶ 2108]	Numerical value of the minimum width of this column
minWidthUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the minimum width of this column.
maxnWidth	number [▶ 2108], null [▶ 2108]	Numerical value of the maximum width of this column
maxWidthUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the maximum width of this column.
overflow	boolean [▶ 2107]	Determines whether controls should be truncated if they are outside the cell.

See also

Scheme: tchmi:framework#/definitions/ColumnOptions [▶ 2214]

15.1.1.2.1.6. IRowOptions
15

[Interface]

```
export interface IRowOptions {
    height: number | null;
    heightUnit: TcHmi.DimensionUnit | 'factor';
    heightMode: 'Value' | 'Content';
    minHeight: number | null;
    minHeightUnit: TcHmi.DimensionUnit;
    maxHeight: number | null;
    maxHeightUnit: TcHmi.DimensionUnit;
    overflow: boolean;
}
```

Properties

Name	Type	Description
height	number [▶ 2108], null [▶ 2108]	Numerical value of the height of this column
heightUnit	TcHmi.DimensionUnit [▶ 2086] | 'factor'	Unit of the height of this column. This can also be 'factor' at this point
heightMode	'Value' | 'Content'	Only value and content is allowed here.
minHeight	number [▶ 2108], null [▶ 2108]	Numerical value of the minimum height of this column
minHeightUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the minimum height of this column.
maxHeight	number [▶ 2108], null [▶ 2108]	Numerical value of the maximum height of this column
maxHeightUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the maximum height of this column.
overflow	boolean [▶ 2107]	Determines whether controls should be truncated if they are outside the cell.

See also

Scheme: tchmi:framework#/definitions/RowOptions [▶ 2249]

15.1.1.2.1.7 TcHmiHtmlHost

[Class]

Container for HTML content. Can be edited in the HTML editor of the Designer.

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.

Name	Origin	Description
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

HTML Host

See also

Reference [▶ 486] of this control

Note: Available since version 1.8

15.1.1.2.1.8 TcHmiPartial

[Class]

No instance of this class can be created. This page serves only as a list of the attributes that this class provides.

A base class for reusable control containers.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1180]	TcHmiContainerControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1181]	TcHmiContainerControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1179]	TcHmiContainerControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1180]	TcHmiContainerControl	Setter for the HeightMode attribute.

Name	Origin	Description
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

Container Control [▶ 1174]

Partial

See also

Reference [▶ 490] of this control

Note: Available since version 1.8

15.1.1.2.1.9 TcHmiRegion

[Class]

Container for content files (these are added to the project via "Add new file") .

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getHeightMode [▶ 1223]	TcHmiRegion	Getter for the HeightMode attribute.
setHeightMode [▶ 1224]	TcHmiRegion	Setter for the HeightMode attribute.
getWidthMode [▶ 1224]	TcHmiRegion	Getter for the WidthMode attribute.
setWidthMode [▶ 1225]	TcHmiRegion	Setter for the WidthMode attribute.

Name	Origin	Description
getTargetContent [▶ 1225]	TcHmiRegion	Getter for the TargetContent attribute.
setTargetContent [▶ 1226]	TcHmiRegion	Setter for the TargetContent attribute.
getScaleMode [▶ 1226]	TcHmiRegion	Getter for the ScaleMode attribute.
setScaleMode [▶ 1227]	TcHmiRegion	Setter for the ScaleMode attribute.
getScrolling [▶ 1227]	TcHmiRegion	Getter for the Scrolling attribute.
setScrolling [▶ 1228]	TcHmiRegion	Setter for the Scrolling attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.

Name	Origin	Description
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Region

See also

Reference [▶ 496] of this control

Note: Available since version 1.8

15.1.1.2.1.9. getHeightMode

1

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 502] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1224]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. setHeightMode
2

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 502] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1223]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. getWidthMode
3

[Function]

```
public getWidthMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the WidthMode [▶ 502] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setWidthMode [▶ 1225]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. setWidthMode
4

[Function]

```
public setWidthMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the WidthMode [▶ 502] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWidthMode [▶ 1224]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. getTargetContent
5

[Function]

```
public getTargetContent(): string | null;
```

This function returns the currently set value for the TargetContent [▶ 502] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTargetContent [▶ 1226]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. setTargetContent
6

[Function]

```
public setTargetContent(newValue: string | null): void;
```

This function sets a new value for the TargetContent [▶ 502] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTargetContent [▶ 1225]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. getScaleMode
7

[Function]

```
public getScaleMode(): TcHmi.ScaleModeString;
```

This function returns the currently set value for the ScaleMode [▶ 503] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.ScaleModeString [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setScaleMode [▶ 1227]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. setScaleMode
8

[Function]

```
public setScaleMode(newValue: TcHmi.ScaleModeString | null): void;
```

This function sets a new value for the ScaleMode [▶ 503] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.ScaleModeString [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getScaleMode [▶ 1226]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. getScrolling
9

[Function]

```
public getScrolling(): 'No' | 'Yes' | 'Auto';
```

This function returns the currently set value for the Scrolling [▶ 503] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'No' | 'Yes' | 'Auto'	The current value of the attribute.

See also

Attribute setter: setScrolling [▶ 1228]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.9. setScrolling
10

[Function]

```
public setScrolling(newValue: 'No' | 'Yes' | 'Auto' | null): void;
```

This function sets a new value for the Scrolling [▶ 503] attribute.

Parameter

Name	Type	Description
newValue	'No' | 'Yes' | 'Auto', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getScrolling [▶ 1227]

Origin: TcHmiRegion [▶ 496]

Note: Available since version 1.8

15.1.1.2.1.1 TcHmiUserControl
0

[Class]

No instance of this class can be created. This page serves only as a list of the attributes that this class provides.

A reusable collection of controls.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1180]	TcHmiContainerControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1181]	TcHmiContainerControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1179]	TcHmiContainerControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1180]	TcHmiContainerControl	Setter for the HeightMode attribute.

Name	Origin	Description
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

Container Control [▶ 1174]

Partial [▶ 1213]

User Control

See also

Reference [▶ 504] of this control

Note: Available since version 1.8

15.1.1.2.1.1 TcHmiUserControlHost
1

[Class]

No instance of this class can be created. This page serves only as a list of the attributes that this class provides.

Container for user controls.

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1240]	TcHmiUserControlHost	Getter for the WidthMode attribute.
setWidthMode [▶ 1240]	TcHmiUserControlHost	Setter for the WidthMode attribute.
getIsEnabled [▶ 1242]	TcHmiUserControlHost	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1242]	TcHmiUserControlHost	Setter for the IsEnabled attribute.

Name	Origin	Description
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.

getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

User Control Host

See also

Reference [▶ 509] of this control

Note: Available since version 1.8

15.1.1.2.1.1 getHeightMode 1.1

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 514] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1239]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 setHeightMode 1.2

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 514] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1239]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 getWidthMode
1.3

[Function]

```
public getWidthMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the WidthMode [▶ 514] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setWidthMode [▶ 1240]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 setWidthMode
1.4

[Function]

```
public setWidthMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the WidthMode [▶ 514] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWidthMode [▶ 1240]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 getTargetUserControl 1.5

[Function]

```
public getTargetUserControl(): string | null;
```

This function returns the currently set value for the TargetUserControl [▶ 515] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTargetUserControl [▶ 1241]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 setTargetUserControl 1.6

[Function]

```
public setTargetUserControl(newValue: string | null): void;
```

This function sets a new value for the TargetUserControl [▶ 515] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTargetUserControl [▶ 1241]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available from version 1.8, *not applicable from version 1.10*

15.1.1.2.1.1 getIsEnabled
1.7

[Function]

```
public getIsEnabled(): boolean;
```

This function returns the currently set value for the IsEnabled [▶ 515] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setIsEnabled [▶ 1242]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 setIsEnabled
1.8

[Function]

```
public setIsEnabled(newValue: boolean | null): void;
```

This function sets a new value for the IsEnabled [▶ 515] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIsEnabled [▶ 1242]

Origin: TcHmiUserControlHost [▶ 509]

Note: Available since version 1.8

15.1.1.2.1.1 TcHmiView
2

[Class]

No instance of this class can be created. This page serves only as a list of the attributes that this class provides.

Top-level control for a TcHmi visualization.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.

Name	Origin	Description
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getWidthMode [▶ 1180]	TcHmiContainerControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1181]	TcHmiContainerControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1179]	TcHmiContainerControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1180]	TcHmiContainerControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1181]	TcHmiContainerControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1182]	TcHmiContainerControl	Setter for the IsEnabled attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.
addChild [▶ 1182]	TcHmiContainerControl	Adds a child control to this control.
removeChild [▶ 1183]	TcHmiContainerControl	Removes a child control from the control.
getChildren [▶ 1183]	TcHmiContainerControl	Returns an array of all child controls of this control.

Inheritance hierarchy

Control [▶ 1118]

Container Control [▶ 1174]

Partial [▶ 1213]

View

See also

Reference [▶ 515] of this control

Note: Available since version 1.8

15.1.1.2.2 Beckhoff

[Namespace]

Classes

Name	Description
TcHmiBarChart [▶ 1249]	Bar chart that can be configured with various elements.
TcHmiButton [▶ 1309]	Simple button.
TcHmiCheckbox [▶ 1334]	Simple checkbox. A checkbox has a ToggleState parameter. This parameter has the value "Normal" if no checkmark is set and "Active" if the checkmark is set. The .onToggleStateChanged event can be used to perform actions when the state of the checkbox changes.
TcHmiCombobox [▶ 1343]	A combobox is a drop-down menu for selecting one of several options. Each option is defined by its ID and also has a text that is displayed in the drop-down menu, and a value.
TcHmiDatagrid [▶ 1382]	Table for generic display of data.
TcHmiEllipse [▶ 1429]	Ellipse
TcHmiEventGrid [▶ 1439]	Table for displaying events.
TcHmiIFrame [▶ 1449]	Nested browser context. Embeds another HTML page in the current visualization.
TcHmiImage [▶ 1455]	Image display
TcHmiKeyboard [▶ 1465]	On-screen keyboard
TcHmiLine [▶ 1492]	Straight line
TcHmiLinearGauge [▶ 1507]	Linear display
TcHmiLineChart [▶ 1557]	Line chart
TcHmiPolygon [▶ 1618]	Closed shape formed from a set of interconnected straight lines. The last point is automatically connected to the first point.
TcHmiRadialGauge [▶ 1632]	Radial display
TcHmiRectangle [▶ 1682]	Rectangle
TcHmiTextblock [▶ 1694]	Text block for displaying text
TcHmiTextbox [▶ 1711]	Text box for entering text.
TcHmiToggleButton [▶ 1730]	Button that can be toggled between two states.
TcHmiTrendLineChart [▶ 1739]	Line chart for historicized data
TcHmiVideo [▶ 1812]	Display area for a video.

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiBarChart

[Class]

Bar chart that can be configured with various elements.

Attribute functions

Name	Origin	Description
getGridBackgroundColor [▶ 1258]	TcHmiBarChart	Getter for the GridBackgroundColor attribute.
setGridBackgroundColor [▶ 1259]	TcHmiBarChart	Setter for the GridBackgroundColor attribute.
getGridLineColor [▶ 1261]	TcHmiBarChart	Getter for the GridLineColor attribute.
setGridLineColor [▶ 1261]	TcHmiBarChart	Setter for the GridLineColor attribute.
getGridLineWidth [▶ 1281]	TcHmiBarChart	Getter for the GridLineWidth attribute.
setGridLineWidth [▶ 1282]	TcHmiBarChart	Setter for the GridLineWidth attribute.
getGridShowHorizontalLines [▶ 1284]	TcHmiBarChart	Getter for the GridShowHorizontalLines attribute.
setGridShowHorizontalLines [▶ 1284]	TcHmiBarChart	Setter for the GridShowHorizontalLines attribute.
getGridShowVerticalLines [▶ 1278]	TcHmiBarChart	Getter for the GridShowVerticalLines attribute.
setGridShowVerticalLines [▶ 1279]	TcHmiBarChart	Setter for the GridShowVerticalLines attribute.
getYAxisColor [▶ 1259]	TcHmiBarChart	Getter for the YAxisColor attribute.
setYAxisColor [▶ 1259]	TcHmiBarChart	Setter for the YAxisColor attribute.
getYAxisAutoScaling [▶ 1296]	TcHmiBarChart	Getter for the YAxisAutoScaling attribute.
setYAxisAutoScaling [▶ 1296]	TcHmiBarChart	Setter for the YAxisAutoScaling attribute.
getYAxisDecimalPlaces [▶ 1294]	TcHmiBarChart	Getter for the YAxisDecimalPlaces attribute.
setYAxisDecimalPlaces [▶ 1295]	TcHmiBarChart	Setter for the YAxisDecimalPlaces attribute.
getYAxisWidth [▶ 1299]	TcHmiBarChart	Getter for the YAxisWidth attribute.
setYAxisWidth [▶ 1299]	TcHmiBarChart	Setter for the YAxisWidth attribute.
getYAxisUnit [▶ 1293]	TcHmiBarChart	Getter for the YAxisUnit attribute.
setYAxisUnit [▶ 1294]	TcHmiBarChart	Setter for the YAxisUnit attribute.
getYAxisName [▶ 1291]	TcHmiBarChart	Getter for the YAxisName attribute.
setYAxisName [▶ 1292]	TcHmiBarChart	Setter for the YAxisName attribute.
getYAxisNameFontColor [▶ 1265]	TcHmiBarChart	Getter for the YAxisNameFontColor attribute.
setYAxisNameFontColor [▶ 1265]	TcHmiBarChart	Setter for the YAxisNameFontColor attribute.
getYAxisNameFontFamily [▶ 1305]	TcHmiBarChart	Getter for the YAxisNameFontFamily attribute.
setYAxisNameFontFamily [▶ 1306]	TcHmiBarChart	Setter for the YAxisNameFontFamily attribute.
getYAxisNameFontSize [▶ 1303]	TcHmiBarChart	Getter for the YAxisNameFontSize attribute.
setYAxisNameFontSize [▶ 1304]	TcHmiBarChart	Setter for the YAxisNameFontSize attribute.
getYAxisNameFontSizeUnit [▶ 1300]	TcHmiBarChart	Getter for the YAxisNameFontSizeUnit attribute.
setYAxisNameFontSizeUnit [▶ 1300]	TcHmiBarChart	Setter for the YAxisNameFontSizeUnit attribute.

Name	Origin	Description
getYAxisNameFontWeight [▶ 1301]	TcHmiBarChart	Getter for the YAxisNameFontWeight attribute.
setYAxisNameFontWeight [▶ 1301]	TcHmiBarChart	Setter for the YAxisNameFontWeight attribute.
getYLabelFontColor [▶ 1264]	TcHmiBarChart	Getter for the YLabelFontColor attribute.
setYLabelFontColor [▶ 1264]	TcHmiBarChart	Setter for the YLabelFontColor attribute.
getYLabelFontFamily [▶ 1304]	TcHmiBarChart	Getter for the YLabelFontFamily attribute.
setYLabelFontFamily [▶ 1305]	TcHmiBarChart	Setter for the YLabelFontFamily attribute.
getYLabelFontSize [▶ 1298]	TcHmiBarChart	Getter for the YLabelFontSize attribute.
setYLabelFontSize [▶ 1298]	TcHmiBarChart	Setter for the YLabelFontSize attribute.
getYLabelFontSizeUnit [▶ 1302]	TcHmiBarChart	Getter for the YLabelFontSizeUnit attribute.
setYLabelFontSizeUnit [▶ 1303]	TcHmiBarChart	Setter for the YLabelFontSizeUnit attribute.
getYLabelFontWeight [▶ 1301]	TcHmiBarChart	Getter for the YLabelFontWeight attribute.
setYLabelFontWeight [▶ 1302]	TcHmiBarChart	Setter for the YLabelFontWeight attribute.
getYMainTickMaxValue [▶ 1295]	TcHmiBarChart	Getter for the YMainTickMaxValue attribute.
setYMainTickMaxValue [▶ 1295]	TcHmiBarChart	Setter for the YMainTickMaxValue attribute.
getYMainTickMinValue [▶ 1292]	TcHmiBarChart	Getter for the YMainTickMinValue attribute.
setYMainTickMinValue [▶ 1293]	TcHmiBarChart	Setter for the YMainTickMinValue attribute.
getYMainTickSteps [▶ 1297]	TcHmiBarChart	Getter for the YMainTickSteps attribute.
setYMainTickSteps [▶ 1297]	TcHmiBarChart	Setter for the YMainTickSteps attribute.
getYPosition [▶ 1287]	TcHmiBarChart	Getter for the YPosition attribute.
setYPosition [▶ 1287]	TcHmiBarChart	Setter for the YPosition attribute.
getYShowLabels [▶ 1289]	TcHmiBarChart	Getter for the YShowLabels attribute.
setYShowLabels [▶ 1289]	TcHmiBarChart	Setter for the YShowLabels attribute.
getYSubTickSteps [▶ 1289]	TcHmiBarChart	Getter for the YSubTickSteps attribute.
setYSubTickSteps [▶ 1290]	TcHmiBarChart	Setter for the YSubTickSteps attribute.
getXAxisColor [▶ 1260]	TcHmiBarChart	Getter for the XAxisColor attribute.
setXAxisColor [▶ 1260]	TcHmiBarChart	Setter for the XAxisColor attribute.
getXShowLabels [▶ 1269]	TcHmiBarChart	Getter for the XShowLabels attribute.
setXShowLabels [▶ 1270]	TcHmiBarChart	Setter for the XShowLabels attribute.

Name	Origin	Description
getXAxisAutoScaling [▶ 1272]	TcHmiBarChart	Getter for the XAxisAutoScaling attribute.
setXAxisAutoScaling [▶ 1272]	TcHmiBarChart	Setter for the XAxisAutoScaling attribute.
getXAxisDecimalPlaces [▶ 1273]	TcHmiBarChart	Getter for the XAxisDecimalPlaces attribute.
setXAxisDecimalPlaces [▶ 1273]	TcHmiBarChart	Setter for the XAxisDecimalPlaces attribute.
getXMainTickMinValue [▶ 1275]	TcHmiBarChart	Getter for the XMainTickMinValue attribute.
setXMainTickMinValue [▶ 1275]	TcHmiBarChart	Setter for the XMainTickMinValue attribute.
getXMainTickMaxValue [▶ 1271]	TcHmiBarChart	Getter for the XMainTickMaxValue attribute.
setXMainTickMaxValue [▶ 1271]	TcHmiBarChart	Setter for the XMainTickMaxValue attribute.
getXMainTickSteps [▶ 1274]	TcHmiBarChart	Getter for the XMainTickSteps attribute.
setXMainTickSteps [▶ 1274]	TcHmiBarChart	Setter for the XMainTickSteps attribute.
getXSubTickSteps [▶ 1270]	TcHmiBarChart	Getter for the XSubTickSteps attribute.
setXSubTickSteps [▶ 1271]	TcHmiBarChart	Setter for the XSubTickSteps attribute.
getXAxisWidth [▶ 1277]	TcHmiBarChart	Getter for the XAxisWidth attribute.
setXAxisWidth [▶ 1278]	TcHmiBarChart	Setter for the XAxisWidth attribute.
getXLabelFontColor [▶ 1263]	TcHmiBarChart	Getter for the XLabelFontColor attribute.
setXLabelFontColor [▶ 1263]	TcHmiBarChart	Setter for the XLabelFontColor attribute.
getXLabelFontFamily [▶ 1279]	TcHmiBarChart	Getter for the XLabelFontFamily attribute.
setXLabelFontFamily [▶ 1280]	TcHmiBarChart	Setter for the XLabelFontFamily attribute.
getXLabelFontSize [▶ 1280]	TcHmiBarChart	Getter for the XLabelFontSize attribute.
setXLabelFontSize [▶ 1281]	TcHmiBarChart	Setter for the XLabelFontSize attribute.
getXLabelFontSizeUnit [▶ 1277]	TcHmiBarChart	Getter for the XLabelFontSizeUnit attribute.
setXLabelFontSizeUnit [▶ 1277]	TcHmiBarChart	Setter for the XLabelFontSizeUnit attribute.
getXLabelFontWeight [▶ 1282]	TcHmiBarChart	Getter for the XLabelFontWeight attribute.
setXLabelFontWeight [▶ 1283]	TcHmiBarChart	Setter for the XLabelFontWeight attribute.
getXAxisFormat [▶ 1283]	TcHmiBarChart	Getter for the XAxisFormat attribute
setXAxisFormat [▶ 1283]	TcHmiBarChart	Setter for the XAxisFormat attribute
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.

Name	Origin	Description
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getSubgridLineColor [▶ 1262]	TcHmiBarChart	Getter for the SubgridLineColor attribute.
setSubgridLineColor [▶ 1262]	TcHmiBarChart	Setter for the SubgridLineColor attribute.
getSubgridLineWidth [▶ 1308]	TcHmiBarChart	Getter for the SubgridLineWidth attribute.
setSubgridLineWidth [▶ 1308]	TcHmiBarChart	Setter for the SubgridLineWidth attribute.
getSubgridShowHorizontalLines [▶ 1307]	TcHmiBarChart	Getter for the SubgridShowHorizontalLines attribute.
setSubgridShowHorizontalLines [▶ 1307]	TcHmiBarChart	Setter for the SubgridShowHorizontalLines attribute.
getSubgridShowVerticalLines [▶ 1306]	TcHmiBarChart	Getter for the SubgridShowVerticalLines attribute.
setSubgridShowVerticalLines [▶ 1307]	TcHmiBarChart	Setter for the SubgridShowVerticalLines attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.

Name	Origin	Description
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.

Name	Origin	Description
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getBarGraphData [▶ 1266]	TcHmiBarChart	Getter for the BarGraphData attribute.
setBarGraphData [▶ 1267]	TcHmiBarChart	Setter for the BarGraphData attribute.
getBarGraphColors [▶ 1265]	TcHmiBarChart	Getter for the BarGraphColors attribute.
setBarGraphColors [▶ 1266]	TcHmiBarChart	Setter for the BarGraphColors attribute.
getBarGraphWidth [▶ 1267]	TcHmiBarChart	Getter for the BarGraphWidth attribute
setBarGraphWidth [▶ 1268]	TcHmiBarChart	Setter for the BarGraphWidth attribute
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.

Name	Origin	Description
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getShowGrid [▶ 1285]	TcHmiBarChart	Getter for the ShowGrid attribute.
setShowGrid [▶ 1285]	TcHmiBarChart	Setter for the ShowGrid attribute.
getShowXAxis [▶ 1268]	TcHmiBarChart	Getter for the ShowXAxis attribute.
setShowXAxis [▶ 1269]	TcHmiBarChart	Setter for the ShowXAxis attribute.
getShowXSubTicks [▶ 1276]	TcHmiBarChart	Getter for the ShowXSubTicks attribute.
setShowXSubTicks [▶ 1276]	TcHmiBarChart	Setter for the ShowXSubTicks attribute.
getShowYAxis [▶ 1286]	TcHmiBarChart	Getter for the ShowYAxis attribute.
setShowYAxis [▶ 1286]	TcHmiBarChart	Setter for the ShowYAxis attribute.
getShowYAxisName [▶ 1290]	TcHmiBarChart	Getter for the ShowYAxisName attribute.
setShowYAxisName [▶ 1291]	TcHmiBarChart	Setter for the ShowYAxisName attribute.
getShowYSubTicks [▶ 1288]	TcHmiBarChart	Getter for the ShowYSubTicks attribute.

Name	Origin	Description
setShowYSubTicks [▶ 1288]	TcHmiBarChart	Setter for the ShowYSubTicks attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Bar Chart

See also

Reference [▶ 142] of this control

Note: Available since version 1.8

15.1.1.2.2.1. getGridBackgroundColor

[Function]

```
public getGridBackgroundColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the GridBackgroundColor [▶ 153] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridBackgroundColor [▶ 1259]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setGridBackgroundColor
2

[Function]

```
public setGridBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the GridBackgroundColor [▶ 153] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridBackgroundColor [▶ 1258]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisColor
3

[Function]

```
public getYAxisColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the YAxisColor [▶ 162] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisColor [▶ 1259]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisColor
4

[Function]

```
public setYAxisColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the YAxisColor [▶ 162] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisColor [▶ 1259]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXAxisColor
5

[Function]

```
public getXAxisColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XAxisColor [▶ 157] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisColor [▶ 1260]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXAxisColor
6

[Function]

```
public setXAxisColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XAxisColor [▶ 157] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisColor [▶ 1260]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getGridLineColor
7

[Function]

```
public getGridLineColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the GridLineColor [▶ 153] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridLineColor [▶ 1261]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setGridLineColor
8

[Function]

```
public setGridLineColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the GridLineColor [▶ 153] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridLineColor [▶ 1261]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getSubgridLineColor
9

[Function]

```
public getSubgridLineColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the SubgridLineColor [▶ 157] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSubgridLineColor [▶ 1262]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setSubgridLineColor
10

[Function]

```
public setSubgridLineColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the SubgridLineColor [▶ 157] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridLineColor [▶ 1262]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXLabelFontColor
11

[Function]

```
public getXLabelFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XLabelFontColor [▶ 158] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontColor [▶ 1263]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXLabelFontColor
12

[Function]

```
public setXLabelFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XLabelFontColor [▶ 158] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontColor [▶ 1263]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYLabelFontColor 13

[Function]

```
public getYLabelFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the YLabelFontColor [▶ 162] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontColor [▶ 1264]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYLabelFontColor 14

[Function]

```
public setYLabelFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the YLabelFontColor [▶ 162] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontColor [▶ 1264]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisNameFontColor
15

[Function]

```
public getYAxisNameFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the YAxisNameFontColor [▶ 164] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisNameFontColor [▶ 1265]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisNameFontColor
16

[Function]

```
public setYAxisNameFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the YAxisNameFontColor [▶ 164] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisNameFontColor [▶ 1265]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getBarGraphColors
17

[Function]

```
public getBarGraphColors(): TcHmi.SolidColor[] | TcHmi.SolidColor | null;
```

This function returns the currently set value for the BarGraphColors [▶ 152] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084][], TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setBarGraphColors [▶ 1266]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setBarGraphColors 18

[Function]

```
public setBarGraphColors(newValue: TcHmi.SolidColor[] | TcHmi.SolidColor | null): void;
```

This function sets a new value for the BarGraphColors [▶ 152] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084][] | TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getBarGraphColors [▶ 1265]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getBarGraphData 19

[Function]

```
public getBarGraphData(): number[][] | number[] | null;
```

This function returns the currently set value for the BarGraphData [▶ 153] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108][][] | number [▶ 2108][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setBarGraphData [▶ 1267]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setBarGraphData 20

[Function]

```
public setBarGraphData(newValue: number[][] | number[] | null): void;
```

This function sets a new value for the BarGraphData [▶ 153] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108][][], number [▶ 2108][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getBarGraphData [▶ 1266]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getBarGraphWidth 21

[Function]

```
public get BarGraphWidth(): number | null;
```

This function returns the currently set value for the BarGraphWidth [▶ 153] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: set BarGraphWidth [▶ 1268]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.10

15.1.1.2.2.1. setBarGraphWidth 22

[Function]

```
public setBarGraphWidth(newValue: number | null): void;
```

This function sets a new value for the BarGraphWidth [▶ 153] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getBarGraphWidth [▶ 1267]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.10

15.1.1.2.2.1. getShowXAxis 23

[Function]

```
public getShowXAxis(): boolean;
```

This function returns the currently set value for the ShowXAxis [▶ 155] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXAxis [▶ 1269]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setShowXAxis
24

[Function]

```
public setShowXAxis(newValue: boolean | null): void;
```

This function sets a new value for the ShowXAxis [▶ 155] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXAxis [▶ 1268]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXShowLabels
25

[Function]

```
public getXShowLabels(): boolean;
```

This function returns the currently set value for the XShowLabels [▶ 155] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setXShowLabels [▶ 1270]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXShowLabels
26

[Function]

```
public setXShowLabels(newValue: boolean | null): void;
```

This function sets a new value for the XShowLabels [▶ 155] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXShowLabels [▶ 1269]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXSubTickSteps
27

[Function]

```
public getXSubTickSteps(): number | null;
```

This function returns the currently set value for the XSubTickSteps [▶ 160] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXSubTickSteps [▶ 1271]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXSubTickSteps
28

[Function]

```
public setXSubTickSteps(newValue: number | null): void;
```

This function sets a new value for the XSubTickSteps [▶ 160] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXSubTickSteps [▶ 1270]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXMainTickMaxValue
29

[Function]

```
public getXMainTickMaxValue(): number | null;
```

This function returns the currently set value for the XMainTickMaxValue [▶ 159] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickMaxValue [▶ 1271]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXMainTickMaxValue
30

[Function]

```
public setXMainTickMaxValue(newValue: number | null): void;
```

This function sets a new value for the XMainTickMaxValue [▶ 159] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickMaxValue [▶ 1271]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXAxisAutoScaling 31

[Function]

```
public getXAxisAutoScaling(): boolean;
```

This function returns the currently set value for the XAxisAutoScaling [▶ 157] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setXAxisAutoScaling [▶ 1272]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXAxisAutoScaling 32

[Function]

```
public setXAxisAutoScaling(newValue: boolean | null): void;
```

This function sets a new value for the XAxisAutoScaling [▶ 157] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisAutoScaling [▶ 1272]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXAxisDecimalPlaces
33

[Function]

```
public getXAxisDecimalPlaces(): number;
```

This function returns the currently set value for the XAxisDecimalPlaces [▶ 158] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisDecimalPlaces [▶ 1273]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXAxisDecimalPlaces
34

[Function]

```
public setXAxisDecimalPlaces(newValue: number | null): void;
```

This function sets a new value for the XAxisDecimalPlaces [▶ 158] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisDecimalPlaces [▶ 1273]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXMainTickSteps
35

[Function]

```
public getXMainTickSteps(): number;
```

This function returns the currently set value for the XMainTickSteps [▶ 160] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickSteps [▶ 1274]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXMainTickSteps
36

[Function]

```
public setXMainTickSteps(newValue: number | null): void;
```

This function sets a new value for the XMainTickSteps [▶ 160] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickSteps [▶ 1274]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXMainTickMinValue
37

[Function]

```
public getXMainTickMinValue(): number | null;
```

This function returns the currently set value for the XMainTickMinValue [▶ 159] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickMinValue [▶ 1275]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXMainTickMinValue
38

[Function]

```
public setXMainTickMinValue(newValue: number | null): void;
```

This function sets a new value for the XMainTickMinValue [▶ 159] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickMinValue [▶ 1275]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getShowXSubTicks 39

[Function]

```
public getShowXSubTicks(): boolean;
```

This function returns the currently set value for the ShowXSubTicks [▶ 155] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXSubTicks [▶ 1276]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setShowXSubTicks 40

[Function]

```
public setShowXSubTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowXSubTicks [▶ 155] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXSubTicks [▶ 1276]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXLabelFontSizeUnit
41

[Function]

```
public getXLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the XLabelFontSizeUnit [▶ 159] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setXLabelFontSizeUnit [▶ 1277]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXLabelFontSizeUnit
42

[Function]

```
public setXLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the XLabelFontSizeUnit [▶ 159] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontSizeUnit [▶ 1277]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXAxisWidth
43

[Function]

```
public getXAxisWidth(): number;
```

This function returns the currently set value for the XAxisWidth [▶ 160] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisWidth [▶ 1278]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXAxisWidth
44

[Function]

```
public setXAxisWidth(newValue: number | null): void;
```

This function sets a new value for the XAxisWidth [▶ 160] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisWidth [▶ 1277]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getGridShowVerticalLines
45

[Function]

```
public getGridShowVerticalLines(): boolean;
```

This function returns the currently set value for the GridShowVerticalLines [▶ 154] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setGridShowVerticalLines [▶ 1279]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setGridShowVerticalLines
46

[Function]

```
public setGridShowVerticalLines(newValue: boolean | null): void;
```

This function sets a new value for the GridShowVerticalLines [▶ 154] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridShowVerticalLines [▶ 1278]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXLabelFontFamily
47

[Function]

```
public getXLabelFontFamily(): string;
```

This function returns the currently set value for the XLabelFontFamily [▶ 158] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontFamily [▶ 1280]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXLabelFontFamily 48

[Function]

```
public setXLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the XLabelFontFamily [▶ 158] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontFamily [▶ 1279]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXLabelFontSize 49

[Function]

```
public getXLabelFontSize(): number;
```

This function returns the currently set value for the XLabelFontSize [▶ 158] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontSize [▶ 1281]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXLabelFontSize
50

[Function]

```
public setXLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the XLabelFontSize [▶ 158] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontSize [▶ 1280]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getGridLineWidth
51

[Function]

```
public getGridLineWidth(): number;
```

This function returns the currently set value for the GridLineWidth [▶ 154] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridLineWidth [▶ 1282]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setGridLineWidth
52

[Function]

```
public setGridLineWidth(newValue: number | null): void;
```

This function sets a new value for the GridLineWidth [▶ 154] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridLineWidth [▶ 1281]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXLabelFontWeight
53

[Function]

```
public getXLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the XLabelFontFontWeight [▶ 159] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setXLabelFontWeight [▶ 1283]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setXLabelFontWeight
54

[Function]

```
public setXLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the XLabelFontWeight [▶ 159] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontWeight [▶ 1282]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getXAxisFormat
55

[Function]

```
public getXAxisFormat(): TcHmi.IFunction;
```

This function returns the currently set value for the XAxisFormat [▶ 160] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095]	The current value of the attribute.

See also

Attribute setter: setXAxisFormat [▶ 1283]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.10

15.1.1.2.2.1. setXAxisFormat
56

[Function]

```
public setXAxisFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the XAxisFormat [▶ 160] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisFormat [▶ 1283]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.10

15.1.1.2.2.1. getGridShowHorizontalLines 57

[Function]

```
public getGridShowHorizontalLines(): boolean;
```

This function returns the currently set value for the GridShowHorizontalLines [▶ 154] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setGridShowHorizontalLines [▶ 1284]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setGridShowHorizontalLines 58

[Function]

```
public setGridShowHorizontalLines(newValue: boolean | null): void;
```

This function sets a new value for the GridShowHorizontalLines [▶ 154] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridShowHorizontalLines [▶ 1284]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getShowGrid
59

[Function]

```
public getShowGrid(): boolean;
```

This function returns the currently set value for the ShowGrid [▶ 154] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowGrid [▶ 1285]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setShowGrid
60

[Function]

```
public setShowGrid(newValue: boolean | null): void;
```

This function sets a new value for the ShowGrid [▶ 154] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowGrid [▶ 1285]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getShowYAxis
61

[Function]

```
public getShowYAxis(): boolean;
```

This function returns the currently set value for the ShowYAxis [▶ 156] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowYAxis [▶ 1286]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setShowYAxis
62

[Function]

```
public setShowYAxis(newValue: boolean | null): void;
```

This function sets a new value for the ShowYAxis [▶ 156] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowYAxis [▶ 1286]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYPosition 63

[Function]

```
public getYPosition(): string;
```

This function returns the currently set value for the YPosition [▶ 166] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYPosition [▶ 1287]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYPosition 64

[Function]

```
public setYPosition(newValue: string | null): void;
```

This function sets a new value for the YPosition [▶ 166] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYPosition [▶ 1287]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getShowYSubTicks 65

[Function]

```
public getShowYSubTicks(): boolean;
```

This function returns the currently set value for the ShowYSubTicks [▶ 156] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowYSubTicks [▶ 1288]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setShowYSubTicks 66

[Function]

```
public setShowYSubTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowYSubTicks [▶ 156] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowYSubTicks [▶ 1288]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYShowLabels
67

[Function]

```
public getYShowLabels(): boolean;
```

This function returns the currently set value for the YShowLabels [▶ 156] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setYShowLabels [▶ 1289]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYShowLabels
68

[Function]

```
public setYShowLabels(newValue: boolean | null): void;
```

This function sets a new value for the YShowLabels [▶ 156] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYShowLabels [▶ 1289]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYSubTickSteps
69

[Function]

```
public getYSubTickSteps(): number | null;
```

This function returns the currently set value for the YSubTickSteps [▶ 166] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYSubTickSteps [▶ 1290]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYSubTickSteps
70

[Function]

```
public setYSubTickSteps(newValue: number | null): void;
```

This function sets a new value for the YSubTickSteps [▶ 166] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYSubTickSteps [▶ 1289]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getShowYAxisName
71

[Function]

```
public getShowYAxisName(): boolean;
```

This function returns the currently set value for the ShowYAxisName [▶ 156] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowYAxisName [▶ 1291]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setShowYAxisName
72

[Function]

```
public setShowYAxisName(newValue: boolean | null): void;
```

This function sets a new value for the ShowYAxisName [▶ 156] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowYAxisName [▶ 1290]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisName
73

[Function]

```
public getYAxisName(): string | null;
```

This function returns the currently set value for the YAxisName [▶ 164] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisName [▶ 1292]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisName
74

[Function]

```
public setYAxisName(newValue: string | null): void;
```

This function sets a new value for the YAxisName [▶ 164] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisName [▶ 1291]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYMainTickMinValue
75

[Function]

```
public getYMainTickMinValue(): number | null;
```

This function returns the currently set value for the YMainTickMinValue [▶ 164] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYMainTickMinValue [▶ 1293]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYMainTickMinValue
76

[Function]

```
public setYMainTickMinValue(newValue: number | null): void;
```

This function sets a new value for the YMainTickMinValue [▶ 164] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYMainTickMinValue [▶ 1292]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisUnit
77

[Function]

```
public getYAxisUnit(): string | null;
```

This function returns the currently set value for the YAxisUnit [▶ 166] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisUnit [▶ 1294]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisUnit
78

[Function]

```
public setYAxisUnit(newValue: string | null): void;
```

This function sets a new value for the YAxisUnit [▶ 166] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisUnit [▶ 1293]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisDecimalPlaces
79

[Function]

```
public getYAxisDecimalPlaces(): number;
```

This function returns the currently set value for the YAxisDecimalPlaces [▶ 162] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisDecimalPlaces [▶ 1295]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisDecimalPlaces
80

[Function]

```
public setYAxisDecimalPlaces(newValue: number | null): void;
```

This function sets a new value for the YAxisDecimalPlaces [▶ 162] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisDecimalPlaces [▶ 1294]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYMainTickMaxValue
81

[Function]

```
public getYMainTickMaxValue(): number | null;
```

This function returns the currently set value for the YMainTickMaxValue [▶ 163] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYMainTickMaxValue [▶ 1295]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYMainTickMaxValue
82

[Function]

```
public setYMainTickMaxValue(newValue: number | null): void;
```

This function sets a new value for the YMainTickMaxValue [▶ 163] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYMainTickMaxValue [▶ 1295]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisAutoScaling 83

[Function]

```
public getYAxisAutoScaling(): boolean;
```

This function returns the currently set value for the YAxisAutoScaling [▶ 161] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setYAxisAutoScaling [▶ 1296]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisAutoScaling 84

[Function]

```
public setYAxisAutoScaling(newValue: boolean | null): void;
```

This function sets a new value for the YAxisAutoScaling [▶ 161] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisAutoScaling [▶ 1296]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYMainTickSteps
85

[Function]

```
public getYMainTickSteps(): number;
```

This function returns the currently set value for the YMainTickSteps [▶ 164] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYMainTickSteps [▶ 1297]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYMainTickSteps
86

[Function]

```
public setYMainTickSteps(newValue: number | null): void;
```

This function sets a new value for the YMainTickSteps [▶ 164] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYMainTickSteps [▶ 1297]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYLabelFontSize
87

[Function]

```
public getYLabelFontSize(): number;
```

This function returns the currently set value for the YLabelFontSize [▶ 163] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontSize [▶ 1298]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYLabelFontSize
88

[Function]

```
public setYLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the YLabelFontSize [▶ 163] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontSize [▶ 1298]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisWidth
89

[Function]

```
public getYAxisWidth(): number;
```

This function returns the currently set value for the YAxisWidth [▶ 166] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisWidth [▶ 1299]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisWidth
90

[Function]

```
public setYAxisWidth(newValue: number | null): void;
```

This function sets a new value for the YAxisWidth [▶ 166] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisWidth [▶ 1299]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisNameFontSizeUnit 91

[Function]

```
public getYAxisNameFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the YAxisNameFontSizeUnit [▶ 165] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setYAxisNameFontSizeUnit [▶ 1300]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisNameFontSizeUnit 92

[Function]

```
public setYAxisNameFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the YAxisNameFontSizeUnit [▶ 165] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisNameFontSizeUnit [▶ 1300]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisNameFontWeight
93

[Function]

```
public getYAxisNameFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the YAxisNameFontWeight [▶ 165] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setYAxisNameFontWeight [▶ 1301]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisNameFontWeight
94

[Function]

```
public setYAxisNameFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the YAxisNameFontWeight [▶ 165] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisNameFontWeight [▶ 1301]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYLabelFontWeight
95

[Function]

```
public getYLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the YLabelFontWeight [▶ 163] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setYLabelFontWeight [▶ 1302]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYLabelFontWeight
96

[Function]

```
public setYLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the YLabelFontWeight [▶ 163] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontWeight [▶ 1301]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYLabelFontSizeUnit
97

[Function]

```
public getYLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the YLabelFontSizeUnit [▶ 163] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setYLabelFontSizeUnit [▶ 1303]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYLabelFontSizeUnit
98

[Function]

```
public setYLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the YLabelFontSizeUnit [▶ 163] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontSizeUnit [▶ 1302]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisNameFontSize
99

[Function]

```
public getYAxisNameFontSize(): number;
```

This function returns the currently set value for the YAxisNameFontSize [▶ 165] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisNameFontSize [▶ 1304]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisNameFontSize
100

[Function]

```
public setYAxisNameFontSize(newValue: number | null): void;
```

This function sets a new value for the YAxisNameFontSize [▶ 165] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisNameFontSize [▶ 1303]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYLabelFontFamily
101

[Function]

```
public getYLabelFontFamily(): string;
```

This function returns the currently set value for the YLabelFontFamily [▶ 162] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontFamily [▶ 1305]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYLabelFontFamily
102

[Function]

```
public setYLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the YLabelFontFamily [▶ 162] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontFamily [▶ 1304]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getYAxisNameFontFamily
103

[Function]

```
public getYAxisNameFontFamily(): string;
```

This function returns the currently set value for the YAxisNameFontFamily [▶ 164] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisNameFontFamily [▶ 1306]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setYAxisNameFontFamily 104

[Function]

```
public setYAxisNameFontFamily(newValue: string | null): void;
```

This function sets a new value for the YAxisNameFontFamily [▶ 164] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisNameFontFamily [▶ 1305]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getSubgridShowVerticalLines 105

[Function]

```
public getSubgridShowVerticalLines(): boolean;
```

This function returns the currently set value for the SubgridShowVerticalLines [▶ 155] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setSubgridShowVerticalLines [▶ 1307]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setSubgridShowVerticalLines
106

[Function]

```
public setSubgridShowVerticalLines(newValue: boolean | null): void;
```

This function sets a new value for the SubgridShowVerticalLines [▶ 155] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridShowVerticalLines [▶ 1306]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getSubgridShowHorizontalLines
107

[Function]

```
public getSubgridShowHorizontalLines(): boolean;
```

This function returns the currently set value for the SubgridShowHorizontalLines [▶ 154] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setSubgridShowHorizontalLines [▶ 1307]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setSubgridShowHorizontalLines
108

[Function]

```
public setSubgridShowHorizontalLines(newValue: boolean | null): void;
```

This function sets a new value for the SubgridShowHorizontalLines [▶ 154] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridShowHorizontalLines [▶ 1307]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. getSubgridLineWidth
109

[Function]

```
public getSubgridLineWidth(): number;
```

This function returns the currently set value for the SubgridLineWidth [▶ 157] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSubgridLineWidth [▶ 1308]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.1. setSubgridLineWidth
110

[Function]

```
public setSubgridLineWidth(newValue: number | null): void;
```

This function sets a new value for the SubgridLineWidth [▶ 157] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridLineWidth [▶ 1308]

Origin: TcHmiBarChart [▶ 142]

Note: Available since version 1.8

15.1.1.2.2.2 TcHmiButton

[Class]

Simple button.

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.

Name	Origin	Description
getText [▶ 1316]	TcHmiButton	Getter for the Text attribute.
setText [▶ 1317]	TcHmiButton	Setter for the Text attribute.
getStateSymbol [▶ 1334]	TcHmiButton	Getter for the StateSymbol attribute.
setStateSymbol [▶ 1333]	TcHmiButton	Setter for the StateSymbol attribute.
getTextColor [▶ 1316]	TcHmiButton	Getter for the TextColor attribute.
setTextColor [▶ 1316]	TcHmiButton	Setter for the TextColor attribute.
getTextFontFamily [▶ 1332]	TcHmiButton	Getter for the TextFontFamily attribute.
setTextFontFamily [▶ 1333]	TcHmiButton	Setter for the TextFontFamily attribute.
getTextFontSize [▶ 1326]	TcHmiButton	Getter for the TextFontSize attribute.
setTextFontSize [▶ 1326]	TcHmiButton	Setter for the TextFontSize attribute.
getTextFontSizeUnit [▶ 1327]	TcHmiButton	Getter for the TextFontSizeUnit attribute.
setTextFontSizeUnit [▶ 1327]	TcHmiButton	Setter for the TextFontSizeUnit attribute.
getTextFontStyle [▶ 1330]	TcHmiButton	Getter for the TextFontStyle attribute.
setTextFontStyle [▶ 1331]	TcHmiButton	Setter for the TextFontStyle attribute.
getTextFontWeight [▶ 1331]	TcHmiButton	Getter for the TextFontWeight attribute.
setTextFontWeight [▶ 1332]	TcHmiButton	Setter for the TextFontWeight attribute.
getTextPadding [▶ 1328]	TcHmiButton	Getter for the TextPadding attribute.
setTextPadding [▶ 1329]	TcHmiButton	Setter for the TextPadding attribute.
getTextHorizontalAlignment [▶ 1325]	TcHmiButton	Getter for the TextHorizontalAlignment attribute.
setTextHorizontalAlignment [▶ 1325]	TcHmiButton	Setter for the TextHorizontalAlignment attribute.
getTextVerticalAlignment [▶ 1328]	TcHmiButton	Getter for the TextVerticalAlignment attribute.
setTextVerticalAlignment [▶ 1328]	TcHmiButton	Setter for the TextVerticalAlignment attribute.
getWordWrap [▶ 1329]	TcHmiButton	Getter for the WordWrap attribute.
setWordWrap [▶ 1330]	TcHmiButton	Setter for the WordWrap attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.

Name	Origin	Description
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getIcon [▶ 1317]	TcHmiButton	Getter for the Icon attribute.
setIcon [▶ 1318]	TcHmiButton	Setter for the Icon attribute.
getIconHeight [▶ 1320]	TcHmiButton	Getter for the IconHeight attribute.
setIconHeight [▶ 1321]	TcHmiButton	Setter for the IconHeight attribute.
getIconHeightUnit [▶ 1321]	TcHmiButton	Getter for the IconHeightUnit attribute.
setIconHeightUnit [▶ 1322]	TcHmiButton	Setter for the IconHeightUnit attribute.
getIconWidth [▶ 1318]	TcHmiButton	Getter for the IconWidth attribute.
setIconWidth [▶ 1319]	TcHmiButton	Setter for the IconWidth attribute.
getIconWidthUnit [▶ 1319]	TcHmiButton	Getter for the IconWidthUnit attribute.
setIconWidthUnit [▶ 1320]	TcHmiButton	Setter for the IconWidthUnit attribute.
getIconPadding [▶ 1324]	TcHmiButton	Getter for the IconPadding attribute.
setIconPadding [▶ 1324]	TcHmiButton	Setter for the IconPadding attribute.
getIconHorizontalAlignment [▶ 1323]	TcHmiButton	Getter for the IconHorizontalAlignment attribute.
setIconHorizontalAlignment [▶ 1323]	TcHmiButton	Setter for the IconHorizontalAlignment attribute.
getIconVerticalAlignment [▶ 1322]	TcHmiButton	Getter for the IconVerticalAlignment attribute.
setIconVerticalAlignment [▶ 1322]	TcHmiButton	Setter for the IconVerticalAlignment attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.

Name	Origin	Description
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Button

See also

Reference [▶ 167] of this control

Note: Available since version 1.8

15.1.1.2.2.2. getTextColor 1

[Function]

```
public getTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TextColor [▶ 176] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextColor [▶ 1316]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextColor 2

[Function]

```
public setTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TextColor [▶ 176] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextColor [▶ 1316]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getText 3

[Function]

```
public getText(): string;
```

This function returns the currently set value for the Text [▶ 176] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setText [▶ 1317]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setText
4

[Function]

```
public setText(newValue: string | null): void;
```

This function sets a new value for the Text [▶ 176] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getText [▶ 1316]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getIcon
5

[Function]

```
public getIcon(): string | null;
```

This function returns the currently set value for the Icon [▶ 175] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setIcon [▶ 1318]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.6. setIcon

[Function]

```
public setIcon(newValue: string | null): void;
```

This function sets a new value for the Icon [▶ 175] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIcon [▶ 1317]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.7. getIconWidth

[Function]

```
public getIconWidth(): number | null;
```

This function returns the currently set value for the IconWidth [▶ 177] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setIconWidth [▶ 1319]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setIconWidth

[Function]

```
public setIconWidth(newValue: number | null): void;
```

This function sets a new value for the IconWidth [▶ 177] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconWidth [▶ 1318]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getIconWidthUnit

[Function]

```
public getIconWidthUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the IconWidthUnit [▶ 177] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setIconWidthUnit [▶ 1320]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setIconWidthUnit

[Function]

```
public setIconWidthUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the IconWidthUnit [▶ 177] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconWidthUnit [▶ 1319]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getIconHeight

[Function]

```
public getIconHeight(): number | null;
```

This function returns the currently set value for the IconHeight [▶ 174] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setIconHeight [▶ 1321]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setIconHeight

[Function]

```
public setIconHeight(newValue: number | null): void;
```

This function sets a new value for the IconHeight [▶ 174] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconHeight [▶ 1320]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.3. getIconHeightUnit

[Function]

```
public getIconHeightUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the IconHeightUnit [▶ 174] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setIconHeightUnit [▶ 1322]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.14 setIconHeightUnit

[Function]

```
public setIconHeightUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the IconHeightUnit [▶ 174] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconHeightUnit [▶ 1321]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.15 getIconVerticalAlignment

[Function]

```
public getIconVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the IconVerticalAlignment [▶ 176] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setIconVerticalAlignment [▶ 1322]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.16 setIconVerticalAlignment

[Function]

```
public setIconVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the IconVerticalAlignment [▶ 176] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconVerticalAlignment [▶ 1322]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getIconHorizontalAlignment
17

[Function]

```
public getIconHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the IconHorizontalAlignment [▶ 175] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setIconHorizontalAlignment [▶ 1323]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setIconHorizontalAlignment
18

[Function]

```
public setIconHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the IconHorizontalAlignment [▶ 175] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconHorizontalAlignment [▶ 1323]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getIconPadding
19

[Function]

```
public getIconPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the IconPadding [▶ 175] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setIconPadding [▶ 1324]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setIconPadding
20

[Function]

```
public setIconPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the IconPadding [▶ 175] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getIconPadding [▶ 1324]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.1 getTextHorizontalAlignment

[Function]

```
public getTextHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the TextHorizontalAlignment [▶ 175] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextHorizontalAlignment [▶ 1325]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.2 setTextHorizontalAlignment

[Function]

```
public setTextHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the TextHorizontalAlignment [▶ 175] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextHorizontalAlignment [▶ 1325]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getTextFontSize
23

[Function]

```
public getTextFontSize(): number;
```

This function returns the currently set value for the TextFontSize [▶ 173] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontSize [▶ 1326]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextFontSize
24

[Function]

```
public setTextFontSize(newValue: number | null): void;
```

This function sets a new value for the TextFontSize [▶ 173] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSize [▶ 1326]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.25 getTextFontSizeUnit

[Function]

```
public getTextFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TextFontSizeUnit [▶ 173] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTextFontSizeUnit [▶ 1327]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2.26 setTextFontSizeUnit

[Function]

```
public setTextFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TextFontSizeUnit [▶ 173] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSizeUnit [▶ 1327]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getTextVerticalAlignment 27

[Function]

```
public getTextVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the TextVerticalAlignment [▶ 177] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextVerticalAlignment [▶ 1328]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextVerticalAlignment 28

[Function]

```
public setTextVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the TextVerticalAlignment [▶ 177] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextVerticalAlignment [▶ 1328]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getTextPadding 29

[Function]

```
public getTextPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the TextPadding [▶ 176] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextPadding [▶ 1329]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextPadding
30

[Function]

```
public setTextPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the TextPadding [▶ 176] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextPadding [▶ 1328]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getWordWrap
31

[Function]

```
public getWordWrap(): boolean;
```

This function returns the currently set value for the WordWrap [▶ 178] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setWordWrap [▶ 1330]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setWordWrap

32

[Function]

```
public setWordWrap(newValue: boolean | null): void;
```

This function sets a new value for the WordWrap [▶ 178] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWordWrap [▶ 1329]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getTextFontStyle

33

[Function]

```
public getTextFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the TextFontStyle [▶ 174] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setTextFontStyle [▶ 1331]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextFontStyle
34

[Function]

```
public setTextFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the TextFontStyle [▶ 174] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontStyle [▶ 1330]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getTextFontWeight
35

[Function]

```
public getTextFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the TextFontWeight [▶ 174] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setTextFontWeight [▶ 1332]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextFontWeight

[Function]

```
public setTextFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the TextFontWeight [▶ 174] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontWeight [▶ 1331]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.3. getTextFontFamily

[Function]

```
public getTextFontFamily(): string | null;
```

This function returns the currently set value for the TextFontFamily [▶ 173] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontFamily [▶ 1333]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. setTextFontFamily
38

[Function]

```
public setTextFontFamily(newValue: string | null): void;
```

This function sets a new value for the TextFontFamily [▶ 173] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontFamily [▶ 1332]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.8

15.1.1.2.2.2. getStateSymbol
39

[Function]

```
public getStateSymbol(): string;
```

This function returns the currently set value for the StateSymbol [▶ 178] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Symbol [▶ 2036] | null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStateSymbol [▶ 1334]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.10

15.1.1.2.2.2. setStateSymbol
40

[Function]

```
public setStateSymbol(newValue: string | null): void;
```

This function sets a new value for the StateSymbol [▶ 178] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Symbol [▶ 2036], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStateSymbol [▶ 1333]

Origin: TcHmiButton [▶ 167]

Note: Available since version 1.10

15.1.1.2.2.3 TcHmiCheckbox

[Class]

Simple checkbox. A checkbox has a ToggleState parameter. This parameter has the value "Normal" if no checkmark is set and "Active" if the checkmark is set. The .onToggleStateChanged event can be used to perform actions when the state of the checkbox changes.

Attribute functions

Name	Origin	Description
getToggleStateIconColor [▶ 1340]	TcHmiCheckbox	Getter for the ToggleStateIconColor attribute.
setToggleStateIconColor [▶ 1340]	TcHmiCheckbox	Setter for the ToggleStateIconColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.

Name	Origin	Description
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.

Name	Origin	Description
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getStateSymbol [▶ 1342]	TcHmiCheckbox	Setter for the StateSymbol attribute.
setStateSymbol [▶ 1343]	TcHmiCheckbox	Getter for the StateSymbol attribute.
getToggleState [▶ 1341]	TcHmiCheckbox	Getter for the ToggleState attribute.
setToggleState [▶ 1342]	TcHmiCheckbox	Setter for the ToggleState attribute.
getToggleGroup [▶ 1340]	TcHmiCheckbox	Setter for the ToggleGroup attribute.
setToggleGroup [▶ 1341]	TcHmiCheckbox	Getter for the ToggleGroup attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.

Name	Origin	Description
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Checkbox

See also

Reference [▶ 179] of this control

Note: Available since version 1.8

15.1.1.2.2.3. getToggleStateIconColor 1

[Function]

```
public getToggleStateIconColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the ToggleStateIconColor [▶ 185] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setToggleStateIconColor [▶ 1340]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.8

15.1.1.2.2.3. setToggleStateIconColor 2

[Function]

```
public setToggleStateIconColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the ToggleStateIconColor [▶ 185] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getToggleStateIconColor [▶ 1340]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.8

15.1.1.2.2.3. getToggleGroup 3

[Function]

```
public getToggleGroup(): string | null;
```

This function returns the currently set value for the ToggleGroup [▶ 184] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setToggleGroup [▶ 1341]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.10

15.1.1.2.2.3. setToggleGroup
4

[Function]

```
public setToggleGroup(newValue: string | null): void;
```

This function sets a new value for the ToggleGroup [▶ 184] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getToggleGroup [▶ 1340]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.10

15.1.1.2.2.3. getToggleState
5

[Function]

```
public getToggleState(): TcHmi.ToggleState;
```

This function returns the currently set value for the ToggleState [▶ 184] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.ToggleState [▶ 2097]	The current value of the attribute.

See also

Attribute setter: setToggleState [▶ 1342]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.8

15.1.1.2.2.3. setToggleState
6

[Function]

```
public setToggleState(newValue: TcHmi.ToggleState | null): void;
```

This function sets a new value for the ToggleState [▶ 184] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.ToggleState [▶ 2097], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getToggleState [▶ 1341]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.8

15.1.1.2.2.3. getStateSymbol
7

[Function]

```
public getStateSymbol(): string;
```

This function returns the currently set value for the StateSymbol [▶ 184] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Symbol [▶ 2036] | null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStateSymbol [▶ 1343]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.10

15.1.1.2.2.3. setStateSymbol
8

[Function]

```
public setStateSymbol(newValue: string | null): void;
```

This function sets a new value for the StateSymbol [▶ 184] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Symbol [▶ 2036], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStateSymbol [▶ 1342]

Origin: TcHmiCheckbox [▶ 179]

Note: Available since version 1.10

15.1.1.2.2.4 TcHmiComboBox

[Class]

A combo box is a drop-down menu for selecting one of several options. Each option is defined by its ID and also has a text that is displayed in the drop-down menu, and a value.

From version 1.10 it is also possible to enter free text directly.

Attribute functions

Name	Origin	Description
getDropDownBackgroundColor [▶ 1352]	TcHmiCombobox	Getter for the DropDownBackgroundColor attribute.
setDropDownBackgroundColor [▶ 1353]	TcHmiCombobox	Setter for the DropDownBackgroundColor attribute.
getDropDownTextColor [▶ 1352]	TcHmiCombobox	Getter for the DropDownTextColor attribute.
setDropDownTextColor [▶ 1352]	TcHmiCombobox	Setter for the DropDownTextColor attribute.
getDropDownToggleButtonArrowColor [▶ 1356]	TcHmiCombobox	Getter for the DropDownToggleButtonArrowColor attribute.
setDropDownToggleButtonArrowColor [▶ 1357]	TcHmiCombobox	Setter for the DropDownToggleButtonArrowColor attribute.
getDropDownToggleButtonBackgroundColor [▶ 1354]	TcHmiCombobox	Getter for the DropDownToggleButtonBackgroundColor attribute.
setDropDownToggleButtonBackgroundColor [▶ 1355]	TcHmiCombobox	Setter for the DropDownToggleButtonBackgroundColor attribute.
getDropDownHighlightColor [▶ 1353]	TcHmiCombobox	Getter for the DropDownHighlightColor attribute.
setDropDownHighlightColor [▶ 1354]	TcHmiCombobox	Setter for the DropDownHighlightColor attribute.
getDropDownHighlightTextColor [▶ 1355]	TcHmiCombobox	Getter for the DropDownHighlightTextColor attribute.
setDropDownHighlightTextColor [▶ 1356]	TcHmiCombobox	Setter for the DropDownHighlightTextColor attribute.
getDropDownFontFamily [▶ 1375]	TcHmiCombobox	Getter for the DropDownFontFamily attribute.
setDropDownFontFamily [▶ 1376]	TcHmiCombobox	Setter for the DropDownFontFamily attribute.
getDropDownFontSize [▶ 1376]	TcHmiCombobox	Getter for the DropDownFontSize attribute.
setDropDownFontSize [▶ 1377]	TcHmiCombobox	Setter for the DropDownFontSize attribute.
getDropDownFontSizeUnit [▶ 1377]	TcHmiCombobox	Getter for the DropDownFontSizeUnit attribute.
setDropDownFontSizeUnit [▶ 1377]	TcHmiCombobox	Setter for the DropDownFontSizeUnit attribute.
getDropDownFontStyle [▶ 1378]	TcHmiCombobox	Getter for the DropDownFontStyle attribute.
setDropDownFontStyle [▶ 1378]	TcHmiCombobox	Setter for the DropDownFontStyle attribute.
getDropDownFontWeight [▶ 1374]	TcHmiCombobox	Getter for the DropDownFontWeight attribute.
setDropDownFontWeight [▶ 1375]	TcHmiCombobox	Setter for the DropDownFontWeight attribute.
getDropDownHorizontalAlignment [▶ 1373]	TcHmiCombobox	Getter for the DropDownHorizontalAlignment attribute.

Name	Origin	Description
setDropDownHorizontalAlignment [▶ 1374]	TcHmiCombobox	Setter for the DropDownHorizontalAlignment attribute.
getDropDownVerticalAlignment [▶ 1372]	TcHmiCombobox	Getter for the DropDownVerticalAlignment attribute.
setDropDownVerticalAlignment [▶ 1373]	TcHmiCombobox	Setter for the DropDownVerticalAlignment attribute.
getDropDownStyle [▶ 1357]	TcHmiCombobox	Getter for the attribute DropDownStyle
setDropDownStyle [▶ 1358]	TcHmiCombobox	Setter for the attribute DropDownStyle.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.

Name	Origin	Description
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.

Name	Origin	Description
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.

Name	Origin	Description
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getComboboxText [▶ 1366]	TcHmiCombobox	Getter for the ComboboxText attribute.
setComboboxText [▶ 1366]	TcHmiCombobox	Setter for the ComboboxText attribute.
getSelectedText [▶ 1363]	TcHmiCombobox	Getter for the SelectedText attribute.
getSelectedValue [▶ 1363]	TcHmiCombobox	Getter for the SelectedValue attribute.
getSelectedId [▶ 1364]	TcHmiCombobox	Getter for the SelectedId attribute.
getSelectedIndex [▶ 1363]	TcHmiCombobox	Getter for the SelectedIndex attribute.
setSelectedIndex [▶ 1364]	TcHmiCombobox	Setter for the SelectedIndex attribute.
getSrcData [▶ 1365]	TcHmiCombobox	Getter for the SrcData attribute.
setSrcData [▶ 1365]	TcHmiCombobox	Setter for the SrcData attribute.
getTextColor [▶ 1351]	TcHmiCombobox	Getter for the TextColor attribute.
setTextColor [▶ 1351]	TcHmiCombobox	Setter for the TextColor attribute.
getTextFontFamily [▶ 1367]	TcHmiCombobox	Getter for the TextFontFamily attribute.
setTextFontFamily [▶ 1367]	TcHmiCombobox	Setter for the TextFontFamily attribute.
getTextFontSize [▶ 1368]	TcHmiCombobox	Getter for the TextFontSize attribute.
setTextFontSize [▶ 1368]	TcHmiCombobox	Setter for the TextFontSize attribute.
getTextFontSizeUnit [▶ 1369]	TcHmiCombobox	Getter for the TextFontSizeUnit attribute.
setTextFontSizeUnit [▶ 1369]	TcHmiCombobox	Setter for the TextFontSizeUnit attribute.
getTextFontStyle [▶ 1370]	TcHmiCombobox	Getter for the TextFontStyle attribute.
setTextFontStyle [▶ 1370]	TcHmiCombobox	Setter for the TextFontStyle attribute.

Name	Origin	Description
getTextFontWeight [▶ 1362]	TcHmiCombobox	Getter for the TextFontWeight attribute.
setTextFontWeight [▶ 1362]	TcHmiCombobox	Setter for the TextFontWeight attribute.
getTextHorizontalAlignment [▶ 1371]	TcHmiCombobox	Getter for the TextHorizontalAlignment attribute.
setTextHorizontalAlignment [▶ 1371]	TcHmiCombobox	Setter for the TextHorizontalAlignment attribute.
getTextVerticalAlignment [▶ 1371]	TcHmiCombobox	Getter for the TextVerticalAlignment attribute.
setTextVerticalAlignment [▶ 1372]	TcHmiCombobox	Setter for the TextVerticalAlignment attribute.
getDataHeight [▶ 1360]	TcHmiCombobox	Getter for the DataHeight attribute.
setDataHeight [▶ 1360]	TcHmiCombobox	Setter for the DataHeight attribute.
getDataHeightUnit [▶ 1361]	TcHmiCombobox	Getter for the DataHeightUnit attribute.
setDataHeightUnit [▶ 1361]	TcHmiCombobox	Setter for the DataHeightUnit attribute.
getListHeight [▶ 1358]	TcHmiCombobox	Getter for the ListHeight attribute. (only TcHmi v1.8)
setListHeight [▶ 1358]	TcHmiCombobox	Setter for the ListHeight attribute. (only TcHmi v1.8)
getListHeightUnit [▶ 1359]	TcHmiCombobox	Getter for the ListHeightUnit attribute. (only TcHmi v1.8)
setListHeightUnit [▶ 1359]	TcHmiCombobox	Setter for the ListHeightUnit attribute. (only TcHmi v1.8)
getContentPadding [▶ 1381]	TcHmiCombobox	Getter for the ContentPadding attribute
setContentPadding [▶ 1382]	TcHmiCombobox	Setter for the ContentPadding attribute
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Combobox

See also

Reference [▶ 185] of this control

Note: Available since version 1.8

15.1.1.2.2.4. getTextColor 1

[Function]

```
public getTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TextColor [▶ 202] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextColor [▶ 1351]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextColor 2

[Function]

```
public setTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TextColor [▶ 202] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextColor [▶ 1351]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownTextColor 3

[Function]

```
public getDropDownTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the DropDownTextColor [▶ 197] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownTextColor [▶ 1352]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownTextColor 4

[Function]

```
public setDropDownTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the DropDownTextColor [▶ 197] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownTextColor [▶ 1352]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownBackgroundColor 5

[Function]

```
public getDropDownBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the DropDownBackgroundColor [▶ 195] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownBackgroundColor [▶ 1353]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownBackgroundColor 6

[Function]

```
public setDropDownBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the DropDownBackgroundColor [▶ 195] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownBackgroundColor [▶ 1352]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownHighlightColor 7

[Function]

```
public getDropDownHighlightColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the DropDownHighlightColor [▶ 196] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownHighlightColor [▶ 1354]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownHighlightColor 8

[Function]

```
public setDropDownHighlightColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the DropDownHighlightColor [▶ 196] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownHighlightColor [▶ 1353]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownToggleButtonBackgroundColor 9

[Function]

```
public getDropDownToggleButtonBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the DropDownToggleButtonBackgroundColor [▶ 197] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownToggleButtonBackgroundColor [▶ 1355]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownToggleButtonBackgroundColor 10

[Function]

```
public setDropDownToggleButtonBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the DropDownToggleButtonBackgroundColor [▶ 197] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownToggleButtonBackgroundColor [▶ 1354]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownHighlightTextColor 11

[Function]

```
public getDropDownHighlightTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the DropDownHighlightTextColor [▶ 196] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownHighlightTextColor [▶ 1356]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownHighlightTextColor 12

[Function]

```
public setDropDownHighlightTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the DropDownHighlightTextColor [▶ 196] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownHighlightTextColor [▶ 1355]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownToggleButtonArrowColor 13

[Function]

```
public getDropDownToggleButtonArrowColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the DropDownToggleButtonArrowColor [▶ 197] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownToggleButtonArrowColor [▶ 1357]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownToggleButtonArrowColor 14

[Function]

```
public setDropDownToggleButtonArrowColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the DropDownToggleButtonArrowColor [▶ 197] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownToggleButtonArrowColor [▶ 1356]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownStyle 15

[Function]

```
public getDropDownStyle():'Classic' | 'Mobile' | null;
```

This function returns the currently set value for the DropDownStyle [▶ 198] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Classic', 'Mobile' , null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownStyle [▶ 1358]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. setDropDownStyle
16

[Function]

```
public setDropDownStyle(newValue: 'Classic' | 'Mobile' | null): void;
```

This function sets a new value for the DropDownStyle [▶ 198] attribute.

Parameter

Name	Type	Description
newValue	'Classic', 'Mobile' , null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownStyle [▶ 1357]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. getMaxListHeight
17

[Function]

```
public getMaxListHeight(): number;
```

This function returns the currently set value for the MaxListHeight [▶ 200] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMaxListHeight [▶ 1358]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. setMaxListHeight
18

[Function]

```
public setMaxListHeight(newValue: number | null): void;
```

This function sets a new value for the MaxListHeight [▶ 200] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMaxListHeight [▶ 1358]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. getMaxListHeightUnit
19

[Function]

```
public getMaxListHeightUnit(): 'px';
```

This function returns the currently set value for the ListMaxHeightUnit [▶ 200] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px' [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMaxListHeightUnit [▶ 1359]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. setMaxListHeightUnit
20

[Function]

```
public setMaxListHeightUnit(newValue: 'px' | null): void;
```

This function sets a new value for the MaxListHeightUnit [▶ 200] attribute.

Parameter

Name	Type	Description
newValue	'px' [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMaxListHeightUnit [▶ 1359]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. getDataHeight
21

[Function]

```
public getDataHeight(): number;
```

This function returns the currently set value for the DataHeight [▶ 194] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDataHeight [▶ 1360]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDataHeight
22

[Function]

```
public setDataHeight(newValue: number | null): void;
```

This function sets a new value for the DataHeight [▶ 194] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDataHeight [▶ 1360]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDataHeightUnit 23

[Function]

```
public getDataHeightUnit(): 'px';
```

This function returns the currently set value for the DataHeightUnit [▶ 194] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px' [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDataHeightUnit [▶ 1361]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDataHeightUnit 24

[Function]

```
public setDataHeightUnit(newValue: 'px' | null): void;
```

This function sets a new value for the DataHeightUnit [▶ 194] attribute.

Parameter

Name	Type	Description
newValue	'px' [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDataHeightUnit [▶ 1361]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextFontWeight
25

[Function]

```
public getTextFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the TextFontWeight [▶ 199] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setTextFontWeight [▶ 1362]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextFontWeight
26

[Function]

```
public setTextFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the TextFontWeight [▶ 199] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontWeight [▶ 1362]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getSelectedValue 27

[Function]

```
public getSelectedValue(): string | null;
```

This function returns the currently set value for the SelectedValue [▶ 201] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getSelectedText 28

[Function]

```
public getSelectedText(): string | null;
```

This function returns the currently set value for the SelectedText [▶ 201] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getSelectedIndex 29

[Function]

```
public getSelectedIndex(): number | null;
```

This function returns the currently set value for the SelectedIndex [▶ 201] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSelectedIndex [▶ 1364]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setSelectedIndex
30

[Function]

```
public setSelectedIndex(newValue: number | null): void;
```

This function sets a new value for the SelectedIndex [▶ 201] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSelectedIndex [▶ 1363]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getSelectedId
31

[Function]

```
public getSelectedId(): number | null;
```

This function returns the currently set value for the SelectedId [▶ 201] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getSrcData 32

[Function]

```
public getSrcData(): TcHmi.Controls.Beckhoff.TcHmiCombobox.ListItem[] | any[] | Dictionary<any>;
```

This function returns the currently set value for the SrcData [▶ 202] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiCombobox.ListItem [▶ 1379][] | any [▶ 2107][] | Dictionary<any [▶ 2107]>, null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSrcData [▶ 1365]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setSrcData 33

[Function]

```
public setSrcData(newValue: TcHmi.Controls.Beckhoff.TcHmiCombobox.ListItem[] | any[] |
Dictionary<any> | null): void;
```

This function sets a new value for the SrcData [▶ 202] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiCombobox.ListItem [▶ 1379][] | any [▶ 2107][] | Dictionary<any [▶ 2107]>, null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSrcData [▶ 1365]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getComboboxText 34

[Function]

```
public getComboboxText(): string;
```

This function returns the currently set value for the ComboboxText [▶ 194] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setComboboxText [▶ 1366]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setComboboxText 35

[Function]

```
public setComboboxText(newValue: string | null): void;
```

This function sets a new value for the ComboboxText [▶ 194] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getComboboxText [▶ 1366]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextFontFamily
36

[Function]

```
public getTextFontFamily(): string | null;
```

This function returns the currently set value for the TextFontFamily [▶ 198] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontFamily [▶ 1367]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextFontFamily
37

[Function]

```
public setTextFontFamily(newValue: string | null): void;
```

This function sets a new value for the TextFontFamily [▶ 198] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontFamily [▶ 1367]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextFontSize
38

[Function]

```
public getTextFontSize(): number;
```

This function returns the currently set value for the TextFontSize [▶ 198] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontSize [▶ 1368]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextFontSize
39

[Function]

```
public setTextFontSize(newValue: number | null): void;
```

This function sets a new value for the TextFontSize [▶ 198] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSize [▶ 1368]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextFontSizeUnit 40

[Function]

```
public getTextFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TextFontSizeUnit [▶ 199] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTextFontSizeUnit [▶ 1369]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextFontSizeUnit 41

[Function]

```
public setTextFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TextFontSizeUnit [▶ 199] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSizeUnit [▶ 1369]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextFontStyle
42

[Function]

```
public getTextFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the TextFontStyle [▶ 199] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setTextFontStyle [▶ 1370]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextFontStyle
43

[Function]

```
public setTextFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the TextFontStyle [▶ 199] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontStyle [▶ 1370]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextHorizontalAlignment
44

[Function]

```
public getTextHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the TextHorizontalAlignment [▶ 199] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextHorizontalAlignment [▶ 1371]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextHorizontalAlignment
45

[Function]

```
public setTextHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the TextHorizontalAlignment [▶ 199] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextHorizontalAlignment [▶ 1371]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getTextVerticalAlignment
46

[Function]

```
public getTextVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the TextVerticalAlignment [▶ 200] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextVerticalAlignment [▶ 1372]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setTextVerticalAlignment 47

[Function]

```
public setTextVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the TextVerticalAlignment [▶ 200] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextVerticalAlignment [▶ 1371]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownVerticalAlignment 48

[Function]

```
public getDropDownVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the DropDownVerticalAlignment [▶ 198] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setDropDownVerticalAlignment [▶ 1373]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownVerticalAlignment
49

[Function]

```
public setDropDownVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the DropDownVerticalAlignment [▶ 198] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownVerticalAlignment [▶ 1372]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownHorizontalAlignment
50

[Function]

```
public getDropDownHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the DropDownHorizontalAlignment [▶ 197] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setDropDownHorizontalAlignment [▶ 1374]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownHorizontalAlignment 51

[Function]

```
public setDropDownHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the DropDownHorizontalAlignment [▶ 197] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownHorizontalAlignment [▶ 1373]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownFontWeight 52

[Function]

```
public getDropDownFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the DropDownFontWeight [▶ 196] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setDropDownFontWeight [▶ 1375]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownFontWeight 53

[Function]

```
public setDropDownFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the DropDownFontWeight [▶ 196] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownFontWeight [▶ 1374]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownFontFamily 54

[Function]

```
public getDropDownFontFamily(): string | null;
```

This function returns the currently set value for the DropDownFontFamily [▶ 195] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownFontFamily [▶ 1376]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownFontFamily 55

[Function]

```
public setDropDownFontFamily(newValue: string | null): void;
```

This function sets a new value for the DropDownFontFamily [▶ 195] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownFontFamily [▶ 1375]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownFontSize 56

[Function]

```
public getDropDownFontSize(): number;
```

This function returns the currently set value for the DropDownFontSize [▶ 195] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setDropDownFontSize [▶ 1377]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownFontSize
57

[Function]

```
public setDropDownFontSize(newValue: number | null): void;
```

This function sets a new value for the DropDownFontSize [▶ 195] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownFontSize [▶ 1376]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownFontSizeUnit
58

[Function]

```
public getDropDownFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the DropDownFontSizeUnit [▶ 195] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setDropDownFontSizeUnit [▶ 1377]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownFontSizeUnit
59

[Function]

```
public setDropDownFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the DropDownFontSizeUnit [▶ 195] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownFontSizeUnit [▶ 1377]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. getDropDownFontStyle 60

[Function]

```
public getDropDownFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the DropDownFontStyle [▶ 196] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setDropDownFontStyle [▶ 1378]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setDropDownFontStyle 61

[Function]

```
public setDropDownFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the DropDownFontStyle [▶ 196] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getDropDownFontStyle [▶ 1378]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. ListItem
62

[Interface]

```
export interface ListItem {
    id?: number | null;
    text: string | number | boolean | null;
    value?: any | null;
}
```

Properties

Name	Type	Description
id [optional]	number [▶ 2108], null [▶ 2108]	Id of the entry
text	string [▶ 2108], number [▶ 2108], boolean [▶ 2107], null [▶ 2108]	Display text of the entry
value [optional]	any [▶ 2107], null [▶ 2108]	Value of the entry

Note: Available since version 1.8

15.1.1.2.2.4. getText
63

[Function]

```
public getComboboxText(): string;
```

This function returns the currently set value for the ComboboxText attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setComboboxText

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.8

15.1.1.2.2.4. setText
64

[Function]

```
public setText(newValue: string | null): void;
```

This function sets a new value for the Text [▶ 202] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getText [▶ 1379]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. getAllowTextInput
65

[Function]

```
public getAllowTextInput(): boolean;
```

This function returns the currently set value for the AllowTextInput [▶ 203] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setAllowTextInput [▶ 1381]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. setAllowTextInput 66

[Function]

```
public setAllowTextInput(newValue: boolean | null): void;
```

This function sets a new value for the AllowTextInput [▶ 203] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getAllowTextInput [▶ 1380]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. getContentPadding 67

[Function]

```
public getContentPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the ContentPadding [▶ 203] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setContentPadding [▶ 1382]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.4. setContentPadding
68

[Function]

```
public setContentPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the ContentPadding [▶ 203] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getContentPadding [▶ 1381]

Origin: TcHmiCombobox [▶ 185]

Note: Available since version 1.10

15.1.1.2.2.5 TcHmiDatagrid

[Class]

Table for generic display of data.

Attribute functions

Name	Origin	Description
getHeaderBackgroundColor [▶ 1392]	TcHmiDatagrid	Getter for the HeaderBackgroundColor attribute.
setHeaderBackgroundColor [▶ 1393]	TcHmiDatagrid	Setter for the HeaderBackgroundColor attribute.
getHeaderCellPadding [▶ 1411]	TcHmiDatagrid	Getter for the HeaderCellPadding attribute.
setHeaderCellPadding [▶ 1412]	TcHmiDatagrid	Setter for the HeaderCellPadding attribute.
getHeaderFontFamily [▶ 1410]	TcHmiDatagrid	Getter for the HeaderFontFamily attribute.
setHeaderFontFamily [▶ 1411]	TcHmiDatagrid	Setter for the HeaderFontFamily attribute.
getHeaderFontSize [▶ 1409]	TcHmiDatagrid	Getter for the HeaderFontSize attribute.
setHeaderFontSize [▶ 1410]	TcHmiDatagrid	Setter for the HeaderFontSize attribute.
getHeaderFontSizeUnit [▶ 1408]	TcHmiDatagrid	Getter for the HeaderFontSizeUnit attribute.
setHeaderFontSizeUnit [▶ 1409]	TcHmiDatagrid	Setter for the HeaderFontSizeUnit attribute.
getHeaderFontStyle [▶ 1412]	TcHmiDatagrid	Getter for the HeaderFontStyle attribute.
setHeaderFontStyle [▶ 1412]	TcHmiDatagrid	Setter for the HeaderFontStyle attribute.
getHeaderFontWeight [▶ 1407]	TcHmiDatagrid	Getter for the HeaderFontWeight attribute.
setHeaderFontWeight [▶ 1408]	TcHmiDatagrid	Setter for the HeaderFontWeight attribute.
getHeaderHeight [▶ 1406]	TcHmiDatagrid	Getter for the HeaderHeight attribute.
setHeaderHeight [▶ 1406]	TcHmiDatagrid	Setter for the HeaderHeight attribute.
getHeaderHeightUnit [▶ 1406]	TcHmiDatagrid	Getter for the HeaderHeightUnit attribute.
setHeaderHeightUnit [▶ 1407]	TcHmiDatagrid	Setter for the HeaderHeightUnit attribute.
getHeaderTextColor [▶ 1391]	TcHmiDatagrid	Getter for the HeaderTextColor attribute.
setHeaderTextColor [▶ 1392]	TcHmiDatagrid	Setter for the HeaderTextColor attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.

Name	Origin	Description
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.

Name	Origin	Description
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.

Name	Origin	Description
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getRowHeight [▶ 1399]	TcHmiDatagrid	Getter for the RowHeight attribute.
setRowHeight [▶ 1399]	TcHmiDatagrid	Setter for the RowHeight attribute.
getRowHeightUnit [▶ 1400]	TcHmiDatagrid	Getter for the RowHeightUnit attribute.

Name	Origin	Description
getShowRowNumbers [▶ 1413]	TcHmiDatagrid	Getter for the ShowRowNumbers attribute.
setShowRowNumbers [▶ 1413]	TcHmiDatagrid	Setter for the ShowRowNumbers attribute.
getRowNumbersBackgroundColor [▶ 1390]	TcHmiDatagrid	Getter for the RowNumbersBackgroundColor attribute.
setRowNumbersBackgroundColor [▶ 1391]	TcHmiDatagrid	Setter for the RowNumbersBackgroundColor attribute.
getRowNumbersCellPadding [▶ 1419]	TcHmiDatagrid	Getter for the RowNumbersCellPadding attribute.
setRowNumbersCellPadding [▶ 1420]	TcHmiDatagrid	Setter for the RowNumbersCellPadding attribute.
getRowNumbersFontFamily [▶ 1420]	TcHmiDatagrid	Getter for the RowNumbersFontFamily attribute.
setRowNumbersFontFamily [▶ 1421]	TcHmiDatagrid	Setter for the RowNumbersFontFamily attribute.
getRowNumbersFontSize [▶ 1421]	TcHmiDatagrid	Getter for the RowNumbersFontSize attribute.
setRowNumbersFontSize [▶ 1422]	TcHmiDatagrid	Setter for the RowNumbersFontSize attribute.
getRowNumbersFontSizeUnit [▶ 1422]	TcHmiDatagrid	Getter for the RowNumbersFontSizeUnit attribute.
setRowNumbersFontSizeUnit [▶ 1423]	TcHmiDatagrid	Setter for the RowNumbersFontSizeUnit attribute.
getRowNumbersFontStyle [▶ 1423]	TcHmiDatagrid	Getter for the RowNumbersFontStyle attribute.
setRowNumbersFontStyle [▶ 1424]	TcHmiDatagrid	Setter for the RowNumbersFontStyle attribute.
getRowNumbersFontWeight [▶ 1415]	TcHmiDatagrid	Getter for the RowNumbersFontWeight attribute.
setRowNumbersFontWeight [▶ 1415]	TcHmiDatagrid	Setter for the RowNumbersFontWeight attribute.
getRowNumbersHorizontalAlignment [▶ 1418]	TcHmiDatagrid	Getter for the RowNumbersHorizontalAlignment attribute.
setRowNumbersHorizontalAlignment [▶ 1418]	TcHmiDatagrid	Setter for the RowNumbersHorizontalAlignment attribute.
getRowNumbersResizable [▶ 1414]	TcHmiDatagrid	Getter for the RowNumbersResizable attribute.
setRowNumbersResizable [▶ 1414]	TcHmiDatagrid	Setter for the RowNumbersResizable attribute.
getRowNumbersTextColor [▶ 1394]	TcHmiDatagrid	Getter for the RowNumbersTextColor attribute.
setRowNumbersTextColor [▶ 1395]	TcHmiDatagrid	Setter for the RowNumbersTextColor attribute.
getRowNumbersVerticalAlignment [▶ 1418]	TcHmiDatagrid	Getter for the RowNumbersVerticalAlignment attribute.

Name	Origin	Description
setRowNumbersVerticalAlignment [▶ 1419]	TcHmiDatagrid	Setter for the RowNumbersVerticalAlignment attribute.
getRowNumbersWidth [▶ 1416]	TcHmiDatagrid	Getter for the RowNumbersWidth attribute.
setRowNumbersWidth [▶ 1416]	TcHmiDatagrid	Setter for the RowNumbersWidth attribute.
getRowNumbersWidthUnit [▶ 1417]	TcHmiDatagrid	Getter for the RowNumbersWidthUnit attribute.
setRowNumbersWidthUnit [▶ 1417]	TcHmiDatagrid	Setter for the RowNumbersWidthUnit attribute.
getSrcColumn [▶ 1395]	TcHmiDatagrid	Getter for the SrcColumn attribute.
setSrcColumn [▶ 1395]	TcHmiDatagrid	Setter for the SrcColumn attribute.
getSrcData [▶ 1396]	TcHmiDatagrid	Getter for the SrcData attribute.
setSrcData [▶ 1396]	TcHmiDatagrid	Setter for the SrcData attribute.
getSelectedRowIndex [▶ 1397]	TcHmiDatagrid	Getter for the SelectedRowIndex attribute.
setSelectedRowIndex [▶ 1397]	TcHmiDatagrid	Setter for the SelectedRowIndex attribute.
getSelectedColumnIndex [▶ 1398]	TcHmiDatagrid	Getter for the SelectedColumnIndex attribute.
setSelectedColumnIndex [▶ 1398]	TcHmiDatagrid	Setter for the SelectedColumnIndex attribute.
getGridFontFamily [▶ 1403]	TcHmiDatagrid	Getter for the GridFontFamily attribute.
setGridFontFamily [▶ 1403]	TcHmiDatagrid	Setter for the GridFontFamily attribute.
getGridFontSize [▶ 1402]	TcHmiDatagrid	Getter for the GridFontSize attribute.
setGridFontSize [▶ 1402]	TcHmiDatagrid	Setter for the GridFontSize attribute.
getGridFontSizeUnit [▶ 1401]	TcHmiDatagrid	Getter for the GridFontSizeUnit attribute.
setGridFontSizeUnit [▶ 1401]	TcHmiDatagrid	Setter for the GridFontSizeUnit attribute.
getGridFontStyle [▶ 1400]	TcHmiDatagrid	Getter for the GridFontStyle attribute.
setGridFontStyle [▶ 1400]	TcHmiDatagrid	Setter for the GridFontStyle attribute.
getGridFontWeight [▶ 1405]	TcHmiDatagrid	Getter for the GridFontWeight attribute.
setGridFontWeight [▶ 1405]	TcHmiDatagrid	Setter for the GridFontWeight attribute.
getCellContentPadding [▶ 1404]	TcHmiDatagrid	Getter for the CellContentPadding attribute.
setCellContentPadding [▶ 1404]	TcHmiDatagrid	Setter for the CellContentPadding attribute.
getTableBorderColor [▶ 1393]	TcHmiDatagrid	Getter for the TableBorderColor attribute.
setTableBorderColor [▶ 1394]	TcHmiDatagrid	Setter for the TableBorderColor attribute.
getTableBorderStyle [▶ 1426]	TcHmiDatagrid	Getter for the TableBorderStyle attribute.

Name	Origin	Description
setTableBorderStyle [▶ 1426]	TcHmiDatagrid	Setter for the TableBorderStyle attribute.
getTableBorderWidth [▶ 1424]	TcHmiDatagrid	Getter for the TableBorderWidth attribute.
setTableBorderWidth [▶ 1424]	TcHmiDatagrid	Setter for the TableBorderWidth attribute.
getTableBorderWidthUnit [▶ 1425]	TcHmiDatagrid	Getter for the TableBorderWidthUnit attribute.
setTableBorderWidthUnit [▶ 1425]	TcHmiDatagrid	Setter for the TableBorderWidthUnit attribute.
getSrcDataHasChanged [▶ 1428]	TcHmiDatagrid	Getter for the SrcDataHasChanged attribute
getSelectedRowValue [▶ 1429]	TcHmiDatagrid	Getter for the SelectedRowValue attribute
getSelectedCellValue [▶ 1429]	TcHmiDatagrid	Getter for the SelectedCellValue attribute
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Datagrid

See also

Reference [▶ 203] of this control

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersBackgroundColor 1

[Function]

```
public getRowNumbersBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the RowNumbersBackgroundColor [▶ 223] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowNumbersBackgroundColor [▶ 1391]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersBackgroundColor 2

[Function]

```
public setRowNumbersBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the RowNumbersBackgroundColor [▶ 223] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersBackgroundColor [▶ 1390]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderTextColor 3

[Function]

```
public getHeaderTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the HeaderTextColor [▶ 222] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setHeaderTextColor [▶ 1392]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderTextColor 4

[Function]

```
public setHeaderTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the HeaderTextColor [▶ 222] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderTextColor [▶ 1391]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderBackgroundColor 5

[Function]

```
public getHeaderBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the HeaderBackgroundColor [▶ 220] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setHeaderBackgroundColor [▶ 1393]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderBackgroundColor 6

[Function]

```
public setHeaderBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the HeaderBackgroundColor [▶ 220] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderBackgroundColor [▶ 1392]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getTableBorderColor 7

[Function]

```
public getTableBorderColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TableBorderColor [▶ 228] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTableBorderColor [▶ 1394]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setTableBorderColor
8

[Function]

```
public setTableBorderColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TableBorderColor [▶ 228] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTableBorderColor [▶ 1393]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersTextColor
9

[Function]

```
public getRowNumbersTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the RowNumbersTextColor [▶ 225] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowNumbersTextColor [▶ 1395]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersTextColor 10

[Function]

```
public setRowNumbersTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the RowNumbersTextColor [▶ 225] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersTextColor [▶ 1394]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getSrcColumn 11

[Function]

```
public getSrcColumn(): TcHmi.Controls.Beckhoff.TcHmiDatagrid.Column[];
```

This function returns the currently set value for the SrcColumn [▶ 227] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiDatagrid.Column [▶ 1427][]	The current value of the attribute.

See also

Attribute setter: setSrcColumn [▶ 1395]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setSrcColumn 12

[Function]

```
public setSrcColumn(newValue: TcHmi.Controls.Beckhoff.TcHmiDatagrid.Column[] | null): void;
```

This function sets a new value for the SrcColumn [▶ 227] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiDatagrid.Column [▶ 1427][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSrcColumn [▶ 1395]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getSrcData 13

[Function]

```
public getSrcData(): any[];
```

This function returns the currently set value for the SrcData [▶ 228] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
any [▶ 2107][ ]	The current value of the attribute.

See also

Attribute setter: setSrcData [▶ 1396]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setSrcData 14

[Function]

```
public setSrcData(newValue: any[] | null): void;
```

This function sets a new value for the SrcData [▶ 228] attribute.

Parameter

Name	Type	Description
newValue	any [▶ 2107][ ], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSrcData [▶ 1396]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getSelectedRowIndex
15

[Function]

```
public getSelectedRowIndex(): number | null;
```

This function returns the currently set value for the SelectedRowIndex [▶ 227] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSelectedRowIndex [▶ 1397]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setSelectedRowIndex
16

[Function]

```
public setSelectedRowIndex(newValue: number | null): void;
```

This function sets a new value for the SelectedRowIndex [▶ 227] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSelectedRowIndex [▶ 1397]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getSelectedColumnIndex 17

[Function]

```
public getSelectedColumnIndex(): number | null;
```

This function returns the currently set value for the SelectedColumnIndex [▶ 226] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSelectedColumnIndex [▶ 1398]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setSelectedColumnIndex 18

[Function]

```
public setSelectedColumnIndex(newValue: number | null): void;
```

This function sets a new value for the SelectedColumnIndex [▶ 226] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSelectedColumnIndex [▶ 1398]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowHeight
19

[Function]

```
public getRowHeight(): number;
```

This function returns the currently set value for the RowHeight [▶ 222] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowHeight [▶ 1399]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowHeight
20

[Function]

```
public setRowHeight(newValue: number | null): void;
```

This function sets a new value for the RowHeight [▶ 222] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowHeight [▶ 1399]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowHeightUnit 21

[Function]

```
public getRowHeightUnit(): 'px';
```

This function returns the currently set value for the RowHeightUnit [▶ 222] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getGridFontStyle 22

[Function]

```
public getGridFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the GridFontStyle [▶ 219] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setGridFontStyle [▶ 1400]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setGridFontStyle 23

[Function]

```
public setGridFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the GridFontStyle [▶ 219] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridFontStyle [▶ 1400]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getGridFontSizeUnit 24

[Function]

```
public getGridFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the GridFontSizeUnit [▶ 219] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setGridFontSizeUnit [▶ 1401]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setGridFontSizeUnit 25

[Function]

```
public setGridFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the GridFontSizeUnit [▶ 219] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridFontSizeUnit [▶ 1401]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getGridFontSize
26

[Function]

```
public getGridFontSize(): number;
```

This function returns the currently set value for the GridFontSize [▶ 218] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridFontSize [▶ 1402]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setGridFontSize
27

[Function]

```
public setGridFontSize(newValue: number | null): void;
```

This function sets a new value for the GridFontSize [▶ 218] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridFontSize [▶ 1402]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getGridFontFamily
28

[Function]

```
public getGridFontFamily(): string | null;
```

This function returns the currently set value for the GridFontFamily [▶ 218] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridFontFamily [▶ 1403]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setGridFontFamily
29

[Function]

```
public setGridFontFamily(newValue: string | null): void;
```

This function sets a new value for the GridFontFamily [▶ 218] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridFontFamily [▶ 1403]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getCellContentPadding
30

[Function]

```
public getCellContentPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the CellContentPadding [▶ 218] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setCellContentPadding [▶ 1404]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setCellContentPadding
31

[Function]

```
public setCellContentPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the CellContentPadding [▶ 218] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getCellContentPadding [▶ 1404]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getGridFontWeight 32

[Function]

```
public getGridFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the GridFontWeight [▶ 219] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setGridFontWeight [▶ 1405]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setGridFontWeight 33

[Function]

```
public setGridFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the GridFontWeight [▶ 219] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridFontWeight [▶ 1405]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderHeight
34

[Function]

```
public getHeaderHeight(): number;
```

This function returns the currently set value for the HeaderHeight [▶ 221] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setHeaderHeight [▶ 1406]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderHeight
35

[Function]

```
public setHeaderHeight(newValue: number | null): void;
```

This function sets a new value for the HeaderHeight [▶ 221] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderHeight [▶ 1406]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderHeightUnit
36

[Function]

```
public getHeaderHeightUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the HeaderHeightUnit [▶ 222] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setHeaderHeightUnit [▶ 1407]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderHeightUnit
37

[Function]

```
public setHeaderHeightUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the HeaderHeightUnit [▶ 222] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderHeightUnit [▶ 1406]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderFontWeight
38

[Function]

```
public getHeaderFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the HeaderFontWeight [▶ 221] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setHeaderFontWeight [▶ 1408]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderFontWeight
39

[Function]

```
public setHeaderFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the HeaderFontWeight [▶ 221] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderFontWeight [▶ 1407]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderFontSizeUnit
40

[Function]

```
public getHeaderFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the HeaderFontSizeUnit [▶ 221] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setHeaderFontSizeUnit [▶ 1409]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderFontSizeUnit
41

[Function]

```
public setHeaderFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the HeaderFontSizeUnit [▶ 221] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderFontSizeUnit [▶ 1408]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderFontSize
42

[Function]

```
public getHeaderFontSize(): number;
```

This function returns the currently set value for the HeaderFontSize [▶ 220] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setHeaderFontSize [▶ 1410]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderFontSize 43

[Function]

```
public setHeaderFontSize(newValue: number | null): void;
```

This function sets a new value for the HeaderFontSize [▶ 220] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderFontSize [▶ 1409]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderFontFamily 44

[Function]

```
public getHeaderFontFamily(): string | null;
```

This function returns the currently set value for the HeaderFontFamily [▶ 220] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setHeaderFontFamily [▶ 1411]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderFontFamily
45

[Function]

```
public setHeaderFontFamily(newValue: string | null): void;
```

This function sets a new value for the HeaderFontFamily [▶ 220] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderFontFamily [▶ 1410]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderCellPadding
46

[Function]

```
public getHeaderCellPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the HeaderCellPadding [▶ 220] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setHeaderCellPadding [▶ 1412]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderCellPadding 47

[Function]

```
public setHeaderCellPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the HeaderCellPadding [▶ 220] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderCellPadding [▶ 1411]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getHeaderFontStyle 48

[Function]

```
public getHeaderFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the HeaderFontStyle [▶ 221] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setHeaderFontStyle [▶ 1412]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setHeaderFontStyle 49

[Function]

```
public setHeaderFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the HeaderFontStyle [▶ 221] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeaderFontStyle [▶ 1412]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getShowRowNumbers 50

[Function]

```
public getShowRowNumbers(): boolean;
```

This function returns the currently set value for the ShowRowNumbers [▶ 227] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowRowNumbers [▶ 1413]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setShowRowNumbers 51

[Function]

```
public setShowRowNumbers(newValue: boolean | null): void;
```

This function sets a new value for the ShowRowNumbers [▶ 227] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowRowNumbers [▶ 1413]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersResizable 52

[Function]

```
public getRowNumbersResizable(): boolean;
```

This function returns the currently set value for the RowNumbersResizable [▶ 225] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setRowNumbersResizable [▶ 1414]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersResizable 53

[Function]

```
public setRowNumbersResizable(newValue: boolean | null): void;
```

This function sets a new value for the RowNumbersResizable [▶ 225] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersResizable [▶ 1414]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersFontWeight 54

[Function]

```
public getRowNumbersFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the RowNumbersFontWeight [▶ 224] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setRowNumbersFontWeight [▶ 1415]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersFontWeight 55

[Function]

```
public setRowNumbersFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the RowNumbersFontWeight [▶ 224] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersFontWeight [▶ 1415]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersWidth
56

[Function]

```
public getRowNumbersWidth(): number;
```

This function returns the currently set value for the RowNumbersWidth [▶ 226] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowNumbersWidth [▶ 1416]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersWidth
57

[Function]

```
public setRowNumbersWidth(newValue: number | null): void;
```

This function sets a new value for the RowNumbersWidth [▶ 226] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersWidth [▶ 1416]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersWidthUnit 58

[Function]

```
public getRowNumbersWidthUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the RowNumbersWidthUnit [▶ 226] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setRowNumbersWidthUnit [▶ 1417]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersWidthUnit 59

[Function]

```
public setRowNumbersWidthUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the RowNumbersWidthUnit [▶ 226] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersWidthUnit [▶ 1417]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersHorizontalAlignment 60

[Function]

```
public getRowNumbersHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the RowNumbersHorizontalAlignment [▶ 225] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setRowNumbersHorizontalAlignment [▶ 1418]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersHorizontalAlignment 61

[Function]

```
public setRowNumbersHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the RowNumbersHorizontalAlignment [▶ 225] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersHorizontalAlignment [▶ 1418]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersVerticalAlignment 62

[Function]

```
public getRowNumbersVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the RowNumbersVerticalAlignment [▶ 225] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setRowNumbersVerticalAlignment [▶ 1419]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersVerticalAlignment 63

[Function]

```
public setRowNumbersVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the RowNumbersVerticalAlignment [▶ 225] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersVerticalAlignment [▶ 1418]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersCellPadding 64

[Function]

```
public getRowNumbersCellPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the RowNumbersCellPadding [▶ 223] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowNumbersCellPadding [▶ 1420]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersCellPadding 65

[Function]

```
public setRowNumbersCellPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the RowNumbersCellPadding [▶ 223] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersCellPadding [▶ 1419]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersFontFamily 66

[Function]

```
public getRowNumbersFontFamily(): string | null;
```

This function returns the currently set value for the RowNumbersFontFamily [▶ 223] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowNumbersFontFamily [▶ 1421]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersFontFamily 67

[Function]

```
public setRowNumbersFontFamily(newValue: string | null): void;
```

This function sets a new value for the RowNumbersFontFamily [▶ 223] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersFontFamily [▶ 1420]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersFontSize 68

[Function]

```
public getRowNumbersFontSize(): number;
```

This function returns the currently set value for the RowNumbersFontSize [▶ 223] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRowNumbersFontSize [▶ 1422]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersFontSize 69

[Function]

```
public setRowNumbersFontSize(newValue: number | null): void;
```

This function sets a new value for the RowNumbersFontSize [▶ 223] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersFontSize [▶ 1421]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersFontSizeUnit 70

[Function]

```
public getRowNumbersFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the RowNumbersFontSizeUnit [▶ 224] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setRowNumbersFontSizeUnit [▶ 1423]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersFontSizeUnit 71

[Function]

```
public setRowNumbersFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the RowNumbersFontSizeUnit [▶ 224] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersFontSizeUnit [▶ 1422]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getRowNumbersFontStyle 72

[Function]

```
public getRowNumbersFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the RowNumbersFontStyle [▶ 224] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setRowNumbersFontStyle [▶ 1424]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setRowNumbersFontStyle
73

[Function]

```
public setRowNumbersFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the RowNumbersFontStyle [▶ 224] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRowNumbersFontStyle [▶ 1423]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getTableBorderWidth
74

[Function]

```
public getTableBorderWidth(): number;
```

This function returns the currently set value for the TableBorderWidth [▶ 228] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTableBorderWidth [▶ 1424]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setTableBorderWidth
75

[Function]

```
public setTableBorderWidth(newValue: number | null): void;
```

This function sets a new value for the TableBorderWidth [▶ 228] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTableBorderWidth [▶ 1424]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getTableBorderWidthUnit
76

[Function]

```
public getTableBorderWidthUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TableBorderWidthUnit [▶ 219] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTableBorderWidthUnit [▶ 1425]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setTableBorderWidthUnit
77

[Function]

```
public setTableBorderWidthUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TableBorderWidthUnit [▶ 219] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTableBorderWidthUnit [▶ 1425]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. getTableBorderStyle
78

[Function]

```
public getTableBorderStyle(): TcHmi.BorderStyle | null;
```

This function returns the currently set value for the TableBorderStyle [▶ 228] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.BorderStyle [▶ 2081], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTableBorderStyle [▶ 1426]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. setTableBorderStyle
79

[Function]

```
public setTableBorderStyle(newValue: TcHmi.BorderStyle | null): void;
```

This function sets a new value for the TableBorderStyle [▶ 228] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.BorderStyle [▶ 2081], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTableBorderStyle [▶ 1426]

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.8

15.1.1.2.2.5. Column 80

[Interface]

```
export interface Column {
    control: 'TextBlock' | 'TextBox' | 'CheckBox' | 'ComboBox' | 'Image';
    name: string;
    label: string;    editable?: boolean;
    dataDefinitions?: TcHmi.Controls.Beckhoff.DataGrid.DataDefinition[];
    resize?: boolean;
    cellBackground?: TcHmi.SolidColor;
    textColor?: TcHmi.SolidColor;
    width: number;
    widthUnit: TcHmi.DimensionUnit;
    minWidth?: number;
    minWidthUnit?: TcHmi.DimensionUnit;
    horizontalAlignment?: TcHmi.HorizontalAlignment;
    verticalAlignment?: TcHmi.VerticalAlignment;
    format?: TcHmi.IFunction;
}
```

Properties

Name	Type	Description
control	'TextBlock', 'TextBox', 'CheckBox', 'ComboBox', 'Image'	Display type
name	string [▶ 2108]	Name of the column
label	string [▶ 2108]	Column legend
editable [optional]	boolean [▶ 2107]	Determines whether the user can change this value. Has no effect if the control is 'TextBlock' or 'Image'.
dataDefinitions [optional]	TcHmi.Controls.Beckhoff.DataGrid.DataDefinition [▶ 1428]	Datadefinition if the control is set to 'Combobox' or 'Image'.
resize [optional]	boolean [▶ 2107]	Determines whether the user can change the width of this column.
cellBackground	TcHmi.SolidColor [▶ 2084]	Background color
textColor	TcHmi.SolidColor [▶ 2084]	Text color
width	number [▶ 2108]	Numerical value of the width
widthUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the width
minWidth	number [▶ 2108]	Numerical value of the minimum width
minWidthUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the minimum width
horizontalAlignment [optional]	TcHmi.HorizontalAlignment [▶ 2089]	Horizontal alignment
verticalAlignment [optional]	TcHmi.VerticalAlignment [▶ 2089]	Vertical alignment
format [optional]	TcHmi.IFunction [▶ 2095]	Formatting function

Note: Available since version 1.8

15.1.1.2.2.5. DataDefinition 81

[Interface]

```
export interface DataDefinition {
    value: any;
    text: string;
    source?: string;
    width?: number;
    height?: number;
}
```

Data definition if the column shows 'Image' or 'Combobox'.

Properties

Name	Type	Description
value	any [▶ 2107]	Value
text	string [▶ 2108]	Display text for a 'ComboBox', alternative display text for an 'Image'.
source [optional]	string [▶ 2108]	Web address of an image (only for 'Image')
width [optional]	number [▶ 2108]	Pixel width of the image (only for 'Image')
height [optional]	number [▶ 2108]	Pixel height of the image (only for 'Image')

Note: Available since version 1.8

15.1.1.2.2.5. getSrcDataHasChanged 82

[Function]

```
public getSrcDataHasChanged (): boolean;
```

This function returns the currently set value for the SrcDataHasChanged [▶ 228] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.10

15.1.1.2.2.5. getSelectedRowValue 83

[Function]

```
public getSelectedRowValue(): any;
```

This function returns the currently set value for the SelectedRowValue [▶ 227] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
any [▶ 2107]	The current value of the attribute.

See also

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.10

15.1.1.2.2.5. getSelectedCellValue 84

[Function]

```
public getSelectedCellValue(): any;
```

This function returns the currently set value for the SelectedCellValue [▶ 226] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
any [▶ 2107]	The current value of the attribute.

See also

Origin: TcHmiDatagrid [▶ 203]

Note: Available since version 1.10

15.1.1.2.2.6 TcHmiEllipse

[Class]

Ellipse

Attribute functions

Name	Origin	Description
getFillColor [▶ 1434]	TcHmiEllipse	Getter for the FillColor attribute.
setFillColor [▶ 1435]	TcHmiEllipse	Setter for the FillColor attribute.
getStrokeColor [▶ 1435]	TcHmiEllipse	Getter for the StrokeColor attribute.
setStrokeColor [▶ 1436]	TcHmiEllipse	Setter for the StrokeColor attribute.
getStrokeStyle [▶ 1438]	TcHmiEllipse	Getter for the StrokeStyle attribute.
setStrokeStyle [▶ 1438]	TcHmiEllipse	Setter for the StrokeStyle attribute.
getStrokeThickness [▶ 1436]	TcHmiEllipse	Getter for the StrokeThickness attribute.
setStrokeThickness [▶ 1437]	TcHmiEllipse	Setter for the StrokeThickness attribute.
getStrokeThicknessUnit [▶ 1437]	TcHmiEllipse	Getter for the StrokeThicknessUnit attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.

Name	Origin	Description
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.

Name	Origin	Description
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.

Name	Origin	Description
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Ellipse

See also

Reference [▶ 229] of this control

Note: Available since version 1.8

15.1.1.2.2.6. getFillColor
1

[Function]

```
public getFillColor(): TcHmi.Color | null;
```

This function returns the currently set value for the FillColor [▶ 234] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setFillColor [▶ 1435]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. setFillColor 2

[Function]

```
public setFillColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the FillColor [▶ 234] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getFillColor [▶ 1434]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. getStrokeColor 3

[Function]

```
public getStrokeColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the StrokeColor [▶ 234] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeColor [▶ 1436]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. setStrokeColor
4

[Function]

```
public setStrokeColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the StrokeColor [▶ 234] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeColor [▶ 1435]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. getStrokeThickness
5

[Function]

```
public getStrokeThickness(): number | null;
```

This function returns the currently set value for the StrokeThickness [▶ 235] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeThickness [▶ 1437]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. setStrokeThickness

[Function]

```
public setStrokeThickness(newValue: number | null): void;
```

This function sets a new value for the StrokeThickness [▶ 235] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeThickness [▶ 1436]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. getStrokeThicknessUnit

[Function]

```
public getStrokeThicknessUnit(): 'px';
```

This function returns the currently set value for the StrokeThicknessUnit [▶ 235] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. getStrokeStyle
8

[Function]

```
public getStrokeStyle(): string;
```

This function returns the currently set value for the StrokeStyle [▶ 234] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeStyle [▶ 1438]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.6. setStrokeStyle
9

[Function]

```
public setStrokeStyle(newValue: string | null): void;
```

This function sets a new value for the StrokeStyle [▶ 234] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeStyle [▶ 1438]

Origin: TcHmiEllipse [▶ 229]

Note: Available since version 1.8

15.1.1.2.2.7 TcHmiEventGrid

[Class]

Table for displaying events.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.

Name	Origin	Description
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getFilter [▶ 1445]	TcHmiEventGrid	Getter for the Filter attribute
setFilter [▶ 1445]	TcHmiEventGrid	Setter for the Filter attribute
getColumns [▶ 1445]	TcHmiEventGrid	Getter for the Columns attribute
setColumns [▶ 1446]	TcHmiEventGrid	Setter for the Columns attribute
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.

Name	Origin	Description
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.

Name	Origin	Description
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getMenuBarPosition [▶ 1446]	TcHmiEventGrid	Getter for the MenubarPosition attribute
setMenuBarPosition [▶ 1447]	TcHmiEventGrid	Setter for the MenubarPosition attribute
getShowMenuBar [▶ 1447]	TcHmiEventGrid	Getter for the ShowMenubar attribute
setShowMenuBar [▶ 1448]	TcHmiEventGrid	Setter for the ShowMenubar attribute
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

EventGrid

See also

Reference [▶ 235] of this control

Note: Available since version 1.10

15.1.1.2.2.7. getFilter 1

[Function]

```
public getFilter(): TcHmi.Filter | null;
```

This function returns the currently set value for the Filter [▶ 243] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Filter [▶ 2087], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setFilter [▶ 1445]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. setFilter 2

[Function]

```
public setFilter(newValue: TcHmi.Filter | null): void;
```

This function sets a new value for the Filter [▶ 243] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Filter [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value

See also

Attribute getter: getFilter [▶ 1445]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. getColumns 3

[Function]

```
public getColumns(): TcHmiEventGrid.Column[] | undefined;
```

This function returns the currently set value for the Columns [▶ 243] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiEventGrid.Column [▶ 1448][], undefined [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setColumns [▶ 1446]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. setColumns
4

[Function]

```
public setColumns(newValue: TcHmiEventGrid.Column[] | null): void;
```

This function sets a new value for the Columns [▶ 243] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiEventGrid.Column [▶ 1448][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value

See also

Attribute getter: getColumns [▶ 1445]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. getMenuBarPosition
5

[Function]

```
public getMenubarPosition(): 'Top' | 'Right' | 'Bottom' | 'Left';
```

This function returns the currently set value for the MenuBarPosition [▶ 244] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Top' | 'Right' | 'Bottom' | 'Left'	The current value of the attribute.

See also

Attribute setter: setMenuBarPosition [▶ 1447]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. setMenuBarPosition
6

[Function]

```
public setMenubarPosition(newValue: 'Top' | 'Right' | 'Bottom' | 'Left' | null): void;
```

This function sets a new value for the MenuBarPosition [▶ 244] attribute.

Parameter

Name	Type	Description
newValue	'Top' | 'Right' | 'Bottom' | 'Left', null	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value

See also

Attribute getter: getMenuBarPosition [▶ 1446]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. getShowMenuBar
7

[Function]

```
public getShowMenubar(): boolean;
```

This function returns the currently set value for the ShowMenuBar [▶ 244] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean	The current value of the attribute.

See also

Attribute setter: setShowMenuBar [▶ 1448]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. setShowMenuBar
8

[Function]

```
public setShowMenubar(newValue: boolean | null): void;
```

This function sets a new value for the ShowMenuBar [▶ 244] attribute.

Parameter

Name	Type	Description
newValue	boolean, null	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value

See also

Attribute getter: getShowMenuBar [▶ 1447]

Origin: TcHmiEventGrid [▶ 1439]

Note: Available since version 1.10

15.1.1.2.2.7. Column
9

[Interface]

```
export interface Column {
    columnName: string;
}
```

Name	Type	Description
columnName	'severity' | 'type' | 'domain' | 'name' | 'text' | 'timeAndDateRaised' | 'timeRaised' | 'dateRaised' | 'timeAndDateConfirmed' | 'timeConfirmed' | 'dateConfirmed' | 'timeAndDateCleared' | 'timeCleared' | 'dateCleared'	Name of the column. The EventGrid provides a series of pre-configured columns.

Note: Available since version 1.10

15.1.1.2.2.8 TcHmiIFrame

[Class]

Nested browser context. Embeds another HTML page in the current visualization.

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.

Name	Origin	Description
getSrc [▶ 1454]	TcHmiIFrame	Getter for the Src attribute.
setSrc [▶ 1455]	TcHmiIFrame	Setter for the Src attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.

Name	Origin	Description
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

IFrame

See also

Reference [▶ 246] of this control

Note: Available since version 1.8

15.1.1.2.2.8. getSrc
1

[Function]

```
public getSrc(): string | null;
```

This function returns the currently set value for the Src [▶ 251] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSrc [▶ 1455]

Origin: TcHmiIFrame [▶ 246]

Note: Available since version 1.8

15.1.1.2.2.8. setSrc
2

[Function]

```
public setSrc(newValue: string | null): void;
```

This function sets a new value for the Src [▶ 251] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSrc [▶ 1454]

Origin: TcHmiIFrame [▶ 246]

Note: Available since version 1.8

15.1.1.2.2.9 TcHmiImage

[Class]

Image display

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1461]	TcHmiImage	Getter for the WidthMode attribute.
setWidthMode [▶ 1462]	TcHmiImage	Setter for the WidthMode attribute.
getHeightMode [▶ 1460]	TcHmiImage	Getter for the HeightMode attribute.
setHeightMode [▶ 1461]	TcHmiImage	Setter for the HeightMode attribute.

Name	Origin	Description
getSrc [▶ 1462]	TcHmiImage	Getter for the Src attribute.
setSrc [▶ 1463]	TcHmiImage	Setter for the Src attribute.
getAlt [▶ 1463]	TcHmiImage	Getter for the Alt attribute.
setAlt [▶ 1464]	TcHmiImage	Setter for the Alt attribute.
getOriginalWidth [▶ 1465]	TcHmiImage	Getter for the OriginalWidth attribute.
getOriginalHeight [▶ 1464]	TcHmiImage	Getter for the OriginalHeight attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.

Name	Origin	Description
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Image

See also

Reference [▶ 251] of this control

Note: Available since version 1.8

15.1.1.2.2.9. getHeightMode
1

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 256] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1461]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. setHeightMode
2

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 256] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1460]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. getWidthMode
3

[Function]

```
public getWidthMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the WidthMode [▶ 257] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setWidthMode [▶ 1462]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. setWidthMode
4

[Function]

```
public setWidthMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the WidthMode [▶ 257] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWidthMode [▶ 1461]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. getSrc
5

[Function]

```
public getSrc(): string | null;
```

This function returns the currently set value for the Src [▶ 256] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSrc [▶ 1463]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. setSrc
6

[Function]

```
public setSrc(newValue: string | null): void;
```

This function sets a new value for the Src [▶ 256] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSrc [▶ 1462]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. getAlt
7

[Function]

```
public getAlt(): string;
```

This function returns the currently set value for the Alt [▶ 256] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setAlt [▶ 1464]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. setAlt
8

[Function]

```
public setAlt(newValue: string | null): void;
```

This function sets a new value for the Alt [▶ 256] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getAlt [▶ 1463]

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. getOriginalHeight
9

[Function]

```
public getOriginalHeight(): number;
```

This function returns the currently set value for the OriginalHeight [▶ 257] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.9. getOriginalWidth
10

[Function]

```
public getOriginalWidth(): number;
```

This function returns the currently set value for the OriginalWidth [▶ 258] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiImage [▶ 251]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiKeyboard
0

[Class]

On-screen keyboard

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getAdditionalKeyBackdropColor [▶ 1475]	TcHmiKeyboard	Getter for the AdditionalKeyBackdropColor attribute.
setAdditionalKeyBackdropColor [▶ 1475]	TcHmiKeyboard	Setter for the AdditionalKeyBackdropColor attribute.
getLayoutFile [▶ 1479]	TcHmiKeyboard	Getter for the LayoutFile attribute.
setLayoutFile [▶ 1480]	TcHmiKeyboard	Setter for the LayoutFile attribute.
getLayout [▶ 1480]	TcHmiKeyboard	Getter for the Layout attribute.
setLayout [▶ 1481]	TcHmiKeyboard	Setter for the Value attribute.
getKeyBackgroundColor [▶ 1474]	TcHmiKeyboard	Getter for the KeyBackgroundColor attribute.
setKeyBackgroundColor [▶ 1474]	TcHmiKeyboard	Setter for the KeyBackgroundColor attribute.
getLabelColor [▶ 1473]	TcHmiKeyboard	Getter for the LabelColor attribute.
setLabelColor [▶ 1473]	TcHmiKeyboard	Setter for the LabelColor attribute.
getScaleMode [▶ 1481]	TcHmiKeyboard	Getter for the ScaleMode attribute.
setScaleMode [▶ 1481]	TcHmiKeyboard	Setter for the ScaleMode attribute.
getRepetitionDelay [▶ 1485]	TcHmiKeyboard	Getter for the RepetitionDelay attribute.
setRepetitionDelay [▶ 1485]	TcHmiKeyboard	Setter for the RepetitionDelay attribute.
getFixedTarget [▶ 1482]	TcHmiKeyboard	Getter for the FixedTarget attribute.
setFixedTarget [▶ 1482]	TcHmiKeyboard	Setter for the FixedTarget attribute.
getLongpressDelay [▶ 1484]	TcHmiKeyboard	Getter for the LongpressDelay attribute.
setLongpressDelay [▶ 1484]	TcHmiKeyboard	Setter for the LongpressDelay attribute.
getReleaseStickyKeysManually [▶ 1483]	TcHmiKeyboard	Getter for the ReleaseStickyKeysManually attribute.
setReleaseStickyKeysManually [▶ 1483]	TcHmiKeyboard	Setter for the ReleaseStickyKeysManually attribute.

Name	Origin	Description
getAdditionalKeyBackdropPadding [▶ 1478]	TcHmiKeyboard	Getter for the AdditionalKeyBackdropPadding attribute.
setAdditionalKeyBackdropPadding [▶ 1479]	TcHmiKeyboard	Setter for the AdditionalKeyBackdropPadding attribute.
getKeyboardPadding [▶ 1476]	TcHmiKeyboard	Getter for the KeyboardPadding attribute.
setKeyboardPadding [▶ 1477]	TcHmiKeyboard	Setter for the KeyboardPadding attribute.
getKeyPadding [▶ 1477]	TcHmiKeyboard	Getter for the KeyPadding attribute.
setKeyPadding [▶ 1478]	TcHmiKeyboard	Setter for the KeyPadding attribute.
getLabelStrokeColor [▶ 1472]	TcHmiKeyboard	Getter for the LabelStrokeColor attribute.
setLabelStrokeColor [▶ 1472]	TcHmiKeyboard	Setter for the LabelStrokeColor attribute.
getLabelStrokeThickness [▶ 1486]	TcHmiKeyboard	Getter for the LabelStrokeThickness attribute.
setLabelStrokeThickness [▶ 1486]	TcHmiKeyboard	Setter for the LabelStrokeThickness attribute.
getLabelFillColor [▶ 1475]	TcHmiKeyboard	Getter for the LabelFillColor attribute.
setLabelFillColor [▶ 1476]	TcHmiKeyboard	Setter for the LabelFillColor attribute.
getLabelFontFamily [▶ 1489]	TcHmiKeyboard	Getter for the LabelFontFamily attribute.
setLabelFontFamily [▶ 1490]	TcHmiKeyboard	Setter for the LabelFontFamily attribute.
getLabelFontSize [▶ 1487]	TcHmiKeyboard	Getter for the LabelFontSize attribute.
setLabelFontSize [▶ 1488]	TcHmiKeyboard	Setter for the LabelFontSize attribute.
getLabelFontSizeUnit [▶ 1488]	TcHmiKeyboard	Getter for the LabelFontSizeUnit attribute.
setLabelFontSizeUnit [▶ 1489]	TcHmiKeyboard	Setter for the LabelFontSizeUnit attribute.
getLabelFontStyle [▶ 1490]	TcHmiKeyboard	Getter for the LabelFontStyle attribute.
setLabelFontStyle [▶ 1491]	TcHmiKeyboard	Setter for the LabelFontStyle attribute.
getLabelFontWeight [▶ 1487]	TcHmiKeyboard	Getter for the LabelFontWeight attribute.
setLabelFontWeight [▶ 1487]	TcHmiKeyboard	Setter for the LabelFontWeight attribute.
getPressedKeys [▶ 1491]	TcHmiKeyboard	Getter for the PressedKeys attribute
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.

Name	Origin	Description
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.

Name	Origin	Description
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.

Name	Origin	Description
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Keyboard

See also

Reference [▶ 258] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getLabelStrokeColor
0.1

[Function]

```
public getLabelStrokeColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the LabelStrokeColor [▶ 268] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelStrokeColor [▶ 1472]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelStrokeColor
0.2

[Function]

```
public setLabelStrokeColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the LabelStrokeColor [▶ 268] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelStrokeColor [▶ 1472]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelColor
0.3

[Function]

```
public getLabelColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the LabelColor [▶ 268] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelColor [▶ 1473]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelColor
0.4

[Function]

```
public setLabelColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the LabelColor [▶ 268] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelColor [▶ 1473]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getKeyBackgroundColor
0.5

[Function]

```
public getKeyBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the KeyBackgroundColor [▶ 267] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKeyBackgroundColor [▶ 1474]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setKeyBackgroundColor
0.6

[Function]

```
public setKeyBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the KeyBackgroundColor [▶ 267] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKeyBackgroundColor [▶ 1474]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getAdditionalKeyBackdropColor
0.7

[Function]

```
public getAdditionalKeyBackdropColor(): TcHmi.Color | null;
```

This function returns the currently set value for the AdditionalKeyBackdropColor [▶ 265] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setAdditionalKeyBackdropColor [▶ 1475]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setAdditionalKeyBackdropColor
0.8

[Function]

```
public setAdditionalKeyBackdropColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the AdditionalKeyBackdropColor [▶ 265] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getAdditionalKeyBackdropColor [▶ 1475]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFillColor
0.9

[Function]

```
public getLabelFillColor(): TcHmi.Color | null;
```

This function returns the currently set value for the LabelFillColor [▶ 268] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFillColor [▶ 1476]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFillColor
0.10

[Function]

```
public setLabelFillColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the LabelFillColor [▶ 268] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFillColor [▶ 1475]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getKeyboardPadding
0.11

[Function]

```
public getKeyboardPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the KeyboardPadding [▶ 268] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKeyboardPadding [▶ 1477]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setKeyboardPadding
0.12

[Function]

```
public setKeyboardPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the KeyboardPadding [▶ 268] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKeyboardPadding [▶ 1476]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getKeyPadding
0.13

[Function]

```
public getKeyPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the KeyPadding [▶ 267] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKeyPadding [▶ 1478]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setKeyPadding
0.14

[Function]

```
public setKeyPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the KeyPadding [▶ 267] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKeyPadding [▶ 1477]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getAdditionalKeyBackdropPadding
0.15

[Function]

```
public getAdditionalKeyBackdropPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the AdditionalKeyBackdropPadding [▶ 265] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setAdditionalKeyBackdropPadding [▶ 1479]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setAdditionalKeyBackdropPadding
0.16

[Function]

```
public setAdditionalKeyBackdropPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the AdditionalKeyBackdropPadding [▶ 265] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getAdditionalKeyBackdropPadding [▶ 1478]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLayoutFile
0.17

[Function]

```
public getLayoutFile(): string;
```

This function returns the current Layout [▶ 269] attribute value.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLayoutFile [▶ 1480]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.10

15.1.1.2.2.1 setLayoutFile
0.18

[Function]

```
public setLayout(newValue: string | null): void;
```

This function sets a new value for the LayoutFile [▶ 269] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLayoutFile [▶ 1479]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.10

15.1.1.2.2.1 getLayout
0.19

[Function]

```
public getLayout(): string;
```

This function returns the current Layout [▶ 269] attribute value.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLayout [▶ 1481]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8
Was LayoutPath before version 1.10

15.1.1.2.2.1 setLayout
0.20

[Function]

```
public setLayout(newValue: string | null): void;
```

This function sets a new value for the Layout [▶ 269] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLayout [▶ 1480]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8
Was LayoutPath before version 1.10

15.1.1.2.2.1 getScaleMode
0.21

[Function]

```
public getScaleMode(): TcHmi.ScaleModeString;
```

This function returns the currently set value for the ScaleMode [▶ 270] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.ScaleModeString [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setScaleMode [▶ 1481]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setScaleMode
0.22

[Function]

```
public setScaleMode(newValue: TcHmi.ScaleModeString | null): void;
```

This function sets a new value for the ScaleMode [▶ 270] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.ScaleModeString [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getScaleMode [▶ 1481]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getFixedTarget
0.23

[Function]

```
public getFixedTarget(): string;
```

This function returns the currently set value for the FixedTarget [▶ 265] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setFixedTarget [▶ 1482]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setFixedTarget
0.24

[Function]

```
public setFixedTarget(newValue: string | null): void;
```

This function sets a new value for the FixedTarget [▶ 265] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getFixedTarget [▶ 1482]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getReleaseStickyKeysManually
0.25

[Function]

```
public getReleaseStickyKeysManually(): boolean;
```

This function returns the currently set value for the ReleaseStickyKeysManually [▶ 270] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setReleaseStickyKeysManually [▶ 1483]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setReleaseStickyKeysManually
0.26

[Function]

```
public setReleaseStickyKeysManually(newValue: boolean | null): void;
```

This function sets a new value for the ReleaseStickyKeysManually [▶ 270] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getReleaseStickyKeysManually [▶ 1483]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLongpressDelay
0.27

[Function]

```
public getLongpressDelay(): number;
```

This function returns the currently set value for the LongpressDelay [▶ 269] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLongpressDelay [▶ 1484]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLongpressDelay
0.28

[Function]

```
public setLongpressDelay(newValue: number | null): void;
```

This function sets a new value for the LongpressDelay [▶ 269] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLongpressDelay [▶ 1484]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getRepetitionDelay
0.29

[Function]

```
public getRepetitionDelay(): number;
```

This function returns the currently set value for the RepetitionDelay [▶ 270] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRepetitionDelay [▶ 1485]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setRepetitionDelay
0.30

[Function]

```
public setRepetitionDelay(newValue: number | null): void;
```

This function sets a new value for the RepetitionDelay [▶ 270] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRepetitionDelay [▶ 1485]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelStrokeThickness
0.31

[Function]

```
public getLabelStrokeThickness(): number | null;
```

This function returns the currently set value for the LabelStrokeThickness [▶ 267] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelStrokeThickness [▶ 1486]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelStrokeThickness
0.32

[Function]

```
public setLabelStrokeThickness(newValue: number | null): void;
```

This function sets a new value for the LabelStrokeThickness [▶ 267] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelStrokeThickness [▶ 1486]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontWeight
0.33

[Function]

```
public getLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the LabelFontWeight [▶ 266] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setLabelFontWeight [▶ 1487]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontWeight
0.34

[Function]

```
public setLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the LabelFontWeight [▶ 266] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontWeight [▶ 1487]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontSize
0.35

[Function]

```
public getLabelFontSize(): number;
```

This function returns the currently set value for the LabelFontSize [▶ 266] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFontSize [▶ 1488]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontSize
0.36

[Function]

```
public setLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the LabelFontSize [▶ 266] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontSize [▶ 1487]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontSizeUnit
0.37

[Function]

```
public getLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the LabelFontSizeUnit [▶ 266] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setLabelFontSizeUnit [▶ 1489]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontSizeUnit
0.38

[Function]

```
public setLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the LabelFontSizeUnit [▶ 266] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontSizeUnit [▶ 1488]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontFamily
0.39

[Function]

```
public getLabelFontFamily(): string | null;
```

This function returns the currently set value for the LabelFontFamily [▶ 265] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFontFamily [▶ 1490]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontFamily
0.40

[Function]

```
public setLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the LabelFontFamily [▶ 265] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontFamily [▶ 1489]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontStyle
0.41

[Function]

```
public getLabelFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the LabelFontStyle [▶ 266] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setLabelFontStyle [▶ 1491]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontStyle
0.42

[Function]

```
public setLabelFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the LabelFontStyle [▶ 266] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontStyle [▶ 1490]

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.8

15.1.1.2.2.1 getPressedKeys
0.43

[Function]

```
public getPressedKeys(): TcHmiKeyboard.SimpleKey[];
```

This function returns the currently set value for the PressedKeys [▶ 271] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmiKeyboard.SimpleKey [▶ 1492] []	The current value of the attribute.

See also

Origin: TcHmiKeyboard [▶ 258]

Note: Available since version 1.10

15.1.1.2.2.1 SimpleKey
0.44

[Interface]

```
export interface SimpleKey {
    code: string;
    key: string;
    unmodifiedKey: string;
    location: string;
}
```

15.1.1.2.2.1 TcHmiLine
1

[Class]

Straight line

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.

Name	Origin	Description
getX1 [▶ 1500]	TcHmiLine	Getter for X1 attribute.
setX1 [▶ 1500]	TcHmiLine	Setter for X1 attribute.
getX1Unit [▶ 1501]	TcHmiLine	Getter for the X1Unit attribute.
getX2 [▶ 1501]	TcHmiLine	Getter for X2 attribute.
setX2 [▶ 1502]	TcHmiLine	Setter for X2 attribute.
getX2Unit [▶ 1502]	TcHmiLine	Getter for the X2Unit attribute.
getY1 [▶ 1499]	TcHmiLine	Getter for Y1 attribute.
setY1 [▶ 1499]	TcHmiLine	Setter for Y1 attribute.
getY1Unit [▶ 1500]	TcHmiLine	Getter for the Y1Unit attribute.
getY2 [▶ 1503]	TcHmiLine	Getter for Y2 attribute.
setY2 [▶ 1503]	TcHmiLine	Setter for Y2 attribute.
getY2Unit [▶ 1504]	TcHmiLine	Getter for the Y2Unit attribute.
getScaleMode [▶ 1504]	TcHmiLine	Getter for the ScaleMode attribute.
setScaleMode [▶ 1504]	TcHmiLine	Setter for the ScaleMode attribute.
getStrokeColor [▶ 1498]	TcHmiLine	Getter for the StrokeColor attribute.
setStrokeColor [▶ 1498]	TcHmiLine	Setter for the StrokeColor attribute.
getStrokeStyle [▶ 1506]	TcHmiLine	Getter for the StrokeStyle attribute.
setStrokeStyle [▶ 1507]	TcHmiLine	Setter for the StrokeStyle attribute.
getStrokeThickness [▶ 1505]	TcHmiLine	Getter for the StrokeThickness attribute.
setStrokeThickness [▶ 1505]	TcHmiLine	Setter for the StrokeThickness attribute.
getStrokeThicknessUnit [▶ 1506]	TcHmiLine	Getter for the StrokeThicknessUnit attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.

Name	Origin	Description
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Line

See also

Reference [▶ 272] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeColor 1.1

[Function]

```
public getStrokeColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the StrokeColor [▶ 278] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeColor [▶ 1498]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeColor 1.2

[Function]

```
public setStrokeColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the StrokeColor [▶ 278] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeColor [▶ 1498]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getY1
1.3

[Function]

```
public getY1(): number;
```

This function returns the currently set value for the Y1 [▶ 280] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setY1 [▶ 1499]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setY1
1.4

[Function]

```
public setY1(newValue: number | null): void;
```

This function sets a new value for the Y1 [▶ 280] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getY1 [▶ 1499]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getY1Unit
1.5

[Function]

```
public getY1Unit(): 'px';
```

This function returns the currently set value for the Y1Unit [▶ 280] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getX1
1.6

[Function]

```
public getX1(): number;
```

This function returns the currently set value for the X1 [▶ 279] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setX1 [▶ 1500]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setX1
1.7

[Function]

```
public setX1(newValue: number | null): void;
```

This function sets a new value for the X1 [▶ 279] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getX1 [▶ 1500]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getX1Unit
1.8

[Function]

```
public getX1Unit(): 'px';
```

This function returns the currently set value for the X1Unit [▶ 279] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getX2
1.9

[Function]

```
public getX2(): number;
```

This function returns the currently set value for the X2 [▶ 280] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setX2 [▶ 1502]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setX2
1.10

[Function]

```
public setX2(newValue: number | null): void;
```

This function sets a new value for the X2 [▶ 280] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getX2 [▶ 1501]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getX2Unit
1.11

[Function]

```
public getX2Unit(): 'px';
```

This function returns the currently set value for the X2Unit [▶ 280] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getY2
1.12

[Function]

```
public getY2(): number;
```

This function returns the currently set value for the Y2 [▶ 281] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setY2 [▶ 1503]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setY2
1.13

[Function]

```
public setY2(newValue: number | null): void;
```

This function sets a new value for the Y2 [▶ 281] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getY2 [▶ 1503]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getY2Unit
1.14

[Function]

```
public getY2Unit(): 'px';
```

This function returns the currently set value for the Y2Unit [▶ 281] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getScaleMode
1.15

[Function]

```
public getScaleMode(): TcHmi.ScaleModeString;
```

This function returns the currently set value for the ScaleMode [▶ 278] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.ScaleModeString [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setScaleMode [▶ 1504]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setScaleMode
1.16

[Function]

```
public setScaleMode(newValue: TcHmi.ScaleModeString | null): void;
```

This function sets a new value for the ScaleMode [▶ 278] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.ScaleModeString [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getScaleMode [▶ 1504]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeThickness 1.17

[Function]

```
public getStrokeThickness(): number | null;
```

This function returns the currently set value for the StrokeThickness [▶ 279] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeThickness [▶ 1505]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeThickness 1.18

[Function]

```
public setStrokeThickness(newValue: number | null): void;
```

This function sets a new value for the StrokeThickness [▶ 279] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeThickness [▶ 1505]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeThicknessUnit
1.19

[Function]

```
public getStrokeThicknessUnit(): 'px';
```

This function returns the currently set value for the StrokeThicknessUnit [▶ 279] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeStyle
1.20

[Function]

```
public getStrokeStyle(): string;
```

This function returns the currently set value for the StrokeStyle [▶ 278] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeStyle [▶ 1507]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeStyle
1.21

[Function]

```
public setStrokeStyle(newValue: string | null): void;
```

This function sets a new value for the StrokeStyle [▶ 278] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeStyle [▶ 1506]

Origin: TcHmiLine [▶ 272]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiLinearGauge
2

[Class]

Linear display

Attribute functions

Name	Origin	Description
getProgressBackgroundColor [▶ 1515]	TcHmiLinearGauge	Getter for the ProgressBackgroundColor attribute.
setProgressBackgroundColor [▶ 1516]	TcHmiLinearGauge	Setter for the ProgressBackgroundColor attribute.
getProgressForegroundColor [▶ 1516]	TcHmiLinearGauge	Getter for the ProgressForegroundColor attribute.
setProgressForegroundColor [▶ 1517]	TcHmiLinearGauge	Setter for the ProgressForegroundColor attribute.
getTickColor [▶ 1518]	TcHmiLinearGauge	Getter for the TickColor attribute.
setTickColor [▶ 1518]	TcHmiLinearGauge	Setter for the TickColor attribute.
getLabelColor [▶ 1521]	TcHmiLinearGauge	Getter for the LabelColor attribute.
setLabelColor [▶ 1521]	TcHmiLinearGauge	Setter for the LabelColor attribute.
getKnobColor [▶ 1520]	TcHmiLinearGauge	Getter for the KnobColor attribute.
setKnobColor [▶ 1520]	TcHmiLinearGauge	Setter for the KnobColor attribute.
getKnobArrowsColor [▶ 1517]	TcHmiLinearGauge	Getter for the KnobArrowsColor attribute.
setKnobArrowsColor [▶ 1517]	TcHmiLinearGauge	Setter for the KnobArrowsColor attribute.
getUnitColor [▶ 1522]	TcHmiLinearGauge	Getter for the UnitColor attribute.
setUnitColor [▶ 1522]	TcHmiLinearGauge	Setter for the UnitColor attribute.
getValueColor [▶ 1519]	TcHmiLinearGauge	Getter for the ValueColor attribute.
setValueColor [▶ 1519]	TcHmiLinearGauge	Setter for the ValueColor attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.

Name	Origin	Description
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.

Name	Origin	Description
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getValue [▶ 1524]	TcHmiLinearGauge	Getter for the Value attribute.
setValue [▶ 1525]	TcHmiLinearGauge	Setter for the Value attribute.
getValueFontFamily [▶ 1536]	TcHmiLinearGauge	Getter for the ValueFontFamily attribute.
setValueFontFamily [▶ 1536]	TcHmiLinearGauge	Setter for the ValueFontFamily attribute.
getValueFontSize [▶ 1535]	TcHmiLinearGauge	Getter for the ValueFontSize attribute.
setValueFontSize [▶ 1535]	TcHmiLinearGauge	Setter for the ValueFontSize attribute.
getValueFontSizeUnit [▶ 1534]	TcHmiLinearGauge	Getter for the ValueFontSizeUnit attribute.
setValueFontSizeUnit [▶ 1534]	TcHmiLinearGauge	Setter for the ValueFontSizeUnit attribute.
getValueFontStyle [▶ 1537]	TcHmiLinearGauge	Getter for the ValueFontStyle attribute.
setValueFontStyle [▶ 1538]	TcHmiLinearGauge	Setter for the ValueFontStyle attribute.
getValueFontWeight [▶ 1532]	TcHmiLinearGauge	Getter for the ValueFontWeight attribute.
setValueFontWeight [▶ 1532]	TcHmiLinearGauge	Setter for the ValueFontWeight attribute.
getValueFormat [▶ 1533]	TcHmiLinearGauge	Getter for the ValueFormat attribute.
setValueFormat [▶ 1533]	TcHmiLinearGauge	Setter for the ValueFormat attribute.

Name	Origin	Description
getUnit [▶ 1549]	TcHmiLinearGauge	Getter for the Unit attribute.
setUnit [▶ 1549]	TcHmiLinearGauge	Setter for the Unit attribute.
getUnitFontFamily [▶ 1553]	TcHmiLinearGauge	Getter for the UnitFontFamily attribute.
setUnitFontFamily [▶ 1554]	TcHmiLinearGauge	Setter for the UnitFontFamily attribute.
getUnitFontSize [▶ 1550]	TcHmiLinearGauge	Getter for the UnitFontSize attribute.
setUnitFontSize [▶ 1551]	TcHmiLinearGauge	Setter for the UnitFontSize attribute.
getUnitFontSizeUnit [▶ 1554]	TcHmiLinearGauge	Getter for the UnitFontSizeUnit attribute.
setUnitFontSizeUnit [▶ 1554]	TcHmiLinearGauge	Setter for the UnitFontSizeUnit attribute.
getUnitFontStyle [▶ 1551]	TcHmiLinearGauge	Getter for the UnitFontStyle attribute.
setUnitFontStyle [▶ 1552]	TcHmiLinearGauge	Setter for the UnitFontStyle attribute.
getUnitFontWeight [▶ 1552]	TcHmiLinearGauge	Getter for the UnitFontWeight attribute.
setUnitFontWeight [▶ 1553]	TcHmiLinearGauge	Setter for the UnitFontWeight attribute.
getUnitOrientation [▶ 1549]	TcHmiLinearGauge	Getter for the UnitOrientation attribute.
setUnitOrientation [▶ 1550]	TcHmiLinearGauge	Setter for the UnitOrientation attribute.
getBaseAnimationTime [▶ 1555]	TcHmiLinearGauge	Getter for the BaseAnimationTime attribute.
setBaseAnimationTime [▶ 1555]	TcHmiLinearGauge	Setter for the BaseAnimationTime attribute.
getEditable [▶ 1525]	TcHmiLinearGauge	Getter for the Editable attribute.
setEditable [▶ 1526]	TcHmiLinearGauge	Setter for the Editable attribute.
getStep [▶ 1528]	TcHmiLinearGauge	Getter for the Step attribute.
setStep [▶ 1529]	TcHmiLinearGauge	Setter for the Step attribute.
getRange [▶ 1530]	TcHmiLinearGauge	Getter for the Range attribute.
setRange [▶ 1530]	TcHmiLinearGauge	Setter for the Range attribute.
getPadding [▶ 1523]	TcHmiLinearGauge	Getter for the Padding attribute.
setPadding [▶ 1524]	TcHmiLinearGauge	Setter for the Padding attribute.
getClickAnywhereToEdit [▶ 1536]	TcHmiLinearGauge	Getter for the ClickAnywhereToEdit attribute.
setClickAnywhereToEdit [▶ 1537]	TcHmiLinearGauge	Setter for the ClickAnywhereToEdit attribute.
getStartFromZero [▶ 1531]	TcHmiLinearGauge	Getter for the StartFromZero attribute.
setStartFromZero [▶ 1531]	TcHmiLinearGauge	Setter for the StartFromZero attribute.
getShowTicks [▶ 1547]	TcHmiLinearGauge	Getter for the ShowTicks attribute.
setShowTicks [▶ 1547]	TcHmiLinearGauge	Setter for the ShowTicks attribute.
getTickDefinition [▶ 1548]	TcHmiLinearGauge	Getter for the TickDefinition attribute.

Name	Origin	Description
setTickDefinition [▶ 1548]	TcHmiLinearGauge	Setter for the TickDefinition attribute.
getOrientation [▶ 1523]	TcHmiLinearGauge	Getter for the Orientation attribute.
setOrientation [▶ 1523]	TcHmiLinearGauge	Setter for the Orientation attribute.
getShowLabels [▶ 1538]	TcHmiLinearGauge	Getter for the ShowLabels attribute.
setShowLabels [▶ 1539]	TcHmiLinearGauge	Setter for the ShowLabels attribute.
getLabelFontFamily [▶ 1542]	TcHmiLinearGauge	Getter for the LabelFontFamily attribute.
setLabelFontFamily [▶ 1542]	TcHmiLinearGauge	Setter for the LabelFontFamily attribute.
getLabelFontSize [▶ 1543]	TcHmiLinearGauge	Getter for the LabelFontSize attribute.
setLabelFontSize [▶ 1543]	TcHmiLinearGauge	Setter for the LabelFontSize attribute.
getLabelFontSizeUnit [▶ 1544]	TcHmiLinearGauge	Getter for the LabelFontSizeUnit attribute.
setLabelFontSizeUnit [▶ 1544]	TcHmiLinearGauge	Setter for the LabelFontSizeUnit attribute.
getLabelFontStyle [▶ 1545]	TcHmiLinearGauge	Getter for the LabelFontStyle attribute.
setLabelFontStyle [▶ 1545]	TcHmiLinearGauge	Setter for the LabelFontStyle attribute.
getLabelFontWeight [▶ 1541]	TcHmiLinearGauge	Getter for the LabelFontWeight attribute.
setLabelFontWeight [▶ 1542]	TcHmiLinearGauge	Setter for the LabelFontWeight attribute.
getLabelFormat [▶ 1546]	TcHmiLinearGauge	Getter for the LabelFormat attribute.
setLabelFormat [▶ 1546]	TcHmiLinearGauge	Setter for the LabelFormat attribute.
getLabelPosition [▶ 1540]	TcHmiLinearGauge	Getter for the LabelPosition attribute.
setLabelPosition [▶ 1541]	TcHmiLinearGauge	Setter for the LabelPosition attribute.
getLabelRange [▶ 1539]	TcHmiLinearGauge	Getter for the LabelRange attribute.
setLabelRange [▶ 1540]	TcHmiLinearGauge	Setter for the LabelRange attribute.
getMaxValue [▶ 1526]	TcHmiLinearGauge	Getter for the MaxValue attribute.
setMaxValue [▶ 1527]	TcHmiLinearGauge	Setter for the MaxValue attribute.
getMinValue [▶ 1527]	TcHmiLinearGauge	Getter for the MinValue attribute.
setMinValue [▶ 1528]	TcHmiLinearGauge	Setter for the MinValue attribute.
getShowValueText [▶ 1529]	TcHmiLinearGauge	Getter for the ShowValueText attribute.
setShowValueText [▶ 1529]	TcHmiLinearGauge	Setter for the ShowValueText attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.

Name	Origin	Description
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.

Name	Origin	Description
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Linear Gauge

See also

Reference [▶ 306] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getProgressBackgroundColor 2.1

[Function]

```
public getProgressBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the ProgressBackgroundColor [▶ 320] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setProgressBackgroundColor [▶ 1516]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setProgressBackgroundColor 2.2

[Function]

```
public setProgressBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the ProgressBackgroundColor [▶ 320] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getProgressBackgroundColor [▶ 1515]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getProgressForegroundColor 2.3

[Function]

```
public getProgressForegroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the ProgressForegroundColor [▶ 320] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setProgressForegroundColor [▶ 1517]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setProgressForegroundColor 2.4

[Function]

```
public setProgressForegroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the ProgressForegroundColor [▶ 320] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getProgressForegroundColor [▶ 1516]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getKnobArrowsColor 2.5

[Function]

```
public getKnobArrowsColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the KnobArrowsColor [▶ 315] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKnobArrowsColor [▶ 1517]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setKnobArrowsColor 2.6

[Function]

```
public setKnobArrowsColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the KnobArrowsColor [▶ 315] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKnobArrowsColor [▶ 1517]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getTickColor
2.7

[Function]

```
public getTickColor(): TcHmi.Color | null;
```

This function returns the currently set value for the TickColor [▶ 322] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTickColor [▶ 1518]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setTickColor
2.8

[Function]

```
public setTickColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the TickColor [▶ 322] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTickColor [▶ 1518]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueColor
2.9

[Function]

```
public getValueColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the ValueColor [▶ 324] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueColor [▶ 1519]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueColor
2.10

[Function]

```
public setValueColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the ValueColor [▶ 324] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueColor [▶ 1519]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getKnobColor
2.11

[Function]

```
public getKnobColor(): TcHmi.Color | null;
```

This function returns the currently set value for the KnobColor [▶ 315] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKnobColor [▶ 1520]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setKnobColor
2.12

[Function]

```
public setKnobColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the KnobColor [▶ 315] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKnobColor [▶ 1520]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelColor
2.13

[Function]

```
public getLabelColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the LabelColor [▶ 315] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelColor [▶ 1521]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelColor
2.14

[Function]

```
public setLabelColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the LabelColor [▶ 315] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelColor [▶ 1521]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitColor
2.15

[Function]

```
public getUnitColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the UnitColor [▶ 322] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnitColor [▶ 1522]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitColor
2.16

[Function]

```
public setUnitColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the UnitColor [▶ 322] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitColor [▶ 1522]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getOrientation
2.17

[Function]

```
public getOrientation(): 'Horizontal' | 'Vertical';
```

This function returns the currently set value for the Orientation [▶ 319] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Horizontal' | 'Vertical'	The current value of the attribute.

See also

Attribute setter: setOrientation [▶ 1523]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setOrientation
2.18

[Function]

```
public setOrientation(newValue: 'Horizontal' | 'Vertical' | null): void;
```

This function sets a new value for the Orientation [▶ 319] attribute.

Parameter

Name	Type	Description
newValue	'Horizontal', 'Vertical', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getOrientation [▶ 1523]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getPadding
2.19

[Function]

```
public getPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the Padding [▶ 319] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setPadding [▶ 1524]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setPadding
2.20

[Function]

```
public setPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the Padding [▶ 319] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getPadding [▶ 1523]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValue
2.21

[Function]

```
public getValue(): number;
```

This function returns the currently set value for the Value [▶ 324] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValue [▶ 1525]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValue
2.22

[Function]

```
public setValue(newValue: number | null): void;
```

This function sets a new value for the Value [▶ 324] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValue [▶ 1524]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getEditable
2.23

[Function]

```
public getEditable(): boolean;
```

This function returns the currently set value for the Editable [▶ 314] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setEditable [▶ 1526]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setEditable
2.24

[Function]

```
public setEditable(newValue: boolean | null): void;
```

This function sets a new value for the Editable [▶ 314] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getEditable [▶ 1525]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getMaxValue
2.25

[Function]

```
public getMaxValue(): number;
```

This function returns the currently set value for the MaxValue [▶ 318] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMaxValue [▶ 1527]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setMaxValue
2.26

[Function]

```
public setMaxValue(newValue: number | null): void;
```

This function sets a new value for the MaxValue [▶ 318] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMaxValue [▶ 1526]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getMinValue
2.27

[Function]

```
public getMinValue(): number;
```

This function returns the currently set value for the MinValue [▶ 319] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMinValue [▶ 1528]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setMinValue
2.28

[Function]

```
public setMinValue(newValue: number | null): void;
```

This function sets a new value for the MinValue [▶ 319] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMinValue [▶ 1527]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getStep
2.29

[Function]

```
public getStep(): number;
```

This function returns the currently set value for the Step [▶ 321] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStep [▶ 1529]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setStep
2.30

[Function]

```
public setStep(newValue: number | null): void;
```

This function sets a new value for the Step [▶ 321] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStep [▶ 1528]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getShowValueText
2.31

[Function]

```
public getShowValueText(): boolean;
```

This function returns the currently set value for the ShowValueText [▶ 321] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowValueText [▶ 1529]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setShowValueText
2.32

[Function]

```
public setShowValueText(newValue: boolean | null): void;
```

This function sets a new value for the ShowValueText [▶ 321] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowValueText [▶ 1529]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getRange
2.33

[Function]

```
public getRange(): TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range[] | null;
```

This function returns the currently set value for the Range [▶ 320] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range [▶ 1556][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRange [▶ 1530]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setRange
2.34

[Function]

```
public setRange(newValue: TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range[] | null): void;
```

This function sets a new value for the Range [▶ 320] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range [▶ 1556][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRange [▶ 1530]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getStartFromZero
2.35

[Function]

```
public getStartFromZero(): boolean;
```

This function returns the currently set value for the StartFromZero [▶ 321] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setStartFromZero [▶ 1531]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setStartFromZero
2.36

[Function]

```
public setStartFromZero(newValue: boolean | null): void;
```

This function sets a new value for the StartFromZero [▶ 321] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStartFromZero [▶ 1531]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontWeight 2.37

[Function]

```
public getValueFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the ValueFontWeight [▶ 325] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setValueFontWeight [▶ 1532]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontWeight 2.38

[Function]

```
public setValueFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the ValueFontWeight [▶ 325] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontWeight [▶ 1532]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFormat
2.39

[Function]

```
public getValueFormat(): TcHmi.IFunction | null;
```

This function returns the currently set value for the ValueFormat [▶ 326] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095]	The current value of the attribute.

See also

Attribute setter: setValueFormat [▶ 1533]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFormat
2.40

[Function]

```
public setValueFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the ValueFormat [▶ 326] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFormat [▶ 1533]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontSizeUnit
2.41

[Function]

```
public getValueFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the ValueFontSizeUnit [▶ 325] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setValueFontSizeUnit [▶ 1534]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontSizeUnit
2.42

[Function]

```
public setValueFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the ValueFontSizeUnit [▶ 325] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontSizeUnit [▶ 1534]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontSize
2.43

[Function]

```
public getValueFontSize(): number;
```

This function returns the currently set value for the ValueFontSize [▶ 325] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueFontSize [▶ 1535]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontSize
2.44

[Function]

```
public setValueFontSize(newValue: number | null): void;
```

This function sets a new value for the ValueFontSize [▶ 325] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontSize [▶ 1535]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontFamily
2.45

[Function]

```
public getValueFontFamily(): string | null;
```

This function returns the currently set value for the ValueFontFamily [▶ 324] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueFontFamily [▶ 1536]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontFamily
2.46

[Function]

```
public setValueFontFamily(newValue: string | null): void;
```

This function sets a new value for the ValueFontFamily [▶ 324] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontFamily [▶ 1536]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getClickAnywhereToEdit
2.47

[Function]

```
public getClickAnywhereToEdit(): boolean;
```

This function returns the currently set value for the ClickAnywhereToEdit [▶ 314] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setClickAnywhereToEdit [▶ 1537]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setClickAnywhereToEdit
2.48

[Function]

```
public setClickAnywhereToEdit(newValue: boolean | null): void;
```

This function sets a new value for the ClickAnywhereToEdit [▶ 314] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getClickAnywhereToEdit [▶ 1536]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontStyle
2.49

[Function]

```
public getValueFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the ValueFontStyle [▶ 325] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setValueFontStyle [▶ 1538]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontStyle
2.50

[Function]

```
public setValueFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the ValueFontStyle [▶ 325] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontStyle [▶ 1537]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getShowLabels
2.51

[Function]

```
public getShowLabels(): boolean;
```

This function returns the currently set value for the ShowLabels [▶ 320] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowLabels [▶ 1539]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setShowLabels
2.52

[Function]

```
public setShowLabels(newValue: boolean | null): void;
```

This function sets a new value for the ShowLabels [▶ 320] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowLabels [▶ 1538]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelRange
2.53

[Function]

```
public getLabelRange(): number | null;
```

This function returns the currently set value for the LabelRange [▶ 318] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelRange [▶ 1540]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelRange 2.54

[Function]

```
public setLabelRange(newValue: number | null): void;
```

This function sets a new value for the LabelRange [▶ 318] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelRange [▶ 1539]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelPosition 2.55

[Function]

```
public getLabelPosition(): 'Top' | 'Bottom' | 'Left' | 'Right';
```

This function returns the currently set value for the LabelPosition [▶ 318] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Top' | 'Bottom' | 'Left' | 'Right'	The current value of the attribute.

See also

Attribute setter: setLabelPosition [▶ 1541]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelPosition
2.56

[Function]

```
public setLabelPosition(newValue: 'Top' | 'Bottom' | 'Left' | 'Right' | null): void;
```

This function sets a new value for the LabelPosition [▶ 318] attribute.

Parameter

Name	Type	Description
newValue	'Top', 'Bottom', 'Left', 'Right', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelPosition [▶ 1540]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontWeight
2.57

[Function]

```
public getLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the LabelFontWeight [▶ 317] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setLabelFontWeight [▶ 1542]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontWeight 2.58

[Function]

```
public setLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the LabelFontWeight [▶ 317] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontWeight [▶ 1541]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontFamily 2.59

[Function]

```
public getLabelFontFamily(): string | null;
```

This function returns the currently set value for the LabelFontFamily [▶ 315] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFontFamily [▶ 1542]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontFamily 2.60

[Function]

```
public setLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the LabelFontFamily [▶ 315] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontFamily [▶ 1542]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontSize
2.61

[Function]

```
public getLabelFontSize(): number;
```

This function returns the currently set value for the LabelFontSize [▶ 316] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFontSize [▶ 1543]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontSize
2.62

[Function]

```
public setLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the LabelFontSize [▶ 316] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontSize [▶ 1543]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontSizeUnit
2.63

[Function]

```
public getLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the LabelFontSizeUnit [▶ 316] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setLabelFontSizeUnit [▶ 1544]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontSizeUnit
2.64

[Function]

```
public setLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the LabelFontSizeUnit [▶ 316] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontSizeUnit [▶ 1544]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontStyle
2.65

[Function]

```
public getLabelFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the LabelFontStyle [▶ 316] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setLabelFontStyle [▶ 1545]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontStyle
2.66

[Function]

```
public setLabelFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the LabelFontStyle [▶ 316] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontStyle [▶ 1545]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFormat
2.67

[Function]

```
public getLabelFormat(): TcHmi.IFunction | null;
```

This function returns the currently set value for the LabelFormat [▶ 317] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095]	The current value of the attribute.

See also

Attribute setter: setLabelFormat [▶ 1546]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFormat
2.68

[Function]

```
public setLabelFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the LabelFormat [▶ 317] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFormat [▶ 1546]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getShowTicks
2.69

[Function]

```
public getShowTicks(): boolean;
```

This function returns the currently set value for the ShowTicks [▶ 321] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowTicks [▶ 1547]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setShowTicks
2.70

[Function]

```
public setShowTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowTicks [▶ 321] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowTicks [▶ 1547]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getTickDefinition
2.71

[Function]

```
public getTickDefinition(): TcHmi.Controls.Beckhoff.TcHmiLinearGauge.TickDefinition | null;
```

This function returns the currently set value for the TickDefinition [▶ 322] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiLinearGauge.TickDefinition [▶ 1556]	The current value of the attribute.

See also

Attribute setter: setTickDefinition [▶ 1548]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setTickDefinition
2.72

[Function]

```
public setTickDefinition(newValue: TcHmi.Controls.Beckhoff.TcHmiLinearGauge.TickDefinition | null):
void;
```

This function sets a new value for the TickDefinition [▶ 322] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiLinearGauge.TickDefinition [▶ 1556], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTickDefinition [▶ 1548]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnit
2.73

[Function]

```
public getUnit(): string;
```

This function returns the currently set value for the Unit [▶ 314] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnit [▶ 1549]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnit
2.74

[Function]

```
public setUnit(newValue: string | null): void;
```

This function sets a new value for the Unit [▶ 314] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnit [▶ 1549]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitOrientation
2.75

[Function]

```
public getUnitOrientation(): 'Theme' | 'Horizontal' | 'Vertical';
```

This function returns the currently set value for the UnitOrientation [▶ 319] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Theme' | 'Horizontal' | 'Vertical'	The current value of the attribute.

See also

Attribute setter: setUnitOrientation [▶ 1550]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitOrientation
2.76

[Function]

```
public setUnitOrientation(newValue: 'Theme' | 'Horizontal' | 'Vertical' | null): void;
```

This function sets a new value for the UnitOrientation [▶ 319] attribute.

Parameter

Name	Type	Description
newValue	'Theme', 'Horizontal', 'Vertical', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitOrientation [▶ 1549]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontSize
2.77

[Function]

```
public getUnitFontSize(): number;
```

This function returns the currently set value for the UnitFontSize [▶ 323] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnitFontSize [▶ 1551]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontSize
2.78

[Function]

```
public setUnitFontSize(newValue: number | null): void;
```

This function sets a new value for the UnitFontSize [▶ 323] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontSize [▶ 1550]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontStyle
2.79

[Function]

```
public getUnitFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the UnitFontStyle [▶ 323] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setUnitFontStyle [▶ 1552]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontStyle
2.80

[Function]

```
public setUnitFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the UnitFontStyle [▶ 323] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontStyle [▶ 1551]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontWeight
2.81

[Function]

```
public getUnitFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the UnitFontWeight [▶ 323] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setUnitFontWeight [▶ 1553]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontWeight
2.82

[Function]

```
public setUnitFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the UnitFontWeight [▶ 323] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontWeight [▶ 1552]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontFamily
2.83

[Function]

```
public getUnitFontFamily(): string | null;
```

This function returns the currently set value for the UnitFontFamily [▶ 322] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnitFontFamily [▶ 1554]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontFamily
2.84

[Function]

```
public setUnitFontFamily(newValue: string | null): void;
```

This function sets a new value for the UnitFontFamily [▶ 322] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontFamily [▶ 1553]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontSizeUnit
2.85

[Function]

```
public getUnitFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the UnitFontSizeUnit [▶ 323] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setUnitFontSizeUnit [▶ 1554]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontSizeUnit
2.86

[Function]

```
public setUnitFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the UnitFontSizeUnit [▶ 323] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontSizeUnit [▶ 1554]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 getBaseAnimationTime
2.87

[Function]

```
public getBaseAnimationTime(): number;
```

This function returns the currently set value for the BaseAnimationTime [▶ 314] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setBaseAnimationTime [▶ 1555]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 setBaseAnimationTime
2.88

[Function]

```
public setBaseAnimationTime(newValue: number | null): void;
```

This function sets a new value for the BaseAnimationTime [▶ 314] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getBaseAnimationTime [▶ 1555]

Origin: TcHmiLinearGauge [▶ 306]

Note: Available since version 1.8

15.1.1.2.2.1 TickDefinition
2.89

[Interface]

```
export interface TickDefinition {
    mainTickRange: number;
    subTickRange: number;
}
```

Properties

Name	Type	Description
mainTickRange	number [▶ 2108]	Distance of the main axis markers
subTickRange	number [▶ 2108]	Distance of the secondary axis markers

See also

Scheme: tchmi:framework#/definitions/GaugeTickDefinition [▶ 2237]

Note: Available since version 1.8

15.1.1.2.2.1 Range
2.90

[Interface]

```
export interface Range {
    color: TcHmi.SolidColor;
    start: number;
    end: number;
}
```

Properties

Name	Type	Description
color	TcHmi.SolidColor [▶ 2084]	Color of this entry
start	number [▶ 2108]	Starting value for this color range
end	number [▶ 2108]	End value for this color range

See also

Scheme: tchmi:framework#/definitions/GaugeRangeDefinitionList [▶ 2237]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiLineChart
3

[Class]

Line chart

Attribute functions

Name	Origin	Description
getLineGraphData [▶ 1575]	TcHmiLineChart	Getter for the LineGraphData attribute.
setLineGraphData [▶ 1575]	TcHmiLineChart	Setter for the LineGraphData attribute.
getLineGraphDescriptions [▶ 1575]	TcHmiLineChart	Getter for the LineGraphDescriptions attribute.
setLineGraphDescriptions [▶ 1576]	TcHmiLineChart	Setter for the LineGraphDescriptions attribute.
getShowGrid [▶ 1599]	TcHmiLineChart	Getter for the ShowGrid attribute.
setShowGrid [▶ 1600]	TcHmiLineChart	Setter for the ShowGrid attribute.
getGridLineWidth [▶ 1591]	TcHmiLineChart	Getter for the GridLineWidth attribute.
setGridLineWidth [▶ 1592]	TcHmiLineChart	Setter for the GridLineWidth attribute.
getGridBackgroundColor [▶ 1567]	TcHmiLineChart	Getter for the GridBackgroundColor attribute.
setGridBackgroundColor [▶ 1568]	TcHmiLineChart	Setter for the GridBackgroundColor attribute.
getSectionsBackgroundColor [▶ 1566]	TcHmiLineChart	Getter for the SectionsBackgroundColor attribute
setSectionsBackgroundColor [▶ 1567]	TcHmiLineChart	Setter for the SectionsBackgroundColor attribute
getGridLineColor [▶ 1569]	TcHmiLineChart	Getter for the GridLineColor attribute.
setGridLineColor [▶ 1569]	TcHmiLineChart	Setter for the GridLineColor attribute.
getGridShowHorizontalLines [▶ 1589]	TcHmiLineChart	Getter for the GridShowHorizontalLines attribute.
setGridShowHorizontalLines [▶ 1590]	TcHmiLineChart	Setter for the GridShowHorizontalLines attribute.
getGridShowVerticalLines [▶ 1587]	TcHmiLineChart	Getter for the GridShowVerticalLines attribute.
setGridShowVerticalLines [▶ 1588]	TcHmiLineChart	Setter for the GridShowVerticalLines attribute.
getShowXAxis [▶ 1576]	TcHmiLineChart	Getter for the ShowXAxis attribute.
setShowXAxis [▶ 1577]	TcHmiLineChart	Setter for the ShowXAxis attribute.
getShowXAxisName [▶ 1582]	TcHmiLineChart	Getter for the ShowXAxisName attribute.
setShowXAxisName [▶ 1582]	TcHmiLineChart	Setter for the ShowXAxisName attribute.
getXAxisColor [▶ 1568]	TcHmiLineChart	Getter for the XAxisColor attribute.
setXAxisColor [▶ 1569]	TcHmiLineChart	Setter for the XAxisColor attribute.
getXAxisName [▶ 1583]	TcHmiLineChart	Getter for the XAxisName attribute.
setXAxisName [▶ 1583]	TcHmiLineChart	Setter for the XAxisName attribute.
getXAxisNameFontColor [▶ 1571]	TcHmiLineChart	Getter for the XAxisNameFontColor attribute.
setXAxisNameFontColor [▶ 1571]	TcHmiLineChart	Setter for the XAxisNameFontColor attribute.
getXAxisNameFontFamily [▶ 1590]	TcHmiLineChart	Getter for the XAxisNameFontFamily attribute.

Name	Origin	Description
setXAxisNameFontFamily [▶ 1591]	TcHmiLineChart	Setter for the XAxisNameFontFamily attribute.
getXAxisNameFontSize [▶ 1593]	TcHmiLineChart	Getter for the XAxisNameFontSize attribute.
setXAxisNameFontSize [▶ 1593]	TcHmiLineChart	Setter for the XAxisNameFontSize attribute.
getXAxisNameFontSizeUnit [▶ 1594]	TcHmiLineChart	Getter for the XAxisNameFontSizeUnit attribute.
setXAxisNameFontSizeUnit [▶ 1594]	TcHmiLineChart	Setter for the XAxisNameFontSizeUnit attribute.
getXAxisNameFontWeight [▶ 1595]	TcHmiLineChart	Getter for the XAxisNameFontWeight attribute.
setXAxisNameFontWeight [▶ 1595]	TcHmiLineChart	Setter for the XAxisNameFontWeight attribute.
getXAxisUnit [▶ 1585]	TcHmiLineChart	Getter for the XAxisUnit attribute.
setXAxisUnit [▶ 1585]	TcHmiLineChart	Setter for the XAxisUnit attribute.
getXAxisWidth [▶ 1588]	TcHmiLineChart	Getter for the XAxisWidth attribute.
setXAxisWidth [▶ 1589]	TcHmiLineChart	Setter for the XAxisWidth attribute.
getXAxisAutoScaling [▶ 1584]	TcHmiLineChart	Getter for the XAxisAutoScaling attribute.
setXAxisAutoScaling [▶ 1584]	TcHmiLineChart	Setter for the XAxisAutoScaling attribute.
getXAxisDecimalPlaces [▶ 1586]	TcHmiLineChart	Getter for the XAxisDecimalPlaces attribute.
setXAxisDecimalPlaces [▶ 1586]	TcHmiLineChart	Setter for the XAxisDecimalPlaces attribute.
getXShowLabels [▶ 1577]	TcHmiLineChart	Getter for the XShowLabels attribute.
setXShowLabels [▶ 1578]	TcHmiLineChart	Setter for the XShowLabels attribute.
getXLabelFontColor [▶ 1572]	TcHmiLineChart	Getter for the XLabelFontColor attribute.
setXLabelFontColor [▶ 1572]	TcHmiLineChart	Setter for the XLabelFontColor attribute.
getXLabelFontFamily [▶ 1592]	TcHmiLineChart	Getter for the XLabelFontFamily attribute.
setXLabelFontFamily [▶ 1593]	TcHmiLineChart	Setter for the XLabelFontFamily attribute.
getXLabelFontSize [▶ 1596]	TcHmiLineChart	Getter for the XLabelFontSize attribute.
setXLabelFontSize [▶ 1596]	TcHmiLineChart	Setter for the XLabelFontSize attribute.
getXLabelFontSizeUnit [▶ 1597]	TcHmiLineChart	Getter for the XLabelFontSizeUnit attribute.
setXLabelFontSizeUnit [▶ 1597]	TcHmiLineChart	Setter for the XLabelFontSizeUnit attribute.
getXLabelFontWeight [▶ 1598]	TcHmiLineChart	Getter for the XLabelFontWeight attribute.
setXLabelFontWeight [▶ 1598]	TcHmiLineChart	Setter for the XLabelFontWeight attribute.
getXMainTickSteps [▶ 1587]	TcHmiLineChart	Getter for the XMainTickSteps attribute.

Name	Origin	Description
setXMainTickSteps [▶ 1587]	TcHmiLineChart	Setter for the XMainTickSteps attribute.
getXMainTickMinValue [▶ 1580]	TcHmiLineChart	Getter for the XMainTickMinValue attribute.
setXMainTickMinValue [▶ 1581]	TcHmiLineChart	Setter for the XMainTickMinValue attribute.
getXMainTickMaxValue [▶ 1581]	TcHmiLineChart	Getter for the XMainTickMaxValue attribute.
setXMainTickMaxValue [▶ 1581]	TcHmiLineChart	Setter for the XMainTickMaxValue attribute.
getShowXSubTicks [▶ 1578]	TcHmiLineChart	Getter for the ShowXSubTicks attribute.
setShowXSubTicks [▶ 1579]	TcHmiLineChart	Setter for the ShowXSubTicks attribute.
getXSubTickSteps [▶ 1579]	TcHmiLineChart	Getter for the XSubTickSteps attribute.
setXSubTickSteps [▶ 1580]	TcHmiLineChart	Setter for the XSubTickSteps attribute.
getXAxisFormat [▶ 1599]	TcHmiLineChart	Getter for the XAxisFormat attribute
setXAxisFormat [▶ 1599]	TcHmiLineChart	Setter for the XAxisFormat attribute
getYMainTickSteps [▶ 1601]	TcHmiLineChart	Getter for the YMainTickSteps attribute.
setYMainTickSteps [▶ 1602]	TcHmiLineChart	Setter for the YMainTickSteps attribute.
getShowYSubTicks [▶ 1603]	TcHmiLineChart	Getter for the ShowYSubTicks attribute.
setShowYSubTicks [▶ 1604]	TcHmiLineChart	Setter for the ShowYSubTicks attribute.
getYSubTickSteps [▶ 1602]	TcHmiLineChart	Getter for the YSubTickSteps attribute.
setYSubTickSteps [▶ 1603]	TcHmiLineChart	Setter for the YSubTickSteps attribute.
getSubgridLineColor [▶ 1570]	TcHmiLineChart	Getter for the SubgridLineColor attribute.
setSubgridLineColor [▶ 1570]	TcHmiLineChart	Setter for the SubgridLineColor attribute.
getSubgridLineWidth [▶ 1611]	TcHmiLineChart	Getter for the SubgridLineWidth attribute.
setSubgridLineWidth [▶ 1611]	TcHmiLineChart	Setter for the SubgridLineWidth attribute.
getSubgridShowHorizontalLines [▶ 1610]	TcHmiLineChart	Getter for the SubgridShowHorizontalLines attribute.
setSubgridShowHorizontalLines [▶ 1610]	TcHmiLineChart	Setter for the SubgridShowHorizontalLines attribute.
getSubgridShowVerticalLines [▶ 1609]	TcHmiLineChart	Getter for the SubgridShowVerticalLines attribute.
setSubgridShowVerticalLines [▶ 1609]	TcHmiLineChart	Setter for the SubgridShowVerticalLines attribute.
getYAxis [▶ 1600]	TcHmiLineChart	Getter for the YAxis attribute.

Name	Origin	Description
setYAxis [▶ 1601]	TcHmiLineChart	Setter for the YAxis attribute.
getYAxisWidth [▶ 1607]	TcHmiLineChart	Getter for the YAxisWidth attribute.
setYAxisWidth [▶ 1607]	TcHmiLineChart	Setter for the YAxisWidth attribute.
getYLabelFontFamily [▶ 1608]	TcHmiLineChart	Getter for the YLabelFontFamily attribute.
setYLabelFontFamily [▶ 1608]	TcHmiLineChart	Setter for the YLabelFontFamily attribute.
getYLabelFontSize [▶ 1604]	TcHmiLineChart	Getter for the YLabelFontSize attribute.
setYLabelFontSize [▶ 1605]	TcHmiLineChart	Setter for the YLabelFontSize attribute.
getYLabelFontSizeUnit [▶ 1605]	TcHmiLineChart	Getter for the YLabelFontSizeUnit attribute.
setYLabelFontSizeUnit [▶ 1605]	TcHmiLineChart	Setter for the YLabelFontSizeUnit attribute.
getYLabelFontWeight [▶ 1606]	TcHmiLineChart	Getter for the YLabelFontWeight attribute.
setYLabelFontWeight [▶ 1606]	TcHmiLineChart	Setter for the YLabelFontWeight attribute.
setShowTooltip [▶ 1613]	TcHmiLineChart	Setter for the ShowTooltip attribute.
getShowTooltip [▶ 1613]	TcHmiLineChart	Getter for the ShowTooltip attribute.
setTooltipFormat [▶ 1614]	TcHmiLineChart	Setter for the TooltipFormat attribute.
getTooltipFormat [▶ 1614]	TcHmiLineChart	Getter for the TooltipFormat attribute.
setTooltipFontFamily [▶ 1615]	TcHmiLineChart	Setter for the TooltipFontFamily attribute.
getTooltipFontFamily [▶ 1615]	TcHmiLineChart	Getter for the TooltipFontFamily attribute.
getTooltipFontSize [▶ 1616]	TcHmiLineChart	Getter for the TooltipFontSize attribute.
setTooltipFontSize [▶ 1616]	TcHmiLineChart	Setter for the TooltipFontSize attribute.
getTooltipFontSizeUnit [▶ 1617]	TcHmiLineChart	Getter for the TooltipFontSizeUnit attribute.
setTooltipFontSizeUnit [▶ 1617]	TcHmiLineChart	Setter for the TooltipFontSizeUnit attribute.
getTooltipFontWeight [▶ 1618]	TcHmiLineChart	Getter for the TooltipFontWeight attribute.
setTooltipFontWeight [▶ 1618]	TcHmiLineChart	Setter for the TooltipFontWeight attribute.
getTooltipFontColor [▶ 1573]	TcHmiLineChart	Getter for the TooltipFontColor attribute.
setTooltipFontColor [▶ 1573]	TcHmiLineChart	Setter for the TooltipFontColor attribute.
getTooltipBackgroundColor [▶ 1574]	TcHmiLineChart	Getter for the TooltipBackgroundColor attribute.
setTooltipBackgroundColor [▶ 1574]	TcHmiLineChart	Setter for the TooltipBackgroundColor attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.

Name	Origin	Description
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.

Name	Origin	Description
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.

Name	Origin	Description
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.

Name	Origin	Description
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Line Chart

See also

Reference [▶ 281] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getSectionsBackgroundColor 3.1

[Function]

```
public getSectionsBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the SectionsBackgroundColor [▶ 292] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084]	The current value of the attribute.

See also

Attribute setter: setSectionsBackgroundColor [▶ 1567]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setSectionsBackgroundColor
3.2

[Function]

```
public setSectionsBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the SectionsBackgroundColor [▶ 292] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSectionsBackgroundColor [▶ 1566]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getGridBackgroundColor
3.3

[Function]

```
public getGridBackgroundColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the GridBackgroundColor [▶ 291] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084]	The current value of the attribute.

See also

Attribute setter: setGridBackgroundColor [▶ 1568]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setGridBackgroundColor
3.4

[Function]

```
public setGridBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the GridBackgroundColor [▶ 291] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridBackgroundColor [▶ 1567]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisColor
3.5

[Function]

```
public getXAxisColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XAxisColor [▶ 298] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisColor [▶ 1569]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisColor
3.6

[Function]

```
public setXAxisColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XAxisColor [▶ 298] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisColor [▶ 1568]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getGridLineColor
3.7

[Function]

```
public getGridLineColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the GridLineColor [▶ 291] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridLineColor [▶ 1569]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setGridLineColor
3.8

[Function]

```
public setGridLineColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the GridLineColor [▶ 291] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridLineColor [▶ 1569]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getSubgridLineColor
3.9

[Function]

```
public getSubgridLineColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the SubgridLineColor [▶ 294] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSubgridLineColor [▶ 1570]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setSubgridLineColor
3.10

[Function]

```
public setSubgridLineColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the SubgridLineColor [▶ 294] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridLineColor [▶ 1570]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisNameFontColor
3.11

[Function]

```
public getXAxisNameFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XAxisNameFontColor [▶ 299] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontColor [▶ 1571]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisNameFontColor
3.12

[Function]

```
public setXAxisNameFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XAxisNameFontColor [▶ 299] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontColor [▶ 1571]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXLabelFontColor
3.13

[Function]

```
public getXLabelFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XLabelFontColor [▶ 301] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontColor [▶ 1572]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXLabelFontColor
3.14

[Function]

```
public setXLabelFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XLabelFontColor [▶ 301] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontColor [▶ 1572]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getTooltipFontColor
3.15

[Function]

```
public getTooltipFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TooltipFontColor [▶ 295] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipFontColor [▶ 1573]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipFontColor
3.16

[Function]

```
public seTooltipFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TooltipFontColor [▶ 295] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontColor [▶ 1573]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getTooltipBackgroundColor 3.17

[Function]

```
public getTooltipBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the TooltipBackgroundColor [▶ 295] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipBackgroundColor [▶ 1574]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipBackgroundColor 3.18

[Function]

```
public setTooltipBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TooltipBackgroundColor [▶ 295] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipBackgroundColor [▶ 1574]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getLineGraphData
3.19

[Function]

```
public getLineGraphData(): TcHmiCharting.LineGraph.Point[][] | null;
```

This function returns the currently set value for the LineGraphData [▶ 292] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmiCharting.LineGraph.Point [▶ 2109][][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLineGraphData [▶ 1575]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setLineGraphData
3.20

[Function]

```
public setLineGraphData(newValue: TcHmiCharting.LineGraph.Point[][] | null): void;
```

This function sets a new value for the LineGraphData [▶ 292] attribute.

Parameter

Name	Type	Description
newValue	TcHmiCharting.LineGraph.Point [▶ 2109][][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLineGraphData [▶ 1575]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getLineGraphDescriptions
3.21

[Function]

```
public getLineGraphDescriptions(): TcHmi.Controls.Beckhoff.TcHmiLineChart.lineGraphDescription[] |
null;
```

This function returns the currently set value for the LineGraphDescriptions [▶ 292] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiLineChart.lineGraphDescription [▶ 1612][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLineGraphDescriptions [▶ 1576]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setLineGraphDescriptions
3.22

[Function]

```
public setLineGraphDescriptions(newValue:
TcHmi.Controls.Beckhoff.TcHmiLineChart.lineGraphDescription[] | null): void;
```

This function sets a new value for the LineGraphDescriptions [▶ 292] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiLineChart.lineGraphDescription [▶ 1612][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLineGraphDescriptions [▶ 1575]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getShowXAxis
3.23

[Function]

```
public getShowXAxis(): boolean;
```

This function returns the currently set value for the ShowXAxis [▶ 293] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXAxis [▶ 1577]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setShowXAxis
3.24

[Function]

```
public setShowXAxis(newValue: boolean | null): void;
```

This function sets a new value for the ShowXAxis [▶ 293] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXAxis [▶ 1576]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXShowLabels
3.25

[Function]

```
public getXShowLabels(): boolean;
```

This function returns the currently set value for the XShowLabels [▶ 303] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setXShowLabels [▶ 1578]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXShowLabels
3.26

[Function]

```
public setXShowLabels(newValue: boolean | null): void;
```

This function sets a new value for the XShowLabels [▶ 303] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXShowLabels [▶ 1577]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getShowXSubTicks
3.27

[Function]

```
public getShowXSubTicks(): boolean;
```

This function returns the currently set value for the ShowXSubTicks [▶ 293] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXSubTicks [▶ 1579]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setShowXSubTicks
3.28

[Function]

```
public setShowXSubTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowXSubTicks [▶ 293] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXSubTicks [▶ 1578]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXSubTickSteps
3.29

[Function]

```
public getXSubTickSteps(): number | null;
```

This function returns the currently set value for the XSubTickSteps [▶ 304] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXSubTickSteps [▶ 1580]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXSubTickSteps
3.30

[Function]

```
public setXSubTickSteps(newValue: number | null): void;
```

This function sets a new value for the XSubTickSteps [▶ 304] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXSubTickSteps [▶ 1579]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXMainTickMinValue
3.31

[Function]

```
public getXMainTickMinValue(): number | null;
```

This function returns the currently set value for the XMainTickMinValue [▶ 303] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickMinValue [▶ 1581]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXMainTickMinValue
3.32

[Function]

```
public setXMainTickMinValue(newValue: number | null): void;
```

This function sets a new value for the XMainTickMinValue [▶ 303] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickMinValue [▶ 1580]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXMainTickMaxValue
3.33

[Function]

```
public getXMainTickMaxValue(): number | null;
```

This function returns the currently set value for the XMainTickMaxValue [▶ 303] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickMaxValue [▶ 1581]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXMainTickMaxValue
3.34

[Function]

```
public setXMainTickMaxValue(newValue: number | null): void;
```

This function sets a new value for the XMainTickMaxValue [▶ 303] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickMaxValue [▶ 1581]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getShowXAxisName
3.35

[Function]

```
public getShowXAxisName(): boolean;
```

This function returns the currently set value for the ShowXAxisName [▶ 293] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXAxisName [▶ 1582]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setShowXAxisName
3.36

[Function]

```
public setShowXAxisName(newValue: boolean | null): void;
```

This function sets a new value for the ShowXAxisName [▶ 293] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXAxisName [▶ 1582]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisName
3.37

[Function]

```
public getXAxisName(): string | null;
```

This function returns the currently set value for the XAxisName [▶ 299] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisName [▶ 1583]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisName
3.38

[Function]

```
public setXAxisName(newValue: string | null): void;
```

This function sets a new value for the XAxisName [▶ 299] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisName [▶ 1583]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisAutoScaling
3.39

[Function]

```
public getXAxisAutoScaling(): boolean;
```

This function returns the currently set value for the XAxisAutoScaling [▶ 297] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setXAxisAutoScaling [▶ 1584]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisAutoScaling
3.40

[Function]

```
public setXAxisAutoScaling(newValue: boolean | null): void;
```

This function sets a new value for the XAxisAutoScaling [▶ 297] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisAutoScaling [▶ 1584]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisUnit
3.41

[Function]

```
public getXAxisUnit(): string | null;
```

This function returns the currently set value for the XAxisUnit [▶ 301] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisUnit [▶ 1585]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisUnit
3.42

[Function]

```
public setXAxisUnit(newValue: string | null): void;
```

This function sets a new value for the XAxisUnit [▶ 301] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisUnit [▶ 1585]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisDecimalPlaces
3.43

[Function]

```
public getXAxisDecimalPlaces(): number;
```

This function returns the currently set value for the XAxisDecimalPlaces [▶ 298] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisDecimalPlaces [▶ 1586]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisDecimalPlaces
3.44

[Function]

```
public setXAxisDecimalPlaces(newValue: number | null): void;
```

This function sets a new value for the XAxisDecimalPlaces [▶ 298] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisDecimalPlaces [▶ 1586]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXMainTickSteps
3.45

[Function]

```
public getXMainTickSteps(): number;
```

This function returns the currently set value for the XMainTickSteps [▶ 303] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickSteps [▶ 1587]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXMainTickSteps
3.46

[Function]

```
public setXMainTickSteps(newValue: number | null): void;
```

This function sets a new value for the XMainTickSteps [▶ 303] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickSteps [▶ 1587]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getGridShowVerticalLines
3.47

[Function]

```
public getGridShowVerticalLines(): boolean;
```

This function returns the currently set value for the GridShowVerticalLines [▶ 292] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setGridShowVerticalLines [▶ 1588]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setGridShowVerticalLines
3.48

[Function]

```
public setGridShowVerticalLines(newValue: boolean | null): void;
```

This function sets a new value for the GridShowVerticalLines [▶ 292] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridShowVerticalLines [▶ 1587]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisWidth
3.49

[Function]

```
public getXAxisWidth(): number;
```

This function returns the currently set value for the XAxisWidth [▶ 301] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisWidth [▶ 1589]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisWidth
3.50

[Function]

```
public setXAxisWidth(newValue: number | null): void;
```

This function sets a new value for the XAxisWidth [▶ 301] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisWidth [▶ 1588]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getGridShowHorizontalLines
3.51

[Function]

```
public getGridShowHorizontalLines(): boolean;
```

This function returns the currently set value for the GridShowHorizontalLines [▶ 291] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setGridShowHorizontalLines [▶ 1590]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setGridShowHorizontalLines
3.52

[Function]

```
public setGridShowHorizontalLines(newValue: boolean | null): void;
```

This function sets a new value for the GridShowHorizontalLines [▶ 291] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridShowHorizontalLines [▶ 1589]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisNameFontFamily
3.53

[Function]

```
public getXAxisNameFontFamily(): string;
```

This function returns the currently set value for the XAxisNameFontFamily [▶ 300] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontFamily [▶ 1591]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisNameFontFamily
3.54

[Function]

```
public setXAxisNameFontFamily(newValue: string | null): void;
```

This function sets a new value for the XAxisNameFontFamily [▶ 300] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontFamily [▶ 1590]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getGridLineWidth
3.55

[Function]

```
public getGridLineWidth(): number;
```

This function returns the currently set value for the GridLineWidth [▶ 291] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridLineWidth [▶ 1592]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setGridLineWidth
3.56

[Function]

```
public setGridLineWidth(newValue: number | null): void;
```

This function sets a new value for the GridLineWidth [▶ 291] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridLineWidth [▶ 1591]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXLabelFontFamily
3.57

[Function]

```
public getXLabelFontFamily(): string;
```

This function returns the currently set value for the XLabelFontFamily [▶ 302] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontFamily [▶ 1593]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXLabelFontFamily
3.58

[Function]

```
public setXLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the XLabelFontFamily [▶ 302] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontFamily [▶ 1592]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisNameFontSize
3.59

[Function]

```
public getXAxisNameFontSize(): number;
```

This function returns the currently set value for the XAxisNameFontSize [▶ 300] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontSize [▶ 1593]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisNameFontSize
3.60

[Function]

```
public setXAxisNameFontSize(newValue: number | null): void;
```

This function sets a new value for the XAxisNameFontSize [▶ 300] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontSize [▶ 1593]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisNameFontSizeUnit
3.61

[Function]

```
public getXAxisNameFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the XAxisNameFontSizeUnit [▶ 300] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontSizeUnit [▶ 1594]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisNameFontSizeUnit
3.62

[Function]

```
public setXAxisNameFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the XAxisNameFontSizeUnit [▶ 300] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontSizeUnit [▶ 1594]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisNameFontWeight
3.63

[Function]

```
public getXAxisNameFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the XAxisNameFontWeight [▶ 301] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontWeight [▶ 1595]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXAxisNameFontWeight
3.64

[Function]

```
public setXAxisNameFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the XAxisNameFontWeight [▶ 301] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontWeight [▶ 1595]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXLabelFontSize
3.65

[Function]

```
public getXLabelFontSize(): number;
```

This function returns the currently set value for the XLabelFontSize [▶ 302] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontSize [▶ 1596]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXLabelFontSize
3.66

[Function]

```
public setXLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the XLabelFontSize [▶ 302] attribute.

Parameter

Name	Type	Description
newValue	numbe, nullr [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontSize [▶ 1596]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXLabelFontSizeUnit
3.67

[Function]

```
public getXLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the XLabelFontSizeUnit [▶ 302] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setXLabelFontSizeUnit [▶ 1597]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXLabelFontSizeUnit
3.68

[Function]

```
public setXLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the XLabelFontSizeUnit [▶ 302] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontSizeUnit [▶ 1597]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXLabelFontWeight
3.69

[Function]

```
public getXLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the XLabelFontFontWeight [▶ 302] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setXLabelFontWeight [▶ 1598]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setXLabelFontWeight
3.70

[Function]

```
public setXLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the XLabelFontWeight [▶ 302] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontWeight [▶ 1598]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getXAxisFormat
3.71

[Function]

```
public getXAxisFormat(): TcHmi.IFunction;
```

This function returns the currently set value for the XAxisFormat [▶ 298] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095]	The current value of the attribute.

See also

Attribute setter: setXAxisFormat [▶ 1599]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setXAxisFormat
3.72

[Function]

```
public setXAxisFormat(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the XAxisFormat [▶ 298] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisFormat [▶ 1599]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getShowGrid
3.73

[Function]

```
public getShowGrid(): boolean;
```

This function returns the currently set value for the ShowGrid [▶ 292] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowGrid [▶ 1600]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setShowGrid
3.74

[Function]

```
public setShowGrid(newValue: boolean | null): void;
```

This function sets a new value for the ShowGrid [▶ 292] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowGrid [▶ 1599]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYAxis
3.75

[Function]

```
public getYAxis(): TcHmi.Controls.Beckhoff.TcHmiLineChart.YAxis[] | null;
```

This function returns the currently set value for the YAxis [▶ 304] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiLineChart.YAxis [▶ 1611][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxis [▶ 1601]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYAxis
3.76

[Function]

```
public setYAxis(newValue: TcHmi.Controls.Beckhoff.TcHmiLineChart.YAxis[] | null): void;
```

This function sets a new value for the YAxis [▶ 304] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiLineChart.YAxis [▶ 1611][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxis [▶ 1600]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYMainTickSteps
3.77

[Function]

```
public getYMainTickSteps(): number;
```

This function returns the currently set value for the YMainTickSteps [▶ 305] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYMainTickSteps [▶ 1602]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYMainTickSteps
3.78

[Function]

```
public setYMainTickSteps(newValue: number | null): void;
```

This function sets a new value for the YMainTickSteps [▶ 305] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYMainTickSteps [▶ 1601]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYSubTickSteps
3.79

[Function]

```
public getYSubTickSteps(): number | null;
```

This function returns the currently set value for the YSubTickSteps [▶ 306] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYSubTickSteps [▶ 1603]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYSubTickSteps
3.80

[Function]

```
public setYSubTickSteps(newValue: number | null): void;
```

This function sets a new value for the YSubTickSteps [▶ 306] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYSubTickSteps [▶ 1602]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getShowYSubTicks
3.81

[Function]

```
public getShowYSubTicks(): boolean;
```

This function returns the currently set value for the ShowYSubTicks [▶ 294] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowYSubTicks [▶ 1604]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setShowYSubTicks
3.82

[Function]

```
public setShowYSubTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowYSubTicks [▶ 294] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowYSubTicks [▶ 1603]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYLabelFontSize
3.83

[Function]

```
public getYLabelFontSize(): number;
```

This function returns the currently set value for the YLabelFontSize [▶ 305] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontSize [▶ 1605]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYLabelFontSize
3.84

[Function]

```
public setYLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the YLabelFontSize [▶ 305] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontSize [▶ 1604]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYLabelFontSizeUnit
3.85

[Function]

```
public getYLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the YLabelFontSizeUnit [▶ 305] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setYLabelFontSizeUnit [▶ 1605]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYLabelFontSizeUnit
3.86

[Function]

```
public setYLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the YLabelFontSizeUnit [▶ 305] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontSizeUnit [▶ 1605]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYLabelFontWeight
3.87

[Function]

```
public getYLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the YLabelFontWeight [▶ 305] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setYLabelFontWeight [▶ 1606]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYLabelFontWeight
3.88

[Function]

```
public setYLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the YLabelFontWeight [▶ 305] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontWeight [▶ 1606]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYAxisWidth
3.89

[Function]

```
public getYAxisWidth(): number;
```

This function returns the currently set value for the YAxisWidth [▶ 304] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisWidth [▶ 1607]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYAxisWidth
3.90

[Function]

```
public setYAxisWidth(newValue: number | null): void;
```

This function sets a new value for the YAxisWidth [▶ 304] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisWidth [▶ 1607]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getYLabelFontFamily
3.91

[Function]

```
public getYLabelFontFamily(): string;
```

This function returns the currently set value for the YLabelFontFamily [▶ 304] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontFamily [▶ 1608]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setYLabelFontFamily
3.92

[Function]

```
public setYLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the YLabelFontFamily [▶ 304] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontFamily [▶ 1608]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getSubgridShowVerticalLines
3.93

[Function]

```
public getSubgridShowVerticalLines(): boolean;
```

This function returns the currently set value for the SubgridShowVerticalLines [▶ 295] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setSubgridShowVerticalLines [▶ 1609]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setSubgridShowVerticalLines
3.94

[Function]

```
public setSubgridShowVerticalLines(newValue: boolean | null): void;
```

This function sets a new value for the SubgridShowVerticalLines [▶ 295] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridShowVerticalLines [▶ 1609]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getSubgridShowHorizontalLines 3.95

[Function]

```
public getSubgridShowHorizontalLines(): boolean;
```

This function returns the currently set value for the SubgridShowHorizontalLines [▶ 294] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setSubgridShowHorizontalLines [▶ 1610]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setSubgridShowHorizontalLines 3.96

[Function]

```
public setSubgridShowHorizontalLines(newValue: boolean | null): void;
```

This function sets a new value for the SubgridShowHorizontalLines [▶ 294] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridShowHorizontalLines [▶ 1610]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 getSubgridLineWidth
3.97

[Function]

```
public getSubgridLineWidth(): number;
```

This function returns the currently set value for the SubgridLineWidth [▶ 294] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSubgridLineWidth [▶ 1611]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 setSubgridLineWidth
3.98

[Function]

```
public setSubgridLineWidth(newValue: number | null): void;
```

This function sets a new value for the SubgridLineWidth [▶ 294] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridLineWidth [▶ 1611]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.8

15.1.1.2.2.1 YAxis
3.99

[Interface]

```
export interface YAxis {
    id: number;
    position: TcHmiCharting.IAxis.Position;
    mainTickMinValue?: number;
    mainTickMaxValue?: number;
    showLabels?: boolean;
    labelFontColor?: TcHmi.SolidColor;
    axisColor?: TcHmi.SolidColor;
    showAxisName?: boolean;
    axisName?: string;
    axisNameFontFamily?: string;
    axisNameFontSize?: number;
    axisNameFontSizeUnit?: TcHmi.DimensionUnit;
    axisNameFontWeight?: TcHmi.FontWeight;
    axisNameFontColor?: TcHmi.SolidColor;
    decimalPlaces?: number;
    unit?: string;
    autoScaling?: boolean;
}
```

Properties

Name	Type	Description
id	number [▶ 2108]	id of the y-axis
position	TcHmiCharting.IAxis.Position [▶ 2109]	Position as enumeration
mainTickMinValue	number [▶ 2108]	Minimum value of the main axis marker
mainTickMaxValue	number [▶ 2108]	Maximum value of the main axis marker
showLabels	boolean [▶ 2107]	Is the label to be displayed
labelFontColor (optional)	TcHmi.SolidColor [▶ 2084]	Color of the label text
axisColor (optional)	TcHmi.SolidColor [▶ 2084]	Axis color
showAxisName (optional)	boolean [▶ 2107]	Is the axis name to be displayed
axisName (optional)	string [▶ 2108]	Name of the axis
axisNameFontFamily (optional)	string [▶ 2108]	Font of the axis name
axisNameFontSize (optional)	number [▶ 2108]	Font size of the axis name
axisNameFontSizeUnit (optional)	TcHmi.DimensionUnit [▶ 2086]	Unit of the font size of the axis name
axisNameFontWeight (optional)	TcHmi.FontWeight [▶ 2088]	Font weight of the axis name
axisNameFontColor (optional)	TcHmi.SolidColor [▶ 2084]	Font color of the axis name
decimalPlaces (optional)	number [▶ 2108]	Decimal places of the axis labeling
unit (optional)	string [▶ 2108]	Display unit
autoScaling (optional)	boolean [▶ 2107]	Should the axis adapt automatically

See also

Scheme: tchmi:framework#/definitions/ChartYAxisDefinition [▶ 2209]

Note: Available since version 1.8

15.1.1.2.2.1 lineGraphDescription
3.100

[Interface]

```
export interface lineGraphDescription {
    yAxisId: number;
    lineWidth?: number;
    lineColor?: TcHmi.SolidColor;
    pointDot?: boolean;
```

```
    pointDotRadius?: number;
    pointDotFillColor?: TcHmi.SolidColor;
    pointDotStrokeWidth?: number;
    pointDotStrokeColor?: TcHmi.SolidColor;
}
```

Properties

Name	Type	Description
yAxisId	number [▶ 2108]	id of the y-axis
lineWidth (optional)	number [▶ 2108]	
lineColor (optional)	TcHmi.SolidColor [▶ 2084]	
pointDot	boolean [▶ 2107]	Are points to be drawn
pointDotRadius (optional)	number [▶ 2108]	Radius of the point circles
pointDotFillColor (optional)	TcHmi.SolidColor [▶ 2084]	Fill color of the point circle
pointDotStrokeWidth (optional)	number [▶ 2108]	Stroke width of the point circle
pointDotStrokeColor (optional)	TcHmi.SolidColor [▶ 2084]	Stroke color of the point circle

See also

Scheme: tchmi:framework#/definitions/ChartLineGraphDescriptionDefinitionList [▶ 2207]

Note: Available since version 1.8

15.1.1.2.2.1 getShowTooltip
3.101

[Function]

```
public getShowTooltip(): boolean;
```

This function returns the currently set value for the ShowTooltip [▶ 293] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowTooltip [▶ 1613]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setShowTooltip
3.102

[Function]

```
public setShowTooltip(newValue: boolean | null): void;
```

This function sets a new value for the ShowTooltip attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowTooltip [▶ 1613]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getTooltipFormat
3.103

[Function]

```
public getTooltipFormat(): TcHmi.IFunction;
```

This function returns the currently set value for the TooltipFormat [▶ 296] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 1891]	The current value of the attribute.

See also

Attribute setter: setTooltipFormat [▶ 1614]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipFormat
3.104

[Function]

```
public setTooltipFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the TooltipFormat [▶ 296] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 1891], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFormat [▶ 1613]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getTooltipFontFamily
3.105

[Function]

```
public getTooltipFontFamily(): string;
```

This function returns the currently set value for the TooltipFontFamily [▶ 295] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipFontFamily [▶ 1615]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipFontFamily
3.106

[Function]

```
public setTooltipFontFamily(newValue: string | null): void;
```

This function sets a new value for the TooltipFontFamily [▶ 295] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontFamily [▶ 1615]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getTooltipFontSize 3.107

[Function]

```
public getTooltipFontSize(): number;
```

This function returns the currently set value for the TooltipFontSize [▶ 296] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipFontSize [▶ 1616]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipFontSize 3.108

[Function]

```
public setTooltipFontSize(newValue: number | null): void;
```

This function sets a new value for the TooltipFontSize [▶ 296] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontSize [▶ 1616]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getTooltipFontSizeUnit
3.109

[Function]

```
public getTooltipFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TooltipFontSizeUnit [▶ 296] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTooltipFontSizeUnit [▶ 1617]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipFontSizeUnit
3.110

[Function]

```
public setTooltipFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TooltipFontSizeUnit [▶ 296] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontSizeUnit [▶ 1617]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 getTooltipFontWeight
3.111

[Function]

```
public getTooltipFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the TooltipFontWeight [▶ 296] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setTooltipFontWeight [▶ 1618]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 setTooltipFontWeight
3.112

[Function]

```
public setTooltipFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the TooltipFontWeight [▶ 296] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontWeight [▶ 1618]

Origin: TcHmiLineChart [▶ 281]

Note: Available since version 1.10

15.1.1.2.2.1 TcHmiPolygon
4

[Class]

Closed shape formed from a set of interconnected straight lines. The last point is automatically connected to the first point.

Attribute functions

Name	Origin	Description
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.

Name	Origin	Description
getFillColor [▶ 1625]	TcHmiPolygon	Getter for the FillColor attribute.
setFillColor [▶ 1625]	TcHmiPolygon	Setter for the FillColor attribute.
getStrokeColor [▶ 1626]	TcHmiPolygon	Getter for the StrokeColor attribute.
setStrokeColor [▶ 1626]	TcHmiPolygon	Setter for the StrokeColor attribute.
getPoints [▶ 1626]	TcHmiPolygon	Getter for the Points attribute.
setPoints [▶ 1627]	TcHmiPolygon	Setter for the Points attribute.
getFillMode [▶ 1631]	TcHmiPolygon	Getter for the FillMode attribute.
setFillMode [▶ 1631]	TcHmiPolygon	Setter for the FillMode attribute.
getScaleMode [▶ 1630]	TcHmiPolygon	Getter for the ScaleMode attribute.
setScaleMode [▶ 1630]	TcHmiPolygon	Setter for the ScaleMode attribute.
getStrokeStyle [▶ 1627]	TcHmiPolygon	Getter for the StrokeStyle attribute.
setStrokeStyle [▶ 1628]	TcHmiPolygon	Setter for the StrokeStyle attribute.
getStrokeThickness [▶ 1628]	TcHmiPolygon	Getter for the StrokeThickness attribute.
setStrokeThickness [▶ 1629]	TcHmiPolygon	Setter for the StrokeThickness attribute.
getStrokeThicknessUnit [▶ 1629]	TcHmiPolygon	Getter for the StrokeThicknessUnit attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.

Name	Origin	Description
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Polygon

See also

Reference [▶ 329] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getFillColor 4.1

[Function]

```
public getFillColor(): TcHmi.Color | null;
```

This function returns the currently set value for the FillColor [▶ 335] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setFillColor [▶ 1625]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setFillColor 4.2

[Function]

```
public setFillColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the FillColor [▶ 335] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getFillColor [▶ 1625]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeColor
4.3

[Function]

```
public getStrokeColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the StrokeColor [▶ 336] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeColor [▶ 1626]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeColor
4.4

[Function]

```
public setStrokeColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the StrokeColor [▶ 336] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeColor [▶ 1626]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getPoints
4.5

[Function]

```
public getPoints(): string | null;
```

This function returns the currently set value for the Points [▶ 335] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setPoints [▶ 1627]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setPoints
4.6

[Function]

```
public setPoints(newValue: string | null): void;
```

This function sets a new value for the Points [▶ 335] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getPoints [▶ 1626]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeStyle
4.7

[Function]

```
public getStrokeStyle(): string;
```

This function returns the currently set value for the StrokeStyle [▶ 336] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeStyle [▶ 1628]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeStyle
4.8

[Function]

```
public setStrokeStyle(newValue: string | null): void;
```

This function sets a new value for the StrokeStyle [▶ 336] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeStyle [▶ 1627]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeThickness
4.9

[Function]

```
public getStrokeThickness(): number | null;
```

This function returns the currently set value for the StrokeThickness [▶ 336] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeThickness [▶ 1629]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeThickness
4.10

[Function]

```
public setStrokeThickness(newValue: number | null): void;
```

This function sets a new value for the StrokeThickness [▶ 336] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeThickness [▶ 1628]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeThicknessUnit
4.11

[Function]

```
public getStrokeThicknessUnit(): 'px';
```

This function returns the currently set value for the StrokeThicknessUnit [▶ 337] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getScaleMode
4.12

[Function]

```
public getScaleMode(): TcHmi.ScaleModeString;
```

This function returns the currently set value for the ScaleMode [▶ 335] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.ScaleModeString [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setScaleMode [▶ 1630]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setScaleMode
4.13

[Function]

```
public setScaleMode(newValue: TcHmi.ScaleModeString | null): void;
```

This function sets a new value for the ScaleMode [▶ 335] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.ScaleModeString [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getScaleMode [▶ 1630]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 getFillMode
4.14

[Function]

```
public getFillMode(): 'NonZero' | 'EvenOdd';
```

This function returns the currently set value for the FillMode [▶ 335] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'NonZero' | 'EvenOdd'	The current value of the attribute.

See also

Attribute setter: setFillMode [▶ 1631]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 setFillMode
4.15

[Function]

```
public setFillMode(newValue: 'NonZero' | 'EvenOdd' | null): void;
```

This function sets a new value for the FillMode [▶ 335] attribute.

Parameter

Name	Type	Description
newValue	'NonZero', 'EvenOdd', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getFillMode [▶ 1631]

Origin: TcHmiPolygon [▶ 329]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiRadialGauge
5

[Class]

Radial display

Attribute functions

Name	Origin	Description
getProgressBackgroundColor [▶ 1640]	TcHmiRadialGauge	Getter for the ProgressBackgroundColor attribute.
setProgressBackgroundColor [▶ 1641]	TcHmiRadialGauge	Setter for the ProgressBackgroundColor attribute.
getProgressForegroundColor [▶ 1639]	TcHmiRadialGauge	Getter for the ProgressForegroundColor attribute.
setProgressForegroundColor [▶ 1640]	TcHmiRadialGauge	Setter for the ProgressForegroundColor attribute.
getGaugeBackgroundColor [▶ 1641]	TcHmiRadialGauge	Getter for the GaugeBackgroundColor attribute.
setGaugeBackgroundColor [▶ 1642]	TcHmiRadialGauge	Setter for the GaugeBackgroundColor attribute.
getKnobColor [▶ 1643]	TcHmiRadialGauge	Getter for the KnobColor attribute.
setKnobColor [▶ 1644]	TcHmiRadialGauge	Setter for the KnobColor attribute.
getKnobArrowsColor [▶ 1642]	TcHmiRadialGauge	Getter for the KnobArrowsColor attribute.
setKnobArrowsColor [▶ 1643]	TcHmiRadialGauge	Setter for the KnobArrowsColor attribute.
getValueColor [▶ 1644]	TcHmiRadialGauge	Getter for the ValueColor attribute.
setValueColor [▶ 1644]	TcHmiRadialGauge	Setter for the ValueColor attribute.
getLabelColor [▶ 1645]	TcHmiRadialGauge	Getter for the LabelColor attribute.
setLabelColor [▶ 1645]	TcHmiRadialGauge	Setter for the LabelColor attribute.
getUnitColor [▶ 1646]	TcHmiRadialGauge	Getter for the UnitColor attribute.
setUnitColor [▶ 1646]	TcHmiRadialGauge	Setter for the UnitColor attribute.
getTickColor [▶ 1647]	TcHmiRadialGauge	Getter for the TickColor attribute.
setTickColor [▶ 1647]	TcHmiRadialGauge	Setter for the TickColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getStartPosition [▶ 1649]	TcHmiRadialGauge	Getter for the StartPosition attribute.
setStartPosition [▶ 1649]	TcHmiRadialGauge	Setter for the StartPosition attribute.
getEndPosition [▶ 1648]	TcHmiRadialGauge	Getter for the EndPosition attribute.
setEndPosition [▶ 1648]	TcHmiRadialGauge	Setter for the EndPosition attribute.

Name	Origin	Description
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.

Name	Origin	Description
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getValue [▶ 1650]	TcHmiRadialGauge	Getter for the Value attribute.
setValue [▶ 1650]	TcHmiRadialGauge	Setter for the Value attribute.
getEditable [▶ 1650]	TcHmiRadialGauge	Getter for the Editable attribute.
setEditable [▶ 1651]	TcHmiRadialGauge	Setter for the Editable attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.

Name	Origin	Description
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getShowLabels [▶ 1663]	TcHmiRadialGauge	Getter for the ShowLabels attribute.
setShowLabels [▶ 1664]	TcHmiRadialGauge	Setter for the ShowLabels attribute.

Name	Origin	Description
getLabelPosition [▶ 1664]	TcHmiRadialGauge	Getter for the LabelPosition attribute.
setLabelPosition [▶ 1665]	TcHmiRadialGauge	Setter for the LabelPosition attribute.
getLabelRange [▶ 1665]	TcHmiRadialGauge	Getter for the LabelRange attribute.
setLabelRange [▶ 1666]	TcHmiRadialGauge	Setter for the LabelRange attribute.
getLabelFormat [▶ 1666]	TcHmiRadialGauge	Getter for the LabelFormat attribute.
setLabelFormat [▶ 1667]	TcHmiRadialGauge	Setter for the LabelFormat attribute.
getLabelFontFamily [▶ 1667]	TcHmiRadialGauge	Getter for the LabelFontFamily attribute.
setLabelFontFamily [▶ 1668]	TcHmiRadialGauge	Setter for the LabelFontFamily attribute.
getLabelFontSize [▶ 1668]	TcHmiRadialGauge	Getter for the LabelFontSize attribute.
setLabelFontSize [▶ 1668]	TcHmiRadialGauge	Setter for the LabelFontSize attribute.
getLabelFontSizeUnit [▶ 1669]	TcHmiRadialGauge	Getter for the LabelFontSizeUnit attribute.
setLabelFontSizeUnit [▶ 1669]	TcHmiRadialGauge	Setter for the LabelFontSizeUnit attribute.
getLabelFontStyle [▶ 1670]	TcHmiRadialGauge	Getter for the LabelFontStyle attribute.
setLabelFontStyle [▶ 1670]	TcHmiRadialGauge	Setter for the LabelFontStyle attribute.
getLabelFontWeight [▶ 1671]	TcHmiRadialGauge	Getter for the LabelFontWeight attribute.
setLabelFontWeight [▶ 1671]	TcHmiRadialGauge	Setter for the LabelFontWeight attribute.
getShowTicks [▶ 1672]	TcHmiRadialGauge	Getter for the ShowTicks attribute.
setShowTicks [▶ 1672]	TcHmiRadialGauge	Setter for the ShowTicks attribute.
getSubTickPosition [▶ 1673]	TcHmiRadialGauge	Getter for the SubTickPosition attribute.
setSubTickPosition [▶ 1673]	TcHmiRadialGauge	Setter for the SubTickPosition attribute.
getTickDefinition [▶ 1674]	TcHmiRadialGauge	Getter for the TickDefinition attribute.
setTickDefinition [▶ 1674]	TcHmiRadialGauge	Setter for the TickDefinition attribute.
getUnit [▶ 1675]	TcHmiRadialGauge	Getter for the Unit attribute.
setUnit [▶ 1675]	TcHmiRadialGauge	Setter for the Unit attribute.
getUnitFontFamily [▶ 1676]	TcHmiRadialGauge	Getter for the UnitFontFamily attribute.
setUnitFontFamily [▶ 1677]	TcHmiRadialGauge	Setter for the UnitFontFamily attribute.
getUnitFontSize [▶ 1675]	TcHmiRadialGauge	Getter for the UnitFontSize attribute.
setUnitFontSize [▶ 1676]	TcHmiRadialGauge	Setter for the UnitFontSize attribute.
getUnitFontSizeUnit [▶ 1679]	TcHmiRadialGauge	Getter for the UnitFontSizeUnit attribute.

Name	Origin	Description
setUnitFontSizeUnit [▶ 1680]	TcHmiRadialGauge	Setter for the UnitFontSizeUnit attribute.
getUnitFontStyle [▶ 1677]	TcHmiRadialGauge	Getter for the UnitFontStyle attribute.
setUnitFontStyle [▶ 1678]	TcHmiRadialGauge	Setter for the UnitFontStyle attribute.
getUnitFontWeight [▶ 1678]	TcHmiRadialGauge	Getter for the UnitFontWeight attribute.
setUnitFontWeight [▶ 1679]	TcHmiRadialGauge	Setter for the UnitFontWeight attribute.
getBaseAnimationTime [▶ 1680]	TcHmiRadialGauge	Getter for the BaseAnimationTime attribute.
setBaseAnimationTime [▶ 1681]	TcHmiRadialGauge	Setter for the BaseAnimationTime attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Radial Gauge

See also

Reference [▶ 337] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getProgressForegroundColor
5.1

[Function]

```
public getProgressForegroundColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the ProgressForegroundColor [▶ 351] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setProgressForegroundColor [▶ 1640]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setProgressForegroundColor 5.2

[Function]

```
public setProgressForegroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the ProgressForegroundColor [▶ 351] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getProgressForegroundColor [▶ 1639]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getProgressBackgroundColor 5.3

[Function]

```
public getProgressBackgroundColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the ProgressBackgroundColor [▶ 350] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setProgressBackgroundColor [▶ 1641]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setProgressBackgroundColor 5.4

[Function]

```
public setProgressBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the ProgressBackgroundColor [▶ 350] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getProgressBackgroundColor [▶ 1640]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getGaugeBackgroundColor 5.5

[Function]

```
public getGaugeBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the GaugeBackgroundColor [▶ 346] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGaugeBackgroundColor [▶ 1642]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setGaugeBackgroundColor
5.6

[Function]

```
public setGaugeBackgroundColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the GaugeBackgroundColor [▶ 346] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGaugeBackgroundColor [▶ 1641]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getKnobArrowsColor
5.7

[Function]

```
public getKnobArrowsColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the KnobArrowsColor [▶ 346] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKnobArrowsColor [▶ 1643]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setKnobArrowsColor
5.8

[Function]

```
public setKnobArrowsColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the KnobArrowsColor [▶ 346] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKnobArrowsColor [▶ 1642]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getKnobColor
5.9

[Function]

```
public getKnobColor(): TcHmi.Color | null;
```

This function returns the currently set value for the KnobColor [▶ 346] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setKnobColor [▶ 1644]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setKnobColor
5.10

[Function]

```
public setKnobColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the KnobColor [▶ 346] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getKnobColor [▶ 1643]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueColor
5.11

[Function]

```
public getValueColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the ValueColor [▶ 355] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueColor [▶ 1644]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueColor
5.12

[Function]

```
public setValueColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the ValueColor [▶ 355] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueColor [▶ 1644]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelColor
5.13

[Function]

```
public getLabelColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the LabelColor [▶ 347] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelColor [▶ 1645]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelColor
5.14

[Function]

```
public setLabelColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the LabelColor [▶ 347] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelColor [▶ 1645]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitColor
5.15

[Function]

```
public getUnitColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the UnitColor [▶ 354] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnitColor [▶ 1646]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitColor
5.16

[Function]

```
public setUnitColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the UnitColor [▶ 354] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitColor [▶ 1646]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getTickColor
5.17

[Function]

```
public getTickColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TickColor [▶ 353] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTickColor [▶ 1647]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setTickColor
5.18

[Function]

```
public setTickColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TickColor [▶ 353] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTickColor [▶ 1647]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getEndPosition
5.19

[Function]

```
public getEndPosition(): number;
```

This function returns the currently set value for the EndPosition [▶ 346] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setEndPosition [▶ 1648]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setEndPosition
5.20

[Function]

```
public setEndPosition(newValue: number | null): void;
```

This function sets a new value for the EndPosition [▶ 346] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getEndPosition [▶ 1648]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getStartPosition
5.21

[Function]

```
public getStartPosition(): number;
```

This function returns the currently set value for the StartPosition [▶ 352] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStartPosition [▶ 1649]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setStartPosition
5.22

[Function]

```
public setStartPosition(newValue: number | null): void;
```

This function sets a new value for the StartPosition [▶ 352] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStartPosition [▶ 1649]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValue
5.23

[Function]

```
public getValue(): number;
```

This function returns the currently set value for the Value [▶ 355] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValue [▶ 1650]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValue
5.24

[Function]

```
public setValue(newValue: number | null): void;
```

This function sets a new value for the Value [▶ 355] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValue [▶ 1650]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getEditable
5.25

[Function]

```
public getEditable(): boolean;
```

This function returns the currently set value for the Editable [▶ 345] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setEditable [▶ 1651]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setEditable
5.26

[Function]

```
public setEditable(newValue: boolean | null): void;
```

This function sets a new value for the Editable [▶ 345] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getEditable [▶ 1650]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getMinValue
5.27

[Function]

```
public getMinValue(): number;
```

This function returns the currently set value for the MinValue [▶ 350] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMinValue [▶ 1652]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setMinValue
5.28

[Function]

```
public setMinValue(newValue: number | null): void;
```

This function sets a new value for the MinValue [▶ 350] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMinValue [▶ 1651]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getMaxValue
5.29

[Function]

```
public getMaxValue(): number;
```

This function returns the currently set value for the MaxValue [▶ 350] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMaxValue [▶ 1653]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setMaxValue
5.30

[Function]

```
public setMaxValue(newValue: number | null): void;
```

This function sets a new value for the MaxValue [▶ 350] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMaxValue [▶ 1652]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getStep
5.31

[Function]

```
public getStep(): number;
```

This function returns the currently set value for the Step [▶ 352] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStep [▶ 1654]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setStep
5.32

[Function]

```
public setStep(newValue: number | null): void;
```

This function sets a new value for the Step [▶ 352] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStep [▶ 1653]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getRange
5.33

[Function]

```
public getRange(): TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range[] | null;
```

This function returns the currently set value for the Range [▶ 351] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range [▶ 1556][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRange [▶ 1655]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setRange
5.34

[Function]

```
public setRange(newValue: TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range[] | null): void;
```

This function sets a new value for the Range [▶ 351] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range [▶ 1556][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRange [▶ 1654]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getShowValueText
5.35

[Function]

```
public getShowValueText(): boolean;
```

This function returns the currently set value for the ShowValueText [▶ 352] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowValueText [▶ 1656]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setShowValueText
5.36

[Function]

```
public setShowValueText(newValue: boolean | null): void;
```

This function sets a new value for the ShowValueText [▶ 352] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowValueText [▶ 1655]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getStartFromZero
5.37

[Function]

```
public getStartFromZero(): boolean;
```

This function returns the currently set value for the StartFromZero [▶ 352] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setStartFromZero [▶ 1656]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setStartFromZero
5.38

[Function]

```
public setStartFromZero(newValue: boolean | null): void;
```

This function sets a new value for the StartFromZero [▶ 352] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStartFromZero [▶ 1656]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontStyle
5.39

[Function]

```
public getValueFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the ValueFontStyle [▶ 356] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setValueFontStyle [▶ 1657]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontStyle
5.40

[Function]

```
public setValueFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the ValueFontStyle [▶ 356] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontStyle [▶ 1657]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getClickAnywhereToEdit 5.41

[Function]

```
public getClickAnywhereToEdit(): boolean;
```

This function returns the currently set value for the ClickAnywhereToEdit [▶ 345] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setClickAnywhereToEdit [▶ 1658]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setClickAnywhereToEdit 5.42

[Function]

```
public setClickAnywhereToEdit(newValue: boolean | null): void;
```

This function sets a new value for the ClickAnywhereToEdit [▶ 345] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getClickAnywhereToEdit [▶ 1658]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontFamily
5.43

[Function]

```
public getValueFontFamily(): string | null;
```

This function returns the currently set value for the ValueFontFamily [▶ 356] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueFontFamily [▶ 1659]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontFamily
5.44

[Function]

```
public setValueFontFamily(newValue: string | null): void;
```

This function sets a new value for the ValueFontFamily [▶ 356] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontFamily [▶ 1659]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontSize
5.45

[Function]

```
public getValueFontSize(): number;
```

This function returns the currently set value for the ValueFontSize [▶ 356] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueFontSize [▶ 1660]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontSize
5.46

[Function]

```
public setValueFontSize(newValue: number | null): void;
```

This function sets a new value for the ValueFontSize [▶ 356] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontSize [▶ 1660]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontSizeUnit
5.47

[Function]

```
public getValueFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the ValueFontSizeUnit [▶ 356] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setValueFontSizeUnit [▶ 1661]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontSizeUnit
5.48

[Function]

```
public setValueFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the ValueFontSizeUnit [▶ 356] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontSizeUnit [▶ 1661]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFontWeight
5.49

[Function]

```
public getValueFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the ValueFontWeight [▶ 357] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setValueFontWeight [▶ 1662]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFontWeight
5.50

[Function]

```
public setValueFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the ValueFontWeight [▶ 357] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFontWeight [▶ 1662]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getValueFormat
5.51

[Function]

```
public getValueFormat(): TcHmi.IFunction | null;
```

This function returns the currently set value for the ValueFormat [▶ 357] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setValueFormat [▶ 1663]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setValueFormat
5.52

[Function]

```
public setValueFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the ValueFormat [▶ 357] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getValueFormat [▶ 1662]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getShowLabels
5.53

[Function]

```
public getShowLabels(): boolean;
```

This function returns the currently set value for the ShowLabels [▶ 351] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowLabels [▶ 1664]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setShowLabels
5.54

[Function]

```
public setShowLabels(newValue: boolean | null): void;
```

This function sets a new value for the ShowLabels [▶ 351] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowLabels [▶ 1663]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelPosition
5.55

[Function]

```
public getLabelPosition(): 'Top' | 'Bottom' | 'Left' | 'Right';
```

This function returns the currently set value for the LabelPosition [▶ 349] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Top', 'Bottom', 'Left', 'Right'	The current value of the attribute.

See also

Attribute setter: setLabelPosition [▶ 1665]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelPosition
5.56

[Function]

```
public setLabelPosition(newValue: 'Top' | 'Bottom' | 'Left' | 'Right' | null): void;
```

This function sets a new value for the LabelPosition [▶ 349] attribute.

Parameter

Name	Type	Description
newValue	'Top', 'Bottom', 'Left', 'Right', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelPosition [▶ 1664]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelRange
5.57

[Function]

```
public getLabelRange(): number | null;
```

This function returns the currently set value for the LabelRange [▶ 350] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelRange [▶ 1666]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelRange
5.58

[Function]

```
public setLabelRange(newValue: number | null): void;
```

This function sets a new value for the LabelRange [▶ 350] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelRange [▶ 1665]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFormat
5.59

[Function]

```
public getLabelFormat(): TcHmi.IFunction | null;
```

This function returns the currently set value for the LabelFormat [▶ 348] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFormat [▶ 1667]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFormat
5.60

[Function]

```
public setLabelFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the LabelFormat [▶ 348] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFormat [▶ 1666]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontFamily
5.61

[Function]

```
public getLabelFontFamily(): string | null;
```

This function returns the currently set value for the LabelFontFamily [▶ 347] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFontFamily [▶ 1668]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontFamily
5.62

[Function]

```
public setLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the LabelFontFamily [▶ 347] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontFamily [▶ 1667]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontSize
5.63

[Function]

```
public getLabelFontSize(): number;
```

This function returns the currently set value for the LabelFontSize [▶ 347] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLabelFontSize [▶ 1668]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontSize
5.64

[Function]

```
public setLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the LabelFontSize [▶ 347] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontSize [▶ 1668]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontSizeUnit
5.65

[Function]

```
public getLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the LabelFontSizeUnit [▶ 348] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setLabelFontSizeUnit [▶ 1669]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontSizeUnit
5.66

[Function]

```
public setLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the LabelFontSizeUnit [▶ 348] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontSizeUnit [▶ 1669]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontStyle
5.67

[Function]

```
public getLabelFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the LabelFontStyle [▶ 348] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setLabelFontStyle [▶ 1670]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontStyle
5.68

[Function]

```
public setLabelFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the LabelFontStyle [▶ 348] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontStyle [▶ 1670]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getLabelFontWeight
5.69

[Function]

```
public getLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the LabelFontWeight [▶ 348] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setLabelFontWeight [▶ 1671]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setLabelFontWeight
5.70

[Function]

```
public setLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the LabelFontWeight [▶ 348] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLabelFontWeight [▶ 1671]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getShowTicks
5.71

[Function]

```
public getShowTicks(): boolean;
```

This function returns the currently set value for the ShowTicks [▶ 351] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowTicks [▶ 1672]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setShowTicks
5.72

[Function]

```
public setShowTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowTicks [▶ 351] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowTicks [▶ 1672]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getSubTickPosition
5.73

[Function]

```
public getSubTickPosition(): 'Top' | 'Bottom' | 'Left' | 'Right';
```

This function returns the currently set value for the SubTickPosition [▶ 353] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'Top', 'Bottom', 'Left', 'Right'	The current value of the attribute.

See also

Attribute setter: setSubTickPosition [▶ 1673]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setSubTickPosition
5.74

[Function]

```
public setSubTickPosition(newValue: 'Top' | 'Bottom' | 'Left' | 'Right' | null): void;
```

This function sets a new value for the SubTickPosition [▶ 353] attribute.

Parameter

Name	Type	Description
newValue	'Top', 'Bottom', 'Left', 'Right', null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubTickPosition [▶ 1673]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getTickDefinition
5.75

[Function]

```
public getTickDefinition(): TcHmi.Controls.Beckhoff.TcHmiRadialGauge.TickDefinition | null;
```

This function returns the currently set value for the TickDefinition [▶ 353] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiRadialGauge.TickDefinition [▶ 1681], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTickDefinition [▶ 1674]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setTickDefinition
5.76

[Function]

```
public setTickDefinition(newValue: TcHmi.Controls.Beckhoff.TcHmiRadialGauge.TickDefinition | null):
void;
```

This function sets a new value for the TickDefinition [▶ 353] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiRadialGauge.TickDefinition [▶ 1681], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTickDefinition [▶ 1674]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnit
5.77

[Function]

```
public getUnit(): string;
```

This function returns the currently set value for the Unit [▶ 345] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnit [▶ 1675]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnit
5.78

[Function]

```
public setUnit(newValue: string | null): void;
```

This function sets a new value for the Unit [▶ 345] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnit [▶ 1675]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontSize
5.79

[Function]

```
public getUnitFontSize(): number;
```

This function returns the currently set value for the UnitFontSize [▶ 354] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnitFontSize [▶ 1676]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontSize
5.80

[Function]

```
public setUnitFontSize(newValue: number | null): void;
```

This function sets a new value for the UnitFontSize [▶ 354] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontSize [▶ 1675]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontFamily
5.81

[Function]

```
public getUnitFontFamily(): string | null;
```

This function returns the currently set value for the UnitFontFamily [▶ 354] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setUnitFontFamily [▶ 1677]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontFamily
5.82

[Function]

```
public setUnitFontFamily(newValue: string | null): void;
```

This function sets a new value for the UnitFontFamily [▶ 354] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontFamily [▶ 1676]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontStyle
5.83

[Function]

```
public getUnitFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the UnitFontStyle [▶ 355] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setUnitFontStyle [▶ 1678]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontStyle
5.84

[Function]

```
public setUnitFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the UnitFontStyle [▶ 355] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontStyle [▶ 1677]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontWeight
5.85

[Function]

```
public getUnitFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the UnitFontWeight [▶ 355] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setUnitFontWeight [▶ 1679]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontWeight
5.86

[Function]

```
public setUnitFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the UnitFontWeight [▶ 355] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontWeight [▶ 1678]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getUnitFontSizeUnit
5.87

[Function]

```
public getUnitFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the UnitFontSizeUnit [▶ 354] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setUnitFontSizeUnit [▶ 1680]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setUnitFontSizeUnit
5.88

[Function]

```
public setUnitFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the UnitFontSizeUnit [▶ 354] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getUnitFontSizeUnit [▶ 1679]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 getBaseAnimationTime
5.89

[Function]

```
public getBaseAnimationTime(): number;
```

This function returns the currently set value for the BaseAnimationTime [▶ 345] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setBaseAnimationTime [▶ 1681]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 setBaseAnimationTime
5.90

[Function]

```
public setBaseAnimationTime(newValue: number | null): void;
```

This function sets a new value for the BaseAnimationTime [▶ 345] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getBaseAnimationTime [▶ 1680]

Origin: TcHmiRadialGauge [▶ 337]

Note: Available since version 1.8

15.1.1.2.2.1 TickDefinition
5.91

[Interface]

```
export interface TickDefinition {
    mainTickRange: number;
    subTickRange: number;
}
```

Properties

Name	Type	Description
mainTickRange	number [▶ 2108]	Distance of the main axis markers
subTickRange	number [▶ 2108]	Distance of the secondary axis markers

See also

Scheme: tchmi:framework#/definitions/GaugeTickDefinition [▶ 2237]

Note: Available since version 1.8

15.1.1.2.2.1 Range
5.92

[Interface]

```
export interface Range {
    color: TcHmi.SolidColor;
    start: number;
    end: number;
}
```

Properties

Name	Type	Description
color	TcHmi.SolidColor [▶ 2084]	Color of this entry
start	number [▶ 2108]	Starting value for this color range
end	number [▶ 2108]	End value for this color range

See also

Scheme: tchmi:framework#/definitions/GaugeRangeDefinitionList [▶ 2237]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiRectangle
6

[Class]

Rectangle

Attribute functions

Name	Origin	Description
getFillColor [▶ 1688]	TcHmiRectangle	Getter for the FillColor attribute.
setFillColor [▶ 1688]	TcHmiRectangle	Setter for the FillColor attribute.
getStrokeColor [▶ 1689]	TcHmiRectangle	Getter for the StrokeColor attribute.
setStrokeColor [▶ 1689]	TcHmiRectangle	Setter for the StrokeColor attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.

Name	Origin	Description
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.

Name	Origin	Description
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.

Name	Origin	Description
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getStrokeThickness [▶ 1689]	TcHmiRectangle	Getter for the StrokeThickness attribute.
setStrokeThickness [▶ 1690]	TcHmiRectangle	Setter for the StrokeThickness attribute.
getStrokeThicknessUnit [▶ 1690]	TcHmiRectangle	Getter for the StrokeThicknessUnit attribute.
getStrokeStyle [▶ 1691]	TcHmiRectangle	Getter for the StrokeStyle attribute.
setStrokeStyle [▶ 1691]	TcHmiRectangle	Setter for the StrokeStyle attribute.
getRadiusX [▶ 1692]	TcHmiRectangle	Getter for the RadiusX attribute.
setRadiusX [▶ 1692]	TcHmiRectangle	Setter for the RadiusX attribute.
getRadiusXUnit [▶ 1694]	TcHmiRectangle	Getter for the RadiusXUnit attribute.
getRadiusY [▶ 1693]	TcHmiRectangle	Getter for the RadiusY attribute.
setRadiusY [▶ 1694]	TcHmiRectangle	Setter for the RadiusY attribute.
getRadiusYUnit [▶ 1693]	TcHmiRectangle	Getter for the RadiusYUnit attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Rectangle

See also

Reference [▶ 359] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getFillColor
6.1

[Function]

```
public getFillColor(): TcHmi.Color | null;
```

This function returns the currently set value for the FillColor [▶ 366] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setFillColor [▶ 1688]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 setFillColor
6.2

[Function]

```
public setFillColor(newValue: TcHmi.Color | null): void;
```

This function sets a new value for the FillColor [▶ 366] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getFillColor [▶ 1688]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeColor
6.3

[Function]

```
public getStrokeColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the StrokeColor [▶ 366] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeColor [▶ 1689]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeColor
6.4

[Function]

```
public setStrokeColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the StrokeColor [▶ 366] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeColor [▶ 1689]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeThickness
6.5

[Function]

```
public getStrokeThickness(): number | null;
```

This function returns the currently set value for the StrokeThickness [▶ 367] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeThickness [▶ 1690]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeThickness
6.6

[Function]

```
public setStrokeThickness(newValue: number | null): void;
```

This function sets a new value for the StrokeThickness [▶ 367] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeThickness [▶ 1689]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeThicknessUnit
6.7

[Function]

```
public getStrokeThicknessUnit(): 'px';
```

This function returns the currently set value for the StrokeThicknessUnit [▶ 367] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getStrokeStyle
6.8

[Function]

```
public getStrokeStyle(): string;
```

This function returns the currently set value for the StrokeStyle [▶ 366] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStrokeStyle [▶ 1691]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 setStrokeStyle
6.9

[Function]

```
public setStrokeStyle(newValue: string | null): void;
```

This function sets a new value for the StrokeStyle [▶ 366] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStrokeStyle [▶ 1691]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getRadiusX
6.10

[Function]

```
public getRadiusX(): number;
```

This function returns the currently set value for the RadiusX [▶ 365] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRadiusX [▶ 1692]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 setRadiusX
6.11

[Function]

```
public setRadiusX(newValue: number | null): void;
```

This function sets a new value for the RadiusX [▶ 365] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRadiusX [▶ 1692]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getRadiusYUnit
6.12

[Function]

```
public getRadiusYUnit(): 'px';
```

This function returns the currently set value for the RadiusYUnit [▶ 365] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getRadiusY
6.13

[Function]

```
public getRadiusY(): number;
```

This function returns the currently set value for the RadiusY [▶ 365] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setRadiusY [▶ 1694]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 setRadiusY
6.14

[Function]

```
public setRadiusY(newValue: number | null): void;
```

This function sets a new value for the RadiusY [▶ 365] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getRadiusY [▶ 1693]

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 getRadiusXUnit
6.15

[Function]

```
public getRadiusXUnit(): 'px';
```

This function returns the currently set value for the RadiusXUnit [▶ 365] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
'px'	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiRectangle [▶ 359]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiTextblock
7

[Class]

Text block for displaying text

Attribute functions

Name	Origin	Description
getTextColor [▶ 1700]	TcHmiTextblock	Getter for the TextColor attribute.
setTextColor [▶ 1700]	TcHmiTextblock	Setter for the TextColor attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.

Name	Origin	Description
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.

getText [▶ 1702]	TcHmiTextblock	Getter for the Text attribute.
setText [▶ 1702]	TcHmiTextblock	Setter for the Text attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.

setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getTextHorizontalAlignment [▶ 1706]	TcHmiTextblock	Getter for the TextHorizontalAlignment attribute.
setTextHorizontalAlignment [▶ 1706]	TcHmiTextblock	Setter for the TextHorizontalAlignment attribute.
getTextVerticalAlignment [▶ 1703]	TcHmiTextblock	Getter for the TextVerticalAlignment attribute.
setTextVerticalAlignment [▶ 1703]	TcHmiTextblock	Setter for the TextVerticalAlignment attribute.
getContentPadding [▶ 1706]	TcHmiTextblock	Getter for the ContentPadding attribute.
setContentPadding [▶ 1707]	TcHmiTextblock	Setter for the ContentPadding attribute.
getWordWrap [▶ 1707]	TcHmiTextblock	Getter for the WordWrap attribute.
setWordWrap [▶ 1708]	TcHmiTextblock	Setter for the WordWrap attribute.
getTextFontFamily [▶ 1710]	TcHmiTextblock	Getter for the TextFontFamily attribute.
setTextFontFamily [▶ 1711]	TcHmiTextblock	Setter for the TextFontFamily attribute.
getTextFontSize [▶ 1704]	TcHmiTextblock	Getter for the TextFontSize attribute.
setTextFontSize [▶ 1704]	TcHmiTextblock	Setter for the TextFontSize attribute.
getTextFontSizeUnit [▶ 1705]	TcHmiTextblock	Getter for the TextFontSizeUnit attribute.
setTextFontSizeUnit [▶ 1705]	TcHmiTextblock	Setter for the TextFontSizeUnit attribute.
getTextFontStyle [▶ 1708]	TcHmiTextblock	Getter for the TextFontStyle attribute.
setTextFontStyle [▶ 1709]	TcHmiTextblock	Setter for the TextFontStyle attribute.
getTextFontWeight [▶ 1709]	TcHmiTextblock	Getter for the TextFontWeight attribute.
setTextFontWeight [▶ 1710]	TcHmiTextblock	Setter for the TextFontWeight attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Textblock

See also

Reference [▶ 367] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getTextColor
7.1

[Function]

```
public getTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TextColor [▶ 375] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextColor [▶ 1700]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextColor
7.2

[Function]

```
public setTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TextColor [▶ 375] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextColor [▶ 1700]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getHeightMode
7.3

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 374] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1701]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setHeightMode
7.4

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 374] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1701]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getText
7.5

[Function]

```
public getText(): string;
```

This function returns the currently set value for the Text [▶ 375] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setText [▶ 1702]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setText
7.6

[Function]

```
public setText(newValue: string | null): void;
```

This function sets a new value for the Text [▶ 375] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getText [▶ 1702]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextVerticalAlignment
7.7

[Function]

```
public getTextVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the TextVerticalAlignment [▶ 375] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextVerticalAlignment [▶ 1703]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextVerticalAlignment
7.8

[Function]

```
public setTextVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the TextVerticalAlignment [▶ 375] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextVerticalAlignment [▶ 1703]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontSize
7.9

[Function]

```
public getTextFontSize(): number;
```

This function returns the currently set value for the TextFontSize [▶ 373] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontSize [▶ 1704]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontSize
7.10

[Function]

```
public setTextFontSize(newValue: number | null): void;
```

This function sets a new value for the TextFontSize [▶ 373] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSize [▶ 1704]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontSizeUnit 7.11

[Function]

```
public getTextFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TextFontSizeUnit [▶ 373] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTextFontSizeUnit [▶ 1705]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontSizeUnit 7.12

[Function]

```
public setTextFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TextFontSizeUnit [▶ 373] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSizeUnit [▶ 1705]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextHorizontalAlignment 7.13

[Function]

```
public getTextHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the TextHorizontalAlignment [▶ 375] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextHorizontalAlignment [▶ 1706]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextHorizontalAlignment 7.14

[Function]

```
public setTextHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the TextHorizontalAlignment [▶ 375] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextHorizontalAlignment [▶ 1706]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getContentPadding 7.15

[Function]

```
public getContentPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the ContentPadding [▶ 373] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setContentPadding [▶ 1707]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setContentPadding
7.16

[Function]

```
public setContentPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the ContentPadding [▶ 373] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getContentPadding [▶ 1706]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getWordWrap
7.17

[Function]

```
public getWordWrap(): boolean;
```

This function returns the currently set value for the WordWrap [▶ 376] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setWordWrap [▶ 1708]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setWordWrap
7.18

[Function]

```
public setWordWrap(newValue: boolean | null): void;
```

This function sets a new value for the WordWrap [▶ 376] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWordWrap [▶ 1707]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontStyle
7.19

[Function]

```
public getTextFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the TextFontStyle [▶ 374] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setTextFontStyle [▶ 1709]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontStyle
7.20

[Function]

```
public setTextFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the TextFontStyle [▶ 374] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontStyle [▶ 1708]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontWeight
7.21

[Function]

```
public getTextFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the TextFontWeight [▶ 374] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setTextFontWeight [▶ 1710]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontWeight
7.22

[Function]

```
public setTextFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the TextFontWeight [▶ 374] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontWeight [▶ 1709]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontFamily
7.23

[Function]

```
public getTextFontFamily(): string | null;
```

This function returns the currently set value for the TextFontFamily [▶ 373] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontFamily [▶ 1711]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontFamily
7.24

[Function]

```
public setTextFontFamily(newValue: string | null): void;
```

This function sets a new value for the TextFontFamily [▶ 373] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontFamily [▶ 1710]

Origin: TcHmiTextblock [▶ 367]

Note: Available since version 1.8

15.1.1.2.2.1 TcHmiTextbox
8

[Class]

Text box for entering text.

Attribute functions

Name	Origin	Description
getTextColor [▶ 1717]	TcHmiTextbox	Getter for the TextColor attribute.
setTextColor [▶ 1718]	TcHmiTextbox	Setter for the TextColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.

Name	Origin	Description
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.

Name	Origin	Description
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getText [▶ 1718]	TcHmiTextbox	Getter for the Text attribute.
setText [▶ 1719]	TcHmiTextbox	Setter for the Text attribute.
getAutoFocusOut [▶ 1729]	TcHmiTextbox	Getter for the AutoFocusOut attribute.
setAutoFocusOut [▶ 1730]	TcHmiTextbox	Setter for the AutoFocusOut attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.

Name	Origin	Description
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getTextHorizontalAlignment [▶ 1720]	TcHmiTextbox	Getter for the TextHorizontalAlignment attribute.
setTextHorizontalAlignment [▶ 1720]	TcHmiTextbox	Setter for the TextHorizontalAlignment attribute.
getTextVerticalAlignment [▶ 1719]	TcHmiTextbox	Getter for the TextVerticalAlignment attribute.
setTextVerticalAlignment [▶ 1719]	TcHmiTextbox	Setter for the TextVerticalAlignment attribute.
getMultiline [▶ 1725]	TcHmiTextbox	Getter for the Multiline attribute.
setMultiline [▶ 1726]	TcHmiTextbox	Setter for the Multiline attribute.
getPlaceholder [▶ 1723]	TcHmiTextbox	Getter for the Placeholder attribute.
setPlaceholder [▶ 1723]	TcHmiTextbox	Setter for the Placeholder attribute.
getWordWrap [▶ 1724]	TcHmiTextbox	Getter for the WordWrap attribute.
setWordWrap [▶ 1724]	TcHmiTextbox	Setter for the WordWrap attribute.
getContentPadding [▶ 1725]	TcHmiTextbox	Getter for the ContentPadding attribute.
setContentPadding [▶ 1725]	TcHmiTextbox	Setter for the ContentPadding attribute.
getTextFontFamily [▶ 1728]	TcHmiTextbox	Getter for the TextFontFamily attribute.
setTextFontFamily [▶ 1729]	TcHmiTextbox	Setter for the TextFontFamily attribute.
getTextFontSize [▶ 1721]	TcHmiTextbox	Getter for the TextFontSize attribute.
setTextFontSize [▶ 1721]	TcHmiTextbox	Setter for the TextFontSize attribute.
getTextFontSizeUnit [▶ 1722]	TcHmiTextbox	Getter for the TextFontSizeUnit attribute.
setTextFontSizeUnit [▶ 1722]	TcHmiTextbox	Setter for the TextFontSizeUnit attribute.
getTextFontStyle [▶ 1726]	TcHmiTextbox	Getter for the TextFontStyle attribute.
setTextFontStyle [▶ 1727]	TcHmiTextbox	Setter for the TextFontStyle attribute.

Name	Origin	Description
getTextFontWeight [▶ 1727]	TcHmiTextbox	Getter for the TextFontWeight attribute.
setTextFontWeight [▶ 1728]	TcHmiTextbox	Setter for the TextFontWeight attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Textbox

See also

Reference [▶ 376] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getTextColor
8.1

[Function]

```
public getTextColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TextColor [▶ 384] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextColor [▶ 1718]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextColor
8.2

[Function]

```
public setTextColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TextColor [▶ 384] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextColor [▶ 1717]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getText
8.3

[Function]

```
public getText(): string;
```

This function returns the currently set value for the Text [▶ 384] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setText [▶ 1719]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setText
8.4

[Function]

```
public setText(newValue: string | null): void;
```

This function sets a new value for the Text [▶ 384] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getText [▶ 1718]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getTextVerticalAlignment
8.5

[Function]

```
public getTextVerticalAlignment(): TcHmi.VerticalAlignment;
```

This function returns the currently set value for the TextVerticalAlignment [▶ 383] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.VerticalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextVerticalAlignment [▶ 1719]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

Not applicable from version 1.10.1018.48

15.1.1.2.2.1 setTextVerticalAlignment
8.6

[Function]

```
public setTextVerticalAlignment(newValue: TcHmi.VerticalAlignment | null): void;
```

This function sets a new value for the TextVerticalAlignment [▶ 383] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextVerticalAlignment [▶ 1719]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

Not applicable from version 1.10.1018.48

15.1.1.2.2.1 getTextHorizontalAlignment
8.7

[Function]

```
public getTextHorizontalAlignment(): TcHmi.HorizontalAlignment;
```

This function returns the currently set value for the TextHorizontalAlignment [▶ 383] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.HorizontalAlignment [▶ 2089]	The current value of the attribute.

See also

Attribute setter: setTextHorizontalAlignment [▶ 1720]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextHorizontalAlignment
8.8

[Function]

```
public setTextHorizontalAlignment(newValue: TcHmi.HorizontalAlignment | null): void;
```

This function sets a new value for the TextHorizontalAlignment [▶ 383] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextHorizontalAlignment [▶ 1720]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontSize
8.9

[Function]

```
public getTextFontSize(): number;
```

This function returns the currently set value for the TextFontSize [▶ 382] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontSize [▶ 1721]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontSize
8.10

[Function]

```
public setTextFontSize(newValue: number | null): void;
```

This function sets a new value for the TextFontSize [▶ 382] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSize [▶ 1721]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontSizeUnit
8.11

[Function]

```
public getTextFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TextFontSizeUnit [▶ 382] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTextFontSizeUnit [▶ 1722]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontSizeUnit
8.12

[Function]

```
public setTextFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TextFontSizeUnit [▶ 382] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontSizeUnit [▶ 1722]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getPlaceholder
8.13

[Function]

```
public getPlaceholder(): string | null;
```

This function returns the currently set value for the Placeholder [▶ 384] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setPlaceholder [▶ 1723]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setPlaceholder
8.14

[Function]

```
public setPlaceholder(newValue: string | null): void;
```

This function sets a new value for the Placeholder [▶ 384] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getPlaceholder [▶ 1723]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getWordWrap
8.15

[Function]

```
public getWordWrap(): boolean;
```

This function returns the currently set value for the WordWrap [▶ 385] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setWordWrap [▶ 1724]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setWordWrap
8.16

[Function]

```
public setWordWrap(newValue: boolean | null): void;
```

This function sets a new value for the WordWrap [▶ 385] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWordWrap [▶ 1724]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getContentPadding 8.17

[Function]

```
public getContentPadding(): TcHmi.FourSidedCss | null;
```

This function returns the currently set value for the ContentPadding [▶ 382] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FourSidedCss [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setContentPadding [▶ 1725]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setContentPadding 8.18

[Function]

```
public setContentPadding(newValue: TcHmi.FourSidedCss | null): void;
```

This function sets a new value for the ContentPadding [▶ 382] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getContentPadding [▶ 1725]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getMultiline 8.19

[Function]

```
public getMultiline(): boolean;
```

This function returns the currently set value for the Multiline [▶ 384] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setMultiline [▶ 1726]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setMultiline
8.20

[Function]

```
public setMultiline(newValue: boolean | null): void;
```

This function sets a new value for the attribute Multiline [▶ 384].

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMultiline [▶ 1725]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontStyle
8.21

[Function]

```
public getTextFontStyle(): TcHmi.FontStyle;
```

This function returns the currently set value for the TextFontStyle [▶ 383] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setTextFontStyle [▶ 1727]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontStyle
8.22

[Function]

```
public setTextFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the TextFontStyle [▶ 383] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontStyle [▶ 1726]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontWeight
8.23

[Function]

```
public getTextFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the TextFontWeight [▶ 383] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setTextFontWeight [▶ 1728]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontWeight
8.24

[Function]

```
public setTextFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the TextFontWeight [▶ 383] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontWeight [▶ 1727]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getTextFontFamily
8.25

[Function]

```
public getTextFontFamily(): string | null;
```

This function returns the currently set value for the TextFontFamily [▶ 382] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTextFontFamily [▶ 1729]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 setTextFontFamily
8.26

[Function]

```
public setTextFontFamily(newValue: string | null): void;
```

This function sets a new value for the TextFontFamily [▶ 382] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTextFontFamily [▶ 1728]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.8

15.1.1.2.2.1 getAutoFocusOut
8.27

[Function]

```
public getAutoFocusOut(): boolean;
```

This function returns the currently set value for the AutoFocusOut [▶ 385] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setAutoFocusOut [▶ 1730]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.10

15.1.1.2.2.1 setAutoFocusOut
8.28

[Function]

```
public setAutoFocusOut(newValue: boolean | null): void;
```

This function sets a new value for the AutoFocusOut [▶ 385] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getAutoFocusOut [▶ 1729]

Origin: TcHmiTextbox [▶ 376]

Note: Available since version 1.10

15.1.1.2.2.1 TcHmiToggleButton
9

[Class]

Button that can be toggled between two states.

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.

getText [▶ 1316]	TcHmiButton	Getter for the Text attribute.
setText [▶ 1317]	TcHmiButton	Setter for the Text attribute.
getStateSymbol [▶ 1334]	TcHmiButton	Getter for the StateSymbol attribute.
setStateSymbol [▶ 1333]	TcHmiButton	Setter for the StateSymbol attribute.

getToggleGroup [▶ 1738]	TcHmiToggleButton	Getter for the ToggleGroup attribute.
setToggleGroup [▶ 1738]	TcHmiToggleButton	Setter for the ToggleGroup attribute.
getToggleState [▶ 1738]	TcHmiToggleButton	Getter for the ToggleState attribute.
setToggleState [▶ 1739]	TcHmiToggleButton	Setter for the ToggleState attribute.
getTextColor [▶ 1316]	TcHmiButton	Getter for the TextColor attribute.
setTextColor [▶ 1316]	TcHmiButton	Setter for the TextColor attribute.
getTextFontFamily [▶ 1332]	TcHmiButton	Getter for the TextFontFamily attribute.
setTextFontFamily [▶ 1333]	TcHmiButton	Setter for the TextFontFamily attribute.
getTextFontSize [▶ 1326]	TcHmiButton	Getter for the TextFontSize attribute.
setTextFontSize [▶ 1326]	TcHmiButton	Setter for the TextFontSize attribute.
getTextFontSizeUnit [▶ 1327]	TcHmiButton	Getter for the TextFontSizeUnit attribute.
setTextFontSizeUnit [▶ 1327]	TcHmiButton	Setter for the TextFontSizeUnit attribute.
getTextFontStyle [▶ 1330]	TcHmiButton	Getter for the TextFontStyle attribute.
setTextFontStyle [▶ 1331]	TcHmiButton	Setter for the TextFontStyle attribute.
getTextFontWeight [▶ 1331]	TcHmiButton	Getter for the TextFontWeight attribute.
setTextFontWeight [▶ 1332]	TcHmiButton	Setter for the TextFontWeight attribute.
getTextPadding [▶ 1328]	TcHmiButton	Getter for the TextPadding attribute.
setTextPadding [▶ 1329]	TcHmiButton	Setter for the TextPadding attribute.
getTextHorizontalAlignment [▶ 1325]	TcHmiButton	Getter for the TextHorizontalAlignment attribute.
setTextHorizontalAlignment [▶ 1325]	TcHmiButton	Setter for the TextHorizontalAlignment attribute.
getTextVerticalAlignment [▶ 1328]	TcHmiButton	Getter for the TextVerticalAlignment attribute.
setTextVerticalAlignment [▶ 1328]	TcHmiButton	Setter for the TextVerticalAlignment attribute.
getWordWrap [▶ 1329]	TcHmiButton	Getter for the WordWrap attribute.
setWordWrap [▶ 1330]	TcHmiButton	Setter for the WordWrap attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.

getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getIcon [▶ 1317]	TcHmiButton	Getter for the Icon attribute.
setIcon [▶ 1318]	TcHmiButton	Setter for the Icon attribute.
getIconHeight [▶ 1320]	TcHmiButton	Getter for the IconHeight attribute.
setIconHeight [▶ 1321]	TcHmiButton	Setter for the IconHeight attribute.
getIconHeightUnit [▶ 1321]	TcHmiButton	Getter for the IconHeightUnit attribute.
setIconHeightUnit [▶ 1322]	TcHmiButton	Setter for the IconHeightUnit attribute.
getIconWidth [▶ 1318]	TcHmiButton	Getter for the IconWidth attribute.
setIconWidth [▶ 1319]	TcHmiButton	Setter for the IconWidth attribute.
getIconWidthUnit [▶ 1319]	TcHmiButton	Getter for the IconWidthUnit attribute.
setIconWidthUnit [▶ 1320]	TcHmiButton	Setter for the IconWidthUnit attribute.
getIconPadding [▶ 1324]	TcHmiButton	Getter for the IconPadding attribute.
setIconPadding [▶ 1324]	TcHmiButton	Setter for the IconPadding attribute.
getIconHorizontalAlignment [▶ 1323]	TcHmiButton	Getter for the IconHorizontalAlignment attribute.
setIconHorizontalAlignment [▶ 1323]	TcHmiButton	Setter for the IconHorizontalAlignment attribute.
getIconVerticalAlignment [▶ 1322]	TcHmiButton	Getter for the IconVerticalAlignment attribute.
setIconVerticalAlignment [▶ 1322]	TcHmiButton	Setter for the IconVerticalAlignment attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.

getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Button [▶ 1309]

Toggle Button

See also

Reference [▶ 386] of this control

Note: Available since version 1.8

15.1.1.2.2.1 getToggleGroup 9.1

[Function]

```
public getToggleGroup(): string | null;
```

This function returns the currently set value for the ToggleGroup [▶ 392] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setToggleGroup [▶ 1738]

Origin: TcHmiToggleButton [▶ 386]

Note: Available since version 1.8

15.1.1.2.2.1 setToggleGroup 9.2

[Function]

```
public setToggleGroup(newValue: string | null): void;
```

This function sets a new value for the ToggleGroup [▶ 392] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getToggleGroup [▶ 1738]

Origin: TcHmiToggleButton [▶ 386]

Note: Available since version 1.8

15.1.1.2.2.1 getToggleState 9.3

[Function]

```
public getToggleState(): TcHmi.ToggleState;
```

This function returns the currently set value for the ToggleState [▶ 392] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.ToggleState [▶ 2097]	The current value of the attribute.

See also

Attribute setter: setToggleState [▶ 1739]

Origin: TcHmiToggleButton [▶ 386]

Note: Available since version 1.8

15.1.1.2.2.1 setToggleState
9.4

[Function]

```
public setToggleState(newValue: TcHmi.ToggleState | null): void;
```

This function sets a new value for the ToggleState [▶ 392] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.ToggleState [▶ 2097], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getToggleState [▶ 1738]

Origin: TcHmiToggleButton [▶ 386]

Note: Available since version 1.8

15.1.1.2.2.2 TcHmiTrendLineChart
0

[Class]

Line chart for historicized data

Attribute functions

Name	Origin	Description
getStart [▶ 1758]	TcHmiTrendLineChart	Getter for the Start attribute.
setStart [▶ 1758]	TcHmiTrendLineChart	Setter for the Start attribute.
getActualStartTime [▶ 1759]	TcHmiTrendLineChart	Getter for the ActualStartTime attribute.
getEnd [▶ 1759]	TcHmiTrendLineChart	Getter for the End attribute.
setEnd [▶ 1760]	TcHmiTrendLineChart	Setter for the End attribute.
getActualEndTime [▶ 1760]	TcHmiTrendLineChart	Getter for the ActualEndTime attribute.
getInterval [▶ 1762]	TcHmiTrendLineChart	Getter for the Interval attribute.
setInterval [▶ 1762]	TcHmiTrendLineChart	Setter for the Interval attribute.
getLineGraphDescriptions [▶ 1761]	TcHmiTrendLineChart	Getter for the LineGraphDescriptions attribute.
setLineGraphDescriptions [▶ 1761]	TcHmiTrendLineChart	Setter for the LineGraphDescriptions attribute.
getMouseMode [▶ 1763]	TcHmiTrendLineChart	Getter for the MouseMode attribute.
setMouseMode [▶ 1763]	TcHmiTrendLineChart	Setter for the MouseMode attribute.
getSectionsBackgroundColor [▶ 1757]	TcHmiTrendLineChart	Getter for the SectionsBackgroundColor attribute.
setSectionsBackgroundColor [▶ 1758]	TcHmiTrendLineChart	Setter for the SectionsBackgroundColor attribute.
getShowXAxis [▶ 1763]	TcHmiTrendLineChart	Getter for the ShowXAxis attribute.
setShowXAxis [▶ 1764]	TcHmiTrendLineChart	Setter for the ShowXAxis attribute.
getXAxisFormat [▶ 1778]	TcHmiTrendLineChart	Getter for the XAxisFormat attribute.
setXAxisFormat [▶ 1779]	TcHmiTrendLineChart	Setter for the XAxisFormat attribute.
getXShowLabels [▶ 1764]	TcHmiTrendLineChart	Getter for the XShowLabels attribute.
setXShowLabels [▶ 1765]	TcHmiTrendLineChart	Setter for the XShowLabels attribute.
getXMainTickSteps [▶ 1765]	TcHmiTrendLineChart	Getter for the XMainTickSteps attribute.
setXMainTickSteps [▶ 1766]	TcHmiTrendLineChart	Setter for the XMainTickSteps attribute.
getShowXSubTicks [▶ 1766]	TcHmiTrendLineChart	Getter for the ShowXSubTicks attribute.
setShowXSubTicks [▶ 1767]	TcHmiTrendLineChart	Setter for the ShowXSubTicks attribute.
getXSubTickSteps [▶ 1767]	TcHmiTrendLineChart	Getter for the XSubTickSteps attribute.
setXSubTickSteps [▶ 1768]	TcHmiTrendLineChart	Setter for the XSubTickSteps attribute.
getXLabelFontColor [▶ 1753]	TcHmiTrendLineChart	Getter for the XLabelFontColor attribute.
setXLabelFontColor [▶ 1754]	TcHmiTrendLineChart	Setter for the XLabelFontColor attribute.
getXLabelFontFamily [▶ 1768]	TcHmiTrendLineChart	Getter for the XLabelFontFamily attribute.
setXLabelFontFamily [▶ 1769]	TcHmiTrendLineChart	Setter for the XLabelFontFamily attribute.

Name	Origin	Description
getXLabelFontSize [▶ 1769]	TcHmiTrendLineChart	Getter for the XLabelFontSize attribute.
setXLabelFontSize [▶ 1769]	TcHmiTrendLineChart	Setter for the XLabelFontSize attribute.
getXLabelFontSizeUnit [▶ 1770]	TcHmiTrendLineChart	Getter for the XLabelFontSizeUnit attribute.
setXLabelFontSizeUnit [▶ 1770]	TcHmiTrendLineChart	Setter for the XLabelFontSizeUnit attribute.
getXLabelFontWeight [▶ 1771]	TcHmiTrendLineChart	Getter for the XLabelFontWeight attribute.
setXLabelFontWeight [▶ 1771]	TcHmiTrendLineChart	Setter for the XLabelFontWeight attribute.
getXAxisWidth [▶ 1772]	TcHmiTrendLineChart	Getter for the XAxisWidth attribute.
setXAxisWidth [▶ 1772]	TcHmiTrendLineChart	Setter for the XAxisWidth attribute.
getXAxisColor [▶ 1749]	TcHmiTrendLineChart	Getter for the XAxisColor attribute.
setXAxisColor [▶ 1749]	TcHmiTrendLineChart	Setter for the XAxisColor attribute.
getShowXAxisName [▶ 1773]	TcHmiTrendLineChart	Getter for the ShowXAxisName attribute.
setShowXAxisName [▶ 1773]	TcHmiTrendLineChart	Setter for the ShowXAxisName attribute.
getXAxisName [▶ 1774]	TcHmiTrendLineChart	Getter for the XAxisName attribute.
setXAxisName [▶ 1774]	TcHmiTrendLineChart	Setter for the XAxisName attribute.
getXAxisNameFontColor [▶ 1752]	TcHmiTrendLineChart	Getter for the XAxisNameFontColor attribute.
setXAxisNameFontColor [▶ 1753]	TcHmiTrendLineChart	Setter for the XAxisNameFontColor attribute.
getXAxisNameFontFamily [▶ 1775]	TcHmiTrendLineChart	Getter for the XAxisNameFontFamily attribute.
setXAxisNameFontFamily [▶ 1775]	TcHmiTrendLineChart	Setter for the XAxisNameFontFamily attribute.
getXAxisNameFontSize [▶ 1775]	TcHmiTrendLineChart	Getter for the XAxisNameFontSize attribute.
setXAxisNameFontSize [▶ 1776]	TcHmiTrendLineChart	Setter for the XAxisNameFontSize attribute.
getXAxisNameFontSizeUnit [▶ 1776]	TcHmiTrendLineChart	Getter for the XAxisNameFontSizeUnit attribute.
setXAxisNameFontSizeUnit [▶ 1777]	TcHmiTrendLineChart	Setter for the XAxisNameFontSizeUnit attribute.
getXAxisNameFontWeight [▶ 1777]	TcHmiTrendLineChart	Getter for the XAxisNameFontWeight attribute.
setXAxisNameFontWeight [▶ 1778]	TcHmiTrendLineChart	Setter for the XAxisNameFontWeight attribute.
getYAxis [▶ 1779]	TcHmiTrendLineChart	Getter for the YAxis attribute.
setYAxis [▶ 1780]	TcHmiTrendLineChart	Setter for the YAxis attribute.
getYMainTickSteps [▶ 1780]	TcHmiTrendLineChart	Getter for the YMainTickSteps attribute.
setYMainTickSteps [▶ 1781]	TcHmiTrendLineChart	Setter for the YMainTickSteps attribute.
getShowYSubTicks [▶ 1781]	TcHmiTrendLineChart	Getter for the ShowYSubTicks attribute.
setShowYSubTicks [▶ 1781]	TcHmiTrendLineChart	Setter for the ShowYSubTicks attribute.

Name	Origin	Description
getYSubTickSteps [▶ 1782]	TcHmiTrendLineChart	Getter for the YSubTickSteps attribute.
setYSubTickSteps [▶ 1782]	TcHmiTrendLineChart	Setter for the YSubTickSteps attribute.
getYAxisWidth [▶ 1787]	TcHmiTrendLineChart	Getter for the YAxisWidth attribute.
setYAxisWidth [▶ 1787]	TcHmiTrendLineChart	Setter for the YAxisWidth attribute.
getYLabelFontFamily [▶ 1783]	TcHmiTrendLineChart	Getter for the YLabelFontFamily attribute.
setYLabelFontFamily [▶ 1783]	TcHmiTrendLineChart	Setter for the YLabelFontFamily attribute.
getYLabelFontSize [▶ 1784]	TcHmiTrendLineChart	Getter for the YLabelFontSize attribute.
setYLabelFontSize [▶ 1784]	TcHmiTrendLineChart	Setter for the YLabelFontSize attribute.
getYLabelFontSizeUnit [▶ 1785]	TcHmiTrendLineChart	Getter for the YLabelFontSizeUnit attribute.
setYLabelFontSizeUnit [▶ 1785]	TcHmiTrendLineChart	Setter for the YLabelFontSizeUnit attribute.
getYLabelFontWeight [▶ 1786]	TcHmiTrendLineChart	Getter for the YLabelFontWeight attribute.
setYLabelFontWeight [▶ 1786]	TcHmiTrendLineChart	Setter for the YLabelFontWeight attribute.
getShowGrid [▶ 1787]	TcHmiTrendLineChart	Getter for the ShowGrid attribute.
setShowGrid [▶ 1788]	TcHmiTrendLineChart	Setter for the ShowGrid attribute.
getGridShowHorizontalLines [▶ 1789]	TcHmiTrendLineChart	Getter for the GridShowHorizontalLines attribute.
setGridShowHorizontalLines [▶ 1790]	TcHmiTrendLineChart	Setter for the GridShowHorizontalLines attribute.
getGridShowVerticalLines [▶ 1788]	TcHmiTrendLineChart	Getter for the GridShowVerticalLines attribute.
setGridShowVerticalLines [▶ 1789]	TcHmiTrendLineChart	Setter for the GridShowVerticalLines attribute.
getGridLineWidth [▶ 1790]	TcHmiTrendLineChart	Getter for the GridLineWidth attribute.
setGridLineWidth [▶ 1791]	TcHmiTrendLineChart	Setter for the GridLineWidth attribute.
getGridBackgroundColor [▶ 1750]	TcHmiTrendLineChart	Getter for the GridBackgroundColor attribute.
setGridBackgroundColor [▶ 1750]	TcHmiTrendLineChart	Setter for the GridBackgroundColor attribute.
getGridLineColor [▶ 1751]	TcHmiTrendLineChart	Getter for the GridLineColor attribute.
setGridLineColor [▶ 1751]	TcHmiTrendLineChart	Setter for the GridLineColor attribute.
getSubgridLineColor [▶ 1752]	TcHmiTrendLineChart	Getter for the SubgridLineColor attribute.
setSubgridLineColor [▶ 1752]	TcHmiTrendLineChart	Setter for the SubgridLineColor attribute.
getSubgridLineWidth [▶ 1793]	TcHmiTrendLineChart	Getter for the SubgridLineWidth attribute.
setSubgridLineWidth [▶ 1793]	TcHmiTrendLineChart	Setter for the SubgridLineWidth attribute.

Name	Origin	Description
getSubgridShowHorizontalLines [▶ 1792]	TcHmiTrendLineChart	Getter for the SubgridShowHorizontalLines attribute.
setSubgridShowHorizontalLines [▶ 1793]	TcHmiTrendLineChart	Setter for the SubgridShowHorizontalLines attribute.
getSubgridShowVerticalLines [▶ 1791]	TcHmiTrendLineChart	Getter for the SubgridShowVerticalLines attribute.
setSubgridShowVerticalLines [▶ 1792]	TcHmiTrendLineChart	Setter for the SubgridShowVerticalLines attribute.
setShowMenuBar [▶ 1794]	TcHmiTrendLineChart	Setter for the ShowMenubar attribute.
getShowMenuBar [▶ 1794]	TcHmiTrendLineChart	Getter for the ShowMenubar attribute.
setMenuBarPosition [▶ 1795]	TcHmiTrendLineChart	Setter for the MenubarPosition attribute.
getMenuBarPosition [▶ 1795]	TcHmiTrendLineChart	Getter for the MenubarPosition attribute.
setMenuBarLabelFontFamily [▶ 1796]	TcHmiTrendLineChart	Setter for the MenubarLabelFontFamily attribute.
getMenuBarLabelFontFamily [▶ 1796]	TcHmiTrendLineChart	Getter for the MenubarLabelFontFamily attribute.
getMenuBarLabelFontSize [▶ 1797]	TcHmiTrendLineChart	Getter for the MenubarLabelFontSize attribute.
setMenuBarLabelFontSize [▶ 1797]	TcHmiTrendLineChart	Setter for the MenubarLabelFontSize attribute.
getMenuBarLabelFontSizeUnit [▶ 1798]	TcHmiTrendLineChart	Getter for the MenubarLabelFontSizeUnit attribute.
setMenuBarLabelFontSizeUnit [▶ 1798]	TcHmiTrendLineChart	Setter for the MenubarLabelFontSizeUnit attribute.
getMenuBarLabelFontWeight [▶ 1799]	TcHmiTrendLineChart	Getter for the MenubarLabelFontWeight attribute.
setMenuBarLabelFontWeight [▶ 1799]	TcHmiTrendLineChart	Setter for the MenubarLabelFontWeight attribute.
getMenuBarLabelFontStyle [▶ 1800]	TcHmiTrendLineChart	Getter for the MenubarLabelFontStyle attribute.
setMenuBarLabelFontStyle [▶ 1800]	TcHmiTrendLineChart	Setter for the MenubarLabelFontStyle attribute.
getMenuBarLabelFontColor [▶ 1754]	TcHmiTrendLineChart	Getter for the MenubarLabelFontColor attribute.
setMenuBarLabelFontColor [▶ 1755]	TcHmiTrendLineChart	Setter for the MenubarLabelFontColor attribute.
setShowTooltip [▶ 1801]	TcHmiTrendLineChart	Setter for the ShowTooltip attribute.
getShowTooltip [▶ 1801]	TcHmiTrendLineChart	Getter for the ShowTooltip attribute.
setTooltipFormat [▶ 1802]	TcHmiTrendLineChart	Setter for the TooltipFormat attribute.
getTooltipFormat [▶ 1801]	TcHmiTrendLineChart	Getter for the TooltipFormat attribute.

Name	Origin	Description
setTooltipFontFamily [▶ 1803]	TcHmiTrendLineChart	Setter for the TooltipFontFamily attribute.
getTooltipFontFamily [▶ 1802]	TcHmiTrendLineChart	Getter for the TooltipFontFamily attribute.
getTooltipFontSize [▶ 1803]	TcHmiTrendLineChart	Getter for the TooltipFontSize attribute.
setTooltipFontSize [▶ 1804]	TcHmiTrendLineChart	Setter for the TooltipFontSize attribute.
getTooltipFontSizeUnit [▶ 1804]	TcHmiTrendLineChart	Getter for the TooltipFontSizeUnit attribute.
setTooltipFontSizeUnit [▶ 1805]	TcHmiTrendLineChart	Setter for the TooltipFontSizeUnit attribute.
getTooltipFontWeight [▶ 1805]	TcHmiTrendLineChart	Getter for the TooltipFontWeight attribute.
setTooltipFontWeight [▶ 1806]	TcHmiTrendLineChart	Setter for the TooltipFontWeight attribute.
getTooltipFontColor [▶ 1755]	TcHmiTrendLineChart	Getter for the TooltipFontColor attribute.
setTooltipFontColor [▶ 1756]	TcHmiTrendLineChart	Setter for the TooltipFontColor attribute.
getTooltipBackgroundColor [▶ 1756]	TcHmiTrendLineChart	Getter for the TooltipBackgroundColor attribute.
setTooltipBackgroundColor [▶ 1757]	TcHmiTrendLineChart	Setter for the TooltipBackgroundColor attribute.
stop [▶ 1806]	TcHmiTrendLineChart	Fixes the view.
resetZoom [▶ 1806]	TcHmiTrendLineChart	Resets zooming and panning.
reset [▶ 1807]	TcHmiTrendLineChart	Starts the view.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.

Name	Origin	Description
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.

Name	Origin	Description
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1136]	TcHmiControl	Getter for the WidthMode attribute.
setWidthMode [▶ 1137]	TcHmiControl	Setter for the WidthMode attribute.
getHeightMode [▶ 1135]	TcHmiControl	Getter for the HeightMode attribute.
setHeightMode [▶ 1136]	TcHmiControl	Setter for the HeightMode attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.

Name	Origin	Description
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Trend line chart

See also

Reference [▶ 393] of this control

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisColor
0.1

[Function]

```
public getXAxisColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XAxisColor [▶ 413] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisColor [▶ 1749]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisColor
0.2

[Function]

```
public setXAxisColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XAxisColor [▶ 413] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisColor [▶ 1749]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getGridBackgroundColor 0.3

[Function]

```
public getGridBackgroundColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the GridBackgroundColor [▶ 403] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084]	The current value of the attribute.

See also

Attribute setter: setGridBackgroundColor [▶ 1750]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setGridBackgroundColor 0.4

[Function]

```
public setGridBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the GridBackgroundColor [▶ 403] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridBackgroundColor [▶ 1750]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getGridLineColor 0.5

[Function]

```
public getGridLineColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the GridLineColor [▶ 403] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridLineColor [▶ 1751]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setGridLineColor 0.6

[Function]

```
public setGridLineColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the GridLineColor [▶ 403] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridLineColor [▶ 1751]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getSubgridLineColor
0.7

[Function]

```
public getSubgridLineColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the SubgridLineColor [▶ 410] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSubgridLineColor [▶ 1752]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setSubgridLineColor
0.8

[Function]

```
public setSubgridLineColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the SubgridLineColor [▶ 410] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridLineColor [▶ 1752]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisNameFontColor
0.9

[Function]

```
public getXAxisNameFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XAxisNameFontColor [▶ 415] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontColor [▶ 1753]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisNameFontColor
0.10

[Function]

```
public setXAxisNameFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XAxisNameFontColor [▶ 415] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontColor [▶ 1752]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXLabelFontColor
0.11

[Function]

```
public getXLabelFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the XLabelFontColor [▶ 417] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontColor [▶ 1754]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXLabelFontColor
0.12

[Function]

```
public setXLabelFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the XLabelFontColor [▶ 417] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontColor [▶ 1753]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarLabelFontColor
0.13

[Function]

```
public getMenuBarLabelFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the MenuBarLabelFontColor [▶ 405] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMenuBarLabelFontColor [▶ 1755]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarLabelFontColor
0.14

[Function]

```
public setMenuBarLabelFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the MenuBarLabelFontColor [▶ 405] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarLabelFontColor [▶ 1754]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipFontColor
0.15

[Function]

```
public getTooltipFontColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the TooltipFontColor [▶ 411] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipFontColor [▶ 1756]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipFontColor
0.16

[Function]

```
public seTooltipFontColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TooltipFontColor [▶ 411] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SolidColor [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontColor [▶ 1755]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipBackgroundColor
0.17

[Function]

```
public getTooltipBackgroundColor(): TcHmi.Color | null;
```

This function returns the currently set value for the TooltipBackgroundColor [▶ 411] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipBackgroundColor [▶ 1757]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipBackgroundColor 0.18

[Function]

```
public setTooltipBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the TooltipBackgroundColor [▶ 411] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipBackgroundColor [▶ 1756]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getSectionsBackgroundColor 0.19

[Function]

```
public getSectionsBackgroundColor(): TcHmi.SolidColor | null;
```

This function returns the currently set value for the SectionsBackgroundColor [▶ 407] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Color [▶ 2084], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSectionsBackgroundColor [▶ 1758]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setSectionsBackgroundColor
0.20

[Function]

```
public setSectionsBackgroundColor(newValue: TcHmi.SolidColor | null): void;
```

This function sets a new value for the SectionsBackgroundColor attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Color [▶ 2084], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSectionsBackgroundColor

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getStart
0.21

[Function]

```
public getStart(): string | null;
```

This function returns the currently set value for the Start [▶ 409] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setStart [▶ 1758]

Origin: TcHmiTrendLineChart [▶ 393]

Schema: tchmi:framework#/definitions/TrendLineChartStart [▶ 2264]

Note: Available since version 1.10

15.1.1.2.2.2 setStart
0.22

[Function]

```
public setStart(newValue: string | null): void;
```

This function sets a new value for the Start [▶ 409] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getStart [▶ 1758]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getActualStartTime
0.23

[Function]

```
public getActualStartTime(): string | null;
```

This function returns the currently set value for the ActualStartTime [▶ 403] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Origin: TcHmiTrendLineChart [▶ 393]

Schema: tchmi:framework#/definitions/TrendLineChartStart [▶ 2264]

Note: Available since version 1.10

15.1.1.2.2.2 getEnd
0.24

[Function]

```
public getEnd(): string | null;
```

This function returns the currently set value for the End [▶ 403] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setEnd [▶ 1760]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setEnd
0.25

[Function]

```
public setEnd(newValue: string | null): void;
```

This function sets a new value for the End [▶ 403] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getEnd [▶ 1759]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getActualEndTime
0.26

[Function]

```
public getActualEndTime (): string | null;
```

This function returns the currently set value for the ActualEndTime [▶ 402] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Origin: TcHmiTrendLineChart [▶ 393]

Schema: tchmi:framework#/definitions/TrendLineChartEnd [▶ 2263]

Note: Available since version 1.10

15.1.1.2.2.2 getLineGraphDescriptions
0.27

[Function]

```
public getLineGraphDescriptions():
TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.LineGraphDescription[] | null;
```

This function returns the currently set value for the LineGraphDescriptions [▶ 405] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.LineGraphDescription [▶ 1808][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setLineGraphDescriptions [▶ 1761]

Origin: TcHmiTrendLineChart [▶ 1739]

Note: Available since version 1.10

15.1.1.2.2.2 setLineGraphDescriptions
0.28

[Function]

```
public setLineGraphDescriptions(newValue:
TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.LineGraphDescription[] | null): void;
```

This function sets a new value for the LineGraphDescriptions [▶ 405] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.LineGraphDescription [▶ 1808][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLineGraphDescriptions [▶ 1761]

Origin: TcHmiTrendLineChart [▶ 1739]

Note: Available since version 1.10

15.1.1.2.2.2 getInterval
0.29

[Function]

```
public getInterval(): number;
```

This function returns the currently set value for the Interval [▶ 404] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setInterval [▶ 1762]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setInterval
0.30

[Function]

```
public setInterval(newValue: number | null): void;
```

This function sets a new value for the Interval [▶ 404] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getInterval [▶ 1762]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMouseMode
0.31

[Function]

```
public getMouseMode(): TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.MouseMode;
```

This function returns the currently set value for the MouseMode [▶ 407] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.MouseMode [▶ 1811]	The current value of the attribute.

See also

Attribute setter: setMouseMode [▶ 1763]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMouseMode
0.32

[Function]

```
public setMouseMode(newValue: TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.MouseMode): void;
```

This function sets a new value for the MouseMode [▶ 407] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.MouseMode [▶ 1811]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMouseMode [▶ 1763]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowXAxis
0.33

[Function]

```
public getShowXAxis(): boolean;
```

This function returns the currently set value for the ShowXAxis [▶ 408] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXAxis [▶ 1764]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowXAxis
0.34

[Function]

```
public setShowXAxis(newValue: boolean | null): void;
```

This function sets a new value for the ShowXAxis [▶ 408] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXAxis [▶ 1763]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXShowLabels
0.35

[Function]

```
public getXShowLabels(): boolean;
```

This function returns the currently set value for the XShowLabels [▶ 418] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setXShowLabels [▶ 1765]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXShowLabels
0.36

[Function]

```
public setXShowLabels(newValue: boolean | null): void;
```

This function sets a new value for the XShowLabels [▶ 418] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXShowLabels [▶ 1764]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXMainTickSteps
0.37

[Function]

```
public getXMainTickSteps(): number;
```

This function returns the currently set value for the XMainTickSteps [▶ 418] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXMainTickSteps [▶ 1766]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXMainTickSteps
0.38

[Function]

```
public setXMainTickSteps(newValue: number | null): void;
```

This function sets a new value for the XMainTickSteps [▶ 418] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXMainTickSteps [▶ 1765]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowXSubTicks
0.39

[Function]

```
public getShowXSubTicks(): boolean;
```

This function returns the currently set value for the ShowXSubTicks [▶ 409] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXSubTicks [▶ 1767]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowXSubTicks
0.40

[Function]

```
public setShowXSubTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowXSubTicks [▶ 409] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXSubTicks [▶ 1766]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXSubTickSteps
0.41

[Function]

```
public getXSubTickSteps(): number | null;
```

This function returns the currently set value for the XSubTickSteps [▶ 418] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXSubTickSteps [▶ 1768]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXSubTickSteps
0.42

[Function]

```
public setXSubTickSteps(newValue: number | null): void;
```

This function sets a new value for the XSubTickSteps [▶ 418] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXSubTickSteps [▶ 1767]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXLabelFontFamily
0.43

[Function]

```
public getXLabelFontFamily(): string;
```

This function returns the currently set value for the XLabelFontFamily [▶ 417] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontFamily [▶ 1769]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXLabelFontFamily
0.44

[Function]

```
public setXLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the XLabelFontFamily [▶ 417] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontFamily [▶ 1768]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXLabelFontSize
0.45

[Function]

```
public getXLabelFontSize(): number;
```

This function returns the currently set value for the XLabelFontSize [▶ 417] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXLabelFontSize [▶ 1769]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXLabelFontSize
0.46

[Function]

```
public setXLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the XLabelFontSize [▶ 417] attribute.

Parameter

Name	Type	Description
newValue	numbe, nullr [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontSize [▶ 1769]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXLabelFontSizeUnit
0.47

[Function]

```
public getXLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the XLabelFontSizeUnit [▶ 417] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setXLabelFontSizeUnit [▶ 1770]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXLabelFontSizeUnit
0.48

[Function]

```
public setXLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the XLabelFontSizeUnit [▶ 417] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontSizeUnit [▶ 1770]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXLabelFontWeight
0.49

[Function]

```
public getXLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the XLabelFontFontWeight [▶ 418] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setXLabelFontWeight [▶ 1771]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXLabelFontWeight
0.50

[Function]

```
public setXLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the XLabelFontWeight [▶ 418] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXLabelFontWeight [▶ 1771]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisWidth
0.51

[Function]

```
public getXAxisWidth(): number;
```

This function returns the currently set value for the XAxisWidth [▶ 416] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisWidth [▶ 1772]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisWidth
0.52

[Function]

```
public setXAxisWidth(newValue: number | null): void;
```

This function sets a new value for the XAxisWidth [▶ 416] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisWidth [▶ 1772]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowXAxisName
0.53

[Function]

```
public getShowXAxisName(): boolean;
```

This function returns the currently set value for the ShowXAxisName [▶ 408] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowXAxisName [▶ 1773]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowXAxisName
0.54

[Function]

```
public setShowXAxisName(newValue: boolean | null): void;
```

This function sets a new value for the ShowXAxisName [▶ 408] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowXAxisName [▶ 1773]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisName
0.55

[Function]

```
public getXAxisName(): string | null;
```

This function returns the currently set value for the XAxisName [▶ 415] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisName [▶ 1774]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisName
0.56

[Function]

```
public setXAxisName(newValue: string | null): void;
```

This function sets a new value for the XAxisName [▶ 415] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisName [▶ 1774]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisNameFontFamily
0.57

[Function]

```
public getXAxisNameFontFamily(): string;
```

This function returns the currently set value for the XAxisNameFontFamily [▶ 415] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontFamily [▶ 1775]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisNameFontFamily
0.58

[Function]

```
public setXAxisNameFontFamily(newValue: string | null): void;
```

This function sets a new value for the XAxisNameFontFamily [▶ 415] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontFamily [▶ 1775]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisNameFontSize
0.59

[Function]

```
public getXAxisNameFontSize(): number;
```

This function returns the currently set value for the XAxisNameFontSize [▶ 415] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontSize [▶ 1776]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisNameFontSize
0.60

[Function]

```
public setXAxisNameFontSize(newValue: number | null): void;
```

This function sets a new value for the XAxisNameFontSize [▶ 415] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontSize [▶ 1775]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisNameFontSizeUnit
0.61

[Function]

```
public getXAxisNameFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the XAxisNameFontSizeUnit [▶ 416] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontSizeUnit [▶ 1777]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisNameFontSizeUnit
0.62

[Function]

```
public setXAxisNameFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the XAxisNameFontSizeUnit [▶ 416] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontSizeUnit [▶ 1776]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisNameFontWeight
0.63

[Function]

```
public getXAxisNameFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the XAxisNameFontWeight [▶ 416] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setXAxisNameFontWeight [▶ 1778]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisNameFontWeight
0.64

[Function]

```
public setXAxisNameFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the XAxisNameFontWeight [▶ 416] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisNameFontWeight [▶ 1777]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getXAxisFormat
0.65

[Function]

```
public getXAxisFormat(): TcHmi.IFunction | null;
```

This function returns the currently set value for the XAxisFormat [▶ 414] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 2095], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setXAxisFormat [▶ 1779]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setXAxisFormat
0.66

[Function]

```
public setXAxisFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the XAxisFormat [▶ 414] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 2095], null [▶ 2108]	The new value of the attribute.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getXAxisFormat [▶ 1778]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYAxis
0.67

[Function]

```
public getYAxis(): TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.YAxis[] | null;
```

This function returns the currently set value for the YAxis [▶ 419] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.YAxis [▶ 1807][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxis [▶ 1780]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYAxis
0.68

[Function]

```
public setYAxis(newValue: TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.YAxis[] | null): void;
```

This function sets a new value for the YAxis [▶ 419] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.YAxis [▶ 1807][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxis [▶ 1779]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYMainTickSteps
0.69

[Function]

```
public getYMainTickSteps(): number;
```

This function returns the currently set value for the YMainTickSteps [▶ 420] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYMainTickSteps [▶ 1781]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYMainTickSteps
0.70

[Function]

```
public setYMainTickSteps(newValue: number | null): void;
```

This function sets a new value for the YMainTickSteps [▶ 420] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYMainTickSteps [▶ 1780]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowYSubTicks
0.71

[Function]

```
public getShowYSubTicks(): boolean;
```

This function returns the currently set value for the ShowYSubTicks [▶ 409] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowYSubTicks [▶ 1781]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowYSubTicks
0.72

[Function]

```
public setShowYSubTicks(newValue: boolean | null): void;
```

This function sets a new value for the ShowYSubTicks [▶ 409] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowYSubTicks [▶ 1781]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYSubTickSteps
0.73

[Function]

```
public getYSubTickSteps(): number | null;
```

This function returns the currently set value for the YSubTickSteps [▶ 421] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYSubTickSteps [▶ 1782]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYSubTickSteps
0.74

[Function]

```
public setYSubTickSteps(newValue: number | null): void;
```

This function sets a new value for the YSubTickSteps [▶ 421] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYSubTickSteps [▶ 1782]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYLabelFontFamily
0.75

[Function]

```
public getYLabelFontFamily(): string;
```

This function returns the currently set value for the YLabelFontFamily [▶ 419] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontFamily [▶ 1783]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYLabelFontFamily
0.76

[Function]

```
public setYLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the YLabelFontFamily [▶ 419] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontFamily [▶ 1783]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYLabelFontSize
0.77

[Function]

```
public getYLabelFontSize(): number;
```

This function returns the currently set value for the YLabelFontSize [▶ 419] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYLabelFontSize [▶ 1784]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYLabelFontSize
0.78

[Function]

```
public setYLabelFontSize(newValue: number | null): void;
```

This function sets a new value for the YLabelFontSize [▶ 419] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontSize [▶ 1784]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYLabelFontSizeUnit
0.79

[Function]

```
public getYLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the YLabelFontSizeUnit [▶ 420] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setYLabelFontSizeUnit [▶ 1785]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYLabelFontSizeUnit
0.80

[Function]

```
public setYLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the YLabelFontSizeUnit [▶ 420] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontSizeUnit [▶ 1785]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYLabelFontWeight 0.81

[Function]

```
public getYLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the YLabelFontWeight [▶ 420] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setYLabelFontWeight [▶ 1786]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYLabelFontWeight 0.82

[Function]

```
public setYLabelFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the YLabelFontWeight [▶ 420] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYLabelFontWeight [▶ 1786]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getYAxisWidth
0.83

[Function]

```
public getYAxisWidth(): number;
```

This function returns the currently set value for the YAxisWidth [▶ 419] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setYAxisWidth [▶ 1787]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setYAxisWidth
0.84

[Function]

```
public setYAxisWidth(newValue: number | null): void;
```

This function sets a new value for the YAxisWidth [▶ 419] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getYAxisWidth [▶ 1787]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowGrid
0.85

[Function]

```
public getShowGrid(): boolean;
```

This function returns the currently set value for the ShowGrid [▶ 407] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowGrid [▶ 1788]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowGrid
0.86

[Function]

```
public setShowGrid(newValue: boolean | null): void;
```

This function sets a new value for the ShowGrid [▶ 407] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowGrid [▶ 1787]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getGridShowVerticalLines
0.87

[Function]

```
public getGridShowVerticalLines(): boolean;
```

This function returns the currently set value for the GridShowVerticalLines [▶ 404] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setGridShowVerticalLines [▶ 1789]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setGridShowVerticalLines
0.88

[Function]

```
public setGridShowVerticalLines(newValue: boolean | null): void;
```

This function sets a new value for the GridShowVerticalLines [▶ 404] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridShowVerticalLines [▶ 1788]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getGridShowHorizontalLines
0.89

[Function]

```
public getGridShowHorizontalLines(): boolean;
```

This function returns the currently set value for the GridShowHorizontalLines [▶ 404] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setGridShowHorizontalLines [▶ 1790]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setGridShowHorizontalLines
0.90

[Function]

```
public setGridShowHorizontalLines(newValue: boolean | null): void;
```

This function sets a new value for the GridShowHorizontalLines [▶ 404] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridShowHorizontalLines [▶ 1789]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getGridLineWidth
0.91

[Function]

```
public getGridLineWidth(): number;
```

This function returns the currently set value for the GridLineWidth [▶ 404] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setGridLineWidth [▶ 1791]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setGridLineWidth
0.92

[Function]

```
public setGridLineWidth(newValue: number | null): void;
```

This function sets a new value for the GridLineWidth [▶ 404] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getGridLineWidth [▶ 1790]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getSubgridShowVerticalLines
0.93

[Function]

```
public getSubgridShowVerticalLines(): boolean;
```

This function returns the currently set value for the SubgridShowVerticalLines [▶ 410] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setSubgridShowVerticalLines [▶ 1792]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setSubgridShowVerticalLines
0.94

[Function]

```
public setSubgridShowVerticalLines(newValue: boolean | null): void;
```

This function sets a new value for the SubgridShowVerticalLines [▶ 410] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridShowVerticalLines [▶ 1791]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getSubgridShowHorizontalLines
0.95

[Function]

```
public getSubgridShowHorizontalLines(): boolean;
```

This function returns the currently set value for the SubgridShowHorizontalLines [▶ 410] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setSubgridShowHorizontalLines [▶ 1793]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setSubgridShowHorizontalLines
0.96

[Function]

```
public setSubgridShowHorizontalLines(newValue: boolean | null): void;
```

This function sets a new value for the SubgridShowHorizontalLines [▶ 410] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridShowHorizontalLines [▶ 1792]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getSubgridLineWidth
0.97

[Function]

```
public getSubgridLineWidth(): number;
```

This function returns the currently set value for the SubgridLineWidth [▶ 410] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSubgridLineWidth [▶ 1793]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setSubgridLineWidth
0.98

[Function]

```
public setSubgridLineWidth(newValue: number | null): void;
```

This function sets a new value for the SubgridLineWidth [▶ 410] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSubgridLineWidth [▶ 1793]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowMenuBar
0.99

[Function]

```
public getShowMenuBar(): boolean;
```

This function returns the currently set value for the ShowMenuBar [▶ 408] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowMenuBar [▶ 1794]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowMenuBar
0.100

[Function]

```
public setShowMenuBar(newValue: boolean | null): void;
```

This function sets a new value for the ShowMenuBar [▶ 408] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowMenuBar [▶ 1794]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarPosition
0.101

[Function]

```
public getMenuBarPosition(): string;
```

This function returns the currently set value for the MenuBarPosition [▶ 406] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMenuBarPosition [▶ 1795]

Origin: TcHmiTrendLineChart [▶ 393]

Schema: tchmi:framework#/definitions/ChartMenuBarPosition [▶ 2209]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarPosition
0.102

[Function]

```
public setMenuBarPosition(newValue: string | null): void;
```

This function sets a new value for the MenuBarPosition [▶ 406] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarPosition [▶ 1795]

Origin: TcHmiTrendLineChart [▶ 393]

Schema: tchmi:framework#/definitions/ChartMenuBarPosition [▶ 2209]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarLabelFontFamily 0.103

[Function]

```
public getMenuBarLabelFontFamily(): string;
```

This function returns the currently set value for the MenuBarLabelFontFamily [▶ 405] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMenuBarLabelFontFamily [▶ 1796]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarLabelFontFamily 0.104

[Function]

```
public setMenuBarLabelFontFamily(newValue: string | null): void;
```

This function sets a new value for the MenuBarLabelFontFamily [▶ 405] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarLabelFontFamily [▶ 1796]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarLabelFontSize 0.105

[Function]

```
public getMenuBarLabelFontSize(): number;
```

This function returns the currently set value for the MenuBarLabelFontSize [▶ 405] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setMenuBarLabelFontSize [▶ 1797]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarLabelFontSize 0.106

[Function]

```
public setMenuBarLabelFontSize (newValue: number | null): void;
```

This function sets a new value for the MenuBarLabelFontSize [▶ 405] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarLabelFontSize [▶ 1797]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarLabelFontSizeUnit 0.107

[Function]

```
public getMenuBarLabelFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the MenuBarLabelFontSizeUnit [▶ 406] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setMenuBarLabelFontSizeUnit [▶ 1798]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarLabelFontSizeUnit 0.108

[Function]

```
public setMenuBarLabelFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the MenuBarLabelFontSizeUnit [▶ 406] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarLabelFontSizeUnit [▶ 1798]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarLabelFontWeight 0.109

[Function]

```
public getMenuBarLabelFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the MenuBarLabelFontWeight [▶ 406] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setMenuBarLabelFontWeight [▶ 1799]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarLabelFontWeight 0.110

[Function]

```
public setMenuBarLabelFontWeight (newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the MenuBarLabelFontWeight [▶ 406] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarLabelFontWeight [▶ 1799]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getMenuBarLabelFontStyle 0.111

[Function]

```
public getMenuBarLabelFontStyle(): TcHmi:FontStyle;
```

This function returns the currently set value for the MenuBarLabelFontStyle [▶ 406] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontStyle [▶ 2087]	The current value of the attribute.

See also

Attribute setter: setMenuBarLabelFontStyle [▶ 1800]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setMenuBarLabelFontStyle 0.112

[Function]

```
public setMenuBarLabelFontStyle(newValue: TcHmi.FontStyle | null): void;
```

This function sets a new value for the MenuBarLabelFontStyle [▶ 406] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontStyle, [▶ 2087] null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMenuBarLabelFontStyle [▶ 1800]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getShowTooltip
0.113

[Function]

```
public getShowTooltip(): boolean;
```

This function returns the currently set value for the ShowTooltip [▶ 408] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setShowTooltip [▶ 1801]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setShowTooltip
0.114

[Function]

```
public setShowTooltip(newValue: boolean | null): void;
```

This function sets a new value for the ShowTooltip [▶ 408] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getShowTooltip [▶ 1801]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipFormat
0.115

[Function]

```
public getTooltipFormat(): TcHmi.IFunction;
```

This function returns the currently set value for the TooltipFormat [▶ 412] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.IFunction [▶ 1891]	The current value of the attribute.

See also

Attribute setter: setTooltipFormat [▶ 1802]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipFormat
0.116

[Function]

```
public setTooltipFormat(newValue: TcHmi.IFunction | null): void;
```

This function sets a new value for the TooltipFormat [▶ 412] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.IFunction [▶ 1891], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFormat [▶ 1801]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipFontFamily
0.117

[Function]

```
public getTooltipFontFamily(): string;
```

This function returns the currently set value for the TooltipFontFamily [▶ 411] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipFontFamily [▶ 1803]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipFontFamily
0.118

[Function]

```
public setTooltipFontFamily(newValue: string | null): void;
```

This function sets a new value for the TooltipFontFamily [▶ 411] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontFamily [▶ 1802]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipFontSize
0.119

[Function]

```
public getTooltipFontSize(): number;
```

This function returns the currently set value for the TooltipFontSize [▶ 411] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setTooltipFontSize [▶ 1804]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipFontSize
0.120

[Function]

```
public setTooltipFontSize(newValue: number | null): void;
```

This function sets a new value for the TooltipFontSize [▶ 411] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontSize [▶ 1803]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipFontSizeUnit
0.121

[Function]

```
public getTooltipFontSizeUnit(): TcHmi.DimensionUnit;
```

This function returns the currently set value for the TooltipFontSizeUnit [▶ 412] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.DimensionUnit [▶ 2086]	The current value of the attribute.

See also

Attribute setter: setTooltipFontSizeUnit [▶ 1805]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipFontSizeUnit
0.122

[Function]

```
public setTooltipFontSizeUnit(newValue: TcHmi.DimensionUnit | null): void;
```

This function sets a new value for the TooltipFontSizeUnit [▶ 412] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.DimensionUnit [▶ 2086], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontSizeUnit [▶ 1804]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 getTooltipFontWeight
0.123

[Function]

```
public getTooltipFontWeight(): TcHmi.FontWeight;
```

This function returns the currently set value for the TooltipFontWeight [▶ 412] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.FontWeight [▶ 2088]	The current value of the attribute.

See also

Attribute setter: setTooltipFontWeight [▶ 1806]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 setTooltipFontWeight
0.124

[Function]

```
public setTooltipFontWeight(newValue: TcHmi.FontWeight | null): void;
```

This function sets a new value for the TooltipFontWeight [▶ 412] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getTooltipFontWeight [▶ 1805]

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 stop
0.125

[Function]

```
public stop(): void;
```

Fixes the current view.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value.

See also

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 resetZoom
0.126

[Function]

```
public resetZoom(): void;
```

Restores the stopped state. This is the state before all user interactions.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value.

See also

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 reset
0.127

[Function]

```
public reset(): void;
```

Restores the state before stopping. Start and end value become active again.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value.

See also

Origin: TcHmiTrendLineChart [▶ 393]

Note: Available since version 1.10

15.1.1.2.2.2 YAxis
0.128

[Interface]

```
export interface YAxis {
    id: number;
    position: TcHmiCharting.IAxis.Position;
    mainTickMinValue?: number;
    mainTickMaxValue?: number;
    showLabels?: boolean;
    labelFontColor?: TcHmi.SolidColor;
    axisColor?: TcHmi.SolidColor;
    showAxisName?: boolean;
    axisName?: string;
    axisNameFontFamily?: string;
    axisNameFontSize?: number;
    axisNameFontSizeUnit?: TcHmi.DimensionUnit;
    axisNameFontWeight?: TcHmi.FontWeight;
    axisNameFontColor?: TcHmi.SolidColor;
    decimalPlaces?: number;
```

```
    unit?: string;
    autoScaling?: boolean;
}
```

Properties

Name	Type	Description
id	number [▶ 2108]	id of the y-axis
position	TcHmiCharting.IAxis.Position [▶ 2109]	Position as enumeration
mainTickMinValue	number [▶ 2108]	Minimum value of the main axis marker
mainTickMaxValue	number [▶ 2108]	Maximum value of the main axis marker
showLabels	boolean [▶ 2107]	Is the label to be displayed
labelFontColor (optional)	TcHmi.SolidColor [▶ 2084]	Color of the label text
axisColor (optional)	TcHmi.SolidColor [▶ 2084]	Axis color
showAxisName (optional)	boolean [▶ 2107]	Is the axis name to be displayed
axisName (optional)	string [▶ 2108]	Name of the axis
axisNameFontFamily (optional)	string [▶ 2108]	Font of the axis name
axisNameFontSize (optional)	number [▶ 2108]	Font size of the axis name
axisNameFontSizeUnit (optional)	TcHmi.DimensionUnit [▶ 2086]	Unit of the font size of the axis name
axisNameFontWeight (optional)	TcHmi.FontWeight [▶ 2088]	Font weight of the axis name
axisNameFontColor (optional)	TcHmi.SolidColor [▶ 2084]	Font color of the axis name
decimalPlaces (optional)	number [▶ 2108]	Decimal places of the axis labeling
unit (optional)	string [▶ 2108]	Display unit
autoScaling (optional)	boolean [▶ 2107]	Should the axis adapt automatically

See also

Schema: tchmi:framework#/definitions/ChartYAxisDefinition [▶ 2209]

Note: Available since version 1.10

15.1.1.2.2.2 LineGraphDescription 0.129

[Interface]

```
export interface LineGraphDescription {
    symbol: string;
    yAxisId?: number;
    lineWidth?: number;
    lineColor?: TcHmi.SolidColor;
    pointDot?: boolean;
    pointDotInStopMode?: boolean;
    pointDotRadius?: number;
    pointDotFillColor?: TcHmi.SolidColor;
    pointDotStrokeWidth?: number;
    pointDotStrokeColor?: TcHmi.SolidColor;
}
```

Properties

Name	Type	Description
symbol	string [▶ 2108]	Symbolname
yAxisId (optional)	number [▶ 2108]	id of the y-axis
lineWidth (optional)	number [▶ 2108]	Line width
lineColor (optional)	TcHmi.SolidColor [▶ 2084]	Line color
pointDot	boolean [▶ 2107]	Are points to be drawn
pointDotInStopMode (optional)	boolean [▶ 2107]	Should points be drawn in stop mode
pointDotRadius (optional)	number [▶ 2108]	Radius of the point circles
pointDotFillColor (optional)	TcHmi.SolidColor [▶ 2084]	Fill color of the point circle
pointDotStrokeWidth (optional)	number [▶ 2108]	Stroke width of the point circle
pointDotStrokeColor (optional)	TcHmi.SolidColor [▶ 2084]	Stroke color of the point circle

See also

Schema: tchmi:framework#/definitions/TrendLineGraphDescriptionDefinitionList [▶ 2263]

Note: Available since version 1.10

15.1.1.2.2.2 TooltipInformationObject 0.130

[Interface]

```
export interface TooltipInformationObject {
    point: { x: number, y: number };
    symbolname: string;
    yAxis: TcHmiTrendLineChart.YAxis;
}
```

Properties

Name	Type	Description
point	{x: number, y: number}	Point with x and y value
symbolname	string [▶ 2108]	Symbolname
yAxis	TcHmiTrendLineChart.YAxis [▶ 1807]	Y axis

See also

Schema: tchmi:framework#/definitions/TrendLineChartTooltipInformationObject [▶ 2266]

tchmi:framework#/definitions/TrendLineChartTooltipInformationList [▶ 2265]

Note: Available since version 1.10

15.1.1.2.2.2 TooltipErrorInformationObject 0.131

[Interface]

```
export interface TooltipInformationObject {
    point: { x: number, y: number };
    symbolname: string;
    yAxis: TcHmiTrendLineChart.YAxis;
}
```

Properties

Name	Type	Description
yAxis	TcHmiTrendLineChart.YAxis [▶ 1807]	Y axis
beginTime	number [▶ 2108]	Start time of the error
EndTime	number [▶ 2108]	End time of the error
errorCode(optional)	number [▶ 2108] | null [▶ 2108] | undefined [▶ 2108]	Error code
symbolname	string [▶ 2108]	Symbolname

See also

Schema: tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationObject [▶ 2265]

tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationList [▶ 2264]

Note: Available since version 1.10

15.1.1.2.2.2 MenuBarPosition 0.132

[Enumeration]

```
export enum MenuBarPosition {
     Bottom = 0,
     Top = 1
}
```

The menu bar can be positioned at the top or bottom.

Values

Name	Value	Description
Bottom	0	Menu bar is positioned at the bottom.
Top	1	Menu bar is positioned at the top.

See also

Schema: tchmi:framework#/definitions/ChartMenuBarPosition [▶ 2209]

Note: Available since version 1.10

15.1.1.2.2.2 AxisPosition 0.133

[Enumeration]

```
export enum AxisPosition {
     Left = 0,
     Right = 1
}
```

The axes can be positioned on the right or left.

Values

Name	Value	Description
Left	0	Axis positioned on the left.
Right	1	Axis positioned on the right.

Note: Available since version 1.10

15.1.1.2.2.2 MouseMode
0.134

[Enumeration]

```
export enum MouseMode {
     ZoomX,
     ZoomXY,
     PanX,
     PanXY
}
```

Values

Name	Value	Description
ZoomX	0	Zoom in x-direction only.
ZoomXY	1	Zoom in x- and y-direction.
PanX	2	Move in x-direction only.
PanXY	3	Move in x- and y-direction.

See also

Schema: tchmi:framework#/definitions/ChartMouseMode [▶ 2209]

Note: Available since version 1.10

15.1.1.2.2.2 Datatype
0.135

[Enumeration]

```
export enum Datatype {
     Datetime,
     Timespan,
     Keyword
}
```

Values

Name	Value	Description
Datetime	`0`	Datetime String. Example: 2018-01-29T10:00Z refers to 29 January 2018 at 10 am UTC time.
Timespan	`1`	Timespan String. Example: PT1H10M indicates a time interval of 1 hour and 10 minutes in ISO8601 format.
Keyword	`2`	String Keywords. Previously `First` in the Start attribute and `Latest` in the End attribute.

Note: Available since version 1.10

15.1.1.2.2.2 TcHmiVideo
1

[Class]

Display area for a video.

Attribute functions

Name	Origin	Description
getBorderRadius [▶ 1154]	TcHmiControl	Getter for the BorderRadius attribute.
setBorderRadius [▶ 1154]	TcHmiControl	Setter for the BorderRadius attribute.
getBorderStyle [▶ 1153]	TcHmiControl	Getter for the BorderStyle attribute.
setBorderStyle [▶ 1153]	TcHmiControl	Setter for the BorderStyle attribute.
getBorderWidth [▶ 1152]	TcHmiControl	Getter for the BorderWidth attribute.
setBorderWidth [▶ 1152]	TcHmiControl	Setter for the BorderWidth attribute.
getBackgroundColor [▶ 1154]	TcHmiControl	Getter for the BackgroundColor attribute.
setBackgroundColor [▶ 1155]	TcHmiControl	Setter for the BackgroundColor attribute.
getBorderColor [▶ 1151]	TcHmiControl	Getter for the BorderColor attribute.
setBorderColor [▶ 1151]	TcHmiControl	Setter for the BorderColor attribute.
getLeft [▶ 1128]	TcHmiControl	Getter for the Left attribute.
setLeft [▶ 1128]	TcHmiControl	Setter for the Left attribute.
getLeftUnit [▶ 1129]	TcHmiControl	Getter for the LeftUnit attribute.
setLeftUnit [▶ 1129]	TcHmiControl	Setter for the LeftUnit attribute.
getTop [▶ 1130]	TcHmiControl	Getter for the Top attribute.
setTop [▶ 1130]	TcHmiControl	Setter for the Top attribute.
getTopUnit [▶ 1131]	TcHmiControl	Getter for the TopUnit attribute.
setTopUnit [▶ 1131]	TcHmiControl	Setter for the TopUnit attribute.
getRight [▶ 1139]	TcHmiControl	Getter for the Right attribute.
setRight [▶ 1139]	TcHmiControl	Setter for the Right attribute.
getRightUnit [▶ 1140]	TcHmiControl	Getter for the RightUnit attribute.
setRightUnit [▶ 1140]	TcHmiControl	Setter for the RightUnit attribute.
getBottom [▶ 1137]	TcHmiControl	Getter for the Bottom attribute.
setBottom [▶ 1138]	TcHmiControl	Setter for the Bottom attribute.
getBottomUnit [▶ 1138]	TcHmiControl	Getter for the BottomUnit attribute.
setBottomUnit [▶ 1138]	TcHmiControl	Setter for the BottomUnit attribute.
getWidth [▶ 1133]	TcHmiControl	Getter for the Width attribute.
setWidth [▶ 1134]	TcHmiControl	Setter for the Width attribute.
getWidthUnit [▶ 1134]	TcHmiControl	Getter for the WidthUnit attribute.
setWidthUnit [▶ 1135]	TcHmiControl	Setter for the WidthUnit attribute.
getHeight [▶ 1132]	TcHmiControl	Getter for the Height attribute.
setHeight [▶ 1132]	TcHmiControl	Setter for the Height attribute.
getHeightUnit [▶ 1132]	TcHmiControl	Getter for the HeightUnit attribute.
setHeightUnit [▶ 1133]	TcHmiControl	Setter for the HeightUnit attribute.
getMaxWidth [▶ 1144]	TcHmiControl	Getter for the MaxWidth attribute.
setMaxWidth [▶ 1145]	TcHmiControl	Setter for the MaxWidth attribute.
getMaxWidthUnit [▶ 1145]	TcHmiControl	Getter for the MaxWidthUnit attribute.
setMaxWidthUnit [▶ 1146]	TcHmiControl	Setter for the MaxWidthUnit attribute.

Name	Origin	Description
getMinWidth [▶ 1141]	TcHmiControl	Getter for the MinWidth attribute.
setMinWidth [▶ 1141]	TcHmiControl	Setter for the MinWidth attribute.
getMinWidthUnit [▶ 1142]	TcHmiControl	Getter for the MinWidthUnit attribute.
setMinWidthUnit [▶ 1142]	TcHmiControl	Setter for the MinWidthUnit attribute.
getMaxHeight [▶ 1146]	TcHmiControl	Getter for the MaxHeight attribute.
setMaxHeight [▶ 1147]	TcHmiControl	Setter for the MaxHeight attribute.
getMaxHeightUnit [▶ 1147]	TcHmiControl	Getter for the MaxHeightUnit attribute.
setMaxHeightUnit [▶ 1148]	TcHmiControl	Setter for the MaxHeightUnit attribute.
getMinHeight [▶ 1143]	TcHmiControl	Getter for the MinHeight attribute.
setMinHeight [▶ 1143]	TcHmiControl	Setter for the MinHeight attribute.
getMinHeightUnit [▶ 1144]	TcHmiControl	Getter for the MinHeightUnit attribute.
setMinHeightUnit [▶ 1144]	TcHmiControl	Setter for the MinHeightUnit attribute.
getRenderedLeft [▶ 1148]	TcHmiControl	Getter for the RenderedLeft attribute.
getRenderedTop [▶ 1149]	TcHmiControl	Getter for the RenderedTop attribute.
getRenderedBottom [▶ 1149]	TcHmiControl	Getter for the RenderedBottom attribute.
getRenderedRight [▶ 1149]	TcHmiControl	Getter for the RenderedRight attribute.
getRenderedWidth [▶ 1150]	TcHmiControl	Getter for the RenderedWidth attribute.
getRenderedHeight [▶ 1150]	TcHmiControl	Getter for the RenderedHeight attribute.
getOpacity [▶ 1164]	TcHmiControl	Getter for the Opacity attribute.
setOpacity [▶ 1164]	TcHmiControl	Setter for the Opacity attribute.
getVisibility [▶ 1163]	TcHmiControl	Getter for the Visibility attribute.
setVisibility [▶ 1163]	TcHmiControl	Setter for the Visibility attribute.
getTransform [▶ 1165]	TcHmiControl	Getter for the Transform attribute.
setTransform [▶ 1165]	TcHmiControl	Setter for the Transform attribute.
getGridColumnIndex [▶ 1126]	TcHmiControl	Getter for the GridColumnIndex attribute.
setGridColumnIndex [▶ 1126]	TcHmiControl	Setter for the GridColumnIndex attribute.
getGridRowIndex [▶ 1125]	TcHmiControl	Getter for the GridRowIndex attribute.
setGridRowIndex [▶ 1125]	TcHmiControl	Setter for the GridRowIndex attribute.
getZindex [▶ 1166]	TcHmiControl	Getter for the Zindex attribute.
setZindex [▶ 1166]	TcHmiControl	Setter for the Zindex attribute.
getWidthMode [▶ 1819]	TcHmiVideo	Getter for the WidthMode attribute.
setWidthMode [▶ 1819]	TcHmiVideo	Setter for the WidthMode attribute.
getHeightMode [▶ 1818]	TcHmiVideo	Getter for the HeightMode attribute.
setHeightMode [▶ 1818]	TcHmiVideo	Setter for the HeightMode attribute.

Name	Origin	Description
getOriginalWidth [▶ 1826]	TcHmiVideo	Getter for the OriginalWidth attribute.
getOriginalHeight [▶ 1826]	TcHmiVideo	Getter for the OriginalHeight attribute.
getSrcList [▶ 1820]	TcHmiVideo	Getter for the SrcList attribute.
setSrcList [▶ 1820]	TcHmiVideo	Setter for the SrcList attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getId [▶ 1123]	TcHmiControl	Getter for the id attribute.
getType [▶ 1168]	TcHmiControl	Getter for the type attribute.
getIsEnabled [▶ 1124]	TcHmiControl	Getter for the IsEnabled attribute.
setIsEnabled [▶ 1124]	TcHmiControl	Setter for the IsEnabled attribute.
getClassNames [▶ 1126]	TcHmiControl	Getter for the ClassNames attribute.
setClassNames [▶ 1127]	TcHmiControl	Setter for the ClassNames attribute.
getIsAttached [▶ 1127]	TcHmiControl	Getter for the IsAttached attribute.
getTooltip [▶ 1166]	TcHmiControl	Getter for the Tooltip attribute.
setTooltip [▶ 1167]	TcHmiControl	Setter for the Tooltip attribute.
getAccessConfig [▶ 1169]	TcHmiControl	Getter for the AccessConfig attribute.
setAccessConfig [▶ 1169]	TcHmiControl	Setter for the AccessConfig attribute.
getBackgroundImage [▶ 1155]	TcHmiControl	Getter for the BackgroundImage attribute.
setBackgroundImage [▶ 1156]	TcHmiControl	Setter for the BackgroundImage attribute.
getBackgroundImageHeight [▶ 1159]	TcHmiControl	Getter for the BackgroundImageHeight attribute.
setBackgroundImageHeight [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHeight attribute.
getBackgroundImageHeightUnit [▶ 1156]	TcHmiControl	Getter for the BackgroundImageHeightUnit attribute.
setBackgroundImageHeightUnit [▶ 1157]	TcHmiControl	Setter for the BackgroundImageHeightUnit attribute.
getBackgroundImageWidth [▶ 1157]	TcHmiControl	Getter for the BackgroundImageWidth attribute.
setBackgroundImageWidth [▶ 1158]	TcHmiControl	Setter for the BackgroundImageWidth attribute.
getBackgroundImageWidthUnit [▶ 1158]	TcHmiControl	Getter for the BackgroundImageWidthUnit attribute.
setBackgroundImageWidthUnit [▶ 1159]	TcHmiControl	Setter for the BackgroundImageWidthUnit attribute.
getBackgroundImagePadding [▶ 1162]	TcHmiControl	Getter for the BackgroundImagePadding attribute.

Name	Origin	Description
setBackgroundImagePadding [▶ 1162]	TcHmiControl	Setter for the BackgroundImagePadding attribute.
getBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Getter for the BackgroundImageHorizontalAlignment attribute.
setBackgroundImageHorizontalAlignment [▶ 1160]	TcHmiControl	Setter for the BackgroundImageHorizontalAlignment attribute.
getBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Getter for the BackgroundImageVerticalAlignment attribute.
setBackgroundImageVerticalAlignment [▶ 1161]	TcHmiControl	Setter for the BackgroundImageVerticalAlignment attribute.
getVolume [▶ 1821]	TcHmiVideo	Getter for the Volume attribute.
setVolume [▶ 1821]	TcHmiVideo	Setter for the Volume attribute.
getMute [▶ 1822]	TcHmiVideo	Getter for the Mute attribute.
setMute [▶ 1822]	TcHmiVideo	Setter for the Mute attribute.
getPlaybackrate [▶ 1823]	TcHmiVideo	Getter for the Playbackrate attribute.
setPlaybackrate [▶ 1823]	TcHmiVideo	Setter for the Playbackrate attribute.
getCurrentTime [▶ 1825]	TcHmiVideo	Getter for the CurrentTime attribute.
setCurrentTime [▶ 1825]	TcHmiVideo	Setter for the CurrentTime attribute.
getDuration [▶ 1825]	TcHmiVideo	Getter for the Duration attribute.
getIsPlaying [▶ 1824]	TcHmiVideo	Getter for the isPlaying attribute.
getIsEnded [▶ 1824]	TcHmiVideo	Getter for the isEnded attribute.
getControls [▶ 1827]	TcHmiVideo	Getter for the Controls attribute.
setControls [▶ 1827]	TcHmiVideo	Setter for the Controls attribute.
getAutoplay [▶ 1828]	TcHmiVideo	Getter for the Autoplay attribute.
setAutoplay [▶ 1828]	TcHmiVideo	Setter for the Autoplay attribute.
getLoop [▶ 1829]	TcHmiVideo	Getter for the Loop attribute.
setLoop [▶ 1829]	TcHmiVideo	Setter for the Loop attribute.
getPoster [▶ 1830]	TcHmiVideo	Getter for the Poster attribute.
setPoster [▶ 1830]	TcHmiVideo	Setter for the Poster attribute.
getBoxShadow [▶ 1167]	TcHmiControl	Getter for the BoxShadow attribute.
setBoxShadow [▶ 1168]	TcHmiControl	Setter for the BoxShadow attribute.
play [▶ 1830]	TcHmiControl	Start video
pause [▶ 1831]	TcHmiControl	Pause video.
stop [▶ 1831]	TcHmiControl	Stop video
skipBackward [▶ 1832]	TcHmiControl	Rewind video
skipForward [▶ 1832]	TcHmiControl	Forward video

Functions

Name	Origin	Description
getElement [▶ 1170]	TcHmiControl	Returns the root HTML element of the control.
__previnit [▶ 1170]	TcHmiControl	Pre-initialization
__init [▶ 1171]	TcHmiControl	Initialization
__attach [▶ 1171]	TcHmiControl	Code after insertion in the DOM
__detach [▶ 1172]	TcHmiControl	Code after removal from the DOM
destroy [▶ 1172]	TcHmiControl	Clean-up after deleting the control.
getIsDestroyed [▶ 1173]	TcHmiControl	Query whether control was deleted.
getAttributeDefaultValueInternal [▶ 1173]	TcHmiControl	Query of the default value of a property.

Inheritance hierarchy

Control [▶ 1118]

Video

See also

Reference [▶ 421] of this control

Note: Available since version 1.8

15.1.1.2.2.2 getHeightMode 1.1

[Function]

```
public getHeightMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the HeightMode [▶ 427] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setHeightMode [▶ 1818]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setHeightMode 1.2

[Function]

```
public setHeightMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the HeightMode [▶ 427] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getHeightMode [▶ 1818]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getWidthMode
1.3

[Function]

```
public getWidthMode(): TcHmi.SizeModeWithContent;
```

This function returns the currently set value for the WidthMode [▶ 427] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SizeModeWithContent [▶ 2096]	The current value of the attribute.

See also

Attribute setter: setWidthMode [▶ 1819]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setWidthMode
1.4

[Function]

```
public setWidthMode(newValue: TcHmi.SizeModeWithContent | null): void;
```

This function sets a new value for the WidthMode [▶ 427] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.SizeModeWithContent [▶ 2096], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getWidthMode [▶ 1819]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getSrcList
1.5

[Function]

```
public getSrcList(): TcHmi.Controls.Beckhoff.TcHmiVideo.VideoSource[] | null;
```

This function returns the currently set value for the SrcList [▶ 428] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.Beckhoff.TcHmiVideo.VideoSource [▶ 1833][], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setSrcList [▶ 1820]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setSrcList
1.6

[Function]

```
public setSrcList(newValue: TcHmi.Controls.Beckhoff.TcHmiVideo.VideoSource[] | null): void;
```

This function sets a new value for the SrcList [▶ 428] attribute.

Parameter

Name	Type	Description
newValue	TcHmi.Controls.Beckhoff.TcHmiVideo.VideoSource [▶ 1833][], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getSrcList [▶ 1820]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getVolume
1.7

[Function]

```
public getVolume(): number;
```

This function returns the currently set value for the Volume [▶ 428] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setVolume [▶ 1821]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setVolume
1.8

[Function]

```
public setVolume(newValue: number | null): void;
```

This function sets a new value for the Volume [▶ 428] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getVolume [▶ 1821]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getMute
1.9

[Function]

```
public getMute(): boolean;
```

This function returns the currently set value for the Mute [▶ 428] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setMute [▶ 1822]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setMute
1.10

[Function]

```
public setMute(newValue: boolean | null): void;
```

This function sets a new value for the Mute [▶ 428] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getMute [▶ 1822]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getPlaybackrate
1.11

[Function]

```
public getPlaybackrate(): number;
```

This function returns the currently set value for the Playbackrate [▶ 429] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setPlaybackrate [▶ 1823]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setPlaybackrate
1.12

[Function]

```
public setPlaybackrate(newValue: number | null): void;
```

This function sets a new value for the Playbackrate [▶ 429] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getPlaybackrate [▶ 1823]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getIsEnded
1.13

[Function]

```
public getIsEnded(): boolean;
```

This function returns the currently set value for the isEnded [▶ 429] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getIsPlaying
1.14

[Function]

```
public getIsPlaying(): boolean;
```

This function returns the currently set value for the isPlaying [▶ 429] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getCurrentTime 1.15

[Function]

```
public getCurrentTime(): number;
```

This function returns the currently set value for the CurrentTime [▶ 429] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setCurrentTime [▶ 1825]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setCurrentTime 1.16

[Function]

```
public setCurrentTime(newValue: number | null): void;
```

This function sets a new value for the CurrentTime [▶ 429] attribute.

Parameter

Name	Type	Description
newValue	number [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getCurrentTime [▶ 1825]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getDuration 1.17

[Function]

```
public getDuration(): number;
```

This function returns the currently set value for the Duration [▶ 430] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getOriginalHeight
1.18

[Function]

```
public getOriginalHeight(): number;
```

This function returns the currently set value for the OriginalHeight [▶ 430] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getOriginalWidth
1.19

[Function]

```
public getOriginalWidth(): number;
```

This function returns the currently set value for the OriginalWidth [▶ 430] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current value of the attribute.

See also

This attribute is read-only, so it has no setter.

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getControls
1.20

[Function]

```
public getControls(): boolean;
```

This function returns the currently set value for the Controls [▶ 430] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setControls [▶ 1827]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setControls
1.21

[Function]

```
public setControls(newValue: boolean | null): void;
```

This function sets a new value for the Controls [▶ 430] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getControls [▶ 1827]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getAutoplay
1.22

[Function]

```
public getAutoplay(): boolean;
```

This function returns the currently set value for the Autoplay [▶ 431] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setAutoplay [▶ 1828]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setAutoplay
1.23

[Function]

```
public setAutoplay(newValue: boolean | null): void;
```

This function sets a new value for the Autoplay [▶ 431] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getAutoplay [▶ 1828]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getLoop
1.24

[Function]

```
public getLoop(): boolean;
```

This function returns the currently set value for the Loop [▶ 431] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The current value of the attribute.

See also

Attribute setter: setLoop [▶ 1829]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setLoop
1.25

[Function]

```
public setLoop(newValue: boolean | null): void;
```

This function sets a new value for the Loop [▶ 431] attribute.

Parameter

Name	Type	Description
newValue	boolean [▶ 2107], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getLoop [▶ 1829]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 getPoster 1.26

[Function]

```
public getPoster(): string | null;
```

This function returns the currently set value for the Poster [▶ 431] attribute.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The current value of the attribute.

See also

Attribute setter: setPoster [▶ 1830]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 setPoster 1.27

[Function]

```
public setPoster(newValue: string | null): void;
```

This function sets a new value for the Poster [▶ 431] attribute.

Parameter

Name	Type	Description
newValue	string [▶ 2108], null [▶ 2108]	The new value of the attribute. If null is passed, the internal default value is used.

Return value

Type	Description
void	No return value.

See also

Attribute getter: getPoster [▶ 1830]

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 play 1.28

[Function]

```
public play(): void;
```

Starts the video. This function only has effect if the control is active [▶ 439] and the current user has the operate [▶ 527] right for that control.

Caution: In some browsers an application-based start of a video is not allowed.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value

See also

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 pause
1.29

[Function]

```
public pause(): void;
```

Pauses the video.

Starts the video. This function only has effect if the control is active [▶ 439] and the current user has the operate [▶ 527] right for that control.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value

See also

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 stop
1.30

[Function]

```
public stop(): void;
```

Ends the video. This function only has effect if the control is active [▶ 439] and the current user has the operate [▶ 527] right for that control.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value

See also

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 skipBackward 1.31

[Function]

```
public skipBackward(newValue: number): void;
```

Rewinds the video by a certain time. This function only has effect if the control is active [▶ 439] and the current user has the operate [▶ 527] right for that control.

Parameter

Name	Type	Description
newValue	number [▶ 2108]	Difference time in seconds

Return value

Type	Description
void	No return value

See also

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 skipForward 1.32

[Function]

```
public skipForward(newValue: number): void;
```

Fast-forwards the video by a certain amount of time. This function only has effect if the control is active [▶ 439] and the current user has the operate [▶ 527] right for that control.

Parameter

Name	Type	Description
newValue	number [▶ 2108]	Difference time in seconds

Return value

Type	Description
void	No return value

See also

Origin: TcHmiVideo [▶ 421]

Note: Available since version 1.8

15.1.1.2.2.2 VideoSource
1.33

[Interface]

```
export interface VideoSource {
    source: string;
    type: string;
}
```

See also

Scheme: tchmi:framework#/definitions/VideoSourceList [▶ 2267]

Note: Available since version 1.8

15.1.1.2.3 get

[Function]

```
export function get(
    name: string | null | undefined
): TcHmi.Controls.System.baseTcHmiControl | undefined;
```

Returns a control of a given name.

Parameter

Name	Type	Description
name	string [▶ 2108], null [▶ 2108], undefined [▶ 2108]	Name of the required control.

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118], undefined [▶ 2108]	The requested control object or undefined if no control with this name exists.

Note: Available since version 1.8

Example - JavaScript

```
var someControl = TcHmi.Controls.get('TcHmiButton');
if(someControl !== undefined) {
    someControl.setText('found you');
}
```

15.1.1.2.4 register

[Function]

```
function register(
    name: string,
    constructor: TcHmi.Controls.baseTcHmiControlConstructor,
    directory: string,
    templatePath?: string | null
): void;
```

Registers a control class in the system.

Before using a control, it must be made known to the system using this function.

Parameter

Name	Type	Description
name	string [▶ 2108]	Name of the class.
constructor	Contructor	Constructor of the control.
directory	string [▶ 2108]	Name of the control folder.
templatePath	string [▶ 2108], null [▶ 2108]	Optional: Path of a template HTML file to be loaded into the control.

Return value

Type	Description
void	No return value.

Note: Available since version 1.8

Example - JavaScript 1

```
TcHmi.Controls.register('tchmi-mycontrol', Custom.mycontrol, 'Custom/mycontrol/');
```

Example - JavaScript 2

```
TcHmi.Controls.register('tchmi-mycontrol', Custom.mycontrol, 'Custom/mycontrol/', 'Custom/mycontrol/
Template.html');
```

15.1.1.2.5 tachControls

[Function] [Deprecated]

```
export function tachControls(
    callback: null | (
         () => void
    ) = null
): void;
```

Initiates a check to ascertain whether a control has been added to or removed from the DOM and calls the __attach or __detach function of the corresponding control.

Note: From version 1.10, this function is considered obsolete. The system automatically detects whether controls have been added to or removed from the DOM.

Parameter

Name	Type	Description
callback	null [▶ 2108], () => void	The callback function is called when the system has completed the check for new controls in the DOM.

Return value

Type	Description
void	No return value.

Note: Available since version 1.8

Sample - JavaScript

```
// create control
var subControl = TcHmi.ControlFactory.create('tchmi-button', 'newbutton');
if(subControl !== undefined) {
    // append control to ourself
```

```
    myControl.getElement().append(subControl.getElement());
    // force check if controls has been added
    TcHmi.Controls.tachControls();
}
```

15.1.1.3 Functions

[Namespace]

Provides resources for managing functions.

Functions

Name	Description

registerFrameworkFunction [▶ 1835]	Registers a function created within a function package in the framework.
registerFunction [▶ 1835]	Registers a function created within a TwinCAT HMI project in the framework.
getFunction [▶ 1836]	Returns a reference to a function based on the function name.

Note: Available since version 1.8

15.1.1.3.1 registerFrameworkFunction

[Function]

```
function registerFrameworkFunction(
    name: string,
    func: (args: any[]) => any,
    descriptionUrl: string = ''
): void;
```

Registers a function created within a function package in the framework.

Parameter

Name	Type	Description
name	string [▶ 2107]	The name with which the function is registered in the framework.
func	(args: any [▶ 2107][]) => any [▶ 2107]	A reference to the function to be registered.
descriptionUrl	string [▶ 2108]	

Return value

Type	Description
void	No return value

Note: Available since version 1.8

15.1.1.3.2 registerFunction

[Function]

```
function registerFunction(
    name: string,
    func: (...args: any[]) => any
): void;
```

Registers a function created within a TwinCAT HMI project in the framework.

Parameter

Name	Type	Description
name	string [▶ 2108]	The name with which the function is registered in the framework.
`func`	(...args: any [▶ 2107][]) => any [▶ 2107]	A reference to the function to be registered.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

15.1.1.3.3 getFunction

[Function]

```
function getFunction(
    name: string
):((...args: any[]) => any) | undefined;
```

Returns a reference to a function based on the function name.

Parameter

Name	Type	Description
name	string [▶ 2108]	The name of the function.

Return value

Type	Description
((...args: any [▶ 2107][]) => any [▶ 2107]), undefined [▶ 2108]	A reference to the desired function, or undefined in the event of an error if the function does not exist.

Note: Available since version 1.8

15.1.1.4 Access

[Class]

Provides functions for checking control-specific access rights.

Static functions

Name	Description

checkAccess [▶ 1836]	Checks the access rights of a control

Note: Available since version 1.8

15.1.1.4.1 checkAccess

[Function]

```
public static checkAccess(
   control: TcHmi.Controls.System.baseTcHmiControl,
   requestedAccessright: string
): boolean | null;
```

Checks an access right of a control. All controls provide the **operate** and **observe** rights. Controls can define additional rights names and respond accordingly. Rights are always assigned to groups.

If a right is not explicitly defined on a control itself, its parent control is checked. Therefore, a check is only possible after attaching.

Possible situations:

- In Engineering within Visual Studio a right is always granted.
- If a server does not have a password query, a right is always granted.
- If a control is not attached, no decision is possible.
- If a group of the current user permits the right, the right is granted.
- If no group of the current user permits the right and one group denies the right, the right is denied.
- If no group of the current user permits or denies the right, the parent control is checked.
- If no parent control has explicitly set this right, the standard right applies (**operate** and **observe** have "permit" as default).

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Control to be checked
requestedAccessright	string [▶ 2108]	Name of the access right to be checked

Return value

Type	Description
boolean [▶ 2107], null [▶ 2108]	true: The control currently has this right. false: The control does not currently have this right. null: The access right cannot currently be decided. See above.

Note: Available since version 1.8

Example - JavaScript

A right is to be checked in a separate event handler. This must be added in Description.json, otherwise the right cannot be configured in Visual Studio.

```
var myDiv = $('<div>Hello World!</div>');
myDiv.on('click', function(evt) {
   if(TcHmi.Access.checkAccess(myControl, 'doFancyStuff') !== true){
      return; // current user has no right to do this
   }
   // fancy stuff
});
myControl.getElement().append(myDiv);
```

15.1.1.5 Animation

[Class]

Allows you to create and configure animations that can be played using CSS or JavaScript.

The Animation API is designed for use within controls. [▶ 2123]

Constructors

Name	Description
constructor [▶ 1841]	Creates a new animation

Functions

Name	Description
controlName [▶ 1841]	Returns the name of the control for which this animation was created.
selector [▶ 1842]	Returns the CSS selector used to create the animation.
animationName [▶ 1842]	Returns the name generated for this animation.
state [▶ 1842]	Returns the current state of the animation.
addKeyframe [▶ 1843]	Adds a keyframe.
addKeyframe [▶ 1843]	Adds a keyframe.
addKeyframe [▶ 1844]	Adds a keyframe.
clearKeyframes [▶ 1844]	Deletes all keyframes of this animation.
reverseKeyframes [▶ 1845]	Reverses the order in which the keyframes are played.
keyframes [▶ 1845]	Returns the keyframes of this animation.
duration [▶ 1846]	Returns the configured duration of the animation in milliseconds.
duration [▶ 1846]	Sets the duration in milliseconds that this animation should run.
delay [▶ 1847]	Returns the configured delay after which the animation should start in milliseconds.
delay [▶ 1847]	Sets the delay in milliseconds after which this animation should start.
iterationCount [▶ 1848]	Returns the configured number of repetitions the animation should run through.
iterationCount [▶ 1848]	Sets the number of repetitions the animation should run through.
direction [▶ 1849]	Returns the configured direction of the animation.
direction [▶ 1849]	Sets the direction of the animation.
timingFunction [▶ 1850]	Returns the configured timing function.
timingFunction [▶ 1850]	Sets the timing function of the animation.
fillMode [▶ 1851]	Returns whether the first keyframe should be effective before the start of the animation and/or the last keyframe after the end of the animation.
fillMode [▶ 1851]	Specifies whether the first keyframe should be effective before the start of the animation and/or the last keyframe after the end of the animation.
cleanup [▶ 1852]	Returns whether animation-specific CSS should be removed after the end of the animation.
cleanup [▶ 1853]	Sets whether animation-specific CSS should be removed after the end of the animation.
useCss [▶ 1853]	Returns whether the animation is played with CSS or JavaScript.
useCss [▶ 1854]	Sets whether, if possible, the animation should be played with CSS or JavaScript.
registerEventHandler [▶ 1854]	Registers a callback for an animation event.
unregisterEventHandler [▶ 1855]	Unregister a callback for an animation event.
eventHandlers [▶ 1856]	Returns all registered event handlers.
run [▶ 1856]	Starts or resumes the animation if paused.
pause [▶ 1857]	Pauses the animation.
skip [▶ 1857]	Skips the animation.
reset [▶ 1858]	Resets the animation to the first keyframe.

Interfaces

Name	Description
Keyframe [▶ 1858]	Defines the state of an animation at a certain point in time.
AnimationEvent [▶ 1859]	Contains information about events that occur during the course of an animation.

Enumerations

Name	Description
Status [▶ 1859]	Defines the states an animation can assume.

Note: Available since version 1.8

15.1.1.5.1 constructor

[Function]

```
public constructor(controlName: string, selector: string);
```

Creates a new animation.

Parameter

Name	Type	Description
controlName	string [▶ 2108]	The name of the control to animate.
selector	string [▶ 2108]	A CSS selector to select an HTML element within the control to be animated. An empty string selects the root element of the control.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
```

Note: Available since version 1.8

15.1.1.5.2 controlName

[Function]

```
public controlName(): string;
```

Returns the name of the control for which this animation was created.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The name of the control for which this animation was created.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var name = animation.controlName(); // 'ViewDesktopBeckhoffLogo'
```

Note: Available since version 1.8

15.1.1.5.3 selector

[Function]

```
public selector(): string;
```

Returns the CSS selector used to create the animation.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The CSS selector used to create the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('TcHmiDatagrid', 'td.focused');
var selector = animation.selector(); // 'td.focused'
```

Note: Available since version 1.8

15.1.1.5.4 animationName

[Function]

```
public animationName(): string;
```

Returns the name generated for this animation. This name is used in CSS declarations and animation events.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The name of the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var name = animation.animationName(); // 'anim-f5c1b905-a308-4559-d85b-33bf0a2656ca'
```

Note: Available since version 1.8

15.1.1.5.5 state

[Function]

```
public state(): TcHmi.Animation.Status;
```

Returns the current state of the animation.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation.Status [▶ 1859]	The current status of the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var state1 = animation.state(); // TcHmi.Animation.Status.CONFIGURE
animation.addKeyframe('top', '20px', 0).addKeyframe('top', '50px', 1);
animation.run();
var state2 = animation.state(); // TcHmi.Animation.Status.INITIALIZED
```

Note: Available since version 1.8

15.1.1.5.6 addKeyframe

[Function]

```
public addKeyframe(keyframe: TcHmi.Animation.Keyframe): this;
```

Adds a keyframe.

Parameter

Name	Type	Description
keyframe	TcHmi.Animation.Keyframe [▶ 1858]	Keyframe to be added.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.addKeyframe({
    styles: { top: ['20px'], left: ['20px'] },
    progressPoint: 0
}).addKeyframe({
    styles: { top: ['50px'], left: ['50px'] },
    progressPoint: 1
});
```

Note: Available since version 1.8

15.1.1.5.7 addKeyframe

[Function]

```
public addKeyframe(styles: Dictionary<string | string[]>, progressPoint: number): this;
```

Adds a keyframe.

Parameter

Name	Type	Description
styles	Dictionary [▶ 2086]<string [▶ 2108] | string [▶ 2108][]>	A collection of CSS properties and their desired values.
progressPoint	number [▶ 2108]	A value between 0 and 1 inclusive, which determines the point in time of the animation. 0 is the start and 1 the end of the animation.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.addKeyframe({ top: '20px', left: '20px' }, 0)
    .addKeyframe({ top: '50px', left: '50px' }, 1);
```

Note: Available since version 1.8

15.1.1.5.8 addKeyframe

[Function]

```
public addKeyframe(property: string, value: string, progessPoint: number): this;
```

Adds a keyframe.

Parameter

Name	Type	Description
property	string [▶ 2108]	The name of a CSS property
value	string [▶ 2108]	The desired value for this CSS property
progressPoint	number [▶ 2108]	A value between 0 and 1 inclusive, which determines the point in time of the animation. 0 is the start and 1 the end of the animation.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.addKeyframe('transform', 'translate(0px, 0px)', 0)
    .addKeyframe('transform', 'translate(30px, 30px)', 1);
```

Note: Available since version 1.8

15.1.1.5.9 clearKeyframes

[Function]

```
public clearKeyframes(): this;
```

Deletes all keyframes of this animation.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.addKeyframe('top', '20px', 0);
animation.clearKeyframes();
var keyframes = animation.keyframes(); // empty array
```

Note: Available since version 1.8

15.1.1.5.10 reverseKeyframes

[Function]

```
public reverseKeyframes(): this;
```

Reverses the order in which the keyframes are played by subtracting the respective `progressPoint` from 1 and setting the result as a new `progressPoint`. Can be used to run animations backwards after being played.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.addKeyframe('top', '20px', 0)
    .addKeyframe('top', '50px', 1);
animation.reverseKeyframes();
var keyframes = animation.keyframes();
// [{ styles: { top: ["20px"] }, progressPoint: 1 }, { styles: { top: ["50px"] }, progressPoint:0 }]
```

Note: Available since version 1.8

15.1.1.5.11 keyframes

[Function]

```
public keyframes(): TcHmi.Animation.Keyframe[];
```

Returns the keyframes of this animation.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation.Keyframe [▶ 1858][]	The keyframes of this animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.addKeyframe('top', '20px', 0)
    .addKeyframe('top', '50px', 1);
var keyframes = animation.keyframes();
// [{ styles: { top: ["20px"] }, progressPoint: 0 }, { styles: { top: ["50px"] }, progressPoint:
1 }]
```

Note: Available since version 1.8

15.1.1.5.12 duration

[Function]

```
public duration(): number;
```

Returns the configured duration of the animation in milliseconds. If no duration has been configured, the default is `0`.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The configured duration of the animation in milliseconds.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var durationDefault = animation.duration(); // 0
animation.duration(1500);
var duration = animation.duration(); // 1500
```

Note: Available since version 1.8

15.1.1.5.13 duration

[Function]

```
public duration(valueNew: number | null): this;
```

Sets the duration in milliseconds that this animation should run.

Parameter

Name	Type	Description
valueNew	number [▶ 2108], null [▶ 2108]	The time in milliseconds that the animation should run. If null is passed, the default value of 0 is used.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var durationDefault = animation.duration(); // 0
animation.duration(1500);
var duration = animation.duration(); // 1500
```

Note: Available since version 1.8

15.1.1.5.14 delay

[Function]

```
public delay(): number;
```

Returns the configured delay after which the animation should start in milliseconds. If no delay has been configured, the default is `0`.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The configured delay of the animation in milliseconds.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var delayDefault = animation.delay(); // 0
animation.delay(500);
var delay = animation.delay(); // 500
```

Note: Available since version 1.8

15.1.1.5.15 delay

[Function]

```
public delay(valueNew: number | null): this;
```

Sets the delay in milliseconds after which this animation should start.

Parameter

Name	Type	Description
valueNew	number [▶ 2108], null [▶ 2108]	The time in milliseconds after which the animation should start. If null is passed, the default value of 0 is used.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var delayDefault = animation.delay(); // 0
animation.delay(500);
var delay = animation.delay(); // 500
```

Note: Available since version 1.8

15.1.1.5.16 iterationCount

[Function]

```
public iterationCount(): number | 'infinite';
```

Returns the configured number of repetitions the animation should run through. If no repetition number has been configured, the default is `1`.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], string [▶ 2108]	The number of repetitions the animation should run through. If the animation is to be repeated indefinitely, the value is `'infinite`

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var iterationCountDefault = animation.iterationCount(); // 1
animation.iterationCount(5);
var iterationCount = animation.iterationCount(); // 5
```

Note: Available since version 1.8

15.1.1.5.17 iterationCount

[Function]

```
public iterationCount(valueNew: number | 'infinite' | null): this;
```

Sets the number of repetitions the animation should run through.

Parameter

Name	Type	Description
valueNew	number [▶ 2108], string [▶ 2108], null [▶ 2108]	The number of repetitions the animation should run through. If null is passed, the default value of 1 is used.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var iterationCountDefault = animation.iterationCount(); // 1
animation.iterationCount('infinite');
var iterationCount = animation.iterationCount(); // 'infinite'
```

Note: Available since version 1.8

15.1.1.5.18 direction

[Function]

```
public direction(): 'normal' | 'reverse' | 'alternate' | 'alternate-reverse';
```

Returns the configured direction of the animation. If no direction has been configured, the default is `'normal'`. The direction not only influences the order of the keyframes but also the timing function.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The direction of the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var directionDefault = animation.direction(); // 'normal'
animation.direction('alternate');
var direction = animation.direction(); // 'alternate'
```

Note: Available since version 1.8

15.1.1.5.19 direction

[Function]

```
public direction(valueNew: 'normal' | 'reverse' | 'alternate' | 'alternate-reverse' | null): this;
```

Sets the direction of the animation. The direction not only affects the order of the keyframes but also the timing function.

Parameter

Name	Type	Description
valueNew	string [▶ 2108], null [▶ 2108]	The direction of the animation. If null is passed, the default value `'normal'` is used. `reverse'` lets the animation run backwards, `'alternate'` reverses the direction at each repetition for animations with more than one repetition and `'alternate-reverse'` works like `'alternate'`, only that it starts with the reverse direction.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var directionDefault = animation.direction(); // 'normal'
animation.direction('alternate');
var direction = animation.direction(); // 'alternate'
```

Note: Available since version 1.8

15.1.1.5.20 timingFunction

[Function]

```
public timingFunction(): string | ((t: number) => number);
```

Returns the configured timing function. If no timing function has been configured, the default is `'ease'`. The timing function can be either a string that represents a valid value for the CSS animation-timing-function property, or a function that accepts a number between 0 and 1 and returns a number.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], (t: number [▶ 2108]) => number [▶ 2108]	The timing function of the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var timingDefault = animation.timingFunction(); // 'ease'
animation.timingFunction('cubic-bezier(0.1, -0.6, 0.2, 0)');
var timing = animation.timingFunction(); // 'cubic-bezier(0.1, -0.6, 0.2, 0)'
```

Note: Available since version 1.8

15.1.1.5.21 timingFunction

[Function]

```
public timingFunction(valueNew: string | ((t: number) => number) | null): this;
```

Sets the timing function of the animation.

Parameter

Name	Type	Description
valueNew	string [▶ 2108], (t: number [▶ 2108]) => number [▶ 2108], null [▶ 2108]	The timing function of the animation. If null is passed, the default value `'ease'` is used. The timing function can be either a string that represents a valid value for the CSS animation-timing-function property, or a function that accepts a number between 0 and 1 and returns a number. If a JavaScript function is passed, the animation cannot be played via CSS.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var timingDefault = animation.timingFunction(); // 'ease'
animation.timingFunction('cubic-bezier(0.1, -0.6, 0.2, 0)');
var timing = animation.timingFunction(); // 'cubic-bezier(0.1, -0.6, 0.2, 0)'
```

Note: Available since version 1.8

15.1.1.5.22 fillMode

[Function]

```
public fillMode(): 'none' | 'forwards' | 'backwards' | 'both';
```

Returns whether the first keyframe should be effective before the start of the animation and/or the last keyframe after the end of the animation. The default is `'none'`.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The fill mode of the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var fillModeDefault = animation.fillMode(); // 'none'
animation.fillMode('both');
var fillMode = animation.fillMode(); // 'both'
```

Note: Available since version 1.8

15.1.1.5.23 fillMode

[Function]

```
public fillMode(valueNew: 'none' | 'forwards' | 'backwards' | 'both' | null): this;
```

Specifies whether the first keyframe should be effective before the start of the animation and/or the last keyframe after the end of the animation.

Parameter

Name	Type	Description
valueNew	string [▶ 2108], null [▶ 2108]	The fill mode of the animation. If null is passed, the default value `'none'` is used. `none'` means that the first keyframe is not effective until the delay has elapsed and the animation starts. At the end of the animation, the animated element returns to its original state. `forwards'` causes the last keyframe to continue working after the animation has expired. `backwards'` causes the first keyframe to take effect before the animation starts, while the delay elapses. `both'` combines `'forwards'` and `'backwards'`.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var fillModeDefault = animation.fillMode(); // 'none'
animation.fillMode('both');
var fillMode = animation.fillMode(); // 'both'
```

Note: Available since version 1.8

15.1.1.5.24 cleanup

[Function]

```
public cleanup(): boolean;
```

Returns whether animation-specific CSS should be removed after the end of the animation. If the fill mode is `forwards` or `both`, the CSS properties of the last keyframe are copied into the `style` attribute of the animated element. The default is `false`.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	Whether animation-specific CSS should be removed after the end of the animation.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var cleanupDefault = animation.cleanup(); // false
animation.cleanup(true);
var cleanup = animation.cleanup(); // true
```

Note: Available since version 1.8

15.1.1.5.25 cleanup

[Function]

```
public cleanup(valueNew: boolean | null): this;
```

Sets whether animation-specific CSS should be removed after the end of the animation. If the fill mode is `forwards` or `both`, the CSS properties of the last keyframe are copied into the `style` attribute of the animated element.

Parameter

Name	Type	Description
valueNew	boolean [▶ 2107], null [▶ 2108]	Whether animation-specific CSS should be removed after the end of the animation. If null is passed, the default value `false` is used.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var cleanupDefault = animation.cleanup(); // false
animation.cleanup(true);
var cleanup = animation.cleanup(); // true
```

Note: Available since version 1.8

15.1.1.5.26 useCss

[Function]

```
public useCss(): boolean;
```

Returns whether the animation is played with CSS or JavaScript. Default is `true`. The animation cannot be played with CSS if features are configured that are not supported by CSS.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	Whether the animation is played with CSS or JavaScript.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var useCssDefault = animation.useCss(); // true
animation.useCss(false);
var useCss = animation.useCss(); // false
```

Note: Available since version 1.8

15.1.1.5.27 useCss

[Function]

```
public useCss(valueNew: boolean | null): this;
```

Sets whether, if possible, the animation should be played with CSS or JavaScript.

Parameter

Name	Type	Description
valueNew	boolean [▶ 2107], null [▶ 2108]	Whether the animation should be played with CSS or JavaScript. If null is passed, the default value `true` is used.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var useCssDefault = animation.useCss(); // true
animation.useCss(false);
var useCss = animation.useCss(); // false
```

Note: Available since version 1.8

15.1.1.5.28 registerEventHandler

[Function]

```
public registerEventHandler(
    name: 'animationstart' | 'animationend' | 'animationiteration',
    callback: (event: TcHmi.Animation.AnimationEvent) => void
): this;
```

Registers a callback for an animation event.

Parameter

Name	Type	Description
name	string [▶ 2108]	The name of the event. `animationstart'` is triggered when the animation starts, after the delay has expired. `animationend'` is triggered when the animation ends. `animationiteration'` is triggered when the animation starts in a new repetition.
callback	(event: TcHmi.Animation.AnimationEvent [▶ 1859]) => void	The function to be called when the event occurs.

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
animation.registerEventHandler('animationstart', (event) => console.log('Animation started: ' +
event.animationName));
```

Note: Available since version 1.8

15.1.1.5.29 unregisterEventHandler

[Function]

```
public unregisterEventHandler(
    name: 'animationstart' | 'animationend' | 'animationiteration',
    callback: (event: TcHmi.Animation.AnimationEvent) => void
): this;
```

Unregister a callback for an animation event.

Parameter

Name	Type	Description
name	string [▶ 2108]	The name of the previously registered event. `animationstart'` is triggered when the animation starts, after the delay has expired. `animationend'` is triggered when the animation ends. `animationiteration'` is triggered when the animation starts in a new repetition.
callback	(event: TcHmi.Animation.AnimationEvent [▶ 1859]) => void	The previously registered function. Must be the same reference as the callback parameter previously passed to registerEventHandler [▶ 1854].

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var eventHandler = function (event) {
    console.log('Animation started: ' + event.animationName);
};
animation.registerEventHandler('animationstart', eventHandler);
var handlers = animation.eventHandlers(); // [{ name: 'animationstart', callback: f(event) }]
animation.unregisterEventHandler('animationstart', eventHandler);
handlers = animation.eventHandlers(); // []
```

Note: Available since version 1.8

15.1.1.5.30 eventHandlers

[Function]

```
public eventHandlers(): {
    name: 'animationstart' | 'animationend' | 'animationiteration';
    callback: (event: TcHmi.Animation.AnimationEvent) => void;
}[];
```

Returns all registered event handlers.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
{ name: string [▶ 2108]; callback: (event: TcHmi.Animation.AnimationEvent [▶ 1859]) => void; }[]	An array of all registered event handlers.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '');
var eventHandler = function (event) {
    console.log('Animation started: ' + event.animationName);
};
animation.registerEventHandler('animationstart', eventHandler);
var handlers = animation.eventHandlers(); // [{ name: 'animationstart', callback: f(event) }]
animation.unregisterEventHandler('animationstart', eventHandler);
handlers = animation.eventHandlers(); // []
```

Note: Available since version 1.8

15.1.1.5.31 run

[Function]

```
public run(): this;
```

Starts or resumes the animation if paused. An animation can only be started if a keyframe is defined with progressPoint 0 and a keyframe with progressPoint 1.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '')
    .addKeyframe('left', '20px', 0)
    .addKeyframe('left', '120px', 1)
    .duration(1000);
animation.run();
```

Note: Available since version 1.8

15.1.1.5.32 pause

[Function]

```
public pause(): this;
```

Pauses the animation. A paused animation can be continued with run() [▶ 1856].

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '')
    .addKeyframe('left', '20px', 0)
    .addKeyframe('left', '520px', 1)
    .duration(2000)
    .run();
window.setTimeout(() => animation.pause(), 700);
window.setTimeout(() => animation.run(), 2000);
```

Note: Available since version 1.8

15.1.1.5.33 skip

[Function]

```
public skip(): this;
```

Skips the animation. The CSS properties of the last keyframe are written to the `style` attribute of the animated element and animation-specific CSS is removed from the DOM. This happens independently of the values set by fillMode() [▶ 1851] or cleanup() [▶ 1853]. When determining the last keyframe, however, the current animation direction [▶ 1849] is taken into account. Has no effect when the animation has expired and a cleanup [▶ 1853] has been performed.

If the animation is to be reset to the initial keyframe, reset() [▶ 1858] can be used.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '')
    .addKeyframe('left', '20px', 0)
    .addKeyframe('left', '520px', 1)
    .duration(2000)
    .run();
window.setTimeout(() => animation.skip(), 700);
```

Note: Available since version 1.8

15.1.1.5.34 reset

[Function]

```
public reset(): this;
```

Resets the animation to the first keyframe. The CSS properties of the first keyframe are written to the `style` attribute of the animated element, but animation-specific CSS is not removed from the DOM. This happens regardless of the values set by fillMode() [▶ 1851] or cleanup() [▶ 1853], if the animation has not expired yet. When determining the first keyframe, however, the current animation direction [▶ 1849] is taken into account. Has no effect when the animation has expired and a cleanup [▶ 1853] has been performed.

If the animation is to be fast-forwarded to the endkeyframe, skip() [▶ 1857] can be used.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Animation [▶ 1837]	This method returns its parent object to allow concatenation of method calls.

Sample - JavaScript

```
var animation = new TcHmi.Animation('ViewDesktopBeckhoffLogo', '')
    .addKeyframe('left', '20px', 0)
    .addKeyframe('left', '520px', 1)
    .duration(2000)
    .run();
window.setTimeout(() => animation.reset(), 700);
```

Note: Available since version 1.8

15.1.1.5.35 Keyframe

[Interface]

```
export interface Keyframe {
    styles: Dictionary<string[]>;
    progressPoint: number;
}
```

Defines the state of an animation at a certain point in time.

Properties

Name	Type	Description
styles	Dictionary [▶ 2086]<string [▶ 2108][]>	Collection of CSS properties and their desired values at this point in time of the animation.
progressPoint	number [▶ 2108]	A value between 0 and 1 inclusive, which determines the point in time of the animation. 0 is the start and 1 the end of the animation.

Note: Available since version 1.8

15.1.1.5.36 AnimationEvent

[Interface]

```
export interface AnimationEvent extends IResultObject {
    animationName: string;
    elapsedTime: number;
    JQueryEvent?: JQuery.Event<HTMLElement>;
}
```

Contains information about events that occur during the course of an animation. This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
error	Errors [▶ 2089]	Error state of the request.
details [optional]	any [▶ 2107]	Optional error details. Content depends on the function used and the error.
animationName	string [▶ 2108]	The name of the animation.
elapsedTime	number [▶ 2108]	The elapsed time since the start of the animation.
JQueryEvent [Optional]	JQuery.Event	If the animation runs over CSS, the event object normalized by jQuery.

Note: Available since version 1.8

15.1.1.5.37 Status

[Enumeration]

```
export enum Status {
    CONFIGURE,
    INITIALIZED,
    RUNNING,
    PAUSED,
    ENDED
}
```

Defines the states an animation can assume. It is not allowed to configure an animation that does not have the state `CONFIGURE` or `ENDED`.

The names of the enumeration exist at runtime. For example, you can use `TcHmi.Animation.Status.CONFIGURE` directly in a comparison.

Values

Name	Value	Description
CONFIGURE	0	The initial status of an animation. This status means that the animation has never been played or reset after it has been played.
INITIALIZED	1	An animation has this status when run() was called, but no effects actually work yet because the CSS has not yet been parsed or a delay is set.
RUNNING	2	A currently running animation.
PAUSED	3	The animation was paused.
ENDED	4	The animation has finished or was skipped with skip().

Note: Available since version 1.8

15.1.1.6 Base64BinaryReader

[Class]

Allows reading binary data from a Base64-encoded string.

Reading always starts at the current position of the read pointer. After a new Base64BinaryReader has been created, the read pointer points to the first byte of the data and is incremented by the number of bytes read with each read operation.

Constructors

Name	Description

constructor [▶ 1861]	Creates a new Base64BinaryReader for reading from the passed string.

Functions

Name	Description

readBoolean [▶ 1861]	Reads a Boolean value.
readByte [▶ 1862]	Reads one byte.
readSByte [▶ 1862]	Reads a signed byte.
readInt16 [▶ 1863]	Reads a 16-bit integer.
readUInt16 [▶ 1863]	Reads an unsigned 16-bit integer.
readInt32 [▶ 1863]	Reads a 32-bit integer.
readUInt32 [▶ 1864]	Reads an unsigned 32-bit integer
readFloat [▶ 1864]	Reads a floating-point number.
readDouble [▶ 1865]	Reads a double-precision floating-point number.
readString [▶ 1865]	Reads a string.
getLength [▶ 1866]	Returns the length of the data in bytes.
getOffset [▶ 1866]	Returns the current position of the read pointer.
setOffset [▶ 1866]	Sets the position of the read pointer.

Note: Available since version 1.10

15.1.1.6.1 constructor

[Function]

```
public constructor(data: string, endianness = TcHmi.Endianness.LittleEndian);
```

Creates a new Base64BinaryReader.

Parameter

Name	Type	Description
data	string [▶ 2108]	A string in Base64 format containing the data to be read.
endianness [optional]	Endianness [▶ 2086]	The byte order in which data was stored in the Base64 string. If this parameter is not specified, the default value LittleEndian is used.

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('khAAAA==');
var readerBigEndian = new TcHmi.Base64BinaryReader('AAAQkg==', TcHmi.Endianness.BigEndian);
```

Note: Available since version 1.10

15.1.1.6.2 readBoolean

[Function]

```
public readBoolean(): boolean;
```

Reads a byte and interprets it as a Boolean value. A byte is considered `false` if it has the value 0 and `true` in all other cases. The read pointer is advanced by one byte.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
boolean [▶ 2107]	The read Boolean value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('AQA=');
var result1 = reader.readBoolean(); // true
var result2 = reader.readBoolean(); // false
```

Note: Available since version 1.10

15.1.1.6.3 readByte

[Function]

```
public readByte(): number;
```

Reads a byte and returns it unchanged. A byte is an unsigned integer with a value range between 0 and 255. The read pointer is advanced by one byte.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The value of the read byte

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('yA==');
var result = reader.readByte(); // 200
```

Note: Available since version 1.10

15.1.1.6.4 readSByte

[Function]

```
public readSByte(): number;
```

Reads a byte and interprets this as a signed integer with a value range between -128 and 127. The read pointer is advanced by one byte.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read SByte value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('f4A=');
var result1 = reader.readSByte(); // 127
var result2 = reader.readSByte(); // -128
```

Note: Available since version 1.10

15.1.1.6.5 readInt16

[Function]

```
public readInt16(): number;
```

Reads two bytes and interprets them as a signed integer with a value range between -32,768 and 32,767. The read pointer is advanced by two bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read Int16 value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('/38AgA==');
var result1 = reader.readInt16(); // 32767
var result2 = reader.readInt16(); // -32768
```

Note: Available since version 1.10

15.1.1.6.6 readUInt16

[Function]

```
public readUInt16(): number;
```

Reads two bytes and interprets them as an unsigned integer with a value range between 0 and 65,535. The read pointer is advanced by two bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read UInt16 value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('MdQ=');
var result1 = reader.readUInt16(); // 54321
```

Note: Available since version 1.10

15.1.1.6.7 readInt32

[Function]

```
public readInt32(): number;
```

Reads four bytes and interprets them as a signed integer with a value range between -2,147,483,648 and 2,147,483,647. The read pointer is advanced by four bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read Int32 value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('////fwAAAIA=');
var result1 = reader.readInt32(); // 2147483647
var result2 = reader.readInt32(); // -2147483648
```

Note: Available since version 1.10

15.1.1.6.8 readUInt32

[Function]

```
public readUInt32(): number;
```

Reads four bytes and interprets them as an unsigned integer with a value range between 0 and 4,294,967,295. The read pointer is advanced by four bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read UInt32 value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('0gKWSQ==');
var result1 = reader.readUInt32(); // 1234567890
```

Note: Available since version 1.10

15.1.1.6.9 readFloat

[Function]

```
public readFloat(): number;
```

Reads four bytes and interprets them as floating point number with an accuracy of seven significant digits. The read pointer is advanced by four bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read float value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('2g9JQA==');
var result1 = reader.readFloat(); // 3.141592502593994
```

Note: Available since version 1.10

15.1.1.6.10 readDouble

[Function]

```
public readDouble(): number;
```

Reads eight bytes and interprets them as floating point number with an accuracy of 16 significant digits. The read pointer is advanced by eight bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The read double value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('GC1EVPshCUA=');
var result1 = reader.readDouble(); // 3.141592653589793
```

Note: Available since version 1.10

15.1.1.6.11 readString

[Function]

```
public readString(length?: number): string;
```

Reads bytes and interprets them as UTF-8 encoded Unicode code points. Bytes are read until a byte with the value null is read, the number of bytes read reaches the value of the parameter `length` or the end of the data is reached. If the optional parameter `length` has been set, the read pointer is advanced by the value of `length`, otherwise the read pointer is advanced by the actual number of bytes read, either after the position of the null byte or at the end of the data.

When reading strings from the PLC, please note that TwinCAT reserves one byte more than the specified string length, in order to have space for a final null byte. If, for example, a string is to be read that was declared in the PLC with length n, it must be read with the length n+1.

Parameter

Name	Type	Description
length [optional]	number [▶ 2108]	The length of the string to be read. If a byte with the value null is read before the specified length is reached, the read process is terminated, but the read pointer is still advanced by the specified length.

Return value

Type	Description
string [▶ 2108]	The read string

Example - JavaScript

```
var reader = new
TcHmi.Base64BinaryReader('SGVsbG8sIFdvcmxkIQAAAAAAAADkuK3mlocgZXNwYcOxb2wg4KS54KS/4KSo4KWN4KSm4KWAIN
in2YTYudix2KjZitipINGA0YPRgdGB0LrQuNC5IOaXpeacrOiqng==');
var result1 = reader.readString(20); // 'Hello, World!'
var result2 = reader.readString(); // '□□ español □□□□□□ العربية، русский □□□'
```

Note: Available since version 1.10

15.1.1.6.12 getLength

[Function]

```
public getLength(): number;
```

Returns the length of the data in bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The total number of bytes contained in the data

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('SGVsbG8sIFdvcmxkIQ==');
var result1 = reader.getLength(); // 13
```

Note: Available since version 1.10

15.1.1.6.13 getOffset

[Function]

```
public getOffset(): number;
```

Returns the current position of the read pointer.

Parameters

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The current position of the read pointer

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('KgA=');
var result1 = reader.getOffset(); // 0
var result2 = reader.readInt16();
var result3 = reader.getOffset(); // 2
```

Note: Available since version 1.10

15.1.1.6.14 setOffset

[Function]

```
public setOffset(offset: number): void;
```

Sets the position of the read pointer.

Parameter

Name	Type	Description
offset	number [▶ 2108]	The new position of the read pointer

Return value

Type	Description
void	No return value

Example - JavaScript

```
var reader = new TcHmi.Base64BinaryReader('SGVsbG8sIFdvcmxkIQ==');
reader.setOffset(7);
var result1 = reader.readString(); // 'World!'
```

Note: Available since version 1.10

15.1.1.7 Base64BinaryWriter

[Class]

Allows writing of binary data to a Base64-encoded string.

Constructors

Name	Description
constructor [▶ 1867]	Creates a new Base64BinaryWriter.

Functions

Name	Description
writeBoolean [▶ 1868]	Writes a Boolean value.
writeByte [▶ 1868]	Writes a byte.
writeSByte [▶ 1869]	Writes a signed byte.
writeInt16 [▶ 1869]	Writes a 16-bit integer.
writeUInt16 [▶ 1870]	Writes an unsigned 16-bit integer.
writeInt32 [▶ 1870]	Writes a 32-bit integer.
writeUInt32 [▶ 1871]	Writes an unsigned 32-bit integer.
writeFloat [▶ 1871]	Writes a floating-point number.
writeDouble [▶ 1872]	Writes a double-precision floating-point number.
writeString [▶ 1872]	Writes a string.
getEncodedString [▶ 1873]	Returns the Base64-encoded string containing the data written so far.
getLength [▶ 1873]	Returns the length of the data written so far in bytes.

Note: Available from version 1.10

15.1.1.7.1 constructor

[Function]

```
public constructor(endianness = TcHmi.Endianness.LittleEndian, length?: number)
```

Creates a new Base64BinaryReader.

Parameter

Name	Type	Description
endianness [optional]	Endianness [▶ 2086]	The byte order in which data is to be stored in the Base64 string. If this parameter is not specified, the default value LittleEndian is used.
length [Optional]	number [▶ 2108]	The length of the data in bytes to be written. If this parameter is specified, errors are triggered when an attempt is made to write data that exceeds the specified length. If getEncodedString is called before the specified length is reached, the remaining bytes are filled with 0.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
var writerBigEndian = new TcHmi.Base64BinaryWriter(TcHmi.Endianness.BigEndian);
var writerFixedLength = new TcHmi.Base64BinaryWriter(TcHmi.Endianness.LittleEndian, 20);
```

Note: Available since version 1.10

15.1.1.7.2 writeBoolean

[Function]

```
public writeBoolean(value: boolean): this;
```

Encodes a Boolean value in a byte and appends it to the data. The value `true` is coded as a byte with the value 1, `false` with 0.

Parameter

Name	Type	Description
value	boolean [▶ 2107]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
writer.writeBoolean(true).writeBoolean(false);
var result1 = writer.getEncodedString(); // 'AQA='
```

Note: Available since version 1.10

15.1.1.7.3 writeByte

[Function]

```
public writeByte(value: number): this;
```

Encodes an unsigned integer with a value range from 0 to 255 in a byte and appends it to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
var result1 = writer.writeByte(200).getEncodedString(); // 'yA=='
```

Note: Available since version 1.10

15.1.1.7.4 writeSByte

[Function]

```
public writeSByte(value: number): this;
```

Encodes a signed integer with a value range of -128 to 127 in a byte and appends it to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
writer.writeSByte(127).writeSByte(-128);
var result1 = writer.getEncodedString(); // 'f4A='
```

Note: Available since version 1.10

15.1.1.7.5 writeInt16

[Function]

```
public writeInt16(value: number): this;
```

Encodes a signed integer with a value range of -32,768 to 32,767 in two bytes and appends them to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
writer.writeInt16(32767).writeInt16(-32768);
var result1 = writer.getEncodedString(); // '/38AgA=='
```

Note: Available since version 1.10

15.1.1.7.6 writeUInt16

[Function]

```
public writeUInt16(value: number): this;
```

Encodes an unsigned integer with a value range from 0 to 65,535 in two bytes and appends them to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
var result1 = writer.writeUInt16(54321).getEncodedString(); // 'MdQ='
```

Note: Available since version 1.10

15.1.1.7.7 writeInt32

[Function]

```
public writeInt32(value: number): this;
```

Encodes a signed integer with a value range of -2,147,483,648 to 2,147,483,647 in four bytes and appends them to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
writer.writeInt32(2147483647).writeInt32(-2147483648);
var result1 = writer.getEncodedString(); // '////fwAAAIA='
```

Note: Available since version 1.10

15.1.1.7.8 writeUInt32

[Function]

```
public writeUInt32(value: number): this;
```

Encodes an unsigned integer with a value range from 0 to 4,294,967,295 in four bytes and appends them to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
var result1 = writer.writeUInt32(1234567890).getEncodedString(); // '0gKWSQ=='
```

Note: Available but version 1.10

15.1.1.7.9 writeFloat

[Function]

```
public writeFloat(value: number): this;
```

Encodes a floating point number with an accuracy of seven significant digits in four bytes and appends them to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
var result1 = writer.writeFloat(3.1415926).getEncodedString(); // '2g9JQA=='
```

Note: Available since version 1.10

15.1.1.7.10 writeDouble

[Function]

```
public writeDouble(value: number): this;
```

Encodes a floating point number with an accuracy of 16 significant digits in eight bytes and appends them to the data.

Parameter

Name	Type	Description
value	number [▶ 2108]	The value to be written

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
var result1 = writer.writeDouble(3.141592653589793).getEncodedString(); // 'GC1EVPshCUA='
```

Note: Available since version 1.10

15.1.1.7.11 writeString

[Function]

```
public writeString(value: string, length?: number): this;
```

Encodes a string as UTF-8 encoded Unicode code points. The optional `length` parameter determines how many bytes are to be used. If the passed string does not fit into `length`, an error is triggered. If the passed string is shorter, the remaining bytes are filled with 0. If `length` is not specified, the number of bytes appended to the data depends on the length of the string.

Parameter

Name	Type	Description
value	string [▶ 2108]	The string to be written
length [optional]	number [▶ 2108]	The number of bytes to be written. If the string is too short, the remaining bytes are filled with 0. If the string is too long, an error is triggered.

Return value

Type	Description
Base64BinaryWriter [▶ 1867]	This method returns its parent object to allow concatenation of method calls.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
writer.writeString('Hello, World!', 20);
writer.writeString('□□ español □□□□□□ العربية русский □□□');
var result1 = writer.getEncodedString(); // '
SGVsbG8sIFdvcmxkIQAAAAAAAADkuK3mlocgZXNwYcOxb2wg4KS54KS/4KSo4KWN4KSm4KWAINin2YTYudix2KjZitipINGA0YPR
gdGB0LrQuNC5IOaXpeacrOiqng=='
```

Note: Available since version 1.10

15.1.1.7.12 getEncodedString

[Function]

```
public getEncodedString(): string;
```

Returns the Base64-encoded string containing the data written so far.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The Base64-encoded string containing the data written so far.

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter();
writer.writeString('Hello, World!');
var result1 = writer.getEncodedString(); // 'SGVsbG8sIFdvcmxkIQ=='
```

Note: Available since version 1.10

15.1.1.7.13 getLength

[Function]

```
public getLength(): number;
```

Returns the length of the data written so far in bytes.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108]	The number of bytes written so far

Example - JavaScript

```
var writer = new TcHmi.Base64BinaryWriter(TcHmi.Endianness.LittleEndian, 20);
writer.writeString('Hello, World!');
var result1 = writer.getLength(); // 13
```

Note: Available since version 1.10

15.1.1.8 Binding

[Class]

Provides functions for creating and removing bindings.

Static functions

Name	Description
create [▶ 1874]	Creates a binding between a symbol and a control attribute.
createEx [▶ 1875]	Creates a binding between a symbol and a control attribute.
createEx2 [▶ 1875]	Creates a binding between a symbol and a control attribute.
remove [▶ 1876]	Removes a binding between a symbol and a control attribute.
removeEx [▶ 1876]	Removes a binding between a symbol and a control attribute.
removeEx2 [▶ 1877]	Removes a binding between a symbol and a control attribute.
exists [▶ 1877]	Determines whether a binding exists between a symbol and a control attribute.
resolve [▶ 1878]	Returns a symbol expression if a binding exists between a symbol and the control attribute or null if no binding exists.
resolveEx [▶ 1879]	Returns a symbol expression if a binding exists between a symbol and the control attribute or null if no binding exists.

Note: Available since version 1.8

15.1.1.8.1 create

[Function]

```
public static create(
   expr: string,
   fn: (value: any) => void,
   fnThis: TcHmi.Controls.System.baseTcHmiControl
): void;
```

Creates a binding between a symbol and a control attribute.

Parameter

Name	Type	Description
expr	string [▶ 2108]	Symbol expression
fn	(value: any [▶ 2107]) => void	Reference to the control setter function
fnThis	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    TcHmi.Binding.create('%i%MyInternalSymbol_1%/i%',myControl.setText, myControl);
}
```

15.1.1.8.2 createEx

[Function]

```
public static createEx(
   expr: string,
   fn: string,
   fnThis: TcHmi.Controls.System.baseTcHmiControl
): void;
```

Creates a binding between a symbol and a control attribute using the attribute setter function.

Parameter

Name	Type	Description
expr	string [▶ 2108]	Symbol expression
fn	string [▶ 2108]	Name of the control setter function
fnThis	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    TcHmi.Binding.createEx('%i%MyInternalSymbol_1%/i%', 'setText', myControl);
}
```

15.1.1.8.3 createEx2

[Function]

```
public static createEx2(
   expr: string,
   propertyName: string,
   control: TcHmi.Controls.System.baseTcHmiControl
): void;
```

Creates a binding between a symbol and a control attribute using the attribute name.

Parameter

Name	Type	Description
expr	string [▶ 2108]	Symbol expression
propertyName	string [▶ 2108]	Name of the control property.
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    TcHmi.Binding.createEx2('%i%MyInternalSymbol_1%/i%', 'Text', myControl);
}
```

15.1.1.8.4 remove

[Function]

```
public static remove (
   expr: string,
   fn: (value: any) => void,
   fnThis: TcHmi.Controls.System.baseTcHmiControl
): void;
```

Removes a binding between a symbol and a control attribute.

Parameter

Name	Type	Description
expr	null [▶ 2108], string [▶ 2108]	Symbol expression
fn	(value: any [▶ 2107]) => void	Reference to the control setter function
fnThis	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    TcHmi.Binding.remove('%i%MyInternalSymbol_1%/i%', myControl.setText, myControl);
}
```

15.1.1.8.5 removeEx

[Function]

```
public static removeEx(
   expr: string,
   fn: string,
   fnThis: TcHmi.Controls.System.baseTcHmiControl
): void;
```

Removes a binding between a symbol and a control attribute.

Parameter

Name	Type	Description
expr	null [▶ 2108], string [▶ 2108]	Symbol expression
fn	string [▶ 2108]	Name of the control setter function
fnThis	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    TcHmi.Binding.removeEx('%i%MyInternalSymbol_1%/i%', 'setText', myControl);
}
```

15.1.1.8.6 removeEx2

[Function]

```
public static removeEx2(
   expr: string | null, // obsolete
   propertyName: string,
   control: TcHmi.Controls.System.baseTcHmiControl
): void;
```

Removes a binding between a symbol and a control attribute.

Parameter

Name	Type	Description
expr	null [▶ 2108], string [▶ 2108]	This parameter is no longer required
propertyName	string [▶ 2108]	Name of the control property.
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Sample - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextBox_1');
if(myControl){
    TcHmi.Binding.removeEx2(null, 'Text', myControl);
}
```

15.1.1.8.7 exists

[Function]

```
public static exists(
    propertyName: string,
    control: TcHmi.Controls.System.baseTcHmiControl
): boolean
```

Determines whether a binding exists between a symbol and a control attribute.

Parameter

Name	Type	Description
propertyName	string [▶ 2108]	Name of the attribute
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
boolean [▶ 2107]	Returns true if the attribute is linked to a symbol.

Note: Available since version 1.10

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    var result = TcHmi.Binding.exists('Text',myControl);
    if(result){
         // Binding exists
    } else {
         // Binding exists not
    }
}
```

15.1.1.8.8 resolve

[Function]

```
public static resolve(
    propertyName: string,
    control: TcHmi.Controls.System.baseTcHmiControl
): string | null
```

Returns a symbol expression if a binding exists between a symbol and the control attribute or null if no binding exists.

Parameter

Name	Type	Description
propertyName	string [▶ 2108]	Name of the attribute
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The symbol expression as string or null.

Note: Available since version 1.10

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    var symbolExpression = TcHmi.Binding.resolve('Text',myControl);
    if(symbolExpression){
         // Binding exists
         console.log(myControl.getId() + '::Text is bound to symbol expression: ' +
symbolExpression);
    } else {
         // Binding exists not
    }
}
```

15.1.1.8.9 resolveEx

[Function]

```
public static resolveEx(
    propertyName: string,
    control: TcHmi.Controls.System.baseTcHmiControl
): TcHmi.SymbolExpression | null
```

Returns a symbol expression if a binding exists between a symbol and the control attribute or null if no binding exists.

Parameter

Name	Type	Description
propertyName	string [▶ 2108]	Name of the attribute
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Reference to the control instance

Return value

Type	Description
SymbolExpression [▶ 2051], null [▶ 2108]	The symbol expression as SymbolExpression or null.

Note: Available since version 1.10

Example - JavaScript

```
var myControl = TcHmi.Controls.get('TcHmiTextbox_1');
if(myControl){
    var symbolExpression = TcHmi.Binding.resolveEx('Text',myControl);
    if(symbolExpression){
        // Binding exists
        console.log(myControl.getId() + '::Text is bound to symbol expression: ' +
symbolExpression.toString());
    } else {
        // Binding exists not
    }
}
```

15.1.1.9 ControlFactory

[Class]

Class for creating control instances.

Static functions

Name	Description
createEx [▶ 1879]	Creates a control instance with name, type and additional attributes.
create [▶ 1881]	Creates a control instance with name and type
create [▶ 1883]	Creates a control instance using a JQuery object
create [▶ 1882]	Creates a control instance using HTML source text

Note: Available since version 1.8

15.1.1.9.1 createEx

[Function]

```
public static createEx(
    type: string,
    name: string,
    attributes: null | Dictionary<any>
    parent?: TcHmi.Controls.System.baseTcHmiControl
): TcHmi.Controls.System.baseTcHmiControl | undefined;
```

Creates a control instance with name, type and additional attributes.

Parameter

Name	Type	Description
type	string [▶ 2108]	Type of the new control.
name	string [▶ 2108]	Name of the new control.

attributes	null [▶ 2108], Dictionary<any [▶ 2107]>	A Dictionary (object) consisting of attribute names and values.
		Sample:
		{
		'data-tchmi-attribute1' : true,
		'data-tchmi-attribute2' : false
		}
		You can read the attribute names from the HTML editor [▶ 48]. To do this, configure any control and open the HTML editor.

parent [optional]	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Parent control of the new control. This is used to check access rights if they depend on the parental control, for example. This parameter should not be set if a control is created dynamically at application level (CodeBehind) and then passed to a ContainerControl (View, Content, Container etc.). The parent/child relationship is created dynamically when the control is added to the corresponding ContainerControl using the addChild [▶ 1182] function. If a control is created dynamically within another control, this other control is the logical parent control and must be passed when the new control is created, so that properties such as access rights can be applied to the child control, and the system has the option to destroy the child control if the parent control is destroyed.

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118], undefined [▶ 2108]	The control object or undefined in the event of an error.

Note: Available since version 1.10

Sample - JavaScript

```
TcHmi.EventProvider.register('Desktop.onAttached', function (e, data) {
    var myButton = TcHmi.ControlFactory.createEx(
        'tchmi-button',
        'MyButton',
        {
            'data-tchmi-top': 25,
            'data-tchmi-left': 250,
            'data-tchmi-width': 200,
            'data-tchmi-height': 50,
            'data-tchmi-text': 'MyButton',
            'data-tchmi-background-color': {
                'color': 'rgba(55, 55, 55, 1)'
            }
        }
    );
    var desktop = TcHmi.Controls.get('Desktop');
    if (desktop && myButton) {
        desktop.addChild(myButton);
    }
});
```

15.1.1.9.2 create

[Function] [Deprecated]

```
public static create(
    type: string,
    name: string,
    parent?: TcHmi.Controls.System.baseTcHmiControl
): TcHmi.Controls.System.baseTcHmiControl | undefined;
```

Function for programmatically creating control instances, based on the type name.

Note: From version 1.10 this function is considered obsolete. Instead, the function createEx [▶ 1879] should be used.

Parameter

Name	Type	Description
type	string [▶ 2108]	Type of the new control.
name	string [▶ 2108]	Name of the new control.
parent [optional]	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Parent control of the new control. This is used to check access rights if they depend on the parental control, for example. This parameter should not be set if a control is created dynamically at application level (CodeBehind) and then passed to a ContainerControl (View, Content, Container etc.). The parent/child relationship is created dynamically when the control is added to the corresponding ContainerControl using the addChild [▶ 1182] function. If a control is created dynamically within another control, this other control is the logical parent control and must be passed when the new control is created, so that properties such as access rights can be applied to the child control, and the system has the option to destroy the child control if the parent control is destroyed.

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118], undefined [▶ 2108]	The control object or undefined in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
// Create control within another control
var childControl = TcHmi.ControlFactory.create('tchmi-button', 'newbutton', myControl);
if(childControl!== undefined) {
    // append control to ourself
    myControl.getElement().append(childControl.getElement());
}
```

15.1.1.9.3 create

[Function] [Deprecated]

```
public static create(
    element: string,
    unused?: null,
    parent?: TcHmi.Controls.System.baseTcHmiControl
): TcHmi.Controls.System.baseTcHmiControl | undefined;
```

Function for programmatically creating control instances based on an HTML string.

Note: From version 1.10 this function is considered obsolete. Instead, the function createEx [▶ 1879] should be used.

Parameter

Name	Type	Description
element	string [▶ 2108]	New control as HTML source text
unused [optional]	null [▶ 2108]	Must be set if parent is to be used.
parent [optional]	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Parent control of the new control. This is used to check access rights if they depend on the parental control, for example. This parameter should not be set if a control is created dynamically at application level (CodeBehind) and then passed to a ContainerControl (View, Content, Container etc.). The parent/child relationship is created dynamically when the control is added to the corresponding ContainerControl using the addChild [▶ 1182] function. If a control is created dynamically within another control, this other control is the logical parent control and must be passed when the new control is created, so that properties such as access rights can be applied to the child control, and the system has the option to destroy the child control if the parent control is destroyed.

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118], undefined [▶ 2108]	The control object or undefined in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
// Create control within another control
var childControl = TcHmi.ControlFactory.create(
    '<div id="newbutton" data-tchmi-type="tchmi-button"></div>',
    null,
    myControl
);
if(childControl!== undefined) {
    // append control to ourself
    myControl.getElement().append(childControl.getElement());
}
```

15.1.1.9.4 create

[Function] [Deprecated]

```
public static create(
    element: JQuery,
    unused?: null,
    parent?: TcHmi.Controls.System.baseTcHmiControl
): TcHmi.Controls.System.baseTcHmiControl | undefined;
```

Function for programmatically creating control instances based on a JQuery element object.

Note: From version 1.10 this function is considered obsolete. Instead, the function createEx [▶ 1879] should be used.

Parameter

Name	Type	Description
element	JQuery [▶ 2108]	New control as a JQuery object.
unused [optional]	null [▶ 2108]	Must be set if parent is to be used.
parent [optional]	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Parent control of the new control. This is used to check access rights if they depend on the parental control, for example. This parameter should not be set if a control is created dynamically at application level (CodeBehind) and then passed to a ContainerControl (View, Content, Container etc.). The parent/child relationship is created dynamically when the control is added to the corresponding ContainerControl using the addChild [▶ 1182] function. If a control is created dynamically within another control, this other control is the logical parent control and must be passed when the new control is created, so that properties such as access rights can be applied to the child control, and the system has the option to destroy the child control if the parent control is destroyed.

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118], undefined [▶ 2108]	The control object or undefined in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
// Create control within another control
var childControl = TcHmi.ControlFactory.create(
    $('<div id="newbutton" data-tchmi-type="tchmi-button"></div>'),
    null,
    myControl
);
if(childControl!== undefined) {
```

```
    // append control to ourself
    myControl.getElement().append(childControl.getElement());
}
```

15.1.1.10 Environment

[Namespace]

Provides environment information.

Static functions

Name	Description
getBasePath [▶ 1885]	Returns the basic path for this environment.

Note: Available since version 1.8

15.1.1.10.1 getBasePath

[Function]

```
public static getBasePath(): string;
```

Returns the basic path for this environment. This is required, for example, if you want to access the directory of a control.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	Basic path of this environment. The basic path is the path to the TwinCAT HMI framework directory, starting from the application directory.

Note: Available since version 1.8

Example - JavaScript

```
var fullImagePath = TcHmi.Environment.getBasePath() + "/Controls/Custom/myControl/images/myImg.png";
```

15.1.1.11 EventProvider

[Class]

Provides functions for managing events.

Static functions

Name	Description
register [▶ 1887]	Registers a callback to an event
raise [▶ 1886]	Notifies the system about an event
raiseEx [▶ 1886]	Notifies the system about an event
watchEventRegistration [▶ 1888]	Monitors an event name for registration events

Note: Available since version 1.8

15.1.1.11.1 raise

[Function]

```
static raise(
    name: string,
    ...args: any[]
): void;
```

Triggers an event, based on the event name for all event registrations relating to this event.

Parameter

Name	Type	Description
name	string [▶ 2108]	Name of the event
args [residual parameters]	any [▶ 2107]	A user-defined number of parameters that are passed through to the event callbacks in addition to the first parameter.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
// Register event
var destroyEvent = TcHmi.EventProvider.register('MyControl.onMyEvent',
    function(evt, data) {
        // Do stuff
        TcHmi.Log.debug('event-data: ' + data);
        // …
        // Destroy to free event resources if event is no longer needed.
        destroyEvent();
    }
);

// Raise event
TcHmi.EventProvider.raise('MyControl.onMyEvent', 42);
```

15.1.1.11.2 raiseEx

[Function]

```
static raiseEx(
    event: EventProvider.IEventEntry,
    ...args: any[]
): void;
```

Triggers an event based on a specific event registration.

See also: EventProvider.watchEventRegistration [▶ 1888]

Parameter

Name	Type	Description
event	EventProvider.IEventEntry [▶ 1890]	An event entry for a specific event registration.
args [residual parameters]	any [▶ 2107]	A user-defined number of parameters that are passed through to the event callbacks in addition to the first parameter.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

See the example for the function EventProvider.watchEventRegistration [▶ 1888]

15.1.1.11.3 register

[Function]

```
static register(
    name: string,
    callback: TcHmi.EventProvider.Callback
    options?: object
): TcHmi.DestroyFunction;
```

Registers a callback to an event and returns a function for deregistering the event.

If the event is no longer required, this DestroyCallback can be used to deregister the event.

Parameter

Name	Type	Description
name	string [▶ 2108]	Name of the event
callback	(e: EventProvider.Event [▶ 1889], ...data: any [▶ 2107][]) => void;	Callback function which should be called for these events. This function receives an event object and optional further parameters. Not all events supplies these.
options [optional] **Note:** Available since version 1.10	object	JSON object with setting values. Options depend on the event.

Return value

Type	Description
TcHmi.DestroyFunction [▶ 2101]	Deregisters the event and releases all associated resources when called.

Note: Available since version 1.8

Example - JavaScript

```
var destroyEvent = TcHmi.EventProvider.register(
    'TcHmiButton.onPressed',
    function (evt, data) {
        // Do stuff
        console.log(evt);
        if(data){
            // some events have custom data
            console.log(data);
        }
        // …
        // Destroy to free event resources if event is no longer needed.
        destroyEvent();
    }
);
```

15.1.1.11.4 watchEventRegistration

[Function]

```
public static watchEventRegistration(
    name: string,
    callback: (data: EventProvider.IEventRegResultObject) => void
): DestroyFunction
```

Monitors an event name for registration events and triggers the associated callback when a registration takes place, e.g. in order to be able to respond to event-specific registration options (see register [▶ 1887]).

Parameter

Name	Type	Description
name	string [▶ 2108]	Name of the event
callback	data: EventProvider.IEventRegResultObject [▶ 1891]) => void	Callback with information about the event registration event.

Return value

Type	Description
TcHmi.DestroyFunction [▶ 2101]	Deregisters event registration monitoring and releases all associated resources when called.

Note: Available since version 1.10

Example - JavaScript 1

```
TcHmi.EventProvider.watchEventRegistration('MyEvent', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        switch (data.type) {
            case TcHmi.EventProvider.EventRegWatchType.REGISTER:
            {
                // Event registration
                console.log(data);
            }
            break;
            case TcHmi.EventProvider.EventRegWatchType.DESTROY:
            {
                // Event destroy
                console.log(data);
            }
            break;
        }
    }
});

var e1 = TcHmi.EventProvider.register('MyEvent', function () { }, { myOption: true });
var e2 = TcHmi.EventProvider.register('MyEvent', function () { });
e2();
e1();
```

Example - JavaScript 2

Extract from the control class TcHmi.Controls.System.TcHmiControl as implementation example using the event [Id].onMouseDown.

The class TcHmi.Controls.System.TcHmiControl monitors the HMI event [Id].onMouseDown for registration events to allow the user to use different options for each registration.

All HMI events at the level TcHmi.Controls.System.TcHmiControl, which are based on DOM events, enable the DOM event properties passive (Boolean) and capture (Boolean) to be activated or deactivated during registration.

The constant TCHMI_EVENT_OPTION_OBJECT_SUPPORTED determines whether the browser supports a current addEventListener API or whether one is limited to the capture (Boolean) option.

```
//...
function TcHmiControl(element, pcElement, attrs) {
    //...
    _this.__resourcesDomEventRegWatchOnMouseDown = {};
    //...
}

TcHmiControl.prototype.__attach = function () {
    //...
    __this.__destroyDomEventRegWatchOnMouseDown =
TcHmi.EventProvider.watchEventRegistration(this.getId() + '.onMouseDown', function (data) {
        if (data.error === TcHmi.Errors.NONE) {
            switch (data.type) {
                case TcHmi.EventProvider.EventRegWatchType.REGISTER:
                    {
                        __this.__resourcesDomEventRegWatchOnMouseDown[data.event.id] = {
                            event: data.event,
                            domEventListener: {
                            eventName: 'mousedown',
                            listener: __this.__onElementMouseDown.call(__this, data.event),
                            options: TCHMI_EVENT_OPTION_OBJECT_SUPPORTED ? (
                                                                    data.event.options ? data.event.options :
{ passive: false, capture: false }
                                                                    ) : (
                                                                     (data.event.options &&
data.event.options.capture) ? data.event.options.capture : false
                                                                    )
                            }
                        };
                        var list =
__this.__resourcesDomEventRegWatchOnMouseDown[data.event.id].domEventListener;
                        __this.getElement()[0].addEventListener(list.eventName, list.listener,
list.options);
                    }
                    break;
                case TcHmi.EventProvider.EventRegWatchType.DESTROY:
                    {
                        var list =
__this.__resourcesDomEventRegWatchOnMouseDown[data.event.id].domEventListener;
                        __this.getElement()[0].removeEventListener(list.eventName, list.listener,
list.options);
                    }
                    break;
            }
        }
    });
    //...
};

TcHmiControl.prototype.__detach = function () {
    //...
    for (var key in this.__resourcesDomEventRegWatchOnMouseDown) {
        var list = this.__resourcesDomEventRegWatchOnMouseDown[key].domEventListener;
        this.__element[0].removeEventListener(list.eventName, list.listener, list.options);
    }
    //...
    if (this.__destroyDomEventRegWatchOnMouseDown) {
        this.__destroyDomEventRegWatchOnMouseDown();
        this.__destroyDomEventRegWatchOnMouseDown = null;
    }
    //...
};

TcHmiControl.prototype.__onElementMouseDown = function (event) {
    var _this = this;
    return function (e) {
        //...
        TcHmi.EventProvider.raiseEx(event, e);
        //...
    };
};
//...
```

15.1.1.11.5 Event

[Interface]

```
interface Event extends EventContext {
    name: string;
    destroy: TcHmi.DestroyFunction
}
```

This interface extends TcHmi.EventContext [▶ 2101]

Properties

Name	Type	Description
name	string [▶ 2108]	Name of the event
destroy	TcHmi.DestroyFunction [▶ 2101]	Deregisters the event and releases all associated resources when called.

Note: Available since version 1.8

15.1.1.11.6 EventRegWatchType

[Enumeration]

```
export enum EventRegWatchType {
    REGISTER = 100,
    DESTROY = 200,
}
```

Properties

Name	Value	Description
REGISTER	100	Event was registered.
DESTROY	200	Event registration was canceled.

Note: Available since version 1.10

15.1.1.11.7 IEventEntry

[Interface]

```
export interface IEventEntry {
    id: number;
    name: string;
    callback: (e: EventProvider.Event, ...data: any[]) => void;
    options: any;
    destroy: DestroyFunction | null;
}
```

Properties

Name	Type	Description
id	number [▶ 2108]	
name	string [▶ 2108]	
callback	(e: EventProvider.Event [▶ 1889], ...data: any [▶ 2107][]) => void;	
options	any [▶ 2107]	
destroy	DestroyFunction [▶ 2101], null [▶ 2108]	

Note: Available since version 1.10

15.1.1.11.8 IEventRegResultObject

[Interface]

```
export interface IEventRegResultObject extends IResultObject {
    type: EventRegWatchType;
    event: IEventEntry;
}
```

Properties

Name	Type	Description
type	EventRegWatchType [▶ 1890]	Type of event registration event.
event	IEventEntry [▶ 1890]	Event entry relating to the event registration.

Note: Available since version 1.10

15.1.1.12 IFunction

[Namespace]

Note: Available since version 1.8

15.1.1.12.1 Value

[Interface]

```
export type Value = StaticValue | Symbol | EventDataObject | FunctionExpression;
```

This interface can be implemented by either a StaticValue [▶ 1891], a Symbol [▶ 1891], an EventDataObject [▶ 1892] or a FunctionExpression [▶ 1892].

Note: Available since version 1.8

15.1.1.12.2 StaticValue

[Interface]

```
export interface StaticValue {
    objectType: 'StaticValue';
    value: any;
    valueType: string;
}
```

This interface implements the TcHmi.IFunction.Value [▶ 1891] interface.

Properties

Name	Type	Description
objectType	'StaticValue'	Fixed string
value	any [▶ 2107]	Any static value.
valueType	string [▶ 2108]	Schema name of the data type

Note: Available since version 1.8

15.1.1.12.3 Symbol

[Interface]

```
export interface Symbol {
         objectType: 'Symbol';
         symbolExpression: string;
     }
```

This interface implements the TcHmi.IFunction.Value [▶ 1891] interface.

Properties

Name	Type	Description
objectType	'Symbol'	Fixed string
SymbolExpression	string [▶ 2108]	Symbol expression

Note: Available since version 1.8

15.1.1.12.4 EventDataObject

[Interface]

```
export interface EventDataObject {
    objectType: 'EventDataObject';
    propertyPath: string;
}
```

This interface implements the TcHmi.IFunction.Value [▶ 1891] interface.

Properties

Name	Type	Description
objectType	'EventDataObject'	Fixed string
propertyPath	string [▶ 2108]	Path to property

Note: Available since version 1.8

15.1.1.12.5 FunctionExpression

[Interface]

```
export interface FunctionExpression {
    objectType: 'FunctionExpression';
    functionExpression: string;
}
```

This interface implements the TcHmi.IFunction.Value [▶ 1891] interface.

Properties

Name	Type	Description
objectType	'FunctionExpression'	Fixed string
functionExpression	string [▶ 2108]	Function expression

Note: Available since version 1.8

15.1.1.13 Locale

[Class]

Provides resources for managing the localization.

Static functions

Name	Description
load [▶ 1893]	Loads a new localization.
get [▶ 1893]	Returns the name of the current localization.
getRegisteredLocales [▶ 1894]	Returns a list of all registered localizations as an array of strings.

Note: Available since version 1.8

15.1.1.13.1 load

[Function]

```
public static load(
    locale: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null)
): void;
```

Loads a new localization in the client.

Parameter

Name	Type	Description
locale	string [▶ 2108], null [▶ 2108]	The name of the localization to be loaded. Example: 'de-DE', 'en-US'
callback [optional]	TcHmi.IResultObject [▶ 2094]	Asynchronous callback function that is triggered when the language was switched.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

15.1.1.13.2 get

[Function]

```
public static get(): string;
```

Returns the current localization.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108]	The name of the current localization. Example: 'de-DE', 'en-US'

Note: Available since version 1.8

15.1.1.13.3 getRegisteredLocales

[Function]

```
public static getRegisteredLocales (): string[];
```

Returns a list of all registered localizations as an array of strings.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108][]	List of all registered localizations.

Supported from version: 1.8.659.0

15.1.1.14 Localization

[Class]

Provides localization functions.

Static functions

Name	Description

formatDate [▶ 1894]	Formats a Javascript timestamp.

Note: Available since version 1.10.1336.10

15.1.1.14.1 formatDate

[Function]

```
public static formatDate(
    date: Date,
    options?: TcHmi.Localization.FormatOptions
): string | null;
```

Formats a Javascript timestamp in a specific time zone, language, and level of detail as a string. The output also contains milliseconds.

If the user does not specify a time zone or locale, the settings of the current user apply. These are also available under TcHmi.Server.getCurrentUserConfig [▶ 1916].

Parameter

Name	Type	Description
date	Date [▶ 2108]	To the formatting date
options [optional]	TcHmi.Localization.FormatOptions [▶ 1895]	Formatting options

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	Formatted string or null in case of error.

Note: Available since version 1.10.1336.10

Sample - JavaScript

```
var date = new Date('2019-12-02T14:15:16.017Z');
var result = TcHmi.Localization.formatDate(date);
console.log(result); // For a user with timeFormatLocale 'de-DE' and timeZone set to 'Europe/
Berlin': 2.12.2019, 15:15:16,017
```

```
var date = new Date('2019-12-02T14:15:16.017Z');
var result = TcHmi.Localization.formatDate(date, {
    type: 'time',
    timeZone: 'Europe/Helsinki',
    locale: 'fr-FR'
});
console.log(result); // '16:15:16,017'
```

15.1.1.14.2 FormatOptions

[Interface]

```
export interface FormatOptions {
    timeZone?: string | undefined;
    type?: 'full' | 'date' | 'time';
    locale?: string | undefined | null;
}
```

Properties

Name	Type	Description
timeZone [Optional]	string [▶ 2108], undefined [▶ 2108]	If this time zone is not specified or undefined, the setting of the current user is used.
type [Optional]	'full', 'date', 'time'	Specifies whether to display date, time, or both.
locale [Optional]	string [▶ 2108], undefined [▶ 2108], null [▶ 2108]	If this localization is not specified or undefined/null, the timeFormatLocale [▶ 2005] setting of the current user is used.

The settings of the current user are also available under TcHmi.Server.getCurrentUserConfig [▶ 1916].

Note: Available since version 1.10.1336.10

15.1.1.15 Log

[Namespace]

Functions

Name	Description
error [▶ 1896]	Issues an error message on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 1.
warn [▶ 1897]	Issues a warning on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 2.
info [▶ 1897]	Issues a warning on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 3.
debug [▶ 1898]	Issues a debug message on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 4.

Note: Available since version 1.8

15.1.1.15.1 error

```
[ Funktion ]
```

```
function error(message: string | any): void;
```

Issues an error message on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 1.

Parameter

Name	Type	Description
message	**Note:** Version 1.8 string [▶ 2108] **Note:** Version 1.10 string [▶ 2108], any [▶ 2107] This change is backward compatible.	The message text that is displayed in the browser console. From version 1.10, if a data type other than string is passed, this value is no longer converted into a string and forwarded directly to the browser console. This makes it possible to use browser-specific data type representations. This is especially useful for complex values such as objects or arrays. Most current browsers offer the option to expand object properties.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Log.error(My error text…');
```

15.1.1.15.2 warn

```
[ Funktion ]
```

```
function warn(message: string | any): void;
```

Issues a warning on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 2.

Parameter

Name	Type	Description
message	**Note:** Version 1.8 string [▶ 2108] **Note:** Version 1.10 string [▶ 2108], any [▶ 2107] This change is backward compatible.	The message text that is displayed in the browser console. From version 1.10, if a data type other than string is passed, this value is no longer converted into a string and forwarded directly to the browser console. This makes it possible to use browser-specific data type representations. This is especially useful for complex values such as objects or arrays. Most current browsers offer the option to expand object properties.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Log.warn('My warning text…');
```

15.1.1.15.3 info

```
[ Funktion ]
```

```
function info(message: string | any): void;
```

Issues a message on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 3.

Parameter

Name	Type	Description
message	**Note:** Version 1.8 string [▶ 2108] **Note:** Version 1.10 string [▶ 2108], any [▶ 2107] This change is backward compatible.	The message text that is displayed in the browser console. From version 1.10, if a data type other than string is passed, this value is no longer converted into a string and forwarded directly to the browser console. This makes it possible to use browser-specific data type representations. This is especially useful for complex values such as objects or arrays. Most current browsers offer the option to expand object properties.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Log.info('My info text…');
```

15.1.1.15.4 debug

```
[ Funktion ]
```

```
function debug(message: string | any): void;
```

Issues a debug message on the browser console if the constant TCHMI_CONSOLE_LOG_LEVEL is greater than or equal to 4.

Parameter

Name	Type	Description
message	**Note:** Version 1.8 string [▶ 2108] **Note:** Version 1.10 string [▶ 2108], any [▶ 2107] This change is backward compatible.	The message text that is displayed in the browser console. From version 1.10, if a data type other than string is passed, this value is no longer converted into a string and forwarded directly to the browser console. This makes it possible to use browser-specific data type representations. This is especially useful for complex values such as objects or arrays. Most current browsers offer the option to expand object properties.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Log.debug('My debug text…');
```

15.1.1.15.5 buildMessage

```
[ Funktion ]
```

```
export function buildMessage(
    error: TcHmi.IErrorDetails | undefined
): string;
```

Formats a recursive error detail object for output to the console. Please do not rely on the exact output format.

Parameter

Name	Type	Description
error	TcHmi.IErrorDetails [▶ 2094].	Error detail object

Return value

Type	Description
string [▶ 2108]	Formatted output. If the input parameter is not an error object, an empty string is output.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Symbol.readEx2(
    '%l%nonExisting%/l%',
    function(data){
        TcHmi.Log.error('Did not work because of: ' + TcHmi.Log.buildMessage(data.details))})
/* Results in console message with something like that:
-------------------
[Error] Did not work because of: Code: 2010/0x7da, Message: E_SYMBOL_UNKNOWN
  Reason: %l%nonExisting%/l%
  Domain: TcHmi.System.Symbol
-------------------
*/



TcHmi.Symbol.readEx2(
    '%f%%l%nonExisting%/l%.toString()%/f%',
    function(data){
        TcHmi.Log.error('Did not work because of: ' + TcHmi.Log.buildMessage(data.details))
    }
)
/* Results in console message with something like that:
-------------------
[Error] Did not work because of: Code: 1/0x1, Message: ERROR
  Reason: %f%%l%nonExisting%/l%.toString()%/f%, could not read value
  Domain: TcHmi.System.Symbol
  as result of: Code: 2010/0x7da, Message: E_SYMBOL_UNKNOWN
    Reason: %l%nonExisting%/l%
    Domain: TcHmi.System.Symbol
-------------------
*/
```

15.1.1.16 Server

[Class]

Provides functions for communication with the TwinCAT HMI Server.

Static functions

Name	Description
getWebsocketReadyState [▶ 1901]	Returns the current value of the web socket state.
isWebsocketReady [▶ 1901]	Returns the communication state of the web socket.
writeSymbol [▶ 1902]	Writes one or more values to a TwinCAT HMI server symbol.
writeSymbolEx [▶ 1903]	Writes one or more values to a TwinCAT HMI server symbol.
readSymbol [▶ 1905]	Reads one or more values from a TwinCAT HMI server symbol.
readSymbolEx [▶ 1906]	Reads one or more values from a TwinCAT HMI server symbol.
request [▶ 1907]	Sends a user-defined request to the TwinCAT HMI server.
requestEx [▶ 1908]	Sends a user-defined request to the TwinCAT HMI server.
subscribe [▶ 1910]	Registers a subscription on a list of commands.
subscribeEx [▶ 1911]	Registers a subscription on a list of commands.
unsubscribe [▶ 1912]	Cancels an existing subscription.
unsubscribeEx [▶ 1914]	Cancels an existing subscription.
getCurrentUser [▶ 1915]	Returns the name of the current user as string or null.
getGroupsOfCurrentUser [▶ 1915]	Returns the names of the groups to which the current user is assigned as a string array.
getCurrentUserConfig [▶ 1916]	Returns the current user configuration.
logout [▶ 1916]	Logs out the current user.
logoutEx [▶ 1917]	Logs out the current user.
forceLogout [▶ 1917]	Logs out any user if the appropriate user privileges are available.
forceLogoutEx [▶ 1919]	Logs out any user if the appropriate user privileges are available.

Classes

Name	Description
Events [▶ 1920]	Enables receiving of server events and acknowledging of alarms.
Historize [▶ 1928]	Provides functions to edit the server's Historize configuration.
UserManagement [▶ 1938]	Enables reading and modifying of users.
RecipeManagement [▶ 1956]	Enables reading and modifying of recipes

Enumerations

Name	Description
userConfigState [▶ 2006]	Information about the state of the current user configuration.
ACCESS [▶ 2007]	Access rights within the server.

Interfaces

Name	Description
ICallbackParameter [▶ 2001]	
IMessage [▶ 2004]	
IError [▶ 2003]	
ICommand [▶ 2002]	
ICommandError [▶ 2002]	
userConfigOnServer [▶ 2005]	

Types

Name	Description
IRequestType [▶ 2004]	
ICommandOptions [▶ 2003]	

15.1.1.16.1 getWebsocketReadyState

[Function]

```
public static getWebsocketReadyState(): number | null;
```

Returns the current value of the web socket state.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	State value of the web socket. 0: The connection has not yet been established. 1: The connection is established and ready to communicate through it. 2: The connection is in the process of closing. 3: The connection is closed or could not be established.

Note: Available since version 1.8

Example - JavaScript

```
var readyState = TcHmi.Server.getWebsocketReadyState();
if(readyState === 0){
    // Websocket is ready.
} else {
    // Websocket is not ready.
}
```

15.1.1.16.2 isWebsocketReady

[Function]

```
public static isWebsocketReady(): boolean;
```

Returns the communication state of the web socket.

Parameter

Name	Type	Description
-	-	-w

Return value

Type	Description
boolean [▶ 2107]	State value of the web socket.

Note: Available since version 1.8

Example - JavaScript

```
var ready = TcHmi.Server.isWebsocketReady();
if(ready === true){
    // Websocket is ready.
} else{
    // Websocket is not ready.
}
```

15.1.1.16.3 writeSymbol

[Function]

```
public static writeSymbol(
    symbolName: string | string[],
    value: any | any[],
    callback?: null | (
        (data: TcHmi.Server.IResultObject) => void
    ) = null
): number | null;
```

Writes values to one or more TwinCAT HMI server symbols. If several symbols are to be written, the desired symbol names must be passed as an array and the value parameter as an array must be assigned exactly the same size.

Parameter

Name	Type	Description
symbolName **Note:** Support for string[] from version 1.8.714.0	string [▶ 2108], string [▶ 2108][]	The name of a TwinCAT HMI server symbol. If a string array is passed, several symbols are written simultaneously.
value **Note:** Support for array from version 1.8.714.0	string [▶ 2108], number [▶ 2108], boolean [▶ 2107], Object, Array	The value to be written or an array of values to be written.
callback [Optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
if( TcHmi.Server.isWebsocketReady() ){
    TcHmi.Server.writeSymbol('PLC1.MAIN.bTest', true, function(data){
    if(data.error !== TcHmi.Errors.NONE){
        // Handle TcHmi.Server class level error here.
        return;
    }
    var response = data.response;
    if(response.error !== undefined){
        // Handle TwinCAT HMI Server response level error here.
        return;
    }
    var commands = response.commands;
    if(commands === undefined){
        return;
    }
    var command = response.commands[0];
    if(command === undefined){
        return;
    }
    if(command.error !== undefined){
        // Handle TwinCAT HMI Server command level error here.
        return;
    }
    // Handle result...
    TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);
    });

    // since version 1.8.714.0
    TcHmi.Server.writeSymbol(['PLC1.MAIN.bTest', 'PLC1.MAIN.iTest'], [true, 1], function(data){
    if(data.error !== TcHmi.Errors.NONE){
        // Handle TcHmi.Server class level error here.
        return;
    }
    var response = data.response;
    if(response.error !== undefined){
        // Handle TwinCAT HMI Server response level error here.
        return;
    }
    var commands = response.commands;
    if(commands === undefined){
        return;
    }
    var command = response.commands[0];
    if(command === undefined){
        return;
    }
    if(command.error !== undefined){
        // Handle TwinCAT HMI Server command level error here.
        return;
    }
    // Handle result...
    TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);

    var command1 = response.commands[1];
    if(command1 === undefined){
        return;
    }
    if(command1.error !== undefined){
        // Handle TwinCAT HMI Server command level error here.
        return;
    }
    // Handle result...
    TcHmi.Log.debug('PLC1.MAIN.iTest='+command1.readValue);
    });
}
```

15.1.1.16.4 writeSymbolEx

[Function]

```
public static writeSymbolEx(
    symbolName: string | string[],
    value: any | any[],
    requestOptions: Server.IRequestOptions,
    callback?: null | (
```

```
        (data: TcHmi.Server.IResultObject) => void
    ) = null
): number | null;
```

Writes values to one or more TwinCAT HMI server symbols. If several symbols are to be written, the desired symbol names must be passed as an array and the value parameter as an array must be assigned exactly the same size.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108], string [▶ 2108][]	The name of a TwinCAT HMI server symbol. If a string array is passed, several symbols are written simultaneously.
value	string [▶ 2108], number [▶ 2108], boolean [▶ 2107], Object, Array	The value to be written or an array of values to be written.
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
if( TcHmi.Server.isWebsocketReady() ){
    TcHmi.Server.writeSymbolEx('PLC1.MAIN.bTest', true, { timeout: 2000 }, function(data){
    if(data.error !== TcHmi.Errors.NONE){
        // Handle TcHmi.Server class level error here.
        return;
    }
    var response = data.response;
    if(response.error !== undefined){
        // Handle TwinCAT HMI Server response level error here.
        return;
    }
    var commands = response.commands;
    if(commands === undefined){
        return;
    }
    var command = response.commands[0];
    if(command === undefined){
        return;
    }
    if(command.error !== undefined){
        // Handle TwinCAT HMI Server command level error here.
        return;
    }
    // Handle result...
    TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);
    });

    // since version 1.8.714.0
    TcHmi.Server.writeSymbol(['PLC1.MAIN.bTest', 'PLC1.MAIN.iTest'], [true, 1], function(data){
    if(data.error !== TcHmi.Errors.NONE){
        // Handle TcHmi.Server class level error here.
        return;
    }
    var response = data.response;
    if(response.error !== undefined){
        // Handle TwinCAT HMI Server response level error here.
        return;
    }
```

```
    var commands = response.commands;
    if(commands === undefined){
        return;
    }
    var command = response.commands[0];
    if(command === undefined){
        return;
    }
    if(command.error !== undefined){
        // Handle TwinCAT HMI Server command level error here.
        return;
    }
    // Handle result...
    TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);

    var command1 = response.commands[1];
    if(command1 === undefined){
        return;
    }
    if(command1.error !== undefined){
        // Handle TwinCAT HMI Server command level error here.
        return;
    }
    // Handle result...
    TcHmi.Log.debug('PLC1.MAIN.iTest='+command1.readValue);
    });
}
```

15.1.1.16.5 readSymbol

[Function]

```
public static readSymbol(
    symbolName: string | string[],
    callback?: null | (
         (data: TcHmi.Server.IResultObject) => void
    )
): number | null;
```

Reads one or more values from a TwinCAT HMI server symbol.

Parameter

Name	Type	Description
symbolName	string [▶ 2108], string [▶ 2108][]	The name of a TwinCAT HMI server symbol. If a string array is passed, several symbols are queried simultaneously.
callback [Optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
if( TcHmi.Server.isWebsocketReady() ){
    TcHmi.Server.readSymbol('PLC1.MAIN.bTest', function(data){
        if(data.error !== TcHmi.Errors.NONE){
            // Handle TcHmi.Server class level error here.
            return;
        }
        var response = data.response;
        if(response.error !== undefined){
            // Handle TwinCAT HMI Server response level error here.
```

```
            return;
        }
        var commands = response.commands;
        if(commands === undefined){
            return;
        }
        var command = response.commands[0];
        if(command === undefined){
            return;
        }
        if(command.error !== undefined){
            // Handle TwinCAT HMI Server command level error here.
            return;
        }
        // Handle result...
        TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);
    });

    TcHmi.Server.readSymbol(['PLC1.MAIN.bTest', 'PLC1.MAIN.iTest'], function(data){
        if(data.error !== TcHmi.Errors.NONE){
            // Handle TcHmi.Server class level error here.
            return;
        }
        var response = data.response;
        if(response.error !== undefined){
            // Handle TwinCAT HMI Server response level error here.
            return;
        }
        var commands = response.commands;
        if(commands === undefined){
            return;
        }
        var command = response.commands[0];
        if(command === undefined){
            return;
        }
        if(command.error !== undefined){
            // Handle TwinCAT HMI Server command level error here.
            return;
        }
        // Handle result...
        TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);

        var command1 = response.commands[1];
        if(command1 === undefined){
            return;
        }
        if(command1.error !== undefined){
            // Handle TwinCAT HMI Server command level error here.
            return;
        }
        // Handle result...
        TcHmi.Log.debug('PLC1.MAIN.iTest='+command1.readValue);
    });
}
```

15.1.1.16.6 readSymbolEx

[Function]

```
public static readSymbolEx(
    symbolName: string | string[],
    requestOptions: Server.IRequestOptions,
    callback?: null | (
         (data: TcHmi.Server.IResultObject) => void
    )
): number | null;
```

Reads one or more values from a TwinCAT HMI server symbol.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108], string [▶ 2108][]	The name of a TwinCAT HMI server symbol. If a string array is passed, several symbols are queried simultaneously.
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
if( TcHmi.Server.isWebsocketReady() ){
    TcHmi.Server.readSymbolEx('PLC1.MAIN.bTest', { timeout: 2000 }, function(data){
        if(data.error !== TcHmi.Errors.NONE){
            // Handle TcHmi.Server class level error here.
            return;
        }
        var response = data.response;
        if(response.error !== undefined){
            // Handle TwinCAT HMI Server response level error here.
            return;
        }
        var commands = response.commands;
        if(commands === undefined){
            return;
        }
        var command = response.commands[0];
        if(command === undefined){
            return;
        }
        if(command.error !== undefined){
            // Handle TwinCAT HMI Server command level error here.
            return;
        }
        // Handle result...
        TcHmi.Log.debug('PLC1.MAIN.bTest='+command.readValue);
    });
}
```

15.1.1.16.7 request

[Function]

```
public static request(
    request: TcHmi.Server.IMessage,
    callback?: null | (
     (data: Server.IResultObject) => void
    )
): number | null;
```

Sends a user-defined request to the TwinCAT HMI server.

Parameter

Name	Type	Description
request	TcHmi.Server.IMessage [▶ 2004]	TwinCAT HMI server web socket request object.
callback [Optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2004]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
if( TcHmi.Server.isWebsocketReady() ){
    // Request object.
    var request = {
        'requestType': 'ReadWrite',
        'commands': [
            {
                'symbol': 'PLC1.MAIN.bTest'
            },
            {
                'symbol': 'PLC1.MAIN.nTest'
            }
        ]
    };

    // Send request to TwinCAT HMI Server.
    TcHmi.Server.request(request, function(data){
        // Callback handling.
        if(data.error !== TcHmi.Errors.NONE){
            // Handle TcHmi.Server class level error here.
            return;
        }
        var response = data.response;
        if(response.error !== undefined){
            // Handle TwinCAT HMI Server response level error here.
            return;
        }
        var commands = response.commands;
        if(commands === undefined){
            return;
        }
        for(var i = 0, ii = commands.length; i < ii; i++){
            var command = response.commands[i];
            if(command === undefined){
                return;
            }
            if(command.error !== undefined){
                // Handle TwinCAT HMI Server command level error here.
                return;
            }
            // Handle result...
            TcHmi.Log.debug(command.symbol+'='+command.readValue);
        }
    });
}
```

15.1.1.16.8 requestEx

[Function]

```
public static requestEx(
    request: TcHmi.Server.IMessage,
    requestOptions: Server.IRequestOptions,
```

```
    callback?: null | (
     (data: Server.IResultObject) => void
    )
): number | null;
```

Sends a user-defined request to the TwinCAT HMI server.

Parameter

Name	Type	Description
request	TcHmi.Server.IMessage [▶ 2004]	TwinCAT HMI server web socket request object.
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2004]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
if( TcHmi.Server.isWebsocketReady() ){
    // Request object.
    var request = {
        'requestType': 'ReadWrite',
        'commands': [
            {
                'symbol': 'PLC1.MAIN.bTest'
            },
            {
                'symbol': 'PLC1.MAIN.nTest'
            }
        ]
    };

    // Send request to TwinCAT HMI Server.
    TcHmi.Server.requestEx(request, { timeout: 2000 }, function(data){
        // Callback handling.
        if(data.error !== TcHmi.Errors.NONE){
            // Handle TcHmi.Server class level error here.
            return;
        }
        var response = data.response;
        if(response.error !== undefined){
            // Handle TwinCAT HMI Server response level error here.
            return;
        }
        var commands = response.commands;
        if(commands === undefined){
            return;
        }
        for(var i = 0, ii = commands.length; i < ii; i++){
            var command = response.commands[i];
            if(command === undefined){
                return;
            }
            if(command.error !== undefined){
                // Handle TwinCAT HMI Server command level error here.
                return;
            }
            // Handle result...
            TcHmi.Log.debug(command.symbol+'='+command.readValue);
        }
    });
}
```

15.1.1.16.9 releaseRequest

[Function]

```
public static releaseRequest(id: number)
```

Releases the resources relating to a request to the TwinCAT HMI server. Releasing request resources manually is only necessary in a few cases.

Cases in which request resources must be released manually:

1. A subscription is canceled using the function request [▶ 1907]/requestEx [▶ 1908] instead of unsubscribe [▶ 1912]/unsubscribeEx [▶ 1914]. In this case, it is necessary to manually approve the original subscription request.

Parameter

Name	Type	Description
id	number [▶ 2108]	Request Id

Return value

Type	Description
void	No return value

Note: Available since version 1.10

15.1.1.16.10 subscribe

[Function]

```
public static subscribe(
    commands: TcHmi.Server.ICommand[],
    interval: number,
    callback?: null | (
         (data: TcHmi.Server.IResultObject) => void
    )
): number | null;
```

Registers a subscription on a list of commands.

Parameter

Name	Type	Description
commands	TcHmi.Server.ICommand [▶ 2002][]	An array of ICommand objects.
interval	number [▶ 2108]	The earliest time after which the TwinCAT HMI server transmits changes.
callback [Optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
if (TcHmi.Server.isWebsocketReady()) {
    var commands = [
        {
            'symbol': 'PLC1.MAIN.bTest'
        },
        {
            'symbol': 'PLC1.MAIN.nTest'
        }];

    TcHmi.Server.subscribe(commands, 500, function (data) {
        if (data.error !== TcHmi.Errors.NONE) {
            // Handle TcHmi.Server class level error here.
            return;
        }

        var response = data.response;
        if (response.error !== undefined) {
            // Handle TwinCAT HMI Server response level error here.
            return;
        }

        var commands = response.commands;
        if (commands === undefined) {
            return;
        }

        for (var i = 0, ii = commands.length; i < ii; i++) {
            var command = response.commands[i];
            if (command === undefined) {
                return;
            }
            if (command.error !== undefined) {
                // Handle TwinCAT HMI Server command level error here.
                return;
            }

            // Handle result...
            TcHmi.Log.debug(command.symbol + '=' + command.readValue);
        }
    });
}
```

15.1.1.16.11 subscribeEx

[Function]

```
public static subscribe(
    commands: TcHmi.Server.ICommand[],
    interval: number,
    requestOptions: Server.IRequestOptions,
    callback?: null | (
         (data: TcHmi.Server.IResultObject) => void
    )
): number | null;
```

Registers a subscription on a list of commands.

Parameter

Name	Type	Description
commands	TcHmi.Server.ICommand [▶ 2002][]	An array of ICommand objects.
interval	number [▶ 2108]	The earliest time after which the TwinCAT HMI server transmits changes.
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
if (TcHmi.Server.isWebsocketReady()) {
    var commands = [
        {
            'symbol': 'PLC1.MAIN.bTest'
        },
        {
            'symbol': 'PLC1.MAIN.nTest'
        }];

    TcHmi.Server.subscribeEx(commands, 500, { timeout: 2000 }, function (data) {
        if (data.error !== TcHmi.Errors.NONE) {
            // Handle TcHmi.Server class level error here.
            return;
        }

        var response = data.response;
        if (response.error !== undefined) {
            // Handle TwinCAT HMI Server response level error here.
            return;
        }

        var commands = response.commands;
        if (commands === undefined) {
            return;
        }

        for (var i = 0, ii = commands.length; i < ii; i++) {
            var command = response.commands[i];
            if (command === undefined) {
                return;
            }
            if (command.error !== undefined) {
                // Handle TwinCAT HMI Server command level error here.
                return;
            }

            // Handle result...
            TcHmi.Log.debug(command.symbol + '=' + command.readValue);
        }
    });
}
```

15.1.1.16.12 unsubscribe

[Function]

```
public static unsubscribe(
    requestId: number,
    callback?: null | (
        (data: TcHmi.Server.IResultObject) => void
    )
): number | null;
```

Cancels an existing subscription.

Parameter

Name	Type	Description
requestId	number [▶ 2108]	The request ID of the subscription that is to be terminated.
callback [Optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
if (TcHmi.Server.isWebsocketReady()) {
    var commands = [{
        'symbol': 'PLC1.MAIN.bTest'
    },{
        'symbol': 'PLC1.MAIN.nTest'
    }];

    var requestId = TcHmi.Server.subscribe(
        commands,
        500,
        function (data) {
            // ...
    });

    setTimeout(function () {
        if (requestId === null) {
            return;
        }
        TcHmi.Server.unsubscribe(
            requestId,
            function (data) {
                if (data.error !== TcHmi.Errors.NONE) {
                    // Handle TcHmi.Server class level error here.
                    return;
                }
                var response = data.response;
                if (response.error !== undefined) {
                    // Handle TwinCAT HMI Server response level error here.
                    return;
                }

                var commands = response.commands;
                if (commands === undefined) {
                    return;
                }

                var command = response.commands[0];
                if (command === undefined) {
                    return;
                }

                if (command.error !== undefined) {
                    TcHmi.Log.debug('Unsubscribe for subscription with request id=' + requestId + '
failed.');
                } else {
                    TcHmi.Log.debug('Unsubscribe for subscription with request id=' + requestId + '
finished.');
                }
        });
    }, 2500);
}
```

15.1.1.16.13 unsubscribeEx

[Function]

```
public static unsubscribeEx(
    requestId: number,
    requestOptions: Server.IRequestOptions,
    callback?: null | (
        (data: TcHmi.Server.IResultObject) => void
    )
): number | null;
```

Cancels an existing subscription.

Parameter

Name	Type	Description
requestId	number [▶ 2108]	The request ID of the subscription that is to be terminated.
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	null [▶ 2108], (data: TcHmi.Server.IResultObject [▶ 2001]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	The request ID as number or null in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
if (TcHmi.Server.isWebsocketReady()) {
    var commands = [{
        'symbol': 'PLC1.MAIN.bTest'
    },{
        'symbol': 'PLC1.MAIN.nTest'
    }];

    var requestId = TcHmi.Server.subscribe(
        commands,
        500,
        function (data) {
            // ...
    });

    setTimeout(function () {
        if (requestId === null) {
            return;
        }
        TcHmi.Server.unsubscribeEx(
            requestId,
            { timeout: 2000 },
            function (data) {
                if (data.error !== TcHmi.Errors.NONE) {
                    // Handle TcHmi.Server class level error here.
                    return;
                }
                var response = data.response;
                if (response.error !== undefined) {
                    // Handle TwinCAT HMI Server response level error here.
                    return;
                }

                var commands = response.commands;
                if (commands === undefined) {
                    return;
                }

                var command = response.commands[0];
```

```
                if (command === undefined) {
                    return;
                }

                if (command.error !== undefined) {
                    TcHmi.Log.debug('Unsubscribe for subscription with request id=' + requestId + '
failed.');
                } else {
                    TcHmi.Log.debug('Unsubscribe for subscription with request id=' + requestId + '
finished.');
                }
        });
    }, 2500);
}
```

15.1.1.16.14 getCurrentUser

[Function]

```
public static getCurrentUser(): string | null;
```

Returns the name of the current user as string or null.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	The name of the currently logged on user or null in the event of an error or if no user is logged on.

Note: Available since version 1.8

Example - JavaScript

```
var user = TcHmi.Server.getCurrentUser();
if(user) {
    console.log('Current user is ' + user);
} else {
    console.log('We have no current user.');
}
```

15.1.1.16.15 getGroupsOfCurrentUser

[Function]

```
public static getGroupsOfCurrentUser(): string[];
```

Returns the names of the groups to which the current user is assigned as a string array.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
string [▶ 2108][]	The names of the groups of the currently logged on user. This list can be empty.

Note: Available since version 1.8

Example - JavaScript

```
var userGroup = TcHmi.Server.getGroupsOfCurrentUser();
if(userGroup.length) {
    console.log('Current user is in groups ' + userGroup.join(', '));
} else {
    console.log('We have no current user group.');
}
```

15.1.1.16.16 getCurrentUserConfig

[Function]

```
public static getCurrentUserConfig(): TcHmi.Server.userConfigOnServer;
```

Returns the current user configuration.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Server.userConfigOnServer [▶ 2005]	The configuration object of the currently logged on user.

Note: Available since version 1.8

Sample - JavaScript

```
var userConfig = TcHmi.Server.getCurrentUserConfig();
if(userConfig.session){
    console.log('Current user session is ' + userConfig.session);
}
```

See also

TcHmi.Localization [▶ 1894]

15.1.1.16.17 logout

[Function]

```
public static logout(
    callback?: null | (
        (data: TcHmi.IResultObject) => void
    ) = null
);
```

Logs out the current user.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered if the logout was not successful.

Return value

Type	Description
boolean [▶ 2107]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.8

Sample - JavaScript

```
TcHmi.Server.logout(function(data) {
    console.log('User logout not successful.');
});
```

15.1.1.16.18 logoutEx

[Function]

```
public static logoutEx(
    requestOptions: Server.IRequestOptions,
    callback: null | (data: TcHmi.IResultObject) => void
);
```

Logs out the current user.

Parameter

Name	Type	Description
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	TcHmi.IResultObject [▶ 2094]	Asynchronous callback function that is triggered if the logout was not successful.

Return value

Type	Description
boolean [▶ 2107]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Server.logoutEx({ timeout: 2000 }, function(data) {
    console.log('User logout not successful.');
});
```

15.1.1.16.19 forceLogout

t[Function]

```
public static forceLogout(
    username: string | TcHmi.Server.IForceLogoutTarget | null | undefined,
    callback?: null | (
        (data: TcHmi.IResultObject) => void
    ) = null
): boolean;
```

Logs out any user if the appropriate user privileges are available.

Parameter

Name	Type	Description
username	string [▶ 2108], TcHmi.Server.IForceLogoutTarget [▶ 2006], null [▶ 2108], undefined [▶ 2108],	The name of the user to be logged out. If an empty string or null is passed, **all** users of the server are logged out. From version 1.10.1172.0: The authentication domain can be added as name: `'Domain::UserName'`. If only the authentication domain is specified, all users of this domain will be logged out. Alternatively, an object can be passed that specifies conditions for a user to log out.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
boolean [▶ 2107]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.8

Note: Up to version 1.10.1172.0 only string and null were allowed as userName.

Sample - JavaScript

```
// Logout a user with this name in all authenfication domains
TcHmi.Server.forceLogout('johnDoe', function(data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

From version 1.10.1172.0

```
// Logout johnDoe from the authenfication domain TcHmiUserManagement
TcHmi.Server.forceLogout(
    'TcHmiUserManagement::johnDoe',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

```
// Logout all users from this authenfication domain
TcHmi.Server.forceLogout(
    'TcHmiUserManagement::',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

```
// Logout all users which matches ONE of the conditions:
// IP: 127.0.0.1 (as seen from the server)
// clientCertificate with this fingerprint/thumbprint
// member of the group 'SystemAdministrator'
TcHmi.Server.forceLogout(
    {
        clientIp: '127.0.0.1',
        clientCertificate:
'8e810e91d365662783480d0e4f54f9e037bc1157c9f5634a36bbcc096530dd960f66e5a072d069b5c8dce66bb633166131b
8ad49467d27a93cd72cd96f346a80',
        group: 'SystemAdministrator'
    },
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.20 forceLogoutEx

[Function]

```
public static forceLogoutEx(
    username: string | TcHmi.Server.IForceLogoutTarget | null | undefined,
    requestOptions: Server.IRequestOptions,
    callback?: null | (
        (data: TcHmi.IResultObject) => void
    ) = null
): boolean;
```

Logs out any user if the appropriate user privileges are available.

Parameter

Name	Type	Description
username	string [▶ 2108], TcHmi.Server.IForceLogoutTarget [▶ 2006], null [▶ 2108], undefined [▶ 2108],	The name of the user to be logged out. If an empty string or null is passed, **all** users of the server are logged out. From version 1.10.1172.0: The authentication domain can be added as name: `'Domain::UserName'`. If only the authentication domain is specified, all users of this domain will be logged out. Alternatively, an object can be passed that specifies conditions for a user to log out.
requestOptions	server.IRequestOptions [▶ 2006]	Options
callback [optional]	null | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered when the operation was terminated.

Return value

Type	Description
boolean [▶ 2107]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Note: Up to version 1.10.1172.0 only string and null were allowed as userName.

Sample - JavaScript

```
// Logout a user
TcHmi.Server.forceLogoutEx('johnDoe',{ timeout: 2000 }, function(data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

From version 1.10.1172.0

```
// Logout johnDoe from the authenfication domain TcHmiUserManagement
TcHmi.Server.forceLogoutEx(
    'TcHmiUserManagement::johnDoe',
    { timeout: 2000 },
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

```
// Logout all users from this authenfication domain
TcHmi.Server.forceLogoutEx(
    'TcHmiUserManagement::',
    { timeout: 2000 },
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

```
// Logout all users which matches ONE of the conditions:
// IP: 127.0.0.1 (as seen from the server)
// clientCertificate with this fingerprint/thumbprint
// member of the group 'SystemAdministrator'
TcHmi.Server.forceLogoutEx(
    {
        endpoint: '127.0.0.1',
        clientCertificate:
'8e810e91d365662783480d0e4f54f9e037bc1157c9f5634a36bbcc096530dd960f66e5a072d069b5c8dce66bb633166131b
8ad49467d27a93cd72cd96f346a80',
        group: 'SystemAdministrator'
    },
    { timeout: 2000 },
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.21 Events

[Class]

Enables receiving of server events and acknowledging of alarms.

Static functions

confirmAlarm [▶ 1921]	Acknowledges an alarm.
registerConsumer [▶ 1921]	Registers a consumer to receive events.

Interfaces

Event [▶ 1923]	A server event. Basic class for message and alarm.
Message [▶ 1923]	A stateless message.
Alarm [▶ 1924]	An alarm.
ListResult [▶ 1925]	Extends TcHmi.IResultObject [▶ 2094]. Parameter for the list callback function of a consumer.
SubscriptionResult [▶ 1925]	Extends TcHmi.IResultObject [▶ 2094]. Parameter for the subscription callback function of a consumer.

Enumerations

ChangeType [▶ 1926]	Indicates how an event has changed.
ConfirmationState [▶ 1926]	Indicates whether an alarm can be acknowledged.
Severity [▶ 1927]	The severity of an event.
Type [▶ 1927]	Indicates whether an event is an alarm or a message.

Note: Available since version 1.10

15.1.1.16.21 confirmAlarm .1

[Function]

```
public static confirmAlarm(alarm: Events.Alarm, callback?: (data: TcHmi.IResultObject) => void):
void;
```

Acknowledges an alarm.

Parameter

Name	Type	Description
alarm	Alarm [▶ 1924]	The alarm to be acknowledged
callback [Optional]	(data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
void	No return value

Example - JavaScript

```
TcHmi.Server.Events.confirmAlarm(myAlarms[0], function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

Note: Available since version 1.10

15.1.1.16.21 registerConsumer .2

[Function]

```
public static registerConsumer(
    filter: Filter | null,
    callbacks: {
        list?: (data: Events.ListResult) => void,
        subscription?: (data: Events.SubscriptionResult) => void
}): DestroyFunction | null
```

Registers a user for server events [▶ 1923]. A consumer can specify two optional functions: `list` is called after the user has logged in or when the location changes, `subscription` is called when alarms change or new events are created.

Only events matching the specified filter [▶ 2087] are forwarded to the consumer. If no filter is to be used, null [▶ 2108] must be passed.

Parameter

Name	Type	Description
filter	Filter [▶ 2087], null [▶ 2108]	Defines which events are to be queried

callbacks	{
	list?: (data: Events.ListResult [▶ 1925]) => void,
	subscription?: (data: Events.SubscriptionResult [▶ 1925]) => void
	}
		An object [▶ 2108] containing the callback functions to which received events are passed.

Return value

Type	Description
DestroyFunction [▶ 2101], null [▶ 2108]	A function with which the consumer can be logged off again. If the user login fails, null is returned.

Example - JavaScript

```
// filter for unconfirmed alarms
var filter = [
    {
        path: 'type',
        comparator: '==',
        value: TcHmi.Server.Events.Type.Alarm
    },
    {
        logic: 'AND'
    },
    {
        path: 'timeConfirmed',
        comparator: '==',
        value: new Date(null)
    }
];

var myAlarms = [];

// function to handle list of events
function consumeEventList(data) {
    if (data.error === TcHmi.Errors.NONE) {
        myAlarms = data.events;
    }
    else {
        myAlarms = [];
    }
}

// function to handle new and changed incoming events
function consumeEventSubscription(data) {
    if (data.error !== TcHmi.Errors.NONE) {
        return;
    }
    if (data.removedByFilter) {
```

```
        myAlarms = myAlarms.filter(function (alarm) {
            return alarm.id !== data.event.id;
        });
    }
    else if (data.changeType === TcHmi.Server.Events.ChangeType.AlarmRaised) {
        myAlarms.push(data.event);
    }
    else {
        for (var i = 0; i < myAlarms.length; i++) {
            if (myAlarms[i].id === data.event.id) {
                myAlarms[i] = data.event;
                break;
            }
        }
    }
}

// register a consumer to receive events
var destroyFunction = TcHmi.Server.Events.registerConsumer(filter, {
    list: consumeEventList,
    subscription: consumeEventSubscription
});
```

Note: Available since version 1.10

15.1.1.16.21 Event .3

[Interface]

```
export interface Event {
    type: Type;
    severity: Severity;
    domain: string;
    name: string;
    text: string;
    timeRaised: Date;
    params: Dictionary<any>;
}
```

Properties

Name	Type	Description
type	Type [▶ 1927]	The type of event
severity	Severity [▶ 1927]	The severity of the event
domain	string [▶ 2108]	The domain of the event
name	string [▶ 2108]	The name of the event
text	string [▶ 2108]	The localized text of the event
timeRaised	Date [▶ 2108]	The time at which the event was triggered
params	Dictionary [▶ 2086]	Parameters set by the trigger of the event

Note: Available since version 1.10

15.1.1.16.21 Message .4

[Interface]

```
export interface Message extends Event {
    type: Type.Message;
}
```

This interface extends Event [▶ 1923].

Properties

Name	Type	Description
type	Type [▶ 1927].Message	Ensures that instances of this interface have always set the type Message.
severity	Severity [▶ 1927]	The severity of the event
domain	string [▶ 2108]	The domain of the event
name	string [▶ 2108]	The name of the event
text	string [▶ 2108]	The localized text of the event
timeRaised	Date [▶ 2108]	The time at which the event was triggered
params	Dictionary [▶ 2086]	Parameters set by the trigger of the event

Note: Available since version 1.10

15.1.1.16.21 Alarm .5

[Interface]

```
export interface Alarm extends Event {
    type: Type.Alarm;
    id: number | string;
    timeCleared: Date;
    timeConfirmed: Date;
    confirmationState: ConfirmationState;
}
```

This interface extends Event [▶ 1923].

Properties

Name	Type	Description
type	Type [▶ 1927].Alarm	Ensures that instances of this interface have always set the type Alarm.
severity	Severity [▶ 1927]	The severity of the event
domain	string [▶ 2108]	The domain of the event
name	string [▶ 2108]	The name of the event
text	string [▶ 2108]	The localized text of the event
timeRaised	Date [▶ 2108]	The time at which the event was triggered
params	Dictionary [▶ 2086]	Parameters set by the trigger of the event
id	number [▶ 2108], string [▶ 2108]	A unique value with which this alarm can be identified
timeCleared	Date [▶ 2108]	The time at which the alarm was confirmed by the trigger as no longer acute
timeConfirmed	Date [▶ 2108]	The time at which the alarm was acknowledged by the user
confirmationState	ConfirmationState [▶ 1926]	The current confirmation status

Note: Available since version 1.10

15.1.1.16.21 isAlarm .6

[Function]

```
export function isAlarm(value: Event): boolean;
```

Checks whether the passed Event [▶ 1923] is an Alarm [▶ 1924].

Parameter

Name	Type	Description
value	Event [▶ 1923]	The event to be checked

Return value

Type	Description
boolean [▶ 2107]	Returns true if the passed event is an alarm, otherwise false.

Note: Available since version 1.10

15.1.1.16.21 ListResult .7

[Interface]

```
export interface ListResult extends TcHmi.IResultObject{
    events?: Event[];
}
```

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
error	Errors [▶ 2089]	Error state of the request
details [Optional]	any [▶ 2107]	Optional error details. Content depends on the function used and the error.
events [Optional]	Event [▶ 1923][]	The list of events returned by the server

Note: Available since version 1.10

15.1.1.16.21 SubscriptionResult .8

[Interface]

```
export interface SubscriptionResult extends TcHmi.IResultObject{
    event?: Event;
    changeType?: ChangeType;
    removedByFilter?: boolean;
}
```

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
error	Errors [▶ 2089]	Error state of the request
details [Optional]	any [▶ 2107]	Optional error details. Content depends on the function used and the error.
event [Optional]	Event [▶ 1923]	The event that has changed
changeType [Optional]	ChangeType [▶ 1926]	How the event has changed
removedByFilter [Optional]	boolean [▶ 2107]	Is true if the event has changed so that it no longer fits the configured filter. No updates will be made for this event unless a future change will make it fit the filter again.

Note: Available since version 1.10

15.1.1.16.21 ChangeType .9

[Enumeration]

```
export enum ChangeType {
    AlarmRaised,
    AlarmChanged,
    AlarmDisposed,
    MessageSent
}
```

The names of the enumeration exist at runtime.

Values

Name	Value	Description
AlarmRaised	0	An alarm was triggered
AlarmChanged	1	An alarm was changed
AlarmDisposed	2	An alarm was deleted
MessageSent	3	A message was sent

Note: Available since version 1.10

15.1.1.16.21 ConfirmationState .10

[Enumeration]

```
export enum ConfirmationState {
    NotSupported = 0,
    NotRequired = 1,
    WaitForConfirmation = 2,
    Confirmed = 3,
    Reset = 4
}
```

The names of the enumeration exist at runtime.

Values

Name	Value	Description
NotSupported	0	Acknowledgement is not supported
NotRequired	1	Acknowledgement not necessary
WaitForConfirmation	2	Waiting for acknowledgement
Confirmed	3	Acknowledged
Reset	4	Reset

Note: Available since version 1.10

15.1.1.16.21.11 Severity

[Enumeration]

```
export enum Severity {
    Verbose = 0,
    Info = 1,
    Warning = 2,
    Error = 3,
    Critical = 4
}
```

The names of the enumeration exist at runtime. For example, you can use `TcHmi.Server.Events.Severity.Warning` directly in a comparison.

Values

Name	Value	Description
Verbose	0	An unimportant piece of information
Info	1	An information
Warning	2	A warning
Error	3	An error
Critical	4	A critical error

Note: Available but version 1.10

15.1.1.16.21.12 Type

[Enumeration]

```
export enum Type {
    Message = 0,
    Alarm = 1
}
```

The names of the enumeration exist at runtime. For example, you can use `TcHmi.Server.Events.Type.Alarm` directly in a comparison.

Values

Name	Value	Description
Message	0	A stateless message
Alarm	1	An alarm

Note: Available since version 1.10

15.1.1.16.22 Historize

[Class]

Provides functions to edit the server's Historize configuration.

Static functions

add [▶ 1928]	Adds an icon to the server's Historize configuration.
addEx [▶ 1929]	Adds an icon to the server's Historize configuration. Function with more options.
addEx2 [▶ 1930]	Adds an icon to the server's Historize configuration. Function with even more options.
remove [▶ 1931]	Removes an icon from the server's Historize configuration.
removeEx [▶ 1931]	Removes an icon from the server's Historize configuration. Function with more options.
removeEx2 [▶ 1932]	Removes an icon from the server's Historize configuration. Function with even more options.
update [▶ 1933]	Updates the settings of an icon in the server's Historize configuration.
updateEx [▶ 1933]	Updates the settings of an icon in the server's Historize configuration. Function with more options.
updateEx2 [▶ 1934]	Updates the settings of an icon in the server's Historize configuration. Function with even more options.
get [▶ 1935]	Returns the settings for an icon from the server's Historize configuration.
getEx [▶ 1936]	Returns the settings for an icon from the server's Historize configuration. Function with even more options.
getEx2 [▶ 1936]	Returns the settings for an icon from the server's Historize configuration. Function with even more options.

Interfaces

IEntrySettings [▶ 1937]	Historize symbol settings.
ISettings [▶ 1938]	Historize settings
IEntryCallbackParameter [▶ 1938]	This interface extends TcHmi.ICallback parameters.

Note: Available since version 1.10

15.1.1.16.22 add
.1

[Function]

```
public static add(
    symbolName: string,
    settings: TcHmi.Server.Historize.IEntrySettings,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Adds an icon to the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
settings	TcHmi.Server.Historize.IEntrySettings [▶ 1937]	The settings of the entry in the Historize configuration.
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
var settings = {
    INTERVAL: 'PT100ms',
    MAXENTRIES: 10000,
    ROWLIMIT: 10000,
    RECORDINGENABLED: true,
};
TcHmi.Server.Historize.add('PLC1.MAIN.nInt2', settings, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 addEx .2

[Function]

```
public static addEx(
    symbolName: string,
    settings: TcHmi.Server.Historize.IEntrySettings,
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Adds an icon to the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
settings	TcHmi.Server.Historize.IEntrySettings [▶ 1937]	The settings of the entry in the Historize configuration.
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
var settings = {
    INTERVAL: 'PT100ms',
    MAXENTRIES: 10000,
    ROWLIMIT: 10000,
    RECORDINGENABLED: true,
};
TcHmi.Server.Historize.addEx('PLC1.MAIN.nInt2', settings, {timeout: 2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 addEx2
.3

[Function]

```
public static addEx2(
    symbolName: string,
    settings: TcHmi.Server.Historize.IEntrySettings,
    options: TcHmi.Server.Historize.ISetting | null,
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Adds an icon to the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
settings	TcHmi.Server.Historize.IEntrySettings [▶ 1937]	The settings of the entry in the Historize configuration.
options	TcHmi.Server.Historize.ISetting [▶ 1938], null [▶ 2108]	Options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.670.0

Example - JavaScript

```
var settings = {
    INTERVAL: 'PT100ms',
    MAXENTRIES: 10000,
    ROWLIMIT: 10000,
    RECORDINGENABLED: true,
};
TcHmi.Server.Historize.addEx('PLC1.MAIN.nInt2', settings, {timeout: 2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 remove .4

[Function]

```
public static remove(
    symbolName: string,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Removes an icon from the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Server.Historize.remove('PLC1.MAIN.nInt2', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 removeEx .5

[Function]

```
public static removeEx(
    symbolName: string,
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Removes an icon from the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Server.Historize.removeEx('PLC1.MAIN.nInt2', {timeout: 2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 removeEx2
.6

[Function]

```
public static removeEx2(
    symbolName: string,
    options: TcHmi.Server.Historize.ISetting | null
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Removes an icon from the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
options	TcHmi.Server.Historize.ISetting [▶ 1938], null [▶ 2108]	Options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.670.0

Example - JavaScript

```
TcHmi.Server.Historize.removeEx2('PLC1.MAIN.nInt2', {domain: 'TcHmiSqliteHistorize'}, {timeout:
2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 update
.7

[Function]

```
public static update(
    symbolName: string,
    settings: TcHmi.Server.Historize.IEntrySettings,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Updates the settings of an icon in the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
settings	TcHmi.Server.Historize.IEntrySettings [▶ 1937]	The settings of the recording.
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
var settings = {
    RECORDINGENABLED: false,
};
TcHmi.Server.Historize.update('PLC1.MAIN.nInt2', settings, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 updateEx
.8

[Function]

```
public static updateEx(
    symbolName: string,
    settings: TcHmi.Server.Historize.IEntrySettings,
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Updates the settings of an icon in the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
settings	TcHmi.Server.Historize.IEntrySettings [▶ 1937]	The settings of the recording.
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
var settings = {
    RECORDINGENABLED: false,
};
TcHmi.Server.Historize.updateEx('PLC1.MAIN.nInt2', settings, {timeout: 2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 updateEx2
.9

[Function]

```
public static updateEx2(
    symbolName: string,
    settings: TcHmi.Server.Historize.IEntrySettings,
    options: TcHmi.Server.Historize.ISetting | null,
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Updates the settings of an icon in the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol.
settings	TcHmi.Server.Historize.IEntrySettings [▶ 1937]	The settings of the recording.
options	TcHmi.Server.Historize.ISetting [▶ 1938], null [▶ 2108]	Options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.670.0

Example - JavaScript

```
var settings = {
    RECORDINGENABLED: false,
};
TcHmi.Server.Historize.updateEx2('PLC1.MAIN.nInt2', settings, {}, {timeout: 2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.22 get
.10

[Function]

```
public static get(
    symbolName: string,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Returns the settings for an icon from the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol to be recorded.
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Server.Historize.get('PLC1.MAIN.nInt2', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
        var key = data.key;
        var settings = data.settings;
    } else {
        // Error
    }
});
```

15.1.1.16.22 getEx .11

[Function]

```
public static getEx(
    symbolName: string,
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Returns the settings for an icon from the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol to be recorded.
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Server.Historize.getEx('PLC1.MAIN.nInt2', {timeout: 2000}, function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
        var key = data.key;
        var settings = data.settings;
    } else {
        // Error
    }
});
```

15.1.1.16.22 getEx2 .12

[Function]

```
public static getEx2(
    symbolName: string,
    options: TcHmi.Server.Historize.ISetting | null
    requestOptions: TcHmi.Server.IRequestOptions | null,
    callback?: null | ((this: void, data: TcHmi.IResultObject) => void)
): TcHmi.IErrorDetails;
```

Returns the settings for an icon from the server's Historize configuration.

Parameter

Name	Type	Description
SymbolName	string [▶ 2108],	The name of the symbol to be recorded.
options	TcHmi.Server.Historize.ISetting [▶ 1938], null [▶ 2108]	Options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	null [▶ 2108] | (data: TcHmi.IResultObject [▶ 2094]) => void	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.670.0

Example - JavaScript

```
TcHmi.Server.Historize.getEx2('PLC1.MAIN.nInt2', {domain: 'TcHmiSqliteHistorize'}, {timeout: 2000},
function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
        var key = data.key;
        var settings = data.settings;
    } else {
        // Error
    }
});
```

15.1.1.16.22 IEntrySettings .13

[Interface]

```
export interface IEntrySettings {
    INTERVAL?: string;
    MAXENTRIES?: number;
    ROWLIMIT?: number;
    RECORDINGENABLED?: boolean;
};
```

Historize symbol settings.

Properties

Name	Type	Description
INTERVAL [optional]	string [▶ 2108]	
MAXENTRIES [optional]	number [▶ 2108]	
ROWLIMIT [Optional]	number [▶ 2108]	
RECORDINGENABLED [optional]	boolean [▶ 2107]	

Note: Available since version 1.10

15.1.1.16.22 ISettings
.14

[Interface]

```
export interface ISettings {
    domain?: string;
};
```

Global historize settings.

Properties

Name	Type	Description
domain [optional]	string [▶ 2108]	Extension which deals with historizing. Default: `'TcHmiSqliteHistorize'`

Note: Available since version 1.10.970.0

15.1.1.16.22 IEntryResultObject
.15

[Interface]

```
export interface IEntryResultObject extends TcHmi.IResultObject{
    key: string;
    settings: IEntrySettings
};
```

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
error	Errors [▶ 2089]	Error state of the request
details [optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
key	string [▶ 2108]	The key of the Historize entry in the server configuration. The key is the name of a symbol mapping.
settings	IEntrySettings [▶ 1937]	Historization settings

Note: Available since version 1.10

15.1.1.16.23 UserManagement

[Class]

Enables reading and modifying of users.

Static functions

addUser [▶ 1939]	Creates a new user.
addUserEx [▶ 1940]	Creates a new user.
listUsernames [▶ 1941]	Supplies a list of all existing system users.
listUsernamesEx [▶ 1941]	Supplies a list of all existing system users.
listUsers [▶ 1942]	Supplies a list of all existing system users as object with all metadata.
listUsersEx [▶ 1943]	Supplies a list of all existing system users as object with all metadata.
listUserGroups [▶ 1944]	Supplies a list of all existing system groups as object with all metadata.
listUserGroupsEx [▶ 1945]	Supplies a list of all existing system groups as object with all metadata.
listUsersInGroup [▶ 1946]	Supplies a list of all existing users in a group.
listUsersInGroupEx [▶ 1947]	Supplies a list of all existing users in a group.
removeUser [▶ 1948]	Deletes a user.
removeUserEx [▶ 1948]	Deletes a user.
updateUser [▶ 1949]	Modifies a user.
updateUserEx [▶ 1950]	Modifies a user.

Interfaces

IUserDetails [▶ 1951]	Saves metadata of a user.
IGroupDetails [▶ 1952]	Saves metadata of a group.
IUpdateUserDetails [▶ 1953]	Change details of the metadata of a user.
IUsernameListResultObject [▶ 1954]	Saves a list of user names.
IUserResultObject [▶ 1955]	Result of a ListUsers call.
IGroupResultObject [▶ 1955]	Result of a ListUserGroups call.
IUserManagementOptions [▶ 1955]	UserManagement options

Note: Available since version 1.10.1018.48

15.1.1.16.23 addUser
.1

[Function]

```
public static addUser(
    userName: string,
    password: string,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new user.

Each new user is by default at least a member of the "__SystemUsers" group and uses the language of the project. If no domain is specified, the extension TcHmiUserManagement is used for user management.

This behavior can be changed on the server configuration page under TcHmiSrv under the name "Default Usergroup" and "Default authentication extension".

Parameter

Name	Type	Description
userName	string [▶ 2108]	Name of the new user
password	string [▶ 2108]	Plain text password of the new user
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.ErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.addUser('newUser', 'Gbt3fC79ZmMEFUFJ', function(data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Success
    } else {
        // Error
    }
});
```

15.1.1.16.23 addUserEx .2

[Function]

```
public static addUserEx(
    userName: string,
    password: string,
    options: Partial<TcHmi.Server.UserManagement.IUserDetails> | null = null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new user.

Each new user is by default at least a member of the "__SystemUsers" group and uses the language of the project. If no domain is specified, the extension TcHmiUserManagement is used for user management.

This behavior can be changed on the server configuration page under TcHmiSrv under the name "Default Usergroup" and "Default authentication extension".

Parameter

Name	Type	Description
userName	string [▶ 2108]	Name of the new user
password	string [▶ 2108]	Plain text password of the new user
options	TcHmi.Server.UserManagement.IUserDetails [▶ 1951], null [▶ 2108]	Details of the new user. If a property is not set, the default applies.
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.addUserEx(
    'newUser',
    'Gbt3fC79ZmMEFUFJ',
    {groups: ['__SystemAdministrator'], enabled: false, locale: 'de'},
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 listUsernames .3

[Function]

```
public static listUsernames(
    callback: null | ((this: void, data: TcHmi.Server.UserManagement.IUsernameListResultObject) =>
void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system users.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.Server.UserManagement.IUsernameListResultObject [▶ 1954]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.listUsernames(function(data) {
    if (data.error === TcHmi.Errors.NONE) {
        console.log(data.userList); // ["__SystemAdministrator", "__SystemGuest"]
    } else {
        // Error
    }
});
```

15.1.1.16.23 listUsernamesEx .4

[Function]

```
public static listUsernames(
    options?: TcHmi.Server.UserManagement.IUserManagementOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.UserManagement.IUsernameListResultObject) =>
void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system users.

Parameter

Name	Type	Description
options	TcHmi.Server.UserManagement.IUserManagementOptions [▶ 1955], null [▶ 2108]	User management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null	Server request options
callback [optional]	(data: TcHmi.Server.UserManagement.IUsernameListResultObject [▶ 1954]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.listUsernamesEx(
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.userList); // ["__SystemAdministrator", "__SystemGuest"]
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 listUsers
.5

[Function]

```
public static listUsers(
    callback: null | ((this: void, data: TcHmi.Server.UserManagement. IUserResultObject) => void) =
null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system users as object with all metadata.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.Server.UserManagement.IUserResultObject [▶ 1955]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.listUsers(function(data) {
    if (data.error === TcHmi.Errors.NONE) {
        console.log(data.userDetails);
/*
{
    "__SystemAdministrator": {
        "domain": "TcHmiUserManagement",
        "enabled": false,
        "autoLogout": "P30D",
        "locale": "en",
        "groups": [
            "__SystemAdministrators"
        ]
    },
    "__SystemGuest": {
        "domain": "TcHmiUserManagement",
        "enabled": true,
        "autoLogout": "P30D",
        "locale": "en",
        "groups": [
         "__SystemGuests"
        ]
    }
}
*/
    } else {
        // Error
    }
});
```

15.1.1.16.23 listUsersEx .6

[Function]

```
public static listUsersEx(
    options?: TcHmi.Server.UserManagement.IUserManagementOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.UserManagement. IUserResultObject) => void) =
null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system users as object with all metadata.

Parameter

Name	Type	Description
options	TcHmi.Server.UserManagement.IUserManagementOptions [▶ 1955], null [▶ 2108]	User management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.UserManagement.IUserResultObject [▶ 1955]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Sample - JavaScript

```
TcHmi.Server.UserManagement.listUsersEx(
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
                console.log(data.userDetails);
/*
{
    "__SystemAdministrator": {
        "domain": "TcHmiUserManagement",
        "enabled": false,
        "autoLogout": "P30D",
        "locale": "en",
        "groups": [
            "__SystemAdministrators"
        ]
    },
    "__SystemGuest": {
        "domain": "TcHmiUserManagement",
        "enabled": true,
        "autoLogout": "P30D",
        "locale": "en",
        "groups": [
         "__SystemGuests"
        ]
    }
}
*/
            } else {
                // Error
            }
        }
);
```

15.1.1.16.23 listUserGroups
.7

[Function]

```
public static listUserGroups(
    callback: null | ((this: void, data: TcHmi.Server.UserManagement.IGroupResultObject) => void) =
null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system groups as object with all metadata.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.Server.UserManagement.IGroupResultObject [▶ 1955]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.21

Example - JavaScript

```
TcHmi.Server.UserManagement.listUserGroups(function(data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Are Guests able to fetch current user status?
        console.log(data.groupDetailsList.__SystemGuests.symbols.GetCurrentUser ===
TcHmi.Server.ACCESS.READWRITE);
        console.log(data.groupDetailsList);
/*
{
    "__SystemAdministrators": {
        "enabled": true,
        "fileAccess": 3,
        "files": {},
        "symbolAccess": 3,
        "symbols": {}
    },
    "__SystemGuests": {
        "enabled": true,
        "fileAccess": 0,
        "files": {},
        "symbolAccess": 0,
        "symbols": {
            "GetCurrentUser": 3,
            "IsAuthRequired": 3,
            "ListUserNames": 3,
            "Login": 3
        }
    } // and more…
}
*/
    } else {
        // Error
    }
});
```

15.1.1.16.23 listUserGroupsEx .8

[Function]

```
public static listUserGroupsEx(
    options?: object | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.UserManagement.IGroupResultObject) => void) =
null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system users as object with all metadata.

Parameter

Name	Type	Description
options	object [▶ 2108], null [▶ 2108]	current unused
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.UserManagement.IGroupResultObject [▶ 1955]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1040.0

Sample - JavaScript

```
TcHmi.Server.UserManagement.listUserGroupsEx (
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Are Guests able to fetch current user status?
            console.log(data.groupDetailsList.__SystemGuests.symbols.GetCurrentUser ===
TcHmi.Server.ACCESS.READWRITE);
            console.log(data.groupDetailsList);
/*
{
    "__SystemAdministrators": {
        "enabled": true,
        "fileAccess": 3,
        "files": {},
        "symbolAccess": 3,
        "symbols": {}
    },
    "__SystemGuests": {
        "enabled": true,
        "fileAccess": 0,
        "files": {},
        "symbolAccess": 0,
        "symbols": {
            "GetCurrentUser": 3,
            "IsAuthRequired": 3,
            "ListUserNames": 3,
            "Login": 3
        }
    } // and more…
}
*/
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 listUsersInGroup .9

[Function]

```
public static listUsersInGroup(
    groupName: string,
    callback: null | ((this: void, data: TcHmi.Server.UserManagement.IUsernameListResultObject) =>
void) = null): TcHmi.IErrorDetails;
```

Supplies a list of all existing users in a group.

Parameter

Name	Type	Description
groupName	string [▶ 2108]	
callback [optional]	(data: TcHmi.Server.UserManagement.IUsernameListResultObject [▶ 1954]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.listUsersInGroup (
    '__SystemAdministrator',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.userList); // ["__SystemAdministrator", "__SystemGuest"]
        } else {
            // Error
        }
});
```

15.1.1.16.23 listUsersInGroupEx .10

[Function]

```
public static listUsersInGroupEx(
    groupName: string,
    options?: TcHmi.Server.UserManagement.IUserManagementOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.UserManagement.IUsernameListResultObject) =>
void) = null): TcHmi.IErrorDetails;
```

Supplies a list of all existing users in a group.

Parameter

Name	Type	Description
groupName	string [▶ 2108]	
options	TcHmi.Server.UserManagement.IUserManagementOptions [▶ 1955], null [▶ 2108]	User management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.UserManagement.IUsernameListResultObject [▶ 1954]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.listUsersInGroupEx (
    '__SystemAdministrator',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.userList); // ["__SystemAdministrator", "__SystemGuest"]
        } else {
            // Error
        }
});
```

15.1.1.16.23 removeUser .11

[Function]

```
public static removeUser(
    userName: string,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a user.

Parameter

Name	Type	Description
userName	string [▶ 2108]	Name of the user
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.removeUser (
    'newUser',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 removeUserEx .12

[Function]

```
public static removeUserEx(
    userName: string,
    options?: TcHmi.Server.UserManagement.IUserManagementOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a user.

Parameter

Name	Type	Description
userName	string [▶ 2108]	Name of the user
options	TcHmi.Server.UserManagement.IUserManagementOptions [▶ 1955], null [▶ 2108]	User management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.removeUserEx (
    'newUser',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 updateUser .13

[Function]

```
public static updateUser(
    userName: string,
    options?: TcHmi.Server.UserManagement.IUpdateUserDetails | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null): TcHmi.IErrorDetails;
```

Modifies a user.

Parameter

Name	Type	Description
userName	string [▶ 2108]	Name of the user
options	TcHmi.Server.UserManagement.IUpdateUserDetails [▶ 1953], null [▶ 2108]	Details of the new user. If a property is not set, it remains unchanged. The "domain" property cannot be changed; it is only used for addressing.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.updateUser(
    'myUser',
    {
        domain: 'myDomain', // Only for adressing, not changeable
        enabled: false,
        newName: 'myNewName',
        password: 'myNewPassword',
        autoLogout: 'PT42S',
        locale: 'fr',
        addGroups: ['myGroup1', 'myGroup2'],
        removeGroups: ['notMyGroup1', 'notMyGroup2']
    },
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 updateUserEx .14

[Function]

```
public static updateUserEx(
    userName: string,
    options?: TcHmi.Server.UserManagement.IUpdateUserDetails | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null): TcHmi.IErrorDetails;
```

Modifies a user.

Parameter

Name	Type	Description
userName	string [▶ 2108]	Name of the user
options	TcHmi.Server.UserManagement.IUpdateUserDetails [▶ 1953], null [▶ 2108]	Details of the new user. If a property is not set, it remains unchanged. The "domain" property cannot be changed; it is only used for addressing.
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1018.48

Example - JavaScript

```
TcHmi.Server.UserManagement.updateUserEx(
    'myUser',
    {
        domain: 'myDomain', // Only for adressing, not changeable
        enabled: false,
        newName: 'myNewName',
        password: 'myNewPassword',
        autoLogout: 'PT42S',
        locale: 'fr',
        addGroups: ['myGroup1', 'myGroup2'],
        removeGroups: ['notMyGroup1', 'notMyGroup2']
    },
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.23 IUserDetails
.15

[Interface]

```
export interface IUserDetails {
    domain: string;
    enabled: boolean;
    autoLogout: string;
    locale?: string | null;
    timeFormatLocale?: string | null;
    timeZone?: string | null;
    groups: string[];
};
```

Saves metadata of a user.

Properties

Name	Type	Description
domain	string [▶ 2108]	User management domain. If this is not specified, the default specified below applies.
enabled	boolean [▶ 2107]	Determines whether the user is active. true if not specified
autoLogout	string [▶ 2108]	Time interval as ISO 8601 timespan. String that indicates after what time the user is logged out. Format: P[*JY*][*MM*][*WW*][*TD*][T[*h*H][*m*M]] Sample: "`P30D`" for 30 days, "`P1DT12H`" for one day and 12 hours, "`PT30M`" for 30 minutes.
locale	string [▶ 2108]	Language settings of the user. From version 1.10.1336.10 'client' the language of the browser or 'project' is valid for the project settings.
timeFormatLocale From version 1.10.1336.10	string [▶ 2108]	Language settings of the user for time/date representations. Here 'client' is valid for the language of the browser or 'project' for the project setting.
timeZone From version 1.10.1336.10	string [▶ 2108]	Time zone of the user for time/date representations. Here 'client' is valid for the language of the browser or 'project' for the project setting.
groups	string [▶ 2108][]	Groups in which the user is a member.

Note: Available since version 1.10.1018.48

Behavior for new users

Each new user is by default at least a member of the "__SystemUsers" group and uses the language of the project. If no domain is specified, the extension TcHmiUserManagement is used for user management.

This behavior can be changed on the server configuration page under TcHmiSrv under the name "Default Usergroup" and "Default authentication extension".

15.1.1.16.23 IGroupDetails .16

[Interface]

```
export interface IUserDetails {
    enabled: boolean;
    fileAccess: TcHmi.Server.ACCESS;
    files: Dictionary<TcHmi.Server.ACCESS>;
    symbolAccess: TcHmi.Server.ACCESS;
    symbols: Dictionary<TcHmi.Server.ACCESS>;
};
```

Saves metadata of a group.

Properties

Name	Type	Description
enabled	boolean [▶ 2107]	Determines whether the user is active; true if not specified
fileAccess	TcHmi.Server.ACCESS [▶ 2007]	Determines whether files can be read
files	Dictionary<TcHmi.Server.ACCESS [▶ 2007]>	Determines access rights for individual files.
symbolAccess	TcHmi.Server.ACCESS [▶ 2007]	Determines whether symbols can be read and/or written
symbols	Dictionary<TcHmi.Server.ACCESS [▶ 2007]>	Determines access rights for individual symbols.

Note: Available since version 1.10.1171.21

15.1.1.16.23 IUpdateUserDetails .17

[Interface]

```
export interface IUpdateUserDetails {
    domain: string;
    newName?: string;
    password?: string;
    enabled: boolean;
    autoLogout: string;
    locale: string;
    addGroups?: string[];
    removeGroups?: string[];
};
```

Change details of the metadata of a user.

Properties

Name	Type	Description
domain	string [▶ 2108]	User management domain. 'TcHmiUserManagement' if not specified
newName	string [▶ 2108]	New name of the user.
password	string [▶ 2108]	New password of the user
enabled	boolean [▶ 2107]	Determines whether the user is active. true if not specified
autoLogout	string [▶ 2108]	Time interval as ISO 8601 timespan string that indicates the time after which the user is logged out. Format: P[*JY*][*M*M][*W*W][*T*D][T[*h*H][*m*M]] Sample: "`P30D`" for 30 days, "`P1DT12H`" for one day and 12 hours, "`PT30M`" for 30 minutes.
locale	string [▶ 2108]	Language settings of the user. From version 1.10.1336.10 'client' the language of the browser or 'project' is valid for the project settings.
timeFormatLocale From version 1.10.1336.10	string [▶ 2108]	Language settings of the user for time/date representations. Here 'client' is valid for the language of the browser or 'project' for the project setting.
timeZone From version 1.10.1336.10	string [▶ 2108]	Time zone of the user for time/date representations. Here 'client' is valid for the language of the browser or 'project' for the project setting.
addGroups	string [▶ 2108][]	Groups in which the user is to become a new member.
removeGroups	string [▶ 2108][]	Groups in which the user is no longer to be a member.

Note: Available since version 1.10.1018.48

15.1.1.16.23 IUsernameListResultObject .18

[Interface]

```
export interface IUsernameListResultObject extends TcHmi.IResultObject {
    userList: string[];
}
```

Saves a list of user names.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
userList	string [▶ 2108][]	List of users registered in the server.

Note: Available since version 1.10.1018.48

15.1.1.16.23 IUserResultObject .19

[Interface]

```
export interface IUserListResultObject extends TcHmi.IResultObject {
    userDetails?: Dictionary<TcHmi.Server.UserManagement.IUserDetails>;
}
```

Result of a ListUsers call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
userDetails [optional]	Dictionary<TcHmi.Server.UserManagement.IUserDetails [▶ 1951]>	List of users registered in the server, including all metadata.

Note: Available since version 1.10.1018.48

15.1.1.16.23 IGroupResultObject .20

[Interface]

```
export interface IGroupResultObject extends TcHmi.IResultObject {
    userDetails?: Dictionary<TcHmi.Server.UserManagement.IGroupDetails>;
}
```

Result of a ListUserGroups call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
groupDetailsList [optional]	Dictionary<TcHmi.Server.UserManagement.IGroupDetails [▶ 1952]>	List of groups registered in the server, including all metadata.

Note: Available since version 1.10.1171.21

15.1.1.16.23 IUserManagementOptions .21

[Interface]

```
export interface IUserManagementOptions {
    domain: string;
};
```

UserManagement options

Properties

Name	Type	Description
domain	string [▶ 2108]	User management domain. 'TcHmiUserManagement' if not specified

Note: Available since version 1.10.1018.48

15.1.1.16.24 RecipeManagement

[Class]

Enables reading and modifying of recipes

Static functions

listRecipeTypes [▶ 1960]	Supplies a list of all existing system recipes.
listRecipeTypesEx [▶ 1960]	Supplies a list of all existing system recipes.
createRecipeTypeFolder [▶ 1961]	Creates a new folder for recipe types.
createRecipeTypeFolderEx [▶ 1962]	Creates a new folder for recipe types.
renameRecipeTypeFolder [▶ 1963]	Renames a recipe type folder.
renameRecipeTypeFolderEx [▶ 1963]	Renames a recipe type folder.
deleteRecipeTypeFolder [▶ 1965]	Deletes a folder for recipe types.
deleteRecipeTypeFolderEx [▶ 1965]	Deletes a folder for recipe types.
createRecipeType [▶ 1966]	Creates a new recipe type.
createRecipeTypeEx [▶ 1968]	Creates a new recipe type.
deleteRecipeType [▶ 1971]	Creates a new recipe type.
deleteRecipeTypeEx [▶ 1959]	Creates a new recipe type.
renameRecipeType [▶ 1969]	Renames a recipe type.
renameRecipeTypeEx [▶ 1970]	Renames a recipe type.
deleteRecipeType [▶ 1971]	Deletes a recipe type.
deleteRecipeTypeEx [▶ 1959]	Deletes a recipe type.
listRecipes [▶ 1974]	Supplies a list of all existing system recipes.
listRecipesEx [▶ 1975]	Supplies a list of all existing system recipes.
createRecipeFolder [▶ 1976]	Creates a new folder for recipes.
createRecipeFolderEx [▶ 1976]	Creates a new folder for recipes.
renameRecipeFolder [▶ 1977]	Renames a recipe folder.
renameRecipeFolderEx [▶ 1978]	Renames a recipe folder.
deleteRecipeFolder [▶ 1979]	Deletes a folder for recipes.
deleteRecipeFolderEx [▶ 1980]	Deletes a folder for recipes.
createRecipe [▶ 1981]	Creates a new recipe type.
createRecipeEx [▶ 1982]	Creates a new recipe type.
getRecipe [▶ 1983]	Supplies the definition of a recipe.
getRecipeEx [▶ 1984]	Supplies the definition of a recipe.
updateRecipe [▶ 1985]	Updates the definition of a recipe.
updateRecipeEx [▶ 1986]	Updates the definition of a recipe.
readFromTarget [▶ 1987]	Reads the current values from the target system.
readFromTargetEx [▶ 1987]	Reads the current values from the target system.
teach [▶ 1988]	Reads the current values from the target system and writes them into the specified recipe.
teachEx [▶ 1989]	Reads the current values from the target system and writes them into the specified recipe.
teachAsNewRecipe [▶ 1990]	Reads the current values from the target system and writes them into a new recipe.
teachAsNewRecipeEx [▶ 1991]	Reads the current values from the target system and writes them into a new recipe.
activate [▶ 1992]	Writes all values of a recipe into the target system, thus activating it.
activateEx [▶ 1993]	Writes all values of a recipe into the target system, thus activating it.
getActiveRecipes [▶ 1994]	Supplies a list of all active recipes.
getActiveRecipesEx [▶ 1995]	Supplies a list of all active recipes.

renameRecipe [▶ 1996]	Renames a recipe.
renameRecipeEx [▶ 1997]	Renames a recipe.
deleteRecipe [▶ 1998]	Deletes a recipe.
deleteRecipeEx [▶ 1998]	Deletes a recipe.

Interfaces

IRecipeOptions [▶ 1972]	Recipe management options
IRecipeTypeListResultObject [▶ 1972]	Result of a listRecipeTypes call.
FolderRecipeType [▶ 1973]	Definition of a recipe type folder.
RecipeType [▶ 1973]	Definition of a recipe type.
IRecipeListResultObject [▶ 1999]	Result of a listRecipe call.
FolderRecipe [▶ 2000]	Definition of a recipe folder.
Recipe [▶ 2000]	Definition of a recipe type.
IGetRecipeResultObject [▶ 2000]	Result of a getRecipe call.
IReadFromTargetResultObject [▶ 2000]	Result of a ReadFromTarget call.
IGetActiveRecipesResultObject [▶ 2001]	Saves a list of recipes.

Note: Available since version 1.10.1171.21

15.1.1.16.24 deleteRecipeTypeEx .1

[Function]

```
public static deleteRecipeTypeEx (
    recipeTypeName: string,
    path: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108]	Path of the destination folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeTypeEx (
    'recipeType10',
    'testRecipeCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 listRecipeTypes .2

[Function]

```
public static listRecipeTypesEx (
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IRecipeTypeListResultObject)
=> void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system recipes.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.Server.RecipeManagement.IRecipeTypeListResultObject [▶ 1972]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.listRecipeTypes (
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 listRecipeTypesEx .3

[Function]

```
public static listRecipeTypesEx (
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IRecipeTypeListResultObject)
=> void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system recipes.

Parameter

Name	Type	Description
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.RecipeManagement.IRecipeTypeListResultObject [▶ 1972]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.listRecipeTypesEx (
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipeTypeFolder .4

[Function]

```
public static createRecipeTypeFolder (
    path: string,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new folder for recipe types.

Parameter

Name	Type	Description
path	string [▶ 2108]	Path of the new folder
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.createRecipeTypeFolder (
    'myTestfolder',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipeTypeFolderEx .5

[Function]

```
public static createRecipeTypeFolderEx (
    path: string,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new folder for recipe types.

Parameter

Name	Type	Description
path	string	Path of the new folder
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.createRecipeTypeFolderEx (
    'myTestfolder',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeTypeFolder .6

[Function]

```
public static renameRecipeTypeFolder (
    recipeTypeFolderName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe type folder.

Parameter

Name	Type	Description
recipeTypeFolderName	string [▶ 2108]	Name of the recipe type folder. The name may include the full path. In this case, '::' must be used as the separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
newName	string [▶ 2108]	New name of the recipe type. The name may include the full path. In this case, '::' must be used as the separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1336.60

Sample - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeTypeFolder (
    'recipeTypeFolder10',
    'testRecipeCategory',
    'rootRecipeTypeFolder11', // new name
    null, // new path not give so the recipe type folder is moved to root folder
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeTypeFolderEx .7

[Function]

```
public static renameRecipeTypeFolderEx (
    recipeTypeFolderName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe type folder.

Parameter

Name	Type	Description
recipeTypeFolderName	string [▶ 2108]	Name of the recipe type folder. The name may include the full path. In this case, '::' must be used as the separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the system searches for the recipe type in the main folder.
newName	string [▶ 2108]	New name of the recipe type. The name may include the full path. In this case, '::' must be used as the separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1336.60

Sample - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeTypeFolderEx (
    'recipeTypeFolder10',
    'testRecipeCategory',
    'rootRecipeTypeFolder11', // new name
    null, // new path not give so the recipe type folder is moved to root folder
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 deleteRecipeTypeFolder .8

[Function]

```
public static deleteRecipeTypeFolder (
    path: string,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a folder for recipe types.

Parameter

Name	Type	Description
path	string [▶ 2108]	Path of the folder
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeTypeFolder (
    'myTestfolder',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 deleteRecipeTypeFolderEx .9

[Function]

```
public static deleteRecipeTypeFolderEx (
    path: string,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a folder for recipe types.

Parameter

Name	Type	Description
path	string [▶ 2108]	Path of the folder
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeTypeFolderEx (
    'myTestfolder',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipeType .10

[Function]

```
public static createRecipeType (
    recipeTypeName: string,
    recipeType: TcHmi.Server.RecipeManagement.RecipeType,
    path: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
recipeType	TcHmi.Server.RecipeManagement.RecipeType [▶ 1973]	Recipe type definition
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
let recipeType = {
    'recipeTypeNames': ['recipeType1', 'recipeType10'],
    'members':
    {
        'INGREDIENT': {
            symbol: 'PLC1.MAIN.test11',
            defaultValue: true
        },
        'test10': {
            symbol: 'PLC1.MAIN.test10',
            defaultValue: true
        }
    },
    'options': {
        'enabled': 'None',
        'comment': ''
    }
};
TcHmi.Server.RecipeManagement.createRecipeType(
    'recipeType10',
    recipeType,
    'testRecipeCategory',
    function (data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);

// or the same:
TcHmi.Server.RecipeManagement.createRecipeType(
    'testRecipeCategory::recipeType10',
    recipeType,
    null, // or '',
    function (data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipeTypeEx .11

[Function]

```
public static createRecipeTypeEx (
    recipeTypeName: string,
    recipeType: TcHmi.Server.RecipeManagement.RecipeType,
    path: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
recipeType	TcHmi.Server.RecipeManagement.RecipeType [▶ 1973]	Recipe type definition
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
let recipeType = {
    'recipeTypeNames': ['recipeType1', 'recipeType10'],
    'members':
    {
        'INGREDIENT': {
            symbol: 'PLC1.MAIN.test11',
            defaultValue: true
        },
        'test10': {
            symbol: 'PLC1.MAIN.test10',
            defaultValue: true
        }
    },
    'options': {
        'enabled': 'None',
        'comment': ''
    }
};
TcHmi.Server.RecipeManagement.createRecipeTypeEx(
    'recipeType10',
    recipeType,
```

```
    'testRecipeCategory',
    null,
    { timeout: 2000 },
    function (data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);

// or the same:
TcHmi.Server.RecipeManagement.createRecipeTypeEx(
    'testRecipeCategory::recipeType10',
    recipeType,
    null, // or ''
    null,
    { timeout: 2000 },
    function (data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeType .12

[Function]

```
public static renameRecipeType (
    recipeTypeName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
newName	string [▶ 2108]	New name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeType (
    'recipeType10',
    'testRecipeCategory',
    'rootRecipeType11', // new name
    null, // new path not give so the recipe type is moved to root folder
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeTypeEx .13

[Function]

```
public static renameRecipeTypeEx (
    recipeTypeName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the system searches for the recipe type in the main folder.
newName	string [▶ 2108]	New name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeTypeEx (
    'recipeType10',
    'testRecipeCategory',
    'rootRecipeType11', // new name
    null, // new path not give so the recipe type is moved to root folder
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 deleteRecipeType .14

[Function]

```
public static deleteRecipeType (
    recipeTypeName: string,
    path: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108]	Path of the destination folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeType (
    'recipeType10',
    'testRecipeCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 IRecipeOptions .15

[Interface]

```
export interface IRecipeOptions {
    domain: string;
};
```

Recipe management options

Properties

Name	Type	Description
domain	string [▶ 2108]	Domain of the recipe management. 'TcHmiRecipeManagement' if not specified

Note: Available since version 1.10.1171.142

15.1.1.16.24 IRecipeTypeListResultObject .16

[Interface]

```
export interface IRecipeTypeListResultObject extends TcHmi.IResultObject {
    value?: TcHmi.Server.RecipeManagement.FolderRecipeType;
}
```

Result of a listRecipeTypes call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
value [optional]	TcHmi.Server.RecipeManagement.FolderRecipeType [▶ 1973]	List of recipe types registered in the server.

Note: Available since version 1.10.1171.142

15.1.1.16.24 FolderRecipeType .17

[Interface]

```
export type FolderRecipeType = object | RecipeType;
```

Definition of a recipe type folder.

This entry can therefore describe a folder [▶ 2108] or a recipe type [▶ 1973].

Note: Available since version 1.10.1171.142

15.1.1.16.24 RecipeType .18

[Interface]

```
export interface RecipeType {
    recipeTypeNames?: string[];
    members: Dictionary<{
        symbol: string;
        defaultValue?: any;
        schema?: string;
        unit?: string;
        comment?: string;
        enabled?: boolean;
        group?: string;
        order?: number;
    } | {
        recipeType: string;
    }>;
    options?: {
        enabled?: 'None' | 'Disabled';
        comment?: string;
    };
}
```

Definition of a recipe type.

Properties

Name	Type	Description
recipeTypeNames [optional]	string [▶ 2108][]	List of recipes from which this recipe type inherits.
members		List of symbols that define the recipe type
options [optional]		Recipe type options
options.enabled [optional]	'None' | 'Disabled'	Specifies whether the recipe type is active. `None`: No restriction at recipe type level. `Disabled`: All members of the recipe are deactivated.
options.comment [optional]	string [▶ 2108]	Free text comment

Properties of the symbol definition

Name	Type	Description
symbol [optional]	string [▶ 2108][]	Name of the symbol
defaultValue [optional]	any [▶ 2107]	Value that applies if a recipe does not define its own value
schema [optional]	string [▶ 2108]	Reference to a schema that defines value restrictions. (For example minimum or maximum value)
unit [optional]	string [▶ 2108]	Physical unit of the value
comment [optional]	string [▶ 2108]	Free text comment
enabled [optional]	boolean [▶ 2107]	Indicates whether the symbol is active in the recipe type
group [optional]	string [▶ 2108]	Group for the display in Visual Studio
order [optional]	number [▶ 2108]	Order for the display in Visual Studio

Note: Available since version 1.10.1171.142

15.1.1.16.24 listRecipes .19

[Function]

```
public static listRecipes (
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IRecipeListResultObject) =>
void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system recipes.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.Server.RecipeManagement.IRecipeListResultObject [▶ 1999]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.listRecipes (
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 listRecipesEx .20

[Function]

```
public static listRecipesEx (
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IRecipeListResultObject) =>
void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all existing system recipes.

Parameter

Name	Type	Description
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.RecipeManagement.IRecipeListResultObject [▶ 1999]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.listRecipesEx (
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
```

```
        }
    }
);
```

15.1.1.16.24 createRecipeFolder .21

[Function]

```
public static createRecipeFolder (
    path: string,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new folder for recipes.

Parameter

Name	Type	Description
path	string	Path of the new folder
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.createRecipeFolder (
    'myTestfolder',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipeFolderEx .22

[Function]

```
public static createRecipeFolderEx (
    path: string,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new folder for recipes.

Parameter

Name	Type	Description
path	string	Path of the new folder
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.createRecipeFolderEx (
    'myTestfolder',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeFolder .23

[Function]

```
public static renameRecipeFolder (
    recipeFolderName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe folder.

Parameter

Name	Type	Description
recipeFolderName	string [▶ 2108]	Name of the recipe folder. The name may include the full path. In this case, '::' must be used as the separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the system searches for the recipe type in the main folder.
newName	string [▶ 2108]	Name of the new recipe. The name may include the full path. In this case, '::' must be used as the separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1336.60

Sample - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeFolder (
    'recipeFolder10',
    'testRecipeCategory',
    'rootRecipeFolder11', // new name
    null, // new path not give so the recipe Folder is moved to root folder
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeFolderEx .24

[Function]

```
public static renameRecipe FolderEx (
    recipe FolderName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe folder.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe folder. The name may include the full path. In this case, '::' must be used as the separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the system searches for the recipe type in the main folder.
newName	string [▶ 2108]	Name of the new recipe. The name may include the full path. In this case, '::' must be used as the separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1336.60

Sample - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeFolderEx (
    'recipeFolder10',
    'testRecipeCategory',
    'rootRecipeFolder11', // new name
    null, // new path not give so the recipe Folder is moved to root folder
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 deleteRecipeFolder .25

[Function]

```
public static deleteRecipeFolder (
    path: string,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a folder for recipes.

Parameter

Name	Type	Description
path	string	Path of the folder
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeFolder (
    'myTestfolder',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 deleteRecipeFolderEx .26

[Function]

```
public static deleteRecipeFolderEx (
    path: string,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a folder for recipes.

Parameter

Name	Type	Description
path	string	Path of the folder
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeFolderEx (
    'myTestfolder',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipe .27

[Function]

```
public static createRecipe (
    recipeName: string,
    recipe: TcHmi.Server.RecipeManagement.Recipe,
    path: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
recipeType	TcHmi.Server.RecipeManagement.Recipe [▶ 2000]	Recipe definition
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
let recipe = {
    'recipeTypeName': 'cookieRecipeType',
    'values': {
        'PLC1.MAIN.sugar': 8
    }
};
TcHmi.Server.RecipeManagement.createRecipe(
    'recipe10',
    'testRecipeCategory',
    recipe,
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
```

```
            // Error
        }
    }
);

// or the same:
TcHmi.Server.RecipeManagement.createRecipe(
    'testRecipeCategory::recipe10',
    null, // or ''
    recipe,
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 createRecipeEx .28

[Function]

```
public static createRecipeEx (
    recipeName: string,
    recipe: TcHmi.Server.RecipeManagement.Recipe,
    path: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Creates a new recipe type.

Parameter

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type. The name may include the full path. In this case, '::' must be used as separator for folders.
recipeType	TcHmi.Server.RecipeManagement.Recipe [▶ 2000]	Recipe definition
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
let recipe = {
    'recipeTypeName': 'cookieRecipeType',
    'values': {
        'PLC1.MAIN.sugar': 8
    }
};
TcHmi.Server.RecipeManagement.createRecipeEx(
    'recipe10',
    'testRecipeCategory',
    recipe,
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);

// or the same:
TcHmi.Server.RecipeManagement.createRecipeEx(
    'testRecipeCategory::recipe10',
    null, // or ''
    recipe,
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 getRecipe .29

[Function]

```
public static getRecipe (
    recipeName: string,
    path: string | null,
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IRecipeListResultObject) =>
void) = null
): TcHmi.IErrorDetails;
```

Supplies the definition of a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.Server.RecipeManagement.IGetRecipeResultObject [▶ 2000]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.getRecipe (
    'recipe10',
    'testRecipeCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 getRecipeEx .30

[Function]

```
public static getRecipeEx (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IRecipeListResultObject) =>
void) = null
): TcHmi.IErrorDetails;
```

Supplies the definition of a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.RecipeManagement.IGetRecipeResultObject [▶ 2000]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.getRecipeEx (
    'recipe10',
    'testRecipeCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 updateRecipe .31

[Function]

```
public static updateRecipe (
    recipeName: string,
    path: string | null,
    newValues: Dictionary<any>,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Updates the definition of a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
newValues	Dictionary<any [▶ 2107]>	New values for the recipe
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.updateRecipe (
    'recipe10',
    'testRecipeCategory',
    {
        'PLC1.MAIN.sugar': 23
    },
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 updateRecipeEx .32

[Function]

```
public static updateRecipeEx (
    recipeName: string,
    path: string | null,
    newValues: Dictionary<any>,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Updates the definition of a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
newValues	Dictionary<any [▶ 2107]>	New values for the recipe
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.updateRecipeEx (
    'recipe10',
    'testRecipeCategory',
    {
        'PLC1.MAIN.sugar': 23
    }
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 readFromTarget .33

[Function]

```
public static readFromTarget (
    recipeName: string,
    path: string | null,
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IReadFromTargetResultObject)
=> void) = null
): TcHmi.IErrorDetails;
```

Reads the current values from the target system. The symbols defined in a recipe are read.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.Server.RecipeManagement.IReadFromTargetResultObject [▶ 2000]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.readFromTarget (
    'recipe10',
    'testRecipeCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 readFromTargetEx .34

[Function]

```
public static readFromTargetEx (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.Server.RecipeManagement.IReadFromTargetResultObject)
=> void) = null
): TcHmi.IErrorDetails;
```

Reads the current values from the target system. The symbols defined in a recipe are read.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.RecipeManagement.IReadFromTargetResultObject [▶ 2000]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.readFromTargetEx (
    'recipe10',
    'testRecipeCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.value);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 teach .35

[Function]

```
public static teach (
    recipeName: string,
    path: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Reads the current values from the target system and writes them into the specified recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.teach (
    'recipe10',
    'testRecipeCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 teachEx .36

[Function]

```
public static teachEx (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Reads the current values from the target system and writes them into the specified recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.teachEx (
    'recipe10',
    'testRecipeCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 teachAsNewRecipe .37

[Function]

```
public static teachAsNewRecipe (
    recipeName: string,
    path: string | null,
    newRecipeName: string,
    newRecipePath: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Reads the current values from the target system (referenced by a recipe) and writes them into a new recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
newRecipeName	string [▶ 2108]	Name of the new recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
newPath	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.teachAsNewRecipe (
    'recipe10',
    'testRecipeCategory',
    'recipe11',
    'newRecipesCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 teachAsNewRecipeEx .38

[Function]

```
public static teachAsNewRecipeEx (
    recipeName: string,
    path: string | null,
    newRecipeName: string,
    newRecipePath: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Reads the current values from the target system (referenced by a recipe) and writes them into a new recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
newRecipeName	string [▶ 2108]	Name of the new recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
newPath	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.teachAsNewRecipeEx (
    'recipe10',
    'testRecipeCategory',
    'recipe11',
    'newRecipesCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 activate .39

[Function]

```
public static activate (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Writes all values of a recipe into the target system, thus activating it.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.activate (
    'recipe10',
    'testRecipeCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 activateEx .40

[Function]

```
public static activateEx (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Writes all values of a recipe into the target system, thus activating it.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.activateEx (
    'recipe10',
    'testRecipeCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 getActiveRecipes .41

[Function]

```
public static getActiveRecipes (
    callback: null | ((this: void, data:
TcHmi.Server.RecipeManagement.IGetActiveRecipesResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all active recipes.

Parameter

Name	Type	Description
callback [optional]	(data: TcHmi.Server.RecipeManagement.IGetActiveRecipesResultObject [▶ 2001]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.getActiveRecipes (
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.recipeList);
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 getActiveRecipesEx .42

[Function]

```
public static getActiveRecipesEx (
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data:
TcHmi.Server.RecipeManagement.IGetActiveRecipesResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Supplies a list of all active recipes.

Parameter

Name	Type	Description
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.Server.RecipeManagement.IGetActiveRecipesResultObject [▶ 2001]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.getActiveRecipesEx (
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            console.log(data.recipeList);
        } else {
            // Error
```

```
        }
    }
);
```

15.1.1.16.24 renameRecipe .43

[Function]

```
public static renameRecipe (
    recipeName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the system searches for the recipe type in the main folder.
newName	string [▶ 2108]	Name of the new recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipe (
    'recipe10',
    'testRecipeCategory',
    'rootRecipe11', // new name
    null, // new path not give so the recipe is moved to root folder
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 renameRecipeEx .44

[Function]

```
public static renameRecipeEx (
    recipeName: string,
    path: string | null,
    newName: string,
    newPath: string | null,
    options?: TcHmi.Server.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Renames a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108]	Path of the destination folder. If this is not specified, the system searches for the recipe type in the main folder.
newName	string [▶ 2108]	Name of the new recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
newPath	string [▶ 2108]	Path of the new destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.renameRecipeEx (
    'recipe10',
    'testRecipeCategory',
    'rootRecipe11', // new name
    null, // new path not give so the recipe is moved to root folder
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
```

```
        }
    }
);
```

15.1.1.16.24 deleteRecipe .45

[Function]

```
public static deleteRecipe (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipe (
    'recipe10',
    'testRecipeCategory',
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 deleteRecipeEx .46

[Function]

```
public static deleteRecipeEx (
    recipeName: string,
    path: string | null,
    options?: TcHmi.RecipeManagement.IRecipeOptions | null,
    requestOptions: TcHmi.Server.IRequestOptions | null = null,
    callback: null | ((this: void, data: TcHmi.IResultObject) => void) = null
): TcHmi.IErrorDetails;
```

Deletes a recipe.

Parameter

Name	Type	Description
recipeName	string [▶ 2108]	Name of the recipe. The name may include the full path. In this case, '::' must be used as separator for folders.
path	string [▶ 2108], null [▶ 2108]	Path of the destination folder. If this is not specified, the recipe type is created in the main folder.
options	TcHmi.Server.RecipeManagement.IRecipeOptions [▶ 1972], null [▶ 2108]	Recipe management options
requestOptions	TcHmi.Server.IRequestOptions [▶ 2006], null [▶ 2108]	Server request options
callback [optional]	(data: TcHmi.IResultObject [▶ 2094]) => void, null [▶ 2108]	Asynchronous callback function that is triggered once the action is completed.

Return value

Type	Description
TcHmi.IErrorDetails [▶ 2094]	Returns confirmation as to whether the operation was successfully sent.

Note: Available since version 1.10.1171.142

Example - JavaScript

```
TcHmi.Server.RecipeManagement.deleteRecipeEx (
    'recipe10',
    'testRecipeCategory',
    null,
    {timeout: 2000},
    function(data) {
        if (data.error === TcHmi.Errors.NONE) {
            // Success
        } else {
            // Error
        }
    }
);
```

15.1.1.16.24 IRecipeListResultObject .47

[Interface]

```
export interface IRecipeListResultObject extends TcHmi.IResultObject {
    value?: TcHmi.Server.RecipeManagement.FolderRecipe;
}
```

Result of a listRecipe [▶ 1974] call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
value [optional]	TcHmi.Server.RecipeManagement.FolderRecipe [▶ 2000]	List of recipes registered in the server.

Note: Available since version 1.10.1171.142

15.1.1.16.24 FolderRecipe .48

[Interface]

```
export type FolderRecipe = object | Recipe;
```

Definition of a recipe folder.

This entry can therefore describe a folder [▶ 2108] or a recipe [▶ 2000].

Note: Available since version 1.10.1171.142

15.1.1.16.24 Recipe .49

[Interface]

```
export interface Recipe {
    recipeTypeName: string;
    values: Dictionary<any>;
}
```

Definition of a recipe type.

Properties

Name	Type	Description
recipeTypeName	string [▶ 2108]	Name of the recipe type for this recipe.
values	Dictionary<any [▶ 2107]>	Values for this recipe.

Note: Available since version 1.10.1171.142

15.1.1.16.24 IGetRecipeResultObject .50

[Interface]

```
export interface IGetRecipeResultObject extends TcHmi.IResultObject {
    value?: TcHmi.Server.RecipeManagement.Recipe;
}
```

Result of a getRecipe [▶ 1983] call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
value [optional]	TcHmi.Server.RecipeManagement.Recipe [▶ 2000]	Definition of the recipe.

Note: Available since version 1.10.1171.142

15.1.1.16.24 IReadFromTargetResultObject .51

[Interface]

```
export interface IReadFromTargetResultObject extends TcHmi.IResultObject {
    value?: Dictionary<TcHmi.Server.RecipeManagement.Recipe>
}
```

Result of a ReadFromTarget [▶ 1987] call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
value [optional]	Dictionary<TcHmi.Server.RecipeManagement.Recipe [▶ 2000]>	List of the target values. For example, the dictionary key is: 'subFolder::myRecipe'

Note: Available since version 1.10.1171.142

15.1.1.16.24 IGetActiveRecipesResultObject .52

[Interface]

```
export interface IGetActiveRecipesResultObject extends TcHmi.IResultObject {
    recipeList: string[];
}
```

Saves a list of recipes and is the result of a GetActiveRecipes [▶ 1994] call.

This interface extends TcHmi.IResultObject [▶ 2094].

Properties

Name	Type	Description
recipeList	string [▶ 2108][]	List of recipes.

Note: Available since version 1.10.1171.142

15.1.1.16.25 IResultObject

[Interface]

```
export interface IResultObject extends TcHmi.IResultObject{
    response: IMessage;
};
```

This interface extends TcHmi.IResultObject [▶ 2094]. The server provides feedback on the success of all requests. You should check whether all requests were successful. See examples at request [▶ 1907].

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional] **Note:** Available since version 1.10	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
response	IMessage [▶ 2004]	Net content of the request itself

Note: Available since version 1.8

15.1.1.16.26 ICommand

[Interface]

```
export interface ICommand {
    symbol: string;
    customerData?: string;
    readValue?: any;
    writeValue?: any;
    error?: TcHmi.IErrorDetails;
    commandOptions?: TcHmi.Server.ICommandOptions[];
};
```

Properties

Name	Type	Description
symbol	string [▶ 2108]	Symbol to be edited
customerData [optional]	string [▶ 2108]	Additional information that is applied from the query to the response
readValue [optional]	any [▶ 2107]	Value that was read
writeValue [optional]	any [▶ 2107]	Value to be written
error [optional]	TcHmi.IErrorDetails [▶ 2094]	Error code of the command
commandOptions [optional]	TcHmi.Server.ICommandOptions [▶ 2003][]	Command parameters

Note: Available since version 1.8

Note: From version 1.10.1018.48 the property "error" is of type "TcHmi.IErrorDetails [▶ 2094]" instead of "TcHmi.Server.ICommandError [▶ 2002]".

15.1.1.16.27 ICommandError

[Interface]

```
export interface ICommandError {
    code: number;
    message?: string;
    errors?: {
        code: number;
        domain?: string;
        message?: string;
        position?: string;
    }[];
};
```

Properties

Name	Type	Description
code	number [▶ 2108]	Error code of the entire command
message [optional]	string [▶ 2108]	Plain text description of the error
errors	See below	Array with more detailed error information

Properties of the error array

Name	Type	Description
code	number [▶ 2108]	Error code of the individual command
domain [optional]	string [▶ 2108]	Origin of the individual error
message [optional]	string [▶ 2108]	Plain text description of the individual error
position [optional]	string [▶ 2108]	Additional information about the individual error

Note: Available since version 1.8

Note: Redundant from version 1.10.1018.48. Replaced by TcHmi.IErrorDetails [▶ 2094].

15.1.1.16.28 ICommandOptions

[Interface]

```
export type ICommandOptions = 'Config' | 'SendWriteValue' | 'Add' | 'Replace' | 'Delete' |
'SendErrorMessage' | 'Poll' | 'Offline' | 'UniqueHash' | 'ForceSingleWrite' | 'Extension1' |
'Extension2' | 'Extension3' | 'Extension4';
```

Keywords as strings.

Note: Available since version 1.8

`'Delete'` is available from version 1.10.1060.0

15.1.1.16.29 IError

[Interface]

```
export interface IError {
    code: number;
    domain?: string;
    message?: string;
    position?: string;
    errors?: TcHmi.Server.IDomainError[];
};
```

Properties

Name	Type	Description
code	number [▶ 2108]	
domain [optional]	string [▶ 2108]	
message [optional]	string [▶ 2108]	
position [optional]	string [▶ 2108]	
errors [optional]	TcHmi.Server.IDomainError [▶ 2003][]	

Note: Available since version 1.8

Note: Redundant from version 1.10.1018.48. Replaced by TcHmi.IErrorDetails [▶ 2094].

15.1.1.16.30 IDomainError

[Interface]

```
export interface IDomainError {
    code: number;
    domain?: string;
```

```
    message?: string;
    position?: string;
};
```

Properties

Name	Type	Description
code	number [▶ 2108]	
domain [optional]	string [▶ 2108]	
message [optional]	string [▶ 2108]	
position [optional]	string [▶ 2108]	

Note: Available since version 1.8

Note: Redundant from version 1.10.1018.48. Replaced by TcHmi.IErrorDetails [▶ 2094].

15.1.1.16.31 IMessage

[Interface]

```
export interface IMessage {
    apiVersion?: string;
    requestType: TcHmi.Server.IRequestType;
    id?: number;
    intervalTime?: number | null;
    sessionId?: string;
    error?: TcHmi.IErrorDetails;
    customerData?: string;
    commands?: TcHmi.Server.ICommand[];
};
```

Properties

Name	Type	Description
apiVersion [optional]	string [▶ 2108]	Version number of the API
requestType	TcHmi.Server.IRequestType [▶ 2004]	Type of access
id [Optional]	number [▶ 2108]	Unique message number
intervalTime [optional]	number [▶ 2108], null [▶ 2108]	Minimum time in which a subscription update (subscription tick) is sent.
sessionId [optional]	string [▶ 2108]	sessionID of the message
error [optional]	TcHmi.IErrorDetails [▶ 2094]	Error code of the message
customerData [optional]	string [▶ 2108]	Additional information that is applied from the query to the response
commands [Optional]	TcHmi.Server.ICommand [▶ 2002][]	List of commands for this message

Note: Available since version 1.8

Note: From version 1.10.1018.48 the property "error" is of type "TcHmi.IErrorDetails [▶ 2094]" instead of "TcHmi.Server.IError [▶ 2003]".

15.1.1.16.32 IRequestType

[Interface]

```
export type IRequestType = 'ReadWrite' | 'Subscription' | 'Event';
```

Note: Available since version 1.8

15.1.1.16.33 userConfigOnServer

[Interface]

```
export interface userConfigOnServer {
    state: TcHmi.Server.userConfigState;
    userIsInGroups: string[];
    name: string | null;
    domain: string | null;
    locale: string | undefined;
    timeFormatLocale: string | undefined;
    timeZone: string | undefined;
    clientIp: string;
    session: string | null;
    clientCertificate: string | null;
    autoLogOffMilliSeconds: number | null;
    defaultAuthExtension: string;
}
```

Properties

Name	Type	Description
state	TcHmi.Server.userConfigState [▶ 2006]	State of the user information in the browser and on the server
userIsInGroups	string [▶ 2108][]	Array of groups in which the current user is a member.
name	string [▶ 2108], null [▶ 2108]	Name of the current user or, if no user information is available: null.
domain	string [▶ 2108], null [▶ 2108]	Source of user information in the server or, if no user information is available: null.
locale	string [▶ 2108], undefined [▶ 2108]	Language settings of the user. From version 1.10.1336.10 undefined stands for the setting of the browser.
timeFormatLocale From version 1.10.1336.10	string [▶ 2108], undefined [▶ 2108]	Language settings of the user for time/date representations. undefined stands for the setting of the browser.
timeZone From version 1.10.1336.10	string [▶ 2108], undefined [▶ 2108]	Time zone of the user for time/date representations. undefined stands for the setting of the browser.
clientIp From version 1.10.1336.10	string [▶ 2108]	IP of the client as it is visible from the server.
session	string [▶ 2108], null [▶ 2108]	SessionId of the communication with the server or, if no user information is available: null.
clientCertificate From version 1.10.1336.10	string [▶ 2108], null [▶ 2108]	Fingerprint of a client certificate or null
autoLogOffMilliSeconds	number [▶ 2108], null [▶ 2108]	Time after which the current user is to be logged out without interaction.
defaultAuthExtension From version 1.10.1336.10	string [▶ 2108]	Name of the authentication extension used by default

Note: Available since version 1.8

See also

TcHmi.Localization [▶ 1894]

15.1.1.16.34 userConfigState

[Enumeration]

```
export const enum userConfigState {
    loading,
    loadingerror,
    noAuthRequired,
    communicationDisabled,
    usergroup
};
```

The names of the enumeration no longer exist at runtime.

Values

Name	Value	Description
loading	0	The user information is currently loaded from the server
loadingerror	1	User information is not available due to an error
noAuthRequired	2	The server does not require authentication.
communicationDisabled	3	The server is an engineering server without active user management
usergroup	4	User and group information is available

Note: Available since version 1.8

15.1.1.16.35 IForceLogoutTarget

[Interface]

```
export interface IForceLogoutTarget {
    clientIp?: string;
    clientCertificate?: string;
    group?: string;
}
```

Condition of a ForceLogout [▶ 1917] call

Properties

Name	Type	Description
clientIp	string [▶ 2108]	IP address of the client
clientCertificate	string [▶ 2108]	Fingerprint of the client certificate
group	string [▶ 2108]	Group membership

Note: Available since version 1.10.1172.0

15.1.1.16.36 IRequestOptions

[Interface]

```
export interface IRequestOptions {
    timeout?: number;
    parallel?: boolean;
}
```

Properties

Name	Type	Description
timeout [optional]	number [▶ 2108]	
parallel [optional]	boolean [▶ 2107]	

Note: Available since version 1.10

15.1.1.16.37 ACCESS

[Enumeration]

```
export enum ACCESS {
   NONE = 0,
   READ = 1,
   WRITE = 2,
   READWRITE = 3
}
```

The names of the enumeration exist at runtime. For example, you can use `TcHmi.Server.ACCESS.NONE` directly in a comparison.

Values

Name	Value	Description
NONE	0	No access right
READ	1	Only reading allowed
WRITE	2	Only writing allowed
READWRITE	3	Reading and writing allowed

Note: Available since version 1.10.1171.21

15.1.1.17 StyleProvider

[Class]

Provides functions for manipulating HTML elements in controls.

Static functions

Name	Description
getSimpleElementStyle [▶ 2010]	Returns all set CSS properties of an HTML element.
getSimpleElementStyle [▶ 2010]	Returns several set CSS properties of an HTML element
getSimpleElementStyle [▶ 2011]	Returns a set CSS property of an HTML element.
getComputedElementStyle [▶ 2012]	Returns all active CSS properties of an HTML element.
getComputedElementStyle [▶ 2013]	Returns several active CSS properties of an HTML element
getComputedElementStyle [▶ 2014]	Returns an active CSS property of an HTML element.
setSimpleElementStyle [▶ 2016]	Sets a CSS property for one or more HTML elements.
setSimpleElementStyle [▶ 2017]	Sets multiple values for one or more HTML elements on a CSS property.
setSimpleElementStyle [▶ 2015]	Sets multiple CSS properties for one or more HTML elements.
processGenericStyle [▶ 2019]	Sets a CSS property for one or more dynamic HTML elements.
processGenericStyle [▶ 2018]	Sets multiple CSS properties for one or more dynamic HTML elements.
processBackground [▶ 2019]	Sets multiple background properties for one or more HTML elements.
processBackgroundColor [▶ 2020]	Sets a background color for one or more HTML elements.
processBackgroundImage [▶ 2021]	Sets a background image for one or more HTML elements.
processBorderColor [▶ 2022]	Sets a border color for one or more HTML elements.
processBorderWidth [▶ 2022]	Sets a border thickness for one or more HTML elements.
processBorderRadius [▶ 2023]	Sets a border radius for one or more HTML elements.
processBorderStyle [▶ 2024]	Sets a border style for one or more HTML elements.
processTextColor [▶ 2025]	Sets a text color for one or more HTML elements.
processFontFamily [▶ 2025]	Sets a font family for one or more HTML elements.
processFontSize [▶ 2026]	Sets a font size for one or more HTML elements.
processFontStyle [▶ 2027]	Sets a font style (normal, italic) for one or more HTML elements.
processFontWeight [▶ 2027]	Sets a font weight (normal, bold) for one or more HTML elements.
processFillColor [▶ 2028]	Sets the fill attribute for one or more SVG elements
processStrokeColor [▶ 2029]	Sets the stroke attribute for one or more SVG elements
processBoxShadow [▶ 2030]	Sets the CSS boxshadow property for one or more HTML elements
processVisibility [▶ 2030]	Sets the visibility for one or more HTML elements
processContentHorizontalAlignment [▶ 2031]	Sets the horizontal alignment of one or more HTML elements
processContentVerticalAlignment [▶ 2032]	Sets the vertical alignment of one or more HTML elements
processContentPadding [▶ 2032]	Sets the padding for one or more HTML elements
processTransform [▶ 2033]	Sets the CSS transform property for one or more HTML elements
resolveSolidColorAsCssValue [▶ 2034]	Resolves a SolidColor value into a CSS property.

Name	Description
resolveLinearGradientColorAsCssValue [▶ 2035]	Resolves a LinearGradientColor value into a CSS property.

Note: Available since version 1.8

15.1.1.17.1 getSimpleElementStyle

[Function]

```
public static getSimpleElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined
): Dictionary<string>;
```

Returns all set CSS properties of an HTML element. If the transferred object contains several elements (like for an jQuery collection), the first element is used.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the element to be verified • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.

Return value

Type	Description
Dictionary [▶ 2086]<string [▶ 2108]>	Object with all CSS properties of the first element in the jQuery collection. Returns an empty object in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton')
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
var allStyles = TcHmi.StyleProvider.getSimpleElementStyle(cntrlElem);
/* returns:
{
    'z-index': '0',
    'border-width': '0px',
    'height': '50px',
    'left': '164px',
    'top': '334px',
    'width': '100px'
}*/
TcHmi.Log.debug('Border Width is: '+ allStyles['border-width']);
```

15.1.1.17.2 getSimpleElementStyle

[Function]

```
public static getSimpleElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    propertyNames: string[]
): Dictionary<string>;
```

Returns several set CSS properties of an HTML element. If the transferred object contains several elements (like for an jQuery collection), the first element is used.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the element to be verified • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.
propertyNames	string [▶ 2108][]	String vector of the names of the CSS properties

Return value

Type	Description
Dictionary [▶ 2086]<string [▶ 2108]>	Object with all CSS properties of the first element in the jQuery collection. If a requested property is not (or no longer) set, this entry is not set in the object. Returns an empty object in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
var someStyles = TcHmi.StyleProvider.getSimpleElementStyle(cntrlElem,
     ['border-width', 'height', 'does-not-exist']
);
/* returns:
{
    'border-width': '0px',
    'height': '50px'
}
*/
TcHmi.Log.debug('Border Width is: '+ someStyles['border-width']);
```

15.1.1.17.3 getSimpleElementStyle

[Function]

```
public static getSimpleElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    propertyName: string
): Dictionary<string>;
```

Returns a set CSS property of an HTML element. If the transferred object contains several elements (like for an jQuery collection), the first element is used.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the element to be verified • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.
propertyName	string [▶ 2108]	Name of the CSS property

Return value

Type	Description
Dictionary [▶ 2086]<string [▶ 2108]>	Object with the CSS property of the first element in the jQuery collection. If a requested property is not (or no longer) set, this entry is not set in the object. Returns an empty object in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
var borderWidthStyle =
TcHmi.StyleProvider.getSimpleElementStyle(cntrlElem, 'border-width');
/* returns:
{
    'border-width': '0px',
}
*/
TcHmi.Log.debug('Border Width is: '+ borderWidthStyle['border-width']);
```

15.1.1.17.4 getComputedElementStyle

[Function]

```
public static getComputedElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined
): Dictionary<string>;
```

Returns all active CSS properties of an HTML element. This also takes into account properties that affect CSS files. If the transferred object contains several elements (like for an jQuery collection), the first element is used.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the element to be verified • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.

Return value

Type	Description
Dictionary [▶ 2086]<string [▶ 2108]>	Object with all CSS properties of the first element in the jQuery collection. Returns an empty object in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
var allStyles = TcHmi.StyleProvider.getComputedElementStyle(cntrlElem);
/* returns:
{
    'animation-delay': '0s',
    'animation-direction': 'normal',
    'animation-duration': '0s',
    'animation-fill-mode': 'none',
    … many many more
}*/
TcHmi.Log.debug('Border Width is: '+ allStyles['border-width']);
```

15.1.1.17.5 getComputedElementStyle

[Function]

```
public static getComputedElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    propertyNames: string[]
): Dictionary<string>;
```

Returns several active CSS properties of an HTML element. This also takes into account properties that affect CSS files. If the transferred object contains several elements (like for an jQuery collection), the first element is used.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the element to be verified • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.
propertyNames	string [▶ 2108][]	String vector of the names of the CSS properties

Return value

Type	Description
Dictionary [▶ 2086]<string [▶ 2108]>	Object with all CSS properties of the first element in the jQuery collection. If a requested property is not (or no longer) set, this entry is not set in the object. Returns an empty object in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
var someStyles = TcHmi.StyleProvider.getComputedElementStyle(cntrlElem,
     ['border-width', 'height', 'does-not-exist']
);
/* returns:
{
    'border-width': '0px',
    'height': '50px'
}
*/
TcHmi.Log.debug('Border Width is: '+ someStyles['border-width']);
```

15.1.1.17.6 getComputedElementStyle

[Function]

```
public static getComputedElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    propertyName: string
): Dictionary<string>;
```

Returns an active CSS property of an HTML element. This also takes into account properties that affect CSS files. If the transferred object contains several elements (like for an jQuery collection), the first element is used.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the element to be verified • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.
propertyName	string [▶ 2108]	Name of the CSS property

Return value

Type	Description
Dictionary [▶ 2086]<string [▶ 2108]>	Object with the CSS property of the first element in the jQuery collection. If a requested property is not (or no longer) set, this entry is not set in the object. Returns an empty object in the event of an error.

Note: Available since version 1.10

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
var borderWidthStyle =
TcHmi.StyleProvider.getComputedElementStyle(cntrlElem, 'border-width');
/* returns:
{
    'border-width': '0px',
}
*/
TcHmi.Log.debug('Border Width is: '+ borderWidthStyle['border-width']);
```

15.1.1.17.7 setSimpleElementStyle

[Function]

```
public static setSimpleElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    styles: Dictionary<string | string[] | null> | null
): void;
```

Sets multiple CSS properties for one or more HTML or SVG elements.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
styles	Dictionary [▶ 2086]<string [▶ 2108], string [▶ 2108][], null [▶ 2108]>, null [▶ 2108]	Desired new values of the CSS property in an object. The value null (not the string "null") deletes the current value.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
TcHmi.StyleProvider.setSimpleElementStyle(cntrlElem,
    {
        'height': '42px',
        'width': '140px'
    }
);
```

15.1.1.17.8 setSimpleElementStyle

```
public static setSimpleElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    property: string,
    value: string | null | undefined
): void;
```

Sets a CSS property for one or more HTML or SVG elements.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.
property	string [▶ 2108]	Name of the CSS property
value	string [▶ 2108], null [▶ 2108], undefined [▶ 2108]	Desired new value of the CSS property. The value null (not the string "null") or undefined deletes the current value.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
TcHmi.StyleProvider.setSimpleElementStyle(cntrlElem, 'height', '42px');
```

15.1.1.17.9 setSimpleElementStyle

[Function]

```
public static setSimpleElementStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    property: string,
    values: string[] | null
): void;
```

Sets multiple values for one or more HTML or SVG elements on a CSS property. In rare cases, several CSS values for the same property may be required for vendor prefixes. The last value known to this browser remains active.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Individual element, for example HTML or SVG element.
property	string [▶ 2108]	Name of the CSS property
values	string [▶ 2108][]	Desired new values of the CSS property. The value null (not the string "null") deletes the current value.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
var control = TcHmi.Controls.get('TcHmiButton');
if(control === undefined){
    return;
}
var cntrlElem = control.getElement();
TcHmi.StyleProvider.setSimpleElementStyle(
    cntrlElem,
    'display',
     ['-ms-flexbox', '-webkit-flex', 'flex']
);
```

15.1.1.17.10 processGenericStyle

[Function]

```
public static processGenericStyle(
    controlName: string,
    selector: string,
    styles: Dictionary<string | string[] | null | undefined> | null
): void;
```

Sets multiple CSS properties for one or more HTML elements within a control.

This API is slower than the setSimpleStyle [▶ 2015] function, but it can reach dynamically changing child elements.

Parameter

Name	Type	Description
controlName	string [▶ 2108]	Name of the control
selector	string [▶ 2108]	CSS selector within the control
styles	Dictionary<string [▶ 2108], string [▶ 2108][], null [▶ 2108], undefined [▶ 2108]>, null [▶ 2108]	Desired new values of the CSS property in an object. The value null (not the string "null") deletes the current value.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.StyleProvider.processGenericStyle(
    'DataGrid1',
    'td[data-row="' + 5 + '"][data-column="' + 2 + '"]',
    {
        'background-color': 'red'
    }
);
```

15.1.1.17.11 processGenericStyle

[Function]

```
public static processGenericStyle(
    controlName: string,
    selector: string,
    property: string,
    value: string | null | undefined
): void;
```

Sets a CSS property for one or more HTML elements within a control.

This API is slower than the setSimpleStyle [▶ 2016] function, but it can reach dynamically changing child elements.

Parameter

Name	Type	Description
controlName	string [▶ 2108]	Name of the control
selector	string [▶ 2108]	CSS selector within the control
property	string [▶ 2108]	Name of the CSS property
value	string [▶ 2108], null [▶ 2108], undefined [▶ 2108]	Desired new value of the CSS property. The value null (not the string "null") or undefined deletes the current value.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.StyleProvider.processGenericStyle(
    'DataGrid1',
    'td[data-row="' + 5 + '"][data-column="' + 2 + '"]',
    'background-color',
    'red'
);
```

15.1.1.17.12 processBackground

[Function]

```
public static processBackground(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.Background | null
): void;
```

Sets multiple background properties for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.Background [▶ 2080], null [▶ 2108]	Complex object for the background color and the background image.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBackground(newElem,
    {
        color: {color: 'red'},
        image: null,
        imageWidth: null,
        imageWidthUnit: 'px',
        imageHeight: null,
        imageHeightUnit: 'px',
        imageHorizontalAlignment: 'Center',
        imageVerticalAlignment: 'Center',
        imagePadding: null
    }
);
```

15.1.1.17.13 processBackgroundColor

[Function]

```
public static processBackgroundColor(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.Color | null | undefined
): void;
```

Sets a background color for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.Color [▶ 2084], null [▶ 2108]	Object for the background color.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBackgroundColor(newElem,
    {color: 'red'}
);
```

15.1.1.17.14 processBackgroundImage

[Function]

```
public static processBackgroundImage(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: string | null | undefined
): void;
```

Sets the background image for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	string [▶ 2108], null [▶ 2108], undefined [▶ 2108]	Path to the background image.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBackgroundImage(newElem,
    'Images/myBackground.png'
);
```

15.1.1.17.15 processBorderColor

[Function]

```
public static processBorderColor(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.Color | null | undefined
): void;
```

Sets the border color for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.Color [▶ 2084], null [▶ 2108], undefined [▶ 2108]	Object for the color.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBorderColor(newElem,
    {color: 'red'}
);
```

15.1.1.17.16 processBorderWidth

[Function]

```
public static processBorderWidth(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: Partial<TcHmi.BorderWidth> | null | undefined
): void;
```

Sets the border width for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	Partial<TcHmi.BorderWidth [▶ 2082]>, null [▶ 2108], undefined [▶ 2108]	Object for the width.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBorderWidth(newElem,
    {
        left: 20,
        top: 30,
        right: 40,
        bottom: 10
    }
);
```

15.1.1.17.17 processBorderRadius

[Function]

```
public static processBorderRadius(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.BorderRadius | null | undefined
): void;
```

Sets the border radius for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.BorderRadius [▶ 2081], null [▶ 2108], undefined [▶ 2108]	Object for the radius.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBorderRadius(newElem,
    {
        topLeft: 20,
        topRight: 30,
        bottomRight: 40,
        bottomLeft: 10
    }
);
```

15.1.1.17.18 processBorderStyle

[Function]

```
public static processBorderStyle(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.BorderStyle | null | undefined
): void;
```

Sets the border style for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.BorderStyle [▶ 2081], null [▶ 2108]	Object for the radius.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBorderStyle(newElem,
    {
        left: 'Solid',
        top: 'Dashed',
        bottom: 'Dotted',
        right: 'None'
    }
);
```

15.1.1.17.19 processTextColor

[Function]

```
public static processTestColor(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.Color | null | undefined
): void;
```

Sets the text color for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.Color [▶ 2084], null [▶ 2108], undefined [▶ 2108]	Object for the color.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processTextColor(newElem,
    {color: 'red'}
);
```

15.1.1.17.20 processFontFamily

[Function]

```
public static processFontFamily(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    valueNew: string | null | undefined
): void;
```

Sets the font family for one or more HTML or SVG elements.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
valueNew	string [▶ 2108], null [▶ 2108], undefined [▶ 2108]	Value for the font family.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processFontFamily(newElem,
    'monospace'
);
```

15.1.1.17.21 processFontSize

[Function]

```
public static processFontSize(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    valueNew: number | null | undefined,
    unitNew: TcHmi.DimensionUnit = 'px'
): void
```

Sets the font size for one or more HTML or SVG elements. Please note that a percentage unit is not relative to the size of the parent element but to the font size of the parent element.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
valueNew	number [▶ 2108], null [▶ 2108], undefined [▶ 2108]	Numerical value for the font size.
unitNew	TcHmi.DimensionUnit [▶ 2086]	Unit for the font size (if not provided, then 'px' applies)

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processFontSize(newElem,
    42,
    'px'
);
```

15.1.1.17.22 processFontStyle

[Function]

```
public static processFontStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    valueNew: TcHmi.FontStyle | null | undefined
): void
```

Sets the font style for one or more HTML or SVG elements.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
valueNew	TcHmi.FontStyle [▶ 2087], null [▶ 2108]	Value for the font style.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processFontStyle(newElem,
    'Italic'
);
```

15.1.1.17.23 processFontWeight

[Function]

```
public static processFontStyle(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    valueNew: TcHmi.FontWeight | null | undefined
): void
```

Sets the font weight for one or more HTML or SVG elements.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
valueNew	TcHmi.FontWeight [▶ 2088], null [▶ 2108]	Value for the font weight.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processFontWeight(newElem,
    'Bold'
);
```

15.1.1.17.24 processFillColor

[Function]

```
public static processFillColor(
    element: JQuery | SVGElement[] | NodeListOf<SVGElement> | HTMLCollectionOf<SVGElement> |
SVGElement | undefined,
    valueNew: TcHmi.Color | null | undefined
): void;
```

Sets the fill color for one or more SVG elements.

Parameter

Name	Type	Description
element **Note:** Type SVGElement supported from version 1.10 **Note:** Types SVGElement[] | NodeListOf<SVGElement> | HTMLCollectionOf<SVGElement> from version 1.10.1171.21	JQuery [▶ 2108], SVGElement[], NodeListOf<SVGElement>, HTMLCollectionOf<SVGElement>, SVGElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single SVG element
valueNew	TcHmi.Color [▶ 2084], null [▶ 2108]	Object for the color.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
var newElem = dummyControl.getElement().find('rect'); // SVG rect as a control child
TcHmi.StyleProvider.processFillColor(newElem,
    {color: 'red'}
);
```

15.1.1.17.25 processStrokeColor

[Function]

```
public static processStrokeColor(
    element: JQuery | SVGElement[] | NodeListOf<SVGElement> | HTMLCollectionOf<SVGElement> |
SVGElement | undefined,
    valueNew: TcHmi.Color | null | undefined
): void;
```

Sets the stroke color for one or more SVG elements.

Parameter

Name	Type	Description
element **Note:** Type SVGElement supported from version 1.10 **Note:** Types SVGElement[] | NodeListOf<SVGElement> | HTMLCollectionOf<SVGElement> from version 1.10.1171.21	JQuery [▶ 2108], SVGElement[], NodeListOf<SVGElement>, HTMLCollectionOf<SVGElement>, SVGElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single SVG element
valueNew	TcHmi.Color [▶ 2084], null [▶ 2108]	Object for the color.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
var newElem = this.getElement().find('rect'); // SVG rect as a control child
TcHmi.StyleProvider.processStrokeColor(newElem,
    {color: 'blue'}
);
```

15.1.1.17.26 processBoxShadow

[Function]

```
public static processBoxShadow(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.BoxShadow[] | null | undefined
): void;
```

Sets the shadow for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.BoxShadow [▶ 2083][], null [▶ 2108], undefined [▶ 2108]	Object for the shadow.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processBoxShadow(newElem,
    [{
        blur: 5.0,
        blurUnit: 'px',
        color: { color: 'rgba(0, 0, 0, 0.6)' },
        offsetX: 0.0,
        offsetXUnit: 'px',
        offsetY: 0.0,
        offsetYUnit: 'px',
        spread: 0.0,
        spreadUnit: 'px'
    }]
);
```

15.1.1.17.27 processVisibility

[Function]

```
public static processVisibility(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    valueNew: TcHmi.Visibility | null | undefined
): void;
```

Sets the visibility for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
valueNew	TcHmi.Visibility [▶ 2097], null [▶ 2108], undefined [▶ 2108]	Value for the visibility.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processVisibility(newElem,
    'Collapsed'
);
```

15.1.1.17.28 processContentHorizontalAlignment

[Function]

```
public static processContentHorizontalAlignment(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.HorizontalAlignment | null | undefined
): void;
```

Sets the alignment for one or more HTML elements. Note that the child elements are positioned here, not the element itself.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.HorizontalAlignment [▶ 2089], null [▶ 2108], undefined [▶ 2108]	Value for the alignment.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processContentHorizontalAlignment(newElem,
    'Center'
);
```

15.1.1.17.29 processContentVerticalAlignment

[Function]

```
public static processContentVerticalAlignment(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.VerticalAlignment | null | undefined
): void;
```

Sets the alignment for one or more HTML elements. Note that the child elements are positioned here, not the element itself.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.VerticalAlignment [▶ 2089], null [▶ 2108], undefined [▶ 2108]	Value for the alignment.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processContentVerticalAlignment(newElem,
    'Center'
);
```

15.1.1.17.30 processContentPadding

[Function]

```
public static processContentPadding(
    element: JQuery | Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> | Element |
undefined,
    valueNew: Partial<TcHmi.FourSidedCss> | null | undefined
): void;
```

Sets the padding for one or more HTML elements. Note that the child elements are positioned here, not the element itself.

Parameter

Name	Type	Description
element **Note:** Element type supported from version 1.10 **Note:** Types Element[] | NodeListOf<Element> | HTMLCollectionOf<Element> from version 1.10.1171.21	JQuery [▶ 2108], Element[], NodeListOf<Element>, HTMLCollectionOf<Element>, Element, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • individual element, for example HTML or SVG element
valueNew	Partial<TcHmi.FourSidedCss [▶ 2088]>, null [▶ 2108], undefined [▶ 2108]	Value for the padding.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processContentPadding(newElem,
    {
        "left": 10,
        "right": 0,
        "leftUnit": "px",
        "rightUnit": "px",
    }
);
```

15.1.1.17.31 processTransform

[Function]

```
public static processTransform(
    element: JQuery | HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> |
HTMLElement | undefined,
    valueNew: TcHmi.Transform | TcHmi.Transform[] | null | undefined
): void;
```

Sets a transformation for one or more HTML elements.

Parameter

Name	Type	Description
element **Note:** Type HTMLElement supported from version 1.10 **Note:** Types HTMLElement[] | NodeListOf<HTMLElement> | HTMLCollectionOf<HTMLElement> from version 1.10.1171.21	JQuery [▶ 2108], HTMLElement[], NodeListOf<HTMLElement>, HTMLCollectionOf<HTMLElement>, HTMLElement, undefined [▶ 2108]	• jQuery collection with the elements to be changed • Any array-type element list (for example from `document.querySelectorAll()` or `document.getElementsByClassName()`) • Single HTML element
valueNew	TcHmi.Transform [▶ 2097], TcHmi.Transform [▶ 2097][], null [▶ 2108], undefined [▶ 2108]	Value for the transformation.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript 1

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processTransform(newElem,
    {
        transformType: 'Rotate',
        angle: 35
    }
);
```

Example - JavaScript 2

```
var newElem = $('<div class="testdiv"></div>');
TcHmi.StyleProvider.processTransform(newElem,
    [
        {
            transformType: 'Rotate',
            angle: 35
        },
        {
            transformType: 'Scale',
            x: 2,
            y: 2
        }
    ]
);
```

15.1.1.17.32 resolveSolidColorAsCssValue

[Function]

```
public static resolveSolidColorAsCssValue(
    colorObject: TcHmi.SolidColor
): string;
```

Converts a TcHmi.SolidColor value to a CSS value. Please note that the value must be a valid SolidColor. To check this, you can use TcHmi.isSolidColor [▶ 2084].

This function also handles the special color "none", which can be set via engineering.

Parameter

Name	Type	Description
colorObject	TcHmi.SolidColor [▶ 2084]	Color value to be translated.

Return value

Type	Description
string [▶ 2108]	CSS color

Note: Available since version 1.10

Example - JavaScript

```
var myColor = {
    color: 'red'
};
var newElem = $('<div class="testdiv"></div>');
if (TcHmi.isSolidColor(myColor)) {
    var cssColor = TcHmi.StyleProvider.resolveSolidColorAsCssValue(myColor);
    TcHmi.StyleProvider.setSimpleElementStyle(newElem, 'color', cssColor);
} else {
    TcHmi.StyleProvider.setSimpleElementStyle(newElem, 'color', null);
}
```

Most applications can use high-level APIs such as TcHmi.StyleProvider.processTextColor [▶ 2025] or similar.

15.1.1.17.33 resolveLinearGradientColorAsCssValue

[Function]

```
public static resolveLinearGradientColorAsCssValue(
    colorObject: TcHmi.LinearGradientColor
): string;
```

Translates a TcHmi.linearGradientColor value to a CSS value. Please note that the value must be a valid LinearGradientColor. To check this, you can use TcHmi.isLinearGradientColor [▶ 2085].

Parameter

Name	Type	Description
colorObject	TcHmi.LinearGradientColor [▶ 2085]	Color value to be translated.

Return value

Type	Description
string [▶ 2108]	CSS color

Note: Available since version 1.10

Example - JavaScript

```
var myColor = {
    "angle": 90.0,
    "stopPoints": [
        {
            "color": "rgba(0, 0, 0, 1)",
            "stop": "0%"
        },
        {
            "color": "rgba(215, 61, 61, 1)",
            "stop": "100%"
        }
    ]
};
var newElem = $('<div class="testdiv"></div>');
if (TcHmi.isLinearGradientColor (myColor)) {
    var cssColor = TcHmi.StyleProvider.resolveLinearGradientColorAsCssValue(myColor);
```

```
    TcHmi.StyleProvider.setSimpleElementStyle(newElem, 'background-image', cssColor);
} else {
    TcHmi.StyleProvider.setSimpleElementStyle(newElem, 'background-image', null);
}
```

Most applications can use high-level APIs such as TcHmi.StyleProvider.processBackgroundColor [▶ 2020] or similar.

15.1.1.18 Symbol

[Class]

Access to symbol information.

Constructors

Name	Description
constructor [▶ 2037]	Creates an instance of the symbol.

Functions

Name	Description
read [▶ 2037]	Reads the value of the symbol.
readEx [▶ 2038]	Reads the value of the symbol.
write [▶ 2039]	Writes the value of the symbol.
watch [▶ 2040]	Monitors the symbol for changes.
exists [▶ 2041]	Checks if the symbol exists.
resolveSchema [▶ 2041]	Resolves the schema of the symbol and returns it using the callback function.
getExpression [▶ 2042]	Returns the TcHmi.SymbolExpression object associated with the symbol.
destroy [▶ 2042]	Destroys the Object symbol.

Static functions

Name	Description
read [▶ 2043]	Reads the value of a symbol addressed by its name and symbol type.
readEx [▶ 2043]	Reads the value of a symbol addressed via the symbol expression.
readEx2 [▶ 2044]	Reads the value of a symbol addressed via the symbol expression.
write [▶ 2044]	Writes the value of a symbol addressed by its name and symbol type.
writeEx [▶ 2045]	Writes the value of a symbol addressed via the symbol expression.
exists [▶ 2046]	Checks if the symbol exists.
resolveSchema [▶ 2047]	Resolves the schema of the symbol and returns it using the callback function.
isSymbolExpression [▶ 2048]	Checks whether an expression is a valid symbol expression.

Interfaces

Name	Description
IReadResultObject [▶ 2048]	Extends TcHmi.IResultObject
IServerReadResultObject [▶ 2048]	Extends IReadResultObject
IWriteResultObject [▶ 2049]	Extends TcHmi.IResultObject
IServerWriteResultObject [▶ 2049]	Extends IWriteResultObject
IWatchResultObject [▶ 2050]	Extends TcHmi.IResultObject
IServerWatchResultObject [▶ 2050]	Extends IWatchResultObject
ISchemaResultObject [▶ 2050]	Extends TcHmi.IResultObject
IExistsResultObject [▶ 2051]	Extends TcHmi.IResultObject

Note: Available since version 1.8

15.1.1.18.1 constructor

[Function]

```
public constructor(expression: string);
```

Creates a symbol instance.

Note: Instances created with new must be destroyed by calling the destroy [▶ 2042] function if they are no longer needed.

Parameter

Name	Type	Description
expression	string [▶ 2108]	Symbol expression

Example - JavaScript

```
var symbol = new TcHmi.Symbol('%i%myInternalSym%/i%');
symbol.read(function(data) {
   // Do something…
   symbol.destroy();
   symbol = null;
   return;
});
```

Note: Available since version 1.8

15.1.1.18.2 read

[Function]

```
public read(): any;
```

Reads the value of the symbol.

Note: From version 1.10, symbol queries are executed directly against the component that makes the data available. Since a synchronous query of symbols from the server is not possible, please use the function readEx [▶ 2038] to query server symbols.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
any [▶ 2107]	Current value of the symbol. Can have any type.

Note: Available since version 1.8

Example - JavaScript

```
var sym = new TcHmi.Symbol('%i%myInternalSym%/i%');
var symVal = sym.read();
```

15.1.1.18.3 readEx

[Function]

```
public readEx(
    callback?: (data: Symbol.IReadResultObject | Symbol.IServerReadResultObject) => void
): any;
```

Reads the value of the symbol.

The callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
callback [Optional]	(data: Symbol.IReadResultObject [▶ 2048] | Symbol.IServerReadResultObject [▶ 2048]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
var symbol = new TcHmi.Symbol('%s%PLC1.MAIN.sTest%/s%');
symbol.readEx(function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var value = data.value;
        console.log(value);
    } else {
        // Handle error...
    }
});
```

15.1.1.18.4 write

[Function]

```
public write(
    value: any,
    callback?: (
     (data: Symbol.IWriteResultObject | TcHmi.Symbol.IServerWriteResultObject) => void
    )
): void;
```

Writes the value of the symbol.

Note: From 1.10, the callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
value	any [▶ 2107]	New value of the symbol. Can have any type.
callback [Optional]	**Note:** Version 1.8 (data: TcHmi.IResultObject [▶ 2094]| TcHmi.Server.IResultObject [▶ 2001]) => void **Note:** Version 1.10 (data: Symbol.IWriteResultObject [▶ 2049] | Symbol.IServerWriteResultObject [▶ 2049]) => void The change is backward compatible.	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var symbol = new TcHmi.Symbol('%s%PLC1.MAIN.sTest%/s%');
symbol.write('Test', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle success...
    } else {
        // Handle error...
    }
});
```

15.1.1.18.5 watch

[Function]

```
public watch(
    callback: (data: IWatchResultObject | IServerWatchResultObject) => void
): TcHmi.DestroyFunction;
```

Monitors the symbol for changes.

The callback function is called either synchronously or asynchronously with the change, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
callback [Optional]	(data: IWatchResultObject [▶ 2050]. IServerWatchResultObject [▶ 2050]) => void	This is triggered when the symbol value changes or an error has occurred. The callback function is initially triggered once with the current value. For details, please refer to the parameter data.

Return value

Type	Description
TcHmi.DestroyFunction [▶ 2101]	When this function is called, monitoring for this symbol is deactivated and all associated resources are released. If the callback function is called synchronously, the return value of the function cannot be accessed because it has not yet been defined. In this case, there is a reference to the function in the parameter of the callback function. See IWatchResultObject [▶ 2050], IServerWatchResultObject [▶ 2050]

Note: Available since version 1.10

```
var symbol = new TcHmi.Symbol('%s%PLC1.MAIN.sTest%/s%');
var destroySymbol = symbol.watch(function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var value = data.value;
        console.log(value);
    } else {
        // Handle error...
    }
    // Stop watch inline
    // data.destroy(); // Call the destroy function inline to stop the watch and free resources.
});
// Stop watch
// destroySymbol(); // Call the destroy function to stop the watch and free resources.
```

15.1.1.18.6 exists

[Function]

```
public exists(
    callback?: (data: Symbol.IExistsResultObject) => void
): any;
```

Checks if the symbol exists.

Parameter

Name	Type	Description
callback [optional]	(data: Symbol.IExistsResultObject [▶ 2051]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.10.1336.10

Sample - JavaScript

```
var symbol = new TcHmi.Symbol('%s%PLC1.MAIN.sTest%/s%');
symbol.exists(function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var symExists = data.result;
        console.log('Symbol exists: '+ symExists);
    } else {
        // Handle error...
    }
});
```

15.1.1.18.7 resolveSchema

[Function]

```
public resolveSchema(
    callback?: (data: Symbol.ISchemaResultObject) => void
): any;
```

Resolves the schema of the symbol and returns it using the callback function.

The callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
callback [Optional]	(data: Symbol.ISchemaResultObject [▶ 2050]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
var symbol = new TcHmi.Symbol('%s%PLC1.MAIN.sTest%/s%');
symbol.resolveSchema(function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var schema = data.schema;
        console.log(schema);
    } else {
        // Handle error...
    }
});
```

15.1.1.18.8 getExpression

[Function]

```
public static getExpression (): TcHmi.SymbolExpression;
```

Returns the TcHmi.SymbolExpression object associated with the symbol.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.SymbolExpression [▶ 2051]	The TcHmi.SymbolExpression object relating to the symbol. This object provides information about the corresponding symbol expression.

Note: Available since version 1.10

15.1.1.18.9 destroy

[Function]

```
public static destroy (): void;
```

Destroys the Object symbol.

Parameter

Name	Type	Description

Return value

Type	Description
void	No return value

Note: Available since version 1.10

15.1.1.18.10 read

```
public static read(name: string, type: TcHmi.SymbolType): any;
```

Reads the value of a symbol addressed by its name and symbol type.

Note: From version 1.10, symbol queries are executed directly against the component that makes the data available. Since a synchronous query of symbols from the server is not possible, please use the function readEx2 [▶ 2044] to query server symbols.

Parameter

Name	Type	Description
name	string [▶ 2108]	Name of the symbol
type	TcHmi.SymbolType [▶ 2100]	Type of symbol

Return value

Type	Description
any [▶ 2107]	Current value of the symbol. Can have any type.

Note: Available since version 1.8

Example - JavaScript

```
var symVal = TcHmi.Symbol.read('myInternalSym', TcHmi.SymbolType.Internal);
```

15.1.1.18.11 readEx

[Function]

```
public static readEx(expression: string): any;
```

Reads the value of a symbol addressed via the symbol expression.
Note: From version 1.10, symbol queries are executed directly against the component that makes the data available. Since a synchronous query of symbols from the server is not possible, please use the function readEx2 [▶ 2044] to query server symbols.

Parameter

Name	Type	Description
expression	string [▶ 2108]	Symbol expression

Return value

Type	Description
any [▶ 2107]	Current value of the symbol.

Note: Available since version 1.8

Example - JavaScript

```
var symVal = TcHmi.Symbol.readEx('%i%myInternalSym%/i%');
```

15.1.1.18.12 readEx2

[Function]

```
public static readEx2(
    expression: string,
    callback?: (data: Symbol.IReadResultObject) => void
): void;
```

Reads the value of a symbol addressed via the symbol expression.

The callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
expression	string [▶ 2108]	Symbol expression
callback [Optional]	(data: Symbol.IReadResultObject [▶ 2048]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Symbol.readEx2('%s%PLC1.MAIN.sTest%/s%', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var value = data.value;
        console.log(value);
    } else {
        // Handle error...
    }
});
```

15.1.1.18.13 write

```
public static write(
    name: string,
    type: TcHmi.SymbolType,
    value: any,
     (data: Symbol.IWriteResultObject | TcHmi.Symbol.IServerWriteResultObject) => void
): void;
```

Writes the value of a symbol addressed by its name and symbol type.

Note: From 1.10, the callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
name	string [▶ 2108]	Name of the symbol
type	TcHmi.SymbolType [▶ 2100]	Type of symbol
value	any [▶ 2107]	New value of the symbol. Can have any type.
callback [Optional]	**Note:** Version 1.8 (data: TcHmi.IResultObject [▶ 2094]| TcHmi.Server.IResultObject [▶ 2001]) => void **Note:** Version 1.10 (data: Symbol.IWriteResultObject [▶ 2049] | Symbol.IServerWriteResultObject [▶ 2049]) => void The change is backward compatible.	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value.

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Symbol.write ('PLC1.MAIN.sTest', TcHmi.SymbolType.Server, 'Test', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle success...
    } else {
        // Handle error...
    }
});
```

15.1.1.18.14 writeEx

[Function]

```
public static writeEx(
    expression: string,
    value: any,
    callback: (data: TcHmi.IResultObject | TcHmi.Server.IResultObject) => void
): void;
```

Writes the value of a symbol addressed via the symbol expression.

Note: From 1.10, the callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
expression	string [▶ 2108]	Symbol expression
value	any [▶ 2107]	New value of the symbol. Can have any type.
callback [Optional]	(data: TcHmi.IResultObject [▶ 2094] TcHmi.Server.IResultObject [▶ 2001]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Symbol.writeEx('%s%PLC1.MAIN.sTest%/s%', 'Test', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle success...
    } else {
        // Handle error...
    }
});
```

15.1.1.18.15 exists

[Function]

```
public static exists(
    expression: string,
    callback?: (data: Symbol.IExistsResultObject) => void
): any;
```

Checks if the symbol exists.

Parameter

Name	Type	Description
expression	string [▶ 2108]	Name of the symbol
callback [optional]	(data: Symbol.IExistsResultObject [▶ 2051]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.10.1336.10

Sample - JavaScript

```
TcHmi.Symbol.exists('%s%PLC1.MAIN.sTest%/s%', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var symExists = data.result;
        console.log('Symbol exists: '+ symExists);
    } else {
        // Handle error...
    }
});
```

15.1.1.18.16 resolveSchema

[Function]

```
public static resolveSchema(
    expression: string,
    callback?: (data: Symbol.ISchemaResultObject) => void
): any;
```

Resolves the schema of the symbol and returns it using the callback function.

The callback function is called either synchronously or asynchronously, depending on the symbol type used.

Server	asynchronous
Internal	synchronous
LocalizedText	synchronous
PartialParam	synchronous
TemplateParam	synchronous
Function	synchronous
Control	synchronous

Parameter

Name	Type	Description
expression	string [▶ 2108]	Name of the symbol
callback [Optional]	(data: Symbol.ISchemaResultObject [▶ 2050]) => void	This is triggered when the action has been successfully completed or an error has occurred. For details, please refer to the parameter data.

Return value

Type	Description
void	No return value

Note: Available since version 1.10

Example - JavaScript

```
TcHmi.Symbol.resolveSchema('%s%PLC1.MAIN.sTest%/s%', function (data) {
    if (data.error === TcHmi.Errors.NONE) {
        // Handle result value...
        var schema = data.schema;
        console.log(schema);
```

```
    } else {
        // Handle error...
    }
});
```

15.1.1.18.17 isSymbolExpression

[Function]

```
public static isSymbolExpression(expression: string): boolean;
```

Checks whether an expression is a valid symbol expression.

Parameter

Name	Type	Description
expression	string [▶ 2108]	Symbol expression for checking

Return value

Type	Description
boolean [▶ 2107]	Result as to whether a valid symbol expression was checked

Note: Available since version 1.8

Example - JavaScript

```
if(TcHmi.Symbol.isSymbolExpression('%i%myInternalSym%/i%')){};     // true
if(TcHmi.Symbol.isSymbolExpression('%s%PLC1.MAIN.dValue%/s%')){}; // true
if(TcHmi.Symbol.isSymbolExpression('%s%PLC1.MAIN.dValue%/f%')){}; // false
```

15.1.1.18.18 IReadResultObject

[Interface]

```
export interface IReadResultObject extends TcHmi.IResultObject{
    value?: any;
}
```

Extends TcHmi.IResultObject [▶ 2094]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
value [Optional]	any [▶ 2107]	Contains the value that was read if no error occurred.

Note: Available since version 1.10

15.1.1.18.19 IServerReadResultObject

[Interface]

```
export interface IServerReadResultObject extends IReadResultObject {
    response?: TcHmi.Server.IMessage;
}
```

Extends IReadResultObject [▶ 2048]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
value [Optional]	any [▶ 2107]	Contains the value that was read if no error occurred.
response [Optional]	TcHmi.Server.IMessage [▶ 2004]	Contains the response object of the server, if available.

Note: Available since version 1.10

15.1.1.18.20 IWriteResultObject

[Interface]

```
export interface IWriteResultObject extends TcHmi.IResultObject {
    value?: any;
}
```

Extends TcHmi.IResultObject [▶ 2094]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
value [Optional]	any [▶ 2107]	Contains the value that was written if no error occurred.

Note: Available since version 1.10

15.1.1.18.21 IServerWriteResultObject

[Interface]

```
export interface IServerWriteResultObject extends IWriteResultObject {
    response?: TcHmi.Server.IMessage;
}
```

Extends IWriteResultObject [▶ 2049]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
value [Optional]	any [▶ 2107]	Contains the value that was written if no error occurred.
response [Optional]	TcHmi.Server.IMessage [▶ 2004]	Contains the response object of the server, if available.

Note: Available since version 1.10

15.1.1.18.22 IWatchResultObject

[Interface]

```
export interface IWatchResultObject extends TcHmi.IResultObject{
    value?: any;
    destroy: TcHmi.DestroyFunction;
}
```

Extends TcHmi.IResultObject [▶ 2094]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
value [Optional]	any [▶ 2107]	Contains the value that was read if no error occurred.
destroy	DestroyFunction [▶ 2101]	When this function is called, monitoring for this symbol is deactivated and all associated resources are released.

Note: Available since version 1.10

15.1.1.18.23 IServerWatchResultObject

[Interface]

```
export interface IServerWatchResultObject extends IWatchesultObject {
    response?: TcHmi.Server.IMessage;
}
```

Extends IWatchResultObject [▶ 2049]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
value [Optional]	any [▶ 2107]	Contains the value that was read if no error occurred.
destroy	DestroyFunction [▶ 2101]	When this function is called, monitoring for this symbol is deactivated and all associated resources are released.
response [Optional]	TcHmi.Server.IMessage [▶ 2004]	Contains the response object of the server, if available.

Note: Available since version 1.10

15.1.1.18.24 ISchemaResultObject

[Interface]

```
export interface ISymbolSchemaResultObject extends TcHmi.IResultObject {
    schema?: TcHmi.JsonSchema;
}
```

Extends TcHmi.IResultObject [▶ 2094]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [Optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
schema [optional]	TcHmi.JsonSchema [▶ 2102]	The JSON schema assigned to the symbol or symbol value.

Note: Available since version 1.10

15.1.1.18.25 IExistsResultObject

[Interface]

```
export interface IExistsResultObject extends TcHmi.IResultObject {
    result?: boolean;
}
```

Extends TcHmi.IResultObject [▶ 2094]

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [optional]	any [▶ 2107]	Optional details about error. Content depends on the function used and the error.
result [Optional]	boolean [▶ 2107]	Result whether the symbol exists or not

Note: Available since version 1.10.1336.10

15.1.1.19 SymbolExpression

[Class]

Symbol expression parser.

Constructors

Name	Description
constructor [▶ 2052]	Creates an instance of SymbolExpression.

Functions

Name	Description
toString [▶ 2052]	Serializes the symbol expression as a string.
getContent [▶ 2053]	Returns the content of the symbol expression.
getTag [▶ 2053]	Returns the tag of the symbol expression.
getName [▶ 2053]	Returns the name of the symbol expression.
getPath [▶ 2054]	Returns the path of the symbol expression as a string.
getPathTokens [▶ 2054]	Returns the path of the symbol expression as an array of its components.
getType [▶ 2055]	Returns the type of the symbol expression.
getOptions [▶ 2056]	Returns an object with the options of the symbol expression.

Note: Available since version 1.10

15.1.1.19.1 constructor

[Function]

```
public constructor(expression: string);
```

Creates a SymbolExpression instance.

Parameter

Name	Type	Description
expression	string [▶ 2108]	Symbol expression

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol::A%/i%');
```

15.1.1.19.2 toString

[Function]

```
public toString();
```

Serializes the symbol expression as a string.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
string [▶ 2108]	Symbol expression

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var str = symbolExpression.toString();
console.log(str); // %i%MySymbol[0]::A::B%/i%
```

15.1.1.19.3 getContent

[Function]

```
public getContent();
```

Returns the content of the symbol expression.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
string [▶ 2108] | null [▶ 2108]	Content of the symbol expression.

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var content = symbolExpression.getContent();
console.log(content); // MySymbol[0]::A::B
```

15.1.1.19.4 getTag

[Function]

```
public getTag();
```

Returns the tag of the symbol expression.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
string [▶ 2108] | null [▶ 2108]	Day of the symbol expression.

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var tag = symbolExpression.getTag();
console.log(tag); // i
```

15.1.1.19.5 getName

[Function]

```
public getName();
```

Returns the name of the symbol expression.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
string [▶ 2108] | null [▶ 2108]	Name of the symbol expression.

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var name = symbolExpression.getName();
console.log(name); // MySymbol
```

15.1.1.19.6 getPath

[Function]

```
public getPah();
```

Returns the path of the symbol expression.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
string [▶ 2108] | null [▶ 2108]	Path of the symbol expression.

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var path = symbolExpression.getPath();
console.log(path); // [0]::A::B
```

15.1.1.19.7 getPathTokens

[Function]

```
public getPathTokens();
```

Returns the path of the symbol expression as an array of its components.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
string [▶ 2108][] | null [▶ 2108]	Path of the symbol expression.

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var pathTokens = symbolExpression.getPathTokens();
console.log(pathTokens); // ["[0]", "A", "B"]
```

15.1.1.19.8 getType

[Function]

```
public getType();
```

Returns the type of the symbol expression.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
TcHmi.SymbolType [▶ 2100] | null [▶ 2108]	Type of the symbol expression.

Note: Available since version 1.10

Example - JavaScript

```
var symbolExpression = new TcHmi.SymbolExpression('%i%MySymbol[0]::A::B%/i%');
var type = symbolExpression.getType();
console.log(type); // 20
console.log(TcHmi.SymbolType[type]); // Internal
```

15.1.1.19.9 getOptions

[Function]

```
public getOptions(): SymbolExpression.IOptions
```

Returns the type of the symbol expression.

Parameter

Name	Type	Description
-	- [▶ 2108]	-

Return value

Type	Description
SymbolExpression.IOptions [▶ 2056]	Symbol expression options.

Note: Available since version 1.10

15.1.1.19.10 IOptions

[Interface]

```
export interface IOptions {
    BindingMode?: TcHmi.BindingMode;
    BindingEvent?: string;
    BindingEvents?: string[];
    Timeout?: number;
    Interval?: number;
    Parallel?: boolean;
}
```

Properties

Name	Type	Description
BindingMode [optional]	TcHmi.BindingMode	Contains the value of the BindingMode option of the symbol expression.
BindingEvent [optional]	string [▶ 2108]	Contains the value of the first BindingEvent option of the symbol expression. The value represents the name of an event of the control to which the symbol is linked. The event is the trigger for reading back the corresponding attribute value to which the symbol is linked.
BindingEvent [optional] **Note:** Available from 1.10.1171.21	string [▶ 2108][]	Contains a list of values of all BindingEvent options of the symbol expression. The values each represent the name of an event of the control to which the symbol is linked. The events are trigger for reading back the corresponding attribute value to which the symbol is linked. The events are registered in the order in which they appear in the symbol expression.
Timeout [optional]	number [▶ 2108]	Contains the value of the symbol expression timeout option. Supported only by symbols of type "Server". Requests against the TwinCAT HMI server based on this symbol object are aborted in milliseconds after the set time has elapsed and return an error.
Interval [optional]	number [▶ 2108]	Contains the value of the interval option of the symbol expression. Supported only by symbols of type "Server". Subscription requests against the TwinCAT HMI server based on this symbol object use this interval time instead of the global time.
Parallel [optional]	boolean [▶ 2107]	Contains the value of the parallel option of the symbol expression.

Name	Type	Description
		Supported only by symbols of type "Server". This option can be used to specify whether ReadWrite requests are sent in parallel or serialized against the TwinCAT HMI server.

Note: Available since version 1.10

15.1.1.20 Theme

[Class]

Provides functions for managing and changing themes. The concept of the theme system can be found on the main theme page [▶ 559].

Static functions

Name	Description
get [▶ 2059]	Returns the name of the currently valid theme.
set [▶ 2060]	Sets a new theme.

Namespaces

Name	Description
Properties [▶ 2060]	Provides functions for control attributes.
Resources [▶ 2062]	Provides functions for control resources.

Interfaces

Name	Description
Resource [▶ 2063]	Definition of a control resource

Note: Available since version 1.10

15.1.1.20.1 get

[Function]

```
public static get(): string;
```

Returns the name of the currently valid theme.

Parameter

Name	Type	Description

Return value

Type	Description
string [▶ 2108]	Name of the current theme

Note: Available since version 1.10

Example - JavaScript

```
var themeName = TcHmi.Theme.get(); // For example: 'Base'
```

15.1.1.20.2 set

[Function]

```
public static set(newValue: string): TcHmi.Errors;
```

Sets a new theme.

Parameter

Name	Type	Description
newValue	string [▶ 2108]	Name of the new theme

Return value

Type	Description
TcHmi.Errors [▶ 2089]	Success or failure as a number.

Note: Available since version 1.10

Example - JavaScript

```
var newTheme = 'myTheme';
var result = TcHmi.Theme.set(newTheme);
if(result === TcHmi.Errors.NONE){
    // valid value
}

newTheme = null;
result = TcHmi.Theme.set(newTheme);
if(result === TcHmi.Errors.NONE){
    // TcHmi.Errors.E_PARAMETER_INVALID
}
```

15.1.1.20.3 Properties

[Class]

This class provides functions for changing the properties/attributes of a control.

Static functions

Name	Description
getDefaultValue [▶ 2060]	Returns the current default value (determined by the theme) of a control property.
setThemeValue [▶ 2061]	Sets the current default value for a control property.

Note: Available since version 1.10

15.1.1.20.3. getDefaultValue
1

[Function]

```
public static getDefaultValue (
    control: TcHmi.Controls.System.baseTcHmiControl,
    propertyName: string
): TcHmi.Theme.Resource;
```

Returns the current default value (determined by the theme) of a control property.

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	
propertyName	string [▶ 2108]	

Return value

Type	Description
TcHmi.Theme.Resource [▶ 2063]	Currently active default value.

Note: Available since version 1.10

Example - JavaScript

```
var themeValue = TcHmi.Theme.Properties.getDefaultValue(this, 'Value');
if(themeValue.origin === 'project'){
    // Value was set in a project theme
}
if(themeValue.value !== null){
    // use themeValue.value
    // Is for example false if this was set in the Theme or defaultValueInternal of its
Description.json
}
```

15.1.1.20.3. setThemeValue
2

[Function]

```
public static setThemeValue (
    control: TcHmi.Controls.System.baseTcHmiControl,
    propertyName: string
): TcHmi.Errors;
```

Sets the current default value of the theme for a control property.

At the same time, this control property is changed when the theme is changed. The value therefore remains set to the theme value.

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	
propertyName	string [▶ 2108]	

Return value

Type	Description
TcHmi.Errors [▶ 2089]	Success or failure as a number.

Note: Available since version 1.10

Example - JavaScript

```
var result = TcHmi.Theme.Properties.setThemeValue(this, 'Value');
if(result === TcHmi.Errors.NONE){
    // valid value
}
var result2 = TcHmi.Theme.Properties.setThemeValue(this, 'RenderedLeft');
if(result2 === TcHmi.Errors.NONE){
    //
```

```
}else{
    // RenderedLeft is readonly!
}
```

15.1.1.20.4 Resources

[Class]

This class provides functions for changing the ThemedResources of a control.

These resources are independent of CSS or attributes. For example, a control can change the appearance of generated elements via the theme. An example is the color of a generated y-axis in a chart control.

They must be made known to the system via the Description.json of the control.

Static functions

Name	Description
get [▶ 2062]	

Note: Available since version 1.10

15.1.1.20.4. get
1

[Function]

```
public static get (
    control: TcHmi.Controls.System.baseTcHmiControl,
    resourceName: string
): TcHmi.Theme.Resource;
```

Returns the current resource value (determined by the theme) of a control. If this value contains a path (e.g. the URL of an image), the base path (i.e. to the current control or project theme directory) must be determined via resolveBasePath [▶ 2063]

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	
resourceName	string [▶ 2108]	

Return value

Type	Description
TcHmi.Theme.Resource [▶ 2063]	Currently active default value.

Note: Available since version 1.10

Sample - JavaScript

```
var themeValue = TcHmi.Theme.Resources.get(this, 'myThemedResource');
if(themeValue.origin === 'project'){
    // Value was set in a project theme
}
if(themeValue.value !== null){
    // use themeValue.value
}
```

15.1.1.20.4. resolveBasePath 2

[Function]

```
public static resolveBasePath (
    control: TcHmi.Controls.System.baseTcHmiControl,
    resource: Theme.Resource
): string;
```

Returns the appropriate base path for a resource value of a control (i.e. to the current control or project theme directory). This is necessary, for example, if the value contains a path (for example, the URL of an image).

This is important because a project may have been overwritten here.

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Control
resource	TcHmi.Theme.Resource [▶ 2063]	Resource value

Return value

Type	Description
string [▶ 2108]	Path

Note: Available since version 1.10.1171.10

Sample - JavaScript

```
var themeValue = TcHmi.Theme.Resources.get(this, 'myThemedResource');
var basePath = TcHmi.Theme.Resources.resolveBasePath(this, themeValue);
var url = basePath;
url += 'myImage.png';
// use url for XmlHttpRequest or in a HTMLImage.src
```

15.1.1.20.5 Resource

[Interface]

```
export interface Resource extends TcHmi.IResultObject {
    value: any | null;
    origin: 'control' | 'project' | 'system';
}
```

This interface extends the TcHmi.IResultObject [▶ 2094] interface.

Properties

Name	Type	Description
value	any [▶ 2107], null [▶ 2108]	Value of the resource.
origin	'control', 'project', 'system'	The value comes from the control, from the project or (in the event of an error) from the system.

Note: Available since version 1.10

15.1.1.21 TopMostLayer

[Class]

Provides functions for an overlay. This area is always above the normal visualization and independent of scrolling.

A jQuery object with one element is passed to the TopMostLayer. This is positioned by the TopMostLayer within an HTMLDivElement, which has the width and height of the browser window. Common CSS attributes for the element are `min-width`, `min-height`, `padding`. To make it easier for the user, an element can be centered on the screen.

It is also possible to react when clicking on the darkened background or when resizing.

If a control adds an element to the TopMostLayer, this control is monitored. If the control is removed (detached) from the DOM, the element is also removed.

Separate functions without this monitoring are available for use outside controls (e.g. codebehind).

Static functions

Name	Description
add [▶ 2064]	Appends an HTML element (from a control) to the overlay area
remove [▶ 2065]	Removes an HTML element (from a control) from the overlay area
addEx [▶ 2066]	Appends an HTML element to the overlay area (codebehind)
removeEx [▶ 2066]	Removes an HTML element from the overlay area (codebehind)

Interfaces

Name	Description
IOptions [▶ 2067]	Options for the add/addEx() function.
IResizeResultObject [▶ 2067]	Callback parameter for resizing.
IElemRemoveResultObject [▶ 2068]	Callback parameters for removing.

Note: Available since version 1.10

15.1.1.21.1 add

[Function]

```
public static add(
    control: TcHmi.Controls.System.baseTcHmiControl,
    element: JQuery | undefined | null,
    options?: TopMostLayer.IOptions
): boolean;
```

Inserts an HTML element into the overlay area above the normal visualization. This function is intended for use from a control. If this is removed (detached) from the DOM, all elements are also removed from the TopMostLayer.

A reference to the element should be stored in the control, in order to be able to call the remove() [▶ 2065] function.

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Calling control.
element	JQuery [▶ 2108]	JQuery collection with an HTML element.
options [optional]	TopMostLayer.IOptions [▶ 2067]	Options for extended functionalities.

Return value

Type	Description
boolean [▶ 2107]	Result as to whether the call was successful

Note: Available since version 1.10

Example - JavaScript

```
var myPopup = $('<div style="background:white;padding:10px;">Hello World<div>');
TcHmi.TopMostLayer.add(this, myPopup, {
    centerHorizontal: true,
    centerVertical: true,
    removeCb: (data) => {
        if (data.canceled) {
            // user clicked on background
        }
    }
});
```

15.1.1.21.2 remove

[Function]

```
public static remove(
    control: TcHmi.Controls.System.baseTcHmiControl,
    element: JQuery
): JQuery;
```

Removes an HTML element from the overlay area. If the element is not currently in the overlay area, it is returned without any changes. This function is intended for use from a control. If this is removed (detached) from the DOM, all elements are also removed from the TopMostLayer.

Parameter

Name	Type	Description
control	TcHmi.Controls.System.baseTcHmiControl [▶ 1118]	Calling control.
element	JQuery [▶ 2108]	JQuery collection with an HTML element.

Return value

Type	Description
JQuery [▶ 2108]	JQuery collection with the removed HTML element.

Note: Available since version 1.10

Example - JavaScript

```
var myPopup = $('<div style="background:white;padding:10px;">Hello World<div>');
TcHmi.TopMostLayer.add(this, myPopup, {
    centerHorizontal: true,
    centerVertical: true,
```

```
    removeCb: (data) => {
        if (data.canceled) {
            // user clicked on background
        }
    }
});
TcHmi.TopMostLayer.remove(this, myPopup);
```

15.1.1.21.3 addEx

[Function]

```
public static add(
    element: JQuery | undefined | null,
    options?: TopMostLayer.IOptions
): boolean;
```

Inserts an HTML element into the overlay area above the normal visualization. This function is intended for use from Code-behind [▶ 2120].

A reference to the element should be saved in order to be able to call the removeEx() [▶ 2066] function.

Parameter

Name	Type	Description
element	JQuery [▶ 2108]	JQuery collection with an HTML element.
options [optional]	TopMostLayer.IOptions [▶ 2067]	Options for extended functionalities.

Return value

Type	Description
boolean [▶ 2107]	Result as to whether the call was successful

Note: Available since version 1.10

Example - JavaScript

```
var myPopup = $('<div style="background:white;padding:10px;">Hello World<div>');
TcHmi.TopMostLayer.addEx(myPopup, {
    centerHorizontal: true,
    centerVertical: true,
    removeCb: (data) => {
        if (data.canceled) {
            // user clicked on background
        }
    }
});
```

15.1.1.21.4 removeEx

[Function]

```
public static removeEx(
    element: JQuery
): JQuery;
```

Removes an HTML element from the overlay area. This function is intended for use from Code-behind [▶ 2120].

If the element is not currently in the overlay area, it is returned without any changes.

Parameter

Name	Type	Description
element	JQuery [▶ 2108]	JQuery collection with an HTML element.

Return value

Type	Description
JQuery [▶ 2108]	JQuery collection with the removed HTML element.

Note: Available since version 1.10

Example - JavaScript

```
var myPopup = $('<div style="background:white;padding:10px;">Hello World<div>');
TcHmi.TopMostLayer.addEx(myPopup, {
    centerHorizontal: true,
    centerVertical: true,
    removeCb: (data) => {
        if (data.canceled) {
            // user clicked on background
        }
    }
});
TcHmi.TopMostLayer.removeEx(myPopup);
```

15.1.1.21.5 IOptions

[Interface]

```
export interface IOptions {
    resizeCb?: (data: TopMostLayer.IResizeResultObject) => void;
    removeCb?: (data: TopMostLayer.IElemRemoveResultObject) => void;
    centerHorizontal?: boolean;
    centerVertical?: boolean;
}
```

Properties

Name	Type	Description
centerHorizontal [Optional]	boolean [▶ 2107]	The element is centered horizontally and held centered when the window is resized.
centerVertical [Optional]	boolean [▶ 2107]	The element is centered vertically and held centered when the window is resized.
resizeCb [Optional]	(data: TopMostLayer.IResizeResultObject [▶ 2067]) => void	Callback which is called when the element has potentially changed in size. A parameter is passed to the callback.
removeCb [Optional]	(data: TopMostLayer.IElemRemoveResultObject [▶ 2068]) => void	Callback which is called when the element is removed from the overlay area. A parameter is passed to the callback. This callback is called regardless of the cause of the removal.

Note: Available since version 1.10

15.1.1.21.6 IResizeResultObject

[Interface]

```
export interface IResizeResultObject extends TcHmi.IResultObject {
    element: JQuery;
    parentPixelSize: {
        width: number;
        height: number;
    };
};
```

This interface provides detailed information in the resize callback of the TopMostLayer.

Extends the interface IResultObject [▶ 2094] by the following properties:

Properties

Name	Type	Description
element	jQuery [▶ 2108]	Enlarged element
parentPixelSize	{width: number [▶ 2108], height: number [▶ 2108]}	Width and height of the DOM parent element

Note: Available since version 1.10

15.1.1.21.7 IElemRemoveResultObject

[Interface]

```
export interface IElemRemoveResultObject extends TcHmi.IResultObject {
    element: JQuery;
    canceled: boolean;
}
```

This interface provides detailed information in the remove callback of the TopMostLayer.

Extends the interface IResultObject [▶ 2094] by the following properties:

Properties

Name	Type	Description
element	jQuery [▶ 2108]	Element
canceled	boolean [▶ 2107]	false: The element was removed from the TopMostLayer via an API call. true: The user clicked next to the element, or the corresponding control was removed from the DOM.

Note: Available since version 1.10

15.1.1.22 ValueConverter

[Class]

Static functions

Name	Description
toBoolean [▶ 2069]	Converts any compatible value to a Boolean value.
toNumber [▶ 2070]	Converts any compatible value to a number.
toObject [▶ 2070]	Converts any compatible value to an object or array.
toSchemaType [▶ 2071]	Converts any compatible value to a type defined by a schema object.
toString [▶ 2071]	Converts any compatible value to a string.
toType [▶ 2072]	Converts any compatible value to a type defined by a type schema.
toAngleUnit [▶ 2072]	Converts any compatible value to an angle unit.
toBorderStyleValue [▶ 2072]	Converts any compatible value to a border style.
toDimensionUnit [▶ 2073]	Converts any compatible value to a dimension unit.
toFontStyle [▶ 2073]	Converts any compatible value into an enumeration of a font style.
toFontWeight [▶ 2074]	Converts any compatible value into an enumeration of a font weight.
toHorizontalAlignment [▶ 2074]	Converts any compatible value into an enumeration of a horizontal alignment.
toScaleModeString [▶ 2075]	Converts any compatible value into an enumeration of a scaling instruction.
toSizeMode [▶ 2075]	Converts any compatible value into an enumeration of a Width/HeightMode string.
toSizeModeWithContent [▶ 2075]	Converts any compatible value into an enumeration of an extended Width/HeightMode string.
toToggleState [▶ 2076]	Converts any compatible value into an enumeration of a ToggleState string.
toVerticalAlignment [▶ 2076]	Converts any compatible value into an enumeration of a vertical alignment.
toVisibility [▶ 2077]	Converts any compatible value into a visibility enumeration.

Note: Available since version 1.8

15.1.1.22.1 toBoolean

[Function]

```
public static toBoolean(
    value: any,
    defaultValue: boolean | null = null
): boolean | null;
```

Converts any compatible value to a Boolean value.

Parameter

Name	Type	Description
value	any [▶ 2107]	The value to convert.
defaultValue	boolean [▶ 2107], null [▶ 2108]	This value is returned if the first parameter cannot be converted. Null applies if this parameter is not passed.

Return value

Type	Description
boolean [▶ 2107], null [▶ 2108]	Returns the Boolean value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.2 toNumber

[Function]

```
public static toNumber(
    value: any,
    defaultValue: number | null = null
): number | null;
```

Converts any compatible value to a number.

Parameter

Name	Type	Description
value	any [▶ 2107]	The value to convert.
defaultValue	number [▶ 2108], null [▶ 2108]	This value is returned if the first parameter cannot be converted. Null applies if this parameter is not passed.

Return value

Type	Description
number [▶ 2108], null [▶ 2108]	Returns the appropriate numerical value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.3 toObject

[Function]

```
public static toObject<T extends object>(
    value: any,
    defaultValue: T | null = null
): T | null;
```

Converts any compatible value to an object or array. If the parameter is already an object or an array, it is returned directly; if it is a string, it is converted into an object or array via JSON.parse.

Note: no type checking is performed during this process. This should be done downstream, for example via TcHmi.isSolidColor [▶ 2084], Array.isArray or similar.

Parameter

Name	Type	Description
value	any [▶ 2107]	The value to convert. Example: A JSON string.
defaultValue	any [▶ 2107], null [▶ 2108]	This value is returned if the first parameter cannot be converted. Null applies if this parameter is not passed.

Return value

Type	Description
T, null [▶ 2108]	Returns an object/array or defaultValue or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.4 toSchemaType

[Function]

```
public static toSchemaType(
    value: any,
    schema: TcHmi.JsonSchema | null
): any | null;
```

Converts any compatible value to a type defined by a schema object.

Parameter

Name	Type	Description
value	any [▶ 2107]	The value to convert.
schema	TcHmi.JsonSchema	The schema object.

Return value

Type	Description
any [▶ 2107], null [▶ 2108]	The value corresponding to the schema object or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.5 toString

[Function]

```
public static toString(
    value: any | null | undefined ,
    defaultValue: string | null = null
): string | null;
```

Converts any compatible value to a string.

Parameter

Name	Type	Description
value	any [▶ 2107]	The value to convert.
defaultValue	string [▶ 2108], null [▶ 2108]	This value is returned if the first parameter cannot be converted. Null applies if this parameter is not passed.

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	Returns the string value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.6 toType

[Function]

```
public static toType(
    value: any,
    typeString: string
): any | null;
```

Converts any compatible value to a type defined by a type schema.

Parameter

Name	Type	Description
value	any [▶ 2107]	The value to convert.
typeString	string [▶ 2108]	The name of the type schema.

Return value

Type	Description
any [▶ 2107], null [▶ 2108]	The value corresponding to the type schema name or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.7 toAngleUnit

[Function]

```
public static toAngleUnit (
    value: string,
    defaultValue: AngleUnit | null = null
): AngleUnit | null;
```

Checks if a value is an angle unit and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	AngleUnit [▶ 2080], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
AngleUnit [▶ 2080], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.8 toBorderStyleValue

[Function]

```
public static toBorderStyleValue(
    value: string,
    defaultValue: BorderStyleValue | null = null
): number | null;
```

Checks if a value is a border style and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	BorderStyleValue [▶ 2082], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
BorderStyleValue [▶ 2082], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.9 toDimensionUnit

[Function]

```
public static toDimensionUnit(
    value: string,
    defaultValue: DimensionUnit | null = null
): DimensionUnit | null;
```

Checks whether a value is a dimension unit and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	DimensionUnit [▶ 2086], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
DimensionUnit [▶ 2086], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.10 toFontStyle

[Function]

```
public static toFontStyle(
    value: string,
    defaultValue: FontStyle | null = null
): FontStyle | null;
```

Checks if a value is a font style and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	FontStyle [▶ 2087], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
FontStyle [▶ 2087], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.11 toFontWeight

[Function]

```
public static toFontWeight(
    value: string,
    defaultValue: FontWeight | null = null
): FontWeight | null;
```

Checks if a value is a font weight and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	FontWeight [▶ 2088], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
FontWeight [▶ 2088], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.12 toHorizontalAlignment

[Function]

```
public static toHorizontalAlignment(
    value: string,
    defaultValue: HorizontalAlignment | null = null
): HorizontalAlignment | null;
```

Checks if a value is a horizontal alignment and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	HorizontalAlignment [▶ 2089], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
HorizontalAlignment [▶ 2089], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.13 toScaleModeString

[Function]

```
public static toScaleModeString(
    value: string,
    defaultValue: ScaleModeString | null = null
): ScaleModeString | null;
```

Checks if a value is a scale and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	ScaleModeString [▶ 2096], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
ScaleModeString [▶ 2096], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.14 toSizeMode

[Function]

```
public static toSizeMode(
    value: string,
    defaultValue: SizeMode | null = null
): SizeMode | null;
```

Checks if a value is a size mode and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	SizeMode [▶ 2096], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
SizeMode [▶ 2096], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.15 toSizeModeWithContent

[Function]

```
public static toSizeModeWithContent (
    value: string,
    defaultValue: SizeModeWithContent | null = null
): SizeModeWithContent | null;
```

Checks if a value is an extended size mode and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	SizeModeWithContent [▶ 2096], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
SizeModeWithContent [▶ 2096], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.16 toToggleState

[Function]

```
public static toToggleState(
    value: string,
    defaultValue: ToggleState | null = null
): ToggleState | null;
```

Checks if a value is a ToggleState and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	ToggleState [▶ 2097], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
ToggleState [▶ 2097], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.17 toVerticalAlignment

[Function]

```
public static toVerticalAlignment(
    value: string,
    defaultValue: VerticalAlignment | null = null
): VerticalAlignment | null;
```

Checks if a value is a vertical alignment and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	VerticalAlignment [▶ 2089], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
VerticalAlignment [▶ 2089], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.22.18 toVisibility

[Function]

```
public static toVisibility(
    value: string,
    defaultValue: Visibility | null = null
): Visibility | null;
```

Checks if a value is a visibility and returns it.

Parameter

Name	Type	Description
value	string [▶ 2108]	The value to convert.
defaultValue	Visibility [▶ 2097], null [▶ 2108]	This value is returned if the first parameter is not valid. Null applies if this parameter is not passed.

Return value

Type	Description
Visibility [▶ 2097], null [▶ 2108]	Returns the appropriate value or null in the event of an error.

Note: Available since version 1.8

15.1.1.23 View

[Class]

Access to the current view level.

Static functions

Name	Description
load [▶ 2078]	Loads a *.view file based on a relative path in the project directory.
get [▶ 2078]	Returns the current view object.

Note: Available since version 1.8

15.1.1.23.1 get

[Function]

```
public static get(): TcHmi.Controls.System.baseTcHmiControl | null;
```

Returns the current view object.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.Controls.System.baseTcHmiControl [▶ 1118], null [▶ 2108]	The current view control object or null in the event of an error.

Note: Available since version 1.8

Example - JavaScript

```
var v1 = TcHmi.View.get();
TcHmi.Log.debug(v1.getId());
TcHmi.View.load('View2.view', function (data) {
    var v2 = TcHmi.View.get();
    TcHmi.Log.debug(v2.getId());
});
```

15.1.1.23.2 load

[Function]

```
public static load(
    url: string,
    callback: null | (
         (data: TcHmi.IResultObject) => void
    ) = null
): void;
```

Loads a *.view file based on a relative path in the project directory.

Parameter

Name	Type	Description
url	string [▶ 2108]	The relative path to a *.view file from the root directory of the application.
callback [Optional]	null [▶ 2108], ((data: TcHmi.IResultObject [▶ 2094]) => void)	Callback function, which is called after loading the view or in the event of an error.

Return value

Type	Description
void	No return value

Note: Available since version 1.8

Example - JavaScript

```
var v1 = TcHmi.View.get();
TcHmi.Log.debug(v1.getId());
TcHmi.View.load('View2.view', function (data) {
```

```
    var v2 = TcHmi.View.get();
    TcHmi.Log.debug(v2.getId());
});
```

15.1.1.24 Type

[Class]

Access type definitions.

Static functions

Name	Description
getSchema [▶ 2079]	Returns the schema object for a type definition.

Note: Available since version 1.10

15.1.1.24.1 getSchema

[Function]

```
public static getSchema(): TcHmi.JsonSchema | null;
```

Returns the schema object for a type definition.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
TcHmi.JsonSchema [▶ 2102], null [▶ 2108]	The schema object.

Note: Available since version 1.10

Example - JavaScript

```
var schema = TcHmi.Type.getSchema('tchmi:general#/definitions/String');
if (schema) {
    console.log(schema); // {default: "", id: "tchmi:general#/definitions/String", type: "string"}
}
```

15.1.1.25 AccessControl

[Interface]

```
export interface AccessControl {
    accessright: string;
    group: string;
    permission: 'Allow' | 'Deny' | 'Inherit';
}
```

Properties

Name	Type	Description
accessright	string [▶ 2108]	Name of the access right, which is described here.
group	string [▶ 2108]	Name of the group that addresses this access right.
permission	'Allow', 'Deny', 'Inherit'	Fixed strings that define whether a right should be granted, denied or undefined.

Schema: tchmi:framework#/definitions/AccessConfig [▶ 2192]

Note: Available since version 1.8

15.1.1.26 AngleUnit

[Interface]

```
'deg' | 'rad' | 'turn' | 'grad'
```

Keywords as strings.

Note: Available since version 1.8

15.1.1.27 Background

[Interface]

```
export interface Background {
    color: TcHmi.Color | null;
    image: string | null;
    imageWidth: number | null;
    imageWidthUnit: TcHmi.DimensionUnit;
    imageHeight: number | null;
    imageHeightUnit: TcHmi.DimensionUnit;
    imageHorizontalAlignment: TcHmi.HorizontalAlignment;
    imageVerticalAlignment: TcHmi.VerticalAlignment;
    imagePadding: TcHmi.FourSidedCss | null;
}
```

Properties

Name	Type	Description
color	Color [▶ 2084], null [▶ 2108]	Definition of the color
image	string [▶ 2108]	Definition of a background image for the background image
imageWidth	number [▶ 2108], null [▶ 2108]	Numerical value of the background width of the background image
imageWidthUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the background width of the background image
imageHeight	number [▶ 2108], null [▶ 2108]	Numerical value of the background height of the background image
imageHeightUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the background height of the background image
imageHorizontalAlignment	TcHmi.HorizontalAlignment [▶ 2089]	Horizontal alignment of the background image
imageVerticalAlignment	TcHmi.VerticalAlignment [▶ 2089]	Vertical alignment of the background image
imagePadding	TcHmi.FourSidedCss [▶ 2088], null [▶ 2108]	Padding of the background image.

Note: Available since version 1.8

15.1.1.28 BorderRadius

[Interface]

```
export interface BorderRadius {
    topLeft: number;
    topLeftUnit?: TcHmi.DimensionUnit;
    topRight: number;
    topRightUnit?: TcHmi.DimensionUnit;
    bottomRight: number;
    bottomRightUnit?: TcHmi.DimensionUnit;
    bottomLeft: number;
    bottomLeftUnit?: TcHmi.DimensionUnit;
}
```

Properties

Name	Type	Description
topLeft	number [▶ 2108]	Numerical value of the radius in the top left corner
topLeftUnit (optional)	TcHmi.DimensionUnit [▶ 2086]	Unit of the radius in the top left corner
topRight	number [▶ 2108]	Numerical value of the radius in the top right corner
topRightUnit (optional)	TcHmi.DimensionUnit [▶ 2086]	Unit of the radius in the top right corner
bottomRight	number [▶ 2108]	Numerical value of the radius in the bottom right corner
topLeftUnit (optional)	TcHmi.DimensionUnit [▶ 2086]	Unit of the radius in the bottom right corner
bottomLeft	number [▶ 2108]	Numerical value of the radius in the bottom left corner
bottomLeftUnit (optional)	TcHmi.DimensionUnit [▶ 2086]	Unit of the radius in the bottom left corner

See also

Schema: tchmi:framework#/definitions/BorderRadius [▶ 2194]

Note: Available since version 1.8

15.1.1.29 BorderStyle

[Interface]

```
export interface BorderStyle {
    left: TcHmi.BorderStyleValue;
    top: TcHmi.BorderStyleValue;
    right: TcHmi.BorderStyleValue;
    bottom: TcHmi.BorderStyleValue;
}
```

Properties

Name	Type	Description
left	TcHmi.BorderStyleValue [▶ 2082]	Definition of the border on the left side.
top	TcHmi.BorderStyleValue [▶ 2082]	Definition of the border on the top side.
right	TcHmi.BorderStyleValue [▶ 2082]	Definition of the border on the right side.
bottom	TcHmi.BorderStyleValue [▶ 2082]	Definition of the border on the bottom side.

See also

Schema: tchmi:framework#/definitions/BorderStyle [▶ 2196]

Note: Available since version 1.8

15.1.1.30 BorderStyleValue

[Interface]

```
'Solid' | 'Dashed' | 'Dotted' | 'None'
```

Keywords as strings.

Note: Available since version 1.8

15.1.1.31 BorderWidth

[Interface]

```
export interface BorderWidth {
    left: number;
    leftUnit?: 'px';
    top: number;
    topUnit?: 'px';
    right: number;
    rightUnit?: 'px';
    bottom: number;
    bottomUnit?: 'px';
}
```

Properties

Name	Type	Description
left	number [▶ 2108]	Numerical value that is applicable on the left side.
leftUnit	'px'	Only pixels are possible as unit.
top	number [▶ 2108]	Numerical value that is applicable on the top side.
topUnit	'px'	Only pixels are possible as unit.
right	number [▶ 2108]	Numerical value that is applicable on the right side.
rightUnit	'px'	Only pixels are possible as unit.
bottom	number [▶ 2108]	Numerical value that is applicable on the bottom side.
bottomUnit	'px'	Only pixels are possible as unit.

Note: Available since version 1.8

15.1.1.32 BoxShadow

[Interface]

```
export interface BoxShadow {
    color: TcHmi.SolidColor;
    offsetX: number;
    offsetXUnit?: TcHmi.DimensionUnit;
    offsetY: number;
    offsetYUnit?: TcHmi.DimensionUnit;
    inset?: boolean;
    blur?: number;
    blurUnit?: TcHmi.DimensionUnit;
    spread?: number;
    spreadUnit?: TcHmi.DimensionUnit;
}
```

Properties

Name	Type	Description
color	TcHmi.SolidColor [▶ 2084]	Color of the shadow
offsetX	number [▶ 2108]	Numerical value of the distance of the shadow in x-direction
offsetXUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the distance of the shadow in x-direction
offsetY	number [▶ 2108]	Numerical value of the distance of the shadow in y-direction
offsetYUnit	TcHmi.DimensionUnit [▶ 2086]	Unit of the distance of the shadow in y-direction
inset [optional]	boolean [▶ 2107]	Defines whether the shadow should be directed inwards.
blur [optional]	number [▶ 2108]	Numerical value of shadow blur
blurUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unity of the shadow blur
spread [optional]	number [▶ 2108]	Numerical value of the enlargement or reduction of the shadow
spreadUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit for enlarging or reducing the shadow

See also

Schema: tchmi:framework#/definitions/BoxShadowList [▶ 2197]

Note: Available since version 1.8

15.1.1.33 Comparison

[Interface]

```
export interface Comparison {
    path?: string;
    comparator: '==' | '!=' | '<' | '>' | '<=' | '>=' | 'contains' | 'contains not' | '== [ignore
case]' | '!= [ignore case]' | 'contains [ignore case]' | 'contains not [ignore case]';
    value: string | number | Date | boolean;
}
```

Defines a comparison between a variable and a specified value. If the variable contains an object, `path` can be used to address a property. If the variable contains a primitive data type, `path` must not be set.

This interface is used in filters [▶ 2087].

Properties

Name	Type	Description
path [optional]	string [▶ 2108]	The path to the property to be compared. Nesting is done via :: addressed, array access via []. Example: myObject::myArray[0]::myProperty
comparator	string [▶ 2108]	Specifies how to compare. Valid values are: == and != for numbers, dates, strings and truth values, <, >, <= and >= for numbers and dates and contains, contains not, == [ignore case], != [ignore case], contains[ignore case] and contains not[ignore case] for strings.
value	string [▶ 2108], number [▶ 2108], Date [▶ 2108], boolean [▶ 2107]	The value to be compared with.

Note: Available since version 1.10

15.1.1.34 Color

[Interface]

```
export type Color = SolidColor | LinearGradientColor;
```

See also

API: SolidColor [▶ 2084], LinearGradientColor [▶ 2085]

Schema: tchmi:framework#/definitions/Color [▶ 2213]

Note: Available since version 1.8

15.1.1.35 SolidColor

[Interface]

```
export interface SolidColor {
    color: string;
}
```

Properties

Name	Type	Description
color	string [▶ 2108]	Definition of the color (see CSS definition)

See also

Schema: tchmi:framework#/definitions/SolidColor [▶ 2253]

Note: Available since version 1.8

15.1.1.36 isSolidColor

[Function]

```
export function isSolidColor(checkColor: any): boolean;
```

Checks if the passed parameter is of type TcHmi.SolidColor [▶ 2084].

Parameter

Name	Type	Description
checkColor	any [▶ 2107]	Variable to be tested

Return value

Type	Description
boolean [▶ 2107]	Returns true if it is a valid SolidColor, otherwise false.

Note: Available since version 1.8

Example - JavaScript

```
if (TcHmi.isSolidColor(this.__myColor)){
    // use the color
} else {
    // reset color
};
```

15.1.1.37 LinearGradientColor

[Interface]

```
export interface LinearGradientColor {
    angle: number;
    stopPoints: TcHmi.StopPoint[];
}
```

Properties

Name	Type	Description
angle	number [▶ 2108]	Numerical value for the angle
stopPoints	TcHmi.StopPoint [▶ 2096][]	Definition of stop points

See also

Schema: tchmi:framework#/definitions/LinearGradientColor [▶ 2240]

Note: Available since version 1.8

15.1.1.38 isLinearGradientColor

[Function]

```
export function isLinearGradientColor(checkColor: any): boolean;
```

Checks if the passed parameter is of type TcHmi.linearGradientColor [▶ 2085].

Parameter

Name	Type	Description
checkColor	any [▶ 2107]	Variable to be tested

Return value

Type	Description
boolean [▶ 2107]	Returns true if it is a valid LinearGradientColor, otherwise false.

Note: Available since version 1.8

Example - JavaScript

```
if (TcHmi.isLinearGradientColor(this.__myColor)){
    // use the color
} else {
    // reset color
};
```

15.1.1.39 Dictionary

[Interface]

```
export interface Dictionary<T> {
    [index: string]: T | undefined;
}
```

An object [▶ 2108], whose properties must be of type `T` and can only be addressed with strings.

Note: Available since version 1.8

15.1.1.40 DimensionUnit

[Interface]

```
'px' | '%'
```

Pixel or percent as string [▶ 2108].

See also

Schema: tchmi:framework#/definitions/MeasurementUnit [▶ 2242]

Note: Available since version 1.8

15.1.1.41 Endianness

[Enumeration]

```
export enum Endianness {
    LittleEndian,
    BigEndian
}
```

Endianness defines the order in which the bytes that make up a data point are stored. Possible sequences are smallest byte first or largest byte first.

The names of the enumeration exist at runtime. For example, you can use `TcHmi.Endianness.LittleEndian` directly.

Values

Name	Value	Description
LittleEndian	0	Byte with the smallest value first. Example: The number 41394 (hexadecimal: 0xA1B2) is stored in two bytes as [0xB2, 0xA1].
BigEndian	1	Byte with the largest value first. Example: The number 41394 (hexadecimal: 0xA1B2) is stored in two bytes as [0xB2, 0xA1].

Note: Available since version 1.10

15.1.1.42 FontStyle

[Interface]

```
'Normal' | 'Italic' | 'Oblique' | 'Auto'
```

Keywords as strings.

If `Auto` is specified, the parent value, for example from a central CSS file that specifies the style for the <html> element, applies.

See also

Schema: tchmi:framework#/definitions/FontStyle [▶ 2228]

Note: Available since version 1.8

The value "`Auto`" is available from version 1.10

15.1.1.43 Filter

[Interface]

```
export interface Filter extends Array<Comparison | LogicOperator | Filter> { }
```

Defines a set of conditions that an object must fulfill.

A filter is an array that can contain Comparisons [▶ 2083], LogicOperators [▶ 2095] and other filters. There must always be a logic operator between two comparisons, two filters or a comparison and a filter. In this way, a logical expression is formed, with nested filters performing the function of parentheses.

Example - JavaScript

```
var filter = [
    {
        path: 'domain',
        comparator: '==',
        value: 'TcHmiEventLogger'
    },
    {
        logic: 'AND'
    },
    {
        path: 'severity',
        comparator: '>=',
        value: TcHmi.Server.Events.Severity.Warning
    },
    {
        logic: 'OR',
    },
    {
        path: 'type',
        comparator: '==',
        value: TcHmi.Server.Events.Type.Alarm
    },
    {
        logic: 'AND'
    },
    [
        {
            path: 'timeConfirmed',
            comparator: '==',
            value: new Date(0)
        },
        {
            logic: 'OR'
        },
        {
            path: 'timeConfirmed',
            comparator: '==',
            value: new Date(0)
```

```
        }
    ]
]
```

In this example, events are filtered. All events of the EventLogger with a severity of warning or higher and all alarms that were not acknowledged or not deleted fit through this filter. The filter corresponds to this logical expression:

```
event.domain == 'TcHmiEventLogger'
&& event.severity >= 2
|| event.type == 1
&& (
    event.timeConfirmed.getTime() == 0
    || event.timeCleared.getTime() == 0
)
```

Note: Available since version 1.10

15.1.1.44 FontWeight

[Interface]

```
'Normal' | 'Bold' | 'Auto'
```

Keywords as strings.

If `Auto` is specified, the parent value, for example from a central CSS file that specifies the style for the <html> element, applies.

See also

Schema: tchmi:framework#/definitions/FontWeight [▶ 2228]

Note: Available since version 1.8

The value "`Auto`" is available from version 1.10

15.1.1.45 FourSidedCss

[Interface]

```
export interface FourSidedCss {
    left: number;
    leftUnit?: TcHmi.DimensionUnit;
    top: number;
    topUnit?: TcHmi.DimensionUnit;
    right: number;
    rightUnit?: TcHmi.DimensionUnit;
    bottom: number;
    bottomUnit?: TcHmi.DimensionUnit;
}
```

Properties

Name	Type	Description
left	number [▶ 2108]	Numerical value that is applicable on the left side.
leftUnit	TcHmi.DimensionUnit [▶ 2086]	Unit on the left side
top	number [▶ 2108]	Numerical value that is applicable on the top side.
topUnit	TcHmi.DimensionUnit [▶ 2086]	Unit on the top side
right	number [▶ 2108]	Numerical value that is applicable on the right side.
rightUnit	TcHmi.DimensionUnit [▶ 2086]	Unit on the right side
bottom	number [▶ 2108]	Numerical value that is applicable on the bottom side.
bottomUnit	TcHmi.DimensionUnit [▶ 2086]	Unit on the bottom side

Note: Available since version 1.8

15.1.1.46 HorizontalAlignment

[Interface]

```
'Left' | 'Center' | 'Right'
```

Keywords as strings.

See also

Schema: tchmi:framework#/definitions/HorizontalAlignment [▶ 2238]

Note: Available since version 1.8

15.1.1.47 VerticalAlignment

[Interface]

```
'Top' | 'Center' | 'Bottom'
```

Keywords as strings.

See also

Schema: tchmi:framework#/definitions/VerticalAlignment [▶ 2266]

Note: Available since version 1.8

15.1.1.48 Errors

[Enumeration]

```
export enum Errors {
   NONE = 0,
   ERROR = 1,
   E_PARAMETER_INVALID = 2,
   E_TIMEOUT = 3,
   E_EXCEPTION = 4,
   E_INVALID = 5,
   E_NOT_SUPPORTED = 100,
   E_NOT_ALLOWED = 110,
   E_UNKNOWN = 115,
   E_UNKNOWN_TYPE = 120,
   E_KEY_NOT_FOUND = 130,
```

```
   E_TYPE_INVALID = 150,
   E_WEBSOCKET_NOT_READY = 1000,
   E_WEBSOCKET_NOT_SUPPORTED = 1404,
   E_WEBSOCKET_OPEN_SERVER_LICENSE_CHECK_FAILED = 1500,
   E_WEBSOCKET_OPEN_SERVER_LICENSE_MISSING = 1501,
   E_WEBSOCKET_OPEN_SERVER_NO_ACCESS = 1502,
   E_SYMBOL_STATE_INVALID = 2000,
   E_SYMBOL_VALUE_INVALID = 2001,
   E_SYMBOL_RESOLVE_SCHEMA = 2002,
   E_SYMBOL_READONLY = 2003,
   E_SYMBOL_UNKNOWN = 2010,
   E_SYMBOL_INVALID_DATA_PROVIDER_ENTRY = 2020,
   E_SERVER_RESPONSE_ERROR = 3000,
   E_SERVER_COMMAND_ERROR = 3005,
   E_SERVER_INVALID_RESPONSE = 3010,
   E_SERVER_COMMANDS_MISSING = 3015,
   E_SERVER_COMMAND_MISSING = 3016,
   E_SERVER_READVALUE_MISSING = 3020,
   E_SERVER_WRITEVALUE_MISSING = 3025,
   E_SERVER_RESPONSE_MISSING = 3030,
   E_FUNCTION_MISSING_FUNCTION_REFERENCE = 4000,
   E_FUNCTION_MISSING_FUNCTION_DESCRIPTION = 4005,
   E_FUNCTION_INVALID_CONFIGURATION = 4010,
   E_FUNCTION_EXCEPTION = 4020,
   E_FUNCTION_UNKNOWN = 4030,
   E_FUNCTION_RESTPARAMETER_DEFINITION_MISSING = 4040,
   E_FUNCTION_EXPRESSION_PARSER_ERROR = 4100,
   E_FUNCTION_EXPRESSION_EXCEPTION = 4120,
   E_TRIGGER_ACTION_EXCEPTION = 5050,
   E_TRIGGER_JAVASCRIPT_EVAL_EXCEPTION = 5055,
   E_TRIGGER_FUNCTION_EXPRESSION_EXCEPTION = 5060,
   E_TRIGGER_RESOLVE_CONDITION_EXPRESSION_EXCEPTION = 5065,
   E_CONTROL_INSTANCE_NOT_FOUND = 6000,
   E_CONTROL_INVALID_CONFIGURATION = 6001,
   E_CONTROL_ATTRIBUTE_NOT_FOUND = 6005,
   E_CONTROL_ATTRIBUTE_INVALID_CONFIGURATION = 6010,
}
```

The names of the enumeration exist at runtime. For example, you can use `TcHmi.Errors.NONE` directly in a comparison.

Values

Name	Value	Description
NONE **Note:** Available since version 1.8	0	No error
ERROR **Note:** Available since version 1.8	1	Unspecific error
E_PARAMETER_INVALID **Note:** Available since version 1.10	2	A parameter is invalid.
E_TIMEOUT **Note:** Available since version 1.10	3	An action has exceeded the maximum execution time.
E_EXCEPTION **Note:** Available since version 1.10.1018.48	4	An exception has occurred.
E_INVALID **Note:** Available since version 1.10.1018.48	5	Invalid
E_NOT_SUPPORTED **Note:** Available since version 1.8	100	An operation is not supported.
E_NOT_ALLOWED **Note:** Available since version 1.8	110	The current user does not have the rights for an operation.
E_UNKNOWN **Note:** Available since version 1.10	115	Something is unknown.
E_UNKNOWN_TYPE **Note:** Available since version 1.10	120	Unknown data type
E_KEY_NOT_FOUND **Note:** Available since version 1.10	130	Entry could not be found in dictionary/map.
E_TYPE_INVALID **Note:** Available since version 1.10.1018.48	150	Invalid data types
E_WEBSOCKET_NOT_READY **Note:** Available since version 1.8	1000	The WebSocket connection to the server is not yet fully established.
E_WEBSOCKET_ABNORMAL_SHUTDOWN **Note:** Available since version 1.8	1006	The WebSocket connection to the server could not be established.
E_WEBSOCKET_NOT_SUPPORTED **Note:** Available since version 1.8	1404	The browser does not support WebSockets.
E_WEBSOCKET_OPEN_SERVER_LICENSE_CHECK_FAILED **Note:** Available since version 1.8	1500	The license test could not be completed successfully.
E_WEBSOCKET_OPEN_SERVER_LICENSE_MISSING **Note:** Available since version 1.8	1501	The server is currently not licensed.
E_WEBSOCKET_OPEN_SERVER_NO_ACCESS **Note:** Available since version 1.8	1502	The current user does not have the authorization to establish a WebSocket connection.
E_SYMBOL_STATE_INVALID **Note:** Available since version 1.8	2000	The symbol has an invalid status.
E_SYMBOL_VALUE_INVALID **Note:** Available since version 1.8	2001	The symbol has an invalid value.
E_SYMBOL_VALUE_EQUAL **Note:** Available since version 1.8	2002	The symbol has not changed its value.
E_SYMBOL_READONLY **Note:** Available since version 1.8	2003	No write access to the symbol is allowed.
E_SYMBOL_UNKNOWN	2010	Symbol is unknown

Name	Value	Description
Note: Available since version 1.10.1018.48		
E_SYMBOL_UNKNOWN_DATA_PROVIDER_ENTRY **Note:** Available since version 1.8	2020	The symbol is not known in the corresponding DataProvider.
E_SERVER_RESPONSE_ERROR **Note:** Available since version 1.10	3000	The server has responded with an error object at response level.
E_SERVER_COMMAND_ERROR **Note:** Available since version 1.10	3005	The server has responded with an error object at command level.
E_SERVER_INVALID_RESPONSE **Note:** Available since version 1.10	3010	Invalid response object.
E_SERVER_COMMANDS_MISSING **Note:** Available since version 1.10	3015	Server response object contains no commands property.
E_SERVER_COMMAND_MISSING **Note:** Available since version 1.10	3016	An expected command object is missing in the list of commands in the server response.
E_SERVER_READVALUE_MISSING **Note:** Available since version 1.10	3020	A readValue property is expected, but the command object does not contain one.
E_SERVER_WRITEVALUE_MISSING **Note:** Available since version 1.10	3025	A writeValue property is expected, but the command object does not contain one.
E_SERVER_RESPONSE_MISSING **Note:** Available since version 1.10	3030	Missing response object.
E_FUNCTION_MISSING_FUNCTION_REFERENCE **Note:** Available since version 1.10	4000	Reference to the function implementation is missing.
E_FUNCTION_MISSING_FUNCTION_DESCRIPTION **Note:** Available since version 1.10	4005	The description of a function is missing.
E_FUNCTION_INVALID_CONFIGURATION **Note:** Available since version 1.10	4010	The configuration of a function is incorrect.
E_FUNCTION_EXCEPTION **Note:** Available since version 1.10	4020	An error occurred when a function was executed.
E_FUNCTION_RESTPARAMETER_DEFINITION_MISSING **Note:** Available since version 1.10.1018.48	4040	Expected residual parameter definition missing.
E_FUNCTION_EXPRESSION_PARSER_ERROR **Note:** Available since version 1.10	4100	An error occurred while parsing a function expression.
E_FUNCTION_EXPRESSION_EXCEPTION **Note:** Available since version 1.10	4120	An error occurred while resolving a function expression.
E_TRIGGER_ACTION_EXCEPTION **Note:** Available since version 1.10	5050	An error occurred while resolving a trigger action.
E_TRIGGER_JAVASCRIPT_EVAL_EXCEPTION **Note:** Available since version 1.10	5055	An error occurred while resolving a JavaScript block in a trigger.

Name	Value	Description
E_TRIGGER_FUNCTION_EXPRESSION_EXCEPTION **Note:** Available since version 1.10	5060	An error occurred while resolving a function expression in a trigger.
E_TRIGGER_RESOLVE_CONDITION_EXPRESSION_EXCEPTION **Note:** Available since version 1.10	5065	An error occurred while resolving a condition in a trigger.
E_CONTROL_INSTANCE_NOT_FOUND **Note:** Available since version 1.10	6000	Control could not be found.
E_CONTROL_INVALID_CONFIGURATION **Note:** Available since version 1.10.1018.48	6001	Invalid control configuration
E_CONTROL_ATTRIBUTE_NOT_FOUND **Note:** Available since version 1.10	6005	Control attribute could not be found.
E_CONTROL_ATTRIBUTE_INVALID_CONFIGURATION **Note:** Available since version 1.10.1018.48	6010	Invalid control attribute configuration

Note: Available since version 1.8

15.1.1.49 IResultObject

[Interface]

```
export interface IResultObject {
    error: Errors;
    details?:IErrorDetails;
}
```

Properties

Name	Type	Description
error	Errors [▶ 2089]	Cause of the fault
details [optional] **Note:** Available since version 1.10	IErrorDetails [▶ 2094]	Optional details about error. Content depends on the function used and the error.

Note: Available since version 1.8

15.1.1.50 IErrorDetails

[Interface]

```
export interface IErrorDetails {
    code: TcHmi.Errors | number;
    domain?: string;
    message?: string;
    reason?: string;
    errors?: TcHmi.IErrorDetails[];
};
```

This interface enables a precise analysis of an error through a hierarchical structure. See also the function TcHmi.log.buildMessage [▶ 1899], which the error chain returns in formatted form.

Properties

Name	Type	Description
code	TcHmi.Errors [▶ 2089] | number [▶ 2108]	Numeric value referenced by a framework or server error.
domain [optional]	string [▶ 2108]	The name of the server or framework domain where the error occurred.
message [optional]	string [▶ 2108]	The text representation of the error
reason [optional]	string [▶ 2108]	Plain text description of the error
errors [optional]	TcHmi.IErrorDetails [▶ 2094][]	A list of errors that led to this current error.

Note: Available since version 1.10.1018.48

15.1.1.51 IFunction

[Interface]

```
export interface IFunction {
    objectType: 'Function';
    active: boolean;
    fn: string;
    fnParams: TcHmi.IFunction.Value[];
}
```

Properties

Name	Type	Description
objectType	'Function'	Keyword
active	boolean [▶ 2107]	Determines whether this function is active.
fn	string [▶ 2108]	Name of the function.
fnParams	TcHmi.IFunction.Value [▶ 1891][]	Array of function values for the function

See also

Schema: tchmi:framework#/definitions/Function [▶ 2229]

Note: Available since version 1.8

15.1.1.52 LogicOperator

[Interface]

```
export interface LogicOperator {
    logic: 'AND' | 'OR';
}
```

Defines a logical operation.

This interface is used in filters [▶ 2087].

Properties

Name	Type	Description
logic	string [▶ 2108]	AND or OR

Note: Available since version 1.10

15.1.1.53 ScaleModeString

[Interface]

```
'None' | 'ScaleToFill' | 'ScaleToFit' | 'ScaleToFitWidth' | 'ScaleToFitHeight'
```

Keywords as strings.

See also

Schema: tchmi:framework#/definitions/ScaleMode [▶ 2251]

Note: Available since version 1.8

15.1.1.54 SizeMode

[Interface]

```
'Value' | 'Parent'
```

Keywords as strings.

See also

Schema: tchmi:framework#/definitions/SizeMode [▶ 2252]

Note: Available since version 1.8

15.1.1.55 SizeModeWithContent

[Interface]

```
'Value' | 'Parent' | 'Content'
```

Keywords as strings.

See also

Schema: tchmi:framework#/definitions/SizeModeWithContent [▶ 2252]

Note: Available since version 1.8

15.1.1.56 StopPoint

[Interface]

```
export interface StopPoint {
    color: string;
    stop?: string;
}
```

Properties

Name	Type	Description
color	string [▶ 2108]	Definition of the color (see CSS definition)
stop [optional]	string [▶ 2108]	Length designation for a stop point (e.g. '60%')

See also

Schema: tchmi:framework#/definitions/LinearGradientColor [▶ 2240]

Note: Available since version 1.8

15.1.1.57 ToggleState

[Interface]

```
'Normal' | 'Active'
```

Keywords as strings

See also

Schema: tchmi:framework#/definitions/ToggleState [▶ 2254]

Note: Available since version 1.8

15.1.1.58 Transform

[Interface]

```
export type Transform = Translate | Rotate | Scale | Skew | Origin | Perspective;
```

See also

Schema: tchmi:framework#/definitions/TransformList [▶ 2255]

Control attribute: Transform [▶ 453]

Note: Available since version 1.8

15.1.1.59 Visibility

[Interface]

```
'Visible' | 'Hidden' | 'Collapsed'
```

Keywords as strings.

- 'Visible' is the normal display
- 'Hidden' hides the element, which nevertheless still occupies the allocated space.
- 'Collapsed' hides the element, which then no longer takes up any space.

See also

Schema: tchmi:framework#/definitions/Visibility [▶ 2268]

Note: Available since version 1.8

15.1.1.60 Perspective

[Interface]

```
export interface Perspective {
    transformType: 'Perspective';
    distance: number;
    distanceUnit?: TcHmi.DimensionUnit;
    origin?: {
        x?: number;
        xUnit?: TcHmi.DimensionUnit;
        y?: number;
        yUnit?: TcHmi.DimensionUnit;
    };
}
```

This interface implements the TcHmi.Transform [▶ 2097] interface.

Properties

Name	Type	Description
transformType	'Perspective'	Keyword
distance	number [▶ 2108]	Numeric value for the distance
distanceUnit	TcHmi.DimensionUnit [▶ 2086]	Unit for distance
origin [Optional]	{x [▶ 2108], xUnit [▶ 2086], y [▶ 2108], yUnit [▶ 2086]}	Numeric value and unit for the x and y position

Note: Available since version 1.8

15.1.1.61 Origin

[Interface]

```
export interface Origin {
    transformType: 'Origin';
    x?: number;
    xUnit?: TcHmi.DimensionUnit;
    y?: number;
    yUnit?: TcHmi.DimensionUnit;
    z?: number;
    zUnit?: TcHmi.DimensionUnit;
}
```

Defines the origin of all transformations.

This interface implements the TcHmi.Transform [▶ 2097] interface.

Properties

Name	Type	Description
transformType	'Origin'	Keyword
x [optional]	number [▶ 2108]	Numeric value of the x-coordinate
xUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the x-coordinate
y [optional]	number [▶ 2108]	Numeric value of the y-coordinate
yUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the y-coordinate
z [optional]	number [▶ 2108]	Numeric value of the z-coordinate
zUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the z-coordinate

Note: Available since version 1.8

15.1.1.62 Rotate

[Interface]

```
export interface Rotate {
    transformType: 'Rotate';
    angle: number;
    angleUnit?: TcHmi.AngleUnit;
    vector?: {
        x: number;
        y: number;
        z: number;
    };
}
```

Properties

Name	Type	Description
transformType	'Rotate'	Keyword
angle	number [▶ 2108]	Numeric value of the rotation
angleUnit [Optional]	TcHmi.AngleUnit [▶ 2080]	Unit of the rotation
vector [optional]	{x [▶ 2108], y [▶ 2108], z [▶ 2108]}	Defines a rotation in three-dimensional space

This interface implements the TcHmi.Transform [▶ 2097] interface.

15.1.1.63 Skew

[Interface]

```
export interface Skew {
    transformType: 'Skew';
    xAngle?: number;
    xAngleUnit?: TcHmi.AngleUnit;
    yAngle?: number;
    yAngleUnit?: TcHmi.AngleUnit;
}
```

Shrinks a control.

This interface implements the TcHmi.Transform [▶ 2097] interface.

Properties

Name	Type	Description
transformType	'Skew'	Keyword
xAngle [optional]	number [▶ 2108]	Numeric value of contraction in x-direction
xAngleUnit [optional]	TcHmi.AngleUnit [▶ 2080]	Unit of contraction in x-direction
yAngle [optional]	number [▶ 2108]	Numeric value of contraction in y-direction
yAngleUnit [optional]	TcHmi.AngleUnit [▶ 2080]	Unit of contraction in y-direction

15.1.1.64 Scale

[Interface]

```
export interface Scale {
    transformType: 'Scale';
    x?: number;
    y?: number;
    z?: number;
}
```

This interface implements the TcHmi.Transform [▶ 2097] interface.

Properties

Name	Type	Description
transformType	'Scale'	Keyword
x [optional]	number [▶ 2108]	Scaling in x-direction
y [optional]	number [▶ 2108]	Scaling in y-direction
z [optional]	number [▶ 2108]	Scaling in z-direction

Note: Available since version 1.8

15.1.1.65 Translate

[Interface]

```
export interface Translate {
    transformType: 'Translate';
    x?: number;
    xUnit?: TcHmi.DimensionUnit;
    y?: number;
    yUnit?: TcHmi.DimensionUnit;
    z?: number;
    zUnit?: TcHmi.DimensionUnit;
}
```

This interface implements the TcHmi.Transform [▶ 2097] interface.

Properties

Name	Type	Description
transformType	'Translate'	Keyword
x [optional]	number [▶ 2108]	Numeric value of the x translation
xUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the x translation
y [optional]	number [▶ 2108]	Numeric value of the y translation
yUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the y translation
z [optional]	number [▶ 2108]	Numeric value of the z translation
zUnit [optional]	TcHmi.DimensionUnit [▶ 2086]	Unit of the z translation

Note: Available since version 1.8

15.1.1.66 SymbolType

[Enumeration]

```
export enum SymbolType {
    Invalid = 0,
    Server = 10,
    Internal = 20,
    LocalizedText = 30,
    PartialParam = 40,
    TemplateParam = 50,
    Function = 60,
    Control = 70,
}
```

The names of the enumeration exist at runtime. For example, you can use `TcHmi.SymbolType.Internal` directly in a comparison.

Values

Name	Value	Description
Invalid	0	No valid symbol type
Server	10	Server symbol
Internal	20	Internal symbol
LocalizedText	30	Localized text
PartialParam	40	Parameters of a User Control [▶ 504]
TemplateParam	50	Parameters of an Action Template [▶ 59]
Function	60	Function symbol
Control	70	Control symbol

Note: Available since version 1.8

15.1.1.67 Context

[Interface]

```
export interface Context {
   success?: ((result?: any) => void);
   error?: ((
      error: Errors,
      details?: IErrorDetails
   ) => void);
```

Properties

Name	Type	Description
success [optional]	(result?: any [▶ 2107]) => void	Function for confirming success. Any return value can be transferred. Available in asynchronous context objects.
error [optional]	(error:Errors [▶ 2089], details?: IErrorDetails [▶ 2094]) => void	Function for reporting errors. An arbitrary error [▶ 2089] value and optional details about the error must be transferred in the form of an IErrorDetails [▶ 2094] object. Available in asynchronous context objects.

Note: Available since version 1.10

15.1.1.68 EventContext

[Interface]

```
export interface EventContext extends Context {
    name: string;
}
```

This interface extends TcHmi.Context [▶ 2101] from version 1.10

Properties

Name	Type	Description
name	string [▶ 2108]	Name of the event

Note: Available since version 1.8

15.1.1.69 DestroyFunction

[callback function]

```
interface DestroyFunction {
    (): void;
}
```

Various actions return a function of this type. It can be called to release resources such as an event registration.

Parameter

Name	Type	Description
-	-	-

Return value

Type	Description
void	No return value

Note: Available since version 1.8

15.1.1.70 JsonSchema

[Interface]

```
/*
    * MIT License
    *
    * Copyright (c) 2016 Richard Adams (https://github.com/enriched)
    *
    * Permission is hereby granted, free of charge, to any person obtaining a copy
    * of this software and associated documentation files (the "Software"), to deal
    * in the Software without restriction, including without limitation the rights
    * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
    * copies of the Software, and to permit persons to whom the Software is
    * furnished to do so, subject to the following conditions:
    *
    * The above copyright notice and this permission notice shall be included in all
    * copies or substantial portions of the Software.
    *
    * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
    * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
    * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
    * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
    * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
    * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
    * SOFTWARE.
    */

export interface JsonSchema {
    $ref?: string;
    ////////////////////////////////////////////////
    // Schema Metadata
    ////////////////////////////////////////////////
    /**
    * This is important because it tells refs where
    * the root of the document is located
    */
    id?: string;
    /**
    * It is recommended that the meta-schema is
    * included in the root of any JSON Schema
    */
    $schema?: string;
    /**
    * Title of the schema
    */
    title?: string;
    /**
    * Schema description
    */
    description?: string;
    /**
    * Default json for the object represented by
    * this schema
    */
    'default'?: any;

    ////////////////////////////////////////////////
    // Number Validation
    ////////////////////////////////////////////////
    /**
    * The value must be a multiple of the number
```

```
    * (e.g. 10 is a multiple of 5)
    */
    multipleOf?: number;
    maximum?: number;
    /**
    * If true maximum must be > value, >= otherwise
    */
    exclusiveMaximum?: boolean;
    minimum?: number;
    /**
    * If true minimum must be < value, <= otherwise
    */
    exclusiveMinimum?: boolean;

    ////////////////////////////////////////////////
    // String Validation
    ////////////////////////////////////////////////
    maxLength?: number;
    minLength?: number;
    /**
    * This is a regex string that the value must
    * conform to
    */
    pattern?: string;
    ////////////////////////////////////////////////
    // Array Validation
    ////////////////////////////////////////////////
    additionalItems?: boolean | JsonSchema;
    items?: JsonSchema | JsonSchema[];
    maxItems?: number;
    minItems?: number;
    uniqueItems?: boolean;

    ////////////////////////////////////////////////
    // Object Validation
    ////////////////////////////////////////////////
    maxProperties?: number;
    minProperties?: number;
    required?: string[];
    additionalProperties?: boolean | JsonSchema;
    /**
    * Holds simple JSON Schema definitions for
    * referencing from elsewhere.
    */
    definitions?: { [key: string]: JsonSchema };
    /**
    * The keys that can exist on the object with the
    * json schema that should validate their value
    */
    properties?: { [property: string]: JsonSchema };
    /**
    * The key of this object is a regex for which
    * properties the schema applies to
    */
    patternProperties?: { [pattern: string]: JsonSchema };
    /**
    * If the key is present as a property then the
    * string of properties must also be present.
    * If the value is a JSON Schema then it must
    * also be valid for the object if the key is
    * present.
    */
    dependencies?: { [key: string]: JsonSchema | string[] };
    ////////////////////////////////////////////////
    // Generic
    ////////////////////////////////////////////////
    /**
    * Enumerates the values that this schema can be
    * e.g.
    * {"type": "string",
    * "enum": ["red", "green", "blue"]}
    */
    'enum'?: any[];
    /**
    * The basic type of this schema, can be one of
    * [string, number, object, array, boolean, null, integer]
    * or an array of the acceptable types
    */
    type?: jsonDataTypeNames | jsonDataTypeNames[];
    ////////////////////////////////////////////////
```

```
    // Combining Schemas
    ////////////////////////////////////////////////
    allOf?: JsonSchema[];
    anyOf?: JsonSchema[];
    oneOf?: JsonSchema[];
    /**
    * The entity being validated must not match this schema
    */
    not?: JsonSchema;

    /**
    * Custom framework related extension to describe an instanceof class while type is 'object'.
    * See tchmi:framework#/definitions/Symbol for further information.
    */
    frameworkInstanceOf?: string;
    /**
    * If frameworkInstanceOf relates to TcHmi.Symbol frameworkSymbolSubType will contain the
excpected value type of the symbol.
    */
    frameworkSymbolSubType?: JsonSchema;
}
```

Properties

Name	Type	Description

Note: Available since version 1.8

15.1.2 tchmi_base64decode

[Function]

```
function tchmi_base64decode(input: string): string | null;
```

Converts a base64 string.

Please note that strings are encoded in JavaScript UTF16.

Parameter

Name	Type	Description
input	string [▶ 2108]	Base64 String

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	Converted string. See note below.

Note: Available since version 1.8:

In version 1.8, null is returned if the input string is not a valid Base64 format.

From version 1.10, an exception is thrown.

Example - JavaScript

```
TcHmi.Log.debug(tchmi_base64decode('VHdpbkNBVCBITUk=')); // 'TwinCAT HMI'
```

15.1.3 tchmi_base64encode

[Function]

```
function tchmi_base64encode(input: string): string | null;
```

Generates a base64 string.

Please note that strings are encoded in JavaScript UTF16.

Parameter

Name	Type	Description
input	string [▶ 2108]	String to convert

Return value

Type	Description
string [▶ 2108], null [▶ 2108]	String converted to Base64 format. See note below.

Note: Available since version 1.8

In version 1.8, null is returned if an error occurred during the conversion.

From version 1.10, an exception is thrown.

Example - JavaScript

```
TcHmi.Log.debug(tchmi_base64encode('TwinCAT HMI')); // 'VHdpbkNBVCBITUk='
```

15.1.4 tchmi_clone_object

[Function]

```
function tchmi_clone_object(obj: any): any | null;
```

Clones an object.

In JavaScript, simple data types (such as Booleans or numbers) are copied when used (by value). However, if a data type is complex, a reference is used (by reference), so that manipulation of the supposed copy also changes the original value.

This reference is desired for variables pointing to HTML elements or TcHmi controls, but not for values such as SolidColor [▶ 2084].

The function `tchmi_clone_object` can be used instead, since it creates a copy in most cases. Exceptions are HTML/SVG elements and TcHmi controls.

Parameter

Name	Type	Description
obj	any [▶ 2107]	any variable

Return value

Type	Description
any [▶ 2107], null [▶ 2108]	Copy of the passed variable

Note: Available since version 1.8

Example - JavaScript

```
var myNumber = 42;
var myNumberAssign = myNumber;
var myNumberClone = tchmi_clone_object(myNumber);
myNumber = 23;
TcHmi.Log.debug(
    'Original: ' + myNumber +
    ' Assign: ' + myNumberAssign +
    ' Clone: ' + myNumberClone); // Original: 23 Assign: 42 Clone: 42
```

```
var myColor = {color: 'red'};
var myColorAssign = myColor;
var myColorClone = tchmi_clone_object(myColor);
myColor.color = 'blue';
TcHmi.Log.debug(
    'Original: ' + myColor.color +
    ' Assign: ' + myColorAssign.color +
    ' Clone: ' + myColorClone.color); // Original: blue Assign: blue Clone: red
```

15.1.5 tchmi_equal

[Function]

```
function tchmi_equal(value1: any, value2: any, bTypeSafe: boolean = true): boolean;
```

Checks whether two variables are equal.

In JavaScript, simple data types (such as Booleans or numbers) are copied when used (by value). However, if a data type is complex, a reference is used (by reference), so that manipulation of the supposed copy also changes the original value.

A comparison for equality is not trivial with values such as SolidColor [▶ 2084], for example since the simple `===` operator checks the references but not the content.

The function `tchmi_equal` can be used instead.

Parameter

Name	Type	Description
value1	any [▶ 2107]	any variable
value2	any [▶ 2107]	any variable
bTypeSafe	boolean [▶ 2107]	

Return value

Type	Description
boolean [▶ 2107]	true if the content of the variable is the same.

Note: Available since version 1.8

Example - JavaScript

```
var myColor = {color: 'red'};
var myColor2 = {color: 'red'};
TcHmi.Log.debug(
    'JS compare: ' + (myColor === myColor2) +
    ' tchmi_equal: ' + tchmi_equal(myColor, myColor2)); // JS compare: false tchmi_equal: true
```

15.1.6 tchmi_path

[Function]

```
function tchmi_path(path: string): string;
```

Restores / standardizes a path.

Replaces all backslashes with normal slashes, replaces multiple slashes with a single slash, removes leading slashes.

Parameter

Name	Type	Description
path	string [▶ 2108]	any variable

Return value

Type	Description
string [▶ 2108]	Standardized value

Note: Available since version 1.8

Example - JavaScript

```
TcHmi.Log.debug(tchmi_path('\images///pumps\\pump.png'));
// => 'images/pumps/pump.png'
```

15.1.7 tchmi_create_guid

[Function]

```
function tchmi_create_guid(): string;
```

Creates a GUID (globally unique identifier) string. This function should be used if resources are to be generated dynamically and a name collision is to be avoided.

Parameter

Name	Type	Description

Return value

Type	Description
string [▶ 2108]	Created GUID

Note: Available since version 1.8

Example - JavaScript

```
var myGuid = tchmi_create_guid(); // for example the string "8e93cbea-4715-4011-84bd-566c5965c741"
```

15.1.8 any

[Type]

```
boolean | number | string | Object | Array | null | undefined
```

Any value without restriction.

Note: Available since version 1.8

15.1.9 Array

[Type]

A list of any number of values of any type that can be addressed using their index.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.10 boolean

[Type]

A Boolean value that can assume `true` or `false`.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.11 JQuery

[Type]

Foreign library element collection jQuery. Any number of HTML elements can be combined in this object.

Note: Available since version 1.8

15.1.12 Date

[Type]

A time stamp with a precision of 1 millisecond. Further information can be found in the MDN documentation.

Date specialized object [▶ 2108].

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.13 number

[Type]

Any numeric value.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.14 null

[Type]

The value null represents the intentional absence of a value.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.15 object

[Type]

Can have any number of properties of any type, which can be addressed via strings or numbers.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.16 string

[Type]

Any string.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.17 undefined

[Type]

A variable that has not yet been assigned a value has the value undefined. Functions that do not have a return value also return undefined.

Note: Defined in ECMAScript 1, supported by all JavaScript-enabled browsers

15.1.18 TcHmiCharting.LineGraph.Point

[Interface]

```
export interface Point{
    x: number;
    y: number;
}
```

Properties

Name	Type	Description
x	number [▶ 2108]	x-coordinate
y	number [▶ 2108]	y-coordinate

See also

Schema: tchmi:framework#/definitions/ChartLineGraphDataDefinitionList [▶ 2207]

Note: Available since version 1.8

15.1.19 TcHmiCharting.IAxis.Position

[Enumeration]

```
enum Position {
    Left = 0,
    Right = 1,
}
```

Properties

Name	Type	Description
Left	number [▶ 2108]	Numeric value for the value Left
Right	number [▶ 2108]	Numeric value for the value Right
0	string [▶ 2108]	String value for the value 0
1	string [▶ 2108]	String value for the value 1

Note: Available since version 1.8

15.2 Scripting languages

Scripting languages enable the client-side extension of a TwinCAT HMI project with complex logics that cannot be mapped using the engineering functions. The scripting languages provide the developer all options and functions of the respective scripting language. In addition, the scripting languages provide access to the Framework API [▶ 1111].

JavaScript is supported from version 1.8.

15.2.1 JavaScript

JavaScript enables the client-side extension of a TwinCAT HMI project with the functions of the JavaScript scripting language and the JavaScript library jQuery.

In a TwinCAT HMI project, a distinction is made between three types of JavaScript applications:

1. Action [▶ 2110]: An action consists of JavaScript code that can only be used within the Actions and Conditions Editor [▶ 50].
   Application: Execution of simple JavaScript functionalities.
2. Functions [▶ 2113]: A function consists of JavaScript code and an associated description; it is added separately to a project.
   Application: JavaScript execution in the Actions and Conditions Editor [▶ 50] and in the Properties window.
3. Code-behind [▶ 2120]: Code-behind consists of JavaScript code and is added to a project as a separate file.
   Application: Execution of complex JavaScript code.

15.2.1.1 Action

15.2.1.1.1 Create

An action is configured in the Actions and Conditions Editor [▶ 50] for an event. To do this, open any event and insert a JavaScript action in the Actions and Conditions Editor using drag & drop.

A JavaScript action consists of the following elements:

1 Edit Source Code: Clicking this button opens the Embedded JavaScript Editor, where the JavaScript statements can be programmed.

2 Wait mode: In this combo box you can select whether the JavaScript block is executed synchronously or asynchronously [▶ 2112].

3 Inject context object: When selecting the checkbox a context object (ctx) [▶ 2112] is available within the JavaScript code, which can be used to provide feedback about the end of an action. The context object is evaluated via a success branch or an error branch [▶ 2112], which can be shown or hidden by right-clicking on the action.

15.2.1.1.2 Editor

The editor for JavaScript actions offers the option to program and edit the JavaScript code. After programming, save the JavaScript code by clicking the **OK** button.

The JavaScript code within the JavaScript action is interpreted when the action is executed. Depending on the event, the action can be executed several times. Variables and functions declared in an action are only available within the action. Internal variables [▶ 73] can be read via the framework API [▶ 2036].

15.2.1.1.3 Use

JavaScript actions are typically used for simple JavaScript logics that have no major connection to other functionalities in the project. The use of a code-behind file is recommended in cases where several JavaScript actions are configured, since the code is available centrally in one place, i.e. not distributed across several places.

Typical applications

- Logging messages to the browser console
- Output of info messages in popup windows for debugging purposes
- Calling simple framework API functions

Special statements

- Breakpoint:

```
debugger;
```

This statement sets a breakpoint at the current location and stops the program when the development tools of the browser [▶ 2286] are open.

15.2.1.1.4 Wait mode

Via the wait mode options of the JavaScript action, the TwinCAT HMI framework can be informed whether the stored logic works synchronously or asynchronously; it is used to determine when the TwinCAT HMI framework considers execution to be finished.

Synchronous

The execution of the action is considered finished when the last line of JavaScript source text has been processed synchronously. If an exception occurs during execution, the error branch of the JavaScript action is executed. If the execution completed without errors, the success branch of the JavaScript action is executed.

The **Wait for completion** option has no effect for a synchronous logic. "Parallel" execution of synchronous logic, even in parts, is not possible.

Asynchronous

The execution of the action is considered completed if this has been signaled using the corresponding functions of a context object. If **Inject context object** is checked, an object with the name "ctx [▶ 2112]" is inserted into the context of the JavaScript block. This object contains functions for communicating the execution result.

If the **success** function is called, the success branch of the JavaScript action is executed. If the **error** function is called, the error branch of the JavaScript action is executed. The error branch of the JavaScript action is also executed if an exception occurs during execution.

With asynchronous logic, the **Wait for completion** option can be used to control whether the execution result is to be awaited when the subsequent actions are executed or whether they are to be executed in parallel with the JavaScript action.

15.2.1.1.4.1 Context

An asynchronous JavaScript [▶ 2110] trigger action must notify the context object (ctx) of success or failure of the execution. The context object has appropriate functions for this.

success

Indicates successful execution.

```
ctx.success("Result");
```

error

Indicates unsuccessful execution.

```
ctx.error(TcHmi.Errors.ERROR);
```

15.2.1.2 Function

15.2.1.2.1 Create

Create a new function via the project node of the TwinCAT HMI project.

Enter a name for the file.

A function consists of a JavaScript file and an associated JSON file, which defines engineering properties. Double-click the JSON file in the Solution Explorer to configure [▶ 2114] the function and its parameters.

15.2.1.2.2 Editor

The function and parameter editor is divided into a function and parameter area.

Function area:

1 Display name: Name of the function used [▶ 2116] in the Actions and Conditions Editor or in the Properties window.

2 Description: Description of the function that is displayed as a tooltip during mouseover in the engineering.

3 Return type: Return value of the function that is transferred to the calling point in the engineering.

4 Context: The context defines whether the function synchronous (None) or asynchronous (Asynchronous) [▶ 2112] is executed. Asynchronous automatically adds a context object [▶ 2119] as a transfer parameter to the function.

5 Internal name: Internal name of the function used in the JavaScript file.

6 Category: The category determines under which category the function is displayed in the Actions and Conditions Editor.

7 Return value description: Description of the return value that is displayed as a tooltip during mouseover in the engineering.

Parameter area:

8 Display name: Name of the parameter used [▶ 2116] in the Actions and Conditions Editor or in the Properties window.

9 Internal name: Internal name of the parameter used in the JavaScript file.

10 Description: Description of the parameter that is displayed as a tooltip during mouseover in the engineering.

11 Data type: Data type of the parameter.

12 Default value: Default value of the parameter that is used if no parameter is transferred.

13 Required: Indicates whether the parameter is required or can be transferred optionally. If the parameter is required and is not transferred, an error message is issued in the browser console.

14 Bindable: Bindable indicates whether the parameter can be bound to a symbol via the Select Binding Source window [▶ 91]. The following options are available:

- **No**: The parameter cannot be bound to a symbol.

- **Yes, pass value**: The parameter can be bound to a symbol. During binding, the value of the symbol is passed to the function (e.g. "20" for an integer).
- **Yes, pass symbol reference**: The parameter can be bound to a symbol. During binding, the reference to the symbol is passed to the function. The symbol can be read (.read() [▶ 2037]) and written (.write() [▶ 2039]) via the framework API "TcHmi.Symbol [▶ 2036]".

15 Rest: Rest indicates whether the last parameter is a rest parameter (var nNumber,...) and can therefore contain a variable list of arguments (cf. va_arg / va_list in C / C++).

Changes (e.g. newly added parameters) in the function and parameter editor can be automatically applied to the corresponding JavaScript file. The automatic application can be confirmed or rejected via a dialog.

15.2.1.2.3 Use

Functions allow the execution of JavaScript code that can be linked directly from the Properties window [▶ 49] in the engineering or used within the Actions and Conditions Editor [▶ 50]. This makes it possible to pass parameters to the function via the engineering.

Typical applications

- Conversion: Conversion of values within a function, e.g. conversion of units.
- Validation: Checking of values within a function and feedback on the validity of the value.

Several functions are already available [▶ 58] in the TwinCAT HMI engineering. To obtain an overview of the functions, open the "Functions" folder in the Actions and Conditions Editor.

15.2.1.2.3.1 Using JS functions

Functions that have been created are available for graphical programming in the Actions and Conditions Editor [▶ 50] under **Functions**.

In addition, the Control Properties functions can be used directly in the Properties window [▶ 49]. They can be found via IntelliSense.

15.2.1.2.4 Structure

A function is implemented within the TcHmi namespace. Implementation outside the namespace is not possible, since otherwise the function would not be available in the TwinCAT HMI framework. The function is registered in the framework via the framework API function registerFunction [▶ 1835], so that it is available in the engineering.

The following example implements the conversion of a temperature from Fahrenheit to Celsius and returns the calling point.

```
// required namespace TcHmi
(function (TcHmi) {

   // name and parameters of the function defined in json
   var FahrToCels = function (TempInFahr) {
      // variable for the return value
      var value = TempInFahr;

      // check the type of the parameter
      if (typeof TempInFahr === 'number') {
         // parameter is already a number

         // convert the parameter value to celsius
         value = ((TempInFahr - 32) * (5 / 9));
      }
      else if (typeof TempInFahr === 'string') {
```

```
         // parameter is a string

         // convert the parameter type and the parameter value
         value = ((parseFloat(TempInFahr) - 32) * (5 / 9));
      }

      // return the converted parameter
      return value;
   };

   // register the function in the framework
   TcHmi.Functions.registerFunction('FahrToCels', FahrToCels);
})(TcHmi);
```

References to the framework and jQuery within the function's JavaScript file are created to enhance IntelliSense support.

```
// Provider for a best effort Intellisense of Visual Studio 2017.
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Lib\jquery.d.ts" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\TcHmi.d.ts" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Controls\System\TcHmiControl\Source.d.ts" />

// Provider for a best effort Intellisense of Visual Studio 2013/2015.
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Lib\jquery\jquery.js" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\TcHmi.js" />
```

15.2.1.2.4.1 Context

An asynchronous JavaScript function [▶ 2113] must inform the context object (`ctx`) of success or error of execution. The context object has appropriate functions for this.

success

Indicates successful execution.

```
ctx.success("Result");
```

error

Indicates unsuccessful execution.

```
ctx.error(TcHmi.Errors.ERROR);
```

15.2.1.2.4.2 Using a function within a function

Within a JavaScript function, the TwinCAT HMI can use another JavaScript function. To do this, a reference to the function [▶ 1836] must be created via the framework API. The function can be used once its validity has been verified.

```
var MyCustomFunction2 = function (par1) {
   // get user function
   var myCustomFunction1 = TcHmi.Functions.getFunction('MyCustomFunction1');
   // check if the reference is valid
   if (myCustomFunction1)
   {
      var myParameter = "Sample";
      // function call
      var output = myCustomFunction1(myParameter);
      // ...
   }
};
```

15.2.1.3 Code-behind

15.2.1.3.1 Create

A code-behind file can be added at the project level or in a project folder. Any number of code-behind files can be added to the project.

Right-click the target node and click **Add New Item**. Select a code-behind file. The code-behind file can be opened by double-clicking in the JavaScript source code editor.

Code-behind sort order

If you have added several code-behind files to the project, it is possible to specify the order in the build process and thus during inclusion within the HTML file of the HMI. This is necessary if dependencies exist within the individual files.

Right-click on the project node of the HMI project and select **CodeBehind Sort Order and Theme Assignment**.

Within the editor, the sort order can be defined under the **JavaScript Files** tab. The order can be changed using the **Up** and **Down** buttons.

15.2.1.3.2 Use

A code-behind file can be used for JavaScript functionalities that are to apply globally to the entire project. Control events can be registered independently of the HMI side, based on the control ID. Therefore, a code-behind file is particularly suitable for mapping complex functionalities with JavaScript that cannot be performed using the graphical engineering.

Typical applications

- Summary of functionalities: Identical event configurations can be combined in a code-behind file and stored in functions to avoid redundancies (see Action Templates [▶ 59] in the Actions and Conditions Editor).
- Global variables: Saving of data used in multiple events to a global variable (see internal symbols [▶ 73] in the Configuration window).
- Evaluation of complex objects (e.g. JSON): Processing, validating and evaluating objects.
- Using the framework API: Access to the framework API functions [▶ 1112], which cannot be executed via the graphical engineering.
- Dynamic elements: Adding or removing dynamic elements at runtime (e.g. dynamically generate controls at runtime, see ControlFactory [▶ 1879]).
- Responsive design: Querying the current browser width, switching of grid configurations and loading of different pages depending on the browser width.

- Evaluating URL parameters: Loading different views depending on the browser URL.
- Disabling client interactions: Disabling interactions with other clients via the session ID of the browser.

15.2.1.3.3 Structure

A newly added code-behind file contains pre-configured JavaScript code.

The commentary lines (triple slash directives) are required for enhanced support of IntelliSenese in JavaScript:

```
// Provider for a best effort Intellisense of Visual Studio 2017.
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Lib\jquery.d.ts" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\TcHmi.d.ts" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Controls\System\TcHmiControl\Source.d.ts" />

// Provider for a best effort Intellisense of Visual Studio 2013/2015.
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Lib\jquery\jquery.js" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\TcHmi.js" />
```

The JavaScript code that is implemented within a code-behind file must be implemented within the TcHmi namespace.

```
(function (TcHmi) {
   // code
})(TcHmi);
```

The TwinCAT HMI framework is not available outside the namespace!

In a new code-behind file the global `onInitialized` event is registered by the framework as an example. Here you can place actions that are to be executed after the initialization of the framework.

```
// If you want to unregister an event outside the event code you need to use the return value of the
method register()
var destroyOnInitialized = TcHmi.EventProvider.register('onInitialized', function (e, data) {
   // This event will be raised only once, so we can free resources.
   // It's best practice to use destroy function of the event object within the callback function to
avoid conflicts.
   e.destroy();

   // ----------------------
   // Place your code here!
   // ----------------------
});
```

Within the code-behind file any events can be registered and deregistered. The registration of TwinCAT HMI framework events is realized via the EventProvider [▶ 1885] framework API. Registration via the EventProvider registers a callback to the event and returns a function for deregistering the event. The function for deregistering the event should be called to release system resources when the event is no longer needed (for example, Event Listener for a MouseOver event).

```
destroyOnInitialized();
```

Control events are registered using the Control ID (e.g. `TcHmiButton.onPressed`). A control event can be registered in several places. If an event is registered in the code-behind file, it is not displayed in the Designer under Events. It is therefore advisable to either register an event in a code-behind file or to configure it in the Designer for enhanced clarity.

15.3 Controls

JavaScript is supported from version 1.8.

15.3.1 Properties

15.3.1.1 Description.json

This section describes the structure of the description file Description.json.

Any changes in Description.json must be followed by a project build (see First Steps [▶ 2164]).

15.3.1.1.1 Introduction

TwinCAT HMI Engineering reads all properties of the control from the Description.json file. The properties allow you to transfer data to the control and vice versa. In TwinCAT HMI Engineering, the control can be configured via the Properties window [▶ 49] or via the Actions and Conditions editor. The developer can define additional attributes, besides the standard attributes of each framework control.

Description.json must correspond to a given JSON schema. The schema is located in the framework control project under Schema/Control/Description.json. The schema must not be changed. It is used to describe the syntax in Description.json.

In many cases the following user-specific properties are defined in Description.json:

- Attributes [▶ 2128] (data interfaces)
- Functions [▶ 2135]
- Events [▶ 2137]
- Access rights [▶ 2139]
- User-specific data types [▶ 2138]

15.3.1.1.2 Properties

Type: Specifies the data type of the property.
Required: Specifies whether the property is required for the framework control (yes) or is optional (no).

15.3.1.1.2.1 $schema

Type: String (relative path)
Required: No

```
"$schema": "./Schema/ControlDescription.Schema.json"
```

This property references the JSON schema that describes the structure of Description.json. The schema is located in the framework control project and does not have to be changed. Including the JSON schema enables Intellisense in Visual Studio for each property.

15.3.1.1.2.2 name

Type: String
Required: Yes

```
"name": "framework-control-js1"
```

This property is the name of the framework control, which is automatically generated based on the project name. The name is added to the control as a CSS class and can be used within the CSS files [▶ 2141] in the framework control project to describe the styles.

15.3.1.1.2.3 classNamespace

Type: String
Required: Yes

```
"classNamespace": "TcHmi.Controls.Custom"
```

This property is the namespace in which the control is added to the controls. Framework controls are added to the "Custom" namespace by default. The namespace is also used in Source.js.

15.3.1.1.2.4 className

Type: String
Required: Yes

```
"className": "FrameworkControlJs1"
```

This property is the internal name of the JavaScript object that defines the control. This name is used in Source.js.

15.3.1.1.2.5 displayName

Type: String
Required: Yes

```
"displayName": "FrameworkControlJs1"
```

This property is the name of the control that is displayed as the control name in the Engineering.

15.3.1.1.2.6 version

Type: JSON object
Required: Yes

```
"version": {
   "full": "1.0.0.0",
   "major": 1,
   "minor": 0,
   "revision": 0,
   "build": 0
}
```

This property is the version number of the framework control that is used for versioning a framework control. The version number is subdivided into the major release number, minor release number, revision number and build number.

For creating a package [▶ 119], the version number must match the version number in Manifest.json [▶ 2163].

15.3.1.1.2.7 visible

Type: Boolean
Required: Yes

```
"visible": true
```

This property makes the control visible or hides it in the toolbox in the Engineering. By default, a control is always displayed in the toolbox.

15.3.1.1.2.8 themeable

Type: Enum string
Required: No

```
"themeable": "Standard"
```

This property determines whether the control is displayed in the theme editor [▶ 561] within in the TwinCAT HMI Engineering.

Permissible values:

- `Standard`: The control is displayed in the theme editor.
- `None`: The control is not displayed in the theme editor.
- `Advanced`: The control is only displayed in the theme editor if the "Advanced" property is enabled in the theme editor.

15.3.1.1.2.9 base

Type: String
Required: Yes

```
"base": "tchmi-control"
```

This property determines whether the control inherits the properties of another control. By default, a framework control inherits all properties of Description.json of the basic control `tchmi-control`, which defines a basic framework for all TwinCAT HMI controls. The control can be made to inherit from other controls, such as a button, if a particular control is to be extended. The control name to be specified here corresponds to the control type or control name defined in Description.json [▶ 2124].

Validity

This property must refer to a valid control and must not have the value NULL, otherwise the control can no longer be used in the Engineering.

15.3.1.1.2.1 uses
0

Type: Array of strings
Required: No

```
"uses": []
```

This property defines a list of controls that are used in this control (static or dynamic at runtime). This property references the JavaScript and stylesheet files of the controls used in this control. By default, no other controls are used. The basic control [▶ 2125] does not have to be referenced again.

If other controls are used in this control, the control type or name of the control [▶ 2124] defined in Description.json must be specified.
The following listing shows an example of a framework control that uses the TcHmiButton [▶ 1309] and a user-specific control:

```
"uses": [
  "tchmi-button",
  "tchmi-my-framework-control"
]
```

For more information about using controls in a framework control, see Examples.

15.3.1.1.2.1 description 1

Type: String
Required: Yes

```
"description": "This description text will be used as a Tooltip in Visual Studio."
```

This property is the description of the control and is displayed as a tooltip when the mouse is moved over the control in the toolbox.

15.3.1.1.2.1 defaultDesignerEvent 2

Type: String
Required: No

```
"defaultDesignerEvent": ""
```

This property determines which event is to be configured in the Actions and Conditions Editor when the control is double-clicked. By default (no entry or empty entry), no event is executed on double-clicking.

All standard and custom events defined in Description.json can be used here. The following listing shows an example of a custom event:

```
"defaultDesignerEvent": ".onMyCustomEvent"
```

Custom events are defined under events [▶ 2137].

15.3.1.1.2.1 properties 3

Type: JSON object
Required: Yes

```
"properties": {
   "containerControl": false,
   "geometry": {
      "width": 150,
      "height": 150
   }
}
```

This property defines the engineering properties required for each framework control and describes general control properties.

- `containerControl` {Boolean}, *not required*: Specifies whether the control is a container control and can include additional controls using drag and drop. In this case, the control must also be derived [▶ 2125] (directly or indirectly) from TcHmiContainerControl [▶ 1174].
- `geometry` {JSON object}: Defines the default geometry properties (width and height) of the control, which are set after dropping the control from the toolbox.

15.3.1.1.2.1 icons 4

Type: Array of JSON objects
Required: Yes

```
"icons": [
   {
      "name": "Icons/16x16.png",
      "width": 16,
      "height": 16
   },
   {
      "name": "Icons/24x24.png",
      "width": 24,
      "height": 24
```

```
   },
   {
      "name": "Icons/32x32.png",
      "width": 32,
      "height": 32
   },
   {
      "name": "Icons/64x64.png",
      "width": 64,
      "height": 64
   }
]
```

This property defines the icon of the control in different dimensions. The array contains JSON objects, where each object corresponds to a control icon in a different resolution. The control icon is used to display the control in the toolbox of the Engineering and the Actions and Conditions Editor.

The name is specified as a relative project path. The dimensions (width/height) are specified in pixels.

15.3.1.1.2.1 dependencyFiles
5

Type: Array of JSON objects
Required: Yes

```
"dependencyFiles": [
   {
      "name": "Source.js",
      "type": "JavaScript",
      "description": ""
   },
   {
      "name": "Style.css",
      "type": "Stylesheet",
      "description": "Theme independent style"
   }
]
```

This property contains all files of the framework control that must be included in the HTML file of the HMI. By default, the Source.js and Style.css files are included. If additional JavaScript or stylesheet files are included in the control, they must be referenced under "dependencyFiles".

Caution: Theme-dependent stylesheet files are not included under "dependencyFiles", but under "themes [▶ 2127]".

The array contains JSON objects that contain information about the file:

- `name` {string}: file name and file extension with the relative project path to the file.
- `type` {string}: File type (JavaScript or stylesheet).
- `description` {string}: Optional file description.

15.3.1.1.2.1 themes
6

Type: JSON object of JSON objects
Required: No

```
"themes": {
   "Base": {
      "resources": [
         {
            "name": "Themes/Base/Style.css",
            "type": "Stylesheet",
            "description": ""
         }
      ]
   }
}
```

This property contains all themes that are included in the control. By default, the base theme is included with a stylesheet file. A theme is an object containing the array "`resources`". The array "`resources`" contains all the files that are allocated to a theme.

A "`resource`" object contains the same properties as a "dependencyFile" object [▶ 2127], with other allowed values for "`type`":

- `name` {string}: file name and file extension with the relative project path to the file.
- `type` {string}: file type (`Stylesheet` or `ThemedValues`, see Theming [▶ 559]).
- `description` {string}: Optional file description.

Note: Images and fonts etc. are not included under the theme resources, but referenced via the stylesheet file (see Theming [▶ 559]).

The following listing illustrates the inclusion of another theme with associated files:

```
"themes": {
   "Base": {
      "resources": [
         {
            "name": "Themes/Base/Style.css",
            "type": "Stylesheet",
            "description": ""
         }
      ]
   },
   "Custom": {
      "resources": [
         {
            "name": "Themes/Custom/Style.css",
            "type": "Stylesheet",
            "description": ""
         },
         {
            "name": "Themes/Custom/MyTheme.theme",
            "type": "ThemedValues"
            "description": "This is theme file created by the theme editor."
         }
      ]
   }
}
```

15.3.1.1.2.1 attributes 7

Type: Array of JSON objects (attributes)
Required: Yes

```
"attributes": [
   {
      "name": "data-tchmi-value",
      "propertyName": "Value",
      "propertySetterName": "setValue",
      "propertyGetterName": "getValue",
      "displayName": "Value",
      "visible": true,
      "themeable": "Standard",
      "displayPriority": 10,
      "type": "tchmi:general#/definitions/Boolean",
      "category": "Value",
      "description": "",
      "readOnly": false,
      "bindable": true,
      "heritable": true,
      "defaultValue": false,
      "defaultValueInternal": false
   }
]
```

The attributes define the data interfaces to the TwinCAT HMI Engineering. The attributes of the control are displayed in the Properties window of the Engineering under "ShowProperties". Any number of attributes can be defined in a control.

By default, the attribute array contains the type attribute that is required for each control and must not be removed. This attribute specifies that the control is of type `tchmi-framework-control-js1`:

```
{
   "name": "data-tchmi-type",
   "displayName": "",
   "propertyName": "type",
   "propertySetterName": "",
   "propertyGetterName": "getType",
   "visible": false,
   "type": "tchmi:framework#/definitions/ControlType",
   "category": "None",
   "description": "Type of the control.",
   "requiredOnCompile": true,
   "readOnly": true,
   "bindable": false,
   "heritable": false,
   "defaultValue": "framework-control-js1",
   "defaultValueInternal": null
}
```

The individual properties of an attribute are described on the following pages.

Type: Specifies the data type of the property.
Required: Specifies whether the property is required for the attribute (yes) or is optional (no).

15.3.1.1.2.1 name
7.1

Type: String
Required: Yes

```
"name": "data-tchmi-value"
```

Name of the attribute. The name is written into the HMTL code of the control as a "data-*" attribute. Therefore, the "data-" prefix is mandatory. The subsequent prefix "tchmi-" before the attribute is recommended for standardizing attributes, but it is not obligatory. An attribute name may only be used once for each control, otherwise unambiguous assignment cannot be guaranteed.

15.3.1.1.2.1 propertyName
7.2

Type: String
Required: Yes

```
"propertyName": "Value"
```

Internal attribute name of the control. The internal attribute name is formed from the attribute name without prefix in Pascal case.

Example:

```
"name": "data-tchmi-my-custom-value"
```

becomes: „`MyCustomValue`“.

For better mapping between JavaScript variables and attributes, it is recommended to name the attribute variable in the JavaScript code of Source.js according to the "`propertyName`".

15.3.1.1.2.1 propertySetterName
7.3

Type: String
Required: Yes

```
"propertySetterName": "setValue"
```

This property creates the reference to the setter function in Source.js. The setter function allows a new value to be set. For the reference between Description.json and Source.js, the setter function must have the same name as the"`propertySetterName`". If the names in Description.json and Source.js are different, an error message appears in the browser console and the setter function cannot be used.

The recommended notation for the setter name is the "`propertyName`" plus the prefix "`set`" in Camel case.

If an attribute is read only [▶ 2133], the empty string should be entered as `propertySetterName`.

15.3.1.1.2.1 propertyGetterName
7.4

Type: String
Required: Yes

```
"propertyGetterName": "getValue"
```

This property creates the reference to the getter function in Source.js. The getter function returns the attribute value when called. For the reference between Description.json and Source.js, the getter function must have the same name as the"`propertyGetterName`". If the names in Description.json and Source.js are different, an error message appears in the browser console and the getter function cannot be used.

The recommended notation for the getter name is the "`propertyName`" plus the prefix "`get`" in Camel case.

15.3.1.1.2.1 displayName
7.5

Type: String
Required: Yes

```
"displayName": "Value"
```

The property is the name of the attribute displayed in the Engineering Properties window for the attribute. The "displayName" can be selected as desired, but it is recommended that it should be based on the "propertyName".

15.3.1.1.2.1 refTo
7.6

Type: String
Required: No

```
"refTo": "basePropertyName"
```

Name of a PropertyName of the control. This attribute references another attribute.

This is used to write the physical unit (pixel or percent) in the development environment next to the numeric value.

Before the introduction of refTo the "Unit" suffix was checked.

Note: Available since version 1.10.1337.10

15.3.1.1.2.1 searchTerms
7.7

Type: Array of strings
Required: No

```
"searchTerms": [
   "Shortcut1",
   "ShortCut2",
   "Shortcut3"
]
```

This property defines a list of search terms for Engineering to optimize the search function within the Actions and Conditions Editor and the Properties window. This makes it possible to find the attribute using terms other than the attribute name.

15.3.1.1.2.1 visible
7.8

Type: Boolean
Required: No

```
"visible": true
```

This property determines whether the attribute is to be displayed in the Engineering. If the property is set to "`false`", the attribute is not available in the Engineering. By default, all attributes are displayed in the Engineering.

15.3.1.1.2.1 themeable
7.9

Type: Enum string
Required: No

```
"themeable": "Standard"
```

This property determines whether the attribute is displayed in the theme editor in the Engineering.

Permissible values:

- `Standard`: The attribute is displayed in the theme editor.
- `None`: The attribute is not displayed in the theme editor.
- `Advanced`: The attribute is only displayed in the theme editor if the "`Advanced`" property is activated in the theme editor.

15.3.1.1.2.1 type
7.10

Type: String (TcHmi data type)
Required: Yes

```
"type": "tchmi:general#/definitions/Boolean"
```

This property indicates the data types of the attribute. All data types of the TwinCAT HMI [▶ 2175] are available. Further user-specific data types can be defined (see dataTypes [▶ 2138]). The data type determines the display of the attribute in the Engineering Properties window (e.g. checkbox for a Boolean value).

15.3.1.1.2.1 displayPriority
7.11

Type: Number
Required: No

```
"displayPriority": 10
```

This property determines the priority with which the attribute is displayed within the category in the Engineering Properties window. User-specific sorting of attributes is possible based on the priority. Attributes with the same priority are displayed according to the order in which they are defined. The prioritization of the attribute categories is set via the property "attributCategories [▶ 2134]".

- Values less than 10 have high priority and are displayed higher up in the category, according to the value.
- Values that equal 10 have main attributes like top, left etc.

- Values greater than 10 have low priority and are displayed lower down in the category, according to the value.
- Values greater than 50 are collapsed by default and have to be expanded using the arrow.

15.3.1.1.2.1 allowedFunctions
7.12

Type: JSON object
Required: No

```
"allowedFunctions": {
   "returnType": "tchmi:general#/definitions/String",
   "requiredArguments": [
      {
         "type": "tchmi:general#/definitions/Number"
      }
   ],
   "requiredWaitMode": "Synchronous"
}
```

For attributes with the data type [▶ 2131]:

```
"type": "tchmi:framework#/definitions/Function"
```

These properties can only be used for "Function" type attributes. This property determines which functions are allowed for the attribute. This means that when configuring the attribute, only those functions are displayed that meet the defined criteria.

Further parameters are possible and can be specified per instance in the engineering.

Attributes:

- `returnType` {string}: Determines which TwinCAT HMI data type may be returned by the function.
- `requiredArguments` {array of JSON objects}: Determines which TwinCAT HMI data types the functions may have as transfer parameters. The parameters mentioned here are set by the control.
- `requiredWaitMode` {enum string}: Determines in which mode (synchronous, asynchronous) the function may be executed (`Synchronous`, `Asynchronous`).

For example, one the Beckhoff controls that uses this property is LinearGauge [▶ 306], namely for the ValueFormat [▶ 326] in order to restrict the functions that can format the values of the gauge.

15.3.1.1.2.1 category
7.13

Type: String
Required: No

```
"category": "Value"
```

This property indicates the category to which the attribute is assigned in the Engineering Properties window. New category names can be assigned or standard category names (e.g. Common) can be used. A new category name does not have to exist under the Attribute categories [▶ 2134], since the category is created independently of the attribute categories. The attribute categories are used to prioritize different attribute categories.

15.3.1.1.2.1 description
7.14

Type: String
Required: No

```
"description": "This description text will be used as a Tooltip in Visual Studio."
```

This property contains the description of the attribute, which is displayed as a tooltip when the mouse is moved over the attribute.

15.3.1.1.2.1 requiredOnCompile
7.15

Type: Boolean
Required: No

```
"requiredOnCompile": false
```

This property determines whether the attribute of the control is required when the HMI is compiled. If the attribute is required for the control and the property is not set, an error message is issued and the control is not valid. This property is normally not required for attributes of a framework control. An example for "`requiredOnCompile`" is `id` and `data-tchmi-type` of a control.

15.3.1.1.2.1 readOnly
7.16

Type: Boolean
Required: No

```
"readOnly": false
```

This property indicates whether the Engineering has write access to the attribute or only read access. By default, write access to an attribute is available. If `readOnly` is set to `true`, the attribute is no longer displayed in the Engineering Properties window, but it can still be used via the Actions and Conditions Editor.

> For read only attributes, the propertySetterName [▶ 2129] is not required and can be replaced by an empty string.

15.3.1.1.2.1 bindable
7.17

Type: Boolean
Required: No

```
"bindable": true
```

This property indicates whether a data Binding [▶ 91] to the attribute can be created in the Engineering. By default, the property is `true`.

If the property is set to `false`, it is not possible to link symbols to the attribute. Nevertheless, this attribute can be used and read at another location.

15.3.1.1.2.1 heritable
7.18

Type: Boolean
Required: No

```
"heritable": true
```

This property indicates whether the attribute can be passed on to other control. By default, the property is `true`. If the property is set to `false`, it is not available in the derived control classes.

15.3.1.1.2.1 defaultValue
7.19

Type: Data type of the attribute
Required: No

```
"defaultValue": false
```

This property indicates the default value for the attribute that is used in the Engineering Properties window when instantiating the control and written to the HTML code.

If `null` is entered here, no attribute is set.

15.3.1.1.2.1 defaultValueInternal
7.20

Type: Data type of the attribute
Required: No

```
"defaultValueInternal": false
```

This property indicates the internal default value for the attribute that can be used in Source.js. The internal default value may differ from the normal default value [▶ 2133].

15.3.1.1.2.1 attributeCategories
8

Type: Array of JSON objects
Required: No

```
"attributeCategories": [
   {
      "name": "Value",
      "displayPriority": 500,
      "defaultCollapsed": false,
      "description": ""
   }
]
```

The property is used to prioritize new attribute categories that are assigned under Attributes [▶ 2132]. Prioritization is used to determine the displayed order of the categories in the Engineering Properties window. The attribute categories are described within a JSON object.

The following properties can be defined:

- `name` {String}: The name of the attribute category. The attribute name must be assigned to at least one control attribute.
- `displayPriority` {Number}: The priority determines the order in which the various attribute categories are displayed within the Engineering Properties window.
  - Values less than 10 mean high priority and are displayed at the top of the Properties window (e.g. colors).
  - Values less than 500 have standard categories (e.g. Common =100, Layout = 200, Border = 300, Background Image = 400).
  - Values greater than 500 should be assigned user-specific categories so that they are listed below the standard categories.
  - Values greater than 800 have low priority and are rarely configured.
- `defaultCollapsed` {Boolean}, *not required*: This property determines whether the category is collapsed by default (`true`) or expanded (`false`).
- `description` {String}, *not required*: Optional description of the attribute category that is displayed as a tooltip for the mouseover event in the Engineering.

15.3.1.1.2.1 themedResources
9

Type: Array of JSON objects
Required: No

```
"themedResources": [
   {
      "name": "MyCustomThemedResource",
      "displayName": "MyCustomThemedResource",
```

```
      "description": "This is a custom themed resource. Look at theming of framework controls.",
      "type": "tchmi:framework#/definitions/MyThemedDefinition"
   }
]
```

This property contains all user-specific theme definitions that are not defined as attributes in the Engineering Properties window and can only be changed using the theme editor. An example of a "ThemedResource" are images within a control that cannot be exchanged via the Properties window (e.g. KnobConfigHorizontal and KnobConfigVertical for the LinearGauge [▶ 306]).

A "themedResources" object contains the following properties:

- `name` {String}: Name of the property to be changed.
- `displayName` {String}: Name displayed in the Engineering theme editor.
- `description` {String}: Optional description of the property that is displayed in the theme editor on mouseover.
- `type` {string of a framework data type}: Data type of the property. The data type can be either a TwinCAT HMI data type or a user-specific data type defined under "dataTypes [▶ 2138]". Usually, user-specific data types are used (see Theming).

15.3.1.1.2.2 functions
0

Type: Array of JSON objects
Required: No

```
"functions": [
   {
      "name": "customFunction",
      "displayName": "customFunction",
      "visible": true,
      "description": "This is a sample for a function that can be called.",
      "category": "Actions",
      "params": [],
      "type": null,
      "heritable": false,
      "searchTerms": []
   }
]
```

This property defines functions of the control that are independent of attributes. The functions can optionally have transfer parameters or a return value. Each function must have the same name in Source.js, otherwise an error message appears in the browser console and the function is not executed.

A JSON object for a function contains the following properties:

- `name` {String}: Internal name of the function.
- `displayName` {String}: Name of the function that is displayed in the Engineering.
- `description` {String}: Description of the function that is displayed as a tooltip during mouseover in the engineering.
- `category` {String}: Category of the function under which the function is displayed in the Actions and Conditions Editor in the Engineering.
- params [▶ 2136] {JSON object}, *not required*: Transfer parameters of the function.
- `type` {string, TwinCAT HMI data type}: Return value of the function. The return value must have a TwinCAT HMI data type.
- `heritable` {Boolean}: This property defines whether the function is to be passed on to derived classes.
- `searchTerms` {array of strings}, *not required*: This property defines search terms for the Engineering that are to optimize the search function within the Actions & Conditions Editor. This makes it possible to find the function using terms other than the function name. For example, if the strings "MyFunction" and "1234" are added to the array, the function can also be found under these terms.

The following figure shows an example of two framework control functions in the Engineering, both of which have no return value and contain an optional transfer parameter function.

15.3.1.1.2.2 params
0.1

Type: JSON object
Required: No

```
"params": [
   {
      "name": "functionParameter1",
      "displayName": "functionParameter1",
      "description": "This is a sample for a function parameter.",
      "type": "tchmi:general#/definitions/Number",
      "visible": true
   },
   {
      "name": "functionParameter2",
      "displayName": "functionParameter2",
      "description": "This is a sample for a function parameter.",
      "type": "tchmi:general#/definitions/Boolean",
      "bindable": false
      "visible": true
   }
]
```

This property defines the optional transfer parameters of a function. The transfer parameters can be transferred via Engineering.

A parameter object contains the following properties:

- `name` {String}: Internal name of the transfer parameter.
- `displayName` {String}: Name of the transfer parameter that is displayed in the Engineering.
- `description` {String}: Description of the transfer parameter that is displayed as a tooltip during mouseover in the Engineering.
- `bindable` {Boolean}: Determines whether a binding [▶ 91] can be set to this parameter. This is allowed by default.
- `visible` {Boolean}: Visibility of the transfer parameter. The visibility determines whether the parameter is to be displayed or hidden in the Engineering.

15.3.1.1.2.2 events
1

Type: Array of JSON objects
Required: No

```
"events": [
   {
      "name": ".onMyCustomEvent",
      "displayName": ".onMyCustomEvent",
      "visible": true,
      "displayPriority": 30,
      "category": "Control",
      "description": "This is a sample for a framework control event.",
      "heritable": true
      "allowsPreventDefault": false
   }
]
```

This property defines events that can be configured through the Engineering and are triggered in this framework control. An event is used to notify the Engineering that a specific event has occurred in the control. The HMI developer can respond to this event accordingly in the Engineering. Triggering of an event is implemented in Source.js.

An event object contains the following properties:

- `name` {String}: Internal name of the event.
- `displayName` {String}: Name of the function that is displayed in the Engineering.
- `visible` {Boolean}: Visibility of the event. If "`visible`" is set to "`false`", the event is not displayed in the Engineering.
- `displayPriority` {number}, *not required*: The priority determines the order in which the various events are displayed within the category.
  - Values = 10 correspond to frequently used events such as ".`onPressed`".
  - Values = 15 correspond to standard events such as ".`onClick`" or "`onRightClick`".
  - Values = 50 correspond to rarely used events such as ".`onMouseUp`".
  - No value assigned: The event is collapsed by default and has to be expanded using the arrow.
- `category` {String}: The category determines under which category the event is listed in the Engineering. Existing categories such as "`Operator`" can be used, or new category names can be assigned.
- `description` {String}: Optional description of the event, which is displayed as a tooltip on mouseover in the Engineering.
- `heritable` {Boolean}: This property defines whether the event is to be passed on to derived classes. By default, this value is "`true`", so that the event is inherited.
- `allowsPreventDefault` {Boolean}, *not required*: This property specifies that a user can explicitly select preventDefault. See also Events [▶ 2152]

The event configured above is displayed in the Engineering as follows:

15.3.1.1.2.2 dataTypes

Type: Array of JSON objects
Required: No

```
"dataTypes": [
   {
      "name": "tchmi:framework#/definitions/CustomDatatype",
      "schema": "Schema/TypeDefinitions/CustomDatatype.Schema.json"
   }
]
```

This property contains a list of all user-specific data types defined in the control. Developer can define their own data types in a JSON schema. The definition usually takes place under "Schema\TypeDefinitions [▶ 2167]". In order for the Engineering to know and be able to process the user-specific data types, they must be referenced in Description.json.

A dataType object contains the following properties:

- `name` {String}: Name of the new data type. Framework data types are always user-specific and have to start with "tchmi:framework#/definitions/{MyName}". For each data type an individual name must be assigned.
- `schema` {string, relative path}: Relative path to the JSON schema file describing the data type.

15.3.1.1.2.2 access 3

Type: Array of JSON objects
Required: No

```
"access": [
   {
      "name": "myCustomRight",
      "displayName": "myCustomRight",
      "description": "This is a sample for access rights in the framework control.",
      "visible": true,
      "defaultValueInternal": null
   }
]
```

This property contains a list of user-specific access rights that can be configured via the Engineering. Access rights restrict user groups in their activities (see authorization system [▶ 527]). A user-specific access right must be evaluated in Source.js with corresponding user-specific logic.

An access right object contains the following properties:

- `name` {String}: Internal name of the access right.
- `displayName` {String}: Name of the access right that is displayed in the Engineering.
- `description` {String}: Description of the access right, which is displayed as a tooltip in the Engineering on mouseover.
- `visible` {Boolean}: This property switches the visibility of the access right. If "`visible`" is set to "`false`", the access right is not displayed in the Engineering.
- `defaultValueInternal` {Boolean}: This property defines the internal default value that is to apply if the right is not configured via the Engineering. "`true`" means access is granted by default. "`false`" means access is not granted. "`null`" means access is not defined and the parent control is queried.

Various user groups are assigned to an access right via the Engineering, which are structured according to the following schema: tchmi:framework#/definitions/ControlAccessRight [▶ 2217]
This schema defines the type of access (`Allow`, `Deny`, `Inherit`) the respective user group has to the access right.

In the Engineering, the access right is displayed in the Properties window as follows (for two different user groups):

For more information, see section Access rights [▶ 2153].

15.3.1.2 Template.html

Like every control in the TwinCAT HMI, a framework control is based on HTML code. As soon as a framework control is instantiated in the Engineering of the TwinCAT HMI, an HTML div element is automatically created for the control. The developer of the framework control can define additional HTML code for the control within the control div element.

HTML code

The definition of the HTML code is implemented in the "Template.html" file. The use of this file is optional. The Template.html file is only used, if this is referenced in the Source.js of the Control [▶ 2145]. The Template.html file defines the basic structure of a framework control.

```
<div class="framework-control-js1-template tchmi-box">
   <!--Place your code here-->
</div>
```

The HTML code defined within this file is automatically inserted in within the control div element (if the file is used) in the Engineering. The code can be viewed in the browser during debugging, for example.

```
<div id="TcHmiFrameworkControlJs1" data-tchmi-type="framework-control-js1" data-tchmi-height="150"
data-tchmi-height-unit="px" data-tchmi-left="137" data-tchmi-left-unit="px" data-tchmi-top="140"
data-tchmi-top-unit="px" data-tchmi-value="True" data-tchmi-width="150" data-tchmi-width-unit="px"
class="tchmi-box tchmi-control framework-control-js1" title="" style="left: 137px; top: 140px;
width: 150px; height: 150px; z-index: 0; border-width: 0px;">
   <div class="framework-control-js1-template tchmi-box">
      <!--Place your code here-->
   </div>
</div>
```

Within the Template.html file any number of HTML elements can be defined. All standard HTML elements are available to the developer. In addition to the standard HTML elements, TwinCAT HMI controls can also be used in the Template.html file. The user-specific HTML code should be placed within the template div ("Place your code here").

Do not assign fixed IDs

No fixed IDs may be assigned within the Template.html file. The assignment of IDs within the Template.html file would contradict the concept of multiple instantiation of a framework control, since an ID may only occur once within the HTML document. The outer control div element is automatically assigned a unique ID through the Engineering.

CSS classes

Within the elements in the Template.html file, CSS classes [▶ 2141] can be assigned, which are described via cascading style sheets files, either detached from the theme or depending on the theme. By default, the template class in the Style.css file is used at project level.

Access via JavaScript

The HTML elements defined within the Template.html file can be further processed in the Source.js file [▶ 2144]. The elements can be changed during runtime or used as placeholders for dynamically created elements, for example. Processing of the HMTL elements in the Source.js file is optional. Static use within the Template.html file is also possible.

If the HMTL elements are to be processed further in the Source.js file, references to the HTML elements must be created there. References to the Template.html file are created within the PrevInit function in the Source.js file.

```
/** Handle template elements. Should be done before call to __previnit of super class. */
this.__elementTemplateRoot = this.getElement().find('.framework-control-js1-template');
```

The listing shows the access to the template div within Template.html. A search for the template div is initiated using the ".find()" JQuery function within the main JQuery object of the control. The reference to the JQuery object of the control can be reached via ".getElement [▶ 1170]()". If additional elements are placed within the template div, they can also be found using the ".find()" JQuery function. For performance reasons, we recommend to avoid using the outer JQuery object for the search in cases with many nested elements, but to use the inner elements that are already referenced instead.

Example:

```
<div class="framework-control-js1-template tchmi-box">
   <div class="framework-control-js1-inner-div">
      <div class="framework-control-js1-inner-div-my-element">

      </div>
   </div>
</div>
```

The following references result from this HMTL code:

```
/** Handle template elements. Should be done before call to __previnit of super class. */
this.__elementTemplateRoot = this.getElement().find('.framework-control-js1-template');
this.__innerDiv = this.__elementTemplateRoot.find('.framework-control-js1-inner-div');
this.__myElement = this.__innerDiv.find('.framework-control-js1-inner-div-my-element');
```

15.3.1.3 Style.css

Within a framework control, the appearance of individual elements is described using cascading style sheets. The default design of the control is defined in the Style.css file at the project level and in the Style.css file, the base theme of the control. In addition to the standard design, further themes can be defined for the framework control. The appearance of a framework control can be changed at four different levels [▶ 570] according to the theme system of the TwinCAT HMI.

15.3.1.3.1 Theme levels

The appearance of a framework control can be described at different levels of the theme system [▶ 570].

Level 4: Attribute definition per control type

Control themes [▶ 565] can be defined at level 4 of the theme system. Control themes allow individual attributes of the framework control to be overwritten. A control theme is defined within a theme file and can be created using the theme editor in Engineering or directly as JSON code. The theme file must be assigned to a specific theme within the project:

The theme file must then be referenced as "`ThemedValues`" under "themes [▶ 2127]" in the Description.json file.

Level 5: Attribute definition through `DefaultValueInternal`

At level 5 of the theme system it is possible to set the properties of the attribute via DefaultValueInternal [▶ 2134]. DefaultValueInternal can be set in the Description.json file of the framework control.

Example of a color attribute:

```
"defaultValueInternal": {
  "color": "rgba(255, 0, 0, 1)"
}
```

Level 8: Cascading style sheets per theme

Cascading style sheet files can be added to a theme at level 8 of the theme system. Within the CSS file any element properties can be defined.

The CSS file "Stylesheet" must then be referenced under "themes [▶ 2127]" in the Description.json file of the framework control.

Level 9: Cascading Style Sheets at project level

At level 9 of the theme system, cascading style sheet files can be added independently of a theme at project level. By default, each framework control project contains a Style.css file at project level. Within this file, element properties are defined that are to apply independently of a theme (e.g. sizes of a div).

All CSS files at the project level must be referenced under "dependencyFiles [▶ 2127]" in the Description.json file of the framework control.

15.3.1.3.2 Using CSS classes

The CSS class selectors are used within the cascading style sheet files of a framework control.

Permitted selectors

The following CSS selectors can be used conditionally within a CSS file of a framework control:

- Type selectors
- Attribute selectors
- Class selectors

With class selectors, new CSS classes are defined for a framework control, which means that CSS styles are not overwritten outside the framework control. If type or attribute selectors are to be used, you must ensure that they are only used for the class of the framework control and do not overwrite any styles outside the framework control (descendant selector):

```
.tchmi-framework-control-js2 a {
   color: red;
}
```

The following CSS selectors should not be used within a CSS file of a framework control:

- ID selectors: Due to multiple instantiation, you cannot assign fixed IDs within a framework control.
- Universal selector: A framework control should not overwrite any styles outside the control during instantiation.

Using the class selectors

CSS classes can be used in the Template.html file, or they can be added dynamically at runtime within the Source.js file.

```
.tchmi-framework-control-js1-template
{
   width: 100%;
   height: 100%;
}
```

Within the Template.html file, a CSS class is assigned via the HTML attribute "`class`":

```
<div class="tchmi-framework-control-js1-template tchmi-box">

</div>
```

> The class "`tchmi-box`" sets CSS properties to facilitate element positioning and can be used with all positioned HTML elements.

The dynamic use of a CSS class within Source.js can be realized via the jQuery function "`.addClass()`", for example:

```
this.__elementTemplateRoot.addClass('tchmi-framework-control-js1-my-custom-class');
```

15.3.1.3.3 Overwriting CSS of controls

Within the CSS files it is possible to overwrite the appearance of embedded TwinCAT HMI controls. For example, if you want to change the appearance of an embedded button within a control, you can do this by applying the CSS styles of the button.

Please ensure that all buttons are not inadvertently assigned this appearance and that the CSS file of the general button does not overwrite that of the embedded control. Both can be achieved by so-called descendant selectors (spaces in CSS):

```
.framework-control-js1 .tchmi-button {
    background-image: linear-gradient(135deg, #eff1f3, #aeb9c2);
    color: #4794da;
    box-shadow: 0px 0px 5px 0px rgba(0,0,0,0.6);
}

.framework-control-js1 .tchmi-button .down {
    background-image: linear-gradient(135deg, #aeb9c2, #eff1f3);
    color: #000000;
    box-shadow: inset 0px 0px 5px 0px rgba(0,0,0,0.6);
}
```

All main elements of a control are assigned the control name as CSS class name (in this case `framework-control-js1`). If this is attached before the style of the control to be overwritten, the new appearance in the example only applies to buttons below the framework control.

CSS selectors are always read from right to left:

- First example: Main element of the button ("`.tchmi-button`") if it is below (" “) FrameworkControlJs1 (“`framework-control-js1`").
- Second example: Element with class down ("`.down`") if it is the main element of the button ("`.tchmi-button`"), if it is below (" ") FrameworkControlJs1 (“`framework-control-js1`").

15.3.1.4 Source.js

A framework control requires JavaScript code. The JavaScript code is defined in the Source.js file at project level, which is included in each framework control project. The Source.js file defines the basic framework for the implementation of a framework control and can be flexibly extended.

Programming conventions:

The programming conventions provide a basis for standardizing framework controls. They are used to distinguish between internal and external access.

- Member variables are marked with the prefix"__" and in Camel case:

```
this.__myVar = …;
```

- Functions with external access (Engineering: setter/getter functions) are identified without prefix and in Camel case:

```
MyConrol.prototype.setMyVar = …
```

- Internal functions are marked with the prefix "__" and in Camel case:

```
MyControl.prototpye.__processMyWar = …
```

> If you make changes in the Source.js file and the framework control is already associated with an HMI project, you must then carry out a project build [▶ 2123] of the framework control to apply the changes.

15.3.1.4.1 Control life cycle

The control life cycle describes which system function of the control is called by the system at which time. In the control life cycle, a distinction is made as to whether the control is on an HMI page for which preloading is activated.

The basic control life cycle is described in the following diagram:

Explanations:

1. The constructor is called during control instantiation.
2. The PrevInit function is called after the constructor and before the attribute setter functions are called.
3. The attribute setter functions are called after the PrevInit function.
4. The init function is called after the attribute setter functions. At this point, the control attributes are initialized by the attribute setter functions and have valid values.
5. The Attach function is called after the Init function, once the control has been added to the DOM on the HMI page.
6. The Detach function is called after a control has been removed from the DOM of the HMI page (e.g. during page switching).
7. The Destroy function is called after the control has been removed from the DOM of the HMI page, so that the system can destroy the controls. In addition, the Destroy function can be called explicitly at another point (for example, when subordinate controls elements are destroyed).

HMI page: Preloading disabled

Normally, preloading of an HMI page is disabled (Preload this partial: false), which means that the controls are not stored in the browser cache. This increases the speed of the initial HMI loading process, but it can lead to a reduction in the loading speed of individual HMI pages on which many controls are instantiated.

In this case, the constructor of the control is called when the page is loaded (for example, when switching to the page or when loading the start page). This is followed by the PrevInit function, the attribute setter functions, the Init function and the Attach function. As soon as the page is unloaded (e.g. when switching to another page), the Detach function is called, followed by the Destroy function. The life cycle of the controls starts again when the HMI page is loaded.

HMI page: Preloading enabled

If preloading is enabled for an HMI page (preload this partial: true), the controls are stored in the browser cache. This increases the loading speed of a single HMI page, but it can lead to a reduction in the initial loading process of the HMI.

In this case, the constructor of the control is called as soon as the HMI is initially loaded. This is followed by the PrevInit function, the attribute setter functions and the init function. The Attach function of the control is not called until the HMI page on which the control is instantiated is loaded. The Detach function is called as soon as the HMI page is unloaded. The Destroy function is not called automatically subsequently. The control remains in the browser cache. When the HMI page is reloaded, the Attach function is called directly.

15.3.1.4.2 Basic framework

The Source.js file already contains preconfigured source code that is required for various functions of the framework control and may not be modified in some cases. In JavaScript, a framework control is a JavaScript object that is described by various properties and methods. In other programming languages, the framework control would correspond to a class from which concrete instances (objects) are created at a later point in time.

Project information

Better support for IntelliSense in JavaScript:

```
// Provider for a best effort Intellisense of Visual Studio 2017.
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Lib\jquery.d.ts" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\TcHmi.d.ts" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Controls\System\TcHmiControl\Source.d.ts" />

// Provider for a best effort Intellisense of Visual Studio 2013/2015.
```

```
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\Lib\jquery\jquery.js" />
/// <reference path="C:\TwinCAT\Functions\TE2000-HMI-Engineering\Infrastructure\TcHmiFramework
\Latest\TcHmi.js" />
```

These comment lines include different files, allowing better IntelliSense support within the Source.js file. These lines are not required for using the control.

Information about the framework control:

```
/*
* Generated 2/15/2018 10:57:13 AM
* Copyright UserXY 2018
* .......................
*/
```

These comment lines contain information about the creation of the framework control, which is generated automatically when the framework control project is created. These lines are not required for using the control.

Basic structure of a framework control

```
/**
* Namespace: TcHmi
* [REQUIRED]
*/
(function (TcHmi) {
   /**
   * Namespace: TcHmi.Controls
   * [REQUIRED]
   */
   (function (Controls) {
      /**
      * Namespace: TcHmi.Controls.Custom
      * [CUSTOM]
      */
      var Custom;
         (function (Custom) {
         /** FrameworkControlJs1 object */
         var FrameworkControlJs1 = (function (_super) {
            __extends(FrameworkControlJs1, _super);

            /** Control specific logic */

            return FrameworkControlJs1;
         })(TcHmi.Controls.System.TcHmiControl);
      Custom.FrameworkControlJs1 = FrameworkControlJs1;
      })(Custom = Controls.Custom || (Controls.Custom = {}));
      /**
      * Register control...
      */
      Controls.register('framework-control-js1', Custom.FrameworkControlJs1, 'Custom/
FrameworkControlJs1/', 'Custom/FrameworkControlJs1/Template.html');
   })(Controls = TcHmi.Controls || (TcHmi.Controls = {}));
})(TcHmi);
```

This code defines the basic structure of a framework control and is required for using the control at runtime. In the event of invalid changes, the control can no longer be used properly.

> Do not change the basic structure of the framework control within Source.js. There is a risk that the control can no longer be used.

Definition of the namespaces

Within Source.js, various namespaces are referenced and defined, which are required for the framework.

TcHmi

```
(function (TcHmi) { })(TcHmi);
```

This function makes the control known within the "TcHmi" namespace. The framework cannot be accessed outside this namespace.

Controls

```
(function (Controls) { })(Controls = TcHmi.Controls || (TcHmi.Controls = {}));
```

This function makes the control known within the "Controls" namespace. This namespace contains all TwinCAT HMI controls (System [▶ 433], Beckhoff [▶ 141], Custom).

The control is registered in the control namespace which is required for the Engineering of the TwinCAT HMI. The transfer of the Template [▶ 2140] parameter is optional.

```
Controls.register('framework-control-js1', Custom.FrameworkControlJs1, 'Custom/
FrameworkControlJs1/', 'Custom/FrameworkControlJs1/Template.html');
```

Custom

```
var Custom;
(function (Custom) { })(Custom = Controls.Custom || (Controls.Custom = {}));
```

This function defines the namespace for user-specific controls and makes the control known within the namespace. The concrete framework control object is defined in the custom namespace (prototypical control object) and then assigned to the custom namespace:

```
Custom.FrameworkControlJs1 = FrameworkControlJs1;
```

Prototypical control object

A framework control is defined by a JavaScript object within the custom namespace. The logic is implemented in this section (Constructor [▶ 2147], System functions [▶ 2148], Attributes [▶ 2149], Functions [▶ 2151], Events [▶ 2152], Access rights [▶ 2153]).

```
var FrameworkControlJs1 = (function (_super) {
   __extends(FrameworkControlJs1, _super);

   /** Control specific logic */

   return FrameworkControlJs1;
})(TcHmi.Controls.System.TcHmiControl);
```

The object (_super) of the basic control [▶ 2125] defined in Description.json [▶ 2123] (usually TcHmiControl [▶ 433]) is passed to the framework control. The framework control inherits all properties of the defined basic control.

The framework control is defined by a function, in which the user-specific code of the framework control is implemented. A function in JavaScript always links a prototypical object. Access to the prototypical object of this function is indicated by the property "`protoype`". All properties (other functions) of the framework control are stored in the function object via the prototype.

```
MyControlName.prototype.myProperty = function () {...};
```

When a specific instance (JavaScript object) is created by the function, the instance has all the properties of the prototypical object of the function.

15.3.1.4.3 Constructor

Each framework control contains a constructor that is structured as follows:

```
function FrameworkControlJs1(element, pcElement, attrs) {
   /** Call base constructor */
   _super.call(this, element, pcElement, attrs);

   /** Attribute variables are initialized with undefined. The System will call the setter later.
See "Attribute philosophy" above. */
   this.__value = undefined;
}
```

Within the constructor, the internal variables of the control (member variables) are initialized with "undefined". Initialization at this point is not necessary, but is recommended for better clarity.

The transfer parameters of the constructor are only used internally and forwarded to the base constructor call.

15.3.1.4.4 System functions

Framework controls have functions that are called by the system at different times. Each system function calls the respective system function of the basic class (TcHmiControl [▶ 433]) via the "_super" object and passes a reference to the current control object.

PrevInit function

```
FrameworkControlJs1.prototype.__previnit = function () {
   /** Handle template elements. Should be done before call to __previnit of super class. */
   this.__elementTemplateRoot = this.getElement().find('.framework-control-js1-template');

   /** Call __previnit of super class. */
   _super.prototype.__previnit.call(this);
};
```

The PrevInit function is automatically called after the system calls the constructor [▶ 2147] and before the attribute setters of each attribute are called. Accordingly, the member variables of the attributes have the value "undefined [▶ 2108]" when the PrevInit function is called. Within the PrevInit function, the references [▶ 2140] to the HMTL elements are created from the Template.html file.

This function may only be called by the system and not explicitly by the developer.

Init function

```
FrameworkControlJs1.prototype.__init = function () {
   _super.prototype.__init.call(this);
};
```

The Init function is called automatically by the system after the attribute setters of the individual attributes have been called. Accordingly, when the Init function is called, the member variables of the attributes already have the values that were passed to the control or defined internally as default values.

This function may only be called by the system and not explicitly by the developer.

Attach function

```
FrameworkControlJs1.prototype.__attach = function () {
   /**
   * Enable everything which was disabled while __detach function was called!
   */
   /**
   * Initialize everything which is only available while the control is part of the active dom.
   */
   /** Call base __attach function */
   _super.prototype.__attach.call(this);
};
```

The Attach function is automatically called by the system after the current control instance has been added to the DOM of the HTML document. Within the Attach function, all calculations should be performed that are only available while the control instance is in the DOM (e.g. .getBoundingClientRect() of DOM elements or .width() of jQuery elements). The opposite of the Attach function is the Detach function.

This function may only be called by the system and not explicitly by the developer.

Detach function

```
FrameworkControlJs1.prototype.__detach = function () {
   /**
   * Disable everything which is not needed while the control is not part of the active dom.
   * No need to listen to events for example!
   */
```

```
   /** Call base __detach function */
   _super.prototype.__detach.call(this);
};
```

The system automatically calls the Detach function after the current control instance has been removed from the DOM of the HTML document. In the Detach function, all functions that are no longer required after the control instance has been removed from the DOM are to be disabled. The opposite of the Detach function is the Attach function.

This function may only be called by the system and not explicitly by the developer.

Destroy function

```
FrameworkControlJs1.prototype.destroy = function () {
   /**
   * Free resources like child controls, events etc.
   */
   /** Call base __destroy function */
   _super.prototype.destroy.call(this);
};
```

The system automatically calls the Destroy function when the control instance is no longer used (after the Detach function). If the control is on an HTML page where "Preload Partial" is enabled, the Destroy function is not called because the page is still saved in the browser cache. Within the Destroy function, the elements that were created during runtime (e.g. dynamically generated TwinCAT HMI controls) are to be destroyed.

This function can also be called explicitly (for example when child controls are destroyed).

15.3.1.4.5 Attributes

Framework control attributes [▶ 2128] defined in Description.json must be initialized within Source.js, and the corresponding setter, getter and process methods must be implemented. The setter and getter methods provide central access to the attribute.
The following examples refer to the following attribute, which is defined in Description.json under "attributes [▶ 2128]":

```
"attributes": [
   {
      "name": "data-tchmi-value",
      "propertyName": "Value",
      "propertySetterName": "setValue",
      "propertyGetterName": "getValue",
      "displayName": "Value",
      "visible": true,
      "themeable": "Standard",
      "displayPriority": 10,
      "type": "tchmi:general#/definitions/Boolean",
      "category": "Value",
      "description": "",
      "readOnly": false,
      "bindable": true,
      "heritable": true,
      "defaultValue": false,
      "defaultValueInternal": false
   }
]
```

Initialization

The attributes are initialized in the constructor of the framework control. Initialization is not required here (variables without keywords are automatically initialized in JavaScript when used), but is recommended for better clarity.

```
this.__value = undefined;
```

Member variables and functions that cannot be accessed from outside are prefixed with "__".

Setter method

The setter method of an attribute is only required for attributes that are not set to "readOnly [▶ 2133]" (all attributes that can be set via Engineering). Like the definition, the setter method must be named in the "propertySetterName [▶ 2129]".

```
/**
* @description Setter function for 'data-tchmi-value' attribute.
* @param {Boolean} valueNew the new value or null
* @returns {void}
*/
FrameworkControlJs1.prototype.setValue = function (valueNew) {
   // convert the value with the value converter
   var convertedValue = TcHmi.ValueConverter.toBoolean(valueNew);

   // check if the converted value is valid
   if (convertedValue === null) {
      // if we have no value to set we have to fall back to the defaultValueInternal from
description.json
      convertedValue = this.getAttributeDefaultValueInternal('Value');
   }

   if (tchmi_equal(convertedValue, this.__value)) {
      // skip processing when the value has not changed
      return;
   }

   // remember the new value
   this.__value = convertedValue;

   // inform the system that the function has a changed result.
   TcHmi.EventProvider.raise(this.__id + ".onFunctionResultChanged", ["getValue"]);

   // call process function to process the new value
   this.__processValue();
};
```

Within the setter method, the parameter that was passed is first converted into the expected data type by Engineering using the framework API function "ValueConverter [▶ 2068]". If an invalid value is passed to the ValueConverter or if the conversion of the value fails, the function returns "null". In this case the DefaultValueInternal [▶ 2134] from Description.json is used. The system then checks whether a value change to the last stored value of the attribute on the member variable has taken place. If no value change has taken place, the setter is exited. If a value change has taken place, the value of the member variable for the attribute is overwritten and the system is notified that a value change has taken place. Finally, the Process method is called for the value in which the response to the value change takes place.

The function "tchmi_equal [▶ 2106]" should be used to check for a change in value. The JS comparison operator "===" can only check the basic JavaScript data types but not complex JSON objects.

Getter method

The getter method of an attribute returns the current value of the attribute and is required for each attribute. Like the definition, the getter method must be named in the "propertyGetterName [▶ 2130]".

```
/**
* @description Getter function for 'data-tchmi-value' attribute.
* @returns {Boolean}
*/
FrameworkControlJs1.prototype.getValue = function () {
   return this.__value;
};
```

Process-Methode

The Process method is always called when the attribute value has been changed. The response to the value changes is to be implemented in the process method. For example, depending on a Boolean value elements can be switched visible or invisible, or a list can be created dynamically based on a transferred JSON object.

```
/**
* @description Processor function for 'data-tchmi-value' attribute.
* @returns {void}
*/
FrameworkControlJs1.prototype.__processValue = function () {
   // process actions with Value
   // ...
};
```

15.3.1.4.6 Functions

Framework control functions [▶ 2135] defined in Description.json must be defined and implemented in Source.js. The function must be named like the "name [▶ 2135]" of the function in Description.json.

Function without transfer parameters

A function without transfer parameters is defined in Description.json as follows:

```
"functions": [
   {
      "name": "customFunction",
      "displayName": "customFunction",
      "visible": true,
      "description": "This is a sample for a function that can be called.",
      "category": "Actions",
      "params": [],
      "type": null,
      "heritable": false,
      "searchTerms": []
   }
]
```

This results in the following code for Source.js:

```
/**
* @description This is the call of the api function 'customFunction'.
* @returns {void}
*/
FrameworkControlJs1.prototype.customFunction = function () {
   /**
   * customFunction is defined in the description.json and called from outside.
   * The api functions are shown in the graphical user interface of the enginnering in the twincat
   */

   // implement function logic here
};
```

Within the function, any execution logic of the function can be implemented.

Function with return value

The functions enables a return value to be returned the calling point in Engineering. The return value is defined in Description.json under "type".

```
"type": "tchmi:general#/definitions/Boolean"
```

Within the function in Source.js it is then possible to return a value of the corresponding type.

```
/**
* @description This is the call of the api function 'customFunction'.
* @returns {Boolean}
*/
FrameworkControlJs1.prototype.customFunction = function () {
   /**
   * customFunction is defined in the description.json and called from outside.
   * The api functions are shown in the graphical user interface of the enginnering in the twincat
   */
```

```
   // implement function logic here
   return true;
};
```

Function with transfer parameters

A function with transfer parameter [▶ 2136] is defined in Description.json as follows:

```
"functions": [
   {
      "name": "customFunction",
      "displayName": " customFunction ",
      "visible": true,
      "description": "This is a sample for a function that can be called by the engineering.",
      "category": "Actions",
      "params": [
         {
            "name": "functionParameter1",
            "displayName": "functionParameter1",
            "description": "This is a sample for a function parameter.",
            "type": "tchmi:general#/definitions/Number",
            "visible": true
         },
         {
            "name": "functionParameter2",
            "displayName": "functionParameter2",
            "description": "This is a sample for a function parameter.",
            "type": "tchmi:general#/definitions/Boolean",
            "visible": true
         }
      ],
      "type": null,
      "heritable": false,
      "searchterms": []
   }
]
```

The function is implemented in Source.js like a normal function. The transfer parameters are passed with the function call. The parameters must be checked for validity before they are used, similar to the attribute setters [▶ 2149].

```
/**
* @description This is the call of the api function 'customFunction'.
* @returns {void}
*/
FrameworkControlJs1.prototype.customFunction = function (functionParameter1, functionParameter2) {
   /**
   * customFunction is defined in the description.json and called from outside.
   * The api functions are shown in the graphical user interface of the enginnering in the twincat
   */
   var convertedParameter1 = TcHmi.ValueConverter.toNumber(functionParameter1);
   if (convertedParameter1 === null) {
      // parameter is null: define what to do in this case

      // exit function call if the execution without this parameter is not possible
      return;
   }

   var convertedParameter2 = TcHmi.ValueConverter.toBoolean(functionParameter2);
   if (convertedParameter2 === null) {
      // parameter is null: define what to do in this case

      // exit function call if the execution without this parameter is not possible
      return;
   }

   // implement function logic with the evaluation of the parameters here
};
```

15.3.1.4.7 Events

Framework control events defined in Description.json should be triggered in Source.js. The name of the event must be named like the "name [▶ 2137]" of the event in Description.json. An event can be triggered in different situations, e.g. when a certain record changes (see ".onDataChanged [▶ 229]" in the data grid [▶ 203]).

An event is triggered in Source.js via the EventProvider [▶ 1885] of the framework. The framework API function "EventProvider.raise() [▶ 1886]" is used for this purpose. When the event is triggered, the concrete control instance must be identified with "this.__id".

```
// raise .onMyCustomEvent event
TcHmi.EventProvider.raise(this.__id + '.onMyCustomEvent);
```

Optionally, parameters can be passed to the event callbacks when the event is triggered.

```
// declare parameter: type anything
var myParameterObject = 42;
// raise .onMyCustomEvent event with event parameters
TcHmi.EventProvider.raise(this.__id + '.onMyCustomEvent, myParameterObject);
```

The response to the triggered events can take place in Engineering via the Properties window [▶ 49] or via the registration of the event in a Code Behind file [▶ 2122].

> If `allowPreventDefault` is allowed in the event definition [▶ 2137], the myParameterObject should offer `preventDefault` as a function. All DOM and jQuery event objects meet this requirement, for example. This function is called if the user has set the option for this event in Engineering.

15.3.1.4.8 Access rights

User-specific access rights [▶ 2139] defined in Description.json and configured in Engineering [▶ 49] should be checked and evaluated in Source.js. Depending on the user-specific access rights [▶ 521], specific elements can be switched visible or invisible or enabled or disabled, for example.

The evaluation of the configured access rights is performed via the framework API function "TcHmi.Access.checkAccess()" [▶ 1836]. The current instance of the control is passed to the function (this) and the name of the right to be checked, which must correspond to the definition in Description.json.

```
// check user access of "myCustomRight" that is defined in the description.json
if (TcHmi.Access.checkAccess(this, 'myCustomRight') === true) {
   // current user has the rights to do this

   // process actions with the right
   // show elements, enable buttons, etc.
}
else {
   // Current user has no rights to do this

   // process actions without the rights
   // hide elements, disable buttons, etc.
}
```

> Since user management only works when authentication is active, the configured functions are not available in live view.

15.3.1.5 CustomDatatype.Schema.json

Within a framework control project, user-specific and complex data types can be implemented if the data type is not covered by a standard data type. In principle, any data structures can be transferred as objects without the need for a user-specific data type. A user-specific data type is required if an editor is to be provided for generating the data in Engineering. The data editor is generically based on a given JSON schema.

The following user-specific data types can be implemented:

- Objects [▶ 2157] with fixed properties
- Array [▶ 2159] of any data types
- Enumerations [▶ 2161]
- Limitation [▶ 2162] of several allowed data types

15.3.1.5.1 Creating a custom data type

Add a new item in the "TypeDefinitions" folder under "Schema".

Add a new JSON file or JSON schema file to the project.

Reference the schema file in Description.json under "dataTypes [▶ 2138]". Note that user-specific data types are assigned to the "Framework" category.

```
"dataTypes": [
   {
      "name": "tchmi:framework#/definitions/CustomDatatype",
      "schema": "Schema/TypeDefinitions/CustomDatatype.Schema.json"
   }
]
```

You can then use the user-specific data type within Description.json.

```
"type": "tchmi:framework#/definitions/CustomDatatype
```

Note: If the framework control is already referenced in an HMI project, a project build (see First Steps [▶ 2164]) must then be executed and the HMI project reloaded so that the new data type is included in the type descriptions of the HMI project.

To reload the project, right-click on the HMI project and select "Unload project".

Then select "Reload project" to reload the project. You can then use all the functions as usual.

15.3.1.5.2 Implementing a custom data type

The configuration of a JSON schema for the framework follows a certain structure.

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "{DatatypeName}": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "title": " {DatatypeName} ",
         "type": "array",
         //...
      }
   }
}
```

Each JSON schema contains a definition that describes the data type ({DatatypeName}). The definition of the data type contains the specification (type) and the description of the data type through attributes depending on the respective data type.

15.3.1.5.2.1 Objects

Objects can contain any properties. A property is described under Properties and can be refined through optional metadata (propertiesMeta).

The following JSON schema describes an object with two properties.

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "CustomDatatype": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "title": "CustomDatatype",
         "type": "object",
         "propertiesMeta": [
            {
               "name": "axisName",
               "category": "Name",
               "displayName": "Axis Name",
               "displayPriority": 10,
               "description": "",
               "defaultValue": null,
               "defaultValueInternal": null
            },
            {
               "name": "axisColor",
               "category": "Colors",
               "displayName": "Axis color",
               "displayPriority": 10,
               "description": "",
               "defaultValue": null,
               "defaultValueInternal": {
                  "color": "#4794da"
               }
            }
         ],
         "properties": {
            "axisName": {
               "type": "string"
            },
            "axisColor": {
               "$ref": "tchmi:framework#/definitions/SolidColor"
            }
         },
         "required": [ "axisName" ]
      }
   }
}
```

This results in the following generic data editor in Engineering.

Explanations

propertiesMeta:

The metadata of the properties are optional. They describe the display within the generic data editor. The metadata are described by the following properties:

- `name` {String}: Internal name of the property, see "name [▶ 2129]" in Description.json.
  Note: The name must match the name under the properties.
- `category` {String}: Name of the category under which the property is displayed in the generic data editor, see "category [▶ 2132]" in Description.json.
- `displayName` {String}: Name of the property displayed in the generic data editor, see "displayName [▶ 2130]" in Description.json.
- `displayPriority` {Number}: Priority of the display, see "displayPriority [▶ 2131]" in Description.json.
- `description` {String}: Description of the property, see "description [▶ 2132]" in Description.json.
- `defaultValue`: Default value used initially in engineering, see "defaultValue [▶ 2133]" in Description.json.
- `defaultValueInternal`: Internal default value, see "defaultValueInternal [▶ 2134]" in Description.json.

properties:

The name and data type of the object properties are described under Properties. The name is determined by the name of the property. The data type can be described as follows:

- `type` {string}: Default JSON data type of the property, or
- `$ref` {string, reference}: Reference to a framework data type. This can be implemented within the project or taken from the existing data type descriptions of the framework.

required:

The required properties are listed under "required". The list is based on the internal name. If a property of the object that is set to required is not set in Engineering, the generic data editor displays a warning.

15.3.1.5.2.2 Arrays

Arrays can be created from simple data types and complex objects.

Array of simple data types

Simple data types are specified within the "items" property under the "type" data type property. The following schema defines an array of strings.

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "CustomDatatype": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "title": "CustomDatatype",
         "type": "array",
         "items": {
            "type": "string"
         }
      }
   }
}
```

This schema generates the following generic data editor in Engineering.

The generic data editor displays the array elements on the rleft-hand side under the name of the respective data type (in the example: string). If a self-defined name is to be displayed instead, an object with a property of the type of the data type must be selected instead of the simple data type, and the "displayName" of the property must be set in the metadata.

Array of complex data types

Complex data types are specified in the property "items" under the reference property "$ref". The following schema defines an array of objects of data type "CustomDatatype". This data type was previously implemented as Object [▶ 2157].

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "MyArray": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "title": "MyArray",
         "type": "array",
         "items": {
            "$ref": "tchmi:framework#/definitions/CustomDatatype"
         }
      }
   }
}
```

Furthermore, it is possible to describe the object directly within the array definition. The following schema describes the same array with the definition of the object within the property "items".

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "MyArrayInlineObject": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "title": "MyArrayInlineObject",
         "type": "array",
         "items": {
            "type": "object",
            "title": "CustomObject",
            "description": "Defines a custom object in the array.",
            "propertiesMeta": [
                  {
                     "name": "axisName",
                     "category": "Name",
                     "displayName": "Axis Name",
                     "displayPriority": 10,
                     "description": "",
                     "defaultValue": null,
                     "defaultValueInternal": null
                  },
                  {
                     "name": "axisColor",
                     "category": "Colors",
                     "displayName": "Axis color",
                     "displayPriority": 10,
                     "description": "",
                     "defaultValue": null,
                     "defaultValueInternal": {
                     "color": "#4794da"
                  }
               }
            ],
            "properties": {
               "axisName": {
                  "type": "string"
               },
               "axisColor": {
                  "$ref": "tchmi:framework#/definitions/SolidColor"
               }
            },
            "required": [ "axisName" ]
         }
      }
   }
}
```

Both schema files generate the following generic data editor in Engineering.

15.3.1.5.2.3 Enumerations

Enumerations can contain any elements for the data type string. In the Engineering, enumerations are not displayed in the generic data editor, but in the combo box in the Properties window. The permissible values of the enumeration are defined in the property "enum" in an array.
The following schema defines an exemplary enumeration.

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "MyEnum": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "title": "MyEnum",
         "type": "string",
         "enum": [ "MyValue1", "MyValue2", "MyValue3", "MyValue4" ]
      }
   }
}
```

This results in the following display in Engineering.

15.3.1.5.2.4 Restrict permitted data types

User-specific data types can be restricted so that only certain data types can be transferred. As a result, when creating a binding [▶ 91] only the symbols corresponding to the permitted data types are displayed.

The following schema defines an exemplary restriction of permissible values (`oneOf`). Only values that have the data type "Time" or "TimeSpan" are permitted.

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "definitions": {
      "MyConstrainedDatatype": {
         "$schema": "http://json-schema.org/draft-04/schema",
         "oneOf": [
            {
               "title": "Time",
               "$ref": "tchmi:general#/definitions/DateTime"
            },
            {
               "title": "Timespan",
               "$ref": "tchmi:general#/definitions/TimeSpan"
            }
         ]
      }
   }
}
```

The display in Engineering is normally an attribute. However, only bindings to symbols that correspond to the permitted data types can be created.

15.3.1.6 Manifest.json

The Manifest.json file contains optional information that can be set to create a TwinCAT HMI package [▶ 118]. The information is placed under the metadata and generated automatically when the project is created.

```
"metadata": {
   "industrialField": "",
   "packageType": "FrameworkExtPrj",
   "id": "FrameworkControlJs1",
   "version": "1.0.0.0",
   "guid": "7001c40b-7081-4c53-8496-f416efe212b9",
   "configurationName": "",
   "platformName": "",
   "creationTime": 1520610054,
   "authors": "",
   "owners": "",
   "summary": "",
   "licenseUrl": "",
   "projectUrl": "",
   "iconUrl": "",
   "requireLicenseAcceptance": false,
   "description": "",
   "releaseNotes": "",
   "copyright": "",
   "tags": "",
   "dependencies": [],
   "references": []
}
```

Note: For creating a package, the version number must match the version number [▶ 2124] in Description.json [▶ 2123].

15.3.2 Introduction

Framework controls are user-specific controls that are not created with the graphical WYSIWYG editor [▶ 27]. Framework controls are used if the desired behavior cannot be mapped with the standard controls or a user control. Unlike a Code Behind file [▶ 2120], a framework control can be used independently of the HMI project, so that the source code can be reused.

Technology

Framework controls are developed using web development tools.

- HTML: Type and structure of the elements of the framework control
- CSS: Design of the elements of the framework control
- JavaScript: Logic behind the elements of the framework control

Developers are no longer bound to the TwinCAT HMI controls and can use all the elements and functionalities they are already familiar with from web development within a framework control. The standard TwinCAT HMI controls are implemented on the same basis as a framework control. It is also possible to reuse the TwinCAT HMI controls in the framework control or to add user-specific properties.

Application scenarios

Framework controls can map customer-specific and application-specific requirements. The following use cases can be implemented with a framework control:

- Implementation of complex JavaScript logic
- Reusing HTML, JS and CSS in a control
- Integration of external libraries and realization of own libraries
- Dynamic elements: Adding and removing HTML elements and controls at runtime
- Developing responsive controls
- Realizing and integrating 3D models

Use

The framework controls can be integrated directly as references in a TwinCAT HMI project if both projects are in the same solution. Furthermore, it is possible to export the framework control as a package [▶ 119] and install it in different HMI projects. In this way, it is possible to implement application-specific control libraries. Within the HMI project, a framework control is used like a standard TwinCAT HMI control.

15.3.3 First steps

1. Open Visual Studio and click **Add new project**. Select the **Extensibility** category under TwinCAT HMI as the project type.

2. Right-click on the project and execute a project build (Build Project).

3. Add a new HMI project to the existing solution by right-clicking the **Add new project** solution. Select an HMI project.

4. Right-click on the "Framework" node in the HMI project and select "Manage Extensions...".

5. Select **Framework Control** in the **TwinCAT HMI Extension Management** window and confirm the dialog with **OK**.

6. If you have opened a View/Content/UserControl file, you have to reload the page. This is illustrated by a note. Click Reload.

⇨ The framework control is now available in the toolbox.

7. Use drag & drop to add the framework control to an open View/Content/UserControl file.

⇨ You can now use the framework control like a normal control.

All changes in the source code of the framework control require a project build (see step 2) to ensure that the changes are adopted in the Engineering.

15.3.4 Development environment

A framework control project is created within a solution in Visual Studio. Multiple framework control projects and associated HMI projects can be placed in a solution. A framework control project can be configured using the standard Visual Studio functions (for example, Build Solution).

A framework control project contains the following elements when it is created:

1. **References**: This project node is not required and is automatically provided by the project type in Visual Studio.
2. **Icons**: This folder contains all icons relating to the control.
3. **Icon**: The .png elements are control icons in different sizes.
4. Schema [▶ 2153]: This folder contains JSON schema files that are required for the control. Additional JSON schema files for project-specific data types can be added to this folder.
5. **ControlDescription.Schema.json**: Standard schema [▶ 2123] describing Description.json [▶ 2123].
6. Themes [▶ 2141]: This folder contains all themes relating to the control. New themes can be added to this folder.
7. **Base**: This folder contains the basic theme of the control. Additional theme-based files can be added to this folder.
8. **Image**: This folder serves as a container for images that are assigned to the base theme.
9. Style.css [▶ 2141]: This stylesheet file contains the properties for the base theme.
10. Description.json [▶ 2123]: This file contains all Engineering interfaces for the TwinCAT HMI
11. Mainfest.json [▶ 2163]: This file describes control data required for creating a TwinCAT HMI package [▶ 118].
12. Source.js [▶ 2144]: This file contains the JavaScript logic required for the control.
13. Style.css [▶ 2141]: This file contains all style properties that are to apply to the control regardless of the theme.
14. Template.html [▶ 2140]: This file contains the static HTML framework of the control and is included in the HMI as HTML code for the control during subsequent instantiation on the HMI.

Please note from version 1.8

To program framework controls, a full version of Visual Studio is required for editing the HTML, CSS, and JavaScript code. The Visual Studio shell is not sufficient.

16 Samples

16.1 Controls

16.1.1 Barchart

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007203226236427.zip

16.1.2 Combobox

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/3922137483.zip

16.1.3 Datagrid

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007203184343435.zip

16.1.4 Grid

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/4010038539.zip

16.1.5 Event Grid

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007204494484107.zip

16.1.6 Linechart

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007203177442059.zip

16.1.7 Trend Line Chart

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/5287752203.zip

16.2 Recipe management

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/9007205127463307.zip

https://infosys.beckhoff.com/content/1033/TE2000_TC3_HMI_Engineering/Resources/zip/6812840587.zip

17 FAQ

17.1 General

Which system requirements do I need for the TwinCAT HMI?

An overview of the system requirements can be found in the system requirements. [▶ 17]

Which browser can I use for the TwinCAT HMI client?

An overview of the recommended HTML5-capable browsers can be found in the system requirements [▶ 17].

Can I use projects from the TC3 PLC HMI (TF1800) in the TwinCAT HMI?

The use of projects created with the TC3 PLC HMI is not possible.

Is there a tutorial that will help me get started with TwinCAT HMI?

The QuickStart [▶ 27] offers a good introduction to the TwinCAT HMI.

Is there a training course for the TwinCAT HMI?

There are three different training courses for the TwinCAT HMI:

1. TR7050: Basics TwinCAT HMI (3 days).
2. TR7060: Extension of TwinCAT HMI Controls [▶ 2163] (1 day).
3. TR7065: Extension of TwinCAT HMI Server Extensions [▶ 610] (1 day).

You will find an overview of the training courses and upcoming dates on our website.

Are there sample projects for the TwinCAT HMI?

An overview of the sample projects can be found in the chapter Samples [▶ 2169].

17.2 Server

How do I transfer the project to my target system?

Carry out the following steps to transfer the project to the target system:

1. Install the TwinCAT HMI Server (install TF2000) on the target system.
2. Start the TwinCAT HMI Server on the target system and assign an administrator password on the configuration page of the server.
3. Perform a project publish [▶ 110] to the target system.

I have no access to the target system. Can I transfer the project offline?

A transfer of the project without access to the target system is possible. To do this, perform the following steps:

4. Install the TwinCAT HMI Server (install TF2000) on the target system and on your engineering system.
5. Perform a project publish [▶ 110] on the TF2000 TwinCAT HMI Server on your engineering system.
6. Copy the entire contents of the folder: *C:\ProgramData\Beckhoff\TF2000 TwinCAT 3 HMI Server.*
7. Transfer the folder content to the target system (e.g. via USB stick).

8. Insert the folder content into the following folder on the target system: *C:\ProgramData\Beckhoff\TF2000 TwinCAT 3 HMI Server.*
9. Start the TwinCAT HMI Server on the target system.

⇨ You can now use the project.

The publish process failed due to a version difference. What can I do?

For a project publish, the server version (TF2000) on the target system must match the engineering version (TE2000) (versions 1.8 and 1.10). Please observe the version overview [▶ 17].

I forgot the password of the user "__SystemAdministrator". What can I do?

If you have forgotten the password of the user "__SystemAdministrator", there are the following possibilities:

1. You have another user who is a member of the "__SystemAdministrators" group. Log in with this user on the configuration page of the server and reset the password of the user "__SystemAdministrator".
2. Reset the server.

How can I reset the server?

To reset the server, perform the following steps:

1. Stop the server via the SysTray icon by clicking on "Shutdown".
2. Start the "TwinCAT HMI Server Factory reset" shortcut. Confirm the confirmation prompt with "Yes". The server starts up briefly and performs a reset.
3. Restart the TwinCAT HMI Server. The server is now reset. The password of the user "__SystemAdministrator" must be initialized again.

I cannot find the target server with the Discovery function in the Publish dialog. What can I do?

This can have the following causes:

1. The password of the user "__SystemAdministrator" was not initialized. For security reasons, the server cannot be found using the discovery function.
   - **Solution 1**: Enter the IP address of the destination server under Connection Settings. Click "Validate Connection". In this case, the dialog for initializing the "__SystemAdministrator" password in engineering opens.
   - **Solution 2**: Initialize the user "__SystemAdministrator" on the target server.
2. The password of the user "__SystemAdministrator" has not been initialized and the server has been starting for more than five minutes. The server must be restarted for security reasons.
   - **Solution:** Connect to the target system, restart the server and initialize the user "__SystemAdministrator".
3. There are connection problems. Check the steps in the question below.

I can't connect to my target server. What can I do?

This can have the following causes:

1. There is no network connection to the system.
   - Check the network connection to the system (e.g. using a ping command).
2. The system's firewall is blocking access.
   - Check the firewall settings on the target system and enable the port on which the TwinCAT HMI Server is running (e.g. port 1010 for http or 1020 for https).
3. The port that you want to use to access the server has not been released in the server.
   - Add Port on the configuration page of the server.

How can I share an additional port on the server?

How to share additional ports on the server can be found here.

The server reports "License error". What can I do?

The TwinCAT HMI Server can only be started if a valid license is available on the target system. Activate a license [▶ 21] on the target system.

How can I start the TwinCAT HMI Server automatically?

A recommendation on this topic can be found in the chapter Autostart Client [▶ 114].

17.3 Client

How can I access the HMI with a client?

After you have published a project on the server, you can access the system from any client. Open a browser and enter the URL of the server. This usually consists of the IP address and the port number.

- http: http://127.0.0.1:1010
- https: https://127.0.0.1:1020

How can I access the HMI from a mobile end device?

You can access the HMI from mobile end devices in the same way as from desktop PCs.

How can I establish an encrypted connection to a mobile end device?

Some mobile browsers block connections with self-signed certificates. In this case, you can install the certificate on the device:

1. Export the self-signed certificate of the server.
2. Transfer the certificate to the mobile device. You can, for example, send it to the device by email.
3. Install the certificate on the mobile end device (click on the certificate and follow the wizard).

⇨ You can now access the HMI.

If you do not want to install the certificate on the device and want to provide a general secure access from mobile end devices, you can issue an official certificate of a CA and import it into the server.

Can I start the client automatically?

The client can be started automatically. Some browsers also support starting in full screen mode. Information on this topic can be found in the Autostart Client [▶ 115] chapter.

17.4 Communication

Which communication protocols does TwinCAT HMI support?

The TwinCAT HMI supports communication via ADS to TwinCAT systems [▶ 600] and communication via OPC UA to OPC UA servers [▶ 601]. In addition, the TwinCAT HMI servers can exchange data with each other via server-to-server communication [▶ 606].

A server extension [▶ 610] offers further communication options.

How do I use the ADS extension?

You can configure the connection to TwinCAT systems in the ADS extension. You will find instructions on how to do this in the ADS chapter [▶ 600]. The data points can then be used [▶ 92] in TwinCAT HMI.

How do I use the OPC UA Client Extension?

In the OPC UA Client Extension you can configure connections to OPC UA Servers. Instructions can be found in the chapter OPC UA Client [▶ 601]. The data points can then be used [▶ 92] in TwinCAT HMI.

How can I check the connection status to my target system?

With the ADS extension and with the OPC UA client extension you can click on "Diagnosis" on the configuration page of the server [▶ 117]. There you can view the current connection status.

I cannot establish a connection to my TwinCAT system. What can I do?

Check whether a route to TwinCAT exists. The TwinCAT system must be in Run mode and the PLC must be started.

I have transferred my project to the target server. Afterwards, no data points from the TwinCAT system are displayed. The data points were displayed on the engineering system. What could be the cause?

The following causes are possible:

1. In the TwinCAT HMI Server, a distinction is made between local profiles and remote profiles [▶ 117]. With a project publish, [▶ 110] a remote profile (selected in the Publish dialog) is transferred to the target server. Check the connection settings of the ADS extension for the remote profile.
2. There are not enough target licenses on the target server. One target license is required per TwinCAT system. Activate [▶ 21] additional target licenses.

I cannot transfer my project with OPC-UA-Client-Extension to the target system. What could be the cause?

The OPC-UA-Client-Extension requires the license TF2110. Activate this license [▶ 21] on the target system and run the project publish [▶ 110] again.

Can a TwinCAT HMI server communicate with another TwinCAT HMI server?

TwinCAT HMI servers can communicate with each other via server-to-server communication [▶ 606].

17.5 Framework

How can I realize a page switching?

Page switching can be implemented via the control TcHmiRegion [▶ 496]. You will find instructions on how to do this in the chapter Creating Navigation [▶ 34].

Can I develop my own controls for the TwinCAT HMI?

The development of user-specific controls in HTML5, JavaScript and CSS 3 is possible via so-called "Framework Controls [▶ 2163]".

How can I integrate a Framework Control into other TwinCAT HMI projects?

A Framework Control can be exported as a package [▶ 119] via Package Management and installed [▶ 123] in other TwinCAT HMI projects.

Is there a training for the development of Framework Controls?

The one-day course TR7060 can be attended for the development of Framework Controls. The dates can be found on our website.

Can I integrate external libraries into the TwinCAT HMI?

The TwinCAT HMI Framework offers an API [▶ 1112] that can be used to implement many functions. The integration of external web libraries is possible within a Framework Control. [▶ 2163] The external library must be integrated into the Framework Control project. In the Description.json [▶ 2123] a reference [▶ 2127] to the files of the library must be created.

I have created a Framework Control. This is not displayed in the TwinCAT HMI Designer or in the LiveView. What could be the cause?

JavaScript is an interpreted scripting language, therefore syntax errors etc. can only be detected at runtime. Such an error may result in the Control not being displayed in the Designer or LiveView. The developer tools [▶ 2286] of the browsers offer a possibility for diagnosis. These developer tools also allow you to debug a control.

18 Appendix

18.1 Schema

18.1.1 General

18.1.1.1 tchmi:general#/definitions/Any

[Schema]

```
{
}
```

Note: Available since version 1.8

See also

API: any [▶ 2107]

18.1.1.2 tchmi:general#/definitions/Array

[Schema]

```
{
    "id": "tchmi:general#/definitions/Array",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: any [▶ 2107][ ]

18.1.1.3 tchmi:general#/definitions/BINARY

[Schema]

```
{
    "format": "base64",
    "id": "tchmi:general#/definitions/BINARY",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: BINARY

18.1.1.4 tchmi:general#/definitions/BIT

[Schema]

```
{
    "default": false,
    "id": "tchmi:general#/definitions/BIT",
    "type": "boolean"
}
```

Note: Available since version 1.8

See also

Data type in ADS: BIT

18.1.1.5 tchmi:general#/definitions/BIT8

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/BIT8",
    "maximum": 255,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: BIT8

18.1.1.6 tchmi:general#/definitions/BITARR8

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/BITARR8",
    "maximum": 255,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: BITARR8

18.1.1.7 tchmi:general#/definitions/BOOL

[Schema]

```
{
    "default": false,
    "id": "tchmi:general#/definitions/BOOL",
    "type": "boolean"
}
```

Note: Available since version 1.8

See also

Data type in ADS: BOOL

API: boolean

18.1.1.8 tchmi:general#/definitions/Boolean

[Schema]

```
{
    "default": false,
    "id": "tchmi:general#/definitions/Boolean",
    "type": "boolean"
}
```

Note: Available since version 1.8

See also

API: boolean [▶ 2107]

18.1.1.9 tchmi:general#/definitions/BYTE

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/BYTE",
    "maximum": 255,
    "minimum": 0,
    "type": "integer"
}
```

See also

Data type in ADS: BYTE

18.1.1.10 tchmi:general#/definitions/DATE

[Schema]

```
{
    "default": "",
    "format": "datetime",
    "id": "tchmi:general#/definitions/DATE",
    "type": "string"
}
```

Note: Available since version 1.8

A string representation according to ISO 8601 is used in the communication with the client, whereas the native data type is used for communication with the PLC via ADS.

See also

Data type in ADS: DATE

18.1.1.11 tchmi:general#/definitions/DATE_AND_TIME

[Schema]

```
{
    "default": "",
    "format": "datetime",
    "id": "tchmi:general#/definitions/DATE_AND_TIME",
    "type": "string"
}
```

Note: Available since version 1.8

A string representation according to ISO 8601 is used in the communication with the client, whereas the native data type is used for communication with the PLC via ADS.

See also

Data type in ADS: DATE_AND_TIME

18.1.1.12 tchmi:general#/definitions/DateTime

[Schema]

```
{
    "default": "",
    "format": "datetime",
    "id": "tchmi:general#/definitions/DateTime",
    "type": "string"
}
```

18.1.1.13 tchmi:general#/definitions/DATETIME

[Schema]

```
{
    "default": "",
    "format": "datetime",
    "id": "tchmi:general#/definitions/DATETIME",
    "type": "string"
}
```

Note: Available since version 1.8

A string representation according to ISO 8601 is used in the communication with the client, whereas the native data type is used for communication with the PLC via ADS.

See also

Data type in ADS: DATETIME

18.1.1.14 tchmi:general#/definitions/DINT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/DINT",
    "maximum": 2147483647,
    "minimum": -2147483648,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: DINT

API: number [▶ 2108]

18.1.1.15 tchmi:general#/definitions/DOUBLE

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/DOUBLE",
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: DOUBLE

API: number [▶ 2108]

18.1.1.16 tchmi:general#/definitions/DT

[Schema]

```
{
    "default": "",
    "format": "datetime",
    "id": "tchmi:general#/definitions/DT",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: DT

18.1.1.17 tchmi:general#/definitions/DWORD

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/DWORD",
    "maximum": 4294967295,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: DWORD

API: number [▶ 2108]

18.1.1.18 tchmi:general#/definitions/FLOAT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/FLOAT",
    "maximum": 3.4e+38,
    "minimum": -3.4e+38,
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: FLOAT

API: number [▶ 2108]

18.1.1.19 tchmi:general#/definitions/HRESULT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/HRESULT",
    "maximum": 4294967295,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: HRESULT

18.1.1.20 tchmi:general#/definitions/INT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/INT",
    "maximum": 32767,
    "minimum": -32768,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: INT

API: number [▶ 2108]

18.1.1.21 tchmi:general#/definitions/INT16

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/INT16",
    "maximum": 32767,
    "minimum": -32768,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: INT16

API: number [▶ 2108]

18.1.1.22 tchmi:general#/definitions/INT32

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/INT32",
    "maximum": 2147483647,
    "minimum": -2147483648,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: INT32

API: number [▶ 2108]

18.1.1.23 tchmi:general#/definitions/INT64

[Schema]

```
{
    "format": "int64",
    "id": "tchmi:general#/definitions/INT64",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: INT64

API: number [▶ 2108]

18.1.1.24 tchmi:general#/definitions/INT8

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/INT8",
    "maximum": 127,
    "minimum": -128,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: INT8

API: number [▶ 2108]

18.1.1.25 tchmi:general#/definitions/Integer

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/Integer",
    "type": "integer"
}
```

Note: Available since version 1.8

See also

API: number [▶ 2108]

18.1.1.26 tchmi:general#/definitions/LINT

[Schema]

```
{
    "format": "int64",
    "id": "tchmi:general#/definitions/LINT",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: LINT

18.1.1.27 tchmi:general#/definitions/Locale

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": " tchmi:general#/definitions/Locale",
    "type": "string",
    "enum": [
        "",
        "af",
        "af-ZA",
        "ar",
        "ar-AE",
        "ar-BH",
        "ar-DZ",
        "ar-EG",
        "ar-IQ",
        "ar-JO",
        "ar-KW",
        "ar-LB",
        "ar-LY",
        "ar-MA",
        "ar-OM",
        "ar-QA",
        "ar-SA",
        "ar-SY",
        "ar-TN",
        "ar-YE",
        "az",
        "az-AZ",
        "az-AZ",
        "be",
        "be-BY",
        "bg",
        "bg-BG",
        "bs-BA",
        "ca",
        "ca-ES",
        "cs",
        "cs-CZ",
        "cy",
        "cy-GB",
        "da",
        "da-DK",
        "de",
        "de-AT",
        "de-CH",
        "de-DE",
        "de-LI",
        "de-LU",
        "dv",
        "dv-MV",
        "el",
        "el-GR",
        "en",
        "en-AU",
        "en-BZ",
        "en-CA",
        "en-CB",
        "en-GB",
        "en-IE",
        "en-JM",
        "en-NZ",
        "en-PH",
        "en-TT",
        "en-US",
        "en-ZA",
        "en-ZW",
        "eo",
        "es",
        "es-AR",
        "es-BO",
```

```
        "es-CL",
        "es-CO",
        "es-CR",
        "es-DO",
        "es-EC",
        "es-ES",
        "es-ES",
        "es-GT",
        "es-HN",
        "es-MX",
        "es-NI",
        "es-PA",
        "es-PE",
        "es-PR",
        "es-PY",
        "es-SV",
        "es-UY",
        "es-VE",
        "et",
        "et-EE",
        "eu",
        "eu-ES",
        "fa",
        "fa-IR",
        "fi",
        "fi-FI",
        "fo",
        "fo-FO",
        "fr",
        "fr-BE",
        "fr-CA",
        "fr-CH",
        "fr-FR",
        "fr-LU",
        "fr-MC",
        "gl",
        "gl-ES",
        "gu",
        "gu-IN",
        "he",
        "he-IL",
        "hi",
        "hi-IN",
        "hr",
        "hr-BA",
        "hr-HR",
        "hu",
        "hu-HU",
        "hy",
        "hy-AM",
        "id",
        "id-ID",
        "is",
        "is-IS",
        "it",
        "it-CH",
        "it-IT",
        "ja",
        "ja-JP",
        "ka",
        "ka-GE",
        "kk",
        "kk-KZ",
        "kn",
        "kn-IN",
        "ko",
        "ko-KR",
        "kok",
        "kok-IN",
        "ky",
        "ky-KG",
        "lt",
        "lt-LT",
        "lv",
        "lv-LV",
        "mi",
        "mi-NZ",
        "mk",
        "mk-MK",
        "mn",
```

```
        "mn-MN",
        "mr",
        "mr-IN",
        "ms",
        "ms-BN",
        "ms-MY",
        "mt",
        "mt-MT",
        "nb",
        "nb-NO",
        "nl",
        "nl-BE",
        "nl-NL",
        "nn-NO",
        "ns",
        "ns-ZA",
        "pa",
        "pa-IN",
        "pl",
        "pl-PL",
        "ps",
        "ps-AR",
        "pt",
        "pt-BR",
        "pt-PT",
        "qu",
        "qu-BO",
        "qu-EC",
        "qu-PE",
        "ro",
        "ro-RO",
        "ru",
        "ru-RU",
        "sa",
        "sa-IN",
        "se",
        "se-FI",
        "se-FI",
        "se-FI",
        "se-NO",
        "se-NO",
        "se-NO",
        "se-SE",
        "se-SE",
        "se-SE",
        "sk",
        "sk-SK",
        "sl",
        "sl-SI",
        "sq",
        "sq-AL",
        "sr-BA",
        "sr-BA",
        "sr-SP",
        "sr-SP",
        "sv",
        "sv-FI",
        "sv-SE",
        "sw",
        "sw-KE",
        "syr",
        "syr-SY",
        "ta",
        "ta-IN",
        "te",
        "te-IN",
        "th",
        "th-TH",
        "tl",
        "tl-PH",
        "tn",
        "tn-ZA",
        "tr",
        "tr-TR",
        "tt",
        "tt-RU",
        "ts",
        "uk",
        "uk-UA",
        "ur",
```

```
        "ur-PK",
        "uz",
        "uz-UZ",
        "uz-UZ",
        "vi",
        "vi-VN",
        "xh",
        "xh-ZA",
        "zh",
        "zh-CN",
        "zh-HK",
        "zh-MO",
        "zh-SG",
        "zh-TW",
        "zu",
        "zu-ZA"
    ],
}
```

This data type contains the keywords of the most important languages of the world.

Note: Available since version 1.10.1336.10

See also tchmi:general#/definitions/ProjectLocale [▶ 2187]

18.1.1.28 tchmi:general#/definitions/LREAL

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/LREAL",
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: LREAL

API: number [▶ 2108]

18.1.1.29 tchmi:general#/definitions/LTIME

[Schema]

```
{
    "default": "",
    "format": "timespan",
    "id": "tchmi:general#/definitions/LTIME",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: LTIME

18.1.1.30 tchmi:general#/definitions/LWORD

[Schema]

```
{
    "format": "base64",
    "id": "tchmi:general#/definitions/LWORD",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: LWORD

18.1.1.31 tchmi:general#/definitions/Null

[Schema]

```
{
    "id": "tchmi:general#/definitions/Null"
}
```

Note: Available since version 1.8

See also

API: null [▶ 2108]

18.1.1.32 tchmi:general#/definitions/Number

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/Number",
    "type": "number"
}
```

Note: Available since version 1.8

See also

API: number [▶ 2108]

18.1.1.33 tchmi:general#/definitions/Object

[Schema]

```
{
    "id": "tchmi:general#/definitions/Object",
    "type": "object"
}
```

Note: Available since version 1.8

18.1.1.34 tchmi:general#/definitions/OTCID

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/OTCID",
    "maximum": 4294967295,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: OTCID

API: number [▶ 2108]

18.1.1.35 tchmi:general#/definitions/ProjectLocale

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": " tchmi:general#/definitions/ProjectLocale",
    "type": "string"
}
```

This data type contains the keywords of all languages that are set in the project in the Localization editor [▶ 90].

Note: Available since version 1.10.1336.10

Replaces the data type tchmi:framework#/definitions/Locale [▶ 2242]

See also tchmi:general#/definitions/Locale [▶ 2182] which allows all languages.

18.1.1.36 tchmi:general#/definitions/REAL

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/REAL",
    "maximum": 3.4e+38,
    "minimum": -3.4e+38,
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: REAL

API: number [▶ 2108]

18.1.1.37 tchmi:general#/definitions/REAL32

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/REAL32",
    "maximum": 3.4e+38,
    "minimum": -3.4e+38,
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: REAL32

API: number [▶ 2108]

18.1.1.38 tchmi:general#/definitions/REAL64

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/REAL64",
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: REAL64

API: number [▶ 2108]

18.1.1.39 tchmi:general#/definitions/REAL80

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/REAL80",
    "type": "number"
}
```

Note: Available since version 1.8

See also

Data type in ADS: REAL80

API: number [▶ 2108]

18.1.1.40 tchmi:general#/definitions/SINT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/SINT",
    "maximum": 127,
    "minimum": -128,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: SINT

API: number [▶ 2108]

18.1.1.41 tchmi:general#/definitions/String

[Schema]

```
{
    "default": "",
    "id": "tchmi:general#/definitions/String",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.1.42 tchmi:general#/definitions/STRING(80)

[Schema]

```
{
    "default": "",
    "id": "tchmi:general#/definitions/STRING(80)",
    "maxLength": 80,
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: STRING(80)

18.1.1.43 tchmi:general#/definitions/TIME

[Schema]

```
{
    "default": "",
    "format": "timespan",
    "id": "tchmi:general#/definitions/TIME",
    "type": "string"
}
```

Note: Available since version 1.8

A string representation according to ISO 8601 is used in the communication with the client, whereas the native data type is used for communication with the PLC via ADS.

See also

Data type in ADS: TIME

18.1.1.44 tchmi:general#/definitions/TIME_OF_DAY

[Schema]

```
{
    "default": "",
    "format": "timespan",
    "id": "tchmi:general#/definitions/TIME_OF_DAY",
    "type": "string"
}
```

Note: Available since version 1.8

A string representation according to ISO 8601 is used in the communication with the client, whereas the native data type is used for communication with the PLC via ADS.

See also

Data type in ADS: TIME_OF_DAY

18.1.1.45 tchmi:general#/definitions/TimeSpan

[Schema]

```
{
    "default": "",
    "format": "timespan",
    "id": "tchmi:general#/definitions/TimeSpan",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.1.46 tchmi:general#/definitions/UDINT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/UDINT",
    "maximum": 4294967295,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: UDINT

API: number [▶ 2108]

18.1.1.47 tchmi:general#/definitions/UINT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/UINT",
    "maximum": 65535,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: UINT

API: number [▶ 2108]

18.1.1.48 tchmi:general#/definitions/UINT16

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/UINT16",
    "maximum": 65535,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: UINT16

API: number [▶ 2108]

18.1.1.49 tchmi:general#/definitions/UINT32

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/UINT32",
    "maximum": 4294967295,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: UINT32

API: number [▶ 2108]

18.1.1.50 tchmi:general#/definitions/UINT64

[Schema]

```
{
    "format": "base64",
    "id": "tchmi:general#/definitions/UINT64",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: UINT64

API: number [▶ 2108]

18.1.1.51 tchmi:general#/definitions/UINT8

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/UINT8",
    "maximum": 255,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: UINT8

API: number [▶ 2108]

18.1.1.52 tchmi:general#/definitions/ULINT

[Schema]

```
{
    "format": "base64",
    "id": "tchmi:general#/definitions/ULINT",
    "type": "string"
}
```

Note: Available since version 1.8

See also

Data type in ADS: ULINT

API: number [▶ 2108]

18.1.1.53 tchmi:general#/definitions/USINT

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/USINT",
    "maximum": 255,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: USINT

API: number [▶ 2108]

18.1.1.54 tchmi:general#/definitions/WORD

[Schema]

```
{
    "default": 0,
    "id": "tchmi:general#/definitions/WORD",
    "maximum": 65535,
    "minimum": 0,
    "type": "integer"
}
```

Note: Available since version 1.8

See also

Data type in ADS: WORD

API: number [▶ 2108]

18.1.2 framework

18.1.2.1 tchmi:framework#/definitions/AccessConfig

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "access": {
            "id": "tchmi:framework#/definitions/access",
            "properties": {
                "accessright": {
                    "description": "Name of the access right.",
                    "id": "tchmi:framework#/definitions/accessright",
                    "minLength": 1,
                    "type": "string"
                },
                "group": {
                    "description": "Name of the user group.",
                    "id": "tchmi:framework#/definitions/group",
```

```
                    "minLength": 1,
                    "type": "string"
                },
                "permission": {
                    "description": "Name of the permission described by this rule.",
                    "enum": [
                        "Allow",
                        "Deny",
                        "Inherit"
                    ],
                    "id": "tchmi:framework#/definitions/permission",
                    "type": "string"
                }
            },
            "required": [
                "accessright",
                "group",
                "permission"
            ],
            "title": "Access",
            "type": "object"
        }
    },
    "id": "tchmi:framework#/definitions/AccessConfig",
    "items": {
        "id": "tchmi:framework#/definitions/AccessConfig/definitions/access",
        "properties": {
            "accessright": {
                "description": "Name of the access right.",
                "id": "tchmi:framework#/definitions/accessright",
                "minLength": 1,
                "type": "string"
            },
            "group": {
                "description": "Name of the user group.",
                "id": "tchmi:framework#/definitions/group",
                "minLength": 1,
                "type": "string"
            },
            "permission": {
                "description": "Name of the permission described by this rule.",
                "enum": [
                    "Allow",
                    "Deny",
                    "Inherit"
                ],
                "id": "tchmi:framework#/definitions/permission",
                "type": "string"
            }
        },
        "required": [
            "accessright",
            "group",
            "permission"
        ],
        "title": "Access",
        "type": "object"
    },
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.AccessControl [▶ 2079]

18.1.2.2 tchmi:framework#/definitions/Alarm

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Alarm",
    "properties": {
        "confirmationState": {
```

```
            "id": "tchmi:framework#/definitions/confirmationState",
            "type": "boolean"
        },
        "description": {
            "id": "tchmi:framework#/definitions/description",
            "type": "string"
        },
        "id": {
            "id": "tchmi:framework#/definitions/id",
            "type": "number"
        },
        "rawData": {
            "id": "tchmi:framework#/definitions/rawData",
            "type": "object"
        },
        "severity": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "Verbose",
                "Info",
                "Warning",
                "Error",
                "Critical"
            ],
            "id": "tchmi:framework#/definitions/MessageSeverity",
            "type": "string"
        },
        "text": {
            "id": "tchmi:framework#/definitions/text",
            "type": "string"
        },
        "timeCleared": {
            "format": "datetime",
            "id": "tchmi:framework#/definitions/timeCleared",
            "type": "string"
        },
        "timeConfirmed": {
            "format": "datetime",
            "id": "tchmi:framework#/definitions/timeConfirmed",
            "type": "string"
        },
        "timeRaised": {
            "format": "datetime",
            "id": "tchmi:framework#/definitions/timeRaised",
            "type": "string"
        },
        "type": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "MESSAGE",
                "ALARM"
            ],
            "id": "tchmi:framework#/definitions/MessageType",
            "type": "string"
        }
    },
    "required": [
        "type",
        "text",
        "severity",
        "timeRaised",
        "description",
        "id"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.2.3 tchmi:framework#/definitions/BorderRadius

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "unit": {
            "enum": [
```

```
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/unit",
            "type": "string"
        },
        "value": {
            "id": "tchmi:framework#/definitions/value",
            "minimum": 0,
            "type": "number"
        }
    },
    "id": "tchmi:framework#/definitions/BorderRadius",
    "properties": {
        "bottomLeft": {
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "bottomLeftUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/unit",
            "type": "string"
        },
        "bottomRight": {
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "bottomRightUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/unit",
            "type": "string"
        },
        "topLeft": {
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "topLeftUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/unit",
            "type": "string"
        },
        "topRight": {
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "topRightUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/BorderRadius/definitions/unit",
            "type": "string"
        }
    },
    "required": [
        "topLeft",
        "topRight",
        "bottomRight",
        "bottomLeft"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.BorderRadius [▶ 2081]

18.1.2.4 tchmi:framework#/definitions/BorderStyle

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "style": {
            "enum": [
                "Solid",
                "Dashed",
                "Dotted",
                "None"
            ],
            "id": "tchmi:framework#/definitions/style",
            "type": "string"
        }
    },
    "id": "tchmi:framework#/definitions/BorderStyle",
    "properties": {
        "bottom": {
            "enum": [
                "Solid",
                "Dashed",
                "Dotted",
                "None"
            ],
            "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
            "type": "string"
        },
        "left": {
            "enum": [
                "Solid",
                "Dashed",
                "Dotted",
                "None"
            ],
            "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
            "type": "string"
        },
        "right": {
            "enum": [
                "Solid",
                "Dashed",
                "Dotted",
                "None"
            ],
            "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
            "type": "string"
        },
        "top": {
            "enum": [
                "Solid",
                "Dashed",
                "Dotted",
                "None"
            ],
            "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
            "type": "string"
        }
    },
    "required": [
        "left",
        "top",
        "right",
        "bottom"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.BorderStyle [▶ 2081]

18.1.2.5 tchmi:framework#/definitions/BorderWidth

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "BorderWidth": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "type": "object",
            "properties": {
        "left": {
            "$ref": "tchmi:framework#/definitions/PositiveNumber"
        },
        "leftUnit": {
            ref": "tchmi:framework#/definitions/PixelUnit"
        },
        "top": {
            ": "tchmi:framework#/definitions/PositiveNumber"
        },
        "topUnit": {
            "$ref": "tchmi:framework#/definitions/PixelUnit"
        },
        "right": {
            ": "tchmi:framework#/definitions/PositiveNumber"
        },
        "rightUnit": {
            "$ref": "tchmi:framework#/definitions/PixelUnit"
        },
        "bottom": {
            ": "tchmi:framework#/definitions/PositiveNumber"
        },
        bottomUnit": {
            ": "tchmi:framework#/definitions/PixelUnit"
        }
    },
    "required": [ "left", "top", "right", "bottom" ]
    }
}
```

Note: Available since version 1.8

See also

API: TcHmi.FourSidedCss [▶ 2088]

18.1.2.6 tchmi:framework#/definitions/BoxShadowList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/BoxShadowList",
    "items": {
        "properties": {
            "blur": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "blurUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px"
                ],
                "id": "tchmi:framework#/definitions/PixelUnit",
                "type": "string"
            },
            "color": {
```

```
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "inset": {
                "id": "tchmi:framework#/definitions/inset",
                "type": "boolean"
            },
            "offsetX": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "offsetXUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px"
                ],
                "id": "tchmi:framework#/definitions/PixelUnit",
                "type": "string"
            },
            "offsetY": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "offsetYUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px"
                ],
                "id": "tchmi:framework#/definitions/PixelUnit",
                "type": "string"
            },
            "spread": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "spreadUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px"
                ],
                "id": "tchmi:framework#/definitions/PixelUnit",
                "type": "string"
            }
        },
        "required": [
            "color",
            "offsetX",
            "offsetY"
        ],
        "title": "BoxShadow definition",
        "type": "object"
    },
    "minItems": 0,
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.BoxShadow [▶ 2083]

18.1.2.7 tchmi:framework#/definitions/CellOptions

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "description": "Definition of cell options in the grid.",
    "id": "tchmi:framework#/definitions/CellOptions",
    "items": {
        "description": "Definition of a cell.",
        "properties": {
            "backgroundColor": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "backgroundImage": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/Path",
                "minLength": 1,
                "type": "string"
            },
            "backgroundImageHeight": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "backgroundImageHeightUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            },
            "backgroundImageHorizontalAlignment": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "Left",
                    "Center",
                    "Right"
                ],
                "id": "tchmi:framework#/definitions/HorizontalAlignment",
                "type": "string"
            },
            "backgroundImagePadding": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "definitions": {
                    "unit": {
                        "enum": [
                            "px",
                            "%"
                        ],
                        "id": "tchmi:framework#/definitions/unit",
                        "type": "string"
                    },
                    "value": {
                        "id": "tchmi:framework#/definitions/value",
                        "minimum": 0,
                        "type": "number"
                    }
                },
                "id": "tchmi:framework#/definitions/Padding",
                "properties": {
                    "bottom": {
                        "id": "tchmi:framework#/definitions/Padding/definitions/value",
```

```
                    "minimum": 0,
                    "type": "number"
                },
                "bottomUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                },
                "left": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "leftUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                },
                "right": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "rightUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                },
                "top": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "topUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                }
            },
            "required": [
                "left",
                "top",
                "right",
                "bottom"
            ],
            "type": "object"
        },
        "backgroundImageVerticalAlignment": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "Top",
                "Center",
                "Bottom"
            ],
            "id": "tchmi:framework#/definitions/VerticalAlignment",
            "type": "string"
        },
        "backgroundImageWidth": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "id": "tchmi:framework#/definitions/MeasurementValue",
            "type": "number"
        },
        "backgroundImageWidthUnit": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "px",
                "%"
            ],
```

```
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            },
            "borderColor": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "borderStyle": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "definitions": {
                    "style": {
                        "enum": [
                            "Solid",
                            "Dashed",
                            "Dotted",
                            "None"
                        ],
                        "id": "tchmi:framework#/definitions/style",
                        "type": "string"
                    }
                },
                "id": "tchmi:framework#/definitions/BorderStyle",
                "properties": {
                    "bottom": {
                        "enum": [
                            "Solid",
                            "Dashed",
                            "Dotted",
                            "None"
                        ],
                        "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
                        "type": "string"
                    },
                    "left": {
                        "enum": [
                            "Solid",
                            "Dashed",
                            "Dotted",
                            "None"
                        ],
                        "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
                        "type": "string"
                    },
                    "right": {
                        "enum": [
                            "Solid",
                            "Dashed",
                            "Dotted",
                            "None"
                        ],
                        "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
                        "type": "string"
                    },
                    "top": {
                        "enum": [
                            "Solid",
                            "Dashed",
                            "Dotted",
                            "None"
                        ],
                        "id": "tchmi:framework#/definitions/BorderStyle/definitions/style",
                        "type": "string"
                    }
                },
                "required": [
                    "left",
                    "top",
```

```
                "right",
                "bottom"
            ],
            "type": "object"
        },
        "borderWidth": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "definitions": {
                "unit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/unit",
                    "type": "string"
                },
                "value": {
                    "id": "tchmi:framework#/definitions/value",
                    "minimum": 0,
                    "type": "number"
                }
            },
            "id": "tchmi:framework#/definitions/BorderWidth",
            "properties": {
                "bottom": {
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "bottomUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/unit",
                    "type": "string"
                },
                "left": {
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "leftUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/unit",
                    "type": "string"
                },
                "right": {
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "rightUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/unit",
                    "type": "string"
                },
                "top": {
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "topUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/BorderWidth/definitions/unit",
                    "type": "string"
                }
            },
            "required": [
                "left",
                "top",
```

```
                "right",
                "bottom"
            ],
            "type": "object"
        },
        "columnIndex": {
            "id": "tchmi:framework#/definitions/columnIndex",
            "minimum": 0,
            "type": "integer"
        },
        "padding": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "definitions": {
                "unit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/unit",
                    "type": "string"
                },
                "value": {
                    "id": "tchmi:framework#/definitions/value",
                    "minimum": 0,
                    "type": "number"
                }
            },
            "id": "tchmi:framework#/definitions/Padding",
            "properties": {
                "bottom": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "bottomUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                },
                "left": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "leftUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                },
                "right": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "rightUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
                },
                "top": {
                    "id": "tchmi:framework#/definitions/Padding/definitions/value",
                    "minimum": 0,
                    "type": "number"
                },
                "topUnit": {
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/Padding/definitions/unit",
                    "type": "string"
```

```
                    }
                },
                "required": [
                    "left",
                    "top",
                    "right",
                    "bottom"
                ],
                "type": "object"
            },
            "rowIndex": {
                "id": "tchmi:framework#/definitions/rowIndex",
                "minimum": 0,
                "type": "integer"
            }
        },
        "propertiesMeta": [
            {
                "category": "Layout",
                "description": "Column index of the defined cell.",
                "displayName": "Column Index",
                "displayPriority": 10,
                "name": "columnIndex"
            },
            {
                "category": "Layout",
                "description": "Row index of the defined cell.",
                "displayName": "Row Index",
                "displayPriority": 10,
                "name": "rowIndex"
            },
            {
                "category": "Layout",
                "description": "Inner padding of the cell.",
                "displayName": "Padding",
                "displayPriority": 60,
                "name": "padding"
            },
            {
                "category": "Colors",
                "description": "Definition of the background color which is used if no or a
transparent Background image is set.",
                "displayName": "Background Color",
                "displayPriority": 10,
                "name": "backgroundColor"
            },
            {
                "category": "Colors",
                "description": "Definition of the border color.",
                "displayName": "Border Color",
                "displayPriority": 10,
                "name": "borderColor"
            },
            {
                "category": "Border",
                "description": "Definition of the border width.",
                "displayName": "Border Width",
                "displayPriority": 10,
                "name": "borderWidth"
            },
            {
                "category": "Border",
                "description": "Definition of the border style.",
                "displayName": "Border Style",
                "displayPriority": 10,
                "name": "borderStyle"
            },
            {
                "category": "Background Image",
                "description": "Definition of the border style.",
                "displayName": "Background Image",
                "displayPriority": 10,
                "name": "backgroundImage"
            },
            {
                "category": "Background Image",
                "description": "Width of the background image.",
                "displayName": "Background Image Width",
                "displayPriority": 20,
                "name": "backgroundImageWidth"
```

```
            },
            {
                "category": "Background Image",
                "defaultValue": null,
                "defaultValueInternal": "px",
                "description": "Width Unit of the background image.",
                "displayName": "Background Image Width Unit",
                "displayPriority": 20,
                "name": "backgroundImageWidthUnit"
            },
            {
                "category": "Background Image",
                "description": "Height of the background image.",
                "displayName": "Background Image Height",
                "displayPriority": 20,
                "name": "backgroundImageHeight"
            },
            {
                "category": "Background Image",
                "defaultValue": null,
                "defaultValueInternal": "px",
                "description": "Height Unit of the background image.",
                "displayName": "Background Image Height Unit",
                "displayPriority": 20,
                "name": "backgroundImageHeightUnit"
            },
            {
                "category": "Background Image",
                "defaultValue": null,
                "defaultValueInternal": "Center",
                "description": "Horizontal Aligment of the background image.",
                "displayName": "Background Image Horizontal Aligment",
                "displayPriority": 30,
                "name": "backgroundImageHorizontalAlignment"
            },
            {
                "category": "Background Image",
                "defaultValue": null,
                "defaultValueInternal": "Center",
                "description": "Vertical Aligment of the background image.",
                "displayName": "Background Image Vertical Aligment",
                "displayPriority": 30,
                "name": "backgroundImageVerticalAlignment"
            },
            {
                "category": "Background Image",
                "description": "Padding of the background image.",
                "displayName": "Background Image Padding",
                "displayPriority": 60,
                "name": "backgroundImagePadding"
            }
        ],
        "title": "Cell definition",
        "type": "object"
    },
    "title": "CellOptions",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.System.TcHmiGrid.ICellOptions [▶ 1205]

18.1.2.8 tchmi:framework#/definitions/ChartBarGraphColorList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "anyOf": [
        {
            "items": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
```

```
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "type": "array"
        },
        {
            "$schema": "http://json-schema.org/draft-04/schema",
            "id": "tchmi:framework#/definitions/SolidColor",
            "properties": {
                "color": {
                    "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                    "id": "tchmi:framework#/definitions/color",
                    "minLength": 1,
                    "type": "string"
                }
            },
            "required": [
                "color"
            ],
            "type": "object"
        }
    ],
    "id": "tchmi:framework#/definitions/ChartBarGraphColorList",
    "title": "ChartBarGraphColorList"
}
```

Note: Available since version 1.8

See also

API: TcHmi.SolidColor [▶ 2084]

18.1.2.9 tchmi:framework#/definitions/ChartBarGraphDataDefinitionList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "anyOf": [
        {
            "items": {
                "items": {
                    "type": "number"
                },
                "type": "array"
            },
            "title": "Array<Array<number>>",
            "type": "array"
        },
        {
            "items": {
                "type": "number"
            },
            "title": "Array<number>",
            "type": "array"
        }
    ],
    "id": "tchmi:framework#/definitions/ChartBarGraphDataDefinitionList",
    "title": "ChartBarGraphDataDefinitionList"
}
```

Note: Available since version 1.8

See also

API: number [▶ 2108]

18.1.2.10 tchmi:framework#/definitions/ChartBarGraphPosition

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Left",
        "Right"
    ],
    "id": "tchmi:framework#/definitions/ChartBarGraphPosition",
    "title": "ChartBarGraphPosition",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.11 tchmi:framework#/definitions/ChartLineGraphDataDefinitionList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/ChartLineGraphDataDefinitionList",
    "items": {
        "items": {
            "properties": {
                "x": {
                    "id": "tchmi:framework#/definitions/x",
                    "type": "number"
                },
                "y": {
                    "id": "tchmi:framework#/definitions/y",
                    "type": "number"
                }
            },
            "required": [
                "x",
                "y"
            ],
            "type": "object"
        },
        "type": "array"
    },
    "title": "ChartLineGraphDataDefinitionList",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmiCharting.LineGraph.Point [▶ 2109]

18.1.2.12 tchmi:framework#/definitions/ChartLineGraphDescriptionDefinitionList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/ChartLineGraphDescriptionDefinitionList",
    "items": {
```

```
        "properties": {
            "lineColor": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "lineWidth": {
                "id": "tchmi:framework#/definitions/lineWidth",
                "type": "number"
            },
            "pointDot": {
                "id": "tchmi:framework#/definitions/pointDot",
                "type": "boolean"
            },
            "pointDotFillColor": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "pointDotRadius": {
                "id": "tchmi:framework#/definitions/pointDotRadius",
                "minimum": 0,
                "type": "number"
            },
            "pointDotStrokeColor": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "pointDotStrokeWidth": {
                "id": "tchmi:framework#/definitions/pointDotStrokeWidth",
                "minimum": 0,
                "type": "number"
            },
            "yAxisId": {
                "id": "tchmi:framework#/definitions/yAxisId",
                "type": "number"
            }
        },
        "required": [
            "yAxisId"
        ],
        "type": "object"
    },
```

```
    "title": "ChartLineGraphDescriptionDefinitionList",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiLineChart.lineGraphDescription [▶ 1612][]

18.1.2.13 tchmi:framework#/definitions/ChartMenuBarPosition

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "ChartMenuBarPosition": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "ChartMenuBarPosition",
      "type": "string",
      "enum": [ "Bottom", "Top" ]
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.MenuBarPosition [▶ 1810]

18.1.2.14 tchmi:framework#/definitions/ChartMouseMode

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "ChartMouseMode": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "ChartMouseMode",
      "type": "string",
      "enum": [ "ZoomX", "ZoomXY", "PanX", "PanXY" ]
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.MouseMode [▶ 1811]

18.1.2.15 tchmi:framework#/definitions/ChartYAxisDefinition

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "ChartYAxisDefinition": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "ChartYAxisDefinition",
      "type": "object",
      "propertiesMeta": [
        {
          "name": "labelFontColor",
          "category": "Colors",
```

```
          "displayName": "Label font color",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": {
            "color": "#4794da"
          }
        },
        {
          "name": "axisColor",
          "category": "Colors",
          "displayName": "Axis color",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": {
            "color": "#4794da"
          }
        },
        {
          "name": "axisNameFontColor",
          "category": "Colors",
          "displayName": "Axis name font color",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": {
            "color": "#4794da"
          }
        },
        {
          "name": "id",
          "category": "General",
          "displayName": "Id",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": 1
        },
        {
          "name": "position",
          "category": "General",
          "displayName": "Position",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": "Left"
        },
        {
          "name": "mainTickMinValue",
          "category": "General",
          "displayName": "Main-tick min-value",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": 0
        },
        {
          "name": "mainTickMaxValue",
          "category": "General",
          "displayName": "Main-tick max-value",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": 100
        },
        {
          "name": "autoScaling",
          "category": "General",
          "displayName": "Auto scaling",
          "displayPriority": 10,
          "description": "",
          "defaultValue": null,
          "defaultValueInternal": false
        },
        {
          "name": "decimalPlaces",
          "category": "General",
          "displayName": "decimalPlaces",
          "displayPriority": 10,
```

```
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": 2
      },
      {
        "name": "showLabels",
        "category": "General",
        "displayName": "showLabels",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": true
      },
      {
        "name": "showAxisName",
        "category": "General",
        "displayName": "Show axis name",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": true
      },
      {
        "name": "axisName",
        "category": "General",
        "displayName": "Axis name",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": "y-axis 1"
      },
      {
        "name": "axisNameFontFamily",
        "category": "General",
        "displayName": "Axis name font-family",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": null
      },
      {
        "name": "axisNameFontSize",
        "category": "General",
        "displayName": "Axis name font-size",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": 15
      },
      {
        "name": "axisNameFontSizeUnit",
        "category": "General",
        "displayName": "Axis name font-size unit",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": "px"
      },
      {
        "name": "axisNameFontWeight",
        "category": "General",
        "displayName": "Axis name font-weight",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": "Bold"
      },
      {
        "name": "unit",
        "category": "General",
        "displayName": "Unit",
        "displayPriority": 10,
        "description": "",
        "defaultValue": null,
        "defaultValueInternal": null
      }
    ],
    "properties": {
      "labelFontColor": {
        "$ref": "tchmi:framework#/definitions/SolidColor"
```

```
        },
        "axisColor": {
          "$ref": "tchmi:framework#/definitions/SolidColor"
        },
        "axisNameFontColor": {
          "$ref": "tchmi:framework#/definitions/SolidColor"
        },
        "id": {
          "type": "number"
        },
        "position": {
          "type": "string",
          "enum": [
            "Left",
            "Right"
          ]
        },
        "mainTickMinValue": {
          "type": "number"
        },
        "mainTickMaxValue": {
          "type": "number"
        },
        "autoScaling": {
          "type": "boolean"
        },
        "decimalPlaces": {
          "type": "number",
          "minimum": 0
        },
        "showLabels": {
          "type": "boolean"
        },
        "showAxisName": {
          "type": "boolean"
        },
        "axisName": {
          "type": "string"
        },
        "axisNameFontFamily": {
          "$ref": "tchmi:framework#/definitions/FontFamily"
        },
        "axisNameFontSize": {
          "$ref": "tchmi:framework#/definitions/MeasurementValue"
        },
        "axisNameFontSizeUnit": {
          "$ref": "tchmi:framework#/definitions/FontSizeUnit"
        },
        "axisNameFontWeight": {
          "$ref": "tchmi:framework#/definitions/FontWeight"
        },
        "unit": {
          "type": "string"
        }
      },
      "required": [ "id", "position" ]
    }
  }
}
```

Note: Available since version 1.8

See also

tchmi:framework#/definitions/ChartYAxisDefinitionList [▶ 2212]

18.1.2.16 tchmi:framework#/definitions/ChartYAxisDefinitionList

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "ChartYAxisDefinitionList": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "ChartYAxisDefinitionList",
```

```
      "type": "array",
      "items": {
        "$ref": "tchmi:framework#/definitions/ChartYAxisDefinition"
      }
    }
  }
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiLineChart.YAxis [▶ 1611][]

API: TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.YAxis [▶ 1807][]

tchmi:framework#/definitions/ChartYAxisDefinition [▶ 2209]

18.1.2.17 tchmi:framework#/definitions/ClassNameList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "ClassNameList": {
            "schema": "http://json-schema.org/draft-04/schema",
            "type": "array",
            "items": {
                "type": "string"
            },
            "minItems": 0
        }
    }
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108][]

18.1.2.18 tchmi:framework#/definitions/Color

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "anyOf": [
        {
            "$schema": "http://json-schema.org/draft-04/schema",
            "id": "tchmi:framework#/definitions/SolidColor",
            "properties": {
                "color": {
                    "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                    "id": "tchmi:framework#/definitions/color",
                    "minLength": 1,
                    "type": "string"
                }
            },
            "required": [
                "color"
            ],
            "type": "object"
        },
        {
            "$schema": "http://json-schema.org/draft-04/schema",
            "id": "tchmi:framework#/definitions/LinearGradientColor",
            "properties": {
                "angle": {
                    "id": "tchmi:framework#/definitions/angle",
```

```
                    "type": "number"
                },
                "stopPoints": {
                    "id": "tchmi:framework#/definitions/stopPoints",
                    "items": {
                        "$schema": "http://json-schema.org/draft-04/schema",
                        "id": "tchmi:framework#/definitions/StopPoint",
                        "properties": {
                            "color": {
                                "description": "Defining a color as CSS color: #-hexadecimal, rgb(),
rgba(), hsl(), hsla(), lowercase css color keyword or transparent.",
                                "id": "tchmi:framework#/definitions/color",
                                "minLength": 1,
                                "type": "string"
                            },
                            "stop": {
                                "id": "tchmi:framework#/definitions/stop",
                                "pattern": "^[0-9]*\\.?[0-9]+(?:px|%)$",
                                "type": "string"
                            }
                        },
                        "required": [
                            "color"
                        ],
                        "type": "object"
                    },
                    "minItems": 2,
                    "type": "array"
                }
            },
            "required": [
                "angle",
                "stopPoints"
            ],
            "type": "object"
        }
    ],
    "id": "tchmi:framework#/definitions/Color"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Color [▶ 2084]

18.1.2.19 tchmi:framework#/definitions/ColumnOptions

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/ColumnOptions",
    "items": {
        "description": "Defines one column.",
        "properties": {
            "maxWidth": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "maxWidthUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            },
            "minWidth": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "minWidthUnit": {
```

```
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            },
            "overflow": {
                "default": false,
                "id": "tchmi:general#/definitions/Boolean",
                "type": "boolean"
            },
            "width": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "widthMode": {
                "enum": [
                    "Value",
                    "Content"
                ],
                "id": "tchmi:framework#/definitions/widthMode",
                "type": "string"
            },
            "widthUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "description": "Defines if the number defines a pixel value (px), percent value
(42%) or grow factor (2).\nThe grow factor determines how much the items will grow relative to the
rest of the items when positive free space is distributed.",
                "enum": [
                    "factor",
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnitOrFactor",
                "type": "string"
            }
        },
        "propertiesMeta": [
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": 1,
                "description": "Defines width of the column in the grid. Can be a pixel value
(42px), percent value (42%), a grow factor (1) or 'auto' (equivalent to grow factor 1).\nThe grow
factor determines how much the items will grow relative to the rest of the items when positive free
space is distributed.",
                "displayName": "Column Width",
                "displayPriority": 10,
                "name": "width"
            },
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": "factor",
                "description": "Defines if the number defines a pixel value (px), percent value
(42%) or grow factor (2).",
                "displayName": "Column Width Unit",
                "displayPriority": 10,
                "name": "widthUnit"
            },
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": "Value",
                "description": "Defines if the width is taken from the width value or calculated
from child controls in this Column.",
                "displayName": "Width Mode",
                "displayPriority": 30,
                "name": "widthMode"
            },
            {
                "category": "Layout",
                "description": "Defines maximal width of the column in the grid. Can be a pixel
value (42px), percent value (42%).\nmax-width overrides width, but min-width overrides max-width.",
                "displayName": "Column Max Width",
                "displayPriority": 60,
                "name": "maxWidth"
```

```
            },
            {
                "category": "Layout",
                "description": "",
                "displayName": "Column Max Width Unit",
                "displayPriority": 60,
                "name": "maxWidthUnit"
            },
            {
                "category": "Layout",
                "description": "Defines minimal width of the column in the grid. Can be a pixel
value (42px), percent value (42%).\nmax-width overrides width, but min-width overrides max-width.",
                "displayName": "Column Min Width",
                "displayPriority": 61,
                "name": "minWidth"
            },
            {
                "category": "Layout",
                "description": "",
                "displayName": "Column Min Width Unit",
                "displayPriority": 61,
                "name": "minWidthUnit"
            },
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": false,
                "description": "Defines if a control should be fully visible if it is outside of
this column.",
                "displayName": "Overflow",
                "displayPriority": 65,
                "name": "overflow"
            }
        ],
        "title": "Column definition",
        "type": "object"
    },
    "title": "ColumnOptions",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.System.TcHmiGrid.IColumnOptions [▶ 1206]

18.1.2.20 tchmi:framework#/definitions/ContentPath

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/ContentPath",
    "minLength": 1,
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.21 tchmi:framework#/definitions/Control

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Control",
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.System.baseTcHmiControl [▶ 1118]

18.1.2.22 tchmi:framework#/definitions/ControlAccessRight

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "description": "An access right which is checked by controls. In the Schema a defaultValue of
true mean access is granted, false is denied, null is ask parent.",
    "id": "tchmi:framework#/definitions/ControlAccessRight",
    "properties": {
        "defaultValueInternal": {
            "description": "Defines the default of this control access right. This will be used if
the check for this access right does not return a result.",
            "id": "tchmi:framework#/definitions/defaultValueInternal",
            "type": "boolean"
        },
        "description": {
            "description": "A long description of this access definition. Could be used as a
tooltip.",
            "id": "tchmi:framework#/definitions/description",
            "type": "string"
        },
        "displayName": {
            "description": "Name which should be presented the user.",
            "id": "tchmi:framework#/definitions/displayName",
            "type": "string"
        },
        "name": {
            "description": "Name of the access definition in the HTML.",
            "id": "tchmi:framework#/definitions/name",
            "minLength": 1,
            "type": "string"
        },
        "visible": {
            "description": "Determine if an access definition is shown to the user.",
            "id": "tchmi:framework#/definitions/visible",
            "type": "boolean"
        }
    },
    "required": [
        "name",
        "displayName",
        "visible",
        "description",
        "defaultValue",
        "defaultValueInternal"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.2.23 tchmi:framework#/definitions/ControlType

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/ControlType",
    "minLength": 1,
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.24 tchmi:framework#/definitions/DGColumnDefinitionList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/DGColumnDefinitionList",
    "items": {
        "properties": {
            "cellBackground": {
                "id": "tchmi:framework#/definitions/cellBackground",
                "oneOf": [
                    {
                        "$schema": "http://json-schema.org/draft-04/schema",
                        "id": "tchmi:framework#/definitions/SolidColor",
                        "properties": {
                            "color": {
                                "description": "Defining a color as CSS color: #-hexadecimal, rgb(),
rgba(), hsl(), hsla(), css color keyword, none or transparent.",
                                "id": "tchmi:framework#/definitions/color",
                                "minLength": 1,
                                "type": "string"
                            }
                        },
                        "required": [
                            "color"
                        ],
                        "type": "object"
                    },
                    {
                        "$schema": "http://json-schema.org/draft-04/schema",
                        "id": "tchmi:framework#/definitions/LinearGradientColor",
                        "properties": {
                            "angle": {
                                "id": "tchmi:framework#/definitions/angle",
                                "type": "number"
                            },
                            "stopPoints": {
                                "id": "tchmi:framework#/definitions/stopPoints",
                                "items": {
                                    "$schema": "http://json-schema.org/draft-04/schema",
                                    "id": "tchmi:framework#/definitions/StopPoint",
                                    "properties": {
                                        "color": {
                                            "description": "Defining a color as CSS color: #-
hexadecimal, rgb(), rgba(), hsl(), hsla(), lowercase css color keyword or transparent.",
                                            "id": "tchmi:framework#/definitions/color",
                                            "minLength": 1,
                                            "type": "string"
                                        },
                                        "stop": {
                                            "id": "tchmi:framework#/definitions/stop",
                                            "pattern": "^[0-9]*\\.?[0-9]+(?:px|%)$",
                                            "type": "string"
                                        }
                                    },
                                    "required": [
                                        "color"
                                    ],
                                    "type": "object"
                                },
                                "minItems": 2,
                                "type": "array"
                            }
                        },
                        "required": [
                            "angle",
                            "stopPoints"
                        ],
                        "type": "object"
                    },
                    {}
                ]
```

```
        },
        "control": {
            "enum": [
                "TextBox",
                "TextBlock",
                "CheckBox",
                "Image",
                "ComboBox"
            ],
            "id": "tchmi:framework#/definitions/control",
            "type": "string"
        },
        "dataDefinitions": {
            "id": "tchmi:framework#/definitions/DGDataDefinitionList",
            "oneOf": [
                {
                    "items": {
                        "properties": {
                            "height": {
                                "id": "tchmi:framework#/definitions/height",
                                "minimum": 0,
                                "type": "number"
                            },
                            "source": {
                                "id": "tchmi:framework#/definitions/source",
                                "type": "string"
                            },
                            "text": {
                                "id": "tchmi:framework#/definitions/text",
                                "type": "string"
                            },
                            "value": {
                                "id": "tchmi:framework#/definitions/value",
                                "type": "number"
                            },
                            "width": {
                                "id": "tchmi:framework#/definitions/width",
                                "minimum": 0,
                                "type": "number"
                            }
                        },
                        "required": [
                            "text",
                            "value",
                            "source",
                            "width",
                            "height"
                        ],
                        "type": "object"
                    },
                    "type": "array"
                },
                {
                    "items": {
                        "properties": {
                            "text": {
                                "id": "tchmi:framework#/definitions/text",
                                "type": "string"
                            },
                            "value": {
                                "id": "tchmi:framework#/definitions/value",
                                "type": "number"
                            }
                        },
                        "required": [
                            "text",
                            "value"
                        ],
                        "type": "object"
                    },
                    "type": "array"
                }
            ]
        },
        "editable": {
            "id": "tchmi:framework#/definitions/editable",
            "type": "boolean"
        },
        "format": {
            "$schema": "http://json-schema.org/draft-04/schema",
```

```
                "definitions": {
                    "symbolExpression": {
                        "id": "tchmi:framework#/definitions/symbolExpression",
                        "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                        "title": "SymbolExpression",
                        "type": "string"
                    },
                    "value": {
                        "anyOf": [
                            {
                                "properties": {
                                    "objectType": {
                                        "description": "An enum defining the variable type.",
                                        "enum": [
                                            "StaticValue"
                                        ],
                                        "id": "tchmi:framework#/definitions/objectType",
                                        "type": "string"
                                    },
                                    "value": {
                                        "anyOf": [
                                            {
                                                "type": "string"
                                            },
                                            {
                                                "type": "number"
                                            },
                                            {
                                                "type": "boolean"
                                            },
                                            {
                                                "type": "array"
                                            },
                                            {
                                                "type": "object"
                                            }
                                        ],
                                        "id": "tchmi:framework#/definitions/value"
                                    },
                                    "valueType": {
                                        "id": "tchmi:framework#/definitions/valueType",
                                        "type": "string"
                                    }
                                },
                                "required": [
                                    "objectType",
                                    "value",
                                    "valueType"
                                ],
                                "title": "StaticValue",
                                "type": "object"
                            },
                            {
                                "properties": {
                                    "objectType": {
                                        "description": "An enum defining the variable type.",
                                        "enum": [
                                            "Symbol"
                                        ],
                                        "id": "tchmi:framework#/definitions/objectType",
                                        "type": "string"
                                    },
                                    "symbolExpression": {
                                        "id": "tchmi:framework#/definitions/Function/definitions/
symbolExpression",
                                        "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                                        "title": "SymbolExpression",
                                        "type": "string"
                                    }
                                },
                                "required": [
                                    "objectType",
                                    "symbolExpression"
                                ],
                                "title": "Symbol",
                                "type": "object"
                            },
                            {
                                "properties": {
                                    "control": {
```

```
                                        "id": "tchmi:framework#/definitions/control",
                                        "type": "string"
                                    },
                                    "fn": {
                                        "id": "tchmi:framework#/definitions/fn",
                                        "type": "string"
                                    },
                                    "objectType": {
                                        "description": "An enum defining the variable type.",
                                        "enum": [
                                            "ControlApiFunction"
                                        ],
                                        "id": "tchmi:framework#/definitions/objectType",
                                        "type": "string"
                                    },
                                    "valueType": {
                                        "id": "tchmi:framework#/definitions/valueType",
                                        "type": "string"
                                    }
                                },
                                "required": [
                                    "objectType",
                                    "control",
                                    "fn",
                                    "valueType"
                                ],
                                "title": "ControlApiFunction",
                                "type": "object"
                            },
                            {
                                "properties": {
                                    "objectType": {
                                        "description": "An enum defining the variable type.",
                                        "enum": [
                                            "EventDataObject"
                                        ],
                                        "id": "tchmi:framework#/definitions/objectType",
                                        "type": "string"
                                    },
                                    "propertyPath": {
                                        "id": "tchmi:framework#/definitions/propertyPath",
                                        "type": "string"
                                    }
                                },
                                "required": [
                                    "objectType",
                                    "propertyPath"
                                ],
                                "title": "EventDataObject",
                                "type": "object"
                            }
                        ],
                        "id": "tchmi:framework#/definitions/value"
                    }
                },
                "id": "tchmi:framework#/definitions/Function",
                "properties": {
                    "fn": {
                        "id": "tchmi:framework#/definitions/fn",
                        "type": "string"
                    },
                    "fnParams": {
                        "id": "tchmi:framework#/definitions/fnParams",
                        "items": {
                            "anyOf": [
                                {
                                    "properties": {
                                        "objectType": {
                                            "description": "An enum defining the variable type.",
                                            "enum": [
                                                "StaticValue"
                                            ],
                                            "id": "tchmi:framework#/definitions/objectType",
                                            "type": "string"
                                        },
                                        "value": {
                                            "anyOf": [
                                                {
                                                    "type": "string"
                                                },
```

```
                                                    {
                                                        "type": "number"
                                                    },
                                                    {
                                                        "type": "boolean"
                                                    },
                                                    {
                                                        "type": "array"
                                                    },
                                                    {
                                                        "type": "object"
                                                    }
                                                ],
                                                "id": "tchmi:framework#/definitions/value"
                                            },
                                            "valueType": {
                                                "id": "tchmi:framework#/definitions/valueType",
                                                "type": "string"
                                            }
                                        },
                                        "required": [
                                            "objectType",
                                            "value",
                                            "valueType"
                                        ],
                                        "title": "StaticValue",
                                        "type": "object"
                                    },
                                    {
                                        "properties": {
                                            "objectType": {
                                                "description": "An enum defining the variable type.",
                                                "enum": [
                                                    "Symbol"
                                                ],
                                                "id": "tchmi:framework#/definitions/objectType",
                                                "type": "string"
                                            },
                                            "symbolExpression": {
                                                "id": "tchmi:framework#/definitions/Function/
definitions/symbolExpression",
                                                "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                                                "title": "SymbolExpression",
                                                "type": "string"
                                            }
                                        },
                                        "required": [
                                            "objectType",
                                            "symbolExpression"
                                        ],
                                        "title": "Symbol",
                                        "type": "object"
                                    },
                                    {
                                        "properties": {
                                            "control": {
                                                "id": "tchmi:framework#/definitions/control",
                                                "type": "string"
                                            },
                                            "fn": {
                                                "id": "tchmi:framework#/definitions/fn",
                                                "type": "string"
                                            },
                                            "objectType": {
                                                "description": "An enum defining the variable type.",
                                                "enum": [
                                                    "ControlApiFunction"
                                                ],
                                                "id": "tchmi:framework#/definitions/objectType",
                                                "type": "string"
                                            },
                                            "valueType": {
                                                "id": "tchmi:framework#/definitions/valueType",
                                                "type": "string"
                                            }
                                        },
                                        "required": [
                                            "objectType",
                                            "control",
                                            "fn",
```

```
                                    "valueType"
                                ],
                                "title": "ControlApiFunction",
                                "type": "object"
                            },
                            {
                                "properties": {
                                    "objectType": {
                                        "description": "An enum defining the variable type.",
                                        "enum": [
                                            "EventDataObject"
                                        ],
                                        "id": "tchmi:framework#/definitions/objectType",
                                        "type": "string"
                                    },
                                    "propertyPath": {
                                        "id": "tchmi:framework#/definitions/propertyPath",
                                        "type": "string"
                                    }
                                },
                                "required": [
                                    "objectType",
                                    "propertyPath"
                                ],
                                "title": "EventDataObject",
                                "type": "object"
                            }
                        ],
                        "id": "tchmi:framework#/definitions/Function/definitions/value"
                    },
                    "type": "array"
                },
                "objectType": {
                    "enum": [
                        "Function"
                    ],
                    "id": "tchmi:framework#/definitions/objectType",
                    "type": "string"
                }
            },
            "required": [
                "objectType",
                "fn",
                "fnParams"
            ],
            "type": "object"
        },
        "horizontalAlignment": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "Left",
                "Center",
                "Right"
            ],
            "id": "tchmi:framework#/definitions/HorizontalAlignment",
            "type": "string"
        },
        "label": {
            "id": "tchmi:framework#/definitions/label",
            "type": "string"
        },
        "minWidth": {
            "id": "tchmi:framework#/definitions/minWidth",
            "minimum": 0,
            "type": "number"
        },
        "minWidthUnit": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/MeasurementUnit",
            "type": "string"
        },
        "name": {
            "id": "tchmi:framework#/definitions/name",
            "minLength": 1,
            "type": "string"
        },
```

```
            "resize": {
                "id": "tchmi:framework#/definitions/resize",
                "type": "boolean"
            },
            "textColor": {
                "id": "tchmi:framework#/definitions/textColor",
                "oneOf": [
                    {
                        "$schema": "http://json-schema.org/draft-04/schema",
                        "id": "tchmi:framework#/definitions/SolidColor",
                        "properties": {
                            "color": {
                                "description": "Defining a color as CSS color: #-hexadecimal, rgb(),
rgba(), hsl(), hsla(), css color keyword, none or transparent.",
                                "id": "tchmi:framework#/definitions/color",
                                "minLength": 1,
                                "type": "string"
                            }
                        },
                        "required": [
                            "color"
                        ],
                        "type": "object"
                    },
                    {}
                ]
            },
            "verticalAlignment": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "Top",
                    "Center",
                    "Bottom"
                ],
                "id": "tchmi:framework#/definitions/VerticalAlignment",
                "type": "string"
            },
            "width": {
                "id": "tchmi:framework#/definitions/width",
                "minimum": 0,
                "type": "number"
            },
            "widthUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            }
        },
        "required": [
            "name",
            "label",
            "control",
            "width",
            "widthUnit",
            "resize",
            "minWidth",
            "minWidthUnit",
            "editable",
            "horizontalAlignment",
            "verticalAlignment"
        ],
        "type": "object"
    },
    "title": "DGColumnDefinitionList",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiDatagrid.Column [▶ 1427]

18.1.2.25 tchmi:framework#/definitions/DGDataDefinitionList

[Schema]

```
{
    "id": "tchmi:framework#/definitions/DGDataDefinitionList",
    "oneOf": [
        {
            "items": {
                "properties": {
                    "height": {
                        "id": "tchmi:framework#/definitions/height",
                        "minimum": 0,
                        "type": "number"
                    },
                    "source": {
                        "id": "tchmi:framework#/definitions/source",
                        "type": "string"
                    },
                    "text": {
                        "id": "tchmi:framework#/definitions/text",
                        "type": "string"
                    },
                    "value": {
                        "id": "tchmi:framework#/definitions/value",
                        "type": "number"
                    },
                    "width": {
                        "id": "tchmi:framework#/definitions/width",
                        "minimum": 0,
                        "type": "number"
                    }
                },
                "required": [
                    "text",
                    "value",
                    "source",
                    "width",
                    "height"
                ],
                "type": "object"
            },
            "type": "array"
        },
        {
            "items": {
                "properties": {
                    "text": {
                        "id": "tchmi:framework#/definitions/text",
                        "type": "string"
                    },
                    "value": {
                        "id": "tchmi:framework#/definitions/value",
                        "type": "number"
                    }
                },
                "required": [
                    "text",
                    "value"
                ],
                "type": "object"
            },
            "type": "array"
        }
    ]
}
```

Note: Available since version 1.8

18.1.2.26 tchmi:framework#/definitions/DropDownStyle

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "DropDownStyle": {
```

```
      "type": "string",
      "enum": [ "Classic", "Mobile" ]
    }
  }
}
```

Note: Available since version 1.10

18.1.2.27 tchmi:framework#/definitions/Empty

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        ""
    ],
    "id": "tchmi:framework#/definitions/Empty",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.28 tchmi:framework#/definitions/EventGridColumnList

[Schema]

```
{
   "$schema": "http://json-schema.org/draft-04/schema",
   "type": "array",
   "items": {
       "type": "object",
       "properties": {
           "columnName": {
               "type": "string",
               "enum": [
                   "severity",
                   "type",
                   "domain",
                   "name",
                   "text",
                   "timeAndDateRaised",
                   "timeRaised",
                   "dateRaised",
                   "timeAndDateConfirmed",
                   "timeConfirmed",
                   "dateConfirmed",
                   "timeAndDateCleared",
                   "timeCleared",
                   "dateCleared"
               ]
           }
       },
       "required": ["columnName"]
   }
}
```

Note: Available since version 1.10

See also

API: TcHmi.Controls.Beckhoff.EventGrid.Column [▶ 1448][]

18.1.2.29 tchmi:framework#/definitions/EventGridMenuBarPosition

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "type": "string",
    "enum": [ "Top", "Bottom", "Left", "Right" ]
}
```

Note: Available since version 1.10

See also

API: `'Top' | 'Bottom' | 'Left' | 'Right'`

18.1.2.30 tchmi:framework#/definitions/ControlKeyboardLayouts

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "description": "Enum of existing Layouts",
    "enum": [
        "EN-US_QWERTY_standard",
        "EN-US_QWERTY_full",
        "EN-US_QWERTY_compact",
        "DE-DE_QWERTZ_standard",
        "DE-DE_QWERTZ_full",
        "DE-DE_QWERTZ_compact",
        "PINpad",
        "Numpad",
        "Calculator"
    ],
    "id": "tchmi:framework#/definitions/ControlKeyboardLayouts",
    "title": "ControlKeyboardLayouts",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.31 tchmi:framework#/definitions/FillMode

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "NonZero",
        "EvenOdd"
    ],
    "id": "tchmi:framework#/definitions/FillMode",
    "title": "FillMode",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'NonZero' | 'EvenOdd'`

18.1.2.32 tchmi:framework#/definitions/FontFamily

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/FontFamily",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.33 tchmi:framework#/definitions/FontSizeUnit

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "px",
        "%"
    ],
    "id": "tchmi:framework#/definitions/FontSizeUnit",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.34 tchmi:framework#/definitions/FontStyle

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Normal",
        "Italic",
        "Oblique",
        "Auto"
    ],
    "id": "tchmi:framework#/definitions/FontStyle",
    "type": "string"
}
```

If `Auto` is specified, the parent value, for example from a central CSS file that specifies the style for the <html> element, applies.

Note: Available since version 1.8

The value "`Auto`" is available from version 1.10

See also

API: TcHmi.FontStyle [▶ 2087]

18.1.2.35 tchmi:framework#/definitions/FontWeight

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Normal",
        "Bold",
        "Auto"
    ],
    "id": "tchmi:framework#/definitions/FontWeight",
    "type": "string"
}
```

If `Auto` is specified, the parent value, for example from a central CSS file that specifies the style for the <html> element, applies.

Note: Available since version 1.8

The value "`Auto`" is available from version 1.10

See also

API: TcHmi.FontWeight [▶ 2088]

18.1.2.36 tchmi:framework#/definitions/Fraction

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Fraction",
    "maximum": 1,
    "minimum": 0,
    "type": "number"
}
```

Note: Available since version 1.8

See also

API: number [▶ 2108]

18.1.2.37 tchmi:framework#/definitions/Function

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "symbolExpression": {
            "id": "tchmi:framework#/definitions/symbolExpression",
            "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
            "title": "SymbolExpression",
            "type": "string"
        },
        "value": {
            "anyOf": [
                {
                    "properties": {
                        "objectType": {
                            "description": "An enum defining the variable type.",
                            "enum": [
                                "StaticValue"
                            ],
                            "id": "tchmi:framework#/definitions/objectType",
                            "type": "string"
                        },
                        "value": {
                            "anyOf": [
                                {
                                    "type": "string"
                                },
                                {
                                    "type": "number"
                                },
                                {
                                    "type": "boolean"
                                },
                                {
                                    "type": "array"
                                },
                                {
                                    "type": "object"
                                }
                            ],
                            "id": "tchmi:framework#/definitions/value"
                        },
                        "valueType": {
                            "id": "tchmi:framework#/definitions/valueType",
                            "type": "string"
                        }
                    },
                    "required": [
```

```
                        "objectType",
                        "value",
                        "valueType"
                    ],
                    "title": "StaticValue",
                    "type": "object"
                },
                {
                    "properties": {
                        "objectType": {
                            "description": "An enum defining the variable type.",
                            "enum": [
                                "Symbol"
                            ],
                            "id": "tchmi:framework#/definitions/objectType",
                            "type": "string"
                        },
                        "symbolExpression": {
                            "id": "tchmi:framework#/definitions/Function/definitions/
symbolExpression",
                            "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                            "title": "SymbolExpression",
                            "type": "string"
                        }
                    },
                    "required": [
                        "objectType",
                        "symbolExpression"
                    ],
                    "title": "Symbol",
                    "type": "object"
                },
                {
                    "properties": {
                        "control": {
                            "id": "tchmi:framework#/definitions/control",
                            "type": "string"
                        },
                        "fn": {
                            "id": "tchmi:framework#/definitions/fn",
                            "type": "string"
                        },
                        "objectType": {
                            "description": "An enum defining the variable type.",
                            "enum": [
                                "ControlApiFunction"
                            ],
                            "id": "tchmi:framework#/definitions/objectType",
                            "type": "string"
                        },
                        "valueType": {
                            "id": "tchmi:framework#/definitions/valueType",
                            "type": "string"
                        }
                    },
                    "required": [
                        "objectType",
                        "control",
                        "fn",
                        "valueType"
                    ],
                    "title": "ControlApiFunction",
                    "type": "object"
                },
                {
                    "properties": {
                        "objectType": {
                            "description": "An enum defining the variable type.",
                            "enum": [
                                "EventDataObject"
                            ],
                            "id": "tchmi:framework#/definitions/objectType",
                            "type": "string"
                        },
                        "propertyPath": {
                            "id": "tchmi:framework#/definitions/propertyPath",
                            "type": "string"
                        }
                    },
                    "required": [
```

```
                        "objectType",
                        "propertyPath"
                    ],
                    "title": "EventDataObject",
                    "type": "object"
                }
            ],
            "id": "tchmi:framework#/definitions/value"
        }
    },
    "id": "tchmi:framework#/definitions/Function",
    "properties": {
        "fn": {
            "id": "tchmi:framework#/definitions/fn",
            "type": "string"
        },
        "fnParams": {
            "id": "tchmi:framework#/definitions/fnParams",
            "items": {
                "anyOf": [
                    {
                        "properties": {
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "StaticValue"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "value": {
                                "anyOf": [
                                    {
                                        "type": "string"
                                    },
                                    {
                                        "type": "number"
                                    },
                                    {
                                        "type": "boolean"
                                    },
                                    {
                                        "type": "array"
                                    },
                                    {
                                        "type": "object"
                                    }
                                ],
                                "id": "tchmi:framework#/definitions/value"
                            },
                            "valueType": {
                                "id": "tchmi:framework#/definitions/valueType",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "value",
                            "valueType"
                        ],
                        "title": "StaticValue",
                        "type": "object"
                    },
                    {
                        "properties": {
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "Symbol"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "symbolExpression": {
                                "id": "tchmi:framework#/definitions/Function/definitions/
symbolExpression",
                                "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                                "title": "SymbolExpression",
                                "type": "string"
                            }
```

```
                        },
                        "required": [
                            "objectType",
                            "symbolExpression"
                        ],
                        "title": "Symbol",
                        "type": "object"
                    },
                    {
                        "properties": {
                            "control": {
                                "id": "tchmi:framework#/definitions/control",
                                "type": "string"
                            },
                            "fn": {
                                "id": "tchmi:framework#/definitions/fn",
                                "type": "string"
                            },
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "ControlApiFunction"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "valueType": {
                                "id": "tchmi:framework#/definitions/valueType",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "control",
                            "fn",
                            "valueType"
                        ],
                        "title": "ControlApiFunction",
                        "type": "object"
                    },
                    {
                        "properties": {
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "EventDataObject"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "propertyPath": {
                                "id": "tchmi:framework#/definitions/propertyPath",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "propertyPath"
                        ],
                        "title": "EventDataObject",
                        "type": "object"
                    }
                ],
                "id": "tchmi:framework#/definitions/Function/definitions/value"
            },
            "type": "array"
        },
        "objectType": {
            "enum": [
                "Function"
            ],
            "id": "tchmi:framework#/definitions/objectType",
            "type": "string"
        }
    },
    "required": [
        "objectType",
        "fn",
        "fnParams"
```

```
    ],
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.IFunction [▶ 2095]

18.1.2.38 tchmi:framework#/definitions/FunctionList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/FunctionList",
    "items": {
        "$schema": "http://json-schema.org/draft-04/schema",
        "definitions": {
            "symbolExpression": {
                "id": "tchmi:framework#/definitions/symbolExpression",
                "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                "title": "SymbolExpression",
                "type": "string"
            },
            "value": {
                "anyOf": [
                    {
                        "properties": {
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "StaticValue"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "value": {
                                "anyOf": [
                                    {
                                        "type": "string"
                                    },
                                    {
                                        "type": "number"
                                    },
                                    {
                                        "type": "boolean"
                                    },
                                    {
                                        "type": "array"
                                    },
                                    {
                                        "type": "object"
                                    }
                                ],
                                "id": "tchmi:framework#/definitions/value"
                            },
                            "valueType": {
                                "id": "tchmi:framework#/definitions/valueType",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "value",
                            "valueType"
                        ],
                        "title": "StaticValue",
                        "type": "object"
                    },
                    {
                        "properties": {
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
```

```
                                    "Symbol"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "symbolExpression": {
                                "id": "tchmi:framework#/definitions/Function/definitions/
symbolExpression",
                                "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                                "title": "SymbolExpression",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "symbolExpression"
                        ],
                        "title": "Symbol",
                        "type": "object"
                    },
                    {
                        "properties": {
                            "control": {
                                "id": "tchmi:framework#/definitions/control",
                                "type": "string"
                            },
                            "fn": {
                                "id": "tchmi:framework#/definitions/fn",
                                "type": "string"
                            },
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "ControlApiFunction"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "valueType": {
                                "id": "tchmi:framework#/definitions/valueType",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "control",
                            "fn",
                            "valueType"
                        ],
                        "title": "ControlApiFunction",
                        "type": "object"
                    },
                    {
                        "properties": {
                            "objectType": {
                                "description": "An enum defining the variable type.",
                                "enum": [
                                    "EventDataObject"
                                ],
                                "id": "tchmi:framework#/definitions/objectType",
                                "type": "string"
                            },
                            "propertyPath": {
                                "id": "tchmi:framework#/definitions/propertyPath",
                                "type": "string"
                            }
                        },
                        "required": [
                            "objectType",
                            "propertyPath"
                        ],
                        "title": "EventDataObject",
                        "type": "object"
                    }
                ],
                "id": "tchmi:framework#/definitions/value"
            }
        },
        "id": "tchmi:framework#/definitions/Function",
        "properties": {
```

```
            "fn": {
                "id": "tchmi:framework#/definitions/fn",
                "type": "string"
            },
            "fnParams": {
                "id": "tchmi:framework#/definitions/fnParams",
                "items": {
                    "anyOf": [
                        {
                            "properties": {
                                "objectType": {
                                    "description": "An enum defining the variable type.",
                                    "enum": [
                                        "StaticValue"
                                    ],
                                    "id": "tchmi:framework#/definitions/objectType",
                                    "type": "string"
                                },
                                "value": {
                                    "anyOf": [
                                        {
                                            "type": "string"
                                        },
                                        {
                                            "type": "number"
                                        },
                                        {
                                            "type": "boolean"
                                        },
                                        {
                                            "type": "array"
                                        },
                                        {
                                            "type": "object"
                                        }
                                    ],
                                    "id": "tchmi:framework#/definitions/value"
                                },
                                "valueType": {
                                    "id": "tchmi:framework#/definitions/valueType",
                                    "type": "string"
                                }
                            },
                            "required": [
                                "objectType",
                                "value",
                                "valueType"
                            ],
                            "title": "StaticValue",
                            "type": "object"
                        },
                        {
                            "properties": {
                                "objectType": {
                                    "description": "An enum defining the variable type.",
                                    "enum": [
                                        "Symbol"
                                    ],
                                    "id": "tchmi:framework#/definitions/objectType",
                                    "type": "string"
                                },
                                "symbolExpression": {
                                    "id": "tchmi:framework#/definitions/Function/definitions/
symbolExpression",
                                    "pattern": "^%(s|i|l|pp|tp|f|ctrl|ca)%.+%\\1%$",
                                    "title": "SymbolExpression",
                                    "type": "string"
                                }
                            },
                            "required": [
                                "objectType",
                                "symbolExpression"
                            ],
                            "title": "Symbol",
                            "type": "object"
                        },
                        {
                            "properties": {
                                "control": {
                                    "id": "tchmi:framework#/definitions/control",
```

```
                                    "type": "string"
                                },
                                "fn": {
                                    "id": "tchmi:framework#/definitions/fn",
                                    "type": "string"
                                },
                                "objectType": {
                                    "description": "An enum defining the variable type.",
                                    "enum": [
                                        "ControlApiFunction"
                                    ],
                                    "id": "tchmi:framework#/definitions/objectType",
                                    "type": "string"
                                },
                                "valueType": {
                                    "id": "tchmi:framework#/definitions/valueType",
                                    "type": "string"
                                }
                            },
                            "required": [
                                "objectType",
                                "control",
                                "fn",
                                "valueType"
                            ],
                            "title": "ControlApiFunction",
                            "type": "object"
                        },
                        {
                            "properties": {
                                "objectType": {
                                    "description": "An enum defining the variable type.",
                                    "enum": [
                                        "EventDataObject"
                                    ],
                                    "id": "tchmi:framework#/definitions/objectType",
                                    "type": "string"
                                },
                                "propertyPath": {
                                    "id": "tchmi:framework#/definitions/propertyPath",
                                    "type": "string"
                                }
                            },
                            "required": [
                                "objectType",
                                "propertyPath"
                            ],
                            "title": "EventDataObject",
                            "type": "object"
                        }
                    ],
                    "id": "tchmi:framework#/definitions/Function/definitions/value"
                },
                "type": "array"
            },
            "objectType": {
                "enum": [
                    "Function"
                ],
                "id": "tchmi:framework#/definitions/objectType",
                "type": "string"
            }
        },
        "required": [
            "objectType",
            "fn",
            "fnParams"
        ],
        "type": "object"
    },
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.IFunction [▶ 2095]

18.1.2.39 tchmi:framework#/definitions/GaugeRangeDefinitionList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/GaugeRangeDefinitionList",
    "items": {
        "properties": {
            "color": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/SolidColor",
                "properties": {
                    "color": {
                        "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(),
hsl(), hsla(), css color keyword, none or transparent.",
                        "id": "tchmi:framework#/definitions/color",
                        "minLength": 1,
                        "type": "string"
                    }
                },
                "required": [
                    "color"
                ],
                "type": "object"
            },
            "end": {
                "id": "tchmi:framework#/definitions/end",
                "type": "number"
            },
            "start": {
                "id": "tchmi:framework#/definitions/start",
                "type": "number"
            }
        },
        "required": [
            "color",
            "start",
            "end"
        ],
        "type": "object"
    },
    "title": "GaugeRangeDefinitionList",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiLinearGauge.Range [▶ 1556]

API: TcHmi.Controls.Beckhoff.TcHmiRadialGauge.Range [▶ 1681]

18.1.2.40 tchmi:framework#/definitions/GaugeTickDefinition

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/GaugeTickDefinition",
    "properties": {
        "mainTickRange": {
            "id": "tchmi:framework#/definitions/mainTickRange",
            "minimum": 0,
            "type": "number"
        },
        "subTickRange": {
            "id": "tchmi:framework#/definitions/subTickRange",
            "minimum": 0,
            "type": "number"
        }
    },
    "required": [
        "mainTickRange",
```

```
        "subTickRange"
    ],
    "title": "GaugeTickDefinition",
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiLinearGauge.TickDefinition [▶ 1556]

API: TcHmi.Controls.Beckhoff.TcHmiRadialGauge.TickDefinition [▶ 1681]

18.1.2.41 tchmi:framework#/definitions/HorizontalAlignment

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Left",
        "Center",
        "Right"
    ],
    "id": "tchmi:framework#/definitions/HorizontalAlignment",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.HorizontalAlignment [▶ 2089]

18.1.2.42 tchmi:framework#/definitions/Id

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Id",
    "minLength": 1,
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.43 tchmi:framework#/definitions/KeyList

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "KeyList": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "KeyList",
      "description": "Array of keys.",
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "code": { "type": "string" },
          "key": { "type": "string" },
```

```
          "unmodifiedKey": { "type": "string" },
          "location": { "type": "string" }
        },
        "required": ["code", "key", "unmodifiedKey", "location"]
      }
    }
  }
}
```

Note: Available since version 1.8

18.1.2.44 tchmi:framework#/definitions/KnobDefinition

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "KnobDefinition": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "title": "KnobDefinition",
            "type": "object",
            "properties": {
                "imagePath": {
                    "$ref": "tchmi:framework#/definitions/Path"
                },
                "width": {
                    "type": "number"
                },
                "height": {
                    "type": "number"
                },
                "offsetX": {
                    "type": "number"
                },
                "offsetY": {
                    "type": "number"
                }
            }
        }
    }
}
```

Note: Available since version 1.10

18.1.2.45 tchmi:framework#/definitions/LinearGaugeLabelPosition

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Top",
        "Bottom",
        "Left",
        "Right"
    ],
    "id": "tchmi:framework#/definitions/LinearGaugeLabelPosition",
    "title": "LinearGaugeLabelPosition",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'Top' | 'Bottom' | 'Left' | 'Right'`

18.1.2.46 tchmi:framework#/definitions/LinearGaugeOrientation

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Horizontal",
        "Vertical"
    ],
    "id": "tchmi:framework#/definitions/LinearGaugeOrientation",
    "title": "LinearGaugeOrientation",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'Horizontal' | 'Vertical'`

18.1.2.47 tchmi:framework#/definitions/LinearGaugeUnitOrientation

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Theme",
        "Horizontal",
        "Vertical"
    ],
    "id": "tchmi:framework#/definitions/LinearGaugeUnitOrientation",
    "title": "LinearGaugeUnitOrientation",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'Theme' | 'Horizontal' | 'Vertical'`

18.1.2.48 tchmi:framework#/definitions/LinearGradientColor

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "LinearGradientColor": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "type": "object",
            "properties": {
                "angle": { "type": "number" },
                "stopPoints": {
                    "type": "array",
                    "items": {
                        "title": "StopPoint definition",
                        "$ref": "tchmi:framework#/definitions/StopPoint"
                    },
                    "minItems": 2
                }
            },
            "required": [ "angle", "stopPoints" ]
        }
    }
}
```

Note: Available since version 1.8

See also

API: TcHmi.LinearGradientColor [▶ 2085]

18.1.2.49 tchmi:framework#/definitions/ListItem

[Schema]

```
{
    "description": "List Item List",
    "id": "tchmi:framework#/definitions/ListItem",
    "properties": {
        "index": {
            "id": "tchmi:framework#/definitions/index",
            "type": "number"
        },
        "text": {
            "anyOf": [
                {
                    "type": "string"
                },
                {
                    "type": "number"
                },
                {
                    "type": "boolean"
                }
            ],
            "id": "tchmi:framework#/definitions/text"
        },
        "value": {
            "id": "tchmi:framework#/definitions/value"
        }
    },
    "required": [
        "text"
    ],
    "title": "List Item",
    "type": "object"
}
```

Note: Available since version 1.8

18.1.2.50 tchmi:framework#/definitions/ListItemList

[Schema]

```
{
    "anyOf": [
        {
            "items": {
                "description": "List Item List",
                "id": "tchmi:framework#/definitions/ListItem",
                "properties": {
                    "index": {
                        "id": "tchmi:framework#/definitions/index",
                        "type": "number"
                    },
                    "text": {
                        "anyOf": [
                            {
                                "type": "string"
                            },
                            {
                                "type": "number"
                            },
                            {
                                "type": "boolean"
                            }
                        ],
                        "id": "tchmi:framework#/definitions/text"
                    },
                    "value": {
                        "id": "tchmi:framework#/definitions/value"
                    }
                },
                "required": [
                    "text"
                ],
                "title": "List Item",
                "type": "object"
```

```
            },
            "type": "array"
        },
        {
            "type": "array"
        },
        {
            "type": "object"
        }
    ],
    "description": "List Item List",
    "id": "tchmi:framework#/definitions/ListItemList",
    "title": "ListItemList"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiCombobox.ListItem [▶ 1379]

18.1.2.51 tchmi:framework#/definitions/Locale

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Locale",
    "type": "string"
}
```

Definition of the languages in the current project

Note: Available since version 1.8

From version 1.10.1336.10 replaced by tchmi:general#/definitions/Locale [▶ 2182]

18.1.2.52 tchmi:framework#/definitions/MeasurementUnit

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "px",
        "%"
    ],
    "id": "tchmi:framework#/definitions/MeasurementUnit",
    "type": "string"
}
```

Note: Available since version 1.8

This data type specifies a unit of measurement.

See also

API: TcHmi.DimensionUnit [▶ 2086]

18.1.2.53 tchmi:framework#/definitions/MeasurementUnitOrFactor

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "description": "Defines if the number defines a pixel value (px), percent value (42%) or grow
factor (2).\nThe grow factor determines how much the items will grow relative to the rest of the
items when positive free space is distributed.",
    "enum": [
```

```
        "factor",
        "px",
        "%"
    ],
    "id": "tchmi:framework#/definitions/MeasurementUnitOrFactor",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.54 tchmi:framework#/definitions/MeasurementValue

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/MeasurementValue",
    "type": "number"
}
```

Note: Available since version 1.8

See also

API: number [▶ 2108]

18.1.2.55 tchmi:framework#/definitions/Message

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Message",
    "properties": {
        "description": {
            "id": "tchmi:framework#/definitions/description",
            "type": "string"
        },
        "rawData": {
            "id": "tchmi:framework#/definitions/rawData",
            "type": "object"
        },
        "severity": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "Verbose",
                "Info",
                "Warning",
                "Error",
                "Critical"
            ],
            "id": "tchmi:framework#/definitions/MessageSeverity",
            "type": "string"
        },
        "text": {
            "id": "tchmi:framework#/definitions/text",
            "type": "string"
        },
        "timeRaised": {
            "format": "datetime",
            "id": "tchmi:framework#/definitions/timeRaised",
            "type": "string"
        },
        "type": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "MESSAGE",
                "ALARM"
            ],
            "id": "tchmi:framework#/definitions/MessageType",
            "type": "string"
        }
    },
    "required": [
```

```
        "type",
        "text",
        "severity",
        "timeRaised",
        "description"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.2.56 tchmi:framework#/definitions/MessageOrAlarm

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/MessageOrAlarm",
    "oneOf": [
        {
            "$schema": "http://json-schema.org/draft-04/schema",
            "id": "tchmi:framework#/definitions/Alarm",
            "properties": {
                "confirmationState": {
                    "id": "tchmi:framework#/definitions/confirmationState",
                    "type": "boolean"
                },
                "description": {
                    "id": "tchmi:framework#/definitions/description",
                    "type": "string"
                },
                "id": {
                    "id": "tchmi:framework#/definitions/id",
                    "type": "number"
                },
                "rawData": {
                    "id": "tchmi:framework#/definitions/rawData",
                    "type": "object"
                },
                "severity": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "Verbose",
                        "Info",
                        "Warning",
                        "Error",
                        "Critical"
                    ],
                    "id": "tchmi:framework#/definitions/MessageSeverity",
                    "type": "string"
                },
                "text": {
                    "id": "tchmi:framework#/definitions/text",
                    "type": "string"
                },
                "timeCleared": {
                    "format": "datetime",
                    "id": "tchmi:framework#/definitions/timeCleared",
                    "type": "string"
                },
                "timeConfirmed": {
                    "format": "datetime",
                    "id": "tchmi:framework#/definitions/timeConfirmed",
                    "type": "string"
                },
                "timeRaised": {
                    "format": "datetime",
                    "id": "tchmi:framework#/definitions/timeRaised",
                    "type": "string"
                },
                "type": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "MESSAGE",
                        "ALARM"
                    ],
                    "id": "tchmi:framework#/definitions/MessageType",
                    "type": "string"
```

```
                }
            },
            "required": [
                "type",
                "text",
                "severity",
                "timeRaised",
                "description",
                "id"
            ],
            "type": "object"
        },
        {
            "$schema": "http://json-schema.org/draft-04/schema",
            "id": "tchmi:framework#/definitions/Message",
            "properties": {
                "description": {
                    "id": "tchmi:framework#/definitions/description",
                    "type": "string"
                },
                "rawData": {
                    "id": "tchmi:framework#/definitions/rawData",
                    "type": "object"
                },
                "severity": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "Verbose",
                        "Info",
                        "Warning",
                        "Error",
                        "Critical"
                    ],
                    "id": "tchmi:framework#/definitions/MessageSeverity",
                    "type": "string"
                },
                "text": {
                    "id": "tchmi:framework#/definitions/text",
                    "type": "string"
                },
                "timeRaised": {
                    "format": "datetime",
                    "id": "tchmi:framework#/definitions/timeRaised",
                    "type": "string"
                },
                "type": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "MESSAGE",
                        "ALARM"
                    ],
                    "id": "tchmi:framework#/definitions/MessageType",
                    "type": "string"
                }
            },
            "required": [
                "type",
                "text",
                "severity",
                "timeRaised",
                "description"
            ],
            "type": "object"
        }
    ]
}
```

Note: Available since version 1.8

18.1.2.57 tchmi:framework#/definitions/MessageSeverity

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Verbose",
        "Info",
```

```
        "Warning",
        "Error",
        "Critical"
    ],
    "id": "tchmi:framework#/definitions/MessageSeverity",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.58 tchmi:framework#/definitions/MessageSortAttribute

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/MessageSortAttribute",
    "properties": {
        "propertyName": {
            "enum": [
                "type",
                "text",
                "severity",
                "timeRaised",
                "description",
                "id",
                "timeCleared",
                "timeConfirmed",
                "confirmationState"
            ],
            "id": "tchmi:framework#/definitions/propertyName",
            "type": "string"
        },
        "sortOrder": {
            "$schema": "http://json-schema.org/draft-04/schema",
            "enum": [
                "Ascending ",
                "Descending "
            ],
            "id": "tchmi:framework#/definitions/SortOrder",
            "title": "SortOrder",
            "type": "string"
        }
    },
    "title": "MessageSortAttribute",
    "type": "object"
}
```

Note: Available since version 1.8

18.1.2.59 tchmi:framework#/definitions/MessageType

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "MESSAGE",
        "ALARM"
    ],
    "id": "tchmi:framework#/definitions/MessageType",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.60 tchmi:framework#/definitions/Padding

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
```

```
        "unit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/unit",
            "type": "string"
        },
        "value": {
            "id": "tchmi:framework#/definitions/value",
            "minimum": 0,
            "type": "number"
        }
    },
    "id": "tchmi:framework#/definitions/Padding",
    "properties": {
        "bottom": {
            "id": "tchmi:framework#/definitions/Padding/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "bottomUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/Padding/definitions/unit",
            "type": "string"
        },
        "left": {
            "id": "tchmi:framework#/definitions/Padding/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "leftUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/Padding/definitions/unit",
            "type": "string"
        },
        "right": {
            "id": "tchmi:framework#/definitions/Padding/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "rightUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/Padding/definitions/unit",
            "type": "string"
        },
        "top": {
            "id": "tchmi:framework#/definitions/Padding/definitions/value",
            "minimum": 0,
            "type": "number"
        },
        "topUnit": {
            "enum": [
                "px",
                "%"
            ],
            "id": "tchmi:framework#/definitions/Padding/definitions/unit",
            "type": "string"
        }
    },
    "required": [
        "left",
        "top",
        "right",
        "bottom"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.FourSidedCss [▶ 2088]

18.1.2.61 tchmi:framework#/definitions/Path

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Path",
    "minLength": 1,
    "type": "string"
}
```

This data type exists so that the development environment [▶ 42] can offer a customized file selection.

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.62 tchmi:framework#/definitions/PixelUnit

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "px"
    ],
    "id": "tchmi:framework#/definitions/PixelUnit",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'px'`

18.1.2.63 tchmi:framework#/definitions/PositiveNumber

[Schema]

```
{
    "default": 0,
    "id": "tchmi:framework#/definitions/PositiveNumber",
    "minimum": 0,
    "type": "number"
}
```

Note: Available since version 1.8

See also

API: number [▶ 2108]

18.1.2.64 tchmi:framework#/definitions/RadialGaugeLabelPosition

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
```

```
        "Outside",
        "Inside"
    ],
    "id": "tchmi:framework#/definitions/RadialGaugeLabelPosition",
    "title": "RadialGaugeLabelPosition",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'Top' | 'Bottom' | 'Left' | 'Right'`

18.1.2.65 tchmi:framework#/definitions/RecipeReference

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Path",
    "minLength": 1,
    "type": "string"
}
```

Note: Available since version 1.10.1171.21

See also

API: string

Also see about this

string [2108]

18.1.2.66 tchmi:framework#/definitions/RecipeTypeReference

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/Path",
    "minLength": 1,
    "type": "string"
}
```

Note: Available since version 1.10.1171.21

See also

API: string

Also see about this

string [2108]

18.1.2.67 tchmi:framework#/definitions/RowOptions

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/RowOptions",
    "items": {
        "description": "Defines one row.",
        "properties": {
            "height": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
```

```
                "type": "number"
            },
            "heightMode": {
                "enum": [
                    "Value",
                    "Content"
                ],
                "id": "tchmi:framework#/definitions/heightMode",
                "type": "string"
            },
            "heightUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "description": "Defines if the number defines a pixel value (px), percent value
(42%) or grow factor (2).\nThe grow factor determines how much the items will grow relative to the
rest of the items when positive free space is distributed.",
                "enum": [
                    "factor",
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnitOrFactor",
                "type": "string"
            },
            "maxHeight": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "maxHeightUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            },
            "minHeight": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/MeasurementValue",
                "type": "number"
            },
            "minHeightUnit": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "enum": [
                    "px",
                    "%"
                ],
                "id": "tchmi:framework#/definitions/MeasurementUnit",
                "type": "string"
            },
            "overflow": {
                "default": false,
                "id": "tchmi:general#/definitions/Boolean",
                "type": "boolean"
            }
        },
        "propertiesMeta": [
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": 1,
                "description": "Defines height of the cells of this row in the grid. Can be a pixel
value (42px), percent value (42%), a grow factor (1) or 'auto' (equivalent to grow factor 1).\nThe
grow factor determines how much the items will grow relative to the rest of the items when positive
free space is distributed.",
                "displayName": "Row Height",
                "displayPriority": 10,
                "name": "height"
            },
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": "factor",
                "description": "Defines if the number defines a pixel value (px), percent value
(42%) or grow factor (2).",
                "displayName": "Row Height Unit",
                "displayPriority": 10,
                "name": "heightUnit"
            },
```

```
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": "Value",
                "description": "Defines if the height is taken from the height value or calculated
from child controls in row.",
                "displayName": "Height Mode",
                "displayPriority": 30,
                "name": "heightMode"
            },
            {
                "category": "Layout",
                "description": "Defines maximal height of the row in the grid. Can be a pixel value
(42px), percent value (42%).\nmax-height overrides height, but min-height overrides max-height.",
                "displayName": "Row Max Height",
                "displayPriority": 60,
                "name": "maxHeight"
            },
            {
                "category": "Layout",
                "description": "",
                "displayName": "Row Max Height Unit",
                "displayPriority": 60,
                "name": "maxHeightUnit"
            },
            {
                "category": "Layout",
                "description": "Defines minimal height of the row in the grid. Can be a pixel value
(42px), percent value (42%).\nmax-height overrides height, but min-height overrides max-height.",
                "displayName": "Row Min Height",
                "displayPriority": 61,
                "name": "minHeight"
            },
            {
                "category": "Layout",
                "description": "",
                "displayName": "Row Min Height Unit",
                "displayPriority": 61,
                "name": "minHeightUnit"
            },
            {
                "category": "Layout",
                "defaultValue": null,
                "defaultValueInternal": false,
                "description": "Defines if a control should be fully visible if it is outside of
this row.",
                "displayName": "Overflow",
                "displayPriority": 65,
                "name": "overflow"
            }
        ],
        "title": "Row definition",
        "type": "object"
    },
    "title": "RowOptions",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.System.TcHmiGrid.IRowOptions [▶ 1207]

18.1.2.68 tchmi:framework#/definitions/ScaleMode

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "None",
        "ScaleToFill",
        "ScaleToFit",
        "ScaleToFitWidth",
        "ScaleToFitHeight"
```

```
    ],
    "id": "tchmi:framework#/definitions/ScaleMode",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.ScaleModeString [▶ 2096]

18.1.2.69 tchmi:framework#/definitions/ScrollMode

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "No",
        "Yes",
        "Auto"
    ],
    "id": "tchmi:framework#/definitions/ScrollMode",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: `'No' | 'Yes' | 'Auto'`

18.1.2.70 tchmi:framework#/definitions/SizeMode

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Value",
        "Parent"
    ],
    "id": "tchmi:framework#/definitions/SizeMode",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.SizeMode [▶ 2096]

18.1.2.71 tchmi:framework#/definitions/SizeModeWithContent

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Value",
        "Parent",
        "Content"
    ],
    "id": "tchmi:framework#/definitions/SizeModeWithContent",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.SizeModeWithContent [▶ 2096]

18.1.2.72 tchmi:framework#/definitions/SolidColor

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/SolidColor",
    "properties": {
        "color": {
            "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(), hsl(),
hsla(), css color keyword, none or transparent.",
            "id": "tchmi:framework#/definitions/color",
            "minLength": 1,
            "type": "string"
        }
    },
    "required": [
        "color"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.SolidColor [▶ 2084]

18.1.2.73 tchmi:framework#/definitions/SortOrder

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Ascending ",
        "Descending "
    ],
    "id": "tchmi:framework#/definitions/SortOrder",
    "title": "SortOrder",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.74 tchmi:framework#/definitions/StopPoint

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/StopPoint",
    "properties": {
        "color": {
            "description": "Defining a color as CSS color: #-hexadecimal, rgb(), rgba(), hsl(),
hsla(), lowercase css color keyword or transparent.",
            "id": "tchmi:framework#/definitions/color",
            "minLength": 1,
            "type": "string"
        },
        "stop": {
            "id": "tchmi:framework#/definitions/stop",
            "pattern": "^[0-9]*\\.?[0-9]+(?:px|%)$",
            "type": "string"
        }
    },
    "required": [
```

```
        "color"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.2.75 tchmi:framework#/definitions/TcHmiButtonStateSymbol

[Schema]

```
{
    "allOf": [
        {
            "$ref": "tchmi:framework#/definitions/Symbol"
        },
        {
            "frameworkSymbolSubType":
            {
                "$ref": "tchmi:general#/definitions/Boolean"
            }
        }
    ]
}
```

Note: Available since version 1.10

18.1.2.76 tchmi:framework#/definitions/TcHmiCheckboxStateSymbol

[Schema]

```
{
    "allOf": [
        {
            "$ref": "tchmi:framework#/definitions/Symbol"
        },
        {
            "frameworkSymbolSubType":
            {
                "$ref": "tchmi:general#/definitions/Boolean"
            }
        }
    ]
}
```

Note: Available since version 1.10

18.1.2.77 tchmi:framework#/definitions/ToggleState

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Normal",
        "Active"
    ],
    "id": "tchmi:framework#/definitions/ToggleState",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.ToggleState [▶ 2097]

18.1.2.78 tchmi:framework#/definitions/TransformList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "definitions": {
        "angleUnit": {
            "enum": [
                "deg",
                "rad",
                "turn",
                "grad"
            ],
            "id": "tchmi:framework#/definitions/angleUnit",
            "type": "string"
        },
        "transform": {
            "id": "tchmi:framework#/definitions/transform",
            "oneOf": [
                {
                    "properties": {
                        "transformType": {
                            "enum": [
                                "Translate"
                            ],
                            "id": "tchmi:framework#/definitions/transformType",
                            "type": "string"
                        },
                        "x": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "xUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        },
                        "y": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "yUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        },
                        "z": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "zUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        }
                    },
                    "required": [
                        "transformType"
                    ],
                    "title": "Translate",
                    "type": "object"
                },
                {
```

```
                    "properties": {
                        "angle": {
                            "id": "tchmi:framework#/definitions/angle",
                            "type": "number"
                        },
                        "angleUnit": {
                            "enum": [
                                "deg",
                                "rad",
                                "turn",
                                "grad"
                            ],
                            "id": "tchmi:framework#/definitions/TransformList/definitions/
angleUnit",
                            "type": "string"
                        },
                        "transformType": {
                            "enum": [
                                "Rotate"
                            ],
                            "id": "tchmi:framework#/definitions/transformType",
                            "type": "string"
                        },
                        "vector": {
                            "id": "tchmi:framework#/definitions/vector",
                            "properties": {
                                "x": {
                                    "id": "tchmi:framework#/definitions/x",
                                    "type": "number"
                                },
                                "y": {
                                    "id": "tchmi:framework#/definitions/y",
                                    "type": "number"
                                },
                                "z": {
                                    "id": "tchmi:framework#/definitions/z",
                                    "type": "number"
                                }
                            },
                            "required": [
                                "x",
                                "y",
                                "z"
                            ],
                            "type": "object"
                        }
                    },
                    "required": [
                        "transformType",
                        "angle"
                    ],
                    "title": "Rotate",
                    "type": "object"
                },
                {
                    "properties": {
                        "transformType": {
                            "enum": [
                                "Scale"
                            ],
                            "id": "tchmi:framework#/definitions/transformType",
                            "type": "string"
                        },
                        "x": {
                            "id": "tchmi:framework#/definitions/x",
                            "type": "number"
                        },
                        "y": {
                            "id": "tchmi:framework#/definitions/y",
                            "type": "number"
                        },
                        "z": {
                            "id": "tchmi:framework#/definitions/z",
                            "type": "number"
                        }
                    },
                    "required": [
                        "transformType"
                    ],
                    "title": "Scale",
```

```
                    "type": "object"
                },
                {
                    "properties": {
                        "transformType": {
                            "enum": [
                                "Skew"
                            ],
                            "id": "tchmi:framework#/definitions/transformType",
                            "type": "string"
                        },
                        "xAngle": {
                            "id": "tchmi:framework#/definitions/xAngle",
                            "type": "number"
                        },
                        "xAngleUnit": {
                            "enum": [
                                "deg",
                                "rad",
                                "turn",
                                "grad"
                            ],
                            "id": "tchmi:framework#/definitions/TransformList/definitions/
angleUnit",
                            "type": "string"
                        },
                        "yAngle": {
                            "id": "tchmi:framework#/definitions/yAngle",
                            "type": "number"
                        },
                        "yAngleUnit": {
                            "enum": [
                                "deg",
                                "rad",
                                "turn",
                                "grad"
                            ],
                            "id": "tchmi:framework#/definitions/TransformList/definitions/
angleUnit",
                            "type": "string"
                        }
                    },
                    "required": [
                        "transformType"
                    ],
                    "title": "Skew",
                    "type": "object"
                },
                {
                    "properties": {
                        "transformType": {
                            "enum": [
                                "Origin"
                            ],
                            "id": "tchmi:framework#/definitions/transformType",
                            "type": "string"
                        },
                        "x": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "xUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        },
                        "y": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "yUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
```

```
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "z": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "zUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                }
            },
            "required": [
                "transformType"
            ],
            "title": "Origin",
            "type": "object"
        },
        {
            "properties": {
                "distance": {
                    "id": "tchmi:framework#/definitions/distance",
                    "type": "number"
                },
                "distanceUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "origin": {
                    "id": "tchmi:framework#/definitions/origin",
                    "properties": {
                        "x": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "xUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        },
                        "y": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "yUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        }
                    },
                    "type": "object"
                },
                "transformType": {
                    "enum": [
                        "Perspective"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
```

```
                        "type": "string"
                    }
                },
                "required": [
                    "transformType",
                    "distance"
                ],
                "title": "Perspective",
                "type": "object"
            }
        ]
    }
},
"id": "tchmi:framework#/definitions/TransformList",
"items": {
    "id": "tchmi:framework#/definitions/TransformList/definitions/transform",
    "oneOf": [
        {
            "properties": {
                "transformType": {
                    "enum": [
                        "Translate"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
                    "type": "string"
                },
                "x": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "xUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "y": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "yUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "z": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "zUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                }
            },
            "required": [
                "transformType"
            ],
            "title": "Translate",
            "type": "object"
        },
        {
            "properties": {
                "angle": {
                    "id": "tchmi:framework#/definitions/angle",
                    "type": "number"
```

```
                },
                "angleUnit": {
                    "enum": [
                        "deg",
                        "rad",
                        "turn",
                        "grad"
                    ],
                    "id": "tchmi:framework#/definitions/TransformList/definitions/angleUnit",
                    "type": "string"
                },
                "transformType": {
                    "enum": [
                        "Rotate"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
                    "type": "string"
                },
                "vector": {
                    "id": "tchmi:framework#/definitions/vector",
                    "properties": {
                        "x": {
                            "id": "tchmi:framework#/definitions/x",
                            "type": "number"
                        },
                        "y": {
                            "id": "tchmi:framework#/definitions/y",
                            "type": "number"
                        },
                        "z": {
                            "id": "tchmi:framework#/definitions/z",
                            "type": "number"
                        }
                    },
                    "required": [
                        "x",
                        "y",
                        "z"
                    ],
                    "type": "object"
                }
            },
            "required": [
                "transformType",
                "angle"
            ],
            "title": "Rotate",
            "type": "object"
        },
        {
            "properties": {
                "transformType": {
                    "enum": [
                        "Scale"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
                    "type": "string"
                },
                "x": {
                    "id": "tchmi:framework#/definitions/x",
                    "type": "number"
                },
                "y": {
                    "id": "tchmi:framework#/definitions/y",
                    "type": "number"
                },
                "z": {
                    "id": "tchmi:framework#/definitions/z",
                    "type": "number"
                }
            },
            "required": [
                "transformType"
            ],
            "title": "Scale",
            "type": "object"
        },
        {
            "properties": {
                "transformType": {
```

```
                    "enum": [
                        "Skew"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
                    "type": "string"
                },
                "xAngle": {
                    "id": "tchmi:framework#/definitions/xAngle",
                    "type": "number"
                },
                "xAngleUnit": {
                    "enum": [
                        "deg",
                        "rad",
                        "turn",
                        "grad"
                    ],
                    "id": "tchmi:framework#/definitions/TransformList/definitions/angleUnit",
                    "type": "string"
                },
                "yAngle": {
                    "id": "tchmi:framework#/definitions/yAngle",
                    "type": "number"
                },
                "yAngleUnit": {
                    "enum": [
                        "deg",
                        "rad",
                        "turn",
                        "grad"
                    ],
                    "id": "tchmi:framework#/definitions/TransformList/definitions/angleUnit",
                    "type": "string"
                }
            },
            "required": [
                "transformType"
            ],
            "title": "Skew",
            "type": "object"
        },
        {
            "properties": {
                "transformType": {
                    "enum": [
                        "Origin"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
                    "type": "string"
                },
                "x": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "xUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "y": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "yUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "z": {
                    "$schema": "http://json-schema.org/draft-04/schema",
```

```
                    "id": "tchmi:framework#/definitions/MeasurementValue",
                    "type": "number"
                },
                "zUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                }
            },
            "required": [
                "transformType"
            ],
            "title": "Origin",
            "type": "object"
        },
        {
            "properties": {
                "distance": {
                    "id": "tchmi:framework#/definitions/distance",
                    "type": "number"
                },
                "distanceUnit": {
                    "$schema": "http://json-schema.org/draft-04/schema",
                    "enum": [
                        "px",
                        "%"
                    ],
                    "id": "tchmi:framework#/definitions/MeasurementUnit",
                    "type": "string"
                },
                "origin": {
                    "id": "tchmi:framework#/definitions/origin",
                    "properties": {
                        "x": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "xUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        },
                        "y": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "id": "tchmi:framework#/definitions/MeasurementValue",
                            "type": "number"
                        },
                        "yUnit": {
                            "$schema": "http://json-schema.org/draft-04/schema",
                            "enum": [
                                "px",
                                "%"
                            ],
                            "id": "tchmi:framework#/definitions/MeasurementUnit",
                            "type": "string"
                        }
                    },
                    "type": "object"
                },
                "transformType": {
                    "enum": [
                        "Perspective"
                    ],
                    "id": "tchmi:framework#/definitions/transformType",
                    "type": "string"
                }
            },
            "required": [
                "transformType",
                "distance"
            ],
```

```
                "title": "Perspective",
                "type": "object"
            }
        ]
    },
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Transform [▶ 2097]

18.1.2.79 tchmi:framework#/definitions/TrendLineChartEnd

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartEnd": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "oneOf": [
        {
           "title": "Time",
          "$ref": "tchmi:general#/definitions/DateTime"
        },
        {
           "title": "Timespan",
          "$ref": "tchmi:general#/definitions/TimeSpan"
        },
        {
          "title": "KeyWords",
          "type": "string",
          "enum": [ "Latest" ]
        }
      ]
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.Datatype [▶ 1811]

18.1.2.80 tchmi:framework#/definitions/ TrendLineChartLineGraphDescriptionDefinitionList

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartLineGraphDescriptionDefinitionList": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "TrendLineChartLineGraphDescriptionDefinitionList",
      "type": "array",
      "items": {
        "type": "object",
        "title": "Line graph description",
        "description": "Defines one line graph descriptions.",
        "properties": {
          "symbol": { "type": "string" },
          "yAxisId": { "type": "number" },
          "lineWidth": { "type": "number" },
          "lineColor": { "$ref": "tchmi:framework#/definitions/SolidColor" },
          "pointDot": { "type": "boolean" },
          "pointDotInStopMode": { "type": "boolean" },
          "pointDotRadius": {
```

```
            "type": "number",
            "minimum": 0
          },
          "pointDotFillColor": { "$ref": "tchmi:framework#/definitions/SolidColor" },
          "pointDotStrokeWidth": {
            "type": "number",
            "minimum": 0
          },
          "pointDotStrokeColor": { "$ref": "tchmi:framework#/definitions/SolidColor" }
        },
        "required": [ "yAxisId" ]
      }
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.LineGraphDescription [▶ 1808]

18.1.2.81 tchmi:framework#/definitions/TrendLineChartStart

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartStart": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "oneOf": [
        {
           "title": "Time",
          "$ref": "tchmi:general#/definitions/DateTime"
        },
        {
           "title": "Timespan",
          "$ref": "tchmi:general#/definitions/TimeSpan"
        },
        {
          "title": "KeyWords",
          "type": "string",
          "enum": [ "First" ]
        }
      ]
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.Datatype [▶ 1811]

18.1.2.82 tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationList

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartTooltipErrorInformationList": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "TrendLineChartTooltipErrorInformationList",
      "type": "array",
      "items": {
        "$ref": "tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationObject"
      }
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.TooltipFormat [▶ 412]

tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationObject [▶ 2265]

18.1.2.83 tchmi:framework#/definitions/ TrendLineChartTooltipErrorInformationObject

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartTooltipErrorInformationObject": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "type": "object",
      "title": "TrendLineChart tooltip error information object",
      "properties": {
          "yAxis": { "$ref": "tchmi:framework#/definitions/ChartYAxisDefinition" },
          "beginTime": { "type": "number" },
          "endTime": { "type": "number" },
          "errorCode": { "type": "number" },
          "symbolname": { "type": "string" }
      },
      "required": [ "symbolname" ]
    }
  }
}
```

Note: Available since version 1.10

See also

tchmi:framework#/definitions/TrendLineChartTooltipErrorInformationList [▶ 2264]

18.1.2.84 tchmi:framework#/definitions/TrendLineChartTooltipInformationList

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartTooltipInformationList": {
      "$schema": "http://json-schema.org/draft-04/schema",
      "title": "TrendLineChartTooltipInformationList",
      "type": "array",
      "items": {
        "$ref": "tchmi:framework#/definitions/TrendLineChartTooltipInformationObject"
      }
    }
  }
}
```

Note: Available since version 1.10

See also

TcHmi.Controls.Beckhoff.TcHmiTrendLineChart.TooltipFormat [▶ 412]

tchmi:framework#/definitions/TrendLineChartTooltipInformationObject [▶ 2266]

18.1.2.85 tchmi:framework#/definitions/TrendLineChartTooltipInformationObject

[Schema]

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "definitions": {
    "TrendLineChartTooltipInformationObject": {
      "$schema": "http://json-schema.org/draft-04/schema",
       "type": "object",
        "title": "TrendLineChart tooltip information object",
      "properties": {
        "point": {
          "type": "object",
          "properties": {
            "x": { "type": "number" },
            "y": { "type": "number" }
          },
          "required": [ "x", "y" ]
        },
        "yAxis": { "$ref": "tchmi:framework#/definitions/ChartYAxisDefinition" },
        "symbolname": { "type": "string" }
      },
        "required": [ "point", "symbolname" ]
    }
  }
}
```

Note: Available since version 1.10

See also

tchmi:framework#/definitions/TrendLineChartTooltipInformationList [▶ 2265]

18.1.2.86 tchmi:framework#/definitions/TypeRef

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/TypeRef",
    "type": "string"
}
```

Note: Available since version 1.8

18.1.2.87 tchmi:framework#/definitions/UserControlPath

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/UserControlPath",
    "minLength": 1,
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: string [▶ 2108]

18.1.2.88 tchmi:framework#/definitions/VerticalAlignment

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Top",
        "Center",
        "Bottom"
    ],
    "id": "tchmi:framework#/definitions/VerticalAlignment",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.VerticalAlignment [▶ 2089]

18.1.2.89 tchmi:framework#/definitions/VideoSourceList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "description": "Video Source List",
    "id": "tchmi:framework#/definitions/VideoSourceList",
    "items": {
        "properties": {
            "source": {
                "$schema": "http://json-schema.org/draft-04/schema",
                "id": "tchmi:framework#/definitions/Path",
                "minLength": 1,
                "type": "string"
            },
            "type": {
                "category": "Common",
                "enum": [
                    "",
                    "video/mp4",
                    "video/webm",
                    "video/ogg",
                    "application/dash+xml",
                    "application/vnd.apple.mpegurl"
                ],
                "id": "tchmi:framework#/definitions/type",
                "type": "string"
            }
        },
        "required": [
            "source",
            "type"
        ],
        "type": "object"
    },
    "title": "VideoSourceList",
    "type": "array"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Controls.Beckhoff.TcHmiVideo.VideoSource [▶ 1833]

18.1.2.90 tchmi:framework#/definitions/VirtualControlRightMappingList

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "id": "tchmi:framework#/definitions/VirtualControlRightMappingList",
    "items": {
        "properties": {
            "controlRight": {
```

```
                "description": "Real control right.",
                "id": "tchmi:framework#/definitions/controlRight",
                "minLength": 1,
                "type": "string"
            },
            "virtualControlRight": {
                "description": "Name of the virtual control right.",
                "id": "tchmi:framework#/definitions/virtualControlRight",
                "minLength": 1,
                "type": "string"
            }
        },
        "type": "object"
    },
    "type": "array"
}
```

Note: Available since version 1.8

18.1.2.91 tchmi:framework#/definitions/Visibility

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema",
    "enum": [
        "Visible",
        "Hidden",
        "Collapsed"
    ],
    "id": "tchmi:framework#/definitions/Visibility",
    "type": "string"
}
```

Note: Available since version 1.8

See also

API: TcHmi.Visibility [▶ 2097]

18.1.3 server

18.1.3.1 tchmi:server#/definitions/accessEnum

[Schema]

```
{
    "default": 3,
    "enum": [
        0,
        1,
        2,
        3
    ],
    "id": "tchmi:server#/definitions/accessEnum",
    "options": [
        {
            "label": "ENUM_NONE",
            "value": 0
        },
        {
            "label": "ENUM_READ",
            "value": 1
        },
        {
            "label": "ENUM_READWRITE",
            "value": 3
        },
        {
            "label": "ENUM_WRITE",
            "value": 2
```

```
        }
    ],
    "type": "integer"
}
```

Note: Available since version 1.8

API: TcHmi.Server.ACCESS [▶ 2007]

18.1.3.2 tchmi:server#/definitions/alarm

[Schema]

```
{
    "id": "tchmi:server#/definitions/alarm",
    "properties": {
        "confirmationState": {
            "default": 1,
            "enum": [
                0,
                1,
                2,
                3,
                4
            ],
            "id": "tchmi:server#/definitions/confirmationState",
            "options": [
                {
                    "label": "ENUM_CONFIRMED",
                    "value": 3
                },
                {
                    "label": "ENUM_NOTREQUIRED",
                    "value": 1
                },
                {
                    "label": "ENUM_NOTSUPPORTED",
                    "value": 0
                },
                {
                    "label": "ENUM_RESET",
                    "value": 4
                },
                {
                    "label": "ENUM_WAIT",
                    "value": 2
                }
            ],
            "type": "integer"
        },
        "domain": {
            "id": "tchmi:server#/definitions/domain",
            "type": "string"
        },
        "id": {
            "id": "tchmi:server#/definitions/id",
            "type": "integer"
        },
        "name": {
            "id": "tchmi:server#/definitions/name",
            "type": "string"
        },
        "params": {
            "additionalProperties": {},
            "id": "tchmi:server#/definitions/params",
            "type": "object"
        },
        "severity": {
            "default": 1,
            "enum": [
                0,
                1,
                2,
                3,
                4
            ],
            "id": "tchmi:server#/definitions/severity",
```

```
            "options": [
                {
                    "label": "ENUM_0_VERBOSE",
                    "value": 0
                },
                {
                    "label": "ENUM_1_INFO",
                    "value": 1
                },
                {
                    "label": "ENUM_2_WARNING",
                    "value": 2
                },
                {
                    "label": "ENUM_3_ERROR",
                    "value": 3
                },
                {
                    "label": "ENUM_4_CRITICAL",
                    "value": 4
                }
            ],
            "type": "integer"
        },
        "timeCleared": {
            "format": "datetime",
            "id": "tchmi:server#/definitions/timeCleared",
            "type": "string"
        },
        "timeConfirmed": {
            "format": "datetime",
            "id": "tchmi:server#/definitions/timeConfirmed",
            "type": "string"
        },
        "timeRaised": {
            "format": "datetime",
            "id": "tchmi:server#/definitions/timeRaised",
            "type": "string"
        }
    },
    "required": [
        "name",
        "domain",
        "confirmationState"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.3.3 tchmi:server#/definitions/confirmationState

[Schema]

```
{
    "default": 1,
    "enum": [
        0,
        1,
        2,
        3,
        4
    ],
    "id": "tchmi:server#/definitions/confirmationState",
    "options": [
        {
            "label": "ENUM_CONFIRMED",
            "value": 3
        },
        {
            "label": "ENUM_NOTREQUIRED",
            "value": 1
        },
        {
            "label": "ENUM_NOTSUPPORTED",
            "value": 0
        },
        {
```

```
            "label": "ENUM_RESET",
            "value": 4
        },
        {
            "label": "ENUM_WAIT",
            "value": 2
        }
    ],
    "type": "integer"
}
```

Note: Available since version 1.8

18.1.3.4 tchmi:server#/definitions/event

[Schema]

```
{
    "id": "tchmi:server#/definitions/event",
    "properties": {
        "domain": {
            "id": "tchmi:server#/definitions/domain",
            "type": "string"
        },
        "name": {
            "id": "tchmi:server#/definitions/name",
            "type": "string"
        },
        "payload": {
            "anyOf": [
                {
                    "id": "tchmi:server#/definitions/alarm",
                    "properties": {
                        "confirmationState": {
                            "default": 1,
                            "enum": [
                                0,
                                1,
                                2,
                                3,
                                4
                            ],
                            "id": "tchmi:server#/definitions/confirmationState",
                            "options": [
                                {
                                    "label": "ENUM_CONFIRMED",
                                    "value": 3
                                },
                                {
                                    "label": "ENUM_NOTREQUIRED",
                                    "value": 1
                                },
                                {
                                    "label": "ENUM_NOTSUPPORTED",
                                    "value": 0
                                },
                                {
                                    "label": "ENUM_RESET",
                                    "value": 4
                                },
                                {
                                    "label": "ENUM_WAIT",
                                    "value": 2
                                }
                            ],
                            "type": "integer"
                        },
                        "domain": {
                            "id": "tchmi:server#/definitions/domain",
                            "type": "string"
                        },
                        "id": {
                            "id": "tchmi:server#/definitions/id",
                            "type": "integer"
                        },
                        "name": {
                            "id": "tchmi:server#/definitions/name",
                            "type": "string"
```

```
                    },
                    "params": {
                        "additionalProperties": {},
                        "id": "tchmi:server#/definitions/params",
                        "type": "object"
                    },
                    "severity": {
                        "default": 1,
                        "enum": [
                            0,
                            1,
                            2,
                            3,
                            4
                        ],
                        "id": "tchmi:server#/definitions/severity",
                        "options": [
                            {
                                "label": "ENUM_0_VERBOSE",
                                "value": 0
                            },
                            {
                                "label": "ENUM_1_INFO",
                                "value": 1
                            },
                            {
                                "label": "ENUM_2_WARNING",
                                "value": 2
                            },
                            {
                                "label": "ENUM_3_ERROR",
                                "value": 3
                            },
                            {
                                "label": "ENUM_4_CRITICAL",
                                "value": 4
                            }
                        ],
                        "type": "integer"
                    },
                    "timeCleared": {
                        "format": "datetime",
                        "id": "tchmi:server#/definitions/timeCleared",
                        "type": "string"
                    },
                    "timeConfirmed": {
                        "format": "datetime",
                        "id": "tchmi:server#/definitions/timeConfirmed",
                        "type": "string"
                    },
                    "timeRaised": {
                        "format": "datetime",
                        "id": "tchmi:server#/definitions/timeRaised",
                        "type": "string"
                    }
                },
                "required": [
                    "name",
                    "domain",
                    "confirmationState"
                ],
                "type": "object"
            },
            {
                "id": "tchmi:server#/definitions/message",
                "properties": {
                    "domain": {
                        "id": "tchmi:server#/definitions/domain",
                        "type": "string"
                    },
                    "name": {
                        "id": "tchmi:server#/definitions/name",
                        "type": "string"
                    },
                    "params": {
                        "additionalProperties": {},
                        "id": "tchmi:server#/definitions/params",
                        "type": "object"
                    },
                    "severity": {
```

```
                            "default": 1,
                            "enum": [
                                0,
                                1,
                                2,
                                3,
                                4
                            ],
                            "id": "tchmi:server#/definitions/severity",
                            "options": [
                                {
                                    "label": "ENUM_0_VERBOSE",
                                    "value": 0
                                },
                                {
                                    "label": "ENUM_1_INFO",
                                    "value": 1
                                },
                                {
                                    "label": "ENUM_2_WARNING",
                                    "value": 2
                                },
                                {
                                    "label": "ENUM_3_ERROR",
                                    "value": 3
                                },
                                {
                                    "label": "ENUM_4_CRITICAL",
                                    "value": 4
                                }
                            ],
                            "type": "integer"
                        },
                        "timeRaised": {
                            "format": "datetime",
                            "id": "tchmi:server#/definitions/timeRaised",
                            "type": "string"
                        }
                    },
                    "required": [
                        "name",
                        "domain",
                        "severity"
                    ],
                    "type": "object"
                },
                {}
            ],
            "doc": "Payload can be alarms, messages or any type of user data.",
            "id": "tchmi:server#/definitions/payload"
        },
        "sessionId": {
            "id": "tchmi:server#/definitions/sessionId",
            "type": "string"
        },
        "timeReceived": {
            "format": "datetime",
            "id": "tchmi:server#/definitions/timeReceived",
            "type": "string"
        }
    },
    "required": [
        "name",
        "domain"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.3.5 tchmi:server#/definitions/eventFilter

[Schema]

```
"type": "array",
"items": {
    "anyOf": [
        {
```

```
            "type": "object",
            "properties": {
                "path": {
                    "type": "string",
                    "enum": [ "domain", "name", "text" ]
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "contains",
                        "contains not",
                        "== [ignore case]",
                        "!= [ignore case]",
                        "contains [ignore case]",
                        "contains not [ignore case]"
                    ]
                },
                "value": {
                    "type": "string"
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
            "properties": {
                "path": {
                    "type": "string",
                    "enum": [ "id" ]
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "<",
                        ">",
                        "<=",
                        ">="
                    ]
                },
                "value": {
                    "type": "number"
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
            "properties": {
                "path": {
                    "type": "string",
                    "enum": [ "timeRaised", "timeCleared", "timeConfirmed" ]
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "<",
                        ">",
                        "<=",
                        ">="
                    ]
                },
                "value": {
                    "type": "string",
                    "format": "date-time"
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
            "properties": {
                "path": {
                    "type": "string",
                    "enum": [ "type" ]
```

```
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "<",
                        ">",
                        "<=",
                        ">="
                    ]
                },
                "value": {
                    "type": "integer",
                    "default": 1,
                    "enum": [ 0, 1 ],
                    "options": [
                        {
                            "label": "ENUM_0_MESSAGE",
                            "value": 0
                        },
                        {
                            "label": "ENUM_1_ALARM",
                            "value": 1
                        }
                    ]
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
            "properties": {
                "path": {
                    "type": "string",
                    "enum": [ "severity" ]
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "<",
                        ">",
                        "<=",
                        ">="
                    ]
                },
                "value": {
                    "$ref": "tchmi:server#/definitions/severity"
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
            "properties": {
                "path": {
                    "type": "string",
                    "enum": [ "confirmationState" ]
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "<",
                        ">",
                        "<=",
                        ">="
                    ]
                },
                "value": {
                    "$ref": "tchmi:server#/definitions/confirmationState"
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
```

```
            "properties": {
                "path": {
                    "type": "string"
                },
                "comparator": {
                    "type": "string",
                    "enum": [
                        "==",
                        "!=",
                        "<",
                        ">",
                        "<=",
                        ">=",
                        "contains",
                        "contains not",
                        "== [ignore case]",
                        "!= [ignore case]",
                        "contains [ignore case]",
                        "contains not [ignore case]"
                    ]
                },
                "value": {
                    "oneOf": [
                        {
                            "type": "string"
                        },
                        {
                            "type": "number"
                        },
                        {
                            "type": "boolean"
                        }
                    ]
                }
            },
            "required": [ "path", "comparator", "value" ]
        },
        {
            "type": "object",
            "properties": {
                "logic": {
                    "type": "string",
                    "enum": [ "AND", "OR" ]
                }
            },
            "required": [ "logic" ]
        },
        {
            "$ref": "tchmi:server#/definitions/eventFilter"
        }
    ]
}
```

Note: Available since version 1.10

18.1.3.6 tchmi:server#/definitions/folderRecipe

[Schema]

```
{
    "additionalProperties": {
        "oneOf":
             [
                {
                    "$ref": "tchmi:server#/definitions/recipe"
                },
                {
                    "$ref": "tchmi:server#/definitions/folderRecipe"
                }
            ]
        },
    "type": "object"
}
```

Note: Available since version 1.10.1171.21

18.1.3.7 tchmi:server#/definitions/message

[Schema]

```
{
    "id": "tchmi:server#/definitions/message",
    "properties": {
        "domain": {
            "id": "tchmi:server#/definitions/domain",
            "type": "string"
        },
        "name": {
            "id": "tchmi:server#/definitions/name",
            "type": "string"
        },
        "params": {
            "additionalProperties": {},
            "id": "tchmi:server#/definitions/params",
            "type": "object"
        },
        "severity": {
            "default": 1,
            "enum": [
                0,
                1,
                2,
                3,
                4
            ],
            "id": "tchmi:server#/definitions/severity",
            "options": [
                {
                    "label": "ENUM_0_VERBOSE",
                    "value": 0
                },
                {
                    "label": "ENUM_1_INFO",
                    "value": 1
                },
                {
                    "label": "ENUM_2_WARNING",
                    "value": 2
                },
                {
                    "label": "ENUM_3_ERROR",
                    "value": 3
                },
                {
                    "label": "ENUM_4_CRITICAL",
                    "value": 4
                }
            ],
            "type": "integer"
        },
        "timeRaised": {
            "format": "datetime",
            "id": "tchmi:server#/definitions/timeRaised",
            "type": "string"
        }
    },
    "required": [
        "name",
        "domain",
        "severity"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

See also

API: TcHmi.ServerLogManager.Message

18.1.3.8 tchmi:server#/definitions/recipe

[Schema]

```
{
    "additionalProperties": false,
    "properties": {
        "recipeTypeName": {
            "type": "string"
        },
        "values": {
            "type": "object"
        }
    },
    "required": [ "recipeTypeName", "values" ],
    "type": "object"
}
```

Note: Available since version 1.10.1171.21

18.1.3.9 tchmi:server#/definitions/schemaRef

[Schema]

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "default": null,
    "id": "tchmi:server#/definitions/schemaRef"
}
```

Note: Available since version 1.8

18.1.3.10 tchmi:server#/definitions/severity

[Schema]

```
{
    "default": 1,
    "enum": [
        0,
        1,
        2,
        3,
        4
    ],
    "id": "tchmi:server#/definitions/severity",
    "options": [
        {
            "label": "ENUM_0_VERBOSE",
            "value": 0
        },
        {
            "label": "ENUM_1_INFO",
            "value": 1
        },
        {
            "label": "ENUM_2_WARNING",
            "value": 2
        },
        {
            "label": "ENUM_3_ERROR",
            "value": 3
        },
        {
            "label": "ENUM_4_CRITICAL",
            "value": 4
        }
    ],
    "type": "integer"
}
```

Note: Available since version 1.8

18.1.3.11 tchmi:server#/definitions/symbol

[Schema]

```
{
    "description": "DESC_SYMBOL",
    "id": "tchmi:server#/definitions/symbol",
    "properties": {
        "ACCESS": {
            "default": 3,
            "enum": [
                0,
                1,
                2,
                3
            ],
            "id": "tchmi:server#/definitions/accessEnum",
            "options": [
                {
                    "label": "ENUM_NONE",
                    "value": 0
                },
                {
                    "label": "ENUM_READ",
                    "value": 1
                },
                {
                    "label": "ENUM_READWRITE",
                    "value": 3
                },
                {
                    "label": "ENUM_WRITE",
                    "value": 2
                }
            ],
            "type": "integer"
        },
        "CUSTOMERDATA": {
            "id": "tchmi:server#/definitions/CUSTOMERDATA",
            "type": "string"
        },
        "DOMAIN": {
            "doc": "Domain of the symbol",
            "id": "tchmi:server#/definitions/DOMAIN",
            "optionMethod": "ListDomains",
            "type": "string"
        },
        "HIDDEN": {
            "default": false,
            "doc": "Hide this symbol in the symbol browser.",
            "id": "tchmi:server#/definitions/HIDDEN",
            "type": "boolean"
        },
        "MAPPING": {
            "doc": "This is how the given domain will access the symbol.",
            "id": "tchmi:server#/definitions/MAPPING",
            "type": "string"
        },
        "OPTIONS": {
            "additionalProperties": {},
            "id": "tchmi:server#/definitions/OPTIONS",
            "type": "object"
        },
        "SCHEMA": {
            "$schema": "http://json-schema.org/draft-04/schema#",
            "default": null,
            "doc": "Json schema describing the output parameters of the symbol.",
            "id": "tchmi:server#/definitions/schemaRef"
        },
        "USEMAPPING": {
            "default": true,
            "doc": "If true the mapping will be used, otherwise the symbol will be evaluated in the
server.",
            "id": "tchmi:server#/definitions/USEMAPPING",
            "type": "boolean"
        }
    },
    "required": [
        "DOMAIN",
```

```
        "MAPPING",
        "ACCESS"
    ],
    "type": "object"
}
```

Note: Available since version 1.8

18.1.3.12 tchmi:server#/definitions/threadPoolSize

[Schema]

```
{
    "id": "tchmi:server#/definitions/threadPoolSize",
    "maximum": 128,
    "minimum": 4,
    "type": "integer"
}
```

Note: Available since version 1.8

18.1.3.13 tchmi:server#/definitions/timing

[Schema]

```
{
    "default": 1,
    "enum": [
        0,
        1,
        2
    ],
    "id": "tchmi:server#/definitions/timing",
    "options": [
        {
            "label": "ENUM_DELAYED",
            "value": 2
        },
        {
            "label": "ENUM_EARLY",
            "value": 0
        },
        {
            "label": "ENUM_NORMAL",
            "value": 1
        }
    ],
    "type": "integer"
}
```

Note: Available since version 1.8

18.1.3.14 tchmi:server#/definitions/valueType

[Schema]

```
{
    "default": 0,
    "enum": [
        0,
        1,
        2,
        3,
        4,
        5,
        6,
        7,
        8,
        9,
        10,
        11,
        64,
        65,
```

```
        66,
        256
    ],
    "id": "tchmi:server#/definitions/valueType",
    "options": [
        {
            "label": "ENUM_VALUETYPE_ALARM",
            "value": 66
        },
        {
            "label": "ENUM_VALUETYPE_ANY",
            "value": 256
        },
        {
            "label": "ENUM_VALUETYPE_BINARY",
            "value": 8
        },
        {
            "label": "ENUM_VALUETYPE_BOOL",
            "value": 1
        },
        {
            "label": "ENUM_VALUETYPE_DATETIME",
            "value": 10
        },
        {
            "label": "ENUM_VALUETYPE_DOUBLE",
            "value": 3
        },
        {
            "label": "ENUM_VALUETYPE_EVENT",
            "value": 64
        },
        {
            "label": "ENUM_VALUETYPE_INT",
            "value": 2
        },
        {
            "label": "ENUM_VALUETYPE_INT64",
            "value": 9
        },
        {
            "label": "ENUM_VALUETYPE_MAP",
            "value": 6
        },
        {
            "label": "ENUM_VALUETYPE_MESSAGE",
            "value": 65
        },
        {
            "label": "ENUM_VALUETYPE_NULL",
            "value": 0
        },
        {
            "label": "ENUM_VALUETYPE_STRING",
            "value": 4
        },
        {
            "label": "ENUM_VALUETYPE_STRUCT",
            "value": 7
        },
        {
            "label": "ENUM_VALUETYPE_TIMESPAN",
            "value": 11
        },
        {
            "label": "ENUM_VALUETYPE_VECTOR",
            "value": 5
        }
    ],
    "type": "integer"
}
```

Note: Available since version 1.8

18.2 Return Codes

18.2.1 ADS Return Codes

Grouping of error codes: 0x000 [▶ 2282]..., 0x500 [▶ 2283]..., 0x700 [▶ 2283]..., 0x1000 [▶ 2285]...

Global error codes

Hex	Dec	HRESULT	Name	Description
0x0	0	0x9811 0000	ERR_NOERROR	No error.
0x1	1	0x9811 0001	ERR_INTERNAL	Internal error.
0x2	2	0x9811 0002	ERR_NORTIME	No real-time.
0x3	3	0x9811 0003	ERR_ALLOCLOCKEDMEM	Allocation locked – memory error.
0x4	4	0x9811 0004	ERR_INSERTMAILBOX	Mailbox full – the ADS message could not be sent. Reducing the number of ADS messages per cycle will help.
0x5	5	0x9811 0005	ERR_WRONGRECEIVEHMSG	Wrong HMSG.
0x6	6	0x9811 0006	ERR_TARGETPORTNOTFOUND	Target port not found – ADS server is not started or is not reachable.
0x7	7	0x9811 0007	ERR_TARGETMACHINENOTFOUND	Target computer not found – AMS route was not found.
0x8	8	0x9811 0008	ERR_UNKNOWNCMDID	Unknown command ID.
0x9	9	0x9811 0009	ERR_BADTASKID	Invalid task ID.
0xA	10	0x9811 000A	ERR_NOIO	No IO.
0xB	11	0x9811 000B	ERR_UNKNOWNAMSCMD	Unknown AMS command.
0xC	12	0x9811 000C	ERR_WIN32ERROR	Win32 error.
0xD	13	0x9811 000D	ERR_PORTNOTCONNECTED	Port not connected.
0xE	14	0x9811 000E	ERR_INVALIDAMSLENGTH	Invalid AMS length.
0xF	15	0x9811 000F	ERR_INVALIDAMSNETID	Invalid AMS Net ID.
0x10	16	0x9811 0010	ERR_LOWINSTLEVEL	Low installation level – TwinCAT 2 license error.
0x11	17	0x9811 0011	ERR_NODEBUGINTAVAILABLE	No debugging available.
0x12	18	0x9811 0012	ERR_PORTDISABLED	Port disabled – TwinCAT system service not started.
0x13	19	0x9811 0013	ERR_PORTALREADYCONNECTED	Port already connected.
0x14	20	0x9811 0014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 error.
0x15	21	0x9811 0015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x9811 0016	ERR_AMSSYNC_AMSERROR	AMS Sync error.
0x17	23	0x9811 0017	ERR_AMSSYNC_NOINDEXINMAP	No index map for AMS Sync available.
0x18	24	0x9811 0018	ERR_INVALIDAMSPORT	Invalid AMS port.
0x19	25	0x9811 0019	ERR_NOMEMORY	No memory.
0x1A	26	0x9811 001A	ERR_TCPSEND	TCP send error.

Hex	Dec	HRESULT	Name	Description
0x1B	27	0x9811 001B	ERR_HOSTUNREACHABLE	Host unreachable.
0x1C	28	0x9811 001C	ERR_INVALIDAMSFRAGMENT	Invalid AMS fragment.
0x1D	29	0x9811 001D	ERR_TLSSEND	TLS send error – secure ADS connection failed.
0x1E	30	0x9811 001E	ERR_ACCESSDENIED	Access denied – secure ADS access denied.

Router error codes

Hex	Dec	HRESULT	Name	Description
0x500	1280	0x9811 0500	ROUTERERR_NOLOCKEDMEMORY	Locked memory cannot be allocated.
0x501	1281	0x9811 0501	ROUTERERR_RESIZEMEMORY	The router memory size could not be changed.
0x502	1282	0x9811 0502	ROUTERERR_MAILBOXFULL	The mailbox has reached the maximum number of possible messages.
0x503	1283	0x9811 0503	ROUTERERR_DEBUGBOXFULL	The mailbox has reached the maximum number of possible messages.
0x504	1284	0x9811 0504	ROUTERERR_UNKNOWNPORTTYPE	The port type is unknown.
0x505	1285	0x9811 0505	ROUTERERR_NOTINITIALIZED	The router is not initialized.
0x506	1286	0x9811 0506	ROUTERERR_PORTALREADYINUSE	The port number is already assigned.
0x507	1287	0x9811 0507	ROUTERERR_NOTREGISTERED	The port is not registered.
0x508	1288	0x9811 0508	ROUTERERR_NOMOREQUEUES	The maximum number of ports has been reached.
0x509	1289	0x9811 0509	ROUTERERR_INVALIDPORT	The port is invalid.
0x50A	1290	0x9811 050A	ROUTERERR_NOTACTIVATED	The router is not active.
0x50B	1291	0x9811 050B	ROUTERERR_FRAGMENTBOXFULL	The mailbox has reached the maximum number for fragmented messages.
0x50C	1292	0x9811 050C	ROUTERERR_FRAGMENTTIMEOUT	A fragment timeout has occurred.
0x50D	1293	0x9811 050D	ROUTERERR_TOBEREMOVED	The port is removed.

General ADS error codes

Hex	Dec	HRESULT	Name	Description
0x700	1792	0x9811 0700	ADSERR_DEVICE_ERROR	General device error.
0x701	1793	0x9811 0701	ADSERR_DEVICE_SRVNOTSUPP	Service is not supported by the server.
0x702	1794	0x9811 0702	ADSERR_DEVICE_INVALIDGRP	Invalid index group.
0x703	1795	0x9811 0703	ADSERR_DEVICE_INVALIDOFFSET	Invalid index offset.
0x704	1796	0x9811 0704	ADSERR_DEVICE_INVALIDACCESS	Reading and writing not permitted.
0x705	1797	0x9811 0705	ADSERR_DEVICE_INVALIDSIZE	Parameter size not correct.
0x706	1798	0x9811 0706	ADSERR_DEVICE_INVALIDDATA	Invalid parameter values.
0x707	1799	0x9811 0707	ADSERR_DEVICE_NOTREADY	Device is not ready to operate.
0x708	1800	0x9811 0708	ADSERR_DEVICE_BUSY	Device is busy.
0x709	1801	0x9811 0709	ADSERR_DEVICE_INVALIDCONTEXT	Invalid operating system context. This can result from use of ADS function blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.
0x70A	1802	0x9811 070A	ADSERR_DEVICE_NOMEMORY	Insufficient memory.
0x70B	1803	0x9811 070B	ADSERR_DEVICE_INVALIDPARM	Invalid parameter values.
0x70C	1804	0x9811 070C	ADSERR_DEVICE_NOTFOUND	Not found (files, ...).
0x70D	1805	0x9811 070D	ADSERR_DEVICE_SYNTAX	Syntax error in file or command.
0x70E	1806	0x9811 070E	ADSERR_DEVICE_INCOMPATIBLE	Objects do not match.
0x70F	1807	0x9811 070F	ADSERR_DEVICE_EXISTS	Object already exists.
0x710	1808	0x9811 0710	ADSERR_DEVICE_SYMBOLNOTFOUND	Symbol not found.

Hex	Dec	HRESULT	Name	Description
0x711	1809	0x9811 0711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	Invalid symbol version. This can occur due to an online change. Create a new handle.
0x712	1810	0x9811 0712	ADSERR_DEVICE_INVALIDSTATE	Device in invalid state.
0x713	1811	0x9811 0713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode not supported.
0x714	1812	0x9811 0714	ADSERR_DEVICE_NOTIFYHNDINVALID	Notification handle is invalid.
0x715	1813	0x9811 0715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification client not registered.
0x716	1814	0x9811 0716	ADSERR_DEVICE_NOMOREHDLS	No further notification handle.
0x717	1815	0x9811 0717	ADSERR_DEVICE_INVALIDWATCHSIZE	Notification size too large.
0x718	1816	0x9811 0718	ADSERR_DEVICE_NOTINIT	Device not initialized.
0x719	1817	0x9811 0719	ADSERR_DEVICE_TIMEOUT	Device has a timeout.
0x71A	1818	0x9811 071A	ADSERR_DEVICE_NOINTERFACE	Interface query failed.
0x71B	1819	0x9811 071B	ADSERR_DEVICE_INVALIDINTERFACE	Wrong interface requested.
0x71C	1820	0x9811 071C	ADSERR_DEVICE_INVALIDCLSID	Class ID is invalid.
0x71D	1821	0x9811 071D	ADSERR_DEVICE_INVALIDOBJID	Object ID is invalid.
0x71E	1822	0x9811 071E	ADSERR_DEVICE_PENDING	Request pending.
0x71F	1823	0x9811 071F	ADSERR_DEVICE_ABORTED	Request is aborted.
0x720	1824	0x9811 0720	ADSERR_DEVICE_WARNING	Signal warning.
0x721	1825	0x9811 0721	ADSERR_DEVICE_INVALIDARRAYIDX	Invalid array index.
0x722	1826	0x9811 0722	ADSERR_DEVICE_SYMBOLNOTACTIVE	Symbol not active.
0x723	1827	0x9811 0723	ADSERR_DEVICE_ACCESSDENIED	Access denied.
0x724	1828	0x9811 0724	ADSERR_DEVICE_LICENSENOTFOUND	Missing license.
0x725	1829	0x9811 0725	ADSERR_DEVICE_LICENSEEXPIRED	License expired.
0x726	1830	0x9811 0726	ADSERR_DEVICE_LICENSEEXCEEDED	License exceeded.
0x727	1831	0x9811 0727	ADSERR_DEVICE_LICENSEINVALID	Invalid license.
0x728	1832	0x9811 0728	ADSERR_DEVICE_LICENSESYSTEMID	System ID license invalid.
0x729	1833	0x9811 0729	ADSERR_DEVICE_LICENSENOTIMELIMIT	License not limited in time.
0x72A	1834	0x9811 072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	License problem: Time in the future.
0x72B	1835	0x9811 072B	ADSERR_DEVICE_LICENSETIMETOLONG	License period too long.
0x72C	1836	0x9811 072C	ADSERR_DEVICE_EXCEPTION	Exception at system startup.
0x72D	1837	0x9811 072D	ADSERR_DEVICE_LICENSEDUPLICATED	License file read twice.
0x72E	1838	0x9811 072E	ADSERR_DEVICE_SIGNATUREINVALID	Invalid signature.
0x72F	1839	0x9811 072F	ADSERR_DEVICE_CERTIFICATEINVALID	Invalid certificate.
0x730	1840	0x9811 0730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	Public key not known from OEM.
0x731	1841	0x9811 0731	ADSERR_DEVICE_LICENSERESTRICTED	License not valid for this system ID.
0x732	1842	0x9811 0732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo license prohibited.
0x733	1843	0x9811 0733	ADSERR_DEVICE_INVALIDFNCID	Invalid function ID.
0x734	1844	0x9811 0734	ADSERR_DEVICE_OUTOFRANGE	Outside the valid range.
0x735	1845	0x9811 0735	ADSERR_DEVICE_INVALIDALIGNMENT	Invalid alignment.
0x736	1846	0x9811 0736	ADSERR_DEVICE_LICENSEPLATFORM	Invalid platform level.
0x737	1847	0x9811 0737	ADSERR_DEVICE_FORWARD_PL	Context – forward to passive level.
0x738	1848	0x9811 0738	ADSERR_DEVICE_FORWARD_DL	Context – forward to dispatch level.
0x739	1849	0x9811 0739	ADSERR_DEVICE_FORWARD_RT	Context – forward to real-time.
0x740	1856	0x9811 0740	ADSERR_CLIENT_W32ERROR	Client error.
0x741	1857	0x9811 0741	ADSERR_CLIENT_INVALIDPARM	Service contains an invalid parameter.
0x742	1858	0x9811 0742	ADSERR_CLIENT_LISTEMPTY	Polling list is empty.
0x743	1859	0x9811 0743	ADSERR_CLIENT_VARUSED	Var connection already in use.
0x744	1860	0x9811 0744	ADSERR_CLIENT_DUPLINVOKEID	The called ID is already in use.
0x745	1861	0x9811 0745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout has occurred – the remote terminal is not responding in the specified ADS timeout. The route setting of the remote terminal may be configured incorrectly.
0x746	1862	0x9811 0746	ADSERR_CLIENT_W32ERROR	Error in Win32 subsystem.
0x747	1863	0x9811 0747	ADSERR_CLIENT_TIMEOUTINVALID	Invalid client timeout value.
0x748	1864	0x9811 0748	ADSERR_CLIENT_PORTNOTOPEN	Port not open.
0x749	1865	0x9811 0749	ADSERR_CLIENT_NOAMSADDR	No AMS address.
0x750	1872	0x9811 0750	ADSERR_CLIENT_SYNCINTERNAL	Internal error in Ads sync.
0x751	1873	0x9811 0751	ADSERR_CLIENT_ADDHASH	Hash table overflow.

Hex	Dec	HRESULT	Name	Description
0x752	1874	0x9811 0752	ADSERR_CLIENT_REMOVEHASH	Key not found in the table.
0x753	1875	0x9811 0753	ADSERR_CLIENT_NOMORESYM	No symbols in the cache.
0x754	1876	0x9811 0754	ADSERR_CLIENT_SYNCRESINVALID	Invalid answer received.

RTime error codes

Hex	Dec	HRESULT	Name	Description
0x1000	4096	0x9811 1000	RTERR_INTERNAL	Internal error in the real-time system.
0x1001	4097	0x9811 1001	RTERR_BADTIMERPERIODS	Timer value is not valid.
0x1002	4098	0x9811 1002	RTERR_INVALIDTASKPTR	Task pointer has the invalid value 0 (zero).
0x1003	4099	0x9811 1003	RTERR_INVALIDSTACKPTR	Task stack pointer has the invalid value 0.
0x1004	4100	0x9811 1004	RTERR_PRIOEXISTS	The request task priority is already assigned.
0x1005	4101	0x9811 1005	RTERR_NOMORETCB	No free TCB (Task Control Block) available. The maximum number of TCBs is 64.
0x1006	4102	0x9811 1006	RTERR_NOMORESEMAS	No free semaphores available. The maximum number of semaphores is 64.
0x1007	4103	0x9811 1007	RTERR_NOMOREQUEUES	No free space available in the queue. The maximum number of positions in the queue is 64.
0x100D	4109	0x9811 100D	RTERR_EXTIRQALREADYDEF	An external synchronization interrupt is already applied.
0x100E	4110	0x9811 100E	RTERR_EXTIRQNOTDEF	No external sync interrupt applied.
0x100F	4111	0x9811 100F	RTERR_EXTIRQINSTALLFAILED	Application of the external synchronization interrupt failed
0x1010	4112	0x9811 1010	RTERR_IRQLNOTLESSOREQUAL	Call of a service function in the wrong context
0x1017	4119	0x9811 1017	RTERR_VMXNOTSUPPORTED	Intel VT-x extension is not supported.
0x1018	4120	0x9811 1018	RTERR_VMXDISABLED	Intel VT-x extension is not enabled in the BIOS.
0x1019	4121	0x9811 1019	RTERR_VMXCONTROLSMISSING	Missing function in Intel VT-x extension.
0x101A	4122	0x9811 101A	RTERR_VMXENABLEFAILS	Activation of Intel VT-x fails.

TCP Winsock error codes

Hex	Dec	Name	Description
0x274C	10060	WSAETIMEDOUT	A connection timeout has occurred. Connection error because the remote terminal did not respond properly after a certain period of time, or the connected host did not respond and the connection could therefore not be maintained.
0x274D	10061	WSAECONNREFUSED	Connection refused. No connection could be established because the target computer explicitly rejected it. This error usually results from an attempt to connect to a service that is inactive on the external host, that is, a service for which no server application is running.
0x2751	10065	WSAEHOSTUNREACH	No route to host A socket operation referred to an unavailable host.
More Winsock error codes: Win32 error codes			

18.3 Troubleshooting

The product version number can be found via the Systray icon, the Visual Studio menu **Help > About** or under **Programs** in the Windows Control Panel. Additional information is required:

- Visual Studio Version
- Operating system

- Windows error reports (if any)
- Console errors (if any)

18.3.1 Windows error reporting

If Visual Studio crashes, the engineering process (devenv.exe) generates a crash dump. This crash dump helps with error analysis. The crash dumps are located under the following path: *\User\Username\AppData\Local\CrashDumps.* Please send this crash dump to Support with the product version number. Additionally, a project and instructions to recreate the problem would be very helpful.

If the Appdata folder is not visible, the hidden files are hidden.

Further information can be found via Windows Error Reporting.

18.3.2 Client Console

Modern HTML5 browsers provide helpful developer tools for debugging and analyzing. For example, a console is provided in which errors and status messages can be displayed. The TwinCAT HMI server and framework use the console to output errors, warnings and debug information. The log level of the output can be adjusted via the project settings [▶ 43].

Further Information:

- Chrome DevTools
- Mozilla Firefox Developer Tools
- Microsoft Edge F12 devtools

18.3.3 Engineering server

When a TwinCAT HMI project is opened, a TwinCAT HMI Engineering server instance is started in the background. The server logs all messages in the Visual Studio output. In case of a crash, a dump of the server process is stored in the folder *TcHmiProject\.engineering_servers* . Please send this crash dump to Support with the product version number. Additionally, a project and instructions to recreate the problem would be very helpful.