

ControlDesk

# Measurement Data API

For ControlDesk 7.4

Release 2021-A – May 2021

## How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	<a href="mailto:info@dspace.de">info@dspace.de</a>
Web:	<a href="http://www.dspace.com">http://www.dspace.com</a>

## How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: <http://www.dspace.com/go/locations>
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.  
Tel.: +49 5251 1638-941 or e-mail: [support@dspace.de](mailto:support@dspace.de)

You can also use the support request form: <http://www.dspace.com/go/supportrequest>. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

## Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit <http://www.dspace.com/go/patches> for software updates and patches.

## Important Notice

This publication contains proprietary information that is protected by copyright. All rights are reserved. The publication may be printed for personal or internal use provided all the proprietary markings are retained on all printed copies. In all other cases, the publication must not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

© 2010 - 2021 by:  
dSPACE GmbH  
Rathenaustraße 26  
33102 Paderborn  
Germany

This publication and the contents hereof are subject to change without notice.

AUTERA, ConfigurationDesk, ControlDesk, MicroAutoBox, MicroLabBox, SCALEXIO, SIMPHERA, SYNECT, SystemDesk, TargetLink and VEOS are registered trademarks of dSPACE GmbH in the United States or other countries, or both. Other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

# Contents

About This Document	9
Basics and Instructions	11
Basics on the Measurement Data API	11
Measurement Data API Demos	12
API Reference Information	15
Overview	16
Overview of Interfaces	16
Interfaces	18
AxisTemplate	20
Class Description (AxisTemplate)	20
Bookmark	21
Class Description (Bookmark)	22
Bookmarks (Collection)	22
Class Description (Bookmarks)	23
Add (Bookmarks)	24
Item (Bookmarks)	24
Remove (Bookmarks)	25
RemoveAll (Bookmarks)	26
DescriptionCategories (Collection)	26
Class Description (DescriptionCategories)	26
Item (DescriptionCategories)	27
DescriptionCategory	28
Class Description (DescriptionCategory)	28
FileDescription	29
Class Description (FileDescription)	29
FormatOption	30
Class Description (FormatOption)	30
FormatOptions (Collection)	31
Class Description (FormatOptions)	31
Item (FormatOptions)	32

FormulaScaling.....	33
Class Description (FormulaScaling).....	33
GeneralDescription.....	33
Class Description (GeneralDescription).....	33
LinearScaling.....	34
Class Description (LinearScaling).....	34
MDFDescription.....	35
Class Description (MDFDescription).....	35
MDFFormatOption.....	36
Class Description (MDFFormatOption).....	36
Measurement.....	36
Class Description (Measurement).....	37
Export (Measurement).....	38
GetDeviceOptions (Measurement).....	39
Save (Measurement).....	39
SetDeviceOptions (Measurement).....	40
SetSection (Measurement).....	41
MeasurementDescription.....	42
Class Description (MeasurementDescription).....	42
Measurements (Collection).....	43
Class Description (Measurements).....	43
Add (Measurements).....	44
Item (Measurements).....	45
Load (Measurements).....	46
Remove (Measurements).....	47
RemoveAll (Measurements).....	48
SaveAll (Measurements).....	49
MultiScaling.....	49
Class Description (MultiScaling).....	49
MultiScalingTableEntries (Collection).....	50
Class Description (MultiScalingTableEntries).....	50
Add (MultiScalingTableEntries).....	51
Item (MultiScalingTableEntries).....	52
Remove (MultiScalingTableEntries).....	53
MultiScalingTableEntry.....	53
Class Description (MultiScalingTableEntry).....	54

NumericTableEntries (Collection).....	54
Class Description (NumericTableEntries).....	55
Add (NumericTableEntries).....	56
Item (NumericTableEntries).....	56
Remove (NumericTableEntries).....	57
NumericTableEntry.....	58
Class Description (NumericTableEntry).....	58
NumericTableScaling.....	59
Class Description (NumericTableScaling).....	59
RationalFunctionScaling.....	59
Class Description (RationalFunctionScaling).....	59
RecordingDescription.....	60
Class Description (RecordingDescription).....	60
ScalarScaling.....	61
Class Description (ScalarScaling).....	61
Scaling.....	62
Class Description (Scaling).....	62
Scalings (Collection).....	63
Class Description (Scalings).....	63
Add (Scalings).....	64
CreateTable (Scalings).....	65
Item (Scalings).....	66
Remove (Scalings).....	66
RemoveAll (Scalings).....	67
Signal.....	68
Class Description (Signal).....	68
LoadData (Signal).....	70
UnloadData (Signal).....	71
Signals (Collection).....	71
Class Description (Signals).....	72
Add (Signals).....	73
CreateAxisTemplate (Signals).....	74
Item (Signals).....	75
LoadData (Signals).....	76
Remove (Signals).....	76
RemoveAll (Signals).....	77
UnloadData (Signals).....	78

SignalWithSource.....	79
Class Description (SignalWithSource).....	79
VerbalTableEntries (Collection).....	79
Class Description (VerbalTableEntries).....	80
Add (VerbalTableEntries).....	80
Item (VerbalTableEntries).....	81
Remove (VerbalTableEntries).....	82
VerbalTableEntry.....	83
Class Description (VerbalTableEntry).....	83
VerbalTableRangeEntries (Collection).....	83
Class Description (VerbalTableRangeEntries).....	84
Add (VerbalTableRangeEntries).....	84
Item (VerbalTableRangeEntries).....	85
Remove (VerbalTableRangeEntries).....	86
VerbalTableRangeEntry.....	87
Class Description (VerbalTableRangeEntry).....	87
VerbalTableRangeScaling.....	88
Class Description (VerbalTableRangeScaling).....	88
VerbalTableScaling.....	88
Class Description (VerbalTableScaling).....	88
XAxes (Collection).....	89
Class Description (XAxes).....	89
Add (XAxes).....	90
Item (XAxes).....	91
Remove (XAxes).....	91
RemoveAll (XAxes).....	92
XAxis.....	93
Class Description (XAxis).....	93
Constants.....	94
BookmarkTypeConstants.....	94
MDFVariableNameStorageConstants.....	95
MeasurementDeviceOptions.....	96
MeasurementLoadingOptions.....	97
MeasurementRemovalOptions.....	97
MeasurementSectionOptions.....	98
ScalingTypeConstants.....	98
SignalTypeConstants.....	99
VariableTypeConstants.....	100

Limitations	103
Limitations for Using the Measurement Data API.....	103
Glossary	105
Index	143







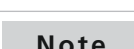





# About This Document

**Content** This document introduces you to ControlDesk's Measurement Data API.

## Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
	Indicates a hazard that, if not avoided, could result in property damage.
	Indicates important information that you should take into account to avoid malfunctions.
	Indicates tips that can make your work easier.
	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
	Precedes the document title in a link that refers to another document.

## Naming conventions

dSPACE user documentation uses the following naming conventions:

**%name%** Names enclosed in percent signs refer to environment variables for file and path names.

**< >** Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

---

## Special folders

Some software products use the following special folders:

**Common Program Data folder** A standard folder for application-specific configuration data that is used by all users.

`%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>`

or

`%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>`

**Documents folder** A standard folder for user-specific documents.

`%USERPROFILE%\Documents\dSPACE\<ProductName>\<VersionNumber>`

**Local Program Data folder** A standard folder for application-specific configuration data that is used by the current, non-roaming user.

`%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\<ProductName>`

---

## Accessing dSPACE Help and PDF Files


After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

**dSPACE Help (local)** You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

**dSPACE Help (Web)** You can access the Web version of dSPACE Help at [www.dspace.com/go/help](http://www.dspace.com/go/help).

To access the Web version, you must have a *mydSPACE* account.

**PDF files** You can access PDF files via the  icon in dSPACE Help. The PDF opens on the first page.

# Basics and Instructions

## Where to go from here

## Information in this section

<a href="#">Basics on the Measurement Data API.....</a>	<a href="#">11</a>
The Measurement Data API allows you to load, edit and create measurement data files.	
<a href="#">Measurement Data API Demos.....</a>	<a href="#">12</a>
Demonstrate how to load, edit and create measurement data files using ControlDesk's Measurement Data API.	

## Basics on the Measurement Data API

### Introduction

The Measurement Data API allows you to load, edit and create measurement data files. You can add and delete time axes, scalings, signals, bookmarks, and descriptions to/from a measurement data file.

#### Note

Scripts are case-sensitive. Script text must match the spelling of the API documentation.

### Accessing the Measurement Data API

You can use the Measurement Data API with Python.

To access the Measurement Data API, import the `measurementdataapilib` Python module:

```
import measurementdataapilib
```

**Related topics****References**

[Overview of Interfaces.....](#) 16

## Measurement Data API Demos

**Description of the demos**

The table below lists the Python demo scripts that show how to use the Measurement Data API interfaces:

Demo Script	Description
CreateAddAndRemove.py	Shows basic aspects of the modification of measurements. It shows how to call the methods Create, Add, and Remove for measurements, x-axes, scalings, signals, bookmarks, and description categories.
DescriptionCategories.py	Explains how the description categories are used with the Measurement Data API. Measurements contain description information arranged in categories. The description categories do not have to be created or removed. They are created automatically with the measurement. The attributes of the description categories can have different data types.
Downsample.py	Performs downsampling on a measurement. The script needs the following command line arguments: <b>Downsample</b> <Factor> <SourceFilePath> [<DestinationFilePath>]
ListMeasurements.py	Lists the measurement files that are passed as arguments to the standard output. The script needs the following command line arguments: <b>ListMeasurements</b> <FilePath1> [<FilePath2> [<FilePath3> [...]] ]
LoadSaveAndExport.py	Shows basic aspects of the creation and removal of measurements. It shows how to call the methods Load, Save, and Export for measurements.
LoadSection.py	Shows how to load a part of a measurement by defining a time section.
Lookups.py	Shows how to create and access look-up tables.
MarkMinMax.py	Marks the minimum and maximum values of a measurement by bookmarks. The script needs the following command line arguments: <b>MarkMinMax</b> <SignalKey> <SourceFilePath> [<DestinationFilePath>]
MeasurementArrays.py	Shows how to create measurement arrays.
ScalingTypes.py	Scalings are used to transform a measurement signal's source value into a converted value. This script explains all the scaling types and demonstrates how scalings are used with the Measurement Data API.
SignalIdentification.py	Shows how signals, x-axes, and scalings are identified within their collections.

**Location of the demos**

All the demo scripts are located in the `.\Demos\MeasurementDataAPI\Python` folder of your ControlDesk installation.

Related topics

Basics

Basics on the Measurement Data API..... 11



# API Reference Information

---

## Where to go from here

## Information in this section

Overview.....	16
Interfaces.....	18
Constants.....	94

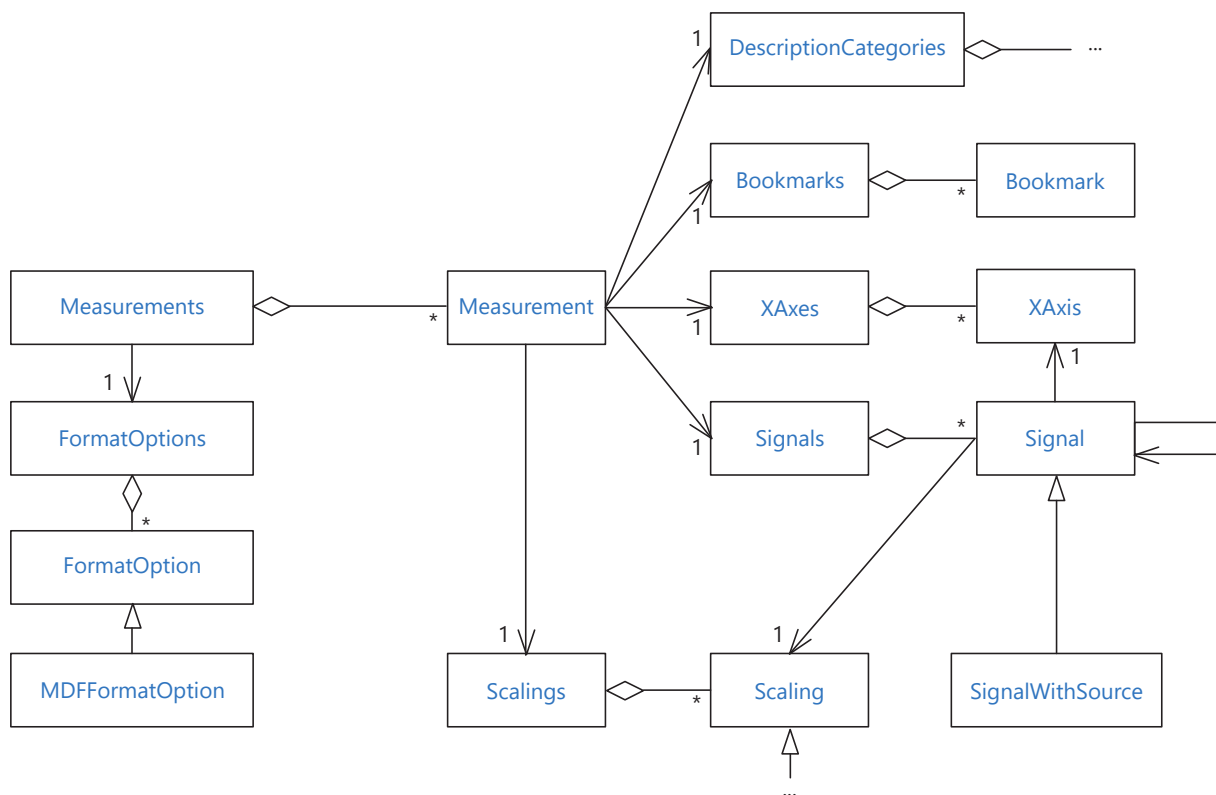
# Overview

## Overview of Interfaces

### Graphical overview

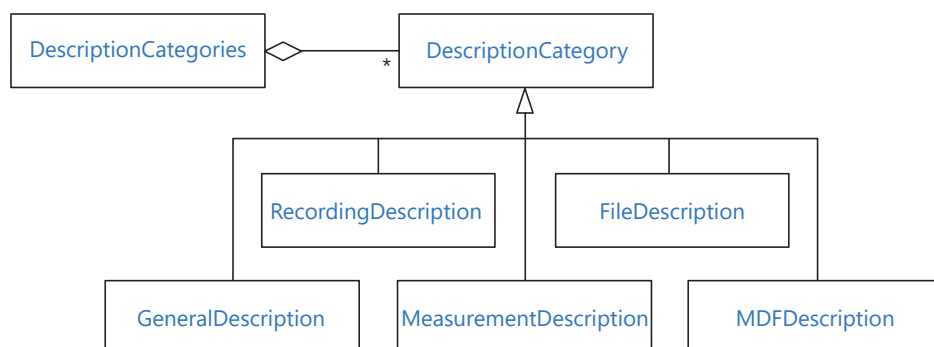
The following illustrations show the dependencies of all interfaces in a graphical overview using UML graphic notation.

### Measurements

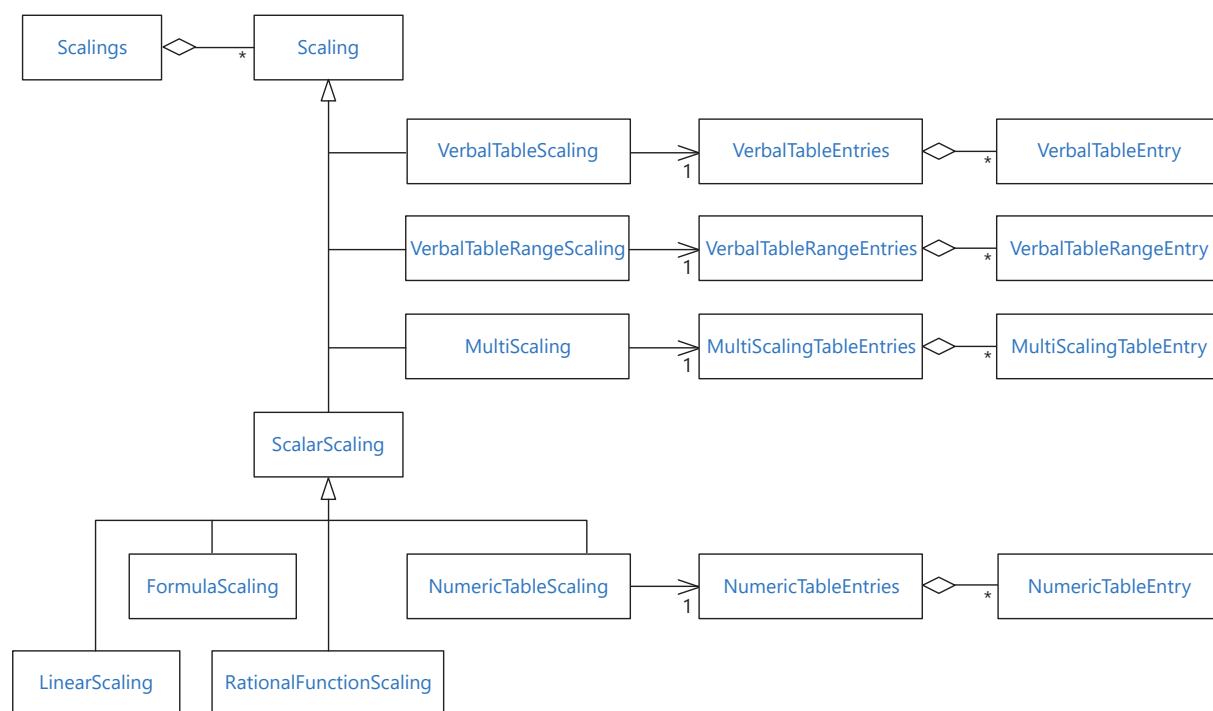




## DescriptionCategories



## Scalings



## Related topics

### Basics

Basics on the Measurement Data API..... 11

# Interfaces

## Introduction

The description of the interfaces is optimized for Python users. Input and output objects are described as attributes and methods of an interface (class). You will also find Python syntax for each method and Python script examples.

## Where to go from here

## Information in this section

<a href="#">AxisTemplate</a> .....	20
To access an axis template for a fixed or standard axis of a curve or map.	
<a href="#">Bookmark</a> .....	21
To access a bookmark of the measurement.	
<a href="#">Bookmarks (Collection)</a> .....	22
To provide a collection of the measurement's bookmarks.	
<a href="#">DescriptionCategories (Collection)</a> .....	26
To provide a collection of the measurement's description categories.	
<a href="#">DescriptionCategory</a> .....	28
To access a description category of the measurement.	
<a href="#">FileDescription</a> .....	29
To access the File description category.	
<a href="#">FormatOption</a> .....	30
To access a file format option of the measurement.	
<a href="#">FormatOptions (Collection)</a> .....	31
To provide a collection of the measurement's file format options. The FormatOption class is currently not used.	
<a href="#">FormulaScaling</a> .....	33
To access a formula-based scaling of the measurement.	
<a href="#">GeneralDescription</a> .....	33
To access the General description category.	
<a href="#">LinearScaling</a> .....	34
To access a linear scaling of the measurement.	
<a href="#">MDFDescription</a> .....	35
To access the MDF description category.	
<a href="#">MDFFormatOption</a> .....	36
To access the MDF file format option.	
<a href="#">Measurement</a> .....	36
To access a measurement.	
<a href="#">MeasurementDescription</a> .....	42
To access the Measurement description category.	

<a href="#">Measurements (Collection)</a> .....	43
To provide a collection of all measurements loaded.	
<a href="#">MultiScaling</a> .....	49
To access a table that maps numerical source value ranges to scalings.	
<a href="#">MultiScalingTableEntries (Collection)</a> .....	50
To provide a collection representing all entries in a multiscaling table, which maps ranges of numerical source values to different scalings.	
<a href="#">MultiScalingTableEntry</a> .....	53
To access an entry of a multiscaling table, which maps ranges of numerical source values to scalings.	
<a href="#">NumericTableEntries (Collection)</a> .....	54
To provide a collection representing all entries in a conversion table for converting numerical values (CompuTab).	
<a href="#">NumericTableEntry</a> .....	58
To access an entry of a conversion table for converting numerical values (CompuTab).	
<a href="#">NumericTableScaling</a> .....	59
To access a collection representing a conversion table for converting numerical values (CompuTab).	
<a href="#">RationalFunctionScaling</a> .....	59
To access a rational function scaling of the measurement.	
<a href="#">RecordingDescription</a> .....	60
To access the Recording description category.	
<a href="#">ScalarScaling</a> .....	61
To access a scalar scaling of the measurement.	
<a href="#">Scaling</a> .....	62
To access a scaling of the measurement.	
<a href="#">Scalings (Collection)</a> .....	63
To provide a collection of the measurement's scalings.	
<a href="#">Signal</a> .....	68
To access a signal of the measurement.	
<a href="#">Signals (Collection)</a> .....	71
To provide a collection of the measurement's signal objects.	
<a href="#">SignalWithSource</a> .....	79
To access a signal of the measurement.	
<a href="#">VerbalTableEntries (Collection)</a> .....	79
To provide a collection representing all entries in a conversion table for converting numerical values into strings (CompuVTab).	
<a href="#">VerbalTableEntry</a> .....	83
To access an entry of a conversion table for converting numerical values into strings (CompuVTab).	

<a href="#">VerbalTableRangeEntries (Collection)</a> .....	83
To provide a collection representing all entries in a conversion table for converting ranges of numerical values into strings (CompuVTabRange).	
<a href="#">VerbalTableRangeEntry</a> .....	87
To access an entry of a conversion table for converting ranges of numerical values into strings (CompuVTabRange).	
<a href="#">VerbalTableRangeScaling</a> .....	88
To access a conversion table for converting ranges of numerical values into strings (CompuVTabRange).	
<a href="#">VerbalTableScaling</a> .....	88
To access a conversion table for converting numerical values into strings (CompuVTab).	
<a href="#">XAxes (Collection)</a> .....	89
To provide a collection of the measurement's x-axis (time axes) objects.	
<a href="#">XAxis</a> .....	93
To access an x-axis (time axis) of the measurement.	

## AxisTemplate

### Where to go from here

### Information in this section

<a href="#">Class Description (AxisTemplate)</a> .....	20
To access an axis template for a fixed or standard axis of a curve or map.	

### Information in other sections

<a href="#">Signals (Collection)</a> .....	71
To provide a collection of the measurement's signal objects.	

## Class Description (AxisTemplate)

### Purpose

To access an axis template for a fixed or standard axis of a curve or map.

**Description**

The axis template stores the values, the scaling, and the input quantity (optional). It is used for the creation of look-up tables (curves or maps) with a fixed or standard axis.

When you add a curve or a map to the signals collection, the axis templates are passed to the list of components of the signal's **Add** method in the following order: x-axis, y-axis.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Data	String	To get/set the values of the axis (axis points).
Scaling	String	To get/set the scaling of the axis.
Components	Signal	To get/set the input quantity. Signals that are used as input quantities must either have the variable type measurement or parameter.

**Methods**

-

**Related topics****References**

[CreateAxisTemplate \(Signals\).....](#) 74

## Bookmark

**Where to go from here****Information in this section**

[Class Description \(Bookmark\).....](#) 22  
To access a bookmark of the measurement.

**Information in other sections**

[Bookmarks \(Collection\).....](#) 22  
To provide a collection of the measurement's bookmarks.

## Class Description (Bookmark)

**Purpose** To access a bookmark of the measurement.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Timestamp	Float	To get/set the bookmark's time stamp. The time stamp is measured in seconds from the beginning of the recording.
Type	<a href="#">BookmarkTypeConstants</a> on page 94	To get/set the bookmark's type. The bookmark type describes the event that invokes the bookmark.
Title	String	To get/set the bookmark's title. The bookmark's title should not exceed one line.
Description	String	To get/set the bookmark's description. You can use multiple lines for the description.
SystemTime	Date	To get/set the bookmark's system time. The system time contains the absolute date and time when the bookmark was created.

**Methods** -

**Related topics**

References

[Class Description \(Bookmarks\)](#)..... 23

## Bookmarks (Collection)

**Where to go from here**

**Information in this section**

<a href="#">Class Description (Bookmarks)</a> .....	23
To provide a collection of the measurement's bookmarks.	
<a href="#">Add (Bookmarks)</a> .....	24
To add a bookmark to the measurement.	
<a href="#">Item (Bookmarks)</a> .....	24
To get a bookmark by index.	
<a href="#">Remove (Bookmarks)</a> .....	25
To remove a bookmark from the measurement.	

<a href="#">RemoveAll (Bookmarks).....</a>	<a href="#">26</a>
To remove all bookmarks from the measurement.	

## Class Description (Bookmarks)

**Syntax** No direct creation

**Purpose** To provide a collection of the measurement's bookmarks.

**Description** Bookmarks do not have a unique key. They are accessed only by their index in the collection.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of bookmarks in a measurement. (read-only)

**Methods** The following list shows the methods of this class.

- [Add \(Bookmarks\)](#) on page 24
- [Item \(Bookmarks\)](#) on page 24
- [Remove \(Bookmarks\)](#) on page 25
- [RemoveAll \(Bookmarks\)](#) on page 26

**Related topics**

References

<a href="#">Bookmark.....</a>	<a href="#">21</a>
<a href="#">Measurement.....</a>	<a href="#">36</a>

## Add (Bookmarks)

**Class** [Bookmarks \(Collection\)](#) on page 22

**Syntax** `RetVal = OBJ.Add(Timestamp ,Type ,Title [ ,Description=""])`

**Purpose** To add a bookmark to the measurement.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Timestamp	Float	Contains the time stamp value.
Type	<a href="#">BookmarkTypeConstants</a> on page 94	Contains the value of the bookmark type.
Title	String	Contains the bookmark title.
Description	String	Contains a description of the bookmark (default = "").

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">Bookmark</a> on page 21	Contains the bookmark object.

**Related topics**

References

[Bookmarks \(Collection\)](#)..... 22

## Item (Bookmarks)

**Class** [Bookmarks \(Collection\)](#)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get a bookmark by index.



**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of a Bookmark object within the Bookmarks collection.

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">Bookmark</a> on page 21	Contains the bookmark object.

**Related topics****References**

[Bookmarks \(Collection\)](#)..... 22

## Remove (Bookmarks)

**Class**

Bookmarks (Collection)

**Syntax**

```
OBJ . Remove ( Index )
```

**Purpose**

To remove a bookmark from the measurement.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of a Bookmark object within the Bookmark collection.

**Return value**

—

**Related topics****References**

[Bookmarks \(Collection\)](#)..... 22

## RemoveAll (Bookmarks)

---

<b>Class</b>	Bookmarks (Collection)
--------------	------------------------

---

<b>Syntax</b>	<code>OBJ.RemoveAll()</code>
---------------	------------------------------

---

<b>Purpose</b>	To remove all bookmarks from the measurement.
----------------	---

---

<b>Parameters</b>	—
-------------------	---

---

<b>Return value</b>	—
---------------------	---

---

<b>Related topics</b>	References
-----------------------	------------

<a href="#">Bookmarks (Collection).....</a>	<a href="#">22</a>
---	--------------------

## DescriptionCategories (Collection)

---

<b>Where to go from here</b>	<b>Information in this section</b>
------------------------------	------------------------------------

<a href="#">Class Description (DescriptionCategories).....</a>	<a href="#">26</a>
To provide a collection of the measurement's description categories.	
<a href="#">Item (DescriptionCategories).....</a>	<a href="#">27</a>
To get a description category by index.	

## Class Description (DescriptionCategories)

---

<b>Purpose</b>	To provide a collection of the measurement's description categories.
----------------	--

**Description** The description categories do not have to be created or removed. They are created automatically with the measurement.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of description categories in the measurement. (read-only)

**Methods** The following list shows the methods of this class.

- [Item \(DescriptionCategories\)](#) on page 27

#### Related topics

#### References

<a href="#">DescriptionCategory.....</a>	<a href="#">28</a>
<a href="#">Measurement.....</a>	<a href="#">36</a>

## Item (DescriptionCategories)

**Class** DescriptionCategories (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get a description category by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the DescriptionCategory object within the DescriptionCategories collection. The index can be either a number or a string that contains the name of the description category (for example, "General", "Recording").

**Return value**

The method returns a value of the following type:

Type	Description
IDispatch	Contains the DescriptionCategory object.

**Related topics****References**

[DescriptionCategories \(Collection\)](#)..... 26

## DescriptionCategory

### Class Description (DescriptionCategory)

**Purpose**

To access a description category of the measurement.

**Description**

This class serves as a base class for all DescriptionCategory objects. When using the Measurement Data API, you work with the objects derived from this class.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Name	String	To get the name of the DescriptionCategory. This attribute can be used as an access key for the DescriptionCategories collection. (read-only)

**Methods**

—

**Derived classes**

- [FileDescription](#) on page 29
- [GeneralDescription](#) on page 33
- [MDFDescription](#) on page 35

- [MeasurementDescription](#) on page 42
- [RecordingDescription](#) on page 60

## Related topics

## References

[DescriptionCategories \(Collection\)](#)..... 26

# FileDescription

## Where to go from here

## Information in this section

[Class Description \(FileDescription\)](#)..... 29  
To access the File description category.

## Information in other sections

[DescriptionCategory](#)..... 28  
To access a description category of the measurement.

## Class Description (FileDescription)

### Purpose

To access the FileDescription category.

### Description

This description category is available only if the measurement has been loaded from a file. The attributes of this category cannot be changed. They are filled automatically when the measurement is created.

### Attributes

The following table shows the attributes of this class.

Attribute	Type	Purpose
PathName	String	To get the absolute path of the measurement file. (read-only)
DateTime	Date	To get the modify date of the measurement file. (read-only)
Size	Float	To get the size of the measurement file. (read-only)

**Methods**

—

**Base class**[DescriptionCategory](#) on page 28

## FormatOption

### Class Description (FormatOption)

**Purpose**

To access a file format option of the measurement.

**Description**

This class serves as a base class for FormatOption objects.

When using the Measurement Data API, you deal only with objects derived from this class.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Name	String	To get the file format option's name. This attribute can be used as an access key for the FormatOptions collection. (read-only)

**Methods**

—

**Derived classes**[MDFFormatOption](#) on page 36**Related topics****References**[FormatOptions \(Collection\)](#)..... 31

## FormatOptions (Collection)

### Where to go from here

### Information in this section

<a href="#">Class Description (FormatOptions).....</a>	<a href="#">31</a>
To provide a collection of the measurement's file format options.	
<a href="#">Item (FormatOptions).....</a>	<a href="#">32</a>
To get a format option by index.	

### Information in other sections

<a href="#">Measurements (Collection).....</a>	<a href="#">43</a>
To provide a collection of all measurements loaded.	
<a href="#">FormatOption.....</a>	<a href="#">30</a>
To access a file format option of the measurement.	

## Class Description (FormatOptions)

### Purpose

To provide a collection of the measurement's file format options.

### Attributes

The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of format options. (read-only)

### Methods

The following list shows the methods of this class.

- [Item \(FormatOptions\)](#) on page 32

### Related topics

### References

<a href="#">FormatOption.....</a>	<a href="#">30</a>
<a href="#">Measurements (Collection).....</a>	<a href="#">43</a>

## Item (FormatOptions)

**Class** FormatOptions (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get a format option by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the FormatOption object within the FormatOptions collection. The index can be either a number or a string that contains the name of the format option.

**Return value** The method returns a value of the following type:

Type	Description
IDispatch	Contains the FormatOption object.

**Related topics**

**References**

[FormatOptions \(Collection\).....31](#)



# FormulaScaling

## Class Description (FormulaScaling)

**Purpose** To access a formula-based scaling of the measurement.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Formula	String	To get/set the formula for a formula-based scaling.

**Methods** –

**Base class** [ScalarScaling](#) on page 61

**Related topics**

References

[Scaling..... 62](#)

# GeneralDescription

## Class Description (GeneralDescription)

**Purpose** To access the GeneralDescription category.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
User	String	To get/set the ControlDesk user (or system user) who created the measurement.
DateTime	Date	To get/set the date and time when the recording was started.

Attribute	Type	Purpose
Origin	String	To get/set the original state of the measurement: ECU Image file and variable description – device – data sets.
Description	String	To get/set the measurement's description. You can use multiple lines for the description.

---

**Methods**

–

---

**Base class**[DescriptionCategory](#) on page 28

## LinearScaling

### Class Description (LinearScaling)

---

**Purpose**

To access a linear scaling of the measurement.

---

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Factor	Float	To get/set the factor for a linear scaling. The value of the factor must be positive. <div><b>Note</b> The factor of the linear scaling does not support negative values. In this case, use a rational function scaling.</div>
Offset	Float	To get/set the offset for a linear scaling.

---

**Methods**

–

---

**Base class** [ScalarScaling](#) on page 61

---

**Related topics**

**References**

[Scaling.....](#) 62

## MDFDescription

### Class Description (MDFDescription)

---

**Purpose** To access the MDF description category.

---

**Description** The MDFDescription class is available for measurements that originate from MDF files. The class contains the attributes specific to the MDF file format.

---

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Department	String	To get/set the Department property.
Project	String	To get/set the Project property.
MeasurementObject	String	To get/set the MeasurementObject property.

---

**Methods** –

---

**Base class** [DescriptionCategory](#) on page 28

# MDFFormatOption

## Class Description (MDFFormatOption)

**Purpose** To access the MDF file format option.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
VariableNameStorage	<a href="#">MDFVariableNameStorageConstants</a> on page 95	To get/set the storage option for the name of an MDF variable. This option is useful only if you use the exported MF4 in an external program that does not require path and/or block group information.

**Methods** —

**Base class** [FormatOption](#) on page 30

### Related topics

### References

<a href="#">FormatOption</a> .....	30
<a href="#">FormatOptions (Collection)</a> .....	31

## Measurement

### Where to go from here

### Information in this section

<a href="#">Class Description (Measurement)</a> .....	37
To access a measurement.	
<a href="#">Export (Measurement)</a> .....	38
To export the measurement to a new file.	

<a href="#">GetDeviceOptions (Measurement)</a> .....	39
To get the current MeasurementDeviceOptions setting.	
<a href="#">Save (Measurement)</a> .....	39
To save the measurement file.	
<a href="#">SetDeviceOptions (Measurement)</a> .....	40
To set the current MeasurementDeviceOptions settings.	
<a href="#">SetSection (Measurement)</a> .....	41
To set a time section to be loaded.	

## Class Description (Measurement)

**Purpose** To access a measurement.

**Attributes** The following table shows the attributes of this interface.

Attribute	Type	Purpose
Bookmarks	<a href="#">Bookmarks (Collection)</a> on page 22	To get a collection of the measurement's bookmarks. (read-only)
DescriptionCategories	<a href="#">DescriptionCategories (Collection)</a> on page 26	To get a collection of the measurement's description categories. (read-only)
FilePath	String	To get the absolute path name. (read-only)
Name	String	To get the measurement's unique name. It can be used as an index to access the collection of measurements. (read-only)
RevisionNumber	Int	To get the revision number. (read-only)
RevisionString	String	To get the revision string. (read-only)
Scalings	<a href="#">Scalings (Collection)</a> on page 63	To get a collection of the measurement's scalings. (read-only)
SectionStart	Float	To get the time stamp of the start section. (read-only)
SectionStop	Float	To get the time stamp of the stop section. (read-only)
SectionOptions	Int	To get the section flags. Refer to SetSection (Measurement). (read-only)
Signals	<a href="#">Signals (Collection)</a> on page 71	To get a collection of the measurement's signals. (read-only)
XAxes	<a href="#">XAxes (Collection)</a> on page 89	To get a collection of the measurement's x-axes. (read-only)

**Methods**

The following list shows the methods of this class.

- [Export \(Measurement\)](#) on page 38
- [GetDeviceOptions \(Measurement\)](#) on page 39
- [Save \(Measurement\)](#) on page 39
- [SetDeviceOptions \(Measurement\)](#) on page 40
- [SetSection \(Measurement\)](#) on page 41

**Related topics****References**

<a href="#">Measurements (Collection)</a> .....	43
<a href="#">SetSection (Measurement)</a> .....	41

## Export (Measurement)

**Class**

Measurement

**Syntax**

```
OBJ.Export(ExportFilePath)
```

**Purpose**

To export the measurement to a new file.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
ExportFilePath	String	Contains the file path of the export file.

**Return value**

—

**Related topics****References**

<a href="#">Measurement</a> .....	36
-----------------------------------	----

## GetDeviceOptions (Measurement)

**Class** Measurement

**Syntax** `RetVal = OBJ.GetDeviceOptions(Device)`

**Purpose** To get the current MeasurementDeviceOptions setting.

**Description** Indicates whether the platform's/device's display identifier is used for the output in graphical user interfaces (GUIs).

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Device	String	Contains the platform/device name.

**Return value** The method returns an object of the following type:

Type	Description
Int	Contains the value of the MeasurementDeviceOptions.

**Related topics**

References

Measurement.....	36
MeasurementDeviceOptions.....	96

## Save (Measurement)

**Class** Measurement

**Syntax** `OBJ.Save()`

**Purpose** To save the measurement file.

---

**Description** If the measurement was created with the Add method, it has no file path. In this case you cannot save the measurement this way. Use the Export (Measurement) method instead.

---

**Parameters** –

---

**Return value** –

---

**Related topics** **References**

Export (Measurement).....	38
Measurement.....	36

## SetDeviceOptions (Measurement)

---

**Class** Measurement

---

**Syntax** `OBJ.SetDeviceOptions(Device ,Options)`

---

**Purpose** To set the current MeasurementDeviceOptions settings.

---

**Description** Specifies whether the platform's/device's display identifier is used for the output in graphical user interfaces (GUIs).

---

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Device	String	Contains the platform/device name.
Options	Int	Contains the value of the MeasurementDeviceOptions.



**Return value**

—

**Related topics****References**

Measurement.....	36
MeasurementDeviceOptions.....	96

## SetSection (Measurement)

**Class**

Measurement

**Syntax**

```
OBJ.SetSection(StartTimestamp ,StopTimestamp [ ,SectionOptions=msIncludeStartTimestamp | msIncludeStopTimestamp])
```

**Purpose**

To set a time section to be loaded.

**Description**

You can define a time section to load only a part of a measurement. For example, you can load only x-axis and signal values that have been measured between the 5th and the 10th second of a measurement.

You can also use the time stamps of bookmarks to define a section.

The time section must be specified before you load any measurement data. SetSection enables you to specify the time section if you have loaded the measurement without data.

SetSection leads to an error if measurement data was already loaded.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
StartTimestamp	Float	Contains the start time stamp in seconds. If the start and stop time stamp are 0, all the values are loaded.
StopTimestamp	Float	Contains the stop time stamp in seconds. If the start and stop time stamp are 0, all the values are loaded.
SectionOptions	Int	Contains the value for the SectionOptions (default = msIncludeStartTimestamp   msIncludeStopTimestamp). If time stamps of samples do exactly match a section boundary, the SectionOptions specify whether these samples are included (default) or excluded from the measurement data.

**Return value**

—

**Related topics****References**[Measurement.....](#) 36

## MeasurementDescription

### Class Description (MeasurementDescription)

**Purpose**

To access the Measurement description category.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
XAxisOffset	Float	To get/set an offset for the x-axis. The x-axis offset is used to shift the x-values of the entire measurement.
Length	Float	To get/set the duration of the recording in seconds.
StartTimestamp	Float	To get/set the StartTimestamp property. This attribute contains the start of the recording in seconds relative to the start of the measurement.
StopTimestamp	Float	To get/set the StopTimestamp property. This attribute contains the stop of the recording in seconds relative to the start of the measurement.

**Methods**

—

**Base class**[DescriptionCategory](#) on page 28

# Measurements (Collection)

## Where to go from here

## Information in this section

<a href="#">Class Description (Measurements)</a> .....	43
To provide a collection of all measurements loaded.	
<a href="#">Add (Measurements)</a> .....	44
To create a new measurement.	
<a href="#">Item (Measurements)</a> .....	45
To get a measurement by index.	
<a href="#">Load (Measurements)</a> .....	46
To load a measurement from a file or from the flash memory of dSPACE real-time hardware.	
<a href="#">Remove (Measurements)</a> .....	47
To remove a measurement.	
<a href="#">RemoveAll (Measurements)</a> .....	48
To remove all measurements.	
<a href="#">SaveAll (Measurements)</a> .....	49
To save all measurements.	

## Class Description (Measurements)

**Purpose** To provide a collection of all measurements loaded.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of loaded measurements. (read-only)
FormatOptions	<a href="#">FormatOptions (Collection)</a> on page 31	To get a collection of file format options. (read-only)

## Methods

The following list shows the methods of this class.

- [Add \(Measurements\)](#) on page 44
- [Item \(Measurements\)](#) on page 45
- [Load \(Measurements\)](#) on page 46
- [Remove \(Measurements\)](#) on page 47

- [RemoveAll \(Measurements\)](#) on page 48
- [SaveAll \(Measurements\)](#) on page 49

**Related topics****References**

[Measurement](#)..... 36

## Add (Measurements)

**Class** Measurements (Collection)

**Syntax** `RetVal = OBJ.Add(Name)`

**Purpose** To create a new measurement.

**Note**

In some cases you may not be able to create a new measurement with the desired name.

After a measurement has been removed from the global collection of measurements, it may still exist because it is referenced by other objects. These can be Measurement objects or any other objects from the removed measurement like Signals, XAxes, Scalings, or Bookmarks. You must destroy these objects explicitly (using `del`) if you want to ensure that the name of the removed measurement can be used again.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Name	String	Contains the name of the new measurement.

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">Measurement</a> on page 36	Contains the new measurement.

---

**Related topics****References**

[Measurements \(Collection\)..... 43](#)

## Item (Measurements)

---

**Class**

Measurements (Collection)

---

**Syntax**

```
RetVal = OBJ.Item(Index)
```

---

**Purpose**

To get a measurement by index.

---

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the Measurement object within the Measurements collection. The index can either be a number or a string that contains the name of the measurement.

---

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">Measurement</a> on page 36	Contains the Measurement object.

---

**Related topics****References**

[Measurements \(Collection\)..... 43](#)

## Load (Measurements)

**Class** Measurements (Collection)

### Syntax

```
RetVal = OBJ.Load(MeasurementFilePath \
    [ ,Options=mlWithoutData [ ,StartTimestamp=0 \
    [ ,StopTimestamp=0 \
    [ ,SectionOptions=msIncludeStartTimestamp | \
    msIncludeStopTimestamp \
    [ ,Name="" ]]]])
```

### Purpose

To load a measurement from a file or from the flash memory of dSPACE real-time hardware.

#### Note

In some cases you may not be able to create a new measurement with the desired name.

After a measurement has been removed from the global collection of measurements, it may still exist because it is referenced by other objects. These can be Measurement objects or any other objects from the removed measurement like Signals, XAxes, Scalings, or Bookmarks. You must destroy these objects explicitly (using `del`) if you want to ensure that the name of the removed measurement can be used again.

### Parameters

The method provides the following parameters:

Parameter	Type	Description
MeasurementFilePath	String	Contains the file path of the measurement file. Instead of a file name you can use the name of registered dSPACE real-time hardware. In this case, the content of the board's flight recording flash memory is uploaded. With the Export (Measurement) method you can save it as a measurement file.
Options	<a href="#">MeasurementLoadingOptions</a> on page 97	Contains the value of the MeasurementLoadingOptions (default = mlWithOutData).
StartTimestamp	Float	Contains the start time stamp (default = 0). Start and stop time stamp define a time section in the measurement from which values are loaded. If the start and stop time stamp are 0, all the values are loaded.

Parameter	Type	Description
StopTimestamp	Float	Contains the stop time stamp (default = 0). Start and stop time stamp define a time section in the measurement from which values are loaded. If the start and stop time stamp are 0, all the values are loaded.
SectionOptions	Int	Contains the value of the MeasurementRemovalOptions (default = msIncludeStartTimestamp   msIncludeStopTimestamp). If time stamps of samples do exactly match a section boundary, the SectionOptions specify whether these samples are included (default) or excluded from the measurement data.
Name	String	Contains the name of the measurement (default = ""). If the measurement's name is an empty string, the measurement is named after the measurement file.

**Return value**

The method returns an object of the following type:

Type	Description
<a href="#">Measurement</a> on page 36	Contains the measurement object.

**Related topics****References**

<a href="#">Export (Measurement)</a> .....	38
<a href="#">Measurement</a> .....	36
<a href="#">MeasurementRemovalOptions</a> .....	97
<a href="#">Measurements (Collection)</a> .....	43

## Remove (Measurements)

**Class** Measurements (Collection)

**Syntax** `OBJ.Remove(Index [, Options = mrIgnore])`

**Purpose** To remove a measurement.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the measurement. The index can either be a number or a string that contains the measurement's name.
Options	<a href="#">MeasurementRemovalOptions</a> on page 97	Contains the value of the MeasurementRemovalOptions (default = mrlgnore).

**Return value**

—

**Related topics****References**

<a href="#">Measurement.....</a>	<a href="#">36</a>
<a href="#">Measurements (Collection).....</a>	<a href="#">43</a>

## RemoveAll (Measurements)

**Class**

Measurements (Collection)

**Syntax**`OBJ.RemoveAll()`**Purpose**

To remove all measurements.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Options	<a href="#">MeasurementRemovalOptions</a> on page 97	Contains the value of the MeasurementRemovalOptions (default = mrlgnore).



<b>Return value</b>	—
<b>Related topics</b>	<b>References</b>
	<a href="#">Measurement.....</a> 36
	<a href="#">Measurements (Collection).....</a> 43

## SaveAll (Measurements)

<b>Class</b>	Measurements (Collection)
<b>Syntax</b>	<code>OBJ.SaveAll()</code>
<b>Purpose</b>	To save all measurements.
<b>Parameters</b>	—
<b>Return value</b>	—
<b>Related topics</b>	<b>References</b>
	<a href="#">Measurement.....</a> 36
	<a href="#">Measurements (Collection).....</a> 43

## MultiScaling

### Class Description (MultiScaling)

<b>Purpose</b>	To access a table that maps numerical source value ranges to scalings.
----------------	--

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Table	<a href="#">MultiScalingTableEntries (Collection)</a> on page 50	To get a collection representing a multiscaling table. (read-only)

**Methods**

-

## MultiScalingTableEntries (Collection)

**Where to go from here****Information in this section**

<a href="#">Class Description (MultiScalingTableEntries)</a> .....	50
To provide a collection representing all entries in a multiscaling table, which maps ranges of numerical source values to different scalings.	
<a href="#">Add (MultiScalingTableEntries)</a> .....	51
To add an entry to the multiscaling table.	
<a href="#">Item (MultiScalingTableEntries)</a> .....	52
To get an entry by index.	
<a href="#">Remove (MultiScalingTableEntries)</a> .....	53
To remove an entry from the table.	

## Class Description (MultiScalingTableEntries)

**Purpose**

To provide a collection representing all entries in a multiscaling table, which maps ranges of numerical source values to different scalings.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of entries (read-only).
DefaultValue	Variant	To get or set the default value used for converting numerical values that are not included in the table.

Attribute	Type	Purpose
		<ul style="list-style-type: none"> <li>▪ If the source value is not included in the table, the default value is returned as the result of the conversion.</li> <li>▪ If no default value is set, <b>None</b> is returned as the result of the conversion.</li> </ul> <p>The default value of a multiscaling table is stored in the measurement data file as string.<sup>1)</sup></p>

<sup>1)</sup> Default values for multiscaling tables are not supported by the CSV and MAT measurement data file formats. In both formats, the default value is replaced by a quiet NaN (QNaN).

## Methods

The following list shows the methods of this class.

- [Add \(MultiScalingTableEntries\)](#) on page 51
- [Item \(MultiScalingTableEntries\)](#) on page 52
- [Remove \(MultiScalingTableEntries\)](#) on page 53

## Related topics

### References

<a href="#">MultiScaling.....</a>	<a href="#">49</a>
<a href="#">MultiScalingTableEntry.....</a>	<a href="#">53</a>

# Add (MultiScalingTableEntries)

## Class

MultiScalingTableEntries (Collection)

## Syntax

```
RetVal = OBJ.Add(MinVal ,MaxVal , OutVal)
```

## Purpose

To add an entry to the multiscaling table.

## Parameters

The method provides the following parameters:

Parameter	Type	Description
MinVal	Float	Contains the minimum value of the input range (source value).
MaxVal	Float	Contains the maximum value of the input range (source value).
OutVal	<a href="#">Scaling</a> on page 62	Contains the scaling that is used to convert the source value. All scaling types are allowed except for multiscalings.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">MultiScalingTableEntry</a> on page 53	Contains the MultiscalingTableEntry object.

## Related topics

## References

[MultiScalingTableEntries \(Collection\)](#)..... 50

# Item (MultiScalingTableEntries)

**Class** MultiScalingTableEntries (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get an entry by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the MultiScalingTableEntry object.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">MultiScalingTableEntry</a> on page 53	Contains the MultiScalingTableEntry object.

## Related topics

## References

[MultiScalingTableEntries \(Collection\)](#)..... 50

## Remove (MultiScalingTableEntries)

**Class** MultiScalingTableEntries (Collection)

**Syntax** `OBJ.Remove(Entry)`

**Purpose** To remove an entry from the table.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the MultiScalingTableEntry object within the MultiScalingTableEntries collection.

### Note

At least one entry has to remain in the table when you modify an existing multiscaling.

**Return value** —

### Related topics

### References

[MultiScalingTableEntries \(Collection\)..... 50](#)

## MultiScalingTableEntry

### Where to go from here

### Information in this section

[Class Description \(MultiScalingTableEntry\)..... 54](#)  
To access an entry of a multiscaling table, which maps ranges of numerical source values to scalings.

Information in other sections

<a href="#">MultiScalingTableEntries (Collection)</a> .....	50
To provide a collection representing all entries in a multiscaling table, which maps ranges of numerical source values to different scalings.	

# Class Description (MultiScalingTableEntry)

**Purpose** To access an entry of a multiscaling table, which maps ranges of numerical source values to scalings.

<b>Attributes</b>		The following table shows the attributes of this class.
Attribute	Type	Purpose
MinVal	Float	To get or set the minimum value of the input range (source value).
MaxVal	Float	To get or set the maximum value of the input range (source value).
OutVal	Scaling	To get or set the scaling that is used to convert the source value.

**Methods** —

<b>Related topics</b>	<b>References</b>
	<a href="#">MultiScalingTableEntries (Collection)</a> ..... 50
	<a href="#">Scaling</a> ..... 62

# NumericTableEntries (Collection)

<b>Where to go from here</b>	<b>Information in this section</b>
	<a href="#">Class Description (NumericTableEntries)</a> ..... 55
	To provide a collection representing all entries in a conversion table for converting numerical values (CompuTab).
	<a href="#">Add (NumericTableEntries)</a> ..... 56
	To add an entry to the conversion table (CompuTab).

<a href="#">Item (NumericTableEntries).....</a>	<a href="#">56</a>
To get an entry by index.	
<a href="#">Remove (NumericTableEntries).....</a>	<a href="#">57</a>
To remove an entry from the table.	

## Class Description (NumericTableEntries)

**Purpose** To provide a collection representing all entries in a conversion table for converting numerical values (CompuTab).

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of entries. (read-only)
DefaultValue	Variant	<p>To get or set the default value used for converting numerical values that are not included in the table.</p> <ul style="list-style-type: none"> <li>▪ If the source value is not included in the table, the default value is returned as the result of the conversion.</li> <li>▪ If no default value is set, <b>None</b> is returned as the result of the conversion.</li> </ul> <p>The default value of a numeric conversion table (CompuTab) is stored in the measurement data file as string.<sup>1)</sup></p>

<sup>1)</sup> Default values for numeric conversion tables are not supported by the CSV and MAT measurement data file formats. In the CSV and MAT file formats, the default value is replaced by a quiet NaN (QNaN).

**Methods** The following list shows the methods of this class.

- [Add \(NumericTableEntries\)](#) on page 56
- [Item \(NumericTableEntries\)](#) on page 56
- [Remove \(NumericTableEntries\)](#) on page 57

### Related topics

#### References

<a href="#">NumericTableEntry.....</a>	<a href="#">58</a>
<a href="#">NumericTableScaling.....</a>	<a href="#">59</a>

## Add (NumericTableEntries)

**Class** NumericTableEntries (Collection)

**Syntax** `RetVal = OBJ.Add(InVal ,OutVal)`

**Purpose** To add an entry to the conversion table (CompuTab).

**Parameters** The method provides the following parameters:

Parameter	Type	Description
InVal	Double	Contains the input value (source value) of the NumericTableEntry object.
OutVal	Float	Contains the output value (converted value) of the NumericTableEntry object.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">NumericTableEntry</a> on page 58	Contains the NumericTableEntry object.

**Related topics**

**References**

[NumericTableEntries \(Collection\)](#)..... 54

## Item (NumericTableEntries)

**Class** NumericTableEntries (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get an entry by index.



**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the NumericTableEntry object within the NumericTableEntries collection.

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">NumericTableEntry</a> on page 58	Contains the NumericTableEntry object.

**Related topics****References**

[NumericTableEntries \(Collection\)](#)..... 54

## Remove (NumericTableEntries)

**Class**

NumericTableEntries (Collection)

**Syntax**

```
OBJ . Remove (Entry)
```

**Purpose**

To remove an entry from the table.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the NumericTableEntry object within the NumericTableEntries collection.

**Note**

At least one entry has to remain in the table if an existing scaling is modified.

Return value	—
Related topics	References
	NumericTableEntries (Collection)..... 54

# NumericTableEntry

## Class Description (NumericTableEntry)

Purpose	To access an entry of a conversion table for converting numerical values (CompuTab).										
Attributes	The following table shows the attributes of this class.										
	<table><tr><th>Attribute</th><th>Type</th><th>Purpose</th></tr><tr><td>InVal</td><td>Double</td><td>To get/set the input value (source value) of the NumericTableEntry object.</td></tr><tr><td>OutVal</td><td>Float</td><td>To get/set the output value (converted value) of the NumericTableEntry object.</td></tr></table>		Attribute	Type	Purpose	InVal	Double	To get/set the input value (source value) of the NumericTableEntry object.	OutVal	Float	To get/set the output value (converted value) of the NumericTableEntry object.
Attribute	Type	Purpose									
InVal	Double	To get/set the input value (source value) of the NumericTableEntry object.									
OutVal	Float	To get/set the output value (converted value) of the NumericTableEntry object.									
Methods	—										
Related topics	References										
	NumericTableEntries (Collection)..... 54										

# NumericTableScaling

## Class Description (NumericTableScaling)

**Purpose** To access a collection representing a conversion table for converting numerical values (CompuTab).

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Table	<a href="#">NumericTableEntries (Collection)</a> on page 54	To get a collection representing an interpolatable or a non-interpolatable scaling table. (read-only)

**Methods** —

**Base class** [ScalarScaling](#) on page 61

### Related topics

#### References

[Scaling.....](#) 62

# RationalFunctionScaling

## Class Description (RationalFunctionScaling)

**Purpose** To access a rational function scaling of the measurement.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Coefficients	Variant	To get/set the coefficients for a scaling based on a rational formula.The Variant must contain exactly six 64-bit floating point values for the six coefficients of the rational function.

**Methods**

—

**Base class**

[ScalarScaling](#) on page 61

**Related topics****References**

[Scaling..... 62](#)

## RecordingDescription

### Class Description (RecordingDescription)

**Purpose**

To access the Recording description category.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
StartCondition	String	To get/set the description of the recording's start condition (for example, none, start trigger, pretrigger).
StopCondition	String	To get/set the description of the recording's stop condition (for example, endless, time limit, stop trigger, posttrigger).

**Methods**

—

**Base class**

[DescriptionCategory](#) on page 28

# ScalarScaling

## Class Description (ScalarScaling)

---

**Purpose** To access a scalar scaling of the measurement.

---

**Description** This class only serves as a base class for all scalar scalings. When using the Measurement Data API, you deal only with objects derived from this class.

---

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Format	String	To get/set the scaling's format. The format string describes how a numerical value is displayed, for example "%8.3f".
Unit	String	To get/set the scaling's physical unit, for example "V".

---

**Methods** —

---

**Base class** [Scaling](#) on page 62

---

**Derived classes**

- [FormulaScaling](#) on page 33
- [LinearScaling](#) on page 34
- [RationalFunctionScaling](#) on page 59
- [NumericTableScaling](#) on page 59

# Scaling

## Class Description (Scaling)

**Purpose** To access a scaling of the measurement.

**Description** This class only serves as a base class for all scalings. When using the Measurement Data API, you deal only with objects derived from this class.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Name	String	To get the scaling's name. It can be used as an index to access the collection of scalings. (read-only)
Type	<a href="#">ScalingTypeConstants</a> on page 98	To get the scaling's type. (read-only)
Description	String	To get/set the scaling's description.

**Derived classes**

- [ScalarScaling](#) on page 61
- [VerbalTableRangeScaling](#) on page 88
- [VerbalTableScaling](#) on page 88

### Related topics

#### References

<a href="#">Scalings (Collection)</a> .....	63
<a href="#">Signal</a> .....	68

# Scalings (Collection)

## Where to go from here

## Information in this section

<a href="#">Class Description (Scalings)</a> .....	63
To provide a collection of the measurement's scalings.	
<a href="#">Add (Scalings)</a> .....	64
To create a scaling for the given type.	
<a href="#">CreateTable (Scalings)</a> .....	65
To create a table for the given scaling type.	
<a href="#">Item (Scalings)</a> .....	66
To get a scaling by index.	
<a href="#">Remove (Scalings)</a> .....	66
To remove a scaling from the measurement.	
<a href="#">RemoveAll (Scalings)</a> .....	67
To remove all scalings from the measurement.	

## Class Description (Scalings)

### Purpose

To provide a collection of the measurement's scalings.

### Attributes

The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of scalings in the measurement. (read-only)

### Methods

The following list shows the methods of this class.

- [Add \(Scalings\)](#) on page 64
- [CreateTable \(Scalings\)](#) on page 65
- [Item \(Scalings\)](#) on page 66

- [Remove \(Scalings\)](#) on page 66
- [RemoveAll \(Scalings\)](#) on page 67

**Related topics****References**

<a href="#">Measurement</a> .....	36
<a href="#">Scaling</a> .....	62

## Add (Scalings)

**Class**

Scalings (Collection)

**Syntax**

```
RetVal = OBJ.Add(Name, Type[, Table])
```

**Purpose**

To create a scaling for the given type.

**Description**

This method adds a scaling to the measurement and performs a consistency check on the parameters.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Name	String	Contains the name of the new Scaling object. The name can be used as an access key for the Scalings collection.
Type	<a href="#">ScalingTypeConstants</a> on page 98	Contains the value of the ScalingTypeConstant.
Table	Table	Contains the Table created by the CreateTable method. The table is required only for those scalings that are based on a table. The table must be filled with at least one entry.

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">Scaling</a> on page 62	Contains the Scaling object.



**Related topics****References**[Scalings \(Collection\)..... 63](#)

## CreateTable (Scalings)

**Class**

Scalings (Collection)

**Syntax**

```
RetVal = OBJ.CreateTable(Type)
```

**Purpose**

To create a table for the given scaling type.

**Description**

The table must be filled with at least one entry. Then it can be added to a measurement.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Type	ScalingTypeConstants	Contains the value of the ScalingTypeConstant.

**Return value**

The method returns a value of the following type:

Type	Description
Table	Contains the Table object.

**Related topics****References**[Scalings \(Collection\)..... 63](#)

## Item (Scalings)

**Class** Scalings (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get a scaling by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the Scaling object within the Scalings collection. The index can be either a number or a string that contains the scaling's name. Note that this index can change if a signal is added to or removed from the measurement.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">Scaling</a> on page 62	Contains the Scaling object.

**Related topics**

**References**

[Scalings \(Collection\)..... 63](#)

## Remove (Scalings)

**Class** Scalings (Collection)

**Syntax** `OBJ.Remove(Scaling)`

**Purpose** To remove a scaling from the measurement.

**Note**

You cannot remove a scaling that is referenced by a signal.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the Scaling object within the Scalings collection. The index can be either a number or a string that contains the scaling's name. <div><b>Note</b> Keep in mind that the index can change if a signal is added to or removed from the measurement.</div>

**Return value** —

**Related topics**

References

[Scalings \(Collection\)..... 63](#)

## RemoveAll (Scalings)

**Class** Scalings (Collection)

**Syntax** `OBJ.RemoveAll()`

**Purpose** To remove all scalings from the measurement.

**Note**

You cannot remove scalings that are referenced by a signal.

**Parameters**

—

**Return value**

—

**Related topics****References**[Scalings \(Collection\).....](#) 63

## Signal

**Where to go from here****Information in this section**[Class Description \(Signal\).....](#) 68

To access a signal of the measurement.

[LoadData \(Signal\).....](#) 70

To load the signal's data from the measurement file.

[UnloadData \(Signal\).....](#) 71

To remove the signal's data from the measurement.

## Class Description (Signal)

**Purpose**

To access a signal of the measurement.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Name	String	To get the signal's name. (read-only)
Data	Variant	To get the signal's data. (read-only) The data of curves and maps is divided into sub-lists in this order: x-axis, y-axis, function values.
XAxis	<a href="#">XAxis</a> on page 93	To get the signal's x-axis. (read-only)
Scaling	<a href="#">Scaling</a> on page 62	To get the signal's scaling. (read-only)
Device	String	To get the signal's platform/device. (read-only)

Attribute	Type	Purpose
Path	String	To get the signal's path. (read-only)
DisplayIdentifier	String	To get/set the signal's display identifier.
Description	String	To get/set the signal's description.
MinHard	Float	To get/set the signal's minimum hard limit.
MaxHard	Float	To get/set the signal's maximum hard limit.
MinWeak	Float	To get/set the signal's minimum weak limit.
MaxWeak	Float	To get/set the signal's maximum weak limit.
Type	<a href="#">SignalTypeConstants</a> on page 99	To get the signal's interface type. This attribute can be used to detect whether the signal has a source data attribute. (read-only)
Key	String	<p>To get the signal's key. (read-only)</p> <p>The signal key can be used to access the Signals collection. It consists of</p> <ul style="list-style-type: none"> <li>▪ The platform/device name (not case-sensitive)</li> <li>▪ The signal path (if present, case-sensitive)</li> <li>▪ The signal name (case-sensitive)</li> <li>▪ The raster name which is the name of the x-axis (case-sensitive)</li> </ul>
BitOffset	Int	To get the signal's bit offset. The bit offset and the number of bits show what bits were significant for data acquisition.
NumberOfBits	Int	To get the signal's number of bits. This attribute contains the bandwidth of the measurement signal.
VariableType	<a href="#">VariableTypeConstants</a> on page 100	To get the signal's variable type. (read-only)
Components	List of objects	<ul style="list-style-type: none"> <li>▪ If the signal is a curve or map: To get the axes objects. (read-only) <ul style="list-style-type: none"> <li>▪ One component for the axis of a curve</li> <li>▪ Two components for the two axes of a map (in this order: x-axis, y-axis)</li> </ul> </li> <li>▪ If the signal is an axis: To get the input quantity. (read-only) <ul style="list-style-type: none"> <li>▪ One component for the input quantity.</li> </ul> </li> </ul> <div> <b>Note</b>  The list is empty if the axis has no input quantity. </div>

## Methods

The following list shows the methods of this class:

- [LoadData \(Signal\)](#) on page 70
- [UnloadData \(Signal\)](#) on page 71

**Derived classes** [SignalWithSource](#) on page 79

**Related topics**

**References**

<a href="#">Scaling.....</a>	<a href="#">62</a>
<a href="#">Signals (Collection).....</a>	<a href="#">71</a>

# LoadData (Signal)

**Class** [Signal](#)

**Syntax** `OBJ.LoadData()`

**Purpose** To load the signal's data from the measurement file.

**Note**

You can use this method only for ASAM MDF 4.1.x files. Files of other formats are always loaded completely.

**Parameters** —

**Return value** —

**Related topics**

**References**

<a href="#">Signal.....</a>	<a href="#">68</a>
-----------------------------	--------------------

## UnloadData (Signal)

**Class** Signal

**Syntax** `OBJ.UnloadData()`

**Purpose** To remove the signal's data from the measurement.

### Note

You can use this method only for ASAM MDF 4.x files. Files of other formats are always loaded completely.

**Parameters** —

**Return value** —

### Related topics

#### References

[Signal](#)..... 68

## Signals (Collection)

### Where to go from here

### Information in this section

<a href="#">Class Description (Signals)</a> .....	72
To provide a collection of the measurement's signal objects.	
<a href="#">Add (Signals)</a> .....	73
To add a signal to the measurement.	
<a href="#">CreateAxisTemplate (Signals)</a> .....	74
To create an axis template for a fixed or standard axis of a curve or map.	
<a href="#">Item (Signals)</a> .....	75
To get a signal by index.	

<a href="#">LoadData (Signals)</a> .....	76
To load the data of the specified signals from the measurement file.	
<a href="#">Remove (Signals)</a> .....	76
To remove a signal from the measurement.	
<a href="#">RemoveAll (Signals)</a> .....	77
To remove all signals from the measurement.	
<a href="#">UnloadData (Signals)</a> .....	78
To remove the data of the specified signals from the measurement.	

## Class Description (Signals)

---

<b>Purpose</b>	To provide a collection of the measurement's signal objects.
----------------	--

---

<b>Attributes</b>	The following table shows the attributes of this class.
-------------------	---

Attribute	Type	Purpose
Count	Int	To get the number of signals. (read-only)

---

<b>Methods</b>	The following list shows the methods of this class.
----------------	---

- [Add \(Signals\)](#) on page 73
- [CreateAxisTemplate \(Signals\)](#) on page 74
- [Item \(Signals\)](#) on page 75
- [LoadData \(Signals\)](#) on page 76
- [Remove \(Signals\)](#) on page 76
- [RemoveAll \(Signals\)](#) on page 77
- [UnloadData \(Signals\)](#) on page 78

---

<b>Related topics</b>	References
-----------------------	------------

<a href="#">Measurement</a> .....	36
<a href="#">Signal</a> .....	68



## Add (Signals)

**Class** Signals (Collection)

**Syntax**

```
RetVal = OBJ.Add(Name ,DisplayIdentifier ,Device ,Path ,Data ,pXAxis ,pScaling [,
VariableType [,Components]])
```

**Purpose** To add a signal to the measurement.

**Description**

The signals have a reference to their x-axis and scaling objects. You must therefore add the x-axes and scalings first. Then you can pass them to the signal's **Add** method.

**Note**

- The x-axis used for a signal must have the same number of points (raster settings) and the same platform/device.
- Adding a signal can change the numeric indices of the measurement's x-axes and scalings in the corresponding collections.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Name	String	Contains the name of the Signal object.
DisplayIdentifier	String	Contains the display identifier of the Signal object.
Device	String	Contains the signal's platform/device.
Path	String	Contains the signal's path.
Data	Variant	Contains the signal's data.
pXAxis	<a href="#">XAxis</a> on page 93	Contains the signal's x-axis.
pScaling	<a href="#">Scaling</a> on page 62	Contains the signal's scaling.
VariableType	<a href="#">VariableTypeConstants</a> on page 100	Contains the signal's variable type. <ul style="list-style-type: none"> <li>▪ If <b>vtMeasurement</b> is selected as the variable type, but the values of an added signal are multidimensional, the variable type is set to <b>vtMeasurementArray</b> automatically.</li> <li>▪ If <b>vtParameter</b> is selected as variable type, but the values of an added signal are multi-dimensional, the variable type is set to <b>vtValueBlock</b> automatically.</li> </ul>
Components	List of objects	<ul style="list-style-type: none"> <li>▪ Contains the axes objects if the signal is a map or curve:<sup>1)</sup> <ul style="list-style-type: none"> <li>▪ One component for the axis of a curve</li> </ul> </li> </ul>

Parameter	Type	Description
		<ul style="list-style-type: none"> <li>Two components for the two axes of a map (in this order: x-axis, y-axis)<sup>2)</sup></li> <li>Contains the input quantity if the signal is an axis: <ul style="list-style-type: none"> <li>One component for the input quantity<sup>3)</sup></li> </ul> </li> </ul>

<sup>1)</sup> Depending on the axis type the components can either be axis templates or signals of the **vtCommonAxis** variable type. For more information on axis templates, refer to **AxisTemplate**.

<sup>2)</sup> A fixed axis must not be combined with another axis type.

<sup>3)</sup> The component must be a signal of the variable type **vtMeasurement** or **vtParameter**.

### Return value

The method returns a value of the following type:

Type	Description
<a href="#">Signal</a> on page 68	Contains the Signal object.

### Related topics

#### References

<a href="#">AxisTemplate</a> .....	20
<a href="#">Signals (Collection)</a> .....	71

## CreateAxisTemplate (Signals)

**Class** Signals (Collection)

**Syntax** `RetVal = OBJ.CreateAxisTemplate()`

**Purpose** To create an axis template for a fixed or standard axis of a curve or map.

**Description** An axis template stores the values, the scaling, and the input quantity (optional) of a standard or fixed axis of a look-up table (curve or map).

When you add a curve or a map to the signals collection, the axis templates are passed to the list of components of the signal's **Add** method in the following order: x-axis, y-axis.

**Parameters** —

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">AxisTemplate</a> on page 20	Contains the AxisTemplate object.

**Related topics****References**

<a href="#">Add (Signals)</a> .....	73
<a href="#">Class Description (AxisTemplate)</a> .....	20
<a href="#">Signals (Collection)</a> .....	71

## Item (Signals)

**Class**

Signals (Collection)

**Syntax**

```
RetVal = OBJ.Item(Index)
```

**Purpose**

To get a signal by index.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the signal. The index can be either a number or a string containing the signal key.

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">Signal</a> on page 68 - or - <a href="#">SignalWithSource</a> on page 79	Contains the Signal object.

**Related topics****References**

<a href="#">Signals (Collection)</a> .....	71
--	----

## LoadData (Signals)

**Class** Signals (Collection)

**Syntax** `OBJ.LoadData(Signals)`

**Purpose** To load the data of the specified signals from the measurement file.

**Note**

You can use this method only for ASAM MDF 4.1.x files. Files of other formats are always loaded completely.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Signals	Variant	Contains the signals to be loaded. The VARIANT must contain a list with the signals. In this list, the signals can be either specified by their numeric index in the collection, by their signal key, or by adding the Signal objects directly to the list. If this parameter is omitted, all the signals are loaded.

**Return value** —

**Related topics**

**References**

[Signals \(Collection\)..... 71](#)

## Remove (Signals)

**Class** Signals (Collection)

**Syntax** `OBJ.Remove(Index)`

---

**Purpose** To remove a signal from the measurement.

**Note**

Removing a signal can change the numeric indices of the measurement's x-axes and scalings in the corresponding collections.

---

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the signal. The index can either be a number or a string containing the signal key.

---

**Return value** —

---

**Related topics**

**References**

[Signals \(Collection\)..... 71](#)

## RemoveAll (Signals)

---

**Class** Scalings (Collection)

---

**Syntax** `OBJ.RemoveAll()`

---

**Purpose** To remove all signals from the measurement.

---

**Parameters** —

**Return value**

—

**Related topics****References**[Scalings \(Collection\)..... 63](#)

## UnloadData (Signals)

**Class**

Signals (Collection)

**Syntax**`OBJ.UnloadData(Signals)`**Purpose**

To remove the data of the specified signals from the measurement.

**Note**

You can use this method only for ASAM MDF 4.x files. Files of other formats are always loaded completely.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Signals	Variant	Contains the signals to be unloaded. The VARIANT must contain a list with the signals. In this list, the signals can either be specified by their numeric index in the collection, by their signal key, or by adding the Signal objects directly to the list.

**Return value**

—

**Related topics****References**[Signals \(Collection\)..... 71](#)

# SignalWithSource

## Class Description (SignalWithSource)

**Purpose** To access a signal of the measurement.

**Description** For SignalWithSource objects, the signal's Type attribute returns stWithSourceData.

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
SourceData	Variant	To get the signal's source data. (read-only)

**Methods** —

**Base class** [Signal](#) on page 68

## VerbalTableEntries (Collection)

**Where to go from here**

**Information in this section**

<a href="#">Class Description (VerbalTableEntries)</a> .....	80
To provide a collection representing all entries in a conversion table for converting numerical values into strings (CompuVTab).	
<a href="#">Add (VerbalTableEntries)</a> .....	80
To add an entry to the conversion table (CompuVTab).	
<a href="#">Item (VerbalTableEntries)</a> .....	81
To get an entry by index.	
<a href="#">Remove (VerbalTableEntries)</a> .....	82
To remove an entry from the table.	

## Class Description (VerbalTableEntries)

**Purpose** To provide a collection representing all entries in a conversion table for converting numerical values into strings (CompuVTab).

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of entries. (read-only)
DefaultValue	Variant	<p>To get or set the default value used for converting numerical values that are not included in the table.</p> <ul style="list-style-type: none"> <li>▪ If the source value is not included in the table, the default value is returned as the result of the conversion.</li> <li>▪ If no default value is set, <b>None</b> is returned as the result of the conversion.</li> </ul> <p>The default value of a verbal table (CompuVTab) is stored in the measurement data file as string.<sup>1)</sup></p>

<sup>1)</sup> Default values for verbal tables are not supported by the CSV and MAT measurement data file formats. In both formats, the default value is replaced by a quiet NaN (QNAN).

**Methods** The following list shows the methods of this class:

- [Add \(VerbalTableEntries\)](#) on page 80
- [Item \(VerbalTableEntries\)](#) on page 81
- [Remove \(VerbalTableEntries\)](#) on page 82

### Related topics

### References

<a href="#">VerbalTableEntry</a> .....	83
<a href="#">VerbalTableScaling</a> .....	88

## Add (VerbalTableEntries)

**Class** VerbalTableEntries (Collection)

**Syntax** `RetVal = OBJ.Add(InVal ,OutVal)`

**Purpose** To add an entry to the conversion table (CompuVTab).



**Parameters** The method provides the following parameters:

Parameter	Type	Description
InVal	Double	Contains the input value (source value) of the VerbalTableEntry object.
OutVal	String	Contains the output value (converted value) of the VerbalTableEntry object.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">VerbalTableEntry</a> on page 83	Contains the VerbalTableEntry object.

**Related topics**

References

[VerbalTableEntries \(Collection\)](#)..... 79

## Item (VerbalTableEntries)

**Class** VerbalTableEntries (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get an entry by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the VerbalTableEntry object.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">VerbalTableEntry</a> on page 83	Contains the VerbalTableEntry object.

**Related topics****References**[VerbalTableEntries \(Collection\)..... 79](#)

## Remove (VerbalTableEntries)

**Class** VerbalTableEntries (Collection)**Syntax** `OBJ.Remove(Entry)`**Purpose** To remove an entry from the table.**Parameters** The method provides the following parameters:

Parameter	Type	Description
Entry	Int	Contains the index number of the VerbalTableEntry object within the VerbalTableEntries collection.

**Note**

At least one entry has to remain in the table when you modify an existing scaling.

**Return value** —**Related topics****References**[VerbalTableEntries \(Collection\)..... 79](#)

# VerbalTableEntry

## Class Description (VerbalTableEntry)

**Purpose** To access an entry of a conversion table for converting numerical values into strings (CompuVTab).

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
InVal	Double	To get/set the input value (source value) of the VerbalTableEntry object.
OutVal	String	To get/set the output value (converted value) of the VerbalTableEntry object.

**Methods** —

### Related topics

### References

[VerbalTableEntries \(Collection\)](#)..... 79

## VerbalTableRangeEntries (Collection)

### Where to go from here

### Information in this section

<a href="#">Class Description (VerbalTableRangeEntries)</a> .....	84
To provide a collection representing all entries in a conversion table for converting ranges of numerical values into strings (CompuVTabRange).	
<a href="#">Add (VerbalTableRangeEntries)</a> .....	84
To add an entry to the conversion table (CompuVTab).	
<a href="#">Item (VerbalTableRangeEntries)</a> .....	85
To get an entry by index.	
<a href="#">Remove (VerbalTableRangeEntries)</a> .....	86
To remove an entry from the table.	

## Class Description (VerbalTableRangeEntries)

**Purpose** To provide a collection representing all entries in a conversion table for converting ranges of numerical values into strings (CompuVTabRange).

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of entries. (read-only)
DefaultValue	Variant	<p>To get or set the default value used for converting numerical values that are not included in the table.</p> <ul style="list-style-type: none"> <li>▪ If the source value is not included in the table, the default value is returned as the result of the conversion.</li> <li>▪ If no default value is set, <b>None</b> is returned as the result of the conversion.</li> </ul> <p>The default value of a verbal range table (CompuVTabRange) is stored in the measurement data file as string.<sup>1)</sup></p>

<sup>1)</sup> Default values for verbal range tables are not supported by the CSV and MAT measurement data file formats. In both formats, the default value is replaced by a quiet NaN (QNaN).

**Methods** The following list shows the methods of this class:

- [Add \(VerbalTableRangeEntries\)](#) on page 84
- [Item \(VerbalTableRangeEntries\)](#) on page 85
- [Remove \(VerbalTableRangeEntries\)](#) on page 86

### Related topics

### References

<a href="#">VerbalTableRangeEntry</a> .....	87
<a href="#">VerbalTableRangeScaling</a> .....	88

## Add (VerbalTableRangeEntries)

**Class** VerbalTableRangeEntries (Collection)

**Syntax** `RetVal = OBJ.Add(MinVal ,MaxVal ,OutVal)`

**Purpose** To add an entry to the conversion table (CompuVTab).

**Parameters** The method provides the following parameters:

Parameter	Type	Description
MinVal	Float	Contains the minimum value of the input range (source value).
MaxVal	Float	Contains the maximum value of the input range (source value).
OutVal	String	Contains the output string (converted value).

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">VerbalTableRangeEntry</a> on page 87	Contains the VerbalTableRangeEntry object.

**Related topics**

References

[VerbalTableRangeEntries \(Collection\)](#)..... 83

## Item (VerbalTableRangeEntries)

**Class** VerbalTableRangeEntries (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get an entry by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the VerbalTableRangeEntry object.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">VerbalTableRangeEntry</a> on page 87	Contains the VerbalTableRangeEntry object.

**Related topics****References**[VerbalTableRangeEntries \(Collection\)..... 83](#)

## Remove (VerbalTableRangeEntries)

**Class**

VerbalTableRangeEntries (Collection)

**Syntax**`OBJ.Remove(Entry)`**Purpose**

To remove an entry from the table.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Index	Int	Contains the index number of the VerbalTableRangeEntry object within the VerbalTableRangeEntries collection.

**Note**

At least one entry has to remain in the table when you modify an existing scaling.

**Return value**

—

**Related topics****References**[VerbalTableRangeEntries \(Collection\)..... 83](#)

# VerbalTableRangeEntry

## Where to go from here

## Information in this section

[Class Description \(VerbalTableRangeEntry\).....87](#)  
To access an entry of a conversion table for converting ranges of numerical values into strings (CompuVTabRange).

## Information in other sections

[VerbalTableRangeEntries \(Collection\).....83](#)  
To provide a collection representing all entries in a conversion table for converting ranges of numerical values into strings (CompuVTabRange).

## Class Description (VerbalTableRangeEntry)

### Purpose

To access an entry of a conversion table for converting ranges of numerical values into strings (CompuVTabRange).

### Attributes

The following table shows the attributes of this class.

Attribute	Type	Purpose
MinVal	Float	To get/set the minimum value of the input range (source value).
MaxVal	Float	To get/set the maximum value of the input range (source value).
OutVal	String	To get/set the output string (converted value).

### Methods

—

### Related topics

#### References

[VerbalTableRangeEntries \(Collection\).....83](#)

## VerbalTableRangeScaling

### Class Description (VerbalTableRangeScaling)

---

**Purpose** To access a conversion table for converting ranges of numerical values into strings (CompuVTabRange).

---

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Table	<a href="#">VerbalTableRangeEntries (Collection)</a> on page 83	To get a collection representing a verbal range scaling table. (read-only)

---

**Methods** –

---

**Base class** [Scaling](#) on page 62

## VerbalTableScaling

### Class Description (VerbalTableScaling)

---

**Purpose** To access a conversion table for converting numerical values into strings (CompuVTab).

---

**Attributes** The following table shows the attributes of this class.

Attribute	Type	Purpose
Table	<a href="#">VerbalTableEntries (Collection)</a> on page 79	To provide a collection representing all entries in a conversion table for converting numerical values into strings (CompuVTab).



**Methods**

—

**Base class**[Scaling](#) on page 62

## XAxes (Collection)

**Where to go from here****Information in this section**

<a href="#">Class Description (XAxes)</a> .....	89
To provide a collection of the measurement's x-axis objects.	
<a href="#">Add (XAxes)</a> .....	90
To add an x-axis to the measurement.	
<a href="#">Item (XAxes)</a> .....	91
To get an x-axis by index.	
<a href="#">Remove (XAxes)</a> .....	91
To remove an x-axis from the measurement.	
<a href="#">RemoveAll (XAxes)</a> .....	92
To remove all x-axes from the measurement.	

## Class Description (XAxes)

**Purpose**

To provide a collection of the measurement's x-axis objects.

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Count	Int	To get the number of the measurement's x-axis objects. (read-only)
XAxisOffset	Float	To get/set the x-axis offset. The x-axis offset is used to shift the x-values of the entire measurement.

**Methods**

The following list shows the methods of this class.

- [Add \(XAxes\)](#) on page 90
- [Item \(XAxes\)](#) on page 91

- [Remove \(XAxes\)](#) on page 91
- [RemoveAll \(XAxes\)](#) on page 92

**Related topics****References**

<a href="#">Measurement</a> .....	36
<a href="#">XAxis</a> .....	93

## Add (XAxes)

**Class**

XAxes (Collection)

**Syntax**

```
RetVal = OBJ.Add(Name ,Device ,Data)
```

**Purpose**

To add an x-axis to the measurement.

**Parameters**

The method provides the following parameters:

Parameter	Type	Description
Name	String	Contains the name of the x-axis.
Device	String	Contains the name of the x-axis' platform/device.
Data	Variant	Contains the data of the x-axis.

**Return value**

The method returns a value of the following type:

Type	Description
<a href="#">XAxis</a> on page 93	Contains the XAxis object.

**Related topics****References**

<a href="#">XAxes (Collection)</a> .....	89
--	----

## Item (XAxes)

**Class** XAxes (Collection)

**Syntax** `RetVal = OBJ.Item(Index)`

**Purpose** To get an x-axis by index.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the XAxis object within the XAxes collection. The index can either be a number or a string containing the x-axis key. Note that this index can change if a signal (!) is added to or removed from the measurement.

**Return value** The method returns a value of the following type:

Type	Description
<a href="#">XAxis</a> on page 93	Contains the XAxis object

**Related topics**

**References**

[XAxes \(Collection\)](#)..... 89

## Remove (XAxes)

**Class** XAxes (Collection)

**Syntax** `OBJ.Remove(Index)`

**Purpose** To remove an x-axis from the measurement.

**Note**

You cannot remove an x-axis that is referenced by a signal.

**Parameters** The method provides the following parameters:

Parameter	Type	Description
Index	Variant	Contains the index of the XAxis object within the XAxes collection. The index can either be a number or a string containing the x-axis key. Note that this index can change if a signal (!) is added to or removed from the measurement.

**Return value** —

**Related topics**

## References

[XAxes \(Collection\).....89](#)

## RemoveAll (XAxes)

**Class** XAxes (Collection)

**Syntax** `OBJ.RemoveAll()`

**Purpose** To remove all x-axes from the measurement.

**Note**

You cannot remove an x-axis that is referenced by a signal.

**Parameters** —

---

**Return value** —

---

**Related topics**

**References**

[XAxes \(Collection\)..... 89](#)

## XAxis

### Class Description (XAxis)

---

**Purpose** To access an x-axis (time axis) of the measurement.

---

**Attributes**

The following table shows the attributes of this class.

Attribute	Type	Purpose
Name	String	To get the x-axis' name. (read-only)
Data	Variant	To get the x-axis' data. (read-only)
DataBlockTimestamps	Variant	To get a list of time stamps that specify the beginning and the end of data captures acquired in a triggered measurement. (read-only)
Device	String	To get the x-axis' platform/device. (read-only)
Key	String	To get the x-axis' key. (read-only) The x-axis key can be used to access the XAxes collection. It consists of <ul style="list-style-type: none"> <li>▪ The platform/device name (not matching case)</li> <li>▪ The x-axis name (matching case)</li> </ul>

---

**Methods** —

---

**Related topics**

**References**

[Signal..... 68](#)  
[XAxes \(Collection\)..... 89](#)

# Constants

## Where to go from here

## Information in this section

<a href="#">BookmarkTypeConstants.....</a>	<a href="#">94</a>
To specify the bookmark type.	
<a href="#">MDFVariableNameStorageConstants.....</a>	<a href="#">95</a>
To specify how to store the name of MDF variables.	
<a href="#">MeasurementDeviceOptions.....</a>	<a href="#">96</a>
To specify whether the platform's/device's display identifier is used in graphical user interfaces (GUIs).	
<a href="#">MeasurementLoadingOptions.....</a>	<a href="#">97</a>
To specify options for the loading of a measurement.	
<a href="#">MeasurementRemovalOptions.....</a>	<a href="#">97</a>
To specify options for the removal of measurements.	
<a href="#">MeasurementSectionOptions.....</a>	<a href="#">98</a>
To specify whether a time section definition includes a start or a stop time stamp.	
<a href="#">ScalingTypeConstants.....</a>	<a href="#">98</a>
To specify the type of computation method that is used to transform a source value into a converted value.	
<a href="#">SignalTypeConstants.....</a>	<a href="#">99</a>
To specify whether the signal has a source data property.	
<a href="#">VariableTypeConstants.....</a>	<a href="#">100</a>
To specify the variable type.	

## BookmarkTypeConstants

**Purpose** To specify the bookmark type.

**Description** The bookmark type describes the event that invokes the bookmark.  
The following values are available:

Value	Description
btCalculatedVariableError	Indicates that the calculation of a calculated variable led to an error.
btDataSetActivation	Indicates that a data set was activated.

Value	Description
btDataSetDownload	Indicates that a data set was downloaded.
btDiagnosticTroubleCode	Indicates that the number of trouble code entries was changed.
btManual	Indicates a manually created bookmark.
btPlatformDataLossDetected	Indicates that data loss was detected on a platform.
btPlatformPlugged	Indicates that a platform was plugged.
btPlatformUnplugged	Indicates that a platform was unplugged.
btStartRecording	Indicates the beginning of a recording.
btStopRecording	Indicates the end of a recording.
btTrigger	Indicates that a trigger was invoked.
btXILAPIEESPortManualTrigger	Indicates that the XIL API EESPort manual trigger was invoked.

These constants are used in the following attributes or methods.

**Attributes**    [Bookmark.Type](#) (refer to [Class Description \(Bookmark\)](#) on page 22)

**Methods**    [Add \(Bookmarks\)](#) on page 24

## Related topics

## References

<a href="#">Bookmark.....</a>	<a href="#">21</a>
<a href="#">Bookmarks (Collection).....</a>	<a href="#">22</a>

# MDFVariableNameStorageConstants

## Purpose

To specify how to store the name of MDF variables.

## Description

The following values are available:

Value	Description
mdfvnsVariableOnly	Only the name of the variable is stored. For example: <code>SignalGenOutput</code> . <sup>1)</sup>
mdfvnsBlockgroupAndVariable	Only the block group and the name of the variable are stored. For example: <code>SignalGenerator/SignalGenOutput</code> . <sup>1)</sup>
mdfvnsFullPath	The full path and the name of the variable are stored. For example: <code>Platform/Model Root/SignalGenerator/SignalGenerator/SignalGenOutput</code> .

<sup>1)</sup> If this option results in identical strings for different variables, ControlDesk adds a consecutive number to the string, for example, `SignalGenOutput(2)`.

The constants are used in the following attributes or methods.

**Attributes**    MDFFormatOption.VariableNameStorage (refer to [Class Description \(MDFFormatOption\)](#) on page 36)

**Methods**        –

---

**Related topics****References**

[MDFFormatOption](#)..... 36

## MeasurementDeviceOptions

---

**Purpose**

To specify whether the platform'/device's display identifier is used in graphical user interfaces (GUIs).

---

**Description**

The following values are available:

Value	Description
mdUseDisplayName	The display identifier is used for the output in GUIs.
mdNone	The variable's name is used for the output in GUIs.

These constants are used in the following attributes or methods.

**Attributes**        –

**Methods**    [GetDeviceOptions \(Measurement\)](#) on page 39; [SetDeviceOptions \(Measurement\)](#) on page 40

---

**Related topics****References**

[Measurement](#)..... 36



## MeasurementLoadingOptions

**Purpose** To specify options for the loading of a measurement.

**Description** The following values are available:

Value	Description
mlWithoutData	Load the structure of a measurement without data.
mlWithData	Load the entire measurement with data.

These constants are used in the following attributes or methods.

**Attributes** –

**Methods** [Load \(Measurements\)](#)

### Related topics

#### References

<a href="#">Load (Measurements)</a> .....	46
<a href="#">Measurements (Collection)</a> .....	43

## MeasurementRemovalOptions

**Purpose** To specify options for the removal of measurements.

**Description** The following values are available:

Value	Description
mrIgnore	Removing a measurement does not save it.
mrAutoSave	Removing a measurement saves it automatically.

These constants are used in the following attributes or methods.

**Attributes** –

**Methods** [Remove \(Measurements\)](#) on page 47; [RemoveAll \(Measurements\)](#) on page 48

### Related topics

#### References

<a href="#">Measurements (Collection)</a> .....	43
---	----

## MeasurementSectionOptions

**Purpose** To specify whether a time section definition includes a start or a stop time stamp.

**Description** If time stamps of measurement samples exactly match a section boundary, the **MeasurementSectionOptions** constant lets you specify whether these samples are included (default) or excluded from the measurement data.

The following values are available:

Value	Description
msIncludeStartTimestamp	Section definition includes the start time stamp.
msIncludeStopTimestamp	Section definition includes the stop time stamp.

These constants are used in the following attributes or methods.

**Attributes** Measurement.SectionOptions

**Methods** [Load \(Measurements\)](#) on page 46; [SetSection \(Measurement\)](#) on page 41

### Related topics

#### References

<a href="#">Measurement</a> .....	36
<a href="#">Measurements (Collection)</a> .....	43

## ScalingTypeConstants

**Purpose** To specify the type of computation method that is used to transform a source value into a converted value.

**Description** The following values are available:

Value	Description
stLinear	Scaling with linear function
stRationalFunction	Scaling with rational function
stFormula	Scaling with formula
stMulti	Multiscaling: value ranges are mapped to different scalings
stTableInterpolation	Scaling with interpolation table (COMPU_TAB _INTP)

Value	Description
stTableNoInterpolation	Scaling without interpolation table (COMPU_TAB_NOINTP)
stTableVerbal	Scaling with verbal table (COMPU_VTAB )
stTableVerbalRange	Scaling with verbal range table (COMPU_VTAB_RANGE)

These constants are used in the following attributes or methods.

**Attributes**    [Scaling.Type](#) (refer to [Class Description \(Scaling\)](#) on page 62)

**Methods**    [Add \(Scalings\)](#) on page 64; [CreateTable \(Scalings\)](#) on page 65

## Related topics

## References

<a href="#">Scaling.....</a>	<a href="#">62</a>
<a href="#">Scalings (Collection).....</a>	<a href="#">63</a>

# SignalTypeConstants

## Purpose

To specify whether the signal has a source data property.

## Description

The following values are available:

Value	Description
stWithoutSourceData	Signal has no source data property.
stWithSourceData	Signal has a source data property.

These constants are used in the following attributes or methods.

**Attributes**    [Signal.Type](#) (refer to [Class Description \(Signal\)](#) on page 68)

**Methods**    –

## Related topics

## References

<a href="#">Signal.....</a>	<a href="#">68</a>
-----------------------------	--------------------

## VariableTypeConstants

**Purpose** To specify the variable type.

**Description** The following values are available:

Value	Description
vtAxisPointsX	A parameter that consists of a 1-dimensional array containing x-axis points of a look-up table. You can use it for fixed and standard axes.
vtAxisPointsY	A parameter that consists of a 1-dimensional array containing y-axis points of a look-up table. You can use it for fixed and standard axes.
vtCommonAxis	A parameter that consists of a 1-dimensional array containing axis points. A common axis can be referenced by one or more curves or maps.
vtCurve	A parameter that consists of <ul style="list-style-type: none"> <li>A 1-dimensional array containing the axis points for the x-axis. This array can also be specified by a reference to a common axis. The Measurement Data API gives you access to this array via the <b>Components</b> attribute of a signal.</li> <li>Another 1-dimensional array containing the data points for the function values. The curve assigns one data point to each axis point.</li> </ul>
vtMap	A parameter that consists of <ul style="list-style-type: none"> <li>A 1-dimensional array containing the axis points for the x-axis. This array can also be specified by a reference to a common axis. The Measurement Data API gives you access to this array via the <b>Components</b> attribute of a signal.</li> <li>A 1-dimensional array containing the axis points for the y-axis. This array can also be specified by a reference to a common axis. The Measurement Data API gives you access to this array via the <b>Components</b> attribute of a signal.</li> <li>A 2-dimensional array containing data points. The map assigns one data point of the array to each pair of x-axis and y-axis points.</li> </ul>
vtMeasurement	A scalar variable that is usually read-only.
vtMeasurementArray	A 1-, 2-, or 3-dimensional array of measurement variables.
vtParameter	A scalar variable that can be calibrated.
vtValueBlock	A 1- or 2-dimensional array of scalar parameters.

These constants are used in the following attributes or methods.

**Attributes**    [Signal.VariableType](#) (refer to [Class Description \(Signal\)](#) on page 68)

**Methods**    [Add \(Signals\)](#) on page 73

## Related topics

## References

<a href="#">Signal.....</a>	<a href="#">68</a>
<a href="#">Signals (Collection).....</a>	<a href="#">71</a>



# Limitations

## Limitations for Using the Measurement Data API

### Introduction

There are some limitations for using the Measurement Data API.

### Unsupported variable types

You cannot create signals with the following variable types:

- Calculated variable
- Cuboid
- String
- Struct
- Struct array

### No support of parameters with a dynamic number of axis points

You cannot create look-up tables with a dynamic number of axis points (A2L keyword: `NO_AXIS_PTS_X`).

### Fixed axes cannot be mixed with other axes types

If you create a look-up table, you cannot mix a fixed axis with other types of axes, such as a standard axis.

### Creating variables with nested multiscaling tables not supported

You cannot create variables with *nested* multiscaling tables with the Measurement Data API.

### Loading a measurement partially according to a time section

When you load a measurement, you can define a time section to load only a part of the measurement. You must specify the time section *before* you load any measurement data.

---

**Adding/removing signals to/from a collection**

When you add or remove a signal to/from a collection, this can change the numeric indices of the measurement's x-axes (time axes) and scalings in the collection.

---

**Removing referenced x-axes (time axes) or scalings**

You cannot remove x-axes (time axes) or scalings that are referenced by signals.

---

**Creating a measurement by the Add method**

If you create a new measurement using the [Add \(Measurements\)](#) on page 44 method, it has no file path. In this case, you cannot save the measurement using the [Save \(Measurement\)](#) on page 39 method.

Use the [Export \(Measurement\)](#) on page 38 method instead.

---

**Related topics****References**

[General Limitations for Measurement and Recording \(ControlDesk Measurement and Recording !\[\]\(e1c624d4757f08486e89482c18364c17\_img.jpg\)\)](#)



# Glossary

Introduction	Briefly explains the most important expressions and naming conventions used in the ControlDesk documentation.
--------------	---

Where to go from here

Information in this section

Numerics.....	106
A.....	106
B.....	107
C.....	108
D.....	112
E.....	116
F.....	119
G.....	120
H.....	120
I.....	121
K.....	123
L.....	123
M.....	124
N.....	127
O.....	127
P.....	129
Q.....	131
R.....	132

S.....	133
T.....	136
U.....	137
V.....	138
W.....	140
X.....	141

## Numerics

**3-D Viewer** An instrument for displaying items in a 3-D environment.

## A

**A2L file** A file that contains all the relevant information on measurement and calibration variables in an [ECU application](#) and the ECU's communication interface(s). This includes information on the variables' memory addresses and conversion methods, the memory layout and data structures in the ECU as well as [interface description data \(IF\\_DATA\)](#).

**Acquisition** An object in the [Measurement Configuration](#) controlbar that specifies the variables to be measured and their measurement configuration.

**Active variable description** The variable description that is currently active for a platform/device. Multiple variable descriptions can be assigned to one platform/device, but only one of them can be active at a time.

**Additional write variable** A scalar parameter or writable measurement variable that can be connected to an instrument in addition to the [main variable](#). When the value of the main variable changes, the changed value is also applied to all the additional write variables connected to the instrument.

**Airspeed Indicator** An instrument for displaying the airspeed of a simulated aircraft.

**Altimeter** An instrument for displaying the altitude of a simulated aircraft.

**Animated Needle** An instrument for displaying the value of a connected variable by a needle deflection.

**Application image** An image file that contains all the files that are created when the user builds a real-time application. It particularly includes the variable

description (SDF) file. To extend a real-time application, ControlDesk lets the user create an updated application image from a data set. The updated application image then contains a real-time application with an additional set of parameter values.

**Artificial Horizon** An instrument displaying the rotation on both the lateral and the longitudinal axis to indicate the angle of pitch and roll of a simulated aircraft. The Artificial Horizon has a pitch scale and a roll scale.

**Automatic Reconnect** Feature for automatically reconnecting to platform/device hardware, for example, when the ignition is turned off and on, or when the physical connection between the ControlDesk PC and the ECU is temporarily interrupted.

If the feature is enabled for a platform/device and if the platform/device is in the 'unplugged' [🔗](#) state, ControlDesk tries to re-establish the logical connection to the platform/device hardware. After the logical connection is re-established, the platform/device has the same state as before the unplugged state was detected. A measurement started before the unplugged state was detected is resumed.

**Automation** A communication mechanism that can be used by various programming languages. A client can use it to control a server by calling methods and properties of the server's automation interface.

**Automation script** A script that uses automation to control an automation server.

**Axis point object** [Common axis](#) [🔗](#)

## B

---

**Bar** An instrument (or a value cell type of the [Variable Array](#) [🔗](#)) for displaying a numerical value as a bar deflection on a horizontal or vertical scale.

**Bitfield** A value cell type of the [Variable Array](#) [🔗](#) for displaying and editing the source value of a parameter as a bit string.

**Bookmark** A marker for a certain event during a measurement or recording.

**Browser** An instrument for displaying HTML and TXT files. It also supports Microsoft Internet Explorer® plug-ins that are installed on your system.

**Bus communication replay** A feature of the [Bus Navigator](#) [🔗](#) that lets you replay logged bus communication data from a log file. You can add replay nodes

to the Bus Navigator tree for this purpose. You can specify filters to replay selected parts of the [logged bus communication](#).

**Bus configuration** A configuration of all the controllers, communication matrices, and messages/frames/PDUs of a specific communication bus such as CAN. ControlDesk lets you display and experiment with bus configurations in the [Bus Navigator](#).

**Bus connection** A mode for connecting dSPACE real-time hardware to the host PC via bus. The list below shows the possible bus connections:

- dSPACE real-time hardware installed directly in the host PC
- dSPACE real-time hardware installed in an expansion box connected to the host PC via dSPACE link board

**Bus instrument** An instrument available for the [Bus Navigator](#). It can be configured for different purposes, for example, to display information on received messages (RX messages) or to manipulate and transmit messages (TX messages). The instrument is tailor-made and displays only the message- and signal-specific settings which are enabled for display and/or manipulation by ControlDesk during run time.

**Bus logging** A feature of the [Bus Navigator](#) that lets you log raw bus communication data. You can add logger nodes on different hierarchy levels of the Bus Navigator tree for this purpose. You can specify filters to log filtered bus communication. The logged bus communication can be [replayed](#).

**Bus monitoring** A feature of the [Bus Navigator](#) that lets you observe bus communication. You can open monitoring lists and add monitor nodes on different hierarchy levels of the Bus Navigator tree for this purpose. You can specify filters to monitor filtered bus communication.

**Bus Navigator** A [controlbar](#) for handling bus messages, such as CAN messages, LIN frames, and Ethernet packets.

**Bus statistics** A feature of the [Bus Navigator](#) that lets you display and log statistical information on the bus load during [bus monitoring](#).

**Bypassing** A method for replacing an existing ECU function by running a new function.

## C

**Calculated variable** A scalar variable that can be measured and recorded, and that is derived from one or more *input variables*.

The following input variable types are supported:

- [Measurement variables](#)
- Single elements of [measurement arrays](#) or [value blocks](#)
- Scalar [parameters](#), or existing calculated variables

The value of a calculated variable is calculated via a user-defined *computation formula* that uses one or more input variables.

Calculated variables are represented by the  symbol.

**CalDemo ECU** A demo program that runs on the same PC as ControlDesk. It simulates an ECU on which the Universal Measurement and Calibration (XCP [🔗](#)) protocol and the Unified Diagnostic Services (UDS) protocol are implemented.

The CalDemo ECU allows you to perform parameter calibration, variable measurement, and ECU diagnostics with ControlDesk under realistic conditions, but without having to have a real ECU connected to the PC. Communication between the CalDemo ECU and ControlDesk can be established via XCP on CAN or XCP on Ethernet, and UDS on CAN.

#### Tip

If communication is established via XCP on Ethernet, the CalDemo ECU can also run on a PC different from the PC on which ControlDesk is running.

The memory of the CalDemo ECU consists of two areas called [memory page](#) [🔗](#). Each page contains a complete set of parameters, but only one page is accessible by the CalDemo ECU at a time. You can easily switch the memory pages of the CalDemo ECU to change from one [parameter](#) [🔗](#) to another in a single step.

Two ECU tasks run on the CalDemo ECU:

- ECU task #1 runs at a fixed sample time of 5 ms. In ControlDesk's Measurement Configuration, ECU task #1 is related to the time-based 5 ms, 10 ms, 50 ms and 100 ms measurement rasters of the CalDemo ECU.
  - ECU task #2 has a variable sample time. Whenever the CalDemo ECU program is started, the initial sample time is 5 ms. This can then be increased or decreased by using the dSPACE CalDemo dialog.
- ECU task #2 is related to the extEvent measurement raster of the CalDemo ECU.

The CalDemo ECU can also be used to execute diagnostic services and jobs, handle DTCs and perform measurement and calibration via ECU diagnostics.

The CalDemo ECU program is run by invoking **CalDemo.exe**. The file is located in the `. \Demos\CalDemo` folder of the ControlDesk installation.

**Calibration** Changing the [parameter](#) [🔗](#) values of [real-time application](#) [🔗](#)s or [ECU application](#) [🔗](#)s.

**Calibration memory segment** Part of the memory of an ECU containing the calibratable parameters. Memory segments can be defined as `MEMORY_SEGMENT` in the A2L file. ControlDesk can use the segments to evaluate the memory pages of the ECU.

ControlDesk lets you perform the calibration of:

- Parameters inside memory segments
- Parameters outside memory segments
- Parameters even if no memory segments are defined in the A2L file.

**CAN Bus Monitoring device** A device that monitors the data stream on a CAN bus connected to the ControlDesk PC.

The CAN Bus Monitoring device works, for example, with PC-based CAN interfaces such as the DCI-CAN2 or the DCI-CAN/LIN1.

The device supports the following variable description file types:

- DBC
- FIBEX
- AUTOSAR system description (ARXML)

**CANGenerator** A demo program that simulates a CAN system, that is, it generates signals that can be measured and recorded with ControlDesk. The program runs on the same PC as ControlDesk.

The CANGenerator allows you to use the [CAN Bus Monitoring device](#) under realistic conditions, but without having to have any device hardware connected to the PC.

The CAN (Controller Area Network) protocol is used for communication between the CANGenerator and ControlDesk. However, since the CANGenerator runs on the same PC as ControlDesk, ControlDesk does not communicate with the device via a real CAN channel, but via a *virtual CAN channel* implemented on the host PC.

You can start the CAN generator program by running **CANGenerator.exe**. The file is located in the `. \Demos\CANGenerator` folder of the ControlDesk installation.

**Capture** A data packet of all the measurement variables assigned to a [measurement raster](#). The packet comprises the data that results from a single triggering of the raster.

**CCP** Abbreviation of CAN Calibration Protocol. This protocol can be implemented on electronic control units (ECUs) and allows users to access ECUs with measurement and calibration systems (MCS) such as ControlDesk.

The basic features of CCP are:

- Read and write access to the ECU memory, i.e., providing access for calibration
- Synchronous data acquisition
- Flash programming for ECU development purposes

The CCP protocol was developed by ASAM e.V. (Association for Standardization of Automation and Measuring Systems e.V.). For the protocol specification, refer to <http://www.asam.net>.

The following device supports ECUs with an integrated CCP service:

- [CCP device](#)

**CCP device** A device that provides access to an ECU with CCP connected to the ControlDesk PC via CAN, for example, for measurement and calibration purposes via [CCP \(CAN Calibration Protocol\)](#).

**Check Button** An instrument (or a cell type of the [Variable Array](#)) for displaying whether the value of a connected variable matches predefined values or for writing a predefined value to a connected variable.

**cmdloader** A command line tool for handling applications without using the user interface of an experiment software.

**Common axis** A [parameter](#) that consists of a 1-dimensional array containing axis points. A common axis can be referenced by one or more [curves](#) and/or [map](#)s. Calibrating the data points of a common axis affects all the curves and/or maps referencing the axis.

Common axes are represented by the  symbol.

**Common Program Data folder** A standard folder for application-specific configuration data that is used by all users.

%PROGRAMDATA%\dSPACE\

or

%PROGRAMDATA%\dSPACE\

**Computation method** A formula or a table that defines the transformation of a source value into a converted value (and vice versa). In addition to the computation methods defined in the variable description file, ControlDesk provides the *\_\_Identity* computation method which means the converted and the source value are equal.

**Connected** A platform/device state defined by the following characteristics:

- A continuous logical connection is established between ControlDesk and the platform/device hardware.
- A platform/device must be in the 'connected' state before it can change to the 'measuring/recording' or 'online calibration started' state.
- Online calibration is impossible. ControlDesk did not yet adjust the memory segments containing calibration data in the platform/device and on the corresponding hardware. Offline calibration is possible.
- Platform/device configuration is not possible. However, you can invoke platform/device configuration for a platform/device that is in the connected state. ControlDesk temporarily sets the platform/device to the disconnected state.

The 'connected' platform/device state is indicated by the  icon.

**Connection mode** dSPACE real-time systems can be installed within the host PC or connected to the host via a bus interface and/or via Ethernet. When the Ethernet is being used, different network clients might exist. The connection type being used and, in the case of Ethernet, the network client being used, determine the dSPACE systems that can be accessed.

**Control primitive** A special diagnostic communication object for changing communication states or protocol parameters, or for identifying (ECU) variants.

**Controlbar** A window or pane outside the working area. Can be docked to an edge of the main window or float in front of it. A controlbar can contain a

document, such as a layout, or a tool, such as the **Bus Navigator**. It can be grouped with other controlbars in a window with tabbed pages.

**ControlDesk** The main version of ControlDesk for creating and running experiments, and for accessing dSPACE real-time hardware and VEOS. The functionality can be extended by optional software modules.

**ControlDesk - Operator Version** A version of ControlDesk that provides only a subset of functionality for running existing experiments. The functionality can be extended by optional software modules.

**ControlDesk Bus Navigator Module** An optional software module for ControlDesk for handling bus messages, such as CAN, LIN, and FlexRay messages, frames, and PDUs and Ethernet packets.

**ControlDesk ECU Diagnostics Module** An optional software module for ControlDesk that facilitates the calibration and validation of ECU diagnostic functions.

**ControlDesk ECU Interface Module** An optional software module for ControlDesk for calibration and measurement access to electronic control units (ECUs). The module is also required for calibration and measurement access to virtual ECUs (V-ECUs) used in SIL testing scenarios.

**ControlDesk Signal Editor Module** An optional software module for ControlDesk for the graphical definition and execution of signal generators for stimulating model variables of real-time/offline simulation applications.

**Controller board** Single-board hardware computing the real-time application. Contains a real-time processor for fast calculation of the model and I/O interfaces for carrying out the control developments.

**Conversion table** A table that specifies the [value conversion](#) of a source value into a converted value. In the case of [verbal conversion](#), the converted value is a string that represents one numerical value or a range of numerical values.

**Conversion type** The type of a [computation method](#), for example a linear function or a verbal computation method.

**Curve** A [parameter](#) that consists of

- A 1-dimensional array containing the axis points for the x-axis. This array can also be specified by a reference to a [common axis](#).
- Another 1-dimensional array containing data points. The curve assigns one data point to each axis point.

Curves are represented by the  symbol.

## D

**DAQ module** A hardware module for the acquisition of physical quantities



**Data Cursor** One or two cursors that are used to display the values of selected chart positions in a [Time Plotter](#) or an [Index Plotter](#).

**Data logger** An object in the [Measurement Configuration](#) controlbar that lets you configure a [data logging](#).

**Data logger signal list** A list that contains the variables to be included in subsequent [data loggings](#) on real-time hardware.

**Data logging** The recording of data on dSPACE real-time hardware that does not require a physical connection between the host PC and the real-time hardware. In contrast to [flight recording](#), data logging is configured in ControlDesk.

**Data set** A set of the parameters and their values of a platform/device derived from the variable description of the platform/device. There are different types of data sets:

- [Reference data set](#)
- [Sub data set](#)
- [Unassigned data set](#)
- [Working data set](#)

**DCI-CAN/LIN1** A dSPACE-specific interface between the host PC and the CAN/CAN FD bus and/or LIN bus. The DCI-CAN/LIN1 transfers messages between the CAN-/LIN-based devices and the host PC via the universal serial bus (USB).

**DCI-CAN2** A dSPACE-specific interface between the host PC and the CAN bus. The DCI-CAN2 transfers CAN and CAN FD messages between the CAN-based devices and the host PC via the universal serial bus (USB).

**DCI-GSI2** Abbreviation of *dSPACE Communication Interface - Generic Serial Interface 2*. A dSPACE-specific interface for ECU calibration, measurement and ECU interfacing.

**DCI-GSI2 device** A device that provides access to an ECU with DCI-GSI2 connected to the ControlDesk PC for measurement, calibration, and bypassing purposes via the ECU's debug interface.

**DCI-KLine1** Abbreviation of *dSPACE Communication Interface - K-Line Interface*. A dSPACE-specific interface between the host PC and the diagnostics bus via K-Line.

**Debug interface** An ECU interface for diagnostics tasks and flashing.

**Default raster** A platform-/device-specific [measurement raster](#) that is used when a variable of the platform/device is connected to a [plotter](#) or a [recorder](#), for example.

**Deposition definition** A definition specifying the sequence in which the axis point values of a curve or map are deposited in memory.

**Device** A software component for carrying out [calibration](#) and/or [measurement](#), [bypassing](#), [ECU flash programming](#), or [ECU diagnostics](#) tasks.

ControlDesk provides the following devices:

- Bus devices:
  - [CAN Bus Monitoring device](#)
  - [Ethernet Bus Monitoring device](#)
  - [LIN Bus Monitoring device](#)
- [ECU Diagnostics device](#)
- [GNSS device](#)
- Measurement and calibration devices:
  - [CCP device](#)
  - [DCI-GSI2 device](#)
  - [XCP on CAN device](#)
  - [XCP on Ethernet device](#)

Each device usually has a [variable description](#) that specifies the device's variables to be calibrated and measured.

**Diagnostic interface** Interface for accessing the [fault memory](#) of an ECU.

**Diagnostic job** (often called Java job) Programmed sequence that is usually built from a sequence of the [diagnostic service](#). A diagnostic job is either a single-ECU job or a multiple-ECU job, depending on whether it communicates with one ECU or multiple ECUs.

**Diagnostic protocol** A protocol that defines how an ECU communicates with a connected diagnostic tester. The protocol must be implemented on the ECU and on the tester. The [diagnostics database](#) specifies the diagnostic protocol(s) supported by a specific ECU.

ControlDesk's ECU Diagnostics device supports CAN and K-Line as the physical layers for communication with an ECU connected to the ControlDesk PC. For information on the supported diagnostic protocols with CAN and K-Line, refer to [Basics of ECU Diagnostics with ControlDesk \(ControlDesk ECU Diagnostics\)](#).

**Diagnostic service** A service implemented on the ECU as a basic diagnostic communication element. Communication is performed by selecting a service, configuring its parameters, executing it, and receiving the ECU results. When a service is executed, a defined request is sent to the ECU and the ECU answers with a specific response.

**Diagnostic trouble code (DTC)** A hexadecimal index for the identification of vehicle malfunctions. DTCs are stored in the [fault memory](#) of ECUs and can be read by diagnostic testers.

**Diagnostics database** A database that completely describes one or more ECUs with respect to diagnostics communication. ControlDesk supports the ASAM MCD-2 D [ODX database](#) format, which was standardized by ASAM e.V. (Association for Standardisation of Automation and Measuring Systems e.V.). For the format specification, refer to <http://www.asam.net>.

Proprietary diagnostics database formats are not supported by ControlDesk.

**Diagnostics Instrument** An instrument for communicating with an ECU via the diagnostic protocol using [diagnostic services](#), [diagnostic jobs](#), and [control primitives](#).


**Disabled** A platform/device state defined by the following characteristics:

- No logical connection is established between ControlDesk and the platform/device hardware.
- When a platform/device is disabled, ControlDesk does not try to establish the logical connection for that platform/device. Any communication between the platform/device hardware and ControlDesk is rejected.
- Online calibration is impossible. Offline calibration is possible.
- Platform/device configuration is possible.

The 'disabled' platform/device state is indicated by the  icon.

**Disconnected** A platform/device state defined by the following characteristics:

- No logical connection is established between ControlDesk and the platform/device hardware.
- When a platform/device is in the disconnected state, ControlDesk does not try to re-establish the logical connection for that platform/device.
- Online calibration is impossible. Offline calibration is possible.
- Platform/device configuration is possible.

The 'disconnected' platform/device state is indicated by the  icon.

**Display** An instrument (or a value cell type of the [Variable Array](#)) for displaying the value of a scalar variable or the text content of an ASCII variable.

**Documents folder** A standard folder for user-specific documents.

%USERPROFILE%\Documents\dSPACE\  
<ProductName>\<VersionNumber>

**DS1006 Processor Board platform** A platform that provides access to a DS1006 Processor Board connected to the host PC for HIL simulation and function prototyping purposes.

**DS1007 PPC Processor Board platform** A platform that provides access to a single multicore DS1007 PPC Processor Board or a DS1007 multiprocessor system consisting of two or more DS1007 PPC Processor Boards, connected to the host PC for HIL simulation and function prototyping purposes.

**DS1104 R&D Controller Board platform** A platform that provides access to a DS1104 R&D Controller Board installed in the host PC for function prototyping purposes.

**DS1202 MicroLabBox platform** A platform that provides access to a MicroLabBox connected to the host PC for function prototyping purposes.

**DsDAQ service** A service in a [real-time application](#) or [offline simulation application \(OSA\)](#) that provides measurement data from the application to the

host PC. Unlike the [host service](#), the DsDAQ service lets you perform, for example, triggered measurements with complex trigger conditions.

The following platforms support applications that contain the DsDAQ service:

- [DS1007 PPC Processor Board platform](#)
- [DS1202 MicroLabBox platform](#)
- [MicroAutoBox III platform](#)
- [SCALEXIO platform](#)
- [VEOS platform](#)
- [XIL API MAPort platform](#)

**dSPACE Calibration and Bypassing Service** An ECU service for measurement, calibration, bypassing, and ECU flash programming. The dSPACE Calibration and Bypassing Service can be integrated on the ECU. It provides access to the ECU application and the ECU resources and is used to control communication between an ECU and a calibration and/or bypassing tool.

With the dSPACE Calibration and Bypassing Service, users can run measurement, calibration, bypassing, and flash programming tasks on an ECU via the DCI-GSI2. The service is also designed for bypassing ECU functions using dSPACE prototyping hardware by means of the RTI Bypass Blockset in connection with DPMEM PODs. The dSPACE Calibration and Bypassing Service allows measurement, calibration, and bypassing tasks to be performed in parallel.

**dSPACE Internal Bypassing Service** An ECU service for on-target prototyping. The dSPACE Internal Bypassing Service can be integrated in the ECU application. It lets you add additional functions to be executed in the context of the ECU application without the need for recompiling the ECU application.

**dSPACE Log** A collection of errors, warnings, information, questions, and advice issued by all dSPACE products and connected systems over more than one session.

**dSPACE system** A hardware system such as a MicroAutoBox III or SCALEXIO system on which the [real-time application](#) runs.

**Duration trigger** A [trigger](#) that defines a duration. Using a duration trigger, you can, for example, specify the duration of data acquisition for a [measurement raster](#). A duration trigger can be used as a [stop trigger](#).

## E

**ECU** Abbreviation of *electronic control unit*.

**ECU application** A sequence of operations executed by an ECU. An ECU application is mostly represented by a group of files such as [ECU Image files](#), MAP files, [A2L files](#) and/or software module description files.

**ECU calibration interface** Interface for accessing an ECU by either emulating the ECU's memory or using a communication protocol (for example, XCP on CAN).

**ECU diagnostics** Functions such as:

- Handling the ECU fault memory: Entries in the ECU's fault memory can be read, cleared, and saved.
- Executing diagnostic services and jobs: Users can communicate with an ECU via a diagnostic protocol using diagnostic services, diagnostic jobs, and control primitives.

ControlDesk provides the [ECU Diagnostics device](#) device to access ECUs for diagnostic tasks. Communication is via [diagnostic protocol](#)s implemented on the ECUs.

ECU diagnostics with ControlDesk are completely based on Open Diagnostic Data Exchange (ODX), the ASAM MCD-2 D diagnostics standard.

ControlDesk provides the [Fault Memory Instrument](#) and the [Diagnostics Instrument](#) for ECU diagnostics tasks.

**ECU Diagnostics device** A device that provides access to ECUs connected to the ControlDesk PC via CAN or K-Line for diagnostics or flash programming purposes.

ControlDesk provides the *ECU Diagnostics v2.0.2* device, which supports the ASAM MCD-3 D V2.0.2 standard.

ControlDesk supports the following ODX database standards:

- ASAM MCD-2 D V2.0.1
- ASAM MCD-2 D V2.2.0 (ISO 22901-1)

**ECU flash programming** A method by which new code or data is stored in ECU flash memory.

**ECU Image file** A binary file that is part of the [ECU application](#). It usually contains the code of an ECU application and the data of the parameters within the application. It can be stored as an Intel Hex (HEX) or Motorola S-Record (MOT or S19) file.

**EESPort Configurations controlbar** A [controlbar](#) for configuring [error configuration](#)s.

**Electrical error simulation** Simulating electrical errors such as loose contacts, broken cables, and short-circuits, in the wiring of an ECU. Electrical error simulation is performed by the failure simulation hardware of an HIL simulator.

**Electrical Error Simulation port (EESPort)** An *Electrical Error Simulation port* (EESPort) provides access to a failure simulation hardware for simulating electrical errors in an ECU wiring according to the ASAM AE XIL API standard.

The configuration of the EESPort is described by a hardware-dependent *port configuration* and one or more *error configurations*.

**Environment model** A model that represents a part or all of the ECU's environment in a simulation scenario.

The environment model is a part of the [simulation system](#).

**Environment VPU** The executable of an [environment model](#) built for the VEOS platform. An environment VPU is part of an offline simulation application (OSA).

**Error** An electrical error that is specified by:

- An error category
- An error type
- A load type

**Error category** The error category defines how a signal is disturbed. Which errors you can create for a signal depends on the connected failure simulation hardware.

**Error configuration** An XML file that describes a sequence of errors you want to switch during electrical error simulation. Each error configuration comprises error sets with one or more errors.

**Error set** An error set is used to group errors (pin failures).

**Error type** The error type specifies the way an error category – i.e., an interruption or short circuit of signals – is provided. The error type defines the disturbance itself.

**Ethernet Bus Monitoring device** A device that monitors the data stream on an Ethernet network connected to the ControlDesk PC.

The device supports the following variable description file type:

- AUTOSAR system description (ARXML)

**Ethernet connection** A mode for connecting dSPACE real-time hardware to the host PC via Ethernet. The list below shows the possible Ethernet connections:

- dSPACE real-time hardware installed in an expansion box connected to the host PC via Ethernet.
- MicroAutoBox II/III and MicroLabBox connected via Ethernet.

**Ethernet decoding** A feature of the [Bus Navigator](#) that lets you view protocol data and raw data of an Ethernet frame.

**Event** An event that is triggered by an action performed in ControlDesk.

**Event context** The scope of validity of [event source](#)s and [event](#)s. There is one [event handler](#) code area for each event context.

**Event handler** Code that is executed when the related [event](#) occurs.

**Event management** Functionality for executing custom code according to actions triggered by ControlDesk.

**Event source** An object providing and triggering [event](#)s. *LayoutManagement* is an example of an event source.

**Event state** State of an [event](#). ControlDesk provides the following event states:

- No [event handler](#) is defined
- Event handler is defined and enabled
- Event handler is defined and disabled
- Event handler is defined, but no Python code is available
- Event handler is deactivated because a run-time error occurred during the execution of the Python code

**Expansion box** A box that hosts dSPACE boards. It can be connected to the host PC via bus connection or via network.

**Experiment** A container for collecting and managing information and files required for a parameter calibration and/or measurement task. A number of experiments can be collected in a project but only one of them can be active.

**Extension script** A Python script (PY or PYC file) that is executed each time ControlDesk starts up. An extension script can be executed for all users or user-specifically.

## F

**Failure insertion unit** Hardware unit used with dSPACE simulators to simulate failures in the wiring of an ECU, such as broken wire and short circuit to ground.

**Fault memory** Part of the ECU memory that stores diagnostic trouble code (DTC) entries with status and environment information.

**Fault Memory Instrument** An instrument for reading, clearing, and saving the content of the ECU's [fault memory](#).

**Firmware update** An update for the firmware installed in the board's flash memory. Firmware should be updated if it is older than required by the real-time application to be downloaded.

**Fixed axis** An axis with data points that are not deposited in the ECU memory. Unlike a [common axis](#), a fixed axis is specified within a [curve](#) or [map](#). The parameters of a fixed axis cannot be calibrated.

**Fixed parameter** A [parameter](#) that has a fixed value during a running simulation. Changing the value of a fixed parameter does not immediately affect the simulation results. The affect occurs only after you stop the simulation and

start it again. A fixed parameter is represented by an added pin in its symbol, for example: .

**Flash job** A specific diagnostic job for flashing the ECU memory. A flash job implements the process control for flashing the ECU memory, such as initialization, security access, writing data blocks, etc.

**Flight recording** The recording of data on dSPACE real-time hardware that does not require a physical connection between the host PC and the real-time hardware. In contrast to [data logging](#), flight recording is not configured in ControlDesk but via RTI and RTLib.

**Frame** An instrument for adding a background frame to a layout, for example, to visualize an instrument group.

## G

---

**Gauge** An instrument for displaying the value of the connected variable by a needle deflection on a circular scale.

**Gigalink module** A dSPACE board for connecting several processor boards in a multiprocessor system. The board allows high-speed serial data transmission via fiber-optic cable.

**GNSS data** Positioning and timing data that is transmitted by a Global Navigation Satellite System (GNSS), such as GPS, GLONASS, or Galileo. GNSS receivers use this data to determine their location.

**GNSS device** A device that provides positioning data from a GNSS receiver (e.g., a serial GPS mouse) in ControlDesk. ControlDesk provides the *GNSS (GPS, GLONASS, Galileo, ...)* device that supports various global navigation satellite systems.

**GPX file** An XML file that contains geodata, such as waypoints, routes, or tracks. In ControlDesk, you can import GPX files to visualize GNSS positioning data in a Map instrument.

**Group** A collection of variables that are grouped according to a certain criterion.

## H

---

**Heading Indicator** An instrument displaying the heading direction of a simulated aircraft on a circular scale.



**Host service** A service in a [real-time application](#) that provides measurement data from the application to the host PC.

The following platforms support applications that contain the host service:

- [DS1006 Processor Board platform](#)
- [DS1104 R&D Controller Board platform](#)
- [MicroAutoBox platform](#)
- [Multiprocessor System platform](#)

**Index Plotter** A [plotter instrument](#) for displaying signals that are measured in an event-based raster (index plots).

**Input quantity** A measurement variable that is referenced by a common axis and that provides the input value of that axis.

**Instrument** An on-screen representation that is designed to monitor and/or control simulator variables interactively and to display data captures. Instruments can be arranged freely on [layout](#)s.

The following instruments can be used in ControlDesk:

- [3-D Viewer](#)
- [Airspeed Indicator](#)
- [Altimeter](#)
- [Animated Needle](#)
- [Artificial Horizon](#)
- [Bar](#)
- [Browser](#)
- [Bus Instrument](#)
- [Check Button](#)
- [Diagnostics Instrument](#)
- [Display](#)
- [Fault Memory Instrument](#)
- [Frame](#)
- [Gauge](#)
- [Heading Indicator](#)
- [Index Plotter](#)
- [Invisible Switch](#)
- [Knob](#)
- [Multistate Display](#)
- [Multiswitch](#)
- [Numeric Input](#)
- [On/Off Button](#)

- [Push Button](#)
- [Radio Button](#)
- [Selection Box](#)
- [Slider](#)
- [Sound Controller](#)
- [Static Text](#)
- [Steering Controller](#)
- [Table Editor](#)
- [Time Plotter](#)
- [Variable Array](#)
- [XY Plotter](#)

**Instrument Navigator** A [controlbar](#) that displays a tree with all the [instrument](#)s of the active [layout](#) and all the variables that are connected to them. The Instrument Navigator's main function is easy selection of instruments in complex layouts.

**Instrument script** A Python script used to extend the functionality of an [instrument](#).

**Instrument Selector** A [controlbar](#) that provides access to ControlDesk's [instrument](#)s. The instruments can be placed on a [layout](#) via double-click or drag & drop.

**Interface description data (IF\_DATA)** An information structure, mostly provided by an [A2L file](#), describing the type, features and configuration of an implemented ECU interface.

**Internal Interpreter** ControlDesk's built-in programming interface for editing, running and importing Python scripts. It contains an [Interpreter controlbar](#) where the user can enter Python commands interactively and which displays output and error messages of Python commands.

**Interpreter controlbar** A [controlbar](#) that can be used to execute line-based commands. It is used by the [Internal Interpreter](#) to print out Python standard error messages and standard output during the execution or import of Python scripts.

**Invisible Switch** An instrument for defining an area that is sensitive to mouse operations.

**IOCNET** IOCNET (I/O carrier network) is a dSPACE-specific high-speed serial communication bus that connects all the real-time hardware in a SCALEXIO system. IOCNET can also be used to build a multiprocessor system that consists of multiple SCALEXIO processor hardware components.

## K

---

**Knob** An instrument for displaying and setting the value of the connected variable by means of a knob on a circular scale.

## L

---

**Label list** A list of user-defined variables that can be used for saving connected variables, etc.

**Layout** A window with [instrument](#) <sup>🔗</sup>s connected to variables of one or more simulation models.

**Layout Navigator** A [controlbar](#) <sup>🔗</sup> that displays all opened [layout](#) <sup>🔗</sup>s. It can be used for switching between layouts.

**Layout script** A Python script used to extend the functionality of a [layout](#) <sup>🔗</sup>.

**Leading raster** The [measurement raster](#) <sup>🔗</sup> that specifies the [trigger](#) <sup>🔗</sup> settings for the [Time Plotter](#) <sup>🔗</sup> display. The leading raster determines the time range that is visible in the plotter if a start and stop trigger is used for displaying the signals.

**LIN Bus Monitoring device** A device that monitors the data stream on a LIN bus connected to the ControlDesk PC.

The LIN Bus Monitoring device works, for example, with PC-based LIN interfaces. The device supports the following variable description file types:

- LDF
- FIBEX
- AUTOSAR system description (ARXML)

**Load type** The load type specifies the option to disturb a signal with or without load rejection.

**Local Program Data folder** A standard folder for application-specific configuration data that is used by the current, non-roaming user.

%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\<ProductName>

**Logical link** A representation of an ECU specified in the diagnostics database. A logical link contains information on the ECU itself, and all the information required for accessing it, such as the [diagnostic protocol](#) <sup>🔗</sup> used for

communication between the ECU and ControlDesk. Each logical link is represented by a unique short name in the [ODX database](#).

**Look-up table** A look-up table maps one or more input values to one output value. You have to differentiate between the following look-up table types:

- A 1-D look-up table maps one input value to one output value.
- A 2-D look-up table maps two input values to one output value.
- An n-D look-up table maps multidimensional table data with 3 or more input values to one output value.

Look-up table is a generic term for [curves](#) and [maps](#).

## M

**Main variable** A scalar variable that is visualized in an instrument that can be used to change parameter values. In addition to the main variable, [additional write variable](#)s can also be connected to (but not visualized in) the same instrument. When you change the value of the main variable in an instrument, the changed value is also applied to all the additional write variables connected to that instrument.

**Map** A [parameter](#) that consists of

- A 1-dimensional array containing the axis points for the x-axis. This array can also be specified by a reference to a [common axis](#).
- A 1-dimensional array containing the axis points for the y-axis. This array can also be specified by a reference to a [common axis](#).
- A 2-dimensional array containing data points. The map assigns one data point of the array to each pair of x-axis and y-axis points.

Maps are represented by the  symbol.

**Map file** A file that contains symbols (symbolic names) and their physical addresses. It is generated during a build process of an ECU application.

**Map instrument** A customized [Browser](#) instrument. It uses an instrument script to open a web map and connect positioning data to the map. The Map instrument offers prepared connection nodes to connect variables with [GNSS data](#).

**Measurement** Viewing and analyzing the time traces of [variables](#), for example, to observe the effects of ECU parameter changes.

ControlDesk provides various [instruments](#) for measuring variables.

**Measurement (variable type)** A scalar variable that can be measured, including individual elements of a measurement array.

Measurement variables are represented by the  symbol.

**Measurement array** A 1-, 2-, or 3-dimensional array of measurement variables. In variable lists, ControlDesk displays entries for the measurement array itself and for each array element.

Measurement arrays are represented by the  symbol.

**Measurement buffer** A ring buffer that buffers measurement data at the start of a [measurement](#). The measurement buffer size determines the amount of data that can be buffered. Earlier values are overwritten by later values when the buffer capacity is exceeded (buffer overflow).

**Measurement Configuration** A [controlbar](#) that allows you to configure [measurement](#), [recording](#) and [data logging](#).

**Measurement Data API** Application programming interface for accessing measurement data. The API lets the user access measurement data without having to use ControlDesk.

**Measurement Data Pool** A [controlbar](#) that provides access to measurement data recorded in measurement data files.

**Measurement raster** Specification of how often a value of a [variable](#) is updated during a [measurement](#). A measurement raster can be derived from a [measurement service](#).

**Measurement service** The generic term for the following services:

- [CCP](#) service
- [DsDAQ service](#)
- [Host service](#)
- [XCP](#) service

**Measurement signal list** A list containing the variables to be included in subsequent measurements and recording. The list is global for all platforms/devices of the current experiment. The measurement signal list is available in the configuration area of the [Measurement Configuration](#) controlbar.

**Measurement variable** Any variable type that can be measured but not calibrated.

**Measuring/recording** A platform/device state defined by the following characteristics:

- A continuous logical connection is established between ControlDesk and the platform/device hardware.
- Online calibration is possible. Parameter values can be changed directly on the platform/device hardware.
- A measurement (or recording) is running.
- Platform/device configuration is not possible.

The 'measuring' / 'recording' platform/device state is indicated by the  icon.

**Memory page** An area of a calibration memory. Each page contains a complete set of parameters of the platform/device hardware, but only one of the pages is "visible" to the microcontroller of the ECU or the real-time processor (RTP) of the platform hardware at a time.

ControlDesk supports platform/device hardware with up to two memory pages. These are usually the [working page](#) and the [reference page](#). The parameter values on the two memory pages usually are different. ControlDesk lets you switch from one page to the other, so that when parameters are changed on one page, the changes can be made available to the ECU or prototyping hardware via a single page switch.

**Messages controlbar** A [controlbar](#) displaying a history of all error and warning messages that occur during work with ControlDesk.

**MicroAutoBox III platform** A platform that provides access to a MicroAutoBox III connected to the host PC for function prototyping purposes such as [Bypassing](#).

**MicroAutoBox platform** A platform that provides access to a MicroAutoBox II connected to the host PC for function prototyping purposes such as bypassing.

**Mirrored memory** A memory area created by ControlDesk on the host PC that mirrors the contents of the available memory pages of calibration and prototyping hardware. For hardware with two memory pages, the mirrored memory is divided into a reference and a working page, each of them containing a complete set of parameters. When a calibration or prototyping platform/device is added to an experiment, ControlDesk initially fills the available memory pages of the mirrored memory with the contents of the [ECU Image file](#) (initial filling for calibration devices) or with the contents of the SDF file (initial filling for platforms).

- **Mirrored memory for offline calibration**  
Parameter values can even be changed [offline](#). Changes to parameter values that are made offline affect only the mirrored memory.
- **Offline-to-online transition for online calibration**  
For online calibration, an offline-to-online transition must be performed. During the transition, ControlDesk compares the [memory page](#)s of the hardware of each platform/device with the corresponding pages of the mirrored memory. If the pages differ, the user has to equalize them by uploading them from the hardware to the host PC, or downloading them from the host PC to the hardware.
- **Mirrored memory for online calibration**  
When ControlDesk is in the online mode, parameter value changes become effective synchronously on the memory pages of the hardware and in the mirrored memory. In other words, parameter values on the hardware and on the host PC are always the same while you are performing online calibration.

**Modular system** A dSPACE processor board and one or more I/O boards connected to it.

**Multi-capture history** The storage of all the [capture](#)s acquired during a [triggered measurement](#). The amount of stored data depends on the measurement buffer.

**Multi-pin error** A feature of the SCALEXIO concept for electrical error simulation that lets you simulate a short circuit between three or more signal

channels and/or bus channels. The channels can be located on the same or different boards or I/O units. You can simulate a short circuit between:

- Channels of the same signal category (e.g., four signal generation channels)
- Channels of different signal categories (e.g., three signal generation channels and two signal measurement channels)
- Signal channels and bus channels (e.g., two signal generation channels, one signal measurement channel, and one bus channel)

**Multiple electrical errors** A feature of the SCALEXIO concept for electrical error simulation that lets you switch electrical errors at the same time or in succession. For example, you can simulate an open circuit for one channel and a short circuit for another channel at the same time, without deactivating the first error.

**Multiprocessor System platform** A platform that provides access to:

- A multicore application running on a multicore DS1006 board
- A multiprocessor application on a multiprocessor system consisting of two or more DS1006 processor boards interconnected via Gigalink.

ControlDesk handles a multiprocessor/multicore system as a unit and uses one system description file (SDF file) to load the applications to all the processor boards/cores in the system.

**Multistate Display** An instrument for displaying the value of a variable as an LED state and/or as a message text.

**Multistate LED** A value cell type of the [Variable Array](#) for displaying the value of a variable as an LED state.

**Multiswitch** An instrument for changing variable values by clicking sensitive areas in the instrument and for visualizing different states depending on the current value of the connected variable.

## N

---

**Numeric Input** An instrument (or a value cell type of the [Variable Array](#)) for displaying and setting the value of the connected variable numerically.

## O

---

**Observing variables** Reading variable values cyclically from the dSPACE real-time hardware and displaying their current values in ControlDesk, even if no [measurement](#) is running. Variable observation is performed without using a measurement buffer, and no value history is kept.

For platforms that support variable observation, variable observation is available for [parameters](#) and [measurement variables](#) that are visualized in [single-shot instruments](#) (all instruments except for a [plotter](#)). If you visualize a variable in a single-shot instrument, the variable is not added to the [measurement signal list](#). Visualizing a parameter or measurement variable in a plotter automatically adds the variable to the measurement signal list.

ControlDesk starts observing variables if one of the following conditions is true:

- [Online Calibration is started](#) for the platform.  
All the parameters and measurement variables that are visualized in single-shot instruments are observed.
- [Measurement is started](#) for the platform.  
All the visualized parameters and measurement variables that are not activated for measurement in the measurement signal list are observed. Data of the activated parameters and measurement variables is acquired using measurement rasters.

**ODX database** Abbreviation of Open Diagnostic Data Exchange, a [diagnostics database](#) that is the central ECU description for working with an [ECU Diagnostics device](#) in ControlDesk. The ODX database contains all the information required to perform diagnostic communication between ControlDesk and a specific ECU or set of ECUs in a vehicle network. ControlDesk expects the database to be compliant with ASAM MCD-2 D (ODX).

**Offline** State in which the parameter values of platform/device hardware in the current experiment cannot be changed. This applies regardless of whether or not the host PC is physically connected to the hardware.

The [mirrored memory](#) allows parameter values to be changed even offline.

**Offline simulation** A PC-based simulation in which the simulator is not connected to a physical system and is thus independent of the real time.

**Offline simulation application (OSA)** An offline simulation application (OSA) file is an executable file for VEOS. After the build process with a tool such as the VEOS Player, the OSA file can be downloaded to VEOS.

An OSA contains one or more [VPUs](#), such as V-ECUs and/or environment VPUs.

**On/Off Button** An instrument (or a value cell type of the [Variable Array](#)) for setting the value of the connected parameter to a predefined value when the button is pressed (On value) and released (Off value).


**Online calibration started** A platform/device state defined by the following characteristics:

- A continuous logical connection is established between ControlDesk and the platform/device hardware.
- Online calibration is possible. Parameter values can be changed directly on the platform/device hardware.
- Platform/device configuration is not possible.

Before starting online calibration, ControlDesk lets you compare the [memory page](#)s on the platform/device hardware with the corresponding pages of the [mirrored memory](#). If the parameter values on the pages differ, they must be



equalized by uploading the values from the hardware to ControlDesk, or downloading the values from ControlDesk to the hardware. However, a page cannot be downloaded if it is read-only.

The 'online calibration started' platform/device state is indicated by the  symbol.

**Operation signal** A [signal](#) which represents the result of an arithmetical operation (such as addition or multiplication) between two other signals.

**Operator mode** A working mode of ControlDesk in which only a subset of the ControlDesk functionality is provided. You can work with existing experiments but not modify them, which protects them from unintentional changes.

**Output parameter** A [parameter](#) or [writable measurement](#) whose memory address is used to write the computed value of a [calculated variable](#) to.

## P

**Parameter** Any variable type that can be calibrated.

**Parameter (variable type)** A scalar [parameter](#), as well as the individual elements of a [value block](#).

Scalar parameters are represented by the  symbol.

**Parameter limits** Limits within which parameters can be changed. Parameters have hard and weak limits.

- Hard limits

Hard limits designate the value range of a parameter that you *cannot* cross during calibration.

The hard limits of a parameter originate from the corresponding [variable description](#) and cannot be edited in ControlDesk.

- Weak limits

Weak limits designate the value range of a parameter that you *should not* cross during calibration. When you cross the value range defined by the weak limits, ControlDesk warns you.

In ControlDesk, you can edit the weak limits of a parameter within the value range given by the parameter's hard limits.

**PHS (Peripheral High Speed) bus** A dSPACE-specific bus for communication between a processor board and the I/O boards in a modular system. It allows direct I/O operations between the processor board (bus master) and I/O boards (bus slaves).

**PHS-bus-based system** A modular dSPACE system consisting of a processor board such as the DS1006 Processor Board and I/O boards. They communicate with each other via the [PHS \(Peripheral High Speed\) bus](#).

**Pitch variable** A variable connected to the pitch scale of an [Artificial Horizon](#).

**Platform** A software component representing a simulator where a simulation application is computed in real-time (on dSPACE real-time hardware) or in non-real-time (on VEOS).

ControlDesk provides the following platforms:

- [DS1006 Processor Board platform](#)
- [DS1007 PPC Processor Board platform](#)
- [DS1104 R&D Controller Board platform](#)
- [DS1202 MicroLabBox platform](#)
- [MicroAutoBox platform](#)
- [MicroAutoBox III platform](#)
- [Multiprocessor System platform](#)
- [SCALEXIO platform](#)
- [VEOS platform](#)
- [XIL API MAPort platform](#)

Each platform usually has a [variable description](#) that specifies its variables.

**Platform trigger** A [trigger](#) that is available for a [platform](#) and that is evaluated on the related dSPACE real-time hardware or VEOS.

**Platforms/Devices controlbar** A [controlbar](#) that provides functions to handle [devices](#), [platforms](#), and the [applications](#) assigned to the platforms.

**Plotter instrument** ControlDesk offers three plotter instruments with different main purposes:

- The [Index Plotter](#) displays signals in relation to events.
- The [Time Plotter](#) displays signals in relation to measurement time.
- The [XY Plotter](#) displays signals in relation to other signals.

**Port configuration** To interface the failure simulation hardware, an EESPort needs the hardware-dependent *port configuration file* (PORTCONFIG file). The file's contents must fit the connected HIL simulator architecture and its failure simulation hardware.

**Postprocessing** The handling of measured and recorded data by the following actions:

- Displaying measured or recorded data
- Zooming into measured or recorded signals with a [plotter](#)
- Displaying the values of measurement variables and parameters as they were at any specific point in time

**Processor board** A board that computes real-time applications. It has an operating system that controls all calculations and communication to other boards.

**Project** A container for collecting and managing the information and files required for experiment/calibration/modification tasks in a number of [experiments](#). A project collects the experiments and manages their common data.

**Project controlbar** A [controlbar](#) that provides access to projects and experiments and all the files they contain.

**Project root directory** The directory on your file system to which ControlDesk saves all the experiments and documents of a [project](#). Every project is associated with a project root directory, and several projects can use the same project root directory. The user can group projects by specifying several project root directories.

ControlDesk uses the [Documents folder](#) as the default project root directory unless a different one is specified.

**Properties controlbar** A [controlbar](#) providing access to the properties of, for example, platforms/devices, layouts/instruments, and measurement/recording configurations.

**Proposed calibration** A calibration mode in which the parameter value changes that the user makes do not become effective on the hardware until they are applied. This allows several parameter changes to be written to the hardware together. Being in proposed calibration mode is like being in the offline calibration mode temporarily.

**Push Button** An instrument (or a value cell type of the [Variable Array](#)) for setting the value of the connected parameter by push buttons.

**Python Editor** An editor for opening and editing PY files.

## Q

---

**Quick start measurement** A type of measurement in which all the ECU variables configured for measurement are measured and recorded, starting with the first execution of an ECU task. ControlDesk supports quick start measurements on ECUs with DCI-GSI2, CCP, and XCP (except for XCP on Ethernet with the TCP transmission protocol).

Quick start measurement can be used to perform cold start measurements. Cold start means that the vehicle and/or the engine are cooled down to the temperature of the environment and then started. One reason for performing cold start measurements is to observe the behavior of an engine during the warm-up phase.

## R

**Radio Button** An instrument for displaying and setting the value of the connected parameter by radio buttons.

**Real-time application** An application that can be executed in real time on dSPACE real-time hardware. A real-time application can be built from a Simulink model containing RTI blocks, for example.

**Record layout** A record layout is used to specify a data type and define the order of the data in the memory of the target system (ECU, for example). For scalar data types, a record layout allows you to add an address mode (direct or indirect). For structured (aggregated) data types, the record layout specifies all the structure elements and the order they appear in.

The **RECORD\_LAYOUT** keyword in an A2L file is used to specify the various record layouts of the data types in the memory. The structural setup of the various data types must be described in such a way that a standard application system will be able to process all data types (reading, writing, operating point display etc.).

**Record layout component** A component of a record layout. A structured record layout consists of several components according to the ASAP2 specification. For example, the **AXIS\_PTS\_X** component specifies the x-axis points, and the **FNC\_VALUES** component describes the function values of a map or a curve.

**Recorder** An object in the [Measurement Configuration](#) controlbar that specifies and executes the [recording](#) of variables according to a specific measurement configuration.

**Recorder signal list** A list that contains the variables to be included in subsequent [recordings](#).

**Recording** Saving the time traces of variables to a file. Both measurement variables and parameters can be recorded. Recorded data can be [postprocessed](#) directly in ControlDesk.

A recording can be started and stopped immediately or via a trigger:

- **Immediate recording**  
The recording is started and stopped without delay, without having to meet a trigger condition.
- **Triggered recording**  
The recording is not started or stopped until certain trigger conditions are met. These conditions can be defined and edited in ControlDesk.

**Reduction data** Additional content in an MF4 file that allows for visualizing the MF4 file data depending on the visualization resolution. Reduction data therefore improves the performance of the visualization and postprocessing of measurement data.

**Reference data set** A read-only data set assigned to the reference page of a device that has two [memory page](#)s. There can be only one reference data set for each device. The reference data set is read-only.

**Reference page** Memory area containing the parameters of an ECU. The reference page contains the read-only [reference data set](#).

**Note**

Some platforms/devices provide only a [working page](#). You cannot switch to a reference page in this case.

**Resynchronization** Mechanism to periodically synchronize the drifting timers of the platform/device hardware ControlDesk is connected to. Resynchronization means adjustment to a common time base.

**Roll variable** A variable connected to the roll scale of an [Artificial Horizon](#).

## S

**Sample count trigger** A [trigger](#) that specifies the number of samples in a data capture.

A sample count trigger can be used as a [stop trigger](#).

**SCALEXIO platform** A platform that provides access to a single-core, multicore or multiprocessor [SCALEXIO system](#) connected to the host PC for HIL simulation and function prototyping purposes.

**SCALEXIO system** A dSPACE hardware-in-the-loop (HIL) system consisting of at least one processing hardware component, I/O boards, and I/O units. They communicate with each other via the [IOCNET](#). In a SCALEXIO system, two types of processing hardware can be used, a DS6001 Processor Board or a real-time industry PC as the SCALEXIO Processing Unit. The SCALEXIO system simulates the environment to test an ECU. It provides the sensor signals for the ECU, measures the signals of the ECU, and provides the power (battery voltage) for the ECU and a bus interface for restbus simulation.

**SDF file** The system description file that describes the files to be loaded to the individual processing units of a simulation platform. It also contains the variable description of the relevant [simulation application](#).

The SDF file is generated automatically when the [TRC file](#) is built.

**Segment** The minimum part a [segment signal](#) can consist of.

There are different kinds of segments to be used in segment signals:

- Segments to form synthetic signal shapes (sine, sawtooth, ramp, etc.)
- Segments to perform arithmetical operations (addition, multiplication) with other segments
- Segments to represent numerical signal data (measured data)

**Segment signal** A [signal](#) consisting of one or more [segment](#)s.

**Selection Box** An instrument for selecting a text-value entry and setting the respective numerical value for the connected variable.

### Signal

- Representation of a [variable](#) measured in a specific [measurement raster](#).
- Generic term for [segment signal](#)s and [operation signal](#)s.

A signal is part of a [signal description set](#) which can be displayed and edited in the working area.

**Signal description set** A group of one or more [signals](#).

A signal description set and its signals can be edited in the working area by means of the [Signal Editor](#). Each signal description set is stored as an [STZ file](#) either in the Signal Description Sets folder or in the Signal Generators folder.

**Signal Editor** A software component to create, configure, display, and manage [signals](#) in [signal description sets](#).

**Signal file** A file that contains the wiring information of a simulator and that is part of the standard dSPACE documentation of dSPACE Simulator Full-Size. Normally, dSPACE generates this file when designing the simulator. Before using a failure simulation system, users can adapt the signal file to their needs.

**Signal generator** An STZ file containing a [signal description set](#) and optional information about the [signal mapping](#), the description of variables, and the real-time platform.

The file is located in the Signal Generators folder and used to generate, download, and control Real-Time Testing sequences, which are executed on the real-time platform to [stimulate](#) model variables in real time.

**Signal Mapping** A [controlbar](#) of the [Signal Editor](#) to map model variables to [signals](#) and [variable aliases](#) of a [signal generator](#).

**Signal Selector** A [controlbar](#) of the [Signal Editor](#). The Signal Selector provides [signals](#) and [segments](#) for arranging and configuring [signal description sets](#) in the working area.

**SIL testing** Abbreviation of *software-in-the-loop testing*.

Simulation and testing of individual software functions, complete virtual ECUs ([V-ECUs](#)), or even V-ECU networks on a local PC or highly parallel in the cloud independently of real-time constraints and real hardware.

**Simulation application** The generic term for [offline simulation application \(OSA\)](#) and [real-time application](#).

**Simulation system** A description of the composition of V-ECU models, environment models, real ECUs, and their interconnections required for simulating the behavior of a system. A simulation system is the basis for the generation of a [simulation application](#) for a given simulator platform.

**Simulation time group** Group of platforms/devices in an experiment whose simulation times are synchronized with each other. If [resynchronization](#) is enabled, ControlDesk synchronizes a simulation time group as a whole, not the single members of the group individually.

**Simulator** A system that imitates the characteristics or behaviors of a selected physical or abstract system.

**Single-processor system** A system that is based on one dSPACE processor or controller board.

**Single-shot instrument** An [instrument](#) that displays an instantaneous value of a connected variable without keeping a value history. In ControlDesk, all instruments except for a [plotter](#) are single-shot instruments. For [platforms](#) that support the [variable observer](#) functionality, you can use single-shot instruments to observe variables.

**Slave application** An application assigned to the [slave DSP](#) of a controller or I/O board. It is usually loaded and started together with the [real-time application](#) running on the corresponding main board.

**Slave DSP** A DSP subsystem installed on a controller or I/O board. Its [slave application](#) can be loaded together with the [real-time application](#) or separately.

**Slider** An instrument (or a value cell type of the [Variable Array](#)) for displaying and setting the value of the connected variable by means of a slide.

**Sound Controller** An instrument for generating sounds to be played.

**Standard axis** An axis with data points that are deposited in the ECU memory. Unlike a [common axis](#), a standard axis is specified within a [curve](#) or [map](#). The parameters of a standard axis can be calibrated, which affects only the related curve or map.

**Start trigger** A [trigger](#) that is used, for example, to start a [measurement raster](#). A [platform trigger](#) can be used as a start trigger.

**Static Text** An instrument for displaying explanations or inscriptions on the layout.

**Steering Controller** An instrument for changing variable values using a game controller device such as a joystick or a steering wheel.

**Stimulation** Writing signals to variables in real-time models during a simulation run.

**Stop trigger** A [trigger](#) that is used, for example, to stop a [measurement raster](#).

**String** A text variable in ASCII format.

Strings are represented by the  symbol.

**Struct** A variable with the struct data type. A struct contains a structured list of variables that can have various data types. In ControlDesk, a struct variable can contain either parameters and value blocks or measurement variables and measurement arrays. ControlDesk supports nested structs, i.e., structs that contain further structs and struct arrays as elements.

Structs are represented by the  symbol.

**Struct array** An array of homogeneous [struct](#)  variables.

Struct arrays are represented by the  symbol.

**STZ file** A ZIP file containing signal descriptions in the STI format. The STZ file can also contain additional MAT files to describe numerical signal data.

**Sub data set** A data set that does not contain the complete set of the parameters of a platform/device.



**Symbol** A symbolic name of a physical address in a MAP file. A symbol can be associated to a variable in the Variable Editor, for example, to support an address updates.

**System variable** A type of variable that represents internal variables of the device or platform hardware and that can be used as measurement signals in ControlDesk to give feedback on the status of the related device or platform hardware. For example, an ECU's power supply status or the simulation state of a dSPACE board can be visualized via system variables.




## T

**Table Editor** An instrument for displaying and setting values of a connected curve, map, value block, or axis in a 2-D, 3-D, and grid view. The Table Editor can also display the values of a measurement array.

The Table Editor can be used for the following variable types:

- [Common axis](#) 
- [Curve](#) 
- [Map](#) 
- [Measurement array](#) 
- [Value block](#) 

**Time cursor** A cursor which is visible at the same time position in the following instruments:

- In all [Time Plotters](#) 
- In all [XY Plotters](#) 
- In all [bus monitoring lists](#) 

You can use the time cursor to view signal values at a specific point in time. If you move the time cursor, all measured signals and the respective parameters are



updated. Instruments and bus monitoring lists display the values that are available at the selected time position.

**Time Plotter** A [plotter instrument](#) for displaying signals that are measured in a time-based raster (time plots).

**Topology** A description of the processor boards belonging to a multiprocessor system and their interconnections via Gigalinks. The topology also contains information on which Gigalink port of each processor board is connected to the Gigalink ports of other processor boards in the multiprocessor system.

Topology information is contained in the real-time application (PPC/x86/RTA) files of the multiprocessor system's processor boards.

**TRC file** A variable description file with information on the variables available in an [environment model](#) running on a dSPACE [platform](#).

**Trigger** A condition for executing an action such as starting and stopping a [measurement raster](#) or a [recorder](#).

The generic term for the following trigger types:

- [Duration trigger](#)
- [Platform trigger](#)
- [Sample count trigger](#)

**Trigger condition** A formula that specifies the condition of a [trigger](#) mathematically.

**Triggered measurement** The measurement of a [measurement raster](#) started by a [platform trigger](#). The data flow between the dSPACE real-time hardware or VEOS and the host PC is not continuous.

## U

**Unassigned data set** A data set that is assigned neither to the working page nor to the reference page of a platform/device. An unassigned data set can be defined as the new working or reference data set. It then replaces the "old" working or reference data set and is written to the corresponding memory page, if one is available on the platform/device.


**Unplugged** A platform/device state defined by the following characteristics:

- The logical connection between ControlDesk and the hardware was interrupted, for example, because the ignition was turned off or the ControlDesk PC and the hardware were disconnected.
- Before the state of a platform/device changes to 'unplugged', the platform/device was in one of the following states:
  - 'Connected'
  - 'Online calibration started'
  - 'Measuring' / 'Recording'


**Tip**



A device for which the connection between ControlDesk and the device hardware currently is interrupted is also set to the 'unplugged' state when you start online calibration if both the following conditions are fulfilled:

- The device's **Start unplugged** property is enabled.
- The **Start online calibration** behavior property is set to 'Ignore differences'.

This is possible for CCP and XCP devices. For details on the two properties listed above, refer to [General Settings Properties \(ControlDesk Platform Management\)](#) .

- If the Automatic Reconnect feature is enabled for a platform/device and if the platform/device is in the 'unplugged' state, ControlDesk periodically tries to re-establish the logical connection for that platform/device.
- Online calibration is impossible. Offline calibration is possible.
- Platform/device configuration is possible.



The 'unplugged' platform/device state is indicated by the  icon.

**Untriggered measurement** The measurement of a [measurement raster](#)  not started by a [platform trigger](#) . The data flow between the dSPACE real-time hardware or VEOS and the host PC is continuous.

**User function** An external function or program that is added to the ControlDesk user interface for quick and easy access during work with ControlDesk.

**User Functions Output** A [controlbar](#)  that provides access to the output of external tools added to the **Automation** ribbon.





## V


**Value block** A [parameter](#)  that consists of a 1- or 2-dimensional array of scalar [parameters](#) .

In variable lists, ControlDesk displays entries for the value block itself and for each array element.

Value blocks are represented by the  symbol.








**Value conversion** The conversion of the original *source values* of variables of an application running on an ECU or dSPACE real-time hardware into the corresponding scaled *converted values*.

**Variable** Any [parameter](#)  or [measurement variable](#)  defined in a [variable description](#) . ControlDesk provides various [instrument](#) s to visualize variables.

**Variable alias** An alias name that lets the user control the property of a [segment](#)  by a model parameter of a real-time application.

**Variable Array** An instrument for calibrating parameters and displaying measurement variable values.

The Variable Array can be used for the following variable types:

- [Measurement](#) 
- [Measurement array](#) 
- [String](#) 
- [Struct](#) 
- [Struct array](#) 
- [Value](#) 
- [Value block](#) 

**Variable connection** The connection of a [variable](#) to an [instrument](#). Via the variable connection, data is exchanged between a variable and the instrument used to measure or calibrate the variable. In other words, variable connections are required to visualize variables in instrument.

**Variable description** A file describing the variables in a simulation application, which are available for measurement, calibration, and stimulation.

**Variable Editor** A tool for viewing, editing, and creating variable descriptions in the ASAM MCD-2MC (A2L) file format. The Variable Editor allows you to create A2L files from scratch, or to import existing A2L files for modification.

**Variable Filter** A variable filter contains the filter configuration of a combined filter, which is used to filter the variable list in the Variables controlbar using a combination of filter conditions.

**Variables controlbar** A [controlbar](#) that provides access to the variables of the currently open experiment.

**V-ECU** Abbreviation of *virtual ECU*.

ECU software that can be executed in a [software-in-the-loop \(SIL\) testing](#) environment such as a local PC or highly parallel in the cloud independently of real-time constraints and real ECU hardware.

**Vehicle information** The [ODX database](#) can contain information for one or more vehicles. Vehicle information data is used for vehicle identification purposes and for access to vehicles. It references the access paths (logical links) to the ECUs.

**VEOS** A [simulator](#) which is part of the PC and allows the user to run an [offline simulation application \(OSA\)](#) without relation to real time.

VEOS Player is the graphical user interface for VEOS.

**VEOS platform** A platform that configures and controls the [offline simulation application \(OSA\)](#) running in [VEOS](#) and that also provides access to the application's [environment VPU](#).

**VEOS Player** An application running on the host PC for editing, configuring and controlling an [offline simulation application \(OSA\)](#) running in VEOS.

**Verbal conversion** A [conversion](#) in which a [conversion table](#) is used to specify the computation of numerical values into strings. The verbal conversion table is used when you switch the value representation from source to converted mode and vice versa.

**Verbal conversion range** A [conversion](#) in which a [conversion table](#) is used to specify the computation of a range of numerical values into strings. The verbal conversion range table is used when you switch the value representation from source to converted mode and vice versa.

**View set** A named configuration of the [controlbar](#)s of ControlDesk. A view set has a default state and a current state that can differ from the default state. The configuration includes the geometry, visibility, and docking or floating state of controlbars.

**Visualization** The representation of [variable](#)s in [instrument](#)s:

- [Measurement variable](#)s are visualized in instruments to view and analyze their time traces.
- [Calibration parameters](#) are visualized in instruments to change their values.

**VPU** Abbreviation of *virtual processing unit*. A VPU is part of an offline simulation application in VEOS. Each VPU runs in a separate process of the PC. VPU is also the generic term for:

- V-ECUs
- Environment VPUs
- Controller VPUs
- Bus VPUs

## W

**Working data set** The data set currently residing in the memory of a platform/device hardware. There can be only one working data set for each calibration platform/device. The working data set is read/write.

**Working page** Memory area containing the parameters of an ECU or prototyping hardware ([memory page](#)). The working page contains the read/write working [data set](#).

If the platform/device also provides a [reference page](#), ControlDesk lets you switch between both pages.

**Writable measurement** A scalar variable that can be measured and calibrated.

**XCP** Abbreviation of *Universal Measurement and Calibration Protocol*. A protocol that is implemented on electronic control units (ECUs) and provides access to ECUs with measurement and calibration systems (MCS) such as ControlDesk.

XCP is based on the *master-slave principle*:

- The ECU is the slave.
- The measurement and calibration system is the master.

The “X” stands for the physical layers for communication between the ECU and the MCS, such as CAN (Controller Area Network) and Ethernet.

The basic features of XCP are:

- ECU parameter calibration (CAL)
- Synchronous data acquisition (DAQ)
- Synchronous data stimulation (STIM), i.e., for bypassing
- ECU flash programming (PGM)

The XCP protocol was developed by ASAM e.V. (Association for Standardisation of Automation and Measuring Systems e.V.). For the protocol specification, refer to <http://www.asam.net>.

The following ControlDesk devices support ECUs with an integrated XCP service:

- [XCP on CAN device](#)
- [XCP on Ethernet device](#)

**XCP on CAN device** A device that provides access to an ECU with XCP connected to the ControlDesk PC via CAN. Using the XCP on CAN device, you can access the ECU for measurement and calibration purposes via XCP (*Universal Measurement and Calibration Protocol*).

**XCP on Ethernet device** A device that provides access to an ECU or [V-ECU](#) with XCP connected to the ControlDesk PC via Ethernet. The XCP on Ethernet device provides access to the ECU/V-ECU via XCP (*Universal Measurement and Calibration Protocol*) for measurement and calibration purposes.

**XIL API EESPort** [Electrical Error Simulation port \(EESPort\)](#)

**XIL API MAPort platform** A platform that provides access to a simulation platform via the ASAM XIL API implementation that is installed on your host PC.

**XY Plotter** A [plotter instrument](#) for displaying signals as functions of other signals.



**A**

AxisTemplate 20

**B**

bookmark 21

**C**

Common Program Data folder 10, 111

**D**

DescriptionCategories 26

DescriptionCategory 28

Documents folder 10, 115

**F**

FileDescription 29

FormatOption 30

FormatOptions 31

FormulaScaling 33

**I**

Instrument Selector 122

**L**

LinearScaling 34

Local Program Data folder 10, 123

**M**

MDFDescription 35

MDFFormatOption 36

Measurement 36

Measurement Data API

AxisTemplate 20

Bookmark 21

DescriptionCategories 26

DescriptionCategory 28

FileDescription 29

FormatOption 30

FormatOptions 31

FormulaScaling 33

GeneralDescription 33

LinearScaling 34

MDFDescription 35

MDFFormatOption 36

Measurement 36

MeasurementDescription 42

Measurements 43

MultiScaling 49

MultiScalingTableEntries 50

MultiScalingTableEntry 53

NumericTableEntries 54

NumericTableEntry 58

NumericTableScaling 59

RationalFunctionScaling 59

RecordingDescription 60

ScalarScaling 61

Scaling 62

Scalings 63

Signal 68

Signals 71

SignalWithSource 79

VerbalTableEntries 79

VerbalTableEntry 83

VerbalTableRangeEntries 83

VerbalTableRangeEntry 87

VerbalTableRangeScaling 88

VerbalTableScaling 88

XAxes 89

XAxis 93

Measurement Data Pool 125

MeasurementDescription 42

MeasurementLoadingOptions 97

Measurements 43

Messages controlbar 126

MultiScalingTableEntries 50

MultiScalingTableEntry 53

**N**

NumericTableEntries 54

NumericTableEntry 58

NumericTableScaling 59

**P**

Platforms/Devices controlbar 130

Project controlbar 131

Project Manager 131

Properties controlbar 131

**R**

RationalFunctionScaling 59

RecordingDescription 60

**S**

ScalarScaling 61

Scaling 62

Scalings 63

Signal 68

Signals 71

SignalWithSource 79

**V**

VerbalTableEntries 79

VerbalTableEntry 83

VerbalTableRangeEntries 83

VerbalTableRangeEntry 87

VerbalTableRangeScaling 88

**X**

XAxes 89

XAxis 93

