Test Automation

Python Modules Quick Reference

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

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

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

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

Contents

About This Reference

Content

This reference provides quick access to information on the available objects, object dependencies, attributes and methods of the Test Automation Python Modules.

It is assumed that you have basic knowledge about Python.

Introduction

Where to go from here

Information in this section

Information in other sections

Structure of the Test Automation Python Modules (Test Automation Python Modules Guide 🕮)

Overview of the Test Automation Python Modules (Test Automation Python Modules Reference 🕮)

General Warning

Danger potential

Using dSPACE software can be dangerous. You must observe the following safety instructions and the relevant instructions in the user documentation.

Improper or negligent use can result in serious personal injury and/or property damage

Using the dSPACE software can have a direct effect on technical systems (electrical, hydraulic, mechanical) connected to it.

The risk of property damage or personal injury also exists when the dSPACE software is controlled via an automation interface. The dSPACE software is then

part of an overall system and may not be visible to the end user. It nevertheless produces a direct effect on the technical system via the controlling application that uses the automation interface.

- Only persons who are qualified to use dSPACE software, and who have been informed of the above dangers and possible consequences, are permitted to use this software.
- All applications where malfunctions or operating errors involve the danger of injury or death must be examined for potential hazards by the user, who must if necessary take additional measures for protection (for example, an emergency off switch).

Liability

It is your responsibility to adhere to instructions and warnings. Any unskilled operation or other improper use of this product in violation of the respective safety instructions, warnings, or other instructions contained in the user documentation constitutes contributory negligence, which may lead to a limitation of liability by dSPACE GmbH, its representatives, agents and regional dSPACE companies, to the point of total exclusion, as the case may be. Any exclusion or limitation of liability according to other applicable regulations, individual agreements, and applicable general terms and conditions remain unaffected.

Data loss during operating system shutdown

The shutdown procedure of Microsoft Windows operating systems causes some required processes to be aborted although they are still being used by dSPACE software. To avoid data loss, the dSPACE software must be terminated manually before a PC shutdown is performed.

Basics on the Test Automation Python Modules Quick Reference

Introduction

This *Test Automation Python Modules Quick Reference* presents every dSPACE-specific Python module of dSPACE Test Automation in tabular form. Object information contained in the Test Automation Python Modules Reference 🕮 is summarized in a set of compact tables, each of which provides a quick overview of the available objects, object dependencies, attributes and methods.

For detailed information on attributes and methods, refer to the Test Automation Python Modules Reference 🕮.

Naming conventions

An overview of the symbols used in this quick reference can be found under Naming Conventions on page 9.

Data types

For a mapping of the data types as used in this quick reference to the Python data types, see Mapping of Data Types on page 9.

Overview

The table gives an overview of the object model provided by the Test Automation Python Modules:

Purpose	See
To communicate via the serial interface.	rs232lib2 Quick Reference on page 11
To exchange data between MATLAB and the Python Interpreter, invoke MATLAB functions or access the MATLAB file format (MAT files).	matlablib2 Quick Reference on page 12

Note

Obsolete functions and attributes are not described in this quick reference. These functions and attributes are described in the *Test Automation Python Modules Reference* 🕮 for compatibility reasons. We strongly recommend you do not use obsolete functions, because they will not be available in future versions.

Packaging and licences

For more information on packages and licences, refer to Introduction to the Python Modules for Test Automation (Test Automation Python Modules Guide 🕮).

Naming Conventions

Naming conventions

The following symbols are used in this quick reference:

Symbol	Description
	Method, function
	Attribute (property, class)
	Level of dependency (0 … 9)

Mapping of Data Types

Data types

The following data types are used in the quick reference:

Quick Reference	Python Data Type
boolean	bool
complex	complex
dictionary	dictionary

Quick Reference	Python Data Type
float	float
float_list	[float,…, float]
int	int
int_list	[int,…, int]
list	list
list_list	[list, …, list]
long	long
object	Arbitrary Python object
string	string
string_list	[string, …, string]
tuple	tuple
tuple_list	[tuple, …, tuple]
various, variant	Different types – the type depends on the related object

Module Reference

Where to go from here

Information in this section

rs232lib2 Quick Reference

Overview

rs232lib2		
	None	Close(int hComRS232)
	int	GetNumInBytes(int hComRS232)
	int	Open(string PcPort)
	string	Read(int hComRS232, int BytesToRead)
	None	SetBuffers(int hComRS232, int InBufferSize, int OutBufferSize)
	None	SetConfig(int hComRS232, int BaudRate, int BitNumber, string Parity, int StopBits)
	None	SetReadTimeout(int hComRS232, int ReadTimeout)
	None	Write(int hComRS232, int ByteToWrite)
	None	WriteString(int hComRS232, string StringToWrite)

Naming conventions For an overview of the symbols used in the object model overview, refer to Naming Conventions on page 9.

Related topics

References

Acquiring Data from External Devices (rs232lib2) (Test Automation Python Modules Reference 🕮)

matlablib2 Quick Reference

Overview

For the usage of the *matlablib2* module, refer to Examples of Using matlablib2 (Test Automation Python Modules Reference 🕮).

matlablib2		
CellArray		
	CellArray(list ListOfDimensionSizes) or (int N1, int N2, int N3)	
int_list	GetDimensions()	
various[1)]	GetItem(int_list IndexList) or (int I1, int I2, int I3)	
int	GetNumberOfDimensions()	
None	SetItem(int_list IndexList, various[1)] Value) or (int I1, int I2, int I3, various[1)] Value)	
Matfile		
	Matfile()	
None	Close()	
None	DeleteArray(string NameOfArray)	
various[1)]	GetArray(string NameOfArray)	
string_list	GetDir()	
None	Load()	
None	Open(string FileName, string Mode, boolean ConvertToDouble)	
None	PutArray(string NameOfArray, various[1)] Value)	
None	PutArrayAsGlobal(string NameOfArray, various[1)] Value)	
string	Whos()	

matlablib2		
Matlab		
	Matlab()	
	tuple_list (string, int)	ConnectedMATLABInstallations
	string	ExecutablePath
	tuple (int, string, string)	IsMUMatlabOpen
	int	ProcessArchitecture
	int	ProcessID
	string	Version
	int	WatchdogMethod
	None	Close(boolean DisconnectOnly)
	None	Execute(string Command)
	various[1)]	GetArray(string ArrayName)
	string	GetOutputs()
	boolean	IsAlive()
	None	Open(boolean ConvertToDouble, int StartNewMLInst, int OpenFlags, string StartCommands, string MLStartupDir, string MLInstallDir)
	None	PutArray(string ArrayName, various[1)] Value)

[1)] The mapping between the MATLAB array class and the Python type is described in Interfacing MATLAB (matlablib2) (Test Automation Python Modules Reference 🕮).

Naming conventions For an overview of the symbols used in the object model overview, refer to Naming Conventions on page 9.

Related topics

References

Interfacing MATLAB (matlablib2) (Test Automation Python Modules Reference 🕮)