Dummy Robot provider

Version 1.2.0

User's guide

April 14, 2021

Remarks:

[Revision History]

Version	Date	Content
1.0.0	2015-09-01	First edition.
1.0.1	2016-12-26	Delete summary installer related description.
1.1.0	2018-09-13	Added system variable of CaoController class. {@ERROR_CODE, @BUSY_STATUS, @NORMAL_STATUS, @CURRENT_DATETIME, @RANDOM} Added system variable of CaoTask class. {@ START, @ STOP}
1.1.1	2020-12-01	Added TASK variable. {@ELAPSED_TIME, @STATUS} Changed @START variable. {@START = n (0 to N-1)} Supports OneCycle operation.
1.2.0	2021-04-14	Supports multiple startups.

Contents

1. Introduction

This is a user's guide of Dummy Robot provider.

Dummy Robot provider allows to simulate the robot values acquisition by using a virtual robot without connecting to a real robot.

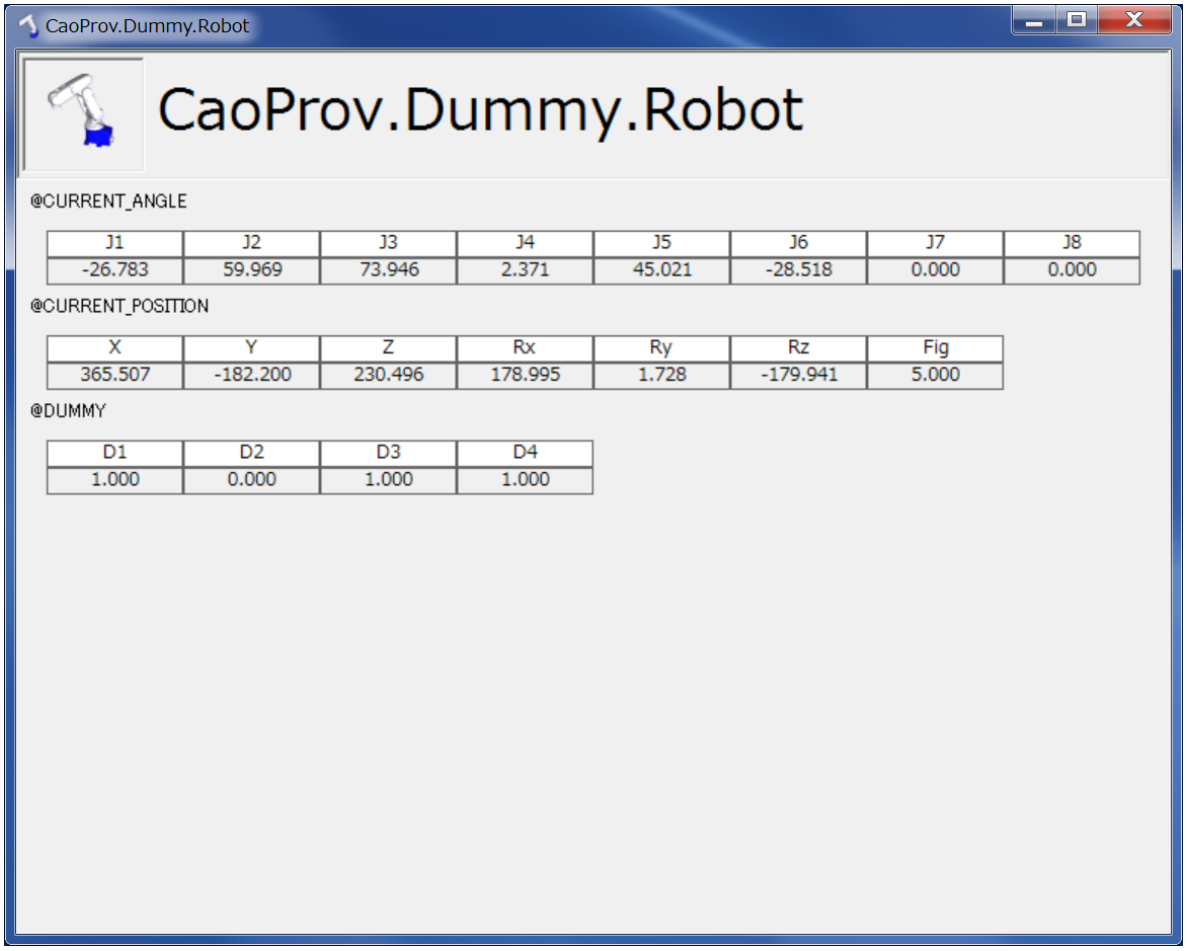

Figure 1 Dummy provider screen for Robot

2. Overview of provider

2.1. Overview

Dummy Robot provider is a CAO provider that allows to obtain data and to check result by connecting to a dummy robot.

The file type is DLL (Dynamic Link Library) and is dynamically loaded by CAO engine when it is used.

Table2-1 Dummy Robot provider

File name	CaoProvDummyRobot.dll
ProgID	CaoProv.Dummy.Robot

2.2. Method and Properties

2.2.1. CaoWorkspace::AddController method

Dummy Robot provider establishes a connection to a virtual controller by referring parameters that have been passed at the AddController method execution.

Format AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPCName:BSTR> [,<bstrOption:BSTR>])

<bstrCtrlName> : [in] Controller name
Specify a unique arbitrary string for each connection.
* An error (0x80000205) occurs if the same name is specified by a different application or computer.
If an empty string ("") is entered, the CAO engine automatically assigns a unique controller name.

<bstrProvName> : [in] Option character string
Fixed value = "CaoProv. Dummy.Robot"

<bstrPcName> [in] Computer name to execute provider
For a local connection, enter an empty string ("").
To establish a remote connection, specify a target computer name.

<bstrOption> [in] Option character string

The list specified for an optional character string is shown as follows.

Table2-2 Optional character string of CaoWorkspace::AddController

INI = <INI file name>	Specify the INI file to use. Default It is treated as Robot.ini specification.
@Multi	Unspecified : Does not use multiple boots. If multiple dummy robot providers are running, the status of the dummy robot generated without checking first is referenced. Specified : Use multiple boots. Start a dummy robot that has an independent state for each controller. (default : Unspecified)

2.2.2. CaoController::AddRobot method

A CaoRobot object is retrieved by calling AddRobot method. For an argument of AddRobot method on CaoController class, specify a robot name (BSTR type). "Robot name" specified here is an arbitrary string and there is no restriction for naming. For example, you can name it AddRobot ("Robot1").

Format AddTask(<bstrName:BSTR > [,<bstrOption:BSTR>])

<bstrName>	:	[in] Robot name
<bstrOption>		[in] Option character string (not used)

2.2.3. CaoController::AddTask method

For an argument of AddTask method on CaoController class, specify a task name (BSTR type). "Task name" specified here should be a task name that can be obtained by CaoController::get_TaskNames property.

Format AddTask(<bstrName:BSTR > [,<bstrOption:BSTR>])

<bstrName> : [in] Task name

<bstrOption> [in] Option character string (not used)

2.2.4. CaoController::AddVariable method

For an argument of AddVariable method on CaoController class, specify a variable name (BSTR type). "Variable name" specified here should be a task name that can be obtained by CaoController::get_VariableNames property.

Format AddVariable(<bstrName:BSTR > [,<bstrOption:BSTR>])

<bstrName> : [in] Variable name

<bstrOption> [in] Option character string (not used)

2.2.5. CaoController::get_TaskNames property

Obtain the list of task names that can be specified by CaoController::AddTask method.

Format TaskNames([<bstrOption:BSTR>])

<bstrOption> [in] Option character string (not used)

2.2.6. CaoController::get_VariableNames property

Obtain the list of variable names that can be specified by CaoController::AddVariable method.

Format VariableNames([<bstrOption:BSTR>])

<bstrOption> [in] Option character string (not used)

2.2.7. CaoRobot::AddVariable method

For an argument of AddVariable method on CaoRobot class, specifies a variable name. “Variable name” specified here should be a variable name that can be obtained by CaoRobot::get_VariableNames property. For details about how to define a variable, refer to “2.4 Setting of Ini file”.

Format AddVariable(<bstrName:BSTR > [,<bstrOption:BSTR>])

<bstrName> : [in] Variable name (Section name that is defined by the form

of ”@<arbitrary data classification>” in the ini file)

<bstrOption> [in] Option character string (not used)

2.2.8. CaoRobot::get_VariableNames property

Obtain a variable name list that can be specified by CaoRobot::AddVariable method. “Variable name” specified here should be a variable name that has been defined in the ini file.

For the details about how to define a variable , refer to “2.4 Setting of Ini file”.

Format VariableNames([<bstrOption:BSTR>])

<bstrOption> [in] Option character string (not used)

2.2.9. CaoTask::AddVariable method

For an argument of AddVariable method on CaoTask class, specify a variable name (BSTR type).

See Table 2-4 for a list of implemented variables.

2.2.10. CaoTask::get_VariableNames property

Obtain a variable name list that can be specified by CaoTask::AddVariable method.

2.2.11. CaoTask::Start method

Run a program that corresponds to an object.

The following shows the argument specifications of Start.

Format Start(<lMode:long > [,<bstrOption:BSTR>])

<IMode>	:	[in] Mode (an arbitrary long value)
<bstrOption>		[in] Option character string (not used)

2.2.12. CaoTask::Stop method

Stop a program that corresponds to an object.

The following shows the argument specifications of Stop.

Format Stop(<lMode:long > [,<bstrOption:BSTR>])

<IMode>	:	[in] Mode (an arbitrary long value)
<bstrOption>		[in] Option character string (not used)

2.2.13. CaoVariable::get_Value property

Obtain a variable value corresponding to an object. The variable’s value obtained here is a variable value (Double type array) that has been specified at the execution of AddVariable on CaoRobot class.

2.3. Variable list

2.3.1. Controller class

Table 2-3 Controller class variable list

Variable identifier	Data type	Explanation	Attribute	
			get	put
@CURRENT_DATETIME	VT_DATE	Current time	√	-
@BUSY_STATUS	VT_BOOL	true = During program operation, false = Program halted	√	-
@NORMAL_STATUS	VT_BOOL	true = normal, false = abnormal (error occurring) (Due to dummy operation, always true)	√	-
@ERROR_CODE	VT_I4	Acquires the number of the error occurring as a decimal value. If no error occurred, 0 is returned. (Due to dummy operation, always 0)	√	-
@RANDOM	VT_R8	Returns a random value between 0.0 and 1.0.	√	-

2.3.2. Task class

Table 2-4 Task class variable list

Variable identifier	Data type	Explanation	Attribute	
			get	put
@START	VT_BOOL	Task execution. When get (get) = True: Running, False: Stopping Setting (put) = By writing n (0 to N-1), execution starts with the n-th CSV of the datafile.	√	√
@STOP	VT_BOOL	Task stop. When get (get) = True: Stopped, False: Running When setting (put) = stops independent of value.	√	√
@ELAPSED_TIME	VT_R8	Elapsed time after start (ms)	√	-
@STATUS	VT_I4	Task status (0: stopped, 1: running)	√	-

2.4. Setting of Ini file

You can set/change the behavior of the virtual robot motion in the Data section of the ini file. The robot motion data used here should be a CSV file.

To configure robot motion data, create a new section with the format of ”@<arbitrary data classification>”, and then specify it.

The ini file that lists the setup is stored in the following path.

"<ORiN2 SDK installation folder>¥CAO¥ProviderLib¥Dummy¥Bin¥Robot.ini"

[data]
datapath=<Path of a folder that contains robot motion data>
interval=<Interval before displaying the next robot data (ms)>
tasklist=<Task name>[,<Task name>]

[@<arbitrary data classification>]
datafile=<CSV file name that stores robot motion data>
columns=<Motion destination position name>[,< Motion destination position name >]
loop=<Number of executions>

2.4.1. Sample file

By the setting of the following sample file, a CSV file in the “Robot.Data” folder under the operation environment will be the robot motion data. In this setting, a robot moves each motion destination position by 100ms.

Based on the "@CURRENT_ANGLE" section, the robot motion will refer data stored in "angle.csv" file. The motion destination positions will be named in order from the left (starting from J1, and end with J8).

Robot.ini

```
[data]
datapath=.¥Robot.Data
interval=100

[@CURRENT_ANGLE]
datafile=angle.csv
columns=J1,J2,J3,J4,J5,J6,J7,J8
```

angle.csv

```
-26.783,38.125,82.602,1.732,58.197,-27.682,0,0
-26.704,38.125,82.602,1.729,58.196,-27.602,0,0
-26.443,38.124,82.603,1.721,58.192,-27.337,0,0
```