CaoTester2

Version 1.0.3

User's Guide

June 17, 2022

[Remark]

[Revision History]

Version	Date	Description
1.0.0	2020-07-01	First edition
1.0.1	2020-09-11	Fixed the loading, saving and closing process of layout.
1.0.2	2021-04-14	Fixed an error in the log when getting the value of an empty file. Fixed OnMessage event processing. Fixed processing when acquiring image data.
1.0.3	2022-06-17	Changed the search function in the drop-down list of controller name and provider name in Workspace window.

Table of Contents

1. Introduction

This document is a user's guide to the comprehensive testing tool CaoTester2 that implements the CAO interfaces defined in ORiN2 SDKs.

CaoTester2 allows you to add and remove providers' variables, set and retrieve values, and execute extended commands.

This document describes the features of this CaoTester2.

2. Screen Description

2.1. Main Screen

The main window of CaoTester2 consists of the following windows.

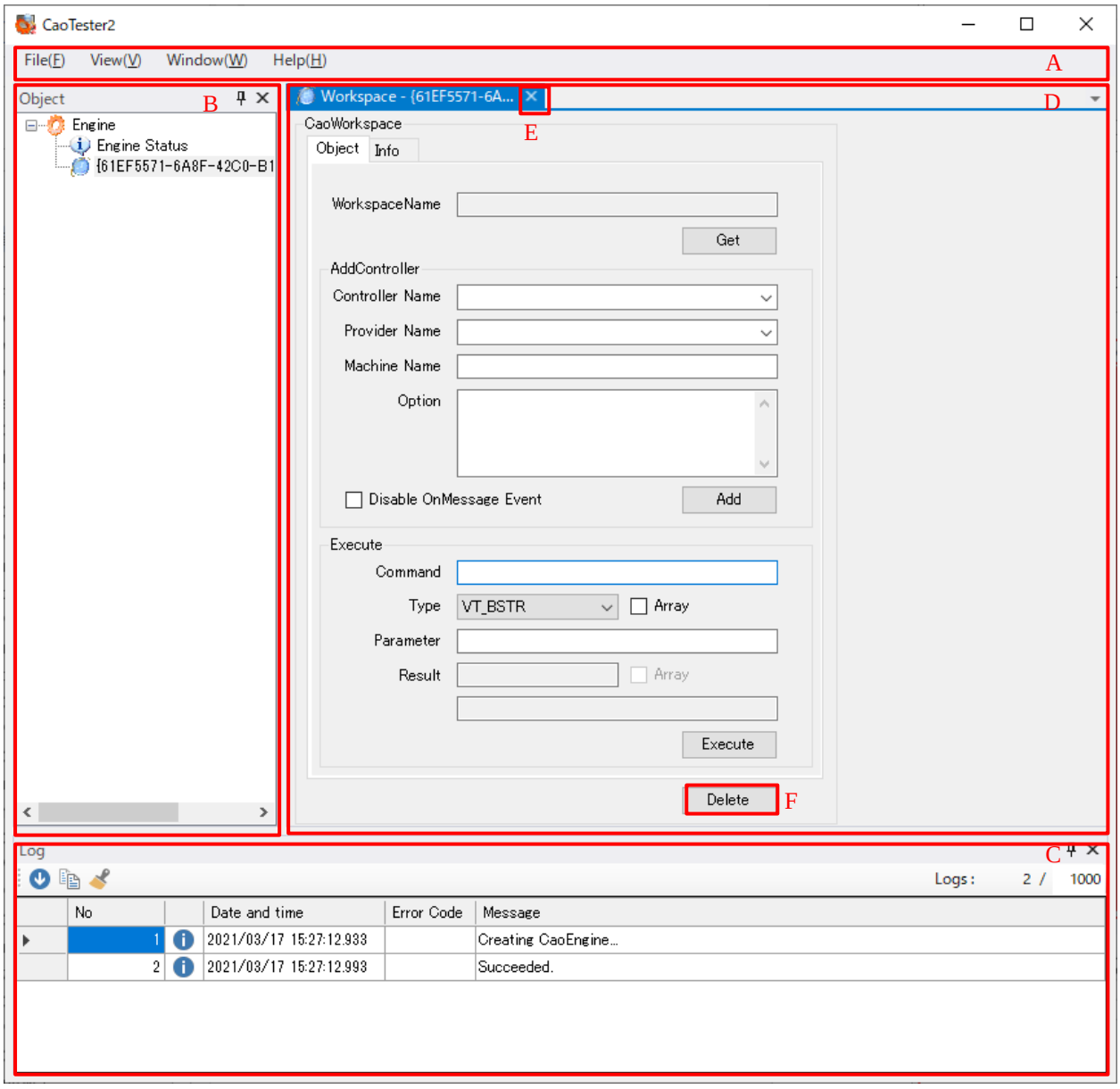


Fig. 2-1 CaoTester2 Main Window

A: Menu bar

Each menu is displayed. Refer to 2.2 for details.

B: Object window

The objects that you created or deleted are displayed in a tree. Refer to 2.14 for details.

C: Log Window

The operation you performed and the results are logged. Refer to 2.15 for details.

D: Item Window

You can set values, obtain values, and perform other operations on the created object.

The Close button in the Items window of E hides the window without destroying the object.

The F Delete button discards objects and windows.

The Item window contains the windows in Table 2-1 Item Window List

Table 2-1 Item Window List

Window name	See:
Workspace window	2.3
Controller window	2.4
Variable window	2.5
Robot window	2.6
File window	2.7
Task window	2.8
Extension window	2.9
Command window	2.10
Message window	2.11
Engine Status window	2.12
Engine window	2.13

When writing a value to a variable or executing a Execute, you must specify the type information of the value to write to clarify the value to write. See Appendix A for parameter type information.

Figure 2-2 Parameter type information

When the result of acquiring a variable or performing a Execute can be recognized as image data, a thumbnail image is displayed on the left side of the acquired result. Clicking a thumbnail image displays Image window. Refer to 2.16 for details.

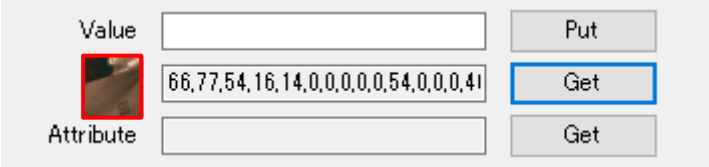


Figure 2-3: Thumbnail display for acquiring image data

2.2. Menu Structure

This section describes CaoTester2 menu items.

2.2.1. File Menu

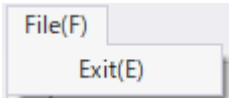


Fig. 2-4 File Menu

Exit

Exit CaoTester2.

2.2.2. View Menu

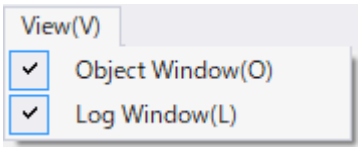


Fig. 2-5 View Menu

Object Window

Shows or hides Object Window.

Log Window

Shows or hides Log Window.

2.2.3. Window Menu

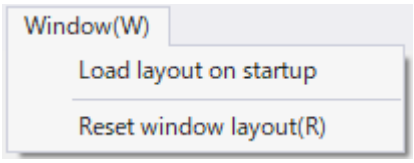


Fig. 2-6 Window Menu

Load layout on Startup

Toggles On/Off of the last window layout info load at startup.

Reset window layout

Resets the window layout information.

2.2.4. Window Menu

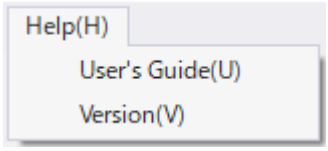


Fig. 2-7 Help Menu

User's Guide

Open CaoTester2 documentation (this document).

Version

Displays the version of CaoTester2.

2.3. Workspace window

In Workspace window, you can create controller objects and execute extended commands.

2.3.1. Object Tabs

Object tab allows you to create controller objects and execute extended commands.

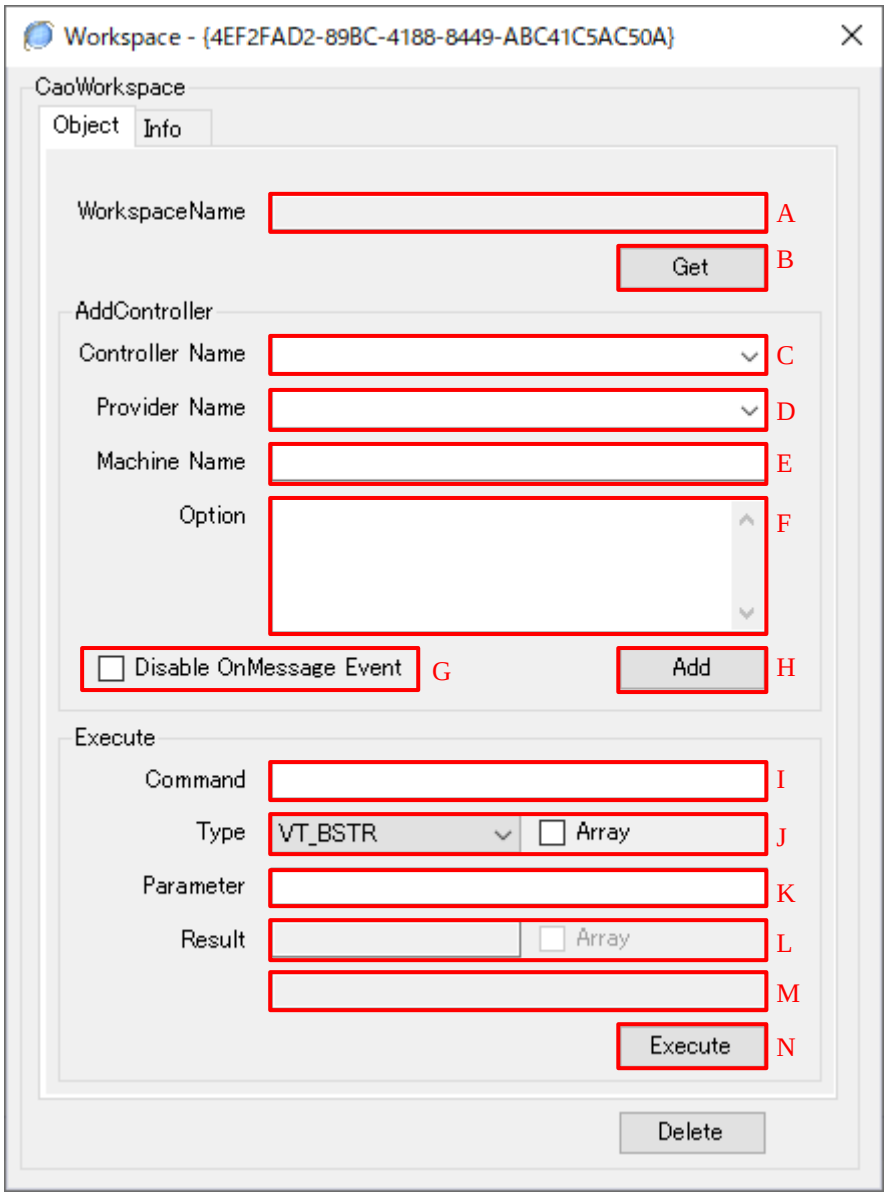


Diagram 2-8 Workspace Window Object

A: Results of Retrieving Workspace Names

Stores the result of getting the workspace name for B.

B: Get workspace name

Gets the workspace name and stores the execution result in A.

C: Controller name

This is used as the controller-name when H is AddController.

D: Provider name

Used as the provider name when AddController H.

E: Machine name

Used as the machine name when H is AddController.

F: Option

Used as an optional AddController of H.

G: Event Enable Flag

Give the @EventEnable to AddController run-time option of H.

The "@EventEnable=True" is given when in the check status, and the "@EventEnable=False" is given when in the unchecked state.

If @EventEnable=True, OnMessage will be received.

H: Running AddController

Use C, D, E, F, and G to execute CaoWorkspace.AddController.

The example in Figure 2-8 looks like this:

AddContoller("DS", "CaoProv.DataStore", "", "")

I: Command name

Used as the commandname when executing N's Execute.

J: Parameter type information

Used as parameter type info when N Execute is executed.

K: Parameter string

Used as a parameter string when Execute N is executed.

L: Type Info of Execute Run Result

Stores the type of N Execute run result.

M: Outcome of Execute

Stores N Execute executions.

N: Running Execute

Execute Execute using I, J, and K.

The result type and value at normal termination are stored in L and M.

2.3.2. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.4. Controller window

In Controller window, you can create variables, robots, files, tasks, extension boards, command objects, and receive messages.

2.4.1. Object Tabs

Object tab allows you to create variables, robots, files, tasks, extension boards, command objects, execute extended commands, and get messages.

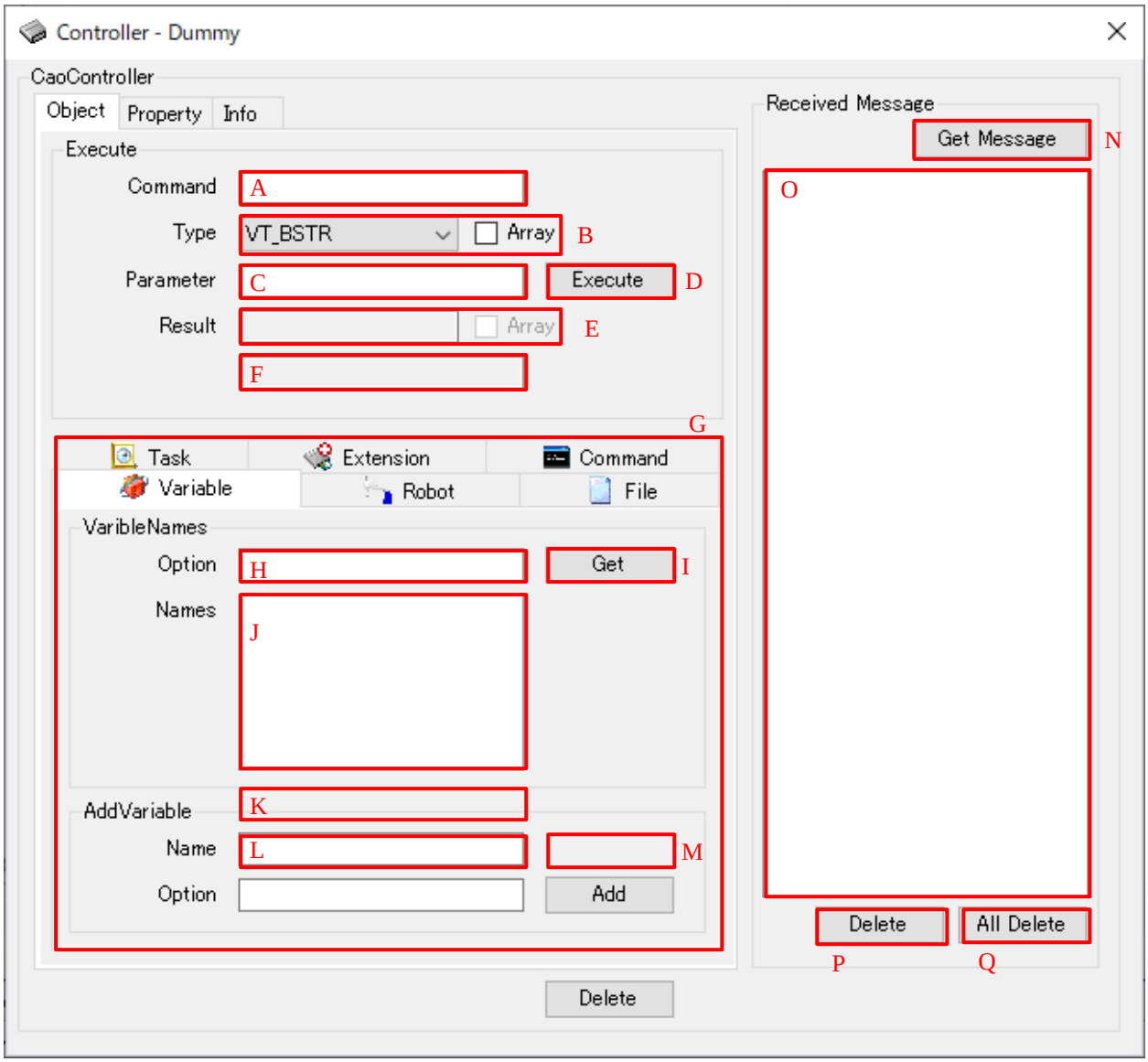


Diagram 2-9 Controller Window Object

A: Command name

Used as the commandname when executing Execute of D.

B: Parameter type information

Used as the type of the parameter when Execute of D is executed.

C: Parameter string

Used as the parameter string when Execute of D is executed.

D: Running Execute

Performs a Execute using A, B, and C.

The result type and value at normal termination are stored in E and F.

E: Type Info of Execute Run Result

Stores the type of Execute executed by D.

F: Outcome of Execute

Stores Execute result of D.

G: Object creation group

The group in which to create the object.

An object creation group is a where all variables, robots, files, tasks, extension boards, and command objects are in the same layout.[1]

H: VariableNames optional [1]

Used as an optional option when executing VariableNames of I.

I: VariableNames run [1]

Stores VariableNames executionresults.

J: VariableNames run [1]

Stores the I VariableNames executionresults.

K: Object-name to be created [1]

It is used by the name of the variable when AddVariable of M is executed.

L: Optional of the object to be created [1]

Used as an optional option when executing M's AddVariable.

M: AddVariable run [1]

[1] It is described as an example of creating a variable object as a representative.

Execute AddVariable using K,L to create the object.

N: Get Message

Retrieves a message from a message queue in the engine.

It can be acquired only when OnMessage event function at AddController is disabled.

When OnMessage event function is enabled, messages are automatically added to O.

O: Received message list

Displays the list of acquired messages.

You can check Message by double-clicking the message. Refer to 2.11 for details.

P: Deleting Messages

Deletes the selected message.

Q: Delete all received messages

Deletes all received messages.

2.4.2. Property Tabs

On Property tab, you can set and retrieve properties of controller objects.

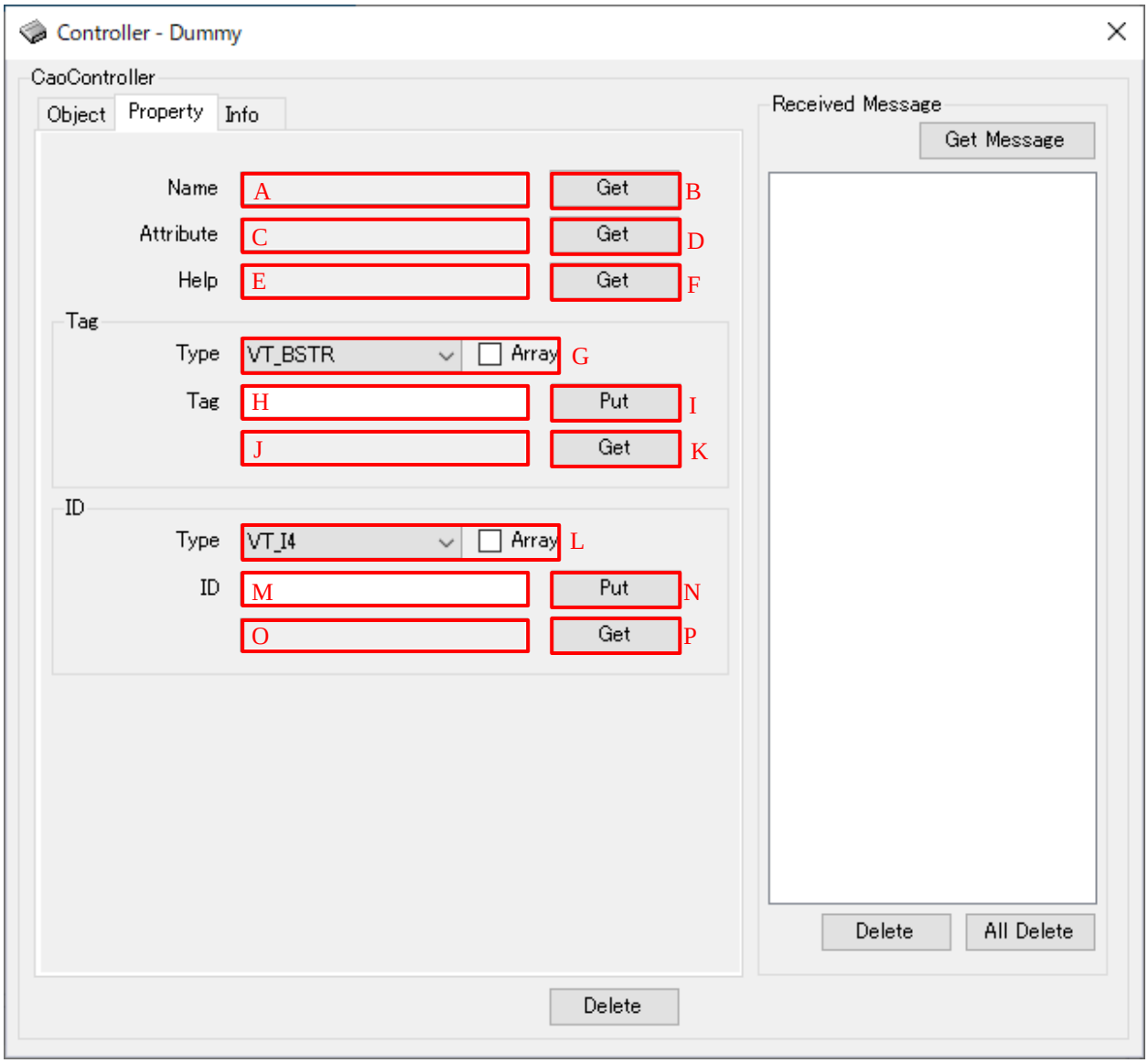


Diagram 2-10 Controller Window Property

A: Result of acquiring the controller name

Stores the execution result of acquiring the controller name of B.

B: Get Controller Name

Acquires the controller name and stores the execution result in A.

C: Results of getting attributes

Stores the result of acquiring the attribute of D.

D: Retrieving Attributes

Retrieves the attribute and stores the execution result in C.

E: Results of getting help

Stores the result of getting help for F.

F: Getting Help

Get help and store the execution result in E.

G: Tag type information

Used as parameter type information when I put_Tag is executed.

Stored as K get_Tag execution result type information.

H: Tag data string

Used as tag data when I's put_Tag is executed.

I: Tag Settings

Run put_Tag.

J: Results of tag acquisition

Stores the execution result of get_Tag of K.

K: Retrieving Tags

Execute get_Tag and the execution result is stored in J.

Type information is stored in I.

L: ID type information

N's put_ID is used as parameter type information at runtime.

Stored as type information of the get_ID execution result of P.

M: ID string

Used as tag data when executing N put_ID.

N: ID setting

Run put_ID.

O: Results of obtaining the ID

Stores the execution result of the get_ID of P.

P: Get ID

Execute get_ID and the execution result is stored in O.

Type information is stored in L.

2.4.3. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.5. Variable window

In Variable window, you can set and retrieve values for variable objects.

2.5.1. Property Tabs

Property tab allows you to set, retrieve, set, and retrieve the values of variable objects.

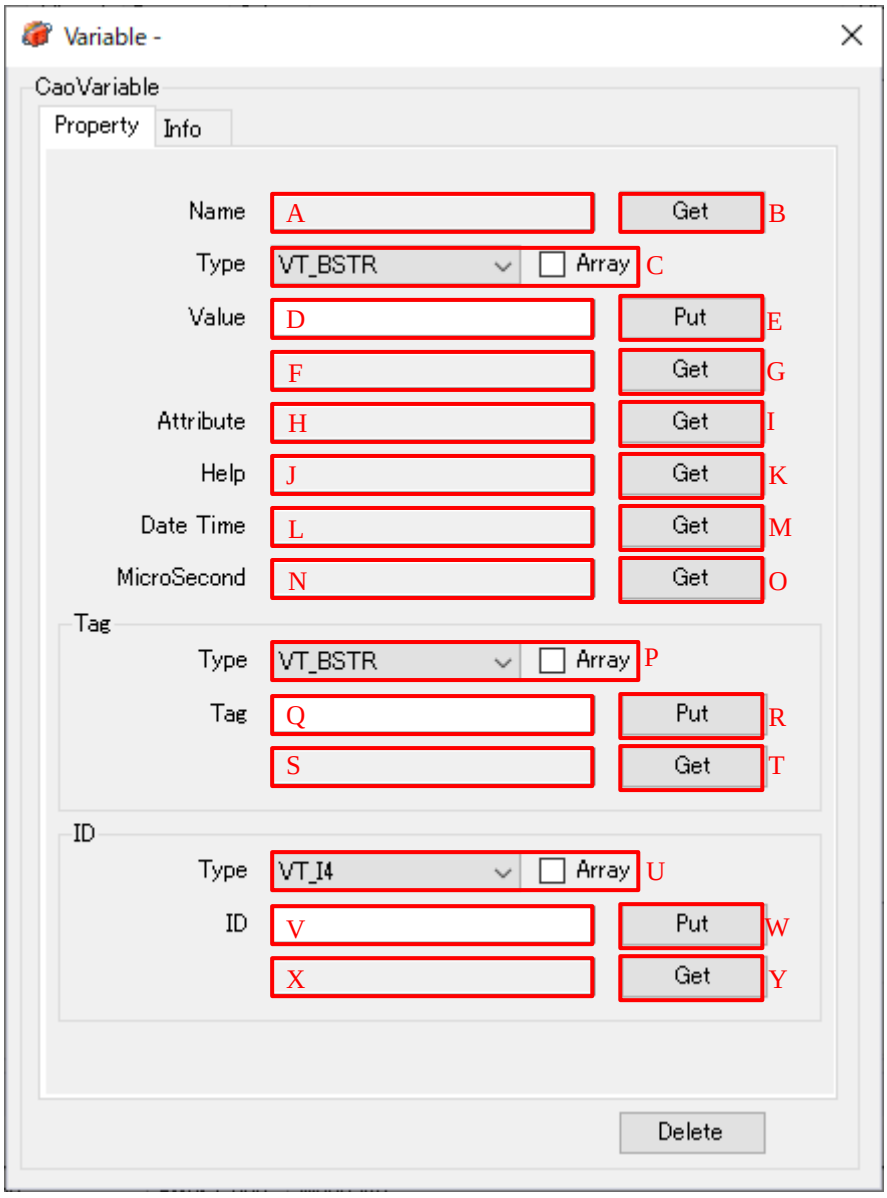


Diagram 2-11 Variable Window Property

A: Results of Retrieving Variable Names

Stores the result of acquiring the variable name of B.

B: Retrieving Variable Names

Gets the variable name and stores the execution result in A.

C: Variable Value Type Information

Used as type information for E's put_Value parameter at runtime.

Stored as type information of get_Value execution result of G.

D: Variable value data string

Used as tag data when executing E put_Value.

E: To set variable values

Run put_Value.

F: Results of obtaining variable values

Stores the result string of the execution result of get_Value of G.

G: Retrieving Variable Values

Execute get_Value and store the execution result in F.

Type information is stored in C.

H: Results of getting attributes

Stores the result of acquiring the attribute of I.

I: Retrieving Attributes

Retrieves the attribute and stores the execution result in H.

J: Results of getting help

Stores the result of getting the help of K.

K: Getting Help

Get help and store the execution result in J.

L: Acquisition result of time stamp (date and time)

Stores the result of acquiring the time stamp (date and time) of M.

M: Obtaining the time stamp (date and time)

Retrieves the time stamp (date and time) and stores the execution result in L.

N: Timestamp (in microseconds) Acquisition Result

Stores the result of acquiring the O timestamp (in microseconds).

O: Obtaining the Timestamp (Microseconds)

Retrieves the time stamp (in microseconds) and stores the execution result in N.

P: Tag type information

Used as parameter type information when executing R put_Tag.

Stored as T get_Tag execution result type information.

Q: Tag data string

Used as tag data when executing R put_Tag.

R: Tag Settings

Run put_Tag.

S: Results of tag acquisition

Stores the execution result of get_Tag of T.

T: Retrieving Tags

Execute get_Tag and store the execution result in S.

Type information is stored in P.

U: ID type information

Used as parameter type information when executing W put_ID.

Stored as Y get_ID execution result type information.

V: ID string

Used as tag data when executing W put_ID.

W: ID setting

Run put_ID.

X: Results of obtaining the ID

Stores the execution result of get_ID of Y.

Y: Get ID

Execute get_ID and the execution result is stored in X.

Type information is stored in U.

2.5.2. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.6. Robot window

In Robot window, you can create variable objects, set and get robot variable values, manipulate robots, and execute extended commands.

2.6.1. Object Tabs

Object tab allows you to create variable objects and execute extended commands.

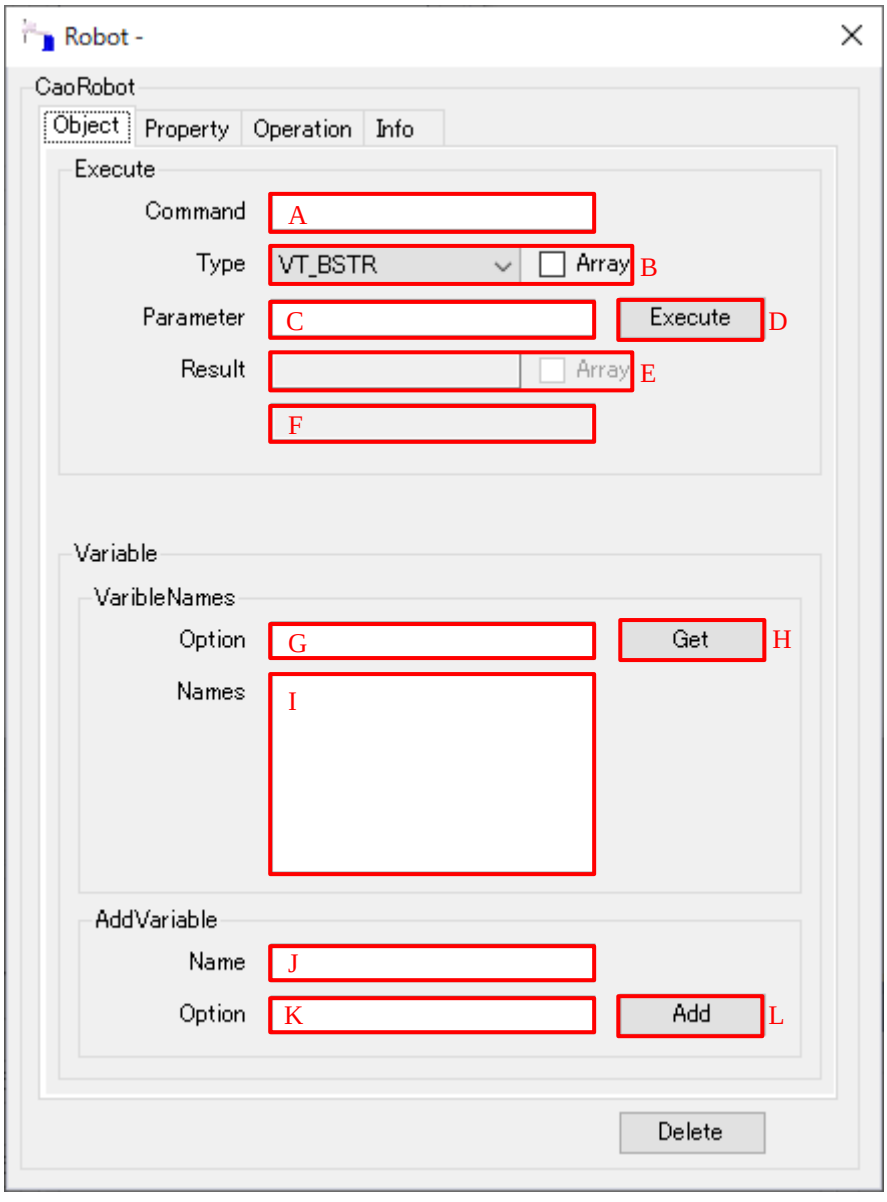


Diagram 2-12 Robot Window Object

A: Command name

Used as the commandname when executing Execute of D.

B: Parameter type information

Used as the type of the parameter when Execute of D is executed.

C: Parameter string

Used as the parameter string when Execute of D is executed.

D: Running Execute

Performs a Execute using A, B, and C.

The result type and value at normal termination are stored in E and F.

E: Type Info of Execute Run Result

Stores the type of Execute executed by D.

F: Outcome of Execute

Stores Execute result of D.

G: VariableNames Optional

Used as an optional VariableNames runtime for H.

H: Running VariableNames

Run VariableNames.

The execution result is stored in I.

I: Outcome of VariableNames

Stores the H VariableNames executionresult.

J: Object name to create

It is used by the variable-name at the time of executing AddVariable of L.

K: Options for the object being created

Used as an optional AddVariable runtime for L.

L: Running AddVariable

Execute AddVariable using the values of J,K to create the object.

2.6.2. Property Tabs

On Property tab, you can set and retrieve properties of the robot object.

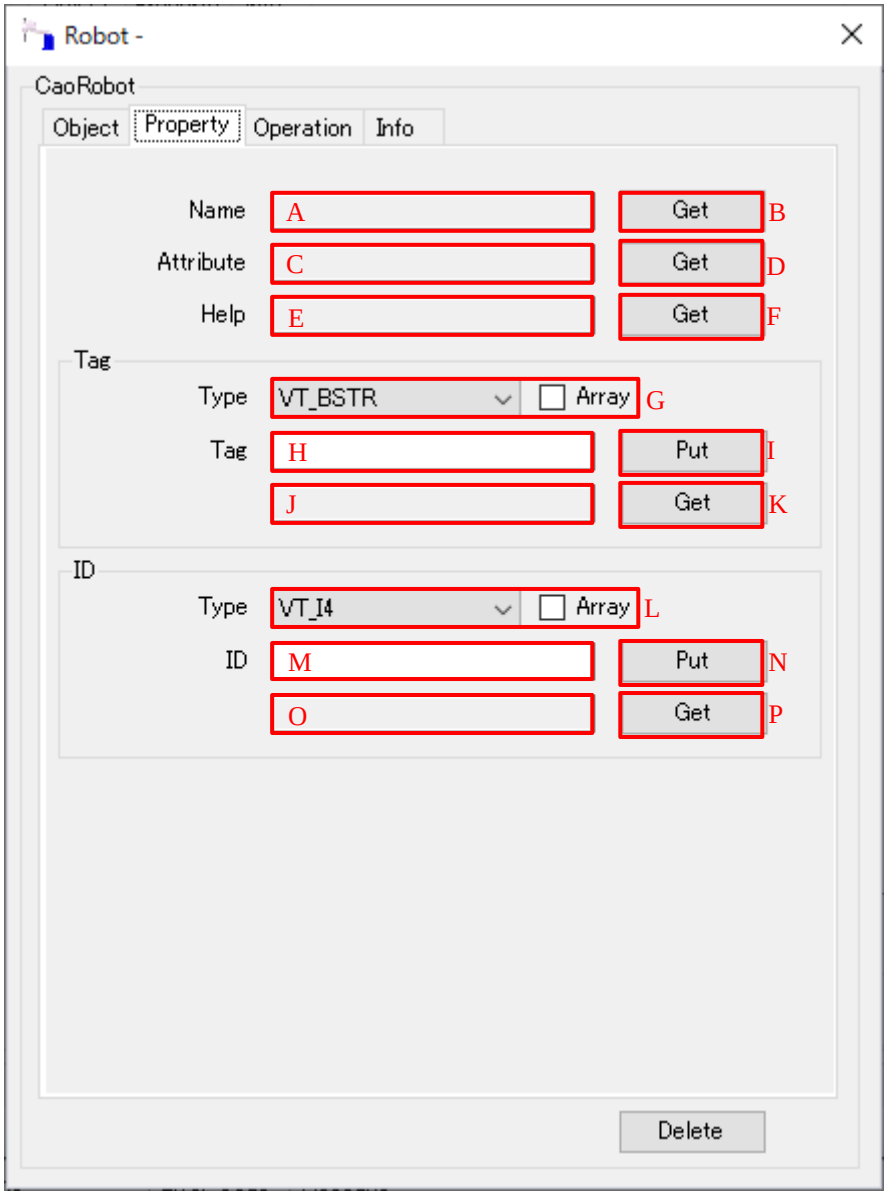


Diagram 2-13 Robot Window Property

A: Result of acquiring robot name

Stores the result of acquiring the robot name of B.

B: Get Robot Name

Acquires the robot name and stores the execution result in A.

C: Results of getting attributes

Stores the result of acquiring the attribute of D.

D: Retrieving Attributes

Retrieves the attribute and stores the execution result in C.

E: Results of getting help

Stores the result of getting help for F.

F: Getting Help

Get help and store the execution result in E.

G: Tag type information

Used as parameter type information when I put_Tag is executed.

Stored as K get_Tag execution result type information.

H: Tag data string

Used as tag data when I's put_Tag is executed.

I: Tag Settings

Run put_Tag.

J: Results of tag acquisition

Stores the execution result of get_Tag of K.

K: Retrieving Tags

Execute get_Tag and the execution result is stored in J.

Type information is stored in G.

L: ID type information

N's put_ID is used as parameter type information at runtime.

Stored as type information of the get_ID execution result of P.

M: ID string

Used as tag data when executing N put_ID.

N: ID setting

Run put_ID.

O: Results of obtaining the ID

Stores the execution result of the get_ID of P.

P: Get ID

Execute get_ID and the execution result is stored in O.

Type information is stored in L.

2.6.3. Operation Tabs

Operation tab allows you to set and manipulate the speeds of the robotics.

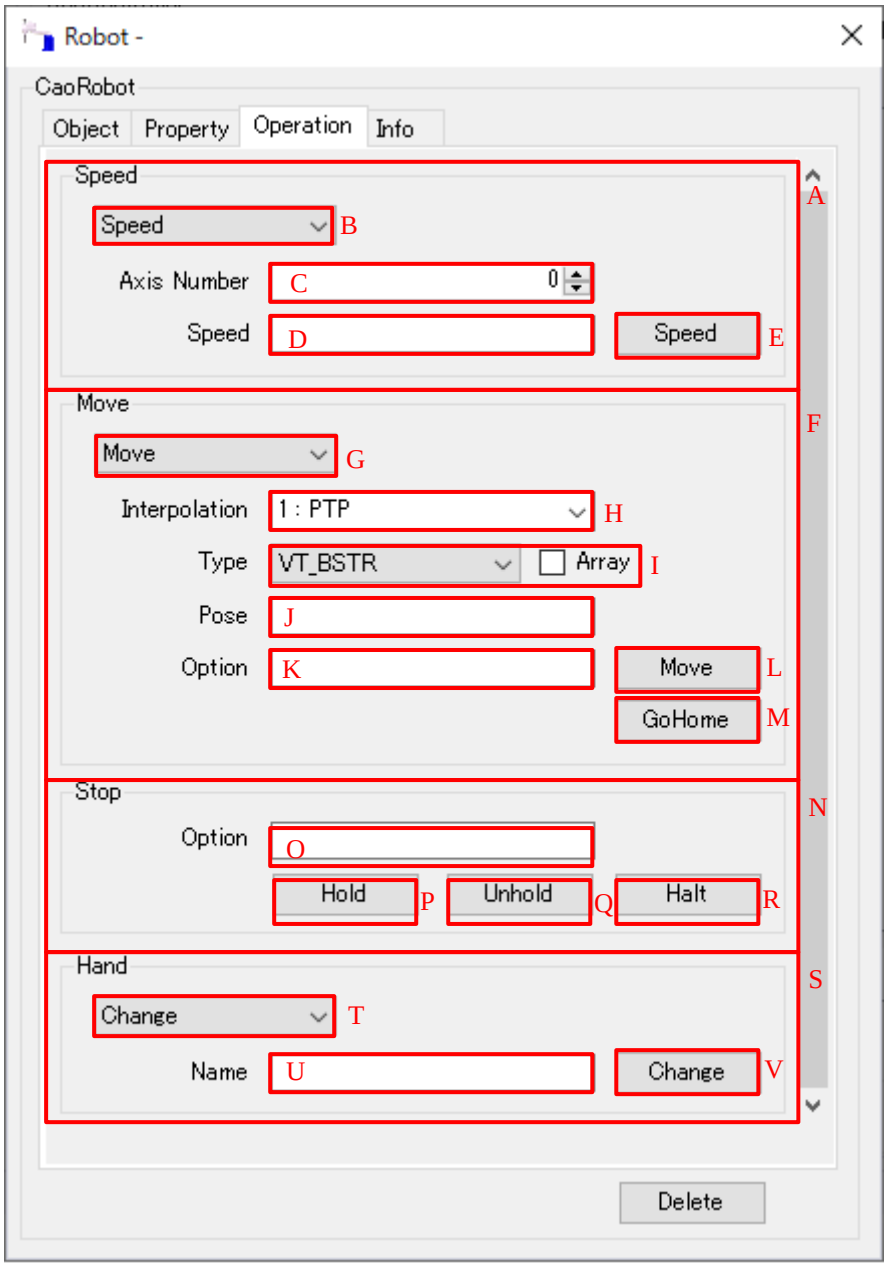


Diagram 2-14 Robot Window Operation

A: Robot Speed Setting Group

This group is used to set the robot speed.

Determine the setting method by type B.

B: Speed setting type

Determine the speed setting method.

A that allows you to select a Accelerate,Speed.[2]

C: Axis number

Used as the first parameter when executing Speed of E.

D: Speed

Used as the second parameter when Speed of E is executed.

E: Running Speed

Execute Speed using C and D.

F: Robot mobile group

A group for moving the robot.

Determine the setting method by G type.

G: Robot movement type

Decide how to move.

A that allows you to select a Drive,Move,Rotate.[3]

H: Completion specification

Used as the first parameter when executing Move of L.

I: Pause column type information

Used as the type of the pose column of the second parameter when Move of L is executed.

J: Pause Row

Used as the pose column of the second parameter when Move of L is executed.

K: Motion Options

Used as the third parameter when executing Move of L.

L: Running Move

Use H, I, J, and K to execute Move.

M: Running GoHome

[2] Speed will be explained as a representative.
[3] Move will be explained as a representative.

Run GoHome.

N: Robot Stop Group

This group is used to stop the robot.

O: Stop option

Used as options for P, Q, and R.

P: Running Hold

Run Hold using O.

Q: Running Unhold

Run Unhold using O.

R: Running Halt

Run Halt using O.

S: Robot hand operation group

A group that performs robot hand operations.

Determine the setting method by T type.

T: Robot hand operation type

Determine how to operate the hands.

A that allows you to select a Change,Chuck/Unchuck.[4]

U: Hand name

Used as the first parameter when Change of V is executed.

V: Running Change

Run Change using U.

2.6.4. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

[4] Change will be explained as a representative.

2.7. File window

In File window, you can create file objects, create file variables, set and retrieve file values, manipulate files, and execute extended commands.

2.7.1. Object Tabs

Object tab allows you to create variables, file objects, and execute extended commands.

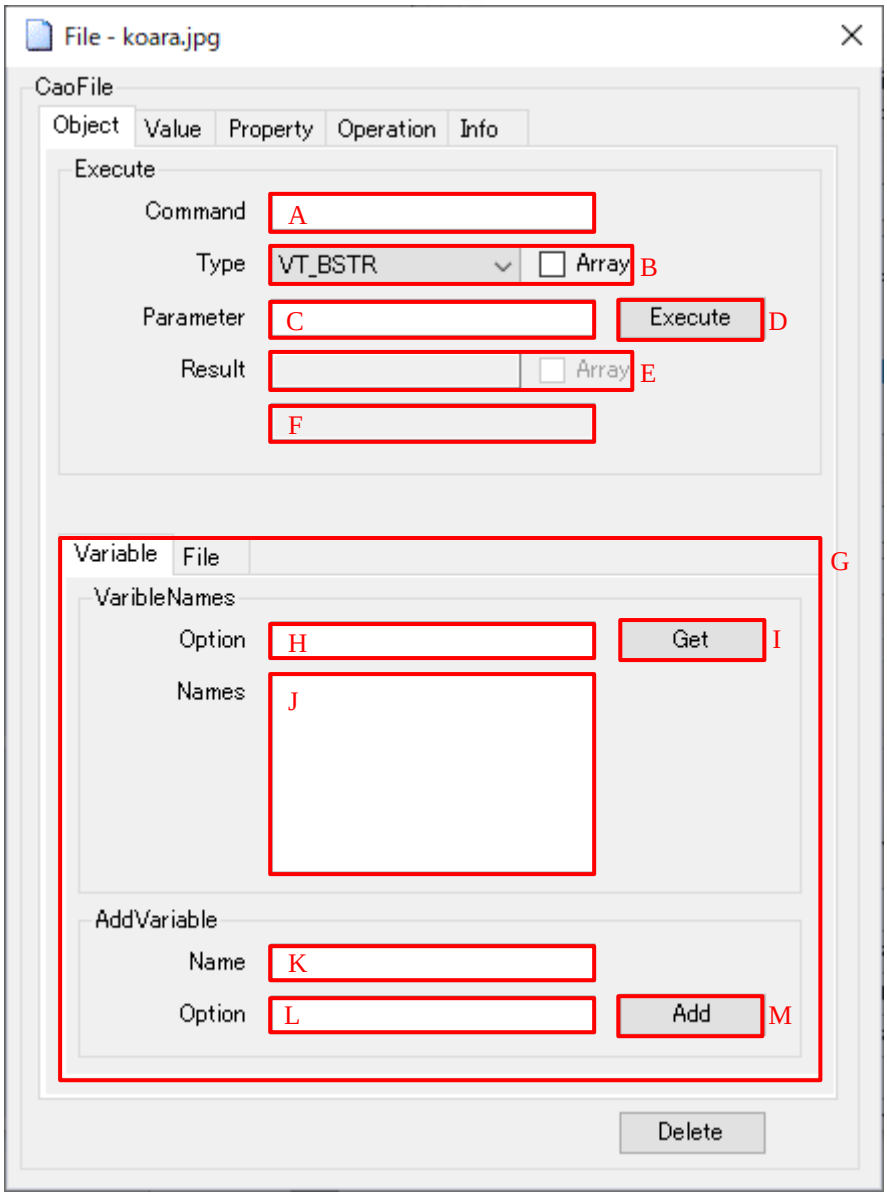


Diagram 2-15 File Window File

A: Command name

Used as the commandname when executing Execute of D.

B: Parameter type information

Used as the type of the parameter when Execute of D is executed.

C: Parameter string

Used as the parameter string when Execute of D is executed.

D: Running Execute

Performs a Execute using A, B, and C.

The result type and value at normal termination are stored in E and F.

E: Execute Run-Result Type Info

Stores the type of Execute executed by D.

F: Outcome of Execute

Stores Execute result of D.

G: Object creation group

The group in which to create the object.

The object creation group is a in which all variables and file objects are in the same layout.[5]

H: VariableNames optional [(1)]

Used as an optional option when executing VariableNames of I.

I: VariableNames run [(1)]

Stores VariableNames executionresults.

J: VariableNames run [(1)]

Stores the I VariableNames executionresults.

K: Object-name to be created[(1)]

It is used by the name of the variable when AddVariable of M is executed.

L: Optional of the object to be created[(1)]

Used as an optional option when executing M's AddVariable.

M: AddVariable run [(1)]

Execute AddVariable using K,L to create the object.

[5] It is described as an example of creating a variable object as a representative.

2.7.2. Value Tabs

Value tab allows you to set and retrieve file object values.

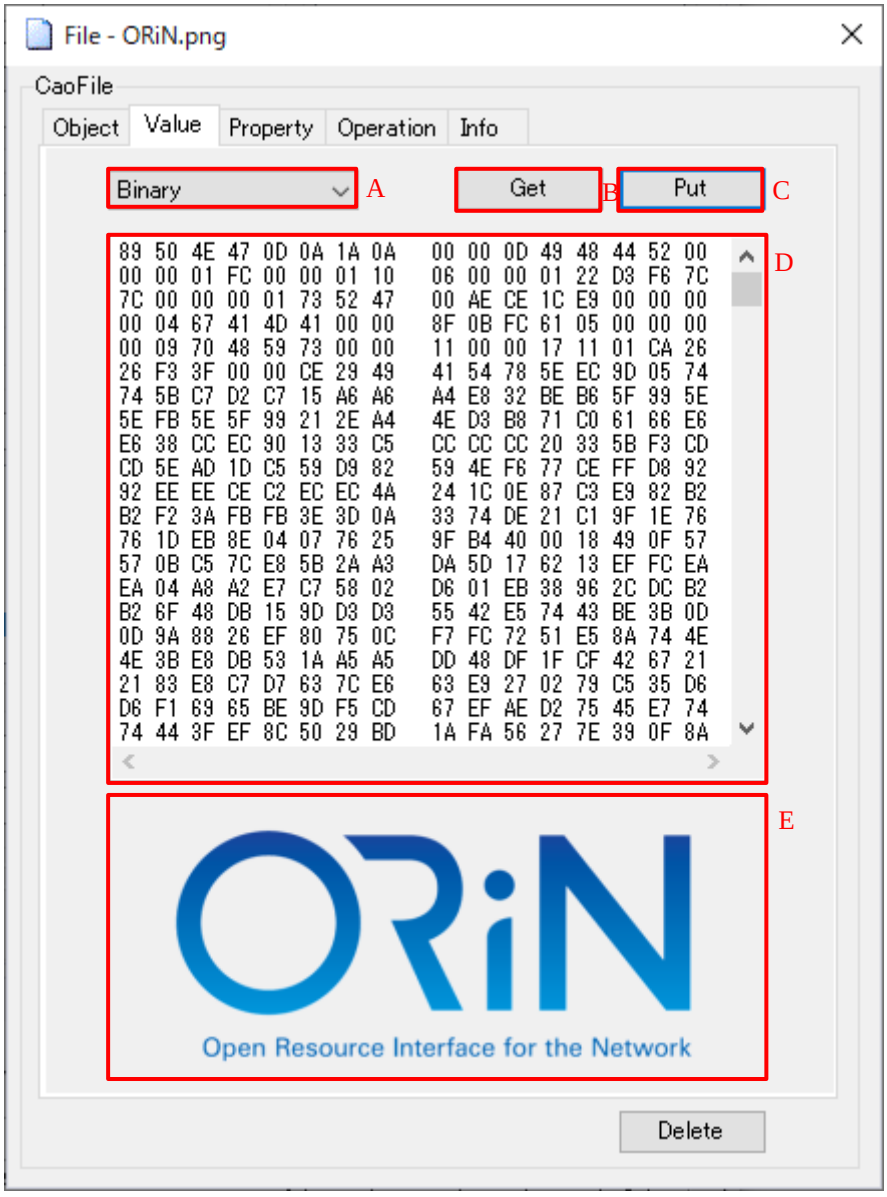


Diagram 2-16 File Window Value

A: Open mode

Stores the result of acquiring the contents of the B file.

Used as open mode for setting the contents of C files.

B: Retrieving the contents of a file

Execute get_Value and store the result in A and C.

For image data, a thumbnail is displayed on E.

Image window can be displayed by clicking the thumbnail images. Refer to 2.16 for details.

C: Setting the contents of the file

Execute put_Value using the contents of A and D.

D: Content of the file

Stores the result of executing the get_Value of B.

Used when executing C put_Value.

E: Results of getting help

When the result of executing get_Value of B is an image file, it is displayed as a thumbnail image.

Image window can be displayed by clicking the thumbnail images. Refer to 2.16 for details.

2.7.3. Property Tabs

Property tab allows you to set and retrieve file object properties.

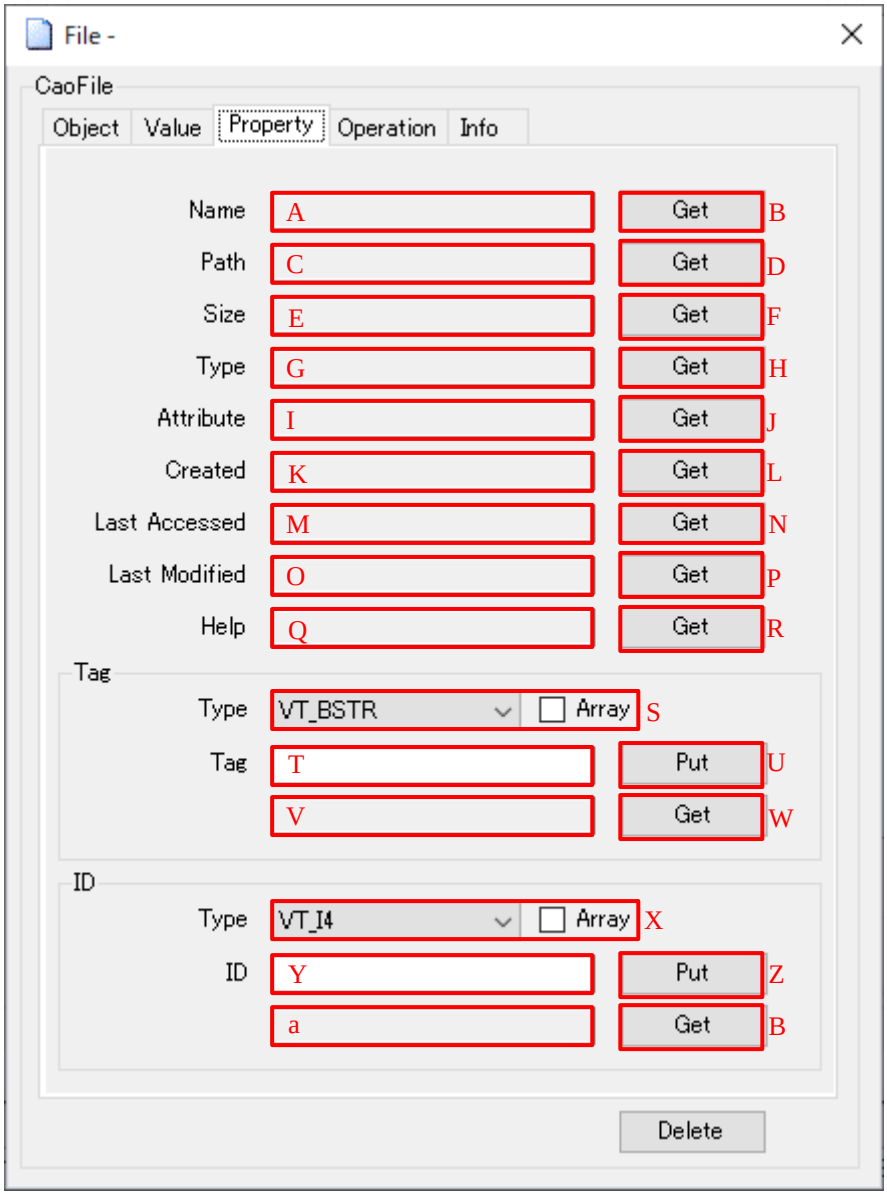


Diagram 2-17 File Window Property

A: Results of Acquiring a File Name

Stores the result of acquiring the file name of B.

B: Retrieval of file name

Acquires the file name and stores the execution result in A.

C: Results of Retrieving the Path of a File

Stores the result of acquiring the path of D file.

D: Retrieving the Path of a File

Gets the path of the file and stores the execution result in C.

E: Results of Acquiring the File Size

Stores the result of acquiring the file size of F.

F: Results of Acquiring the File Size

Stores the result of acquiring the E file size.

G: Results of Retrieving File Types

Stores the result of acquiring the file type of H.

H: Retrieving File Types

Gets the file type and stores the execution result in G.

I: Results of getting attributes

Stores the result of acquiring the attribute of J.

J: Retrieving Attributes

Retrieves the attribute and stores the execution result in I.

K: Acquisition result of creation date and time

Stores the result of acquiring the creation date and time of L.

L: Obtaining Creation Date and Time

Acquires the creation date and time and stores the execution result in K.

M: Acquisition result of the last access date and time

Stores the result of acquiring the last access date and time of N.

N: To obtain the last access date and time

Acquires the last access date and time, and stores the execution result in M.

O: Acquisition result of the last modification date and time

Stores the result of acquiring the last modification date and time of P.

P: Obtaining the Last Modified Date and Time

Acquires the last modification date and time, and stores the execution result in O.

Q: Results of getting help

Stores the result of getting help for R.

R: Getting Help

Get help and store the execution result in Q.

S: Tag type information

Used as parameter type information when executing Up put_Tag.

Stored as type information of get_Tag execution result of W.

T: Tag data string

Used as tag data at the time of execution of Up put_Tag.

U: Tag Settings

Run put_Tag.

V: Results of tag acquisition

Stores the execution result of get_Tag of W.

W: Retrieving Tags

Execute get_Tag and the execution result is stored in V.

Type information is stored in S.

X: ID type information

Used as type information for Z's put_ID runtime parameter.

Stored as type information of get_ID execution result of b.

Y: ID string

Used as tag data when executing put_ID of Z.

Z: ID setting

Run put_ID.

a: Results of obtaining the ID

Stores the execution result of get_ID of b.

b: Get ID

Execute get_ID and the execution result is stored in a.

Type information is stored in X.

2.7.4. Operation Tabs

Operation tab allows you to move, copy/delete files, and create tasks.

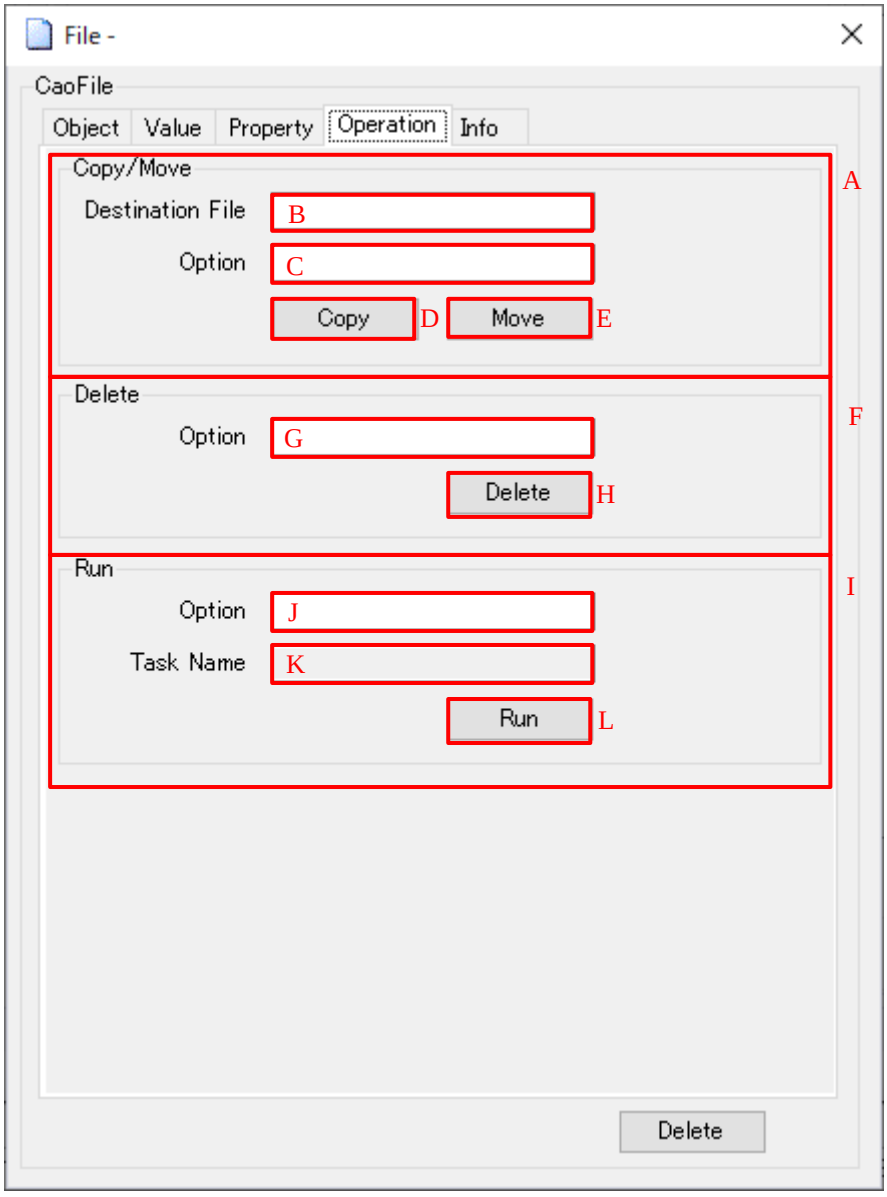


Diagram 2-18 File Window Operation

A: Copy and move groups of files

A group for copying and moving files.

B: Destination

Used as the first parameter when executing Copy,Move of D and E.

C: Destination option

Used as the second parameter when executing Copy,Move of D and E.

D: Copying Files

Execute Copy using A,B.

E: Moving Files

Execute Move using A,B.

F: Delete file group

A group for deleting files.

G: Delete File Options

Used as the first parameter when Delete of H is executed.

H: Deleting Files

Run Delete using G.

I: Task creation group

The group in which to create the task.

J: Task Generation Options

Used as the first argument when running L.

K: Results of task retrieval

Stores the execution result of L Run.

L: Create Task

Run using the value of J and store the result in K.

2.7.5. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.8. Task window

In Task window, you can create variable objects, work with tasks, and execute extended commands.

2.8.1. Object Tabs

Object tab allows you to create variable objects and execute extended commands.

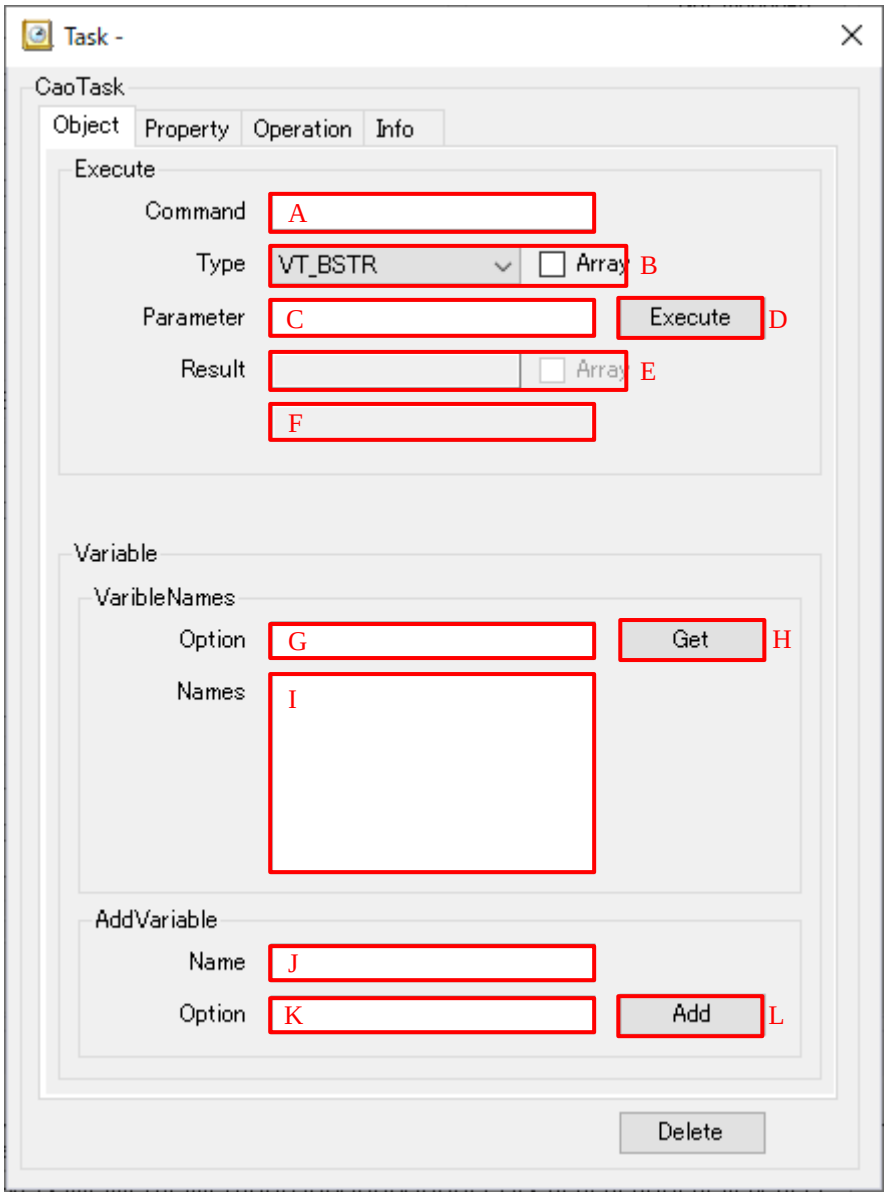


Diagram 2-19 Task Window Object

A: Command name

Used as the commandname when executing Execute of D.

B: Parameter type information

Used as the type of the parameter when Execute of D is executed.

C: Parameter string

Used as the parameter string when Execute of D is executed.

D: Running Execute

Performs a Execute using A, B, and C.

The result type and value at normal termination are stored in E and F.

E: Type Info of Execute Run Result

Stores the type of Execute executed by D.

F: Resulting string from Execute run

Stores Execute result of D.

G: VariableNames Optional

Used as an optional VariableNames runtime for H.

H: Running VariableNames

Run VariableNames.

The execution result is stored in I.

I: Outcome of VariableNames

Stores the H VariableNames executionresult.

J: Object name to create

It is used by the variable-name at the time of executing AddVariable of L.

K: Options for the object being created

Used as an optional AddVariable runtime for L.

L: Running AddVariable

Execute AddVariable using the values of J,K to create the object.

2.8.2. Property Tabs

On Property tab, you can set and retrieve the properties of a task object.

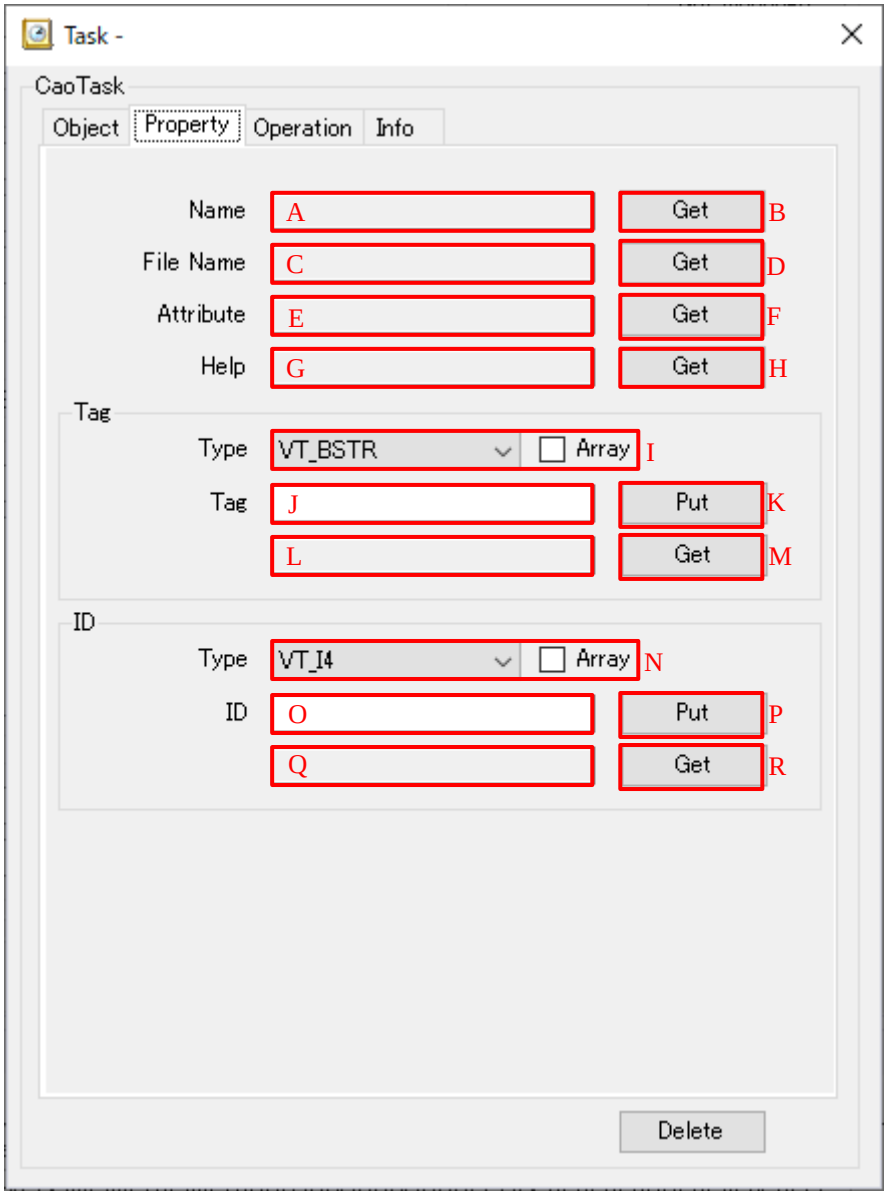


Diagram 2-20 Task Window Property

A: Results of Retrieving Task Names

Stores the result of acquiring the task name of B.

B: Get Task Name

Gets the task name and stores the execution result in A.

C: Result of acquiring the corresponding file name

Stores the result of acquiring the corresponding file name of D.

D: Obtaining Supported File Names

Acquires the corresponding file name and stores the execution result in C.

E: Results of getting attributes

Stores the result of acquiring the attribute of F.

F: Retrieving Attributes

Retrieves the attribute and stores the execution result in E.

G: Results of getting help

Stores the result of getting help for H.

H: Getting Help

Get help and store the execution result in G.

I: Tag type information

Used as parameter type information at the time of K put_Tag execution.

Stored as type information of M get_Tag execution result.

J: Tag data string

Used as tag data when K's put_Tag is executed.

K: Tag Settings

Run put_Tag.

L: Results of tag acquisition

Stores the result string of the execution result of M get_Tag.

M: Retrieving Tags

The result string of the execution result is stored in L by executing get_Tag.

Type information is stored in I.

N: ID type information

Used as parameter type information when executing put_ID of P.

Stored as type information of the result of executing R get_ID.

O: ID string

Used as tag data when executing put_ID of P.

P: ID setting

Run put_ID.

Q: Results of obtaining the ID

Stores the result of executing R's get_ID.

R: Get ID

Execute get_ID and the execution result is stored in Q.

Type information is stored in N.

2.8.3. Operation Tabs

Operation tab allows you to start, stop, and delete tasks.

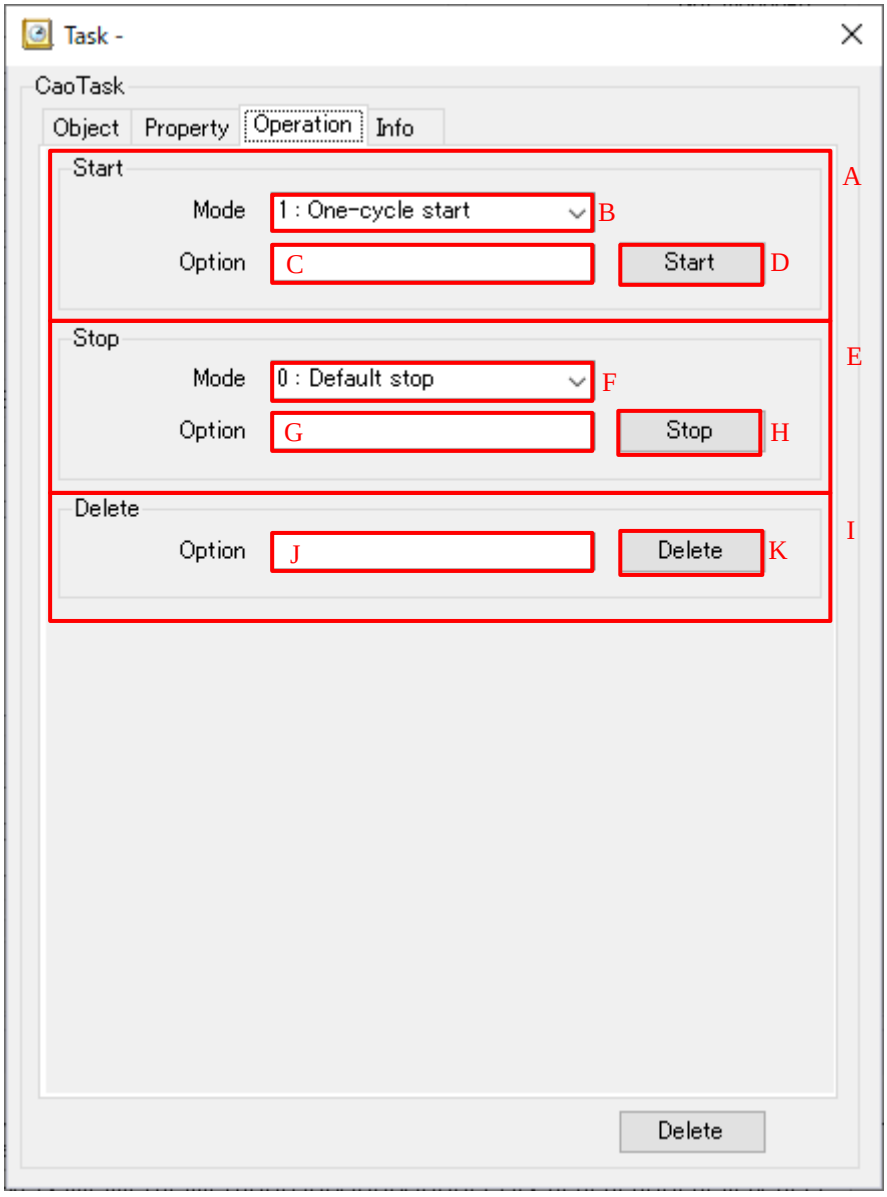


Diagram 2-21 Task Window Operation

A: Started group of tasks

The group in which to start the task.

B: Task start mode

Used as the first parameter when executing Start of D.

C: Start Task Options

Used as the second parameter when Start of D is executed.

D: Starting a Task

Execute Start using B,C.

E: Stop Task Group

A group for stopping tasks.

F: Stop Task Mode

Used as the first parameter when Stop of H is executed.

G: Stop Task Options

Used as the second parameter when Stop of H is executed.

H: Stopping a Task

Execute Stop using F,G.

I: Delete Task Group

The group from which the task is to be deleted.

J: Delete Task Options

It is used as the first parameter when Delete of K is executed.

K: Stopping a Task

Run Delete using J.

2.8.4. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.9. Extension window

In Extension window, you can create variable objects and execute extended commands.

2.9.1. Object Tabs

Object tab allows you to create variable objects and execute extended commands.

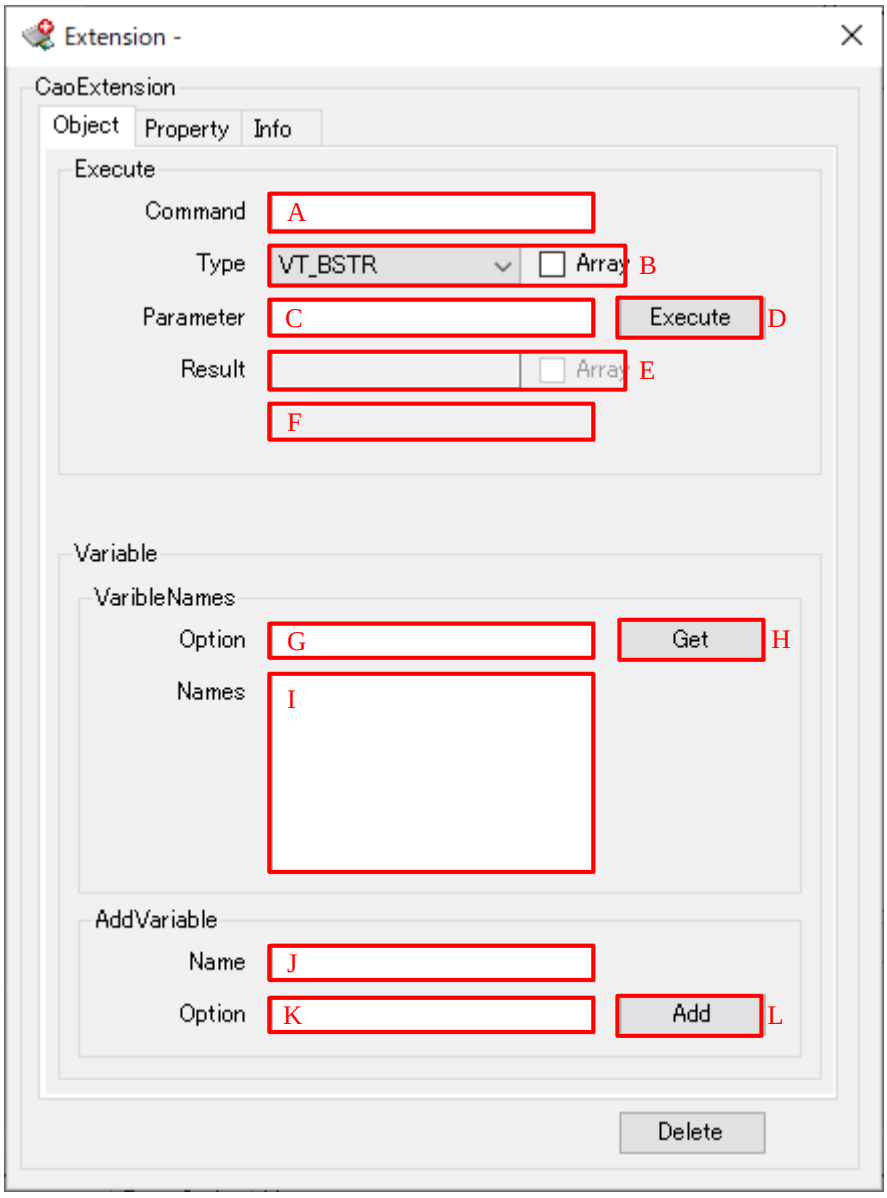


Diagram 2-22 Extension Window Object

A: Command name

Used as the commandname when executing Execute of D.

B: Parameter type information

Used as the type of the parameter when Execute of D is executed.

C: Parameter string

Used as the parameter string when Execute of D is executed.

D: Running Execute

Performs a Execute using A, B, and C.

The result type and value at normal termination are stored in E and F.

E: Type Info of Execute Run Result

Stores the type of Execute executed by D.

F: Outcome of Execute

Stores Execute result of D.

G: VariableNames Optional

Used as an optional VariableNames runtime for H.

H: Running VariableNames

Run VariableNames.

The execution result is stored in I.

I: Outcome of VariableNames

Stores the H VariableNames executionresult.

J: Object name to create

It is used by the variable-name at the time of executing AddVariable of L.

K: Options for the object being created

Used as an optional AddVariable runtime for L.

L: Running AddVariable

Execute AddVariable using the values of J,K to create the object.

2.9.2. Property Tabs

On Property tab, you can set and retrieve the properties of a task object.

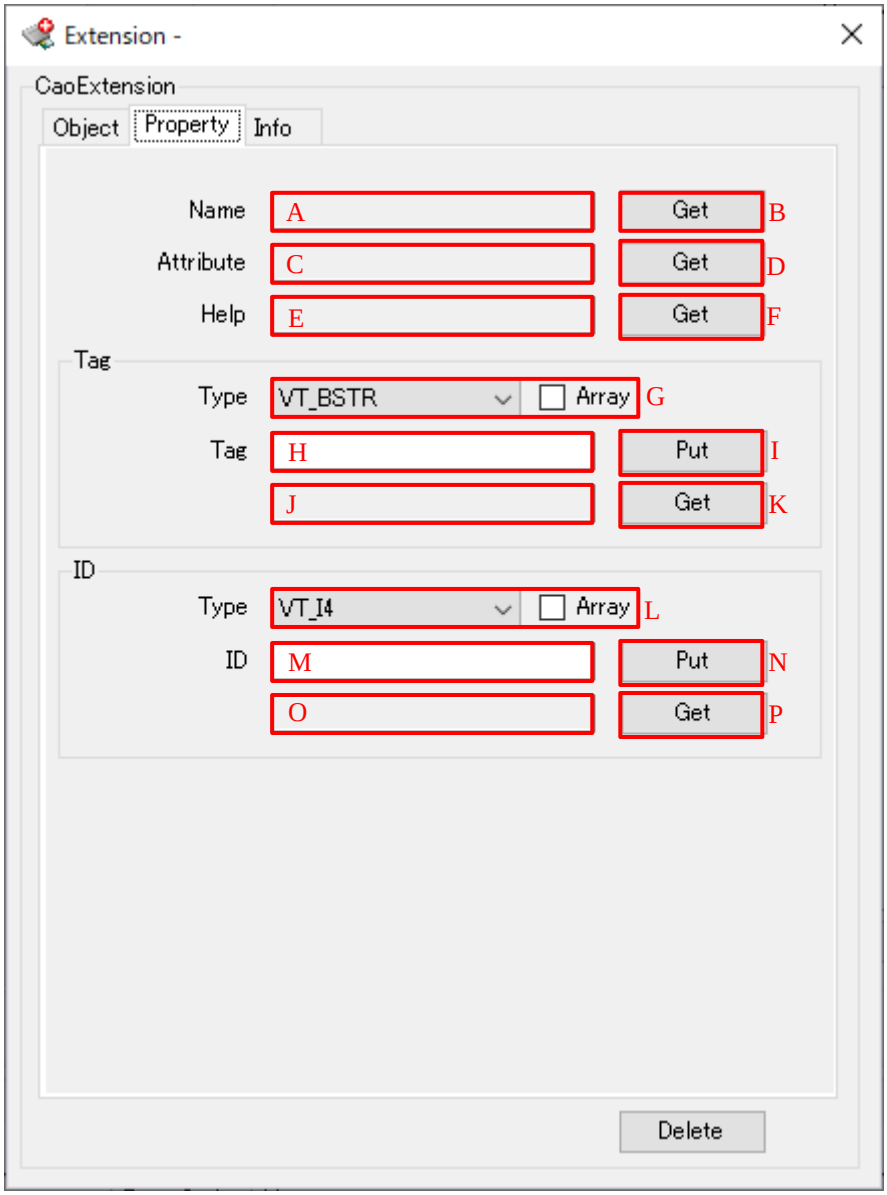


Diagram 2-23 Extension Window Property

A: Result of acquiring the extended board name

Stores the result of acquiring the extended board name of B.

B: Get extended board name

Acquires the extension board name and stores the execution result in A.

C: Results of getting attributes

Stores the result of acquiring the attribute of D.

D: Retrieving Attributes

Retrieves the attribute and stores the execution result in C.

E: Results of getting help

Stores the result of getting help for F.

F: Getting Help

Get help and store the execution result in E.

G: Tag type information

Used as parameter type information when I put_Tag is executed.

Stored as K get_Tag execution result type information.

H: Tag data string

Used as tag data when I's put_Tag is executed.

I: Tag Settings

Run put_Tag.

J: Results of tag acquisition

Stores the execution result of get_Tag of K.

K: Retrieving Tags

Execute get_Tag and the execution result is stored in J.

Type information is stored in I.

L: ID type information

N's put_ID is used as parameter type information at runtime.

Stored as type information of the get_ID execution result of P.

M: ID string

Used as tag data when executing N put_ID.

N: ID setting

Run put_ID.

O: Results of obtaining the ID

Stores the execution result of the get_ID of P.

P: Get ID

Execute get_ID and the execution result is stored in O.

Type information is stored in L.

2.9.3. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.10. Command window

You can execute extended commands in Command window.

2.10.1. Object Tabs

Object tab allows you to execute commands.

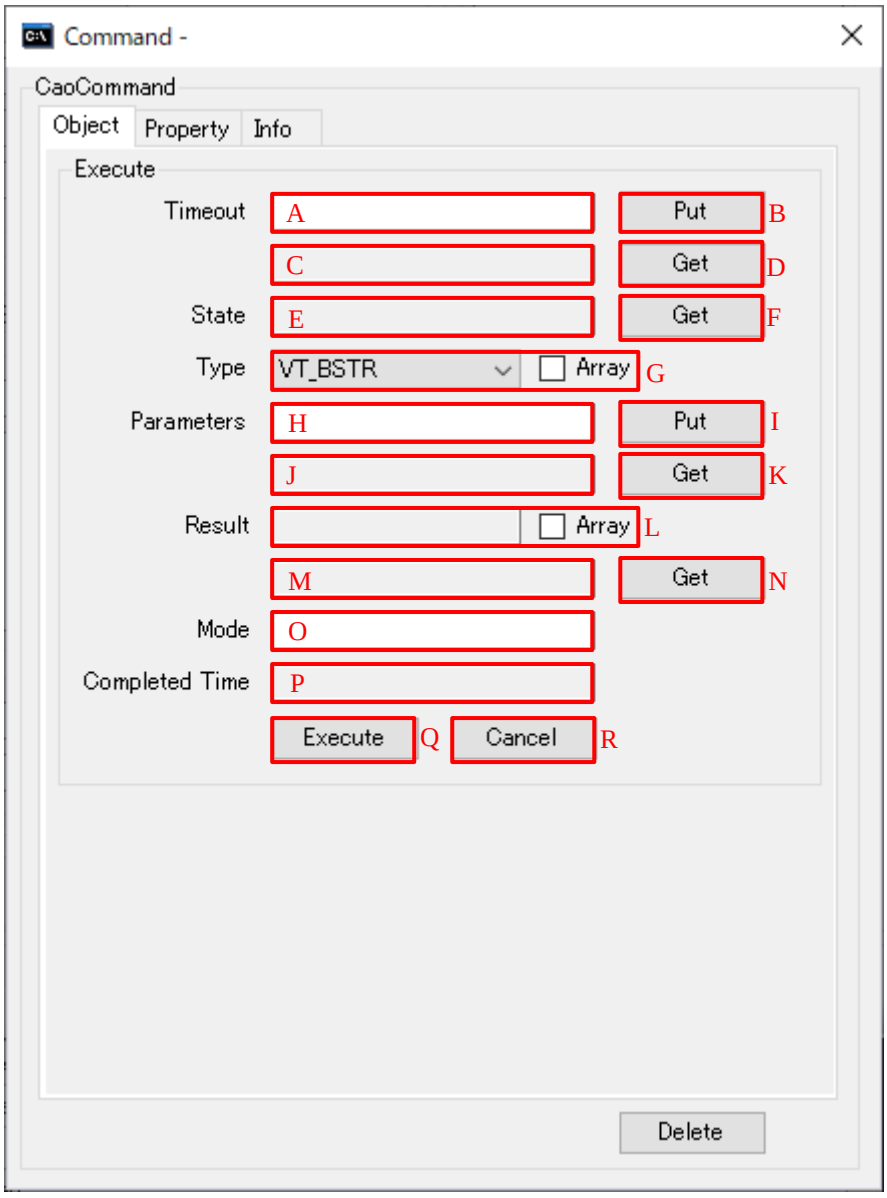


Diagram 2-24 Command Window Object

A: Results of getting a timeout

Stores the result of acquiring the timeout of B.

B: Get Timeout

Gets the timeout and stores the execution result in A.

The settings must be made prior to executing the Q Execute.

C: Results of Setting the Timeout

Stores the execution result of D timeout setting.

D: Setting timeouts

Set the timeout and store the execution result in C.

E: Results of status acquisition

Stores the result of acquiring the status of F.

F: Get Status

Acquires the status and stores the execution result in E.

G: Command parameter type information

Used as the type of the parameter at the time of put_Parameters of I.

Stored as the type of the get_Parameters of K.

H: Command parameter data string

Used as tag data when I put_Parameters is executed.

I: Setting Command Parameters

Run the put_Parameters.

The settings must be made prior to executing the Q Execute.

J: Acquisition result of command parameter

The get_Parameters of K is executed and stored.

K: Retrieving Command Parameters

Executes the get_Parameter and stores the result in J.

Type information is stored in I.

L: Type Info of Result Run Result

This is stored as the type of Result of N.

M: Outcome of Result

Stores N Result executions.

N: Running Result

Run Result.

The result type and value are stored in L,M.

O: Execution mode

Used as the first parameter when Execute of Q is executed.

P: Execute run completion date and time

Stores Execute completion date and time of N.[6]

When the mode is synchronous execution (0), Execute execution completion date and time are stored.

When the mode is asynchronous execution (1), the reception date and time of the command execution completion event is stored.

Q: Running Execute

Run Execute on the.[6]

When the mode is synchronous execution (0), Execute execution completion date and time are stored.

When the mode is asynchronous execution (1), the reception date and time of the command execution completion event is stored.

R: Canceling a Execute

Cancels a running Execute.

[6] Currently, the date and time of completion of Execute is not recorded in asynchronous execution because the command execution completion event is not supported.

2.10.2. Property Tabs

On Property tab, you can set and retrieve the properties of a task object.

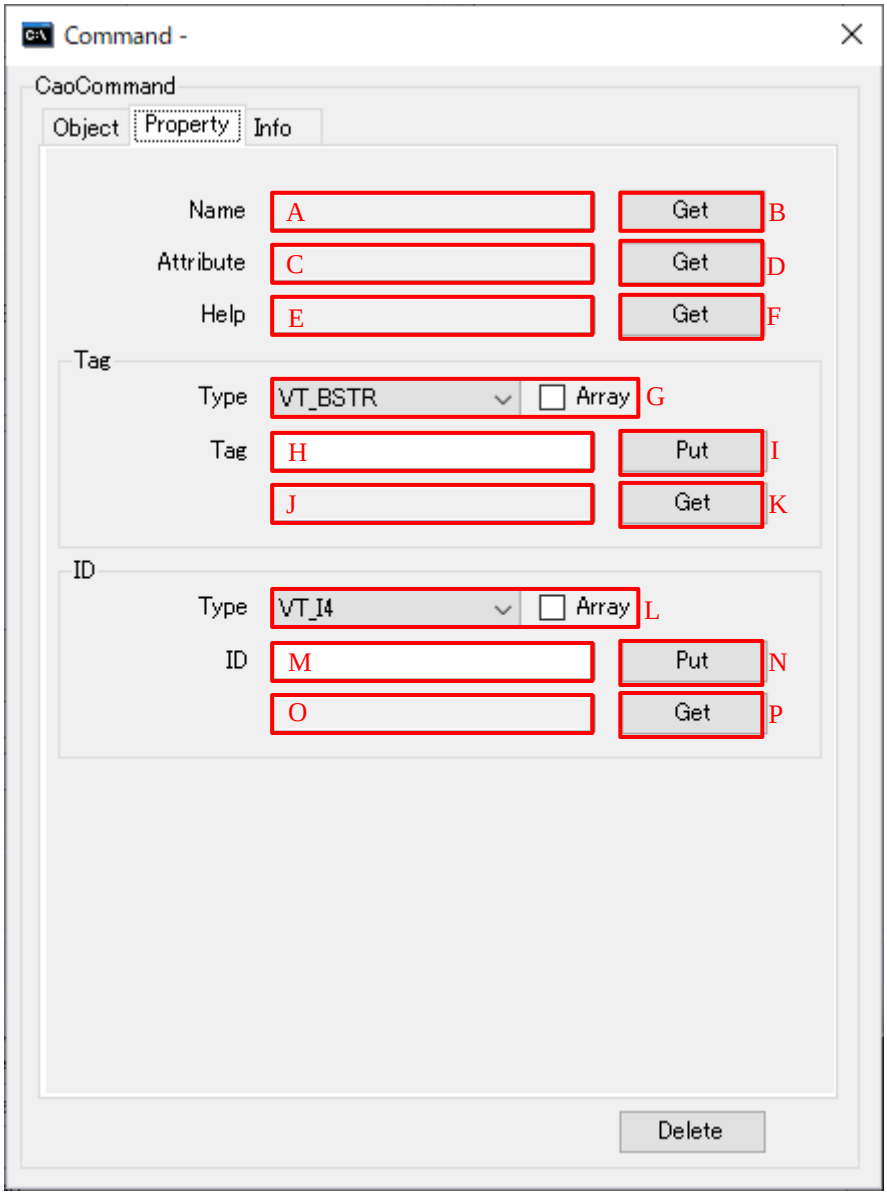


Diagram 2-25 Command Window Property

A: Execution result of command name acquisition

Stores the result of acquiring the command name of B.

B: Get command name

Gets the command name and stores the execution result in A.

C: Results of getting attributes

Stores the result of acquiring the attribute of D.

D: Retrieving Attributes

Retrieves the attribute and stores the execution result in C.

E: Results of getting help

Stores the result of getting help for F.

F: Getting Help

Get help and store the execution result in E.

G: Tag type information

Used as parameter type information when I put_Tag is executed.

Stored as K get_Tag execution result type information.

H: Tag data string

Used as tag data when I's put_Tag is executed.

I: Tag Settings

Run put_Tag.

J: Results of tag acquisition

Stores the execution result of get_Tag of K.

K: Retrieving Tags

Execute get_Tag and the execution result is stored in J.

Type information is stored in I.

L: ID type information

N's put_ID is used as parameter type information at runtime.

Stored as type information of the get_ID execution result of P.

M: ID string

Used as tag data when executing N put_ID.

N: ID setting

Run put_ID.

O: Results of obtaining the ID

Stores the execution result of the get_ID of P.

P: Get ID

Execute get_ID and the execution result is stored in O.

Type information is stored in L.

2.10.3. Info Tabs

On Info tab, you can see what you know when you create the object. Refer to 2.17 for details.

2.11. Message window

In Message window, you can check, reply to, and clear messages.

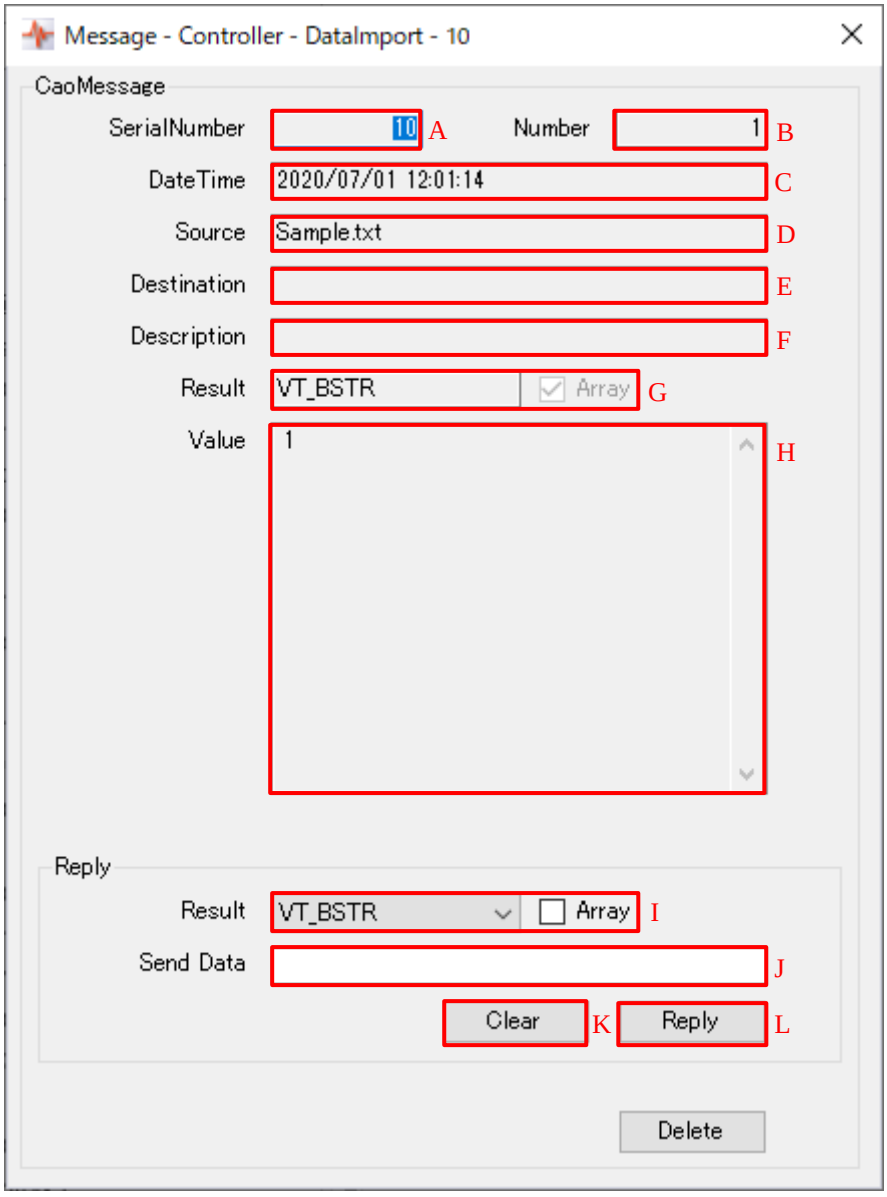


Diagram 2-26 Message Window

A: Message sequence number

The sequential number automatically added by the engine is stored.

B: Message number

Stores the message number.

C: Creation date and time

Contains the creation date and time.

D: Source

The source is stored.

E: Destination

The destination is stored.

F: Description

The description is stored.

G: Message body type information

Stores the type information of the message body of H.

H: Message body data string

Stores the data string of the message body.

I: Reply type information for the message

Contains the type information of the message reply.

J: Reply data string of the message

Stores the reply data string of the message.

K: Clearing Messages

Performs a message clear.

L: Replying to Messages

Returns a message using the values of I and J.

2.12. Engine Status window

In Engine Status window, you can check the number of objects at regular intervals.

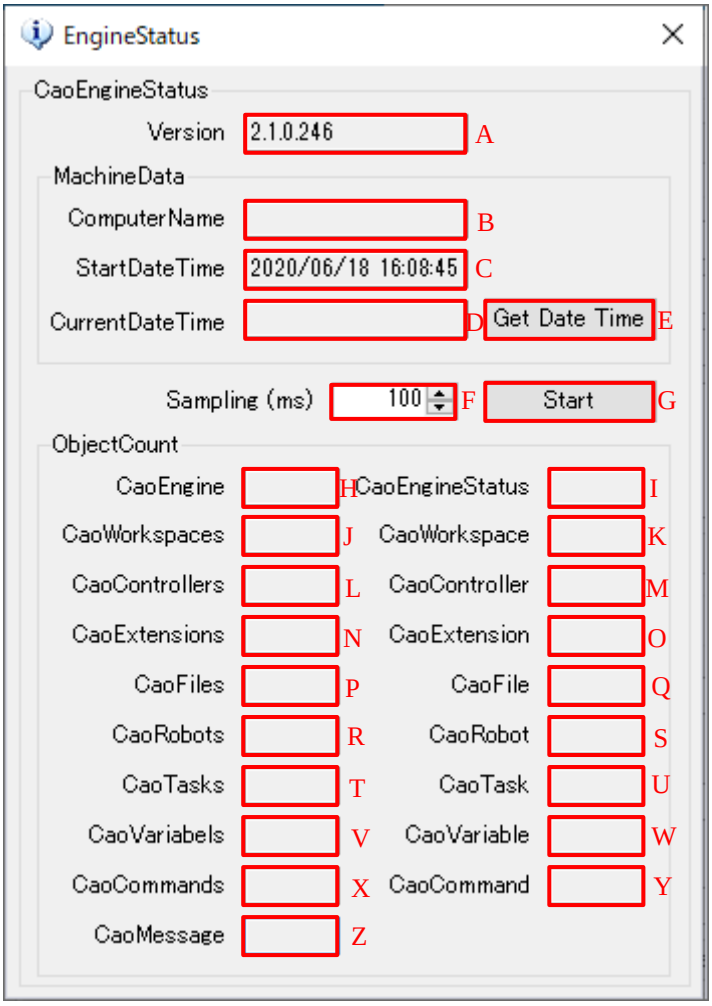


Diagram 2-27 EngineStatus Window

A: Engine version

Contains the engine version.

B: Computer name

Contains the computer name.

C: Start date and time of the engine

Contains the start date and time of the engine.

D: Current date and time

Stores the result of acquiring the current date and time of E.

E: Get current date and time

Gets the current date and time and stores it in D.

F: Sampling interval for object count acquisition

Used as the sampling interval for starting the object count acquisition of G.

G: Start/end of object count acquisition

Starts/stops acquiring the object count at the sampling interval of F.

H: CaoEngine Object-Counting

Gets and stores CaoEngine objectcount at the sampling interval.

I: CaoEngineStatus Object-Counting

Gets and stores CaoEngineStatus objectcount at the sampling interval.

J: CaoWorkspaces Object-Counting

Gets and stores CaoWorkspaces objectcount at the sampling interval.

K: CaoWorkspace Object-Counting

Gets and stores CaoWorkspace objectcount at the sampling interval.

L: CaoControllers Object-Counting

Gets and stores CaoControllers objectcount at the sampling interval.

M: CaoController Object-Counting

Gets and stores CaoController objectcount at the sampling interval.

N: CaoExtensions Object-Counting

Gets and stores CaoExtensions objectcount at the sampling interval.

O: CaoExtension Object-Counting

Gets and stores CaoExtension objectcount at the sampling interval.

P: CaoFiles Object-Counting

Gets and stores CaoFiles objectcount at the sampling interval.

Q: CaoFile Object-Counting

Gets and stores CaoFile objectcount at the sampling interval.

R: CaoRobots Object-Counting

Gets and stores CaoRobots objectcount at the sampling interval.

S: CaoRobot Object-Counting

Gets and stores CaoRobot objectcount at the sampling interval.

T: CaoTasks Object-Counting

Gets and stores CaoTasks objectcount at the sampling interval.

U: CaoTask Object-Counting

Gets and stores CaoTask objectcount at the sampling interval.

V: CaoVariables Object-Counting

Gets and stores CaoVariables objectcount at the sampling interval.

W: CaoVariable Object-Counting

Gets and stores CaoVariable objectcount at the sampling interval.

X: CaoCommands Object-Counting

Gets and stores CaoCommands objectcount at the sampling interval.

Y: CaoCommand Object-Counting

Gets and stores CaoCommand objectcount at the sampling interval.

Z: CaoMessage Object-Counting

Gets and stores CaoMessage objectcount at the sampling interval.

2.13. Engine window

You can create workspace objects in Engine window.

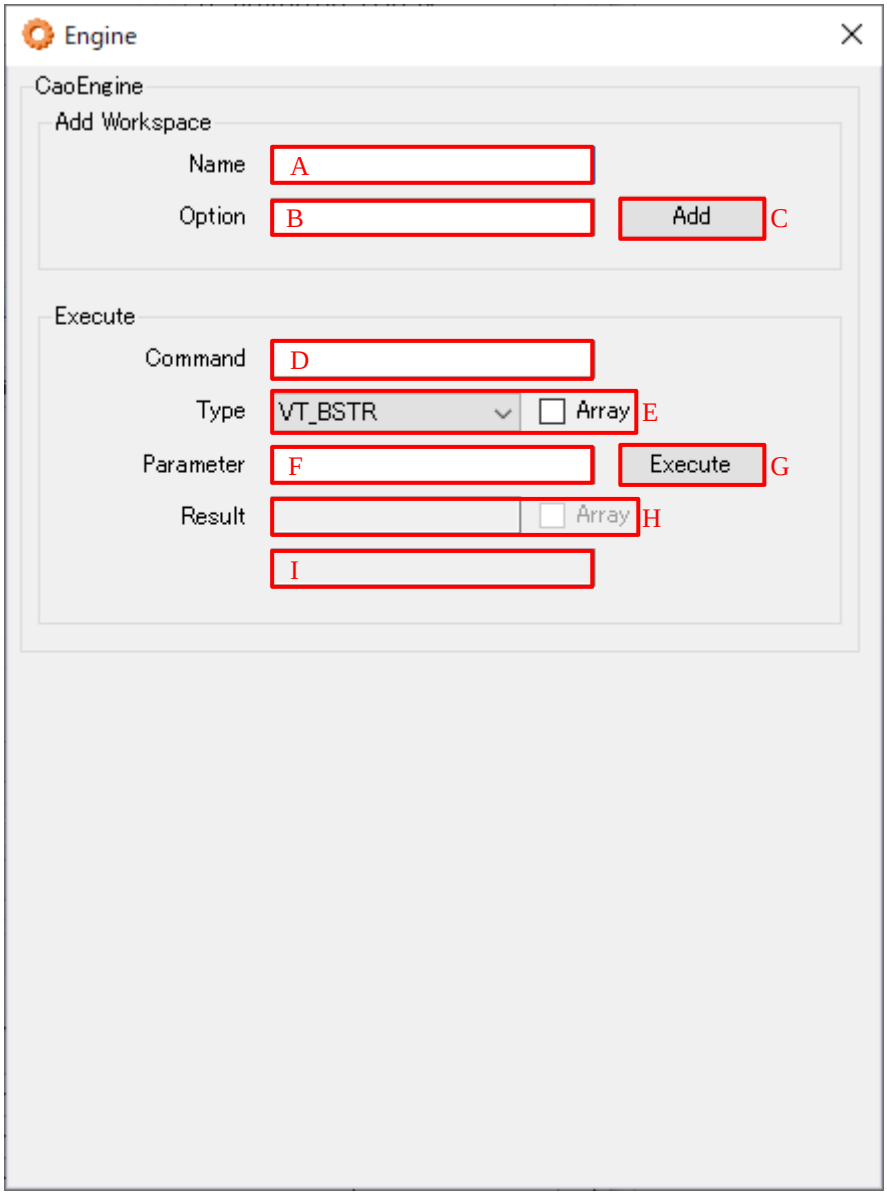


Diagram 2-28 Engine Window

A: Workspace object you want to create

Used as a variable-name when executing a C AddWorkspace.

B: Options for the workspace object that you want to create

Used as an optional C AddWorkspace runtime.

C: Running AddWorkspace

Create a workspace object by running AddWorkspace with the values A,B.

D: Command name

Used as the commandname when executing G's Execute.

E: Parameter type information

Used as the parameter type when executing G's Execute.

F: Parameter string

Used as a parameter string when executing G's Execute.

G: Running Execute

Execute is executed using D, E, and F.

The result type and value at normal termination are stored in H and I.

H: Type Info of Execute Run Result

The type of Execute of G is stored.

I: Outcome of Execute

Stores the G Execute executionresults.

2.14. Object window

The objects that you created or deleted are displayed in a tree.

You can activate the object window by double-clicking the object or clicking [Right-click]-[Show].

Click [Right-click]-[Delete] to delete the object and object window.

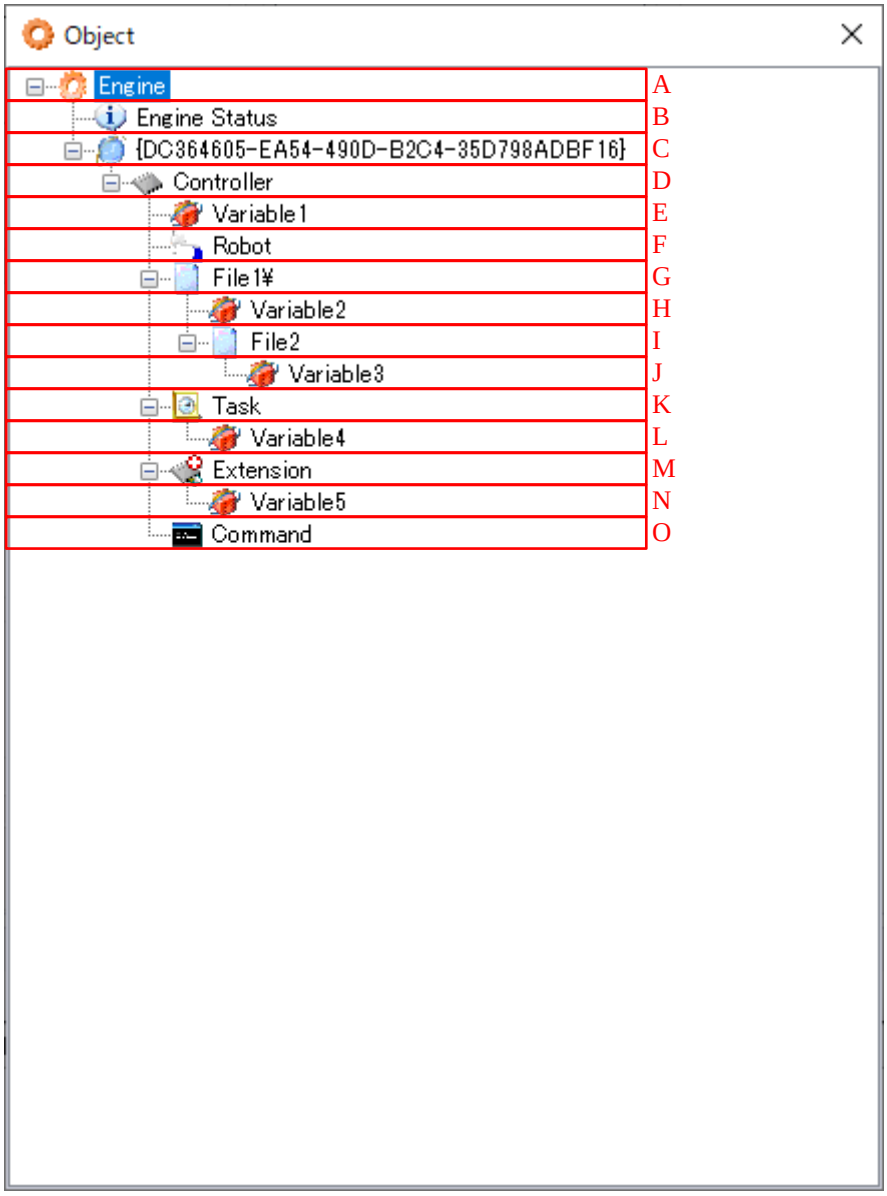


Figure 2-29 Object Window

A: CaoEngine Objects

CaoEngine object.

B: CaoEngineStatus Objects

CaoEngineStatus object.

C: CaoWorkspace Objects

CaoWorkspace object.

D: CaoController Objects

CaoController object.

E: CaoVariable Objects

CaoVariable created from CaoController.

F: CaoRobot Objects

CaoRobot created from CaoController.

G: CaoFile Objects

CaoFile created from CaoController.

H: CaoVariable Objects

CaoVariable created from CaoFile.

I: CaoFile Objects

CaoFile generated from CaoFile.

J: CaoVariable Objects

CaoVariable created from CaoFile.

K: CaoTask Objects

CaoTask created from CaoController.

L: CaoVariable Objects

CaoVariable created from CaoTask.

M: CaoExtension Objects

CaoExtension created from CaoController.

N: CaoVariable Objects

CaoVariable created from CaoExtension.

O: CaoCommand Objects

CaoCommand created from CaoController.

2.15. Log Window

The operation you performed and the results are logged.

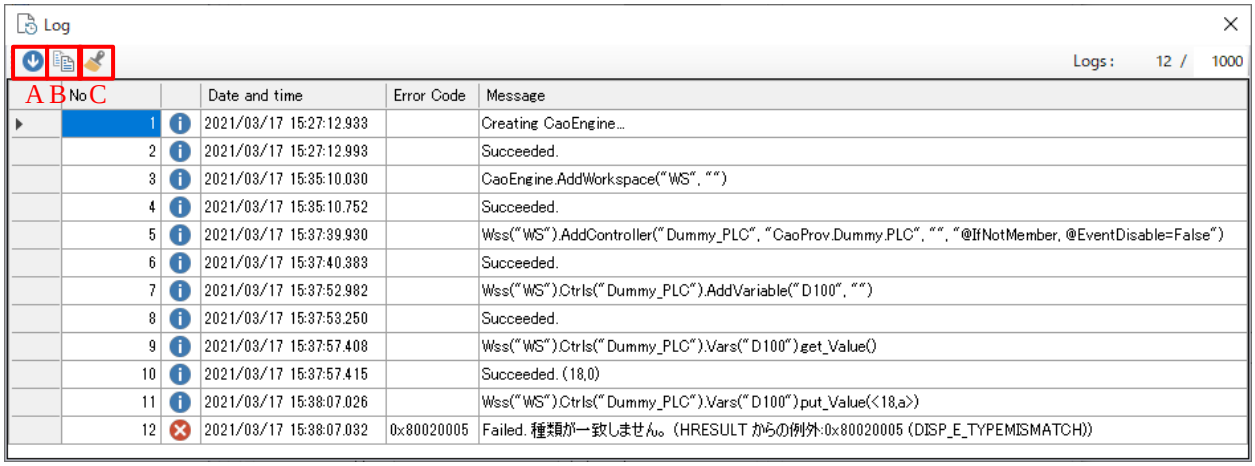


Figure 2-30. Log Window

A: Automatic Scrolling

Enables/disables automatic scrolling when adding logs.

B: Copy

Copies the selected item to the clipboard.

C: Clear

Clears all the displayed logs.

2.16. Image window

In Image window, you can check the image displayed as a thumbnail at the same size.

The thumbnailed image is displayed in the result of get_Value in Value tab of File window and the result of get_Value in Vaiarble window (Fig. 2-31, Fig. 2-32).

Image window can be displayed by clicking the thumbnailed images (Fig 2-33).

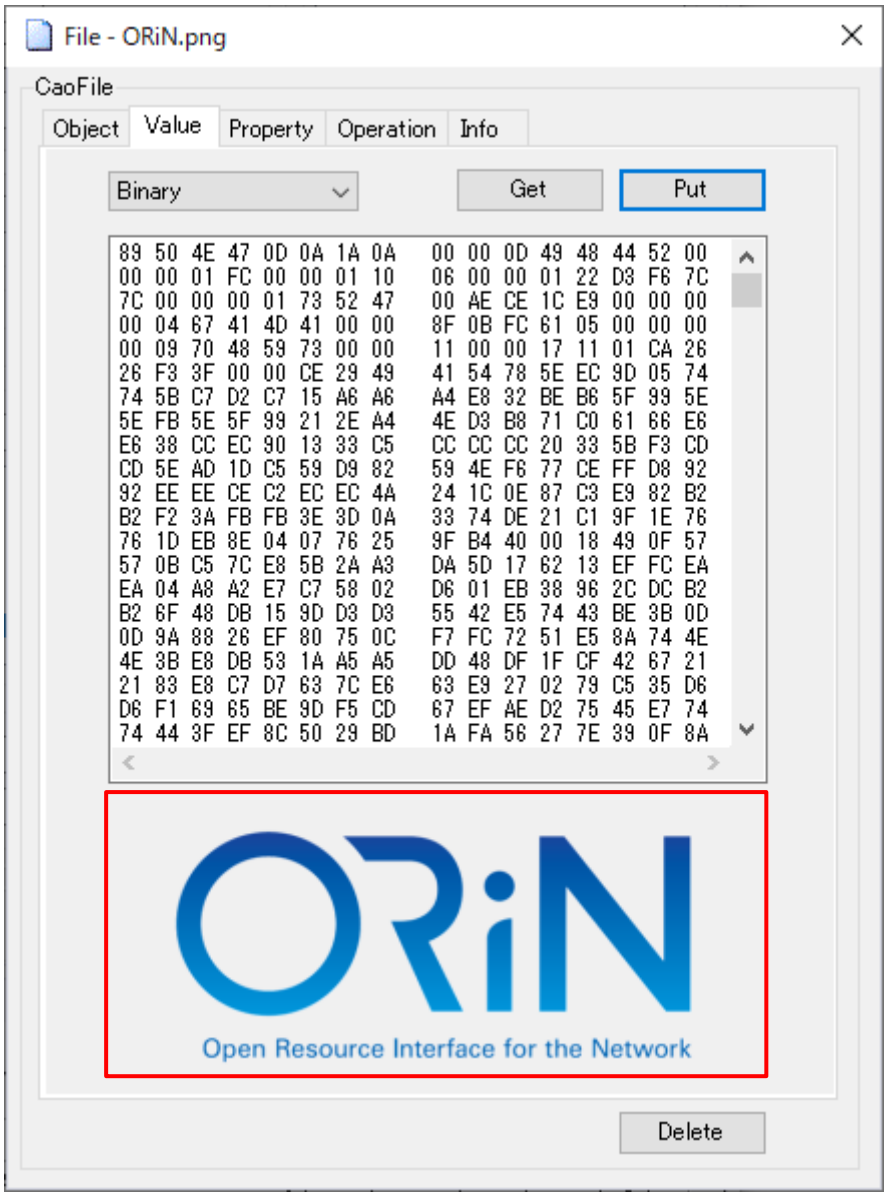


Fig. 2-31 Value properties of File window

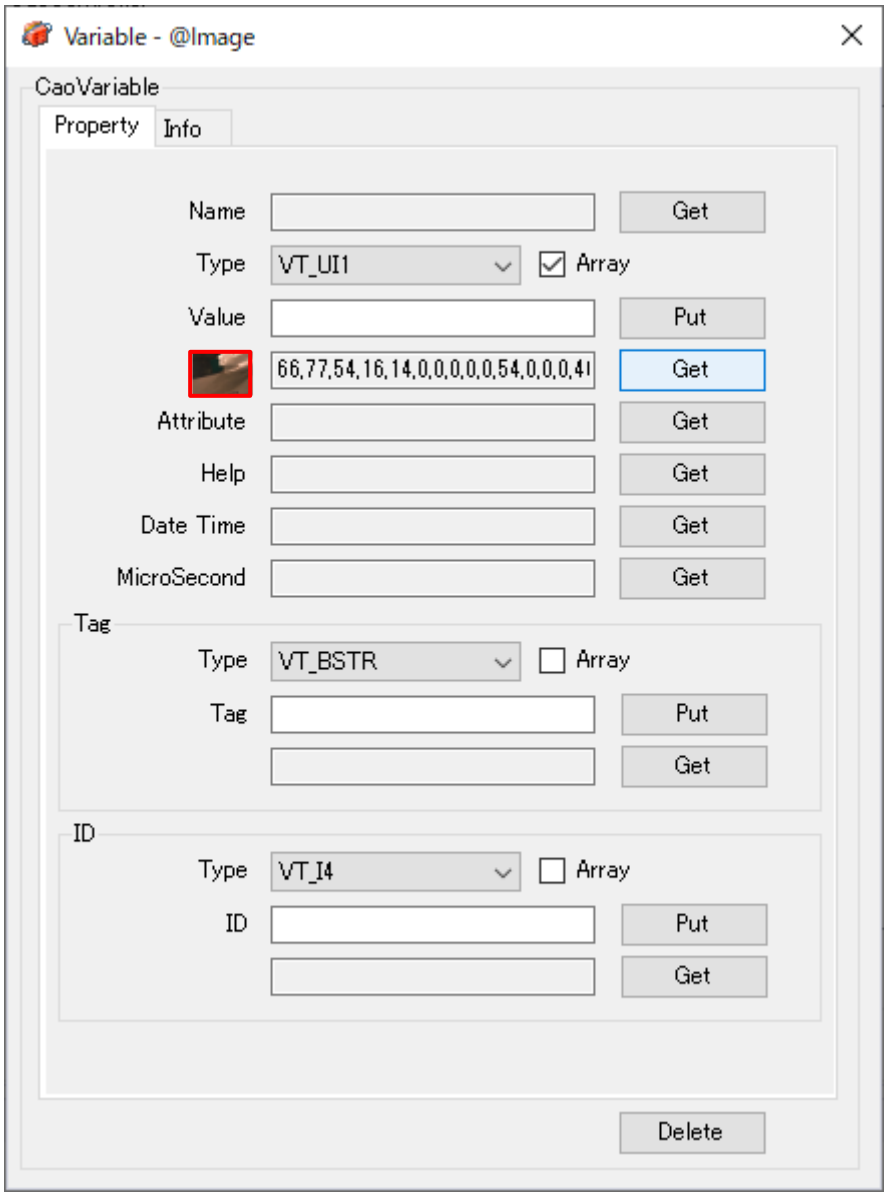


Diagram 2-32 Variable Window Property

Diagram 2-33 Image Window

2.17. Info Tabs

On Info tab, you can see what you are doing when you create the object.

You can see in Figure 2−34 that it was created by specifying "Variable" for the variable name and "Test=1" for the option.

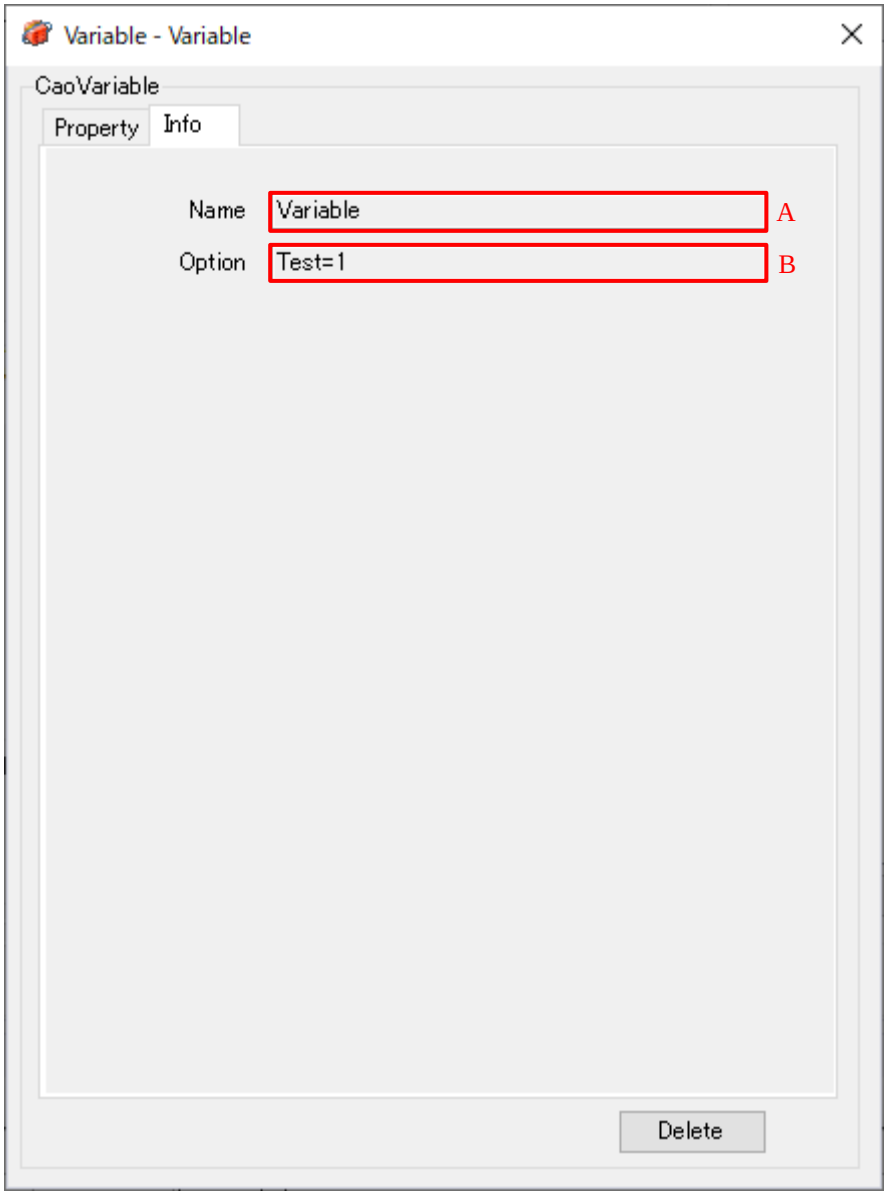


Fig.2-34 Info tabs

A: Object name

Contains the object name of the created object.

B: Option Name

Contains the option name of the created object.

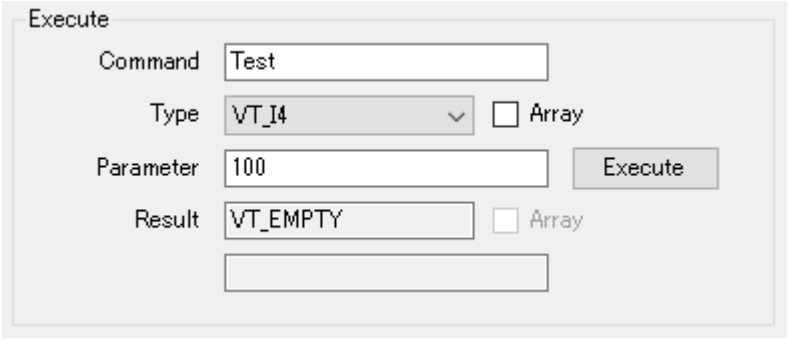
3. Appendix

Appendix A. How to specify write value and type information

You can specify numeric values, character strings, numeric arrays, RAC strings, etc. as arguments for executing CAO's extended method (Execute). The method of setting in CaoTester2 is explained in the form of an extended method. For details on the data types that can be used in CaoTester2, see Appendix B.

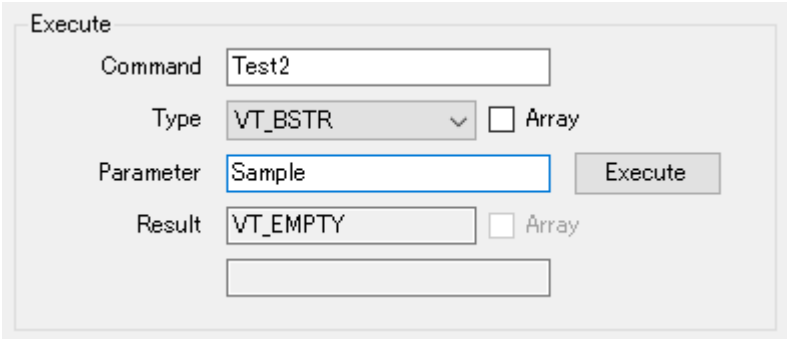
[Specify value]

The following figure shows how 100 of VT_I4 was specified in the "Test" command of Execute.



[Specify character string]

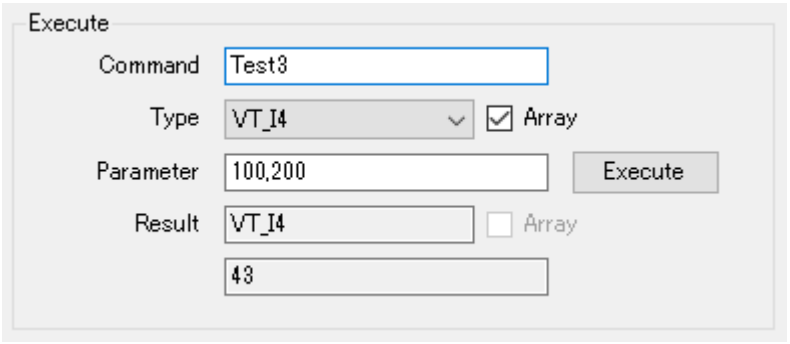
The following figure shows how Execute "Test2" command is executed by specifying "Sample" of VT_BSTR.



[Specify a numeric array]

To specify an array, select Array checkbox.

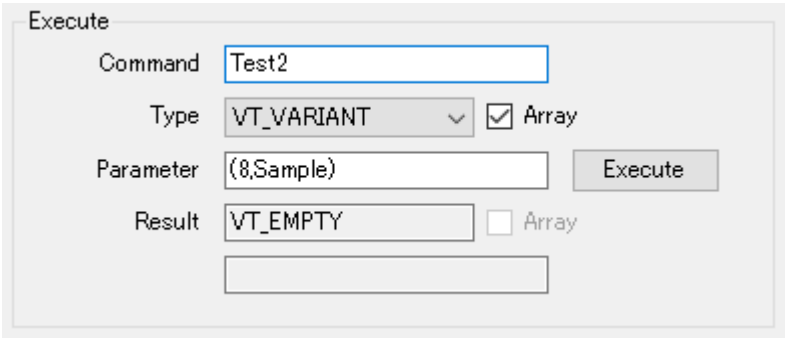
The following is an example of executing Execute "Test3" command with (VT_I4 | VT_ARRAY) (100, 200).



[Specify RAC character string]

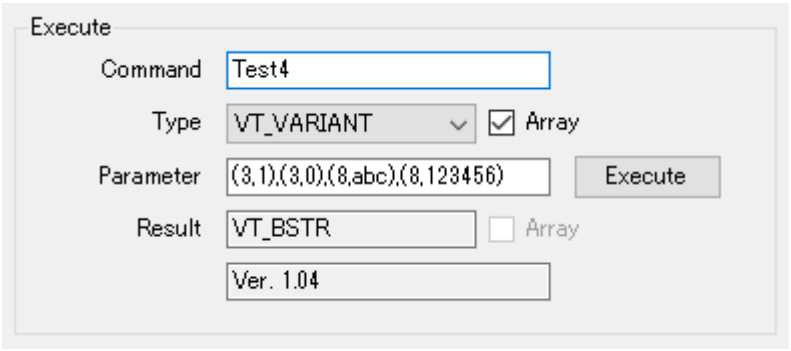
You can also use RAC strings to specify simple types.

The following figure shows how Execute "Test3" command is executed when (8, Sample) of (VT_VARIANT) is specified.



[Specify a complex RAC string]

The following example shows how to execute Execute "Test4" command by specifying (3,1), (3,0), (8,abc), and (8,123456) of (VT_VARIANT | VT_ARRAY).



Appendix B. Available data types

The following table lists the datatypes that can be used in CaoTester2.

For details on how to specify a RAC character string, see "3.2. Data Description Method" in the RAC User's Guide.

Data Type	Value	Meaning
VT_EMPTY	0	Value is unspecified
VT_NULL	1	NULL
VT_BOOL	11	Boolean type
VT_I1	16	Signed 1-Byte Integer Type
VT_UI1	17	Unsigned 1-byte integer type
VT_I2	2	Signed 2-Byte Integer Type
VT_UI2	18	Unsigned 2-byte integer type
VT_I4	3	Signed 4-byte integer type
VT_UI4	19	Unsigned 4-byte integer type
VT_I8	20	Signed 8-byte integer type
VT_UI8	21	Unsigned 8-byte integer type
VT_R4	4	Single-precision floating-point
VT_R8	5	Double floating-point type
VT_BSTR	8	Character string type
VT_DATE	7	Date Types
VT_CY	6	Currency Type
VT_VARIANT	12	VARIANT type [)7]
VT_ARRAY	8192	Array type [)8]

[7] It must always be used in conjunction with VT_ARRAY when using VARIANT type.
[8] If you use VT_ARRAY for a type that cannot be an array of VT_EMPTY or VT_NULL, the array is ignored.

Appendix C. License

Appendix C.1. DockPanel Suite

The MIT License

Copyright (c) 2007-2012 Weifen Luo (email: weifenluo@yahoo.com) and other contributors

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.