Realize your vision

${solution_edition}
${last_modified}

삼성SDS SAMSUNG

Table of contents

Table of contents ..... **ii**

Preface ..... **vi**

1. Overview **1**

1.1 Introduction to IPA Solution 1

1.2 Supported Features 1

1.3 Use Cases 2

2. Getting started **4**

2.1 System requirements 4

2.2 Preparing the license 5

2.3 Installing IPA Designer 5

2.4 Installing IPA Bot 7

3. Using IPA Designer **9**

3.1 Overview of the IPA Designer screen 9

Menu 10

Designer setting 11

Toolbar 13

Project screen 14

Search screen for processes 15

Library screen 16

Screen for process flow library 17

Design screen 17

Finding activities 18

Changing the chart direction 18

Viewing the mini map 19

Changing the screen size 20

Output screen 21

Shared Resource screen 22

Variable screen 22
Properties screen 24
3.2 Starting a new project and process flow 25
Creating a new project or process flow 26
Creating a new task 27
Creating a new process 28
Creating a new event 28
Adding a resource 28
Creating a new group 29
Editing a project 30
Changing task name 31
Copying task 31
Moving task 32
Deleting task 32
3.3 Saving a project and process flow 33
3.4 Running a project 34
3.5 Running a process flow 35
4. Designing projects 36
4.1 Designing with the recording feature 36
4.2 Designing with the library 39
Adding an activity 39
Adding an activity to the link 39
Adding an activity to activity group 40
Adding an activity between groups 41
Editing an activity 41
Deleting an activity 41
Copying an activity 1 41
Copying an activity 2 41
Moving an activity 1 42
Moving an activity 2 42
Adding a task 42
Selecting multiple activities 42

Inverting selection 43
Deactivating 43
Folding an activity group 43
Learn more about the library 43
App and Web libraries 43
Excel library 78
SAP library 111
Image recognition library 119
System library 135
FlowControl library 142
Knox Portal Library 151
Private Script Library 157
5. Designing process flows 160
5.1 Designing with processes 160
Adding a process 160
Adding waiting action 161
5.2 Designing with the library 162
Learn more about the process flow library 163
6. Linking to server 169
6.1 Preparing server linkage 169
6.2 Activating and logging in 170
Activating 171
Logging in 171
6.3 Publishing project and process flow 172
6.4 Importing project and process flow 173
6.5 Using shared resources 174
7. Using IPA Bot 176
7.1 Setting up use environment 176
7.2 Running the bot 178
Appendix A. Troubleshooting 180
A.1 Problems related to installation and registration 180

A.2 Problems related to writing and executing automation scenario 181

A.3 Problems related to linking servers 183

Appendix B. FAQ 186

Terminology 187

Preface

Legal Notice

> **Please read the following information carefully before using this manual.**

Author	**Samsung SDS Co., Ltd.**
Address	125 Olympic-ro 35-gil, Songpa-gu, Seoul
Telephone	+82 2 1661 3388
Email address	global.cs@samsung.com
Website	www.samsungsds.com

Samsung SDS Co. Ltd. (hereinafter Samsung SDS) has verified the information contained in this document. However, Samsung SDS is not responsible for any problems that may arise from inaccurate content or typographical errors.

The content and specifications in this document are subject to change without notice. For more information on the revision history of this manual, visit the SDS website (www.samsungsds.com).

Samsung SDS Co. Ltd. holds all intellectual property rights, including the copyrights, to this document. Using, copying, disclosing to a third party, or distributing this document without explicit permission from Samsung SDS is strictly prohibited. These activities constitute an infringement of the intellectual property rights of this company.

Copyright © SAMSUNG SDS All rights reserved

Users of this guide

This manual is intended for the users who design or apply the work automation process using the Samsung SDS IPA (hereinafter IPA) solution.

The users must have following knowledge and understanding to use the IPA Solution effectively.

- Basic knowledge of job automation
- Basic knowledge of Windows-based apps

Summary

This guide consists of the following chapters:

Chapter 1. Overview

Provides an overview of the IPA Solution.

Chapter 2. Getting started

Explains the prerequisites required for using the IPA Solution and the installation method.

Chapter 3. Using IPA Designer

Explains the basic method of using the IPA Designer.

Chapter 4. Designing projects

Explains how to design work automation projects using the IPA Designer.

Chapter 5. Designing process flows

Explains how to design process flows for work automation using the IPA Designer.

Chapter 6. Linking to server

Explains the features that can be used by linking to the IPA server.

Chapter 7. Using IPA Bot

Explains how to use the IPA Bot.

Appendix A. Troubleshooting

Provides a guide for the main issues that arise when using the IPA Solution.

Appendix B. FAQ

Provides answers to frequently asked questions when using the IPA Solution.

Terminology

Explains the terminology used in the manual.

Conventions

This document uses the following conventions:

Boldface	**Boldface** indicates various graphic user interface elements, menus, and directories within the main text. However, portals, windows, pop-up windows, wizards, and pages are expressed within double quotation marks and the boldfaced typeface is not used.
Italics	*Italics* typeface indicates variables or parameters that a user must supply particular values.
Monospace	Monospace typeface indicates programming terms or codes. For a clear classification in the main text, **`bold monospace`** font is used for commands, parameter names, registry names, script names, and process names. *`Italic monospace`* font is used for variables and parameters.

Notes and notices

A note identifies additional information for users, such as tips, recommendations, exceptions, and restrictions on using the product.

If you create a project, an empty process is automatically created.

An attention notice identifies possible damages to the system, data, or device.

Input values on the Properties window must be entered according to the JavaScript grammar, because the values are converted to JavaScript upon execution.

Revision history

Solution version	Manual version	Revised date	Revised details
v1.0.1.11191	1.0	2018.11.30	First edition of manual

1. Overview

Users can design or apply the work automation processes of IPA Solution using the IPA Designer and the IPA Bot. To use the IPA Solution effectively, learn about the purpose and the features of the solution.

"Overview," explains the following topics:

- Introduction to IPA Solution
- Supported Features
- Use Cases

1.1 Introduction to IPA Solution

The Intelligent Process Automation (IPA) solution enables automation of standardized human tasks to be carried out by bots. Teach the user actions to the bot and the bot will mimic the learned actions. While the bot autonomously processes simple repetitive tasks, the user can focus on the work that requires more creativity and also improve job productivity, accuracy, and efficiency, while reducing costs.

1.2 Supported Features

The IPA Solution supports environments including the Windows app, Internet Explorer, Excel, and SAP.
You can use the solution for a range of work, such as rule-based, simple repetitive tasks, input and output of standardized data between systems, and comparisons of source data and processed data.

1.3 Use Cases

Automation of Knox Portal billing process

Automate the billing process of the Knox Portal using the IPA Solution.
The processes below that were originally performed by three employees in one month are now handled automatically.

Process	Target app	Details
Sending bill by email	Web, Excel	• Access the Knox Portal payment system • Download statements • Organize Excel templates • Send email to the persons in charge of each company
Sending an official document for payment confirmation	Web, Excel	• Check replies • Check and update billing statements • Create official documents • Send official documents
Creating contracts	SAP, Excel	• Access the ITO contract screen • Check SAP contract information • Check person in charge • Create contract

Automation of Daily Work Reporting Process

The processes of creating a work report by extracting the required data from the system and sending the report via email are handled automatically. The example below is based on a shipping work of a business.

Process	Target app	Details
Creating a work report	Web, Excel	• Extract data from the work system (e.g. release quantity, shipment quantity of previous day, and shipment deadline) • Automatically enter data into Excel file • Save
Sending email	Web	• Designate the recipient • Fill out the subject • Attach the report • Send email

2.
Getting started

To use the IPA Solution, you must first get the license ready and then install the IPA Designer and IPA Bot.

"Getting started," explains the following topics:

- System requirements
- Preparing the license
- Installing IPA Designer
- Installing IPA Bot

2.1 System requirements

The hardware and software requirements for installing and running the IPA Solution are as follows:

Hardware

- CPU Dual Core or higher
- Memory 4 GB or higher
- HDD 200 GB or higher

Software

- Windows 7 64-bit or later
- .NET Framework 4.5 or later
- Internet Explorer 11 or later

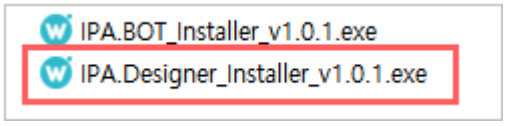
2.2 Preparing the license

Contact your administrator to obtain the license.

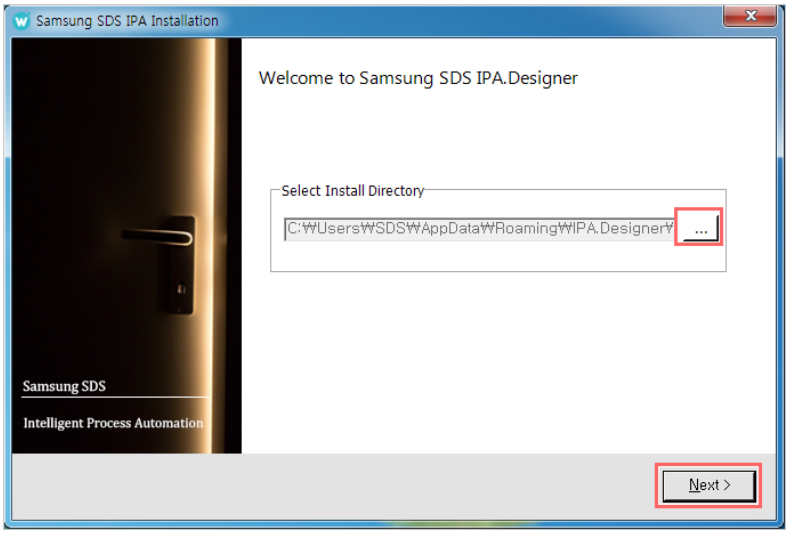
2.3 Installing IPA Designer

To install IPA Designer, complete the following steps:

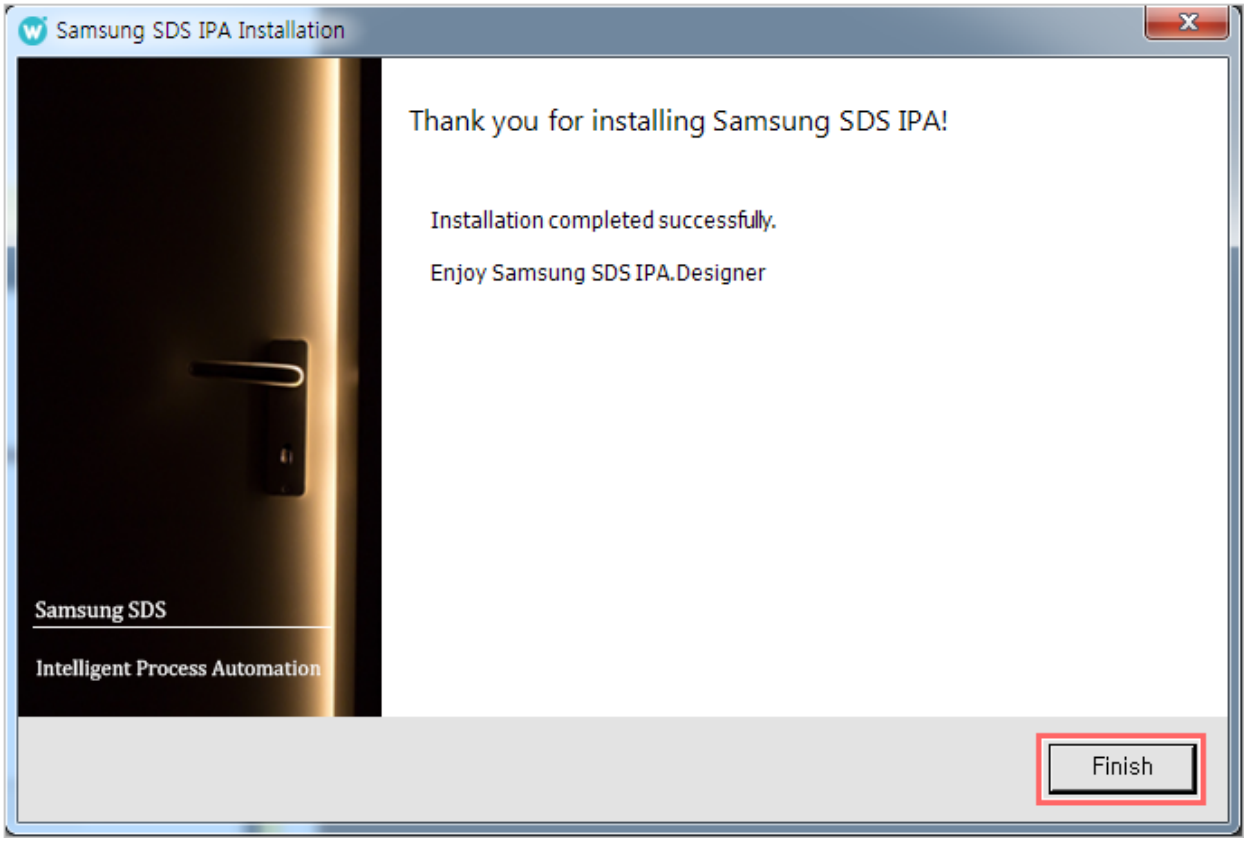
1. Run the IPA Designer installation file.



2. Select the install directory and click **Next**.

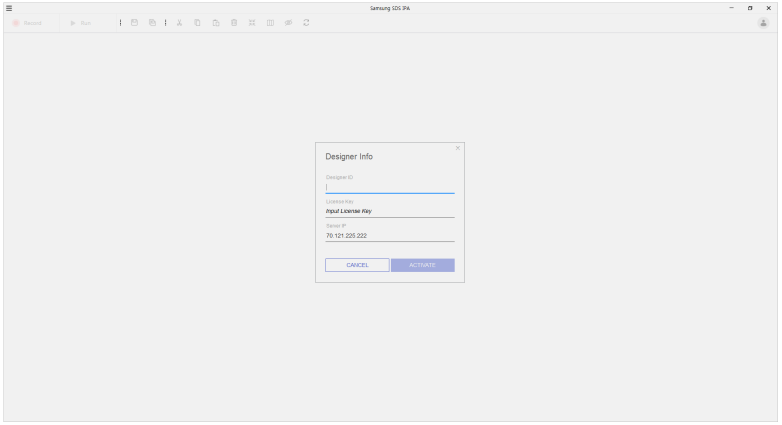


3. Click **Finish** to complete the installation.



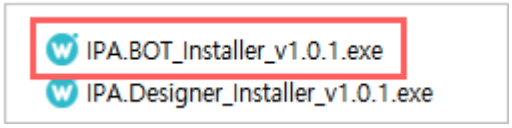
Once the installation is complete, a shortcut icon will be added to your desktop.

4. Double-click the shortcut icon to verify that IPA Designer runs properly.

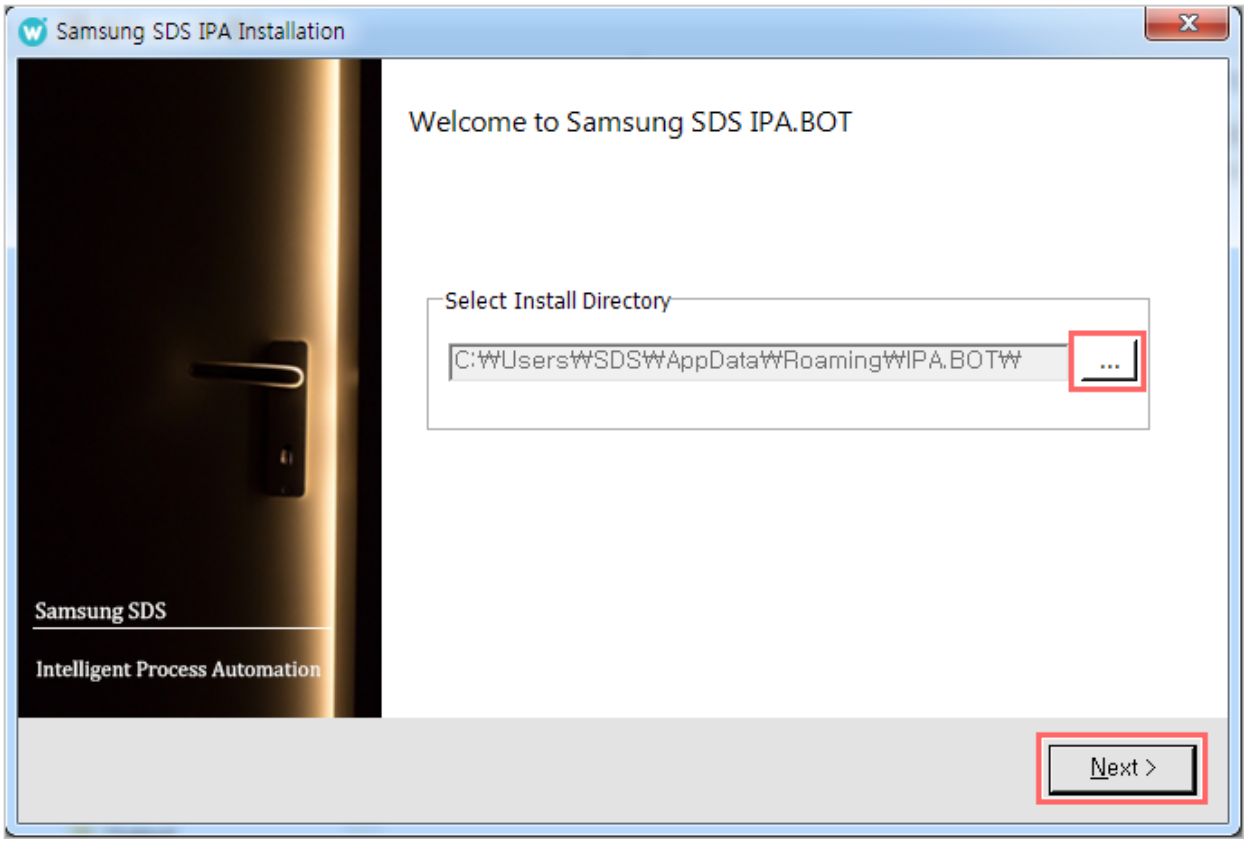


2.4 Installing IPA Bot

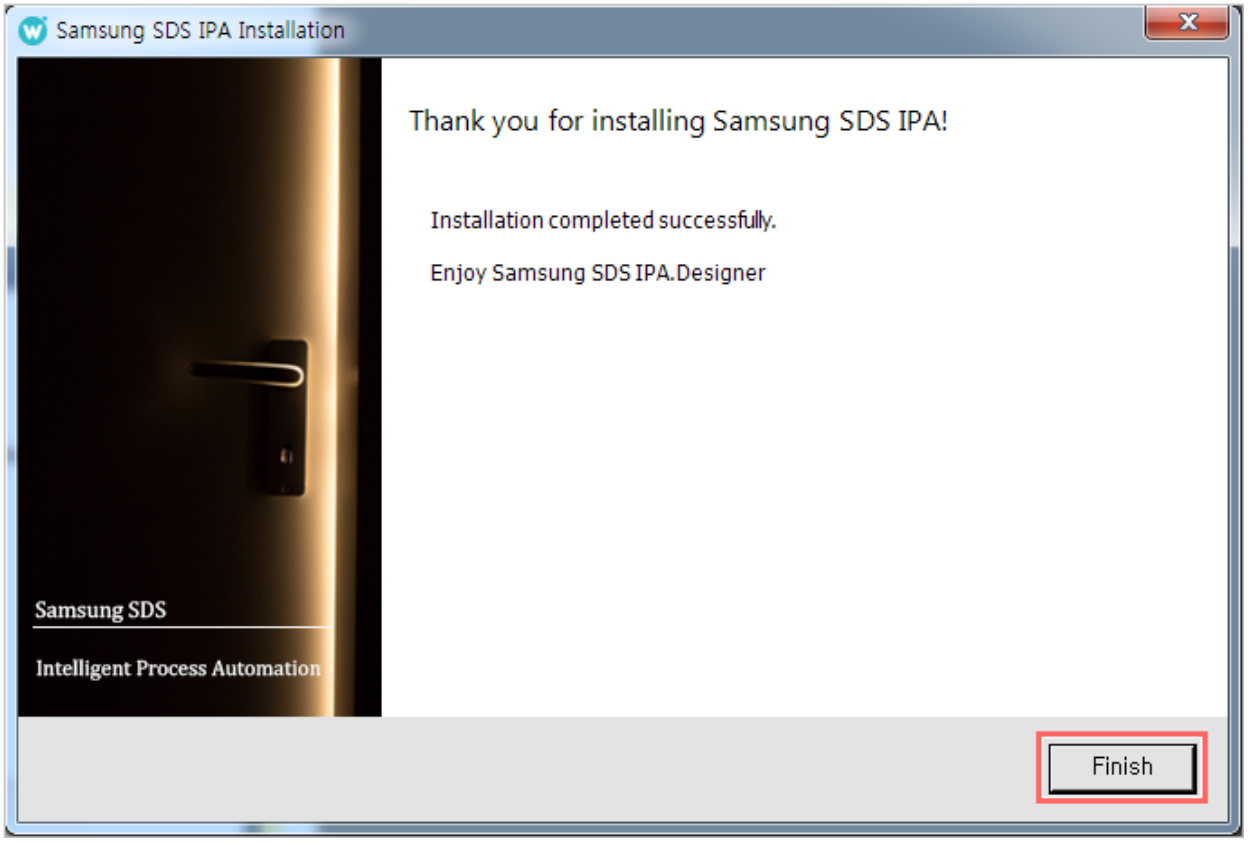
1. Run the IPA Bot installation file.



2. Select the install directory and click **Next**.



3. Click **Finish** to complete the installation.



Once the installation is complete, a shortcut icon will be added to your desktop and also to the bottom-right corner of the taskbar.

4. Double-click the shortcut icon to verify that IPA Bot runs properly.

3.
Using IPA Designer

IPA Designer is a program for users to design automated projects or process flows. Before designing, you should have an understanding of the screen layout and features and an understanding of the basic method for using the IPA Designer.

"Using IPA Designer," explains the following topics:

- Overview of the IPA Designer screen
- Starting a new project and process flow
- Saving a project and process flow
- Running a project
- Running a process flow

3.1 Overview of the IPA Designer screen

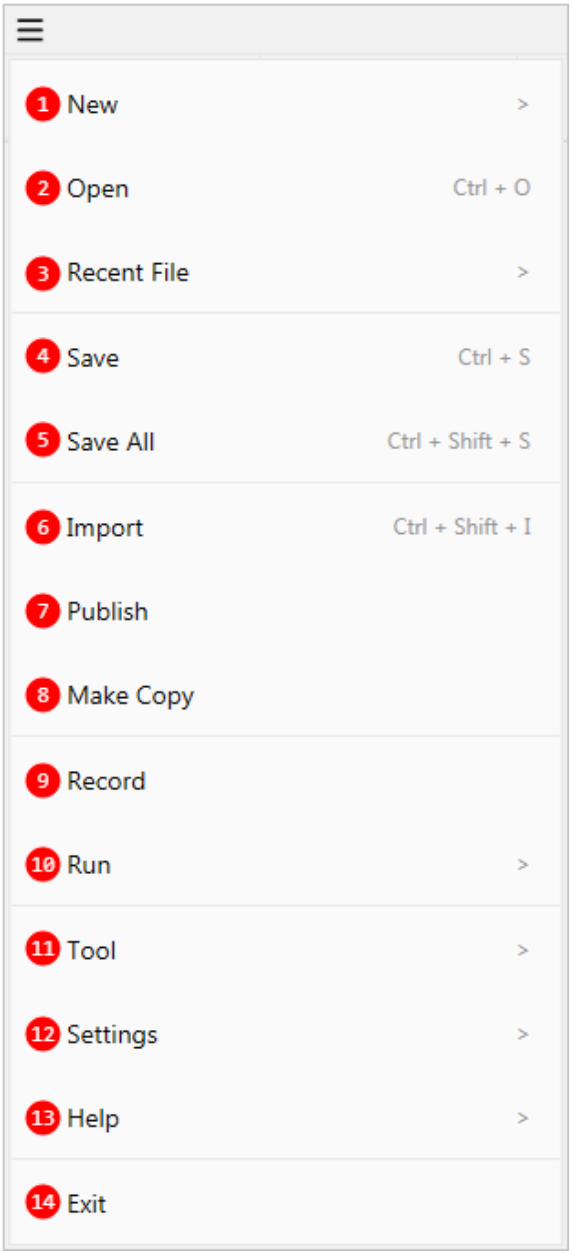
You can see the screen layout of the IPA Designer and learn how to use the program. IPA Designer is made up of the following screen components:

- Menu
- Toolbar
- Project screen
- Search screen for process flow
- Library screen
- Screen for process flow library
- Design screen

- Output screen
- Variable screen
- Public resource screen
- Properties screen

Menu

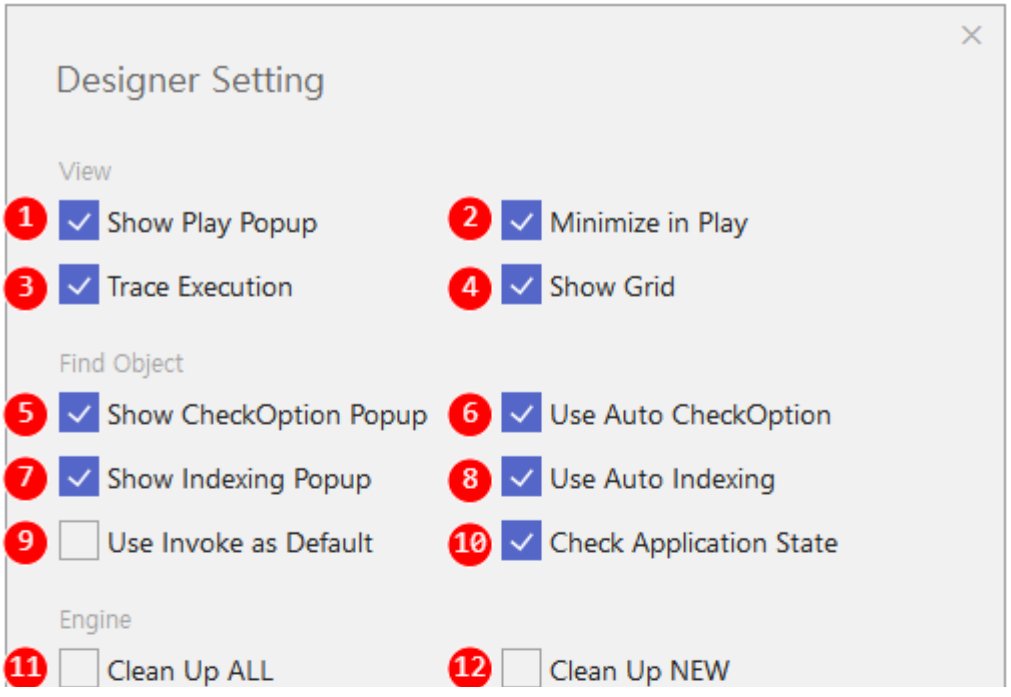
You can use the basic menus and the settings menus of IPA Designer.
Click the hamburger icon (☰) in the upper-left corner of the screen.

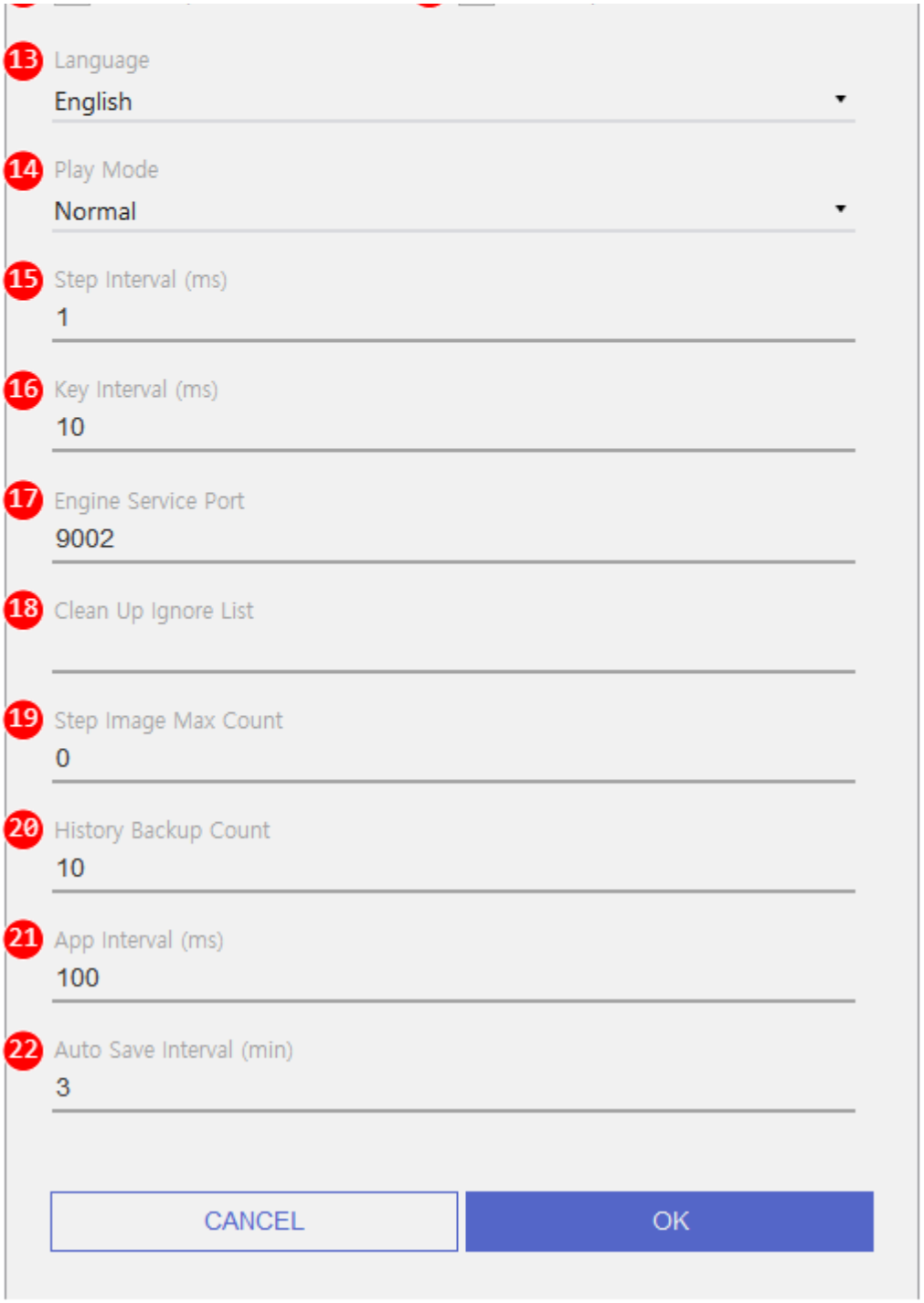


No.	Description
❶	Create new project or process flow.
❷	Open saved project or process flow.
❸	Open most recently used file.
❹	Save the file you are currently working on.
❺	Save all opened files.
❻	Import projects or process flows from the server.
❼	Publish projects or process flows to the server.
❽	Make a copy of the project or process flow you are currently working on.
❾	Design a project with the recording feature.
❿	Run or stop a designed project.
⓫	Check and scan CPU and memory usage.
⓬	Change the IPA Designer and the server settings.
⓭	Check the Help menu.
⓮	Exit the program.

Designer setting

You can change the IPA Designer settings.
Click **Settings** > **Designer Setting** from the menu.





No.	Description
❶	Show popup asking whether to minimize the designer when it runs.
❷	Set whether to minimize designer when it runs.
❸	Track which step is currently in progress and highlights it in the designer.
❹	Show grid in design area.
❺	Show popup asking whether to set check options.

No.	Description
❻	Set check option automatically.
❼	Show popup asking whether to set index information.
❽	Set index automatically.
❾	Set whether to use invoke as default.
❿	Ignore it if the target app is not responding.
⓫	Terminate all apps for stable automation.
⓬	Terminate all apps executed by bot for stable automation.
⓭	Set the designer language pack.
⓮	Set whether to run in normal mode, stability, or performance first.
⓯	Set the execution mode. (normal mode, stability priority mode, performance priority mode)
⓰	Set the key input interval.
⓱	Set the port on which the engine will run.
⓲	Set up a list to ignore when cleaning apps. The list can be divided by semicolon. (ex : notepad.exe;mspaint.exe)
⓳	Set the maximum number of images to save the execution screen. For example, if there are 10 steps and the maximum number is 3, the screen will be saved when the last 3 steps - 8th, 9th, and 10th - are executed.
⓴	Set the number of execution history to back up.
㉑	Set the execution interval when the app changes for each step. Since app is separated by app title, even if the same app, different titles are considered different apps. Separate by name. (ex : notepad - untitled, notepad - test)
㉒	Set the interval to save process or task automatically when editing.

Toolbar

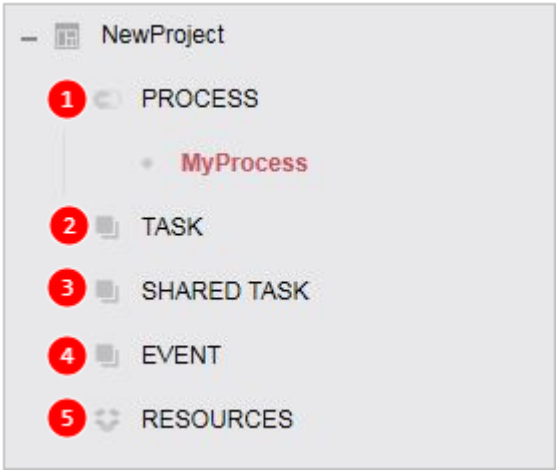
Toolbar is a collection of useful tools that is needed for you to use the IPA Designer. The tools are arranged in the order of Design, Run, Save, and Edit features.

You can change arrangement of tools in the toolbar by dragging the grouped features.

No.	Description
❶	Start recording.
❷	Run the currently open process or task.
❸	Save the currently open process or task.
❹	Save all open processes or task.
❺	Cancel the execution.
❻	Cut the selected node.
❼	Copy the selected node.
❽	Paste the copied or cut node.
❾	Delete the selected node.
❿	Fold or unfold the selected activity group.
⓫	Show or hide the mini map.
⓬	Deactivate the selected node.
⓭	Reverse select the selected nodes.

Project screen

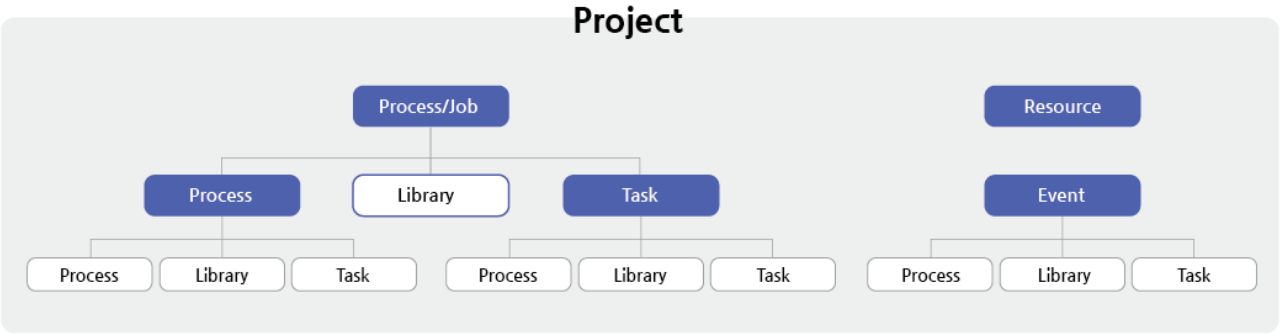
The project screen shows the structure of the project. A project is made up of processes, tasks, events, and resources.



No.	Description
❶	A process is a unit of work performed by a bot. It is composed of various libraries and tasks. It may have a sub-group that can include additional processes other than itself. Right-click on the process to

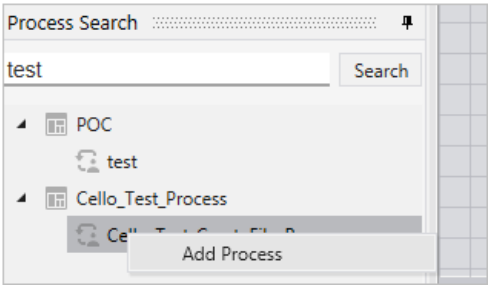
No.	Description
	use the editing features, such as Edit Name, Delete, Import from Server.
❷	A task is a unit of work just like a process that is made up of various libraries and tasks. However, a task is a unit of work that cannot be performed by a bot. It may have a sub-group that can include additional tasks other than itself. Right-click on the process to use the editing features, such as Edit Name, Delete, Import from Server.
❸	A shared task is a unit of work that is published on the server and used within your project when the project is running. It cannot be modified but can be used in its latest version always. If you copy a shared task, you can modify it when necessary.
❹	An event can be used before or after performing a process or task, or when an error occurs. It is composed of the same contents as a task, but the Import from Server feature is not available.
❺	A resource is a unit of the file that is required to carry out a project. It imports files from the user PC and manages them separately in the project folder.

For more information on the relationships between the items that make up the project, see the image below.



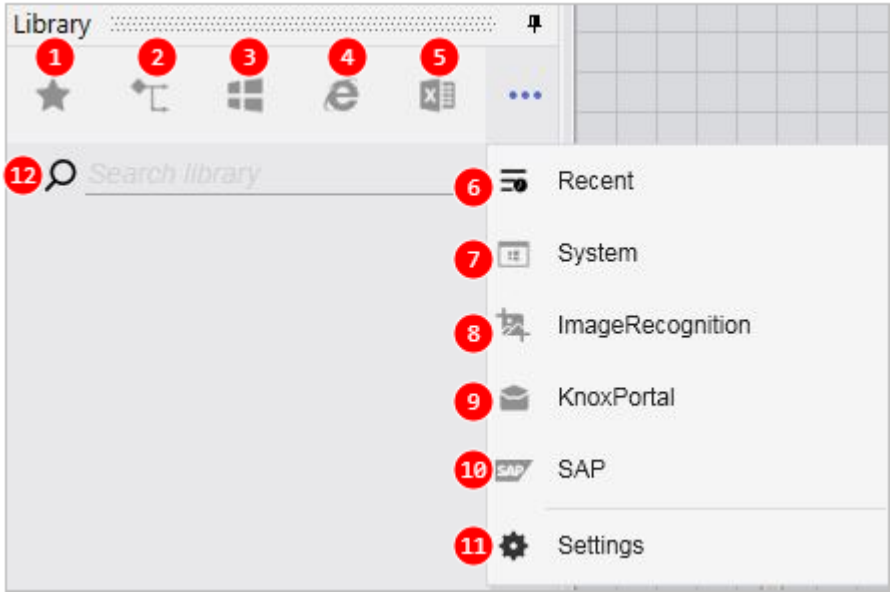
Search screen for processes

The search screen for processes can be used to search and use the processes for designing process flows. For more information, see "4. Designing projects"



Library screen

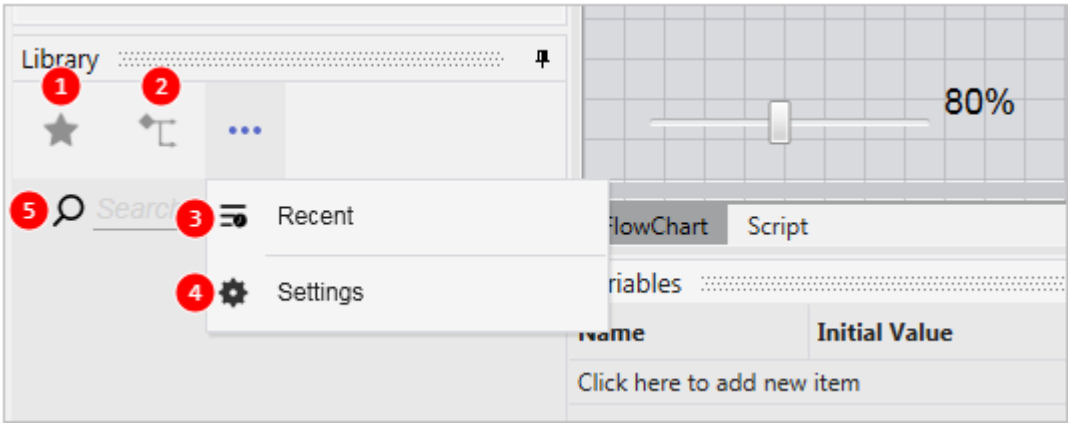
The library screen is composed of various features. Users can find the feature they need and create a flowchart for processes, tasks, and events. A library is classified into app, web, Excel, SAP, image recognition, system, and flow control items depending on the target app or feature. Select an item to use from the library screen and find the feature you want. You can also add the frequently used library to your favorite libraries. For more information on the available libraries and detailed description, see "4.2 Designing with the library" and "Learn more about the library."



No.	Description
❶	Favorite libraries
❷	FlowControl library
❸	App library
❹	Web library
❺	Excel library
❻	Recently used library
❼	System library
❽	Image recognition library
❾	KnoxPortal library
❿	SAP library
⓫	Library settings
⓬	Library search

Screen for process flow library

On the design screen for process flows, you can use the process flow library. The library screen to create a flowchart make up of **If** or **For** is identical to the project design screen. For more information on the process flow library, see "Learn more about the process flow library."

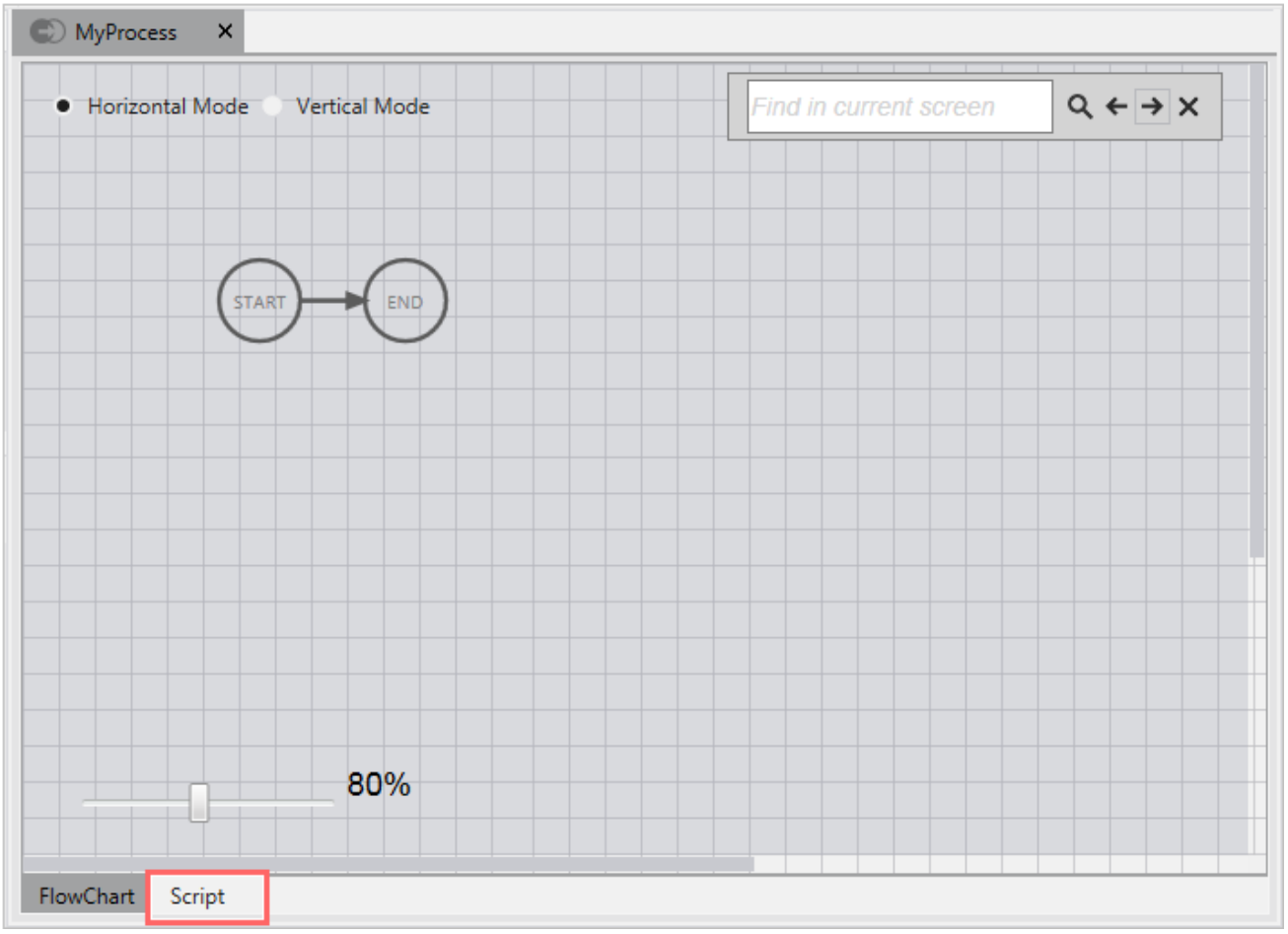


No.	Description
❶	Favorite libraries
❷	Process flow library
❸	Recently used library
❹	Library settings
❺	Library search

Design screen

This is the main screen of IPA Designer. You can create and edit flowcharts by adding various libraries and tasks. All flowcharts are converted into scripts and you can find the converted scripts by clicking the **Script** tab.

- Upon clicking the **Script** tab, the changed flowchart is automatically saved.
- For the libraries that can be used or writing scripts, see Private Script Library.



Finding activities

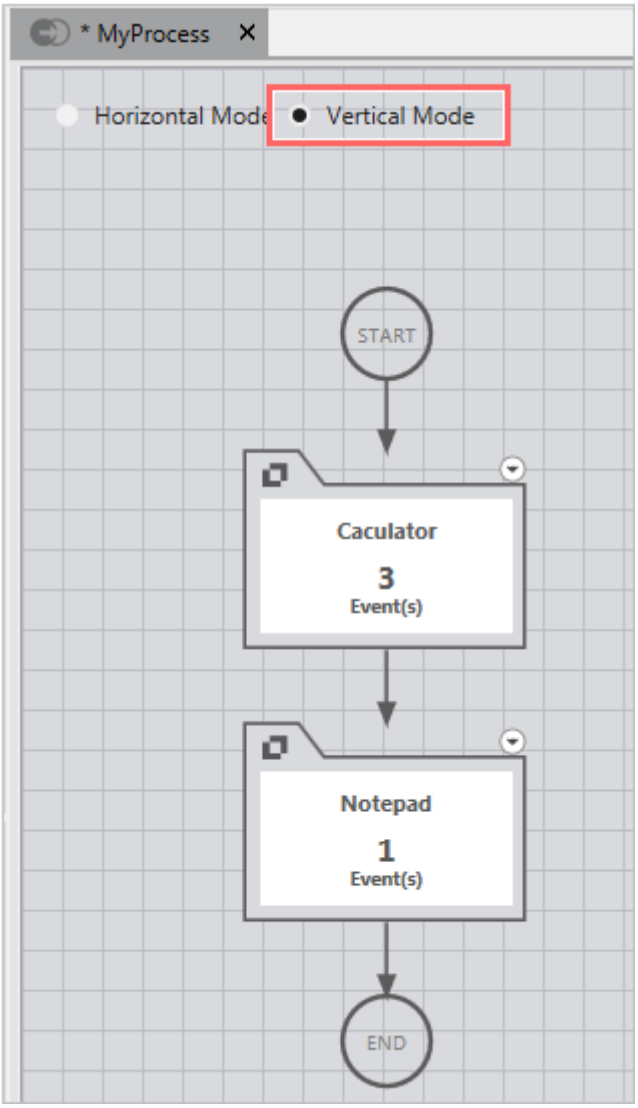
You can search the activities that contain the entered information.

On the search bar of the screen, enter a keyword to search, and click the search icon ( ) or press the Enter key on the keyboard.

The node that contains the keyword is emphasized and the first node is selected.

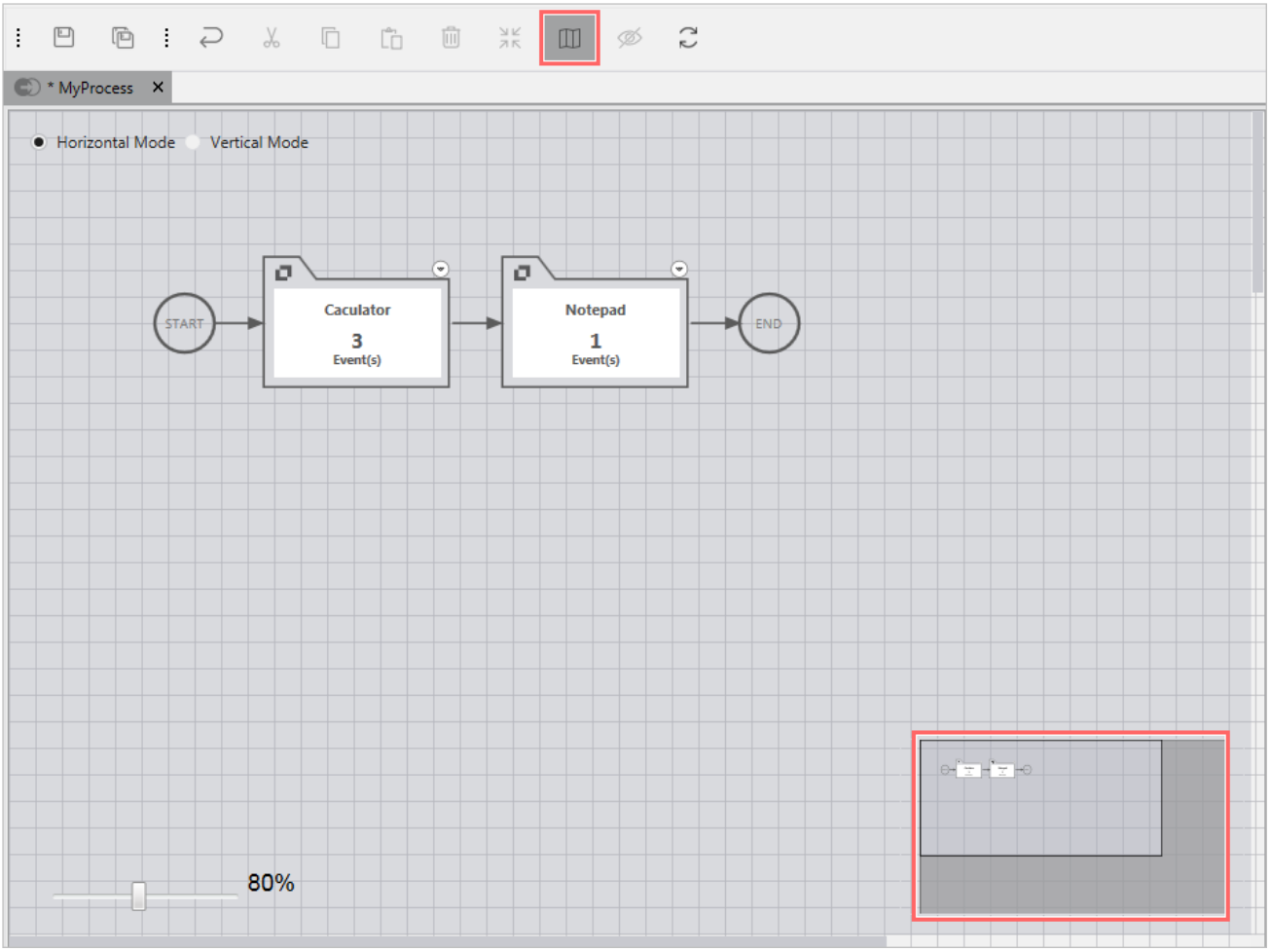
Changing the chart direction

To rotate the chart vertically, click **Vertical Mode** on the design screen.



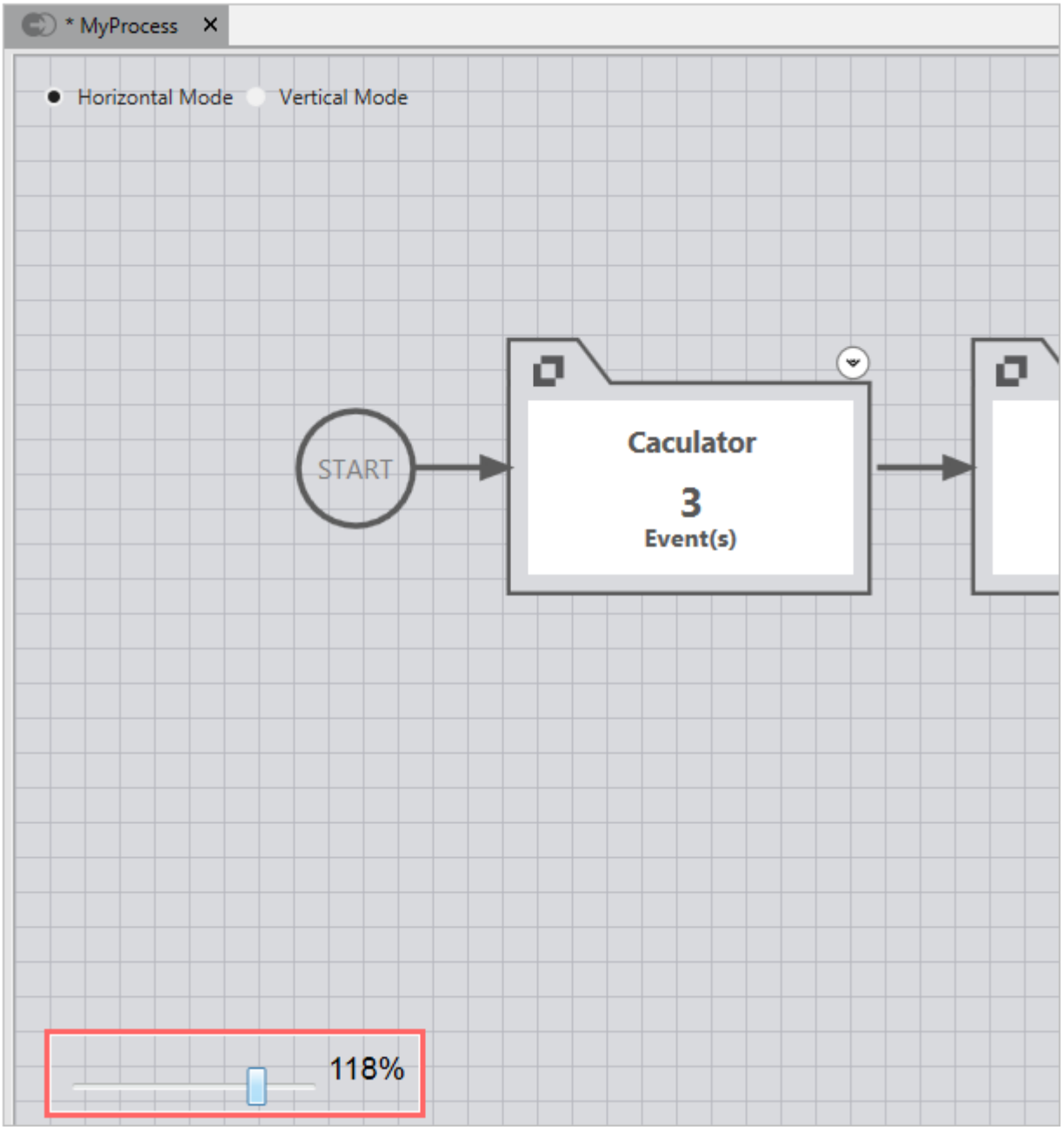
Viewing the mini map

You can click on the mini map icon ( ) to display the mini map on the screen. Click on the mini map to move to a desired position on the screen.



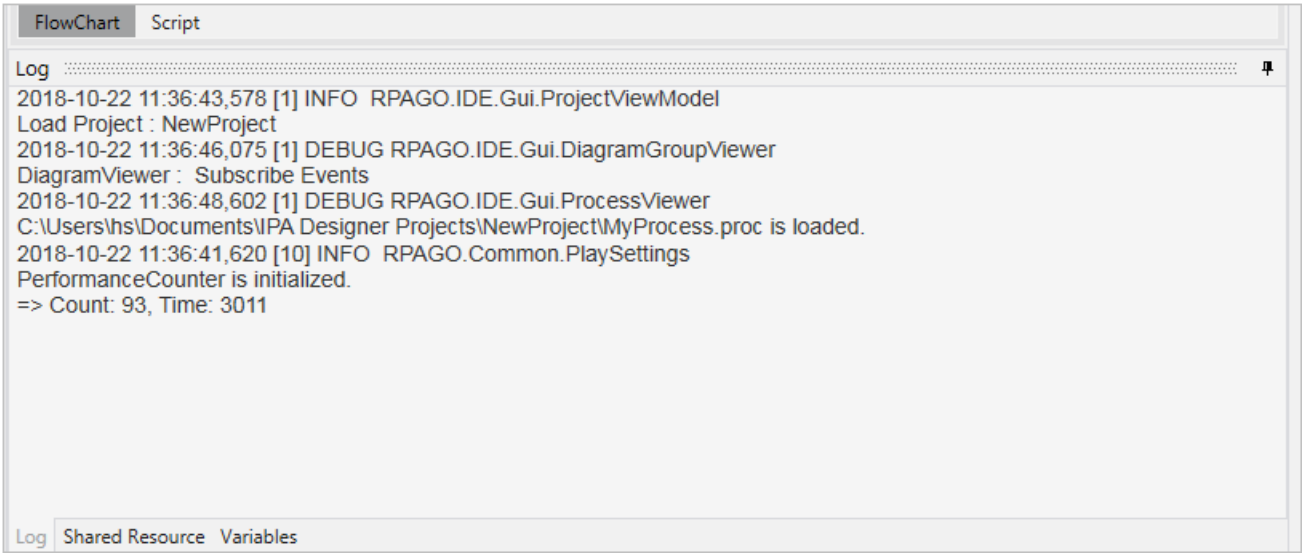
Changing the screen size

If the flowchart becomes too long and it is difficult to see all content in a screen, you can zoom in or out.
Click and drag the zoom slider at the lower-left corner of the screen to change the screen size.



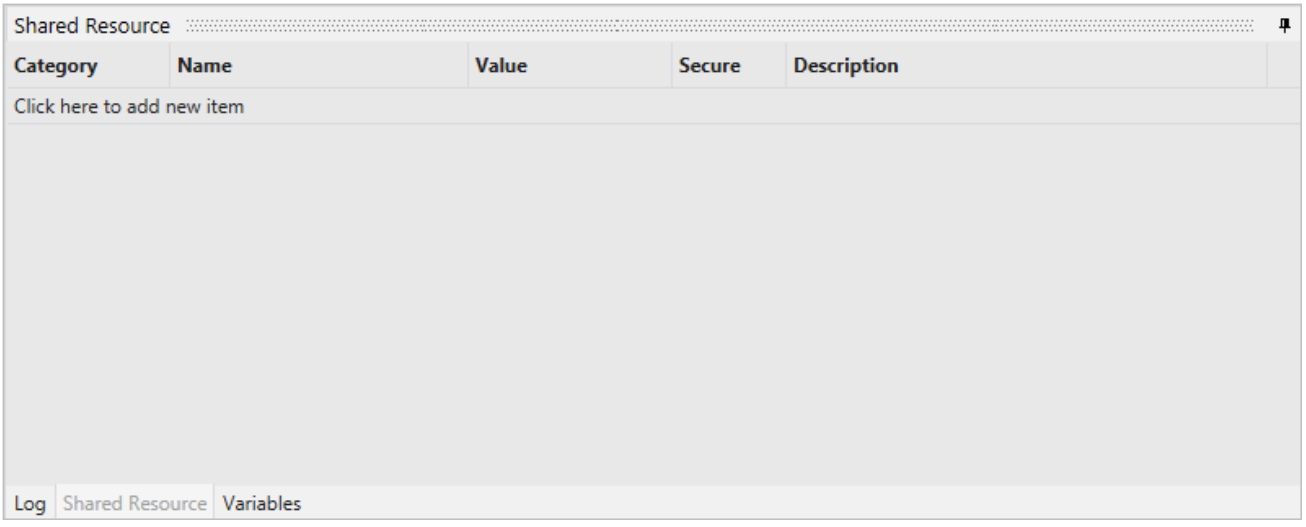
Output screen

This screen outputs logs when a process or task is running. You can check the current progress and error messages.



Shared Resource screen

This is a screen to import and use the public resources registered in the server. To use the public resource, you must first log-in. For more information, see "6.5 Using shared resources."



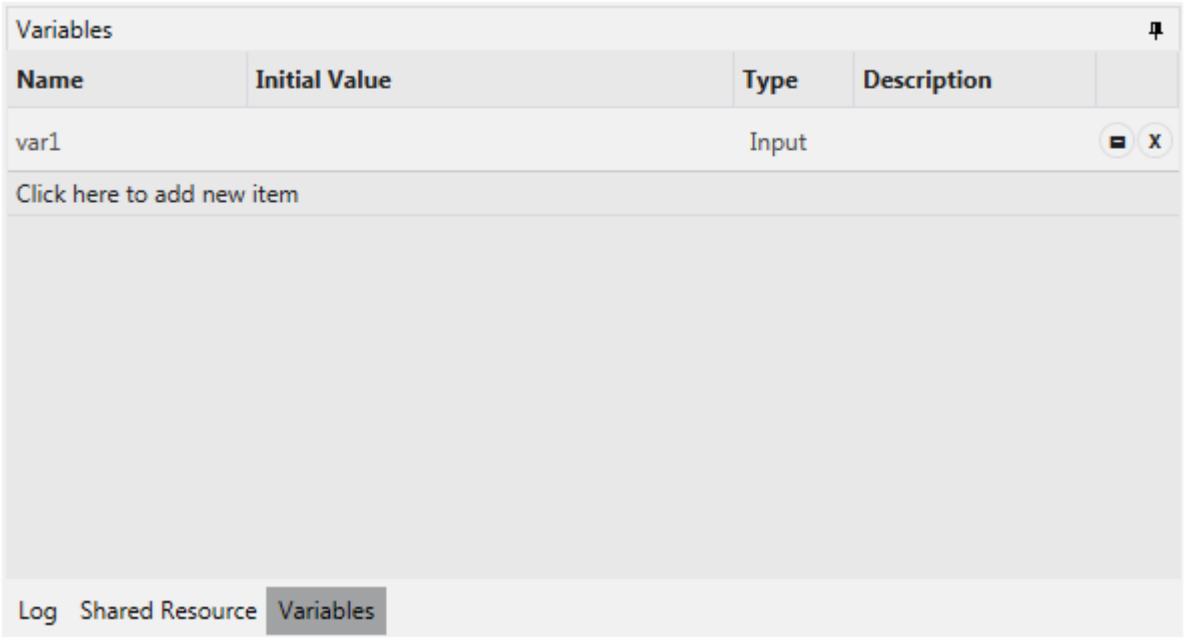
Variable screen

This is a screen to add and manage variables that are used as the property value of the library. You can set the initial value of a variable and add a description upon need to easily distinguish the variables.

A variable can be classified into four types depending on the scope and purpose of use.

- **Global**: A global variable that can be used in all parts of the current project.
- **Normal**: A member variable that can be used in the currently working process or task only.
- **Input**: A member variable that is required to send as a parameter when performing a process or task.
- **Output**: A member variable that is required to get the output value when a process of a task is completed.

For global type of variable, use keywords that include the word "Global," such as "Global.var1."
For normal, input, and output type of variables, use keywords that include the word "this," such as "this.var1."



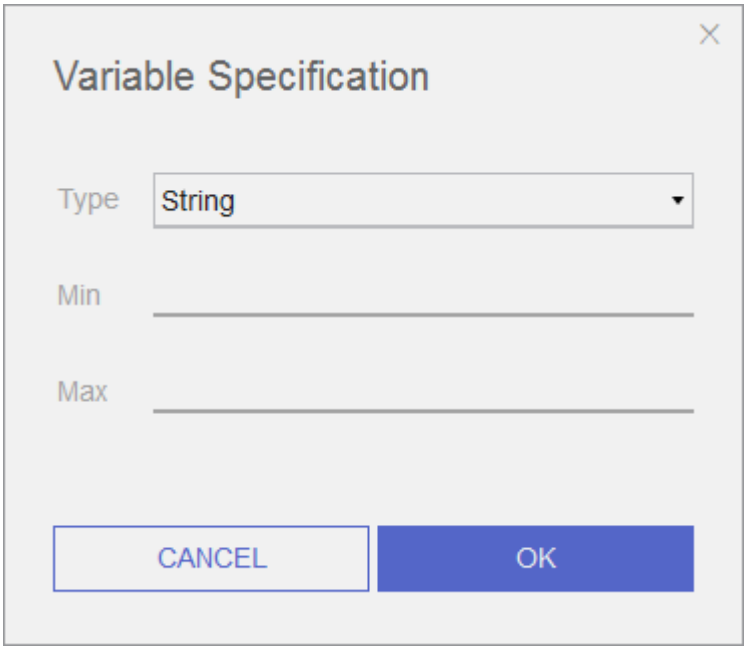
A variable can be deleted through **the Delete button**.

Alternatively, the variable can be deleted by selecting the variable and pressing **the Delete key.**

The detail value (type, range) of the variable can be specified through **the variable detail setting button**.

If you click the variable detail setting button, you can see the pop-up window as shown below.

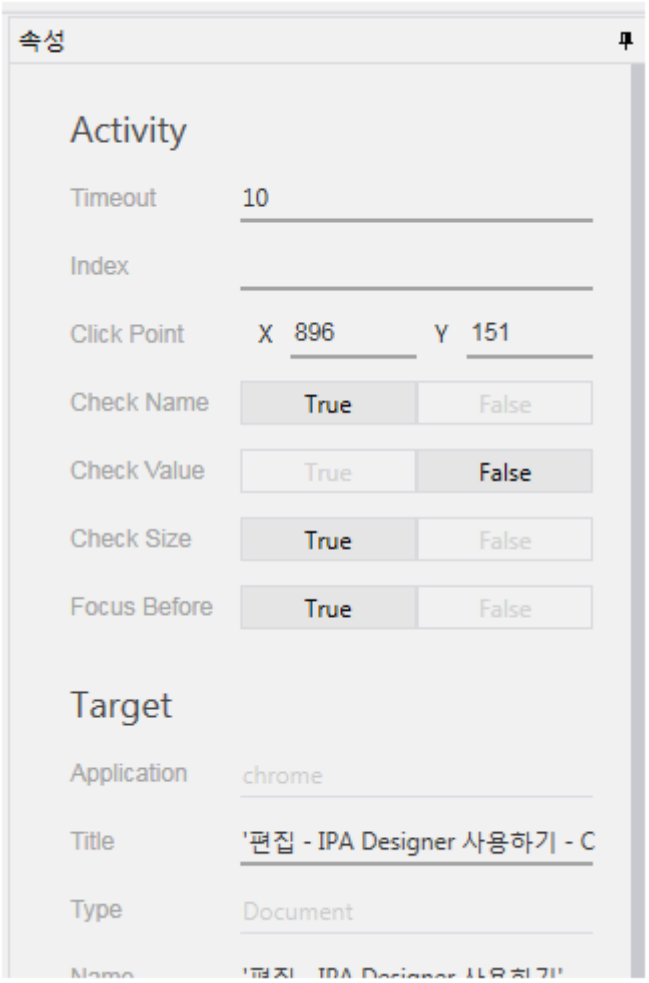
You can specify detail values by specifying **variable types and ranges** in variable detail popup.



The detail value of the variable can be used to limit the range of the input value if the variable value is required as an input, such as a chat bot command.

Properties screen

This is a screen for verifying details of the selected node and setting up the property value. The displayed information varies depending on the selected item. For more information, see the properties screen of each library in Learn more about the library.



Input values on the Properties window must be entered according to the JavaScript grammar, because the values are converted to JavaScript upon execution. For the input value, only the variables declared using "Global" and "this" keywords can be used and local variables declared in the script cannot be used.

3.2 Starting a new project and process flow

You can start a new project or process flow. Design a project by adding processes, tasks, events, or by adding existing resources. You can also manage tasks conveniently by grouping similar tasks.

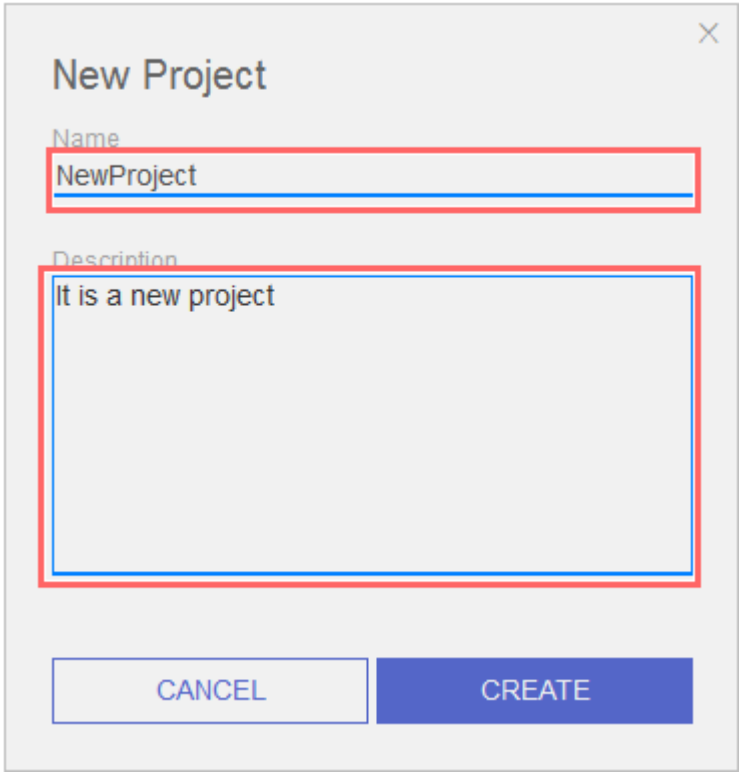
"Starting a new project and process flow" explains the following topics:

- Creating a new project or process flow
- Creating a new task
- Creating a new process
- Creating a new event
- Adding a resource
- Creating a new group
- Editing a project

Creating a new project or process flow

To create a new project or process flow, complete the following steps:

1. Click ☰ > **New**.
2. Click **New Project** or **New Process Flow**.
3. On the "New Project" pop-up window, enter the name and description of the new project or process flow to create.



4. Click **CREATE**.

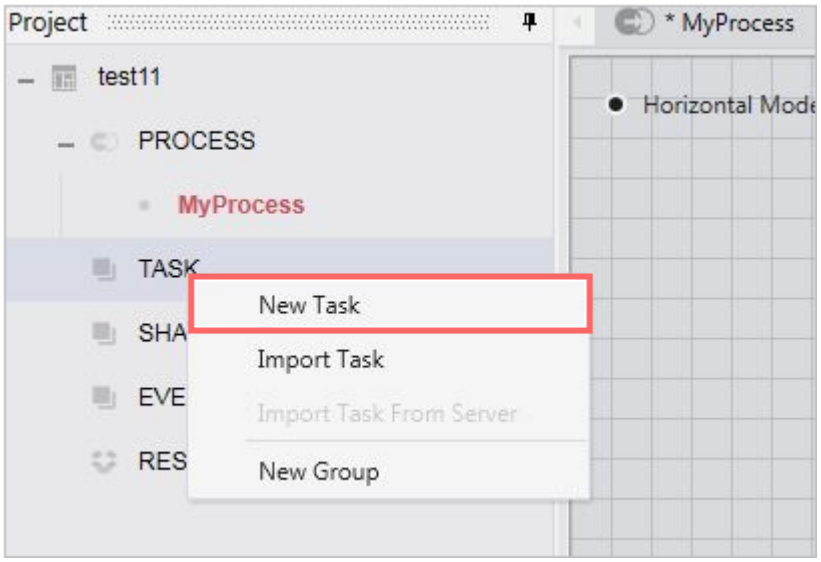
A new project or process flow is created.

When a new project is created, an empty process named "MyProcess" is created for user convenience.

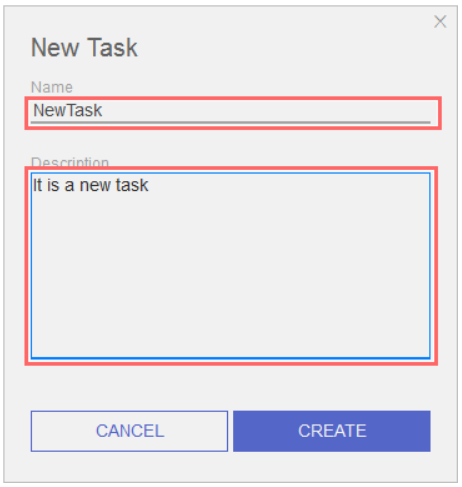
Creating a new task

To create a new task, complete the following steps:

1. On the project screen, right-click **TASK** and click **New Task**.



2. On the "New Task" pop-up window, enter the name of the new task to create and click **CREATE**.



A new task is created and the design screen will launch.

Creating a new process

To create a new process, complete the following steps:

1. On the project screen, right-click **PROCESS** and click **New Process**.
2. On the "New Process" pop-up window, enter the name of the new process to create and click **CREATE**.

A new process is created and the design screen will launch.

Creating a new event

To create a new event, complete the following steps:

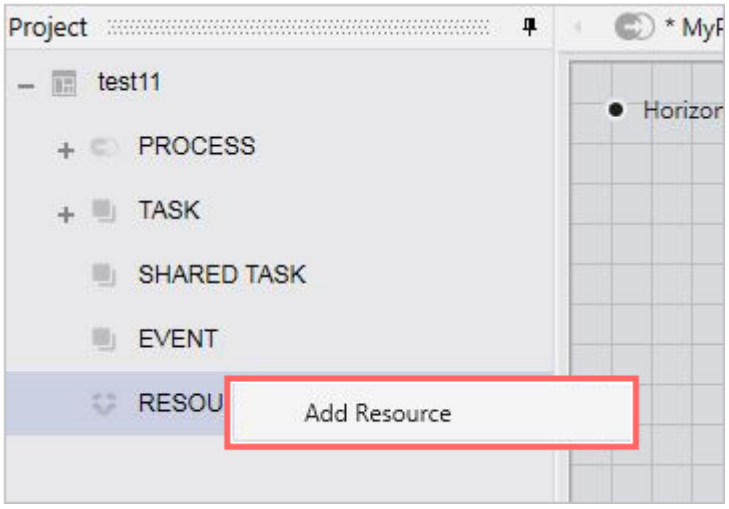
1. On the project screen, right-click **EVENT** and click **New Event**.
2. On the "New Event" pop-up window, enter the name of the new event to create and click **CREATE** .

An event is created and the design screen will launch.

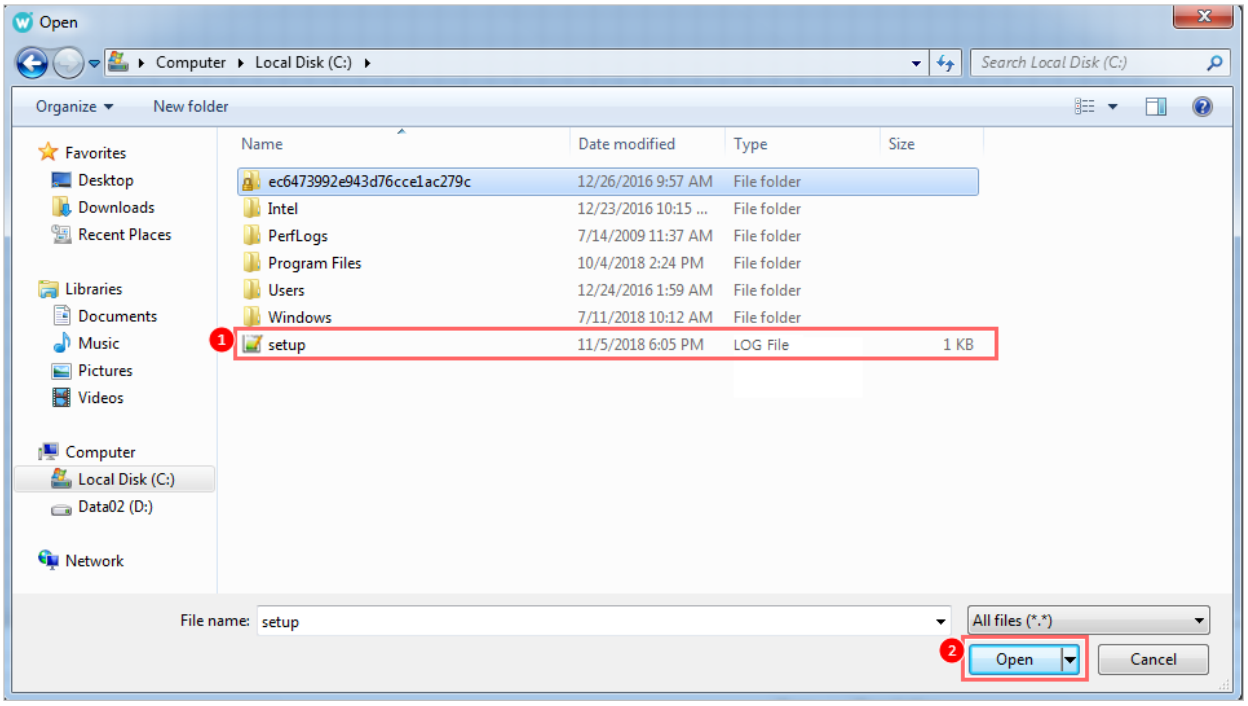
Adding a resource

You can add files to use in the project. The files here are managed in a separate folder in the project.

1. On the project screen, right-click **RESOURCES** and click **Add Resource**.



2. Select the file to add.
3. Click **Open**.



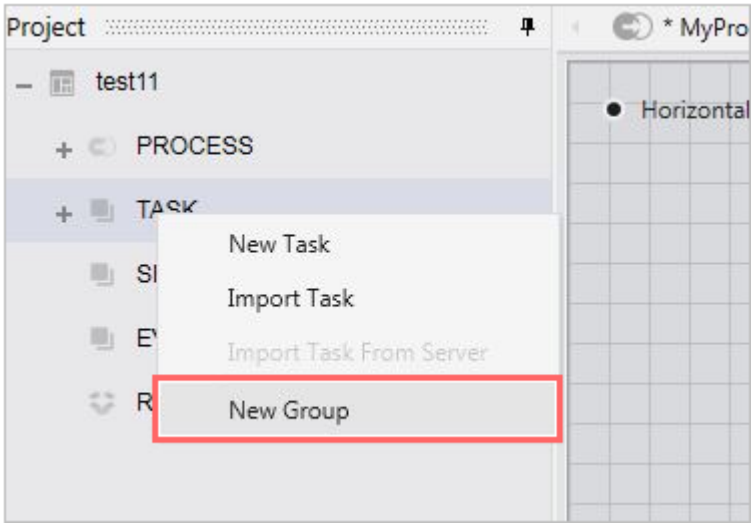
The resource is added.

Creating a new group

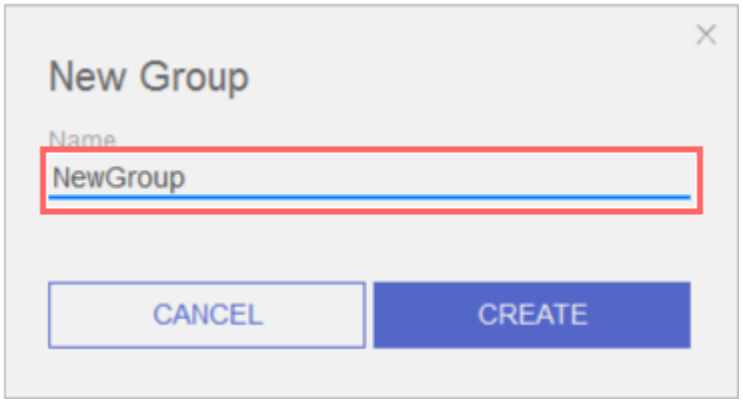
When the number of processes and tasks increase, you can group them for easy management. One group can include other various sub-groups.

You can create groups of processes or tasks using the same method. The example here is based on the creating a group for tasks.

1. On the project screen, right-click **TASK** and click **New Group**.



2. On the "New Group" pop-up window, enter the name of the new group to create and click **CREATE**.



A new group is created.

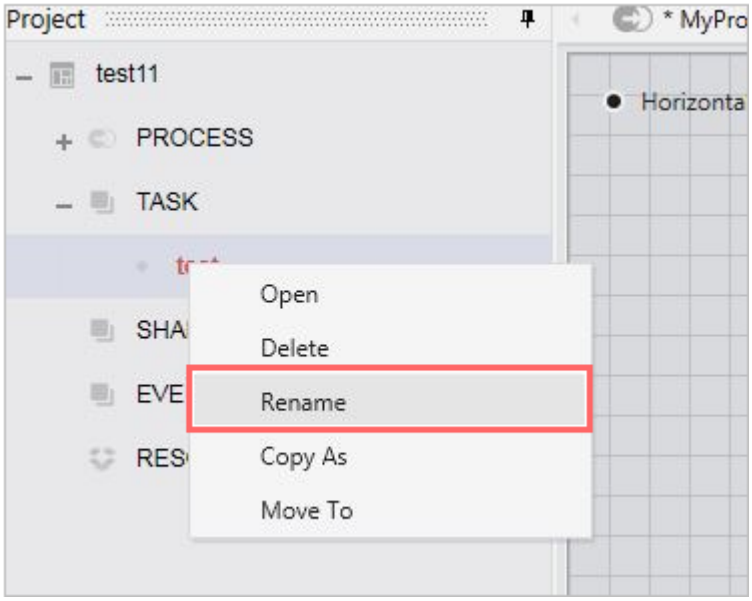
Editing a project

You can edit processes, tasks, events, and resources using the same method. This section uses the tasks as an example.

Changing task name

You can change the task name.

1. On the project screen, right-click on a task to change the name and click **Rename**.



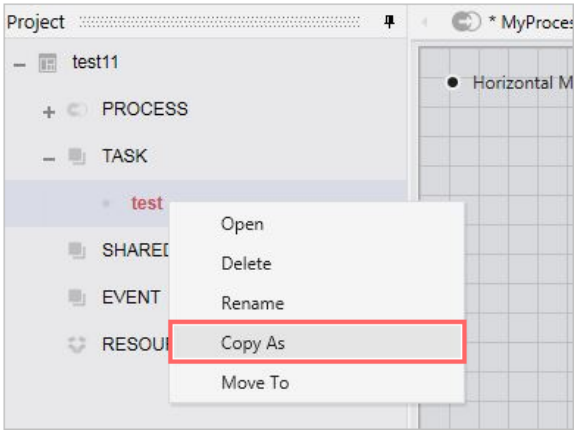
2. One the "Rename" pop-up window, enter a new name and click **OK**.

The task name is changed.

Copying task

You can copy the tasks of similar actions for use instead of creating a new task.

1. On the project screen, right-click on a task to copy and click **Copy As**.



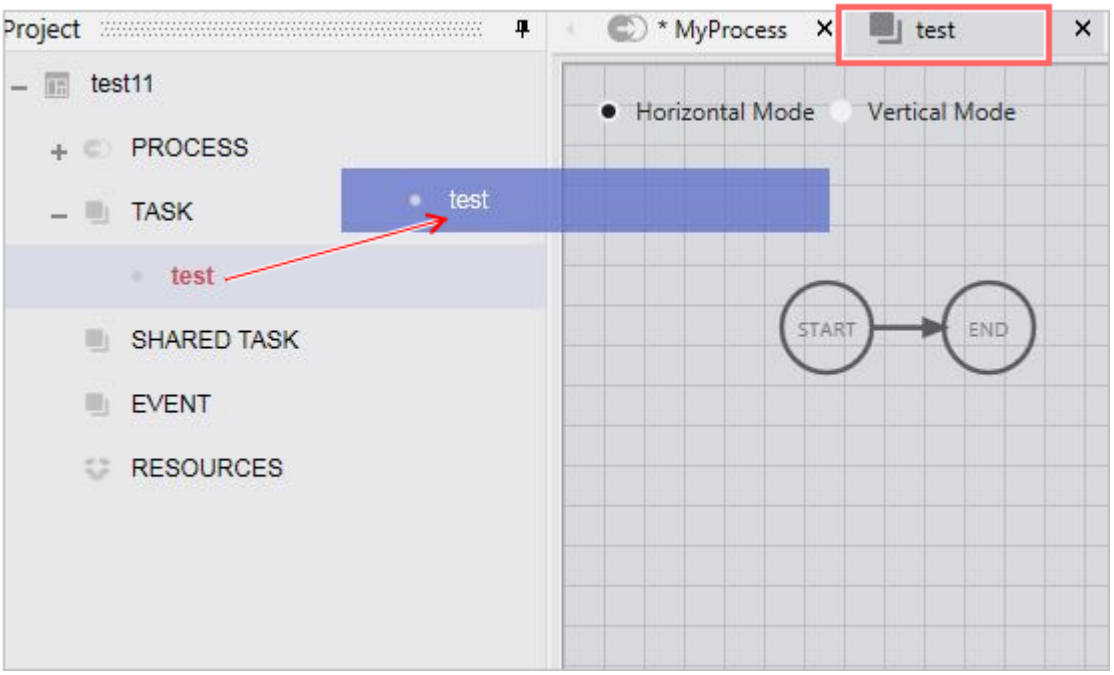
2. On the "Copy" pop-up window, enter a new name for the copied task and click **OK**.

The task is copied and will launch.

Moving task

You can move tasks within a group or outside of a group.

Select a task to move and drag to the desired position.

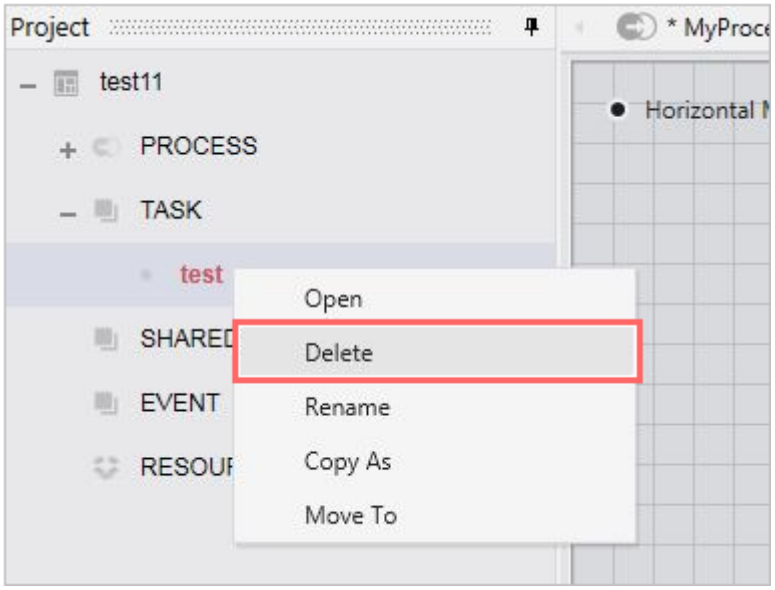


Make sure to close an open task before moving the position.

Deleting task

You can delete a task that is no longer used or not needed.

1. On the project screen, right-click on a task to delete and click **Delete**.

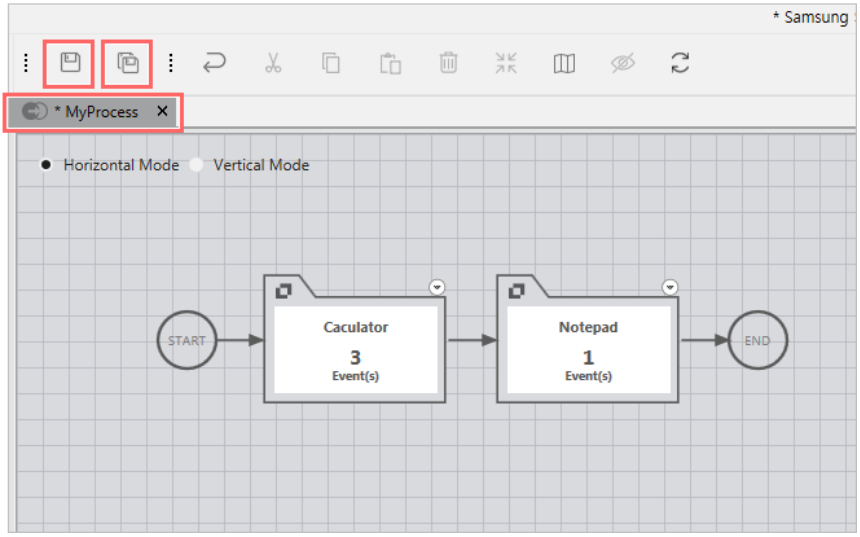


2. Click **OK** on the confirmation pop-up window.

The selected task is deleted.

3.3 Saving a project and process flow

When there are changes to the flowchart, an asterisk (*) is added to the name on the tabs. To save the currently activated flowchart, click the save ( ) icon. To save all open flowcharts, click .
You can also click > **Save** or **Save All** to save.

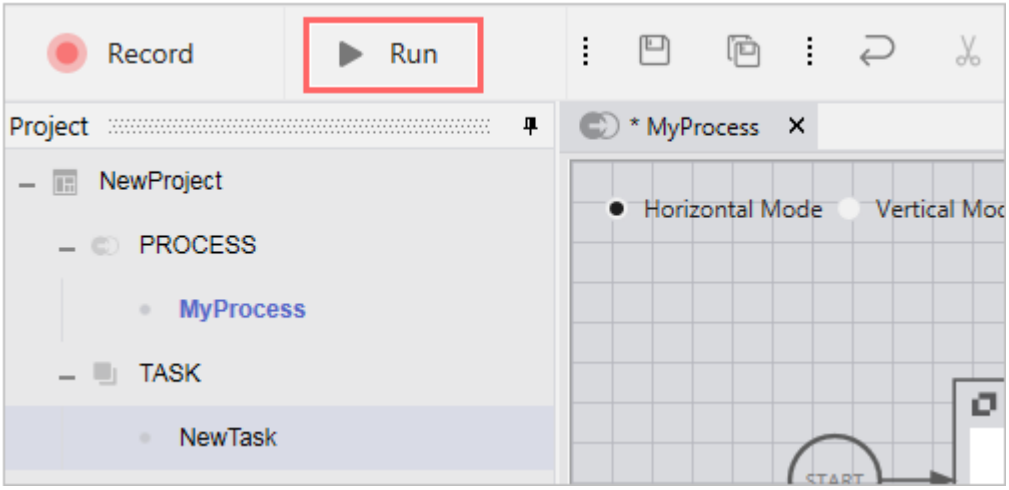


3.4 Running a project

To run an activated process or task, click **Run** on the toolbar screen.

Or you can click ☰ > **Run** > **Run**.

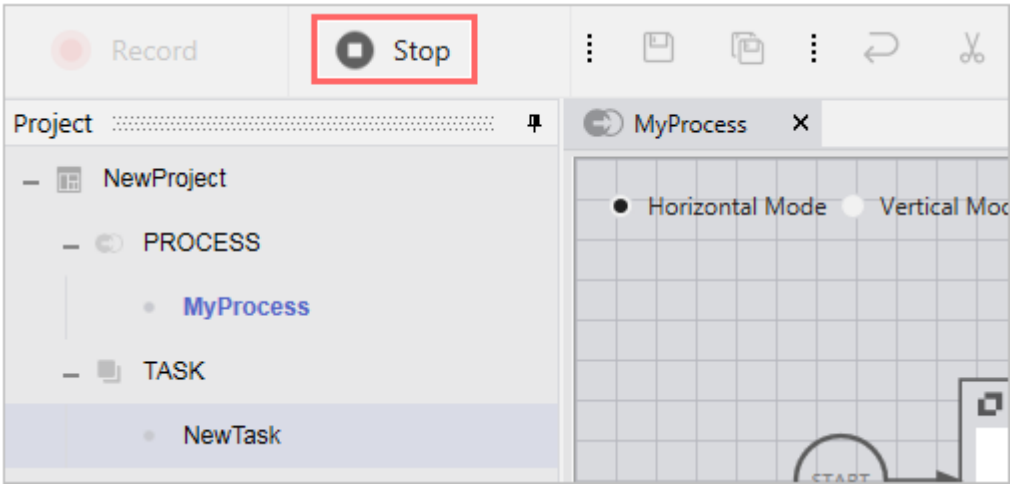
Any unsaved charts will be saved automatically.



If a process or task is performed for the first time, a pop-up will appear asking whether to minimize the Designer screen. If you click **YES**, the IPA Designer program will be hidden and will appear again once the process or task is complete.

To kill the execution, click **Stop** on the toolbar or press the Esc key on the keyboard.

Or you can also click ☰ > **Run** > **Stop**.



If you have selected **Track Execution** in ☰ > **Settings** > **Designer Setting**, the node in progress will be emphasized automatically.

3.5 Running a process flow

The process flow can be executed through the server after being published and cannot be executed directly from IPA Designer. For more information on connecting the server, see "6. Linking to server."

4.
Designing projects

The user can record an action using the recording feature or design a project by finding the library to use on the library screen and adding to the activity.

"Designing projects" explains the following topics:

- Designing with the recording feature
- Designing with the library

4.1 Designing with the recording feature

Click the Record button and create a user action as a scenario. You can record actions, such as mouse-click, right-click, double-click, and keyboard input actions.

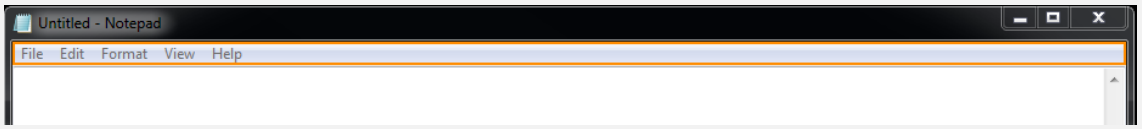
1. Click **Record** to start recording.

   Or you can click ☰ > **Record**.

   Once recording starts, the IPA Designer screen will be minimized.

• When you move the mouse over a target UI object while recording, a red border will appear.

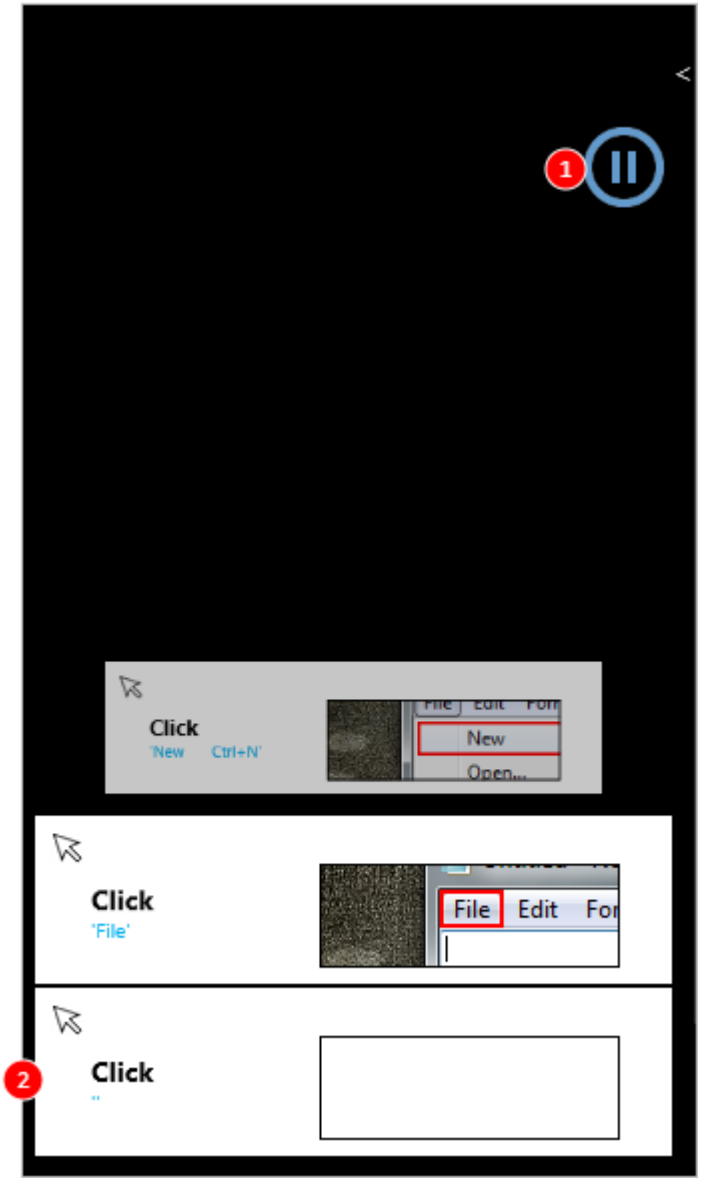
• When you move the mouse over a target UI object that has a sub-object, an orange border will appear.



It is recommended that you record the UI objects with red borders. A UI object with orange borders may not be found after execution as it has sub-objects.

2. Start an action to record from the beginning, such as opening a program or opening a file.

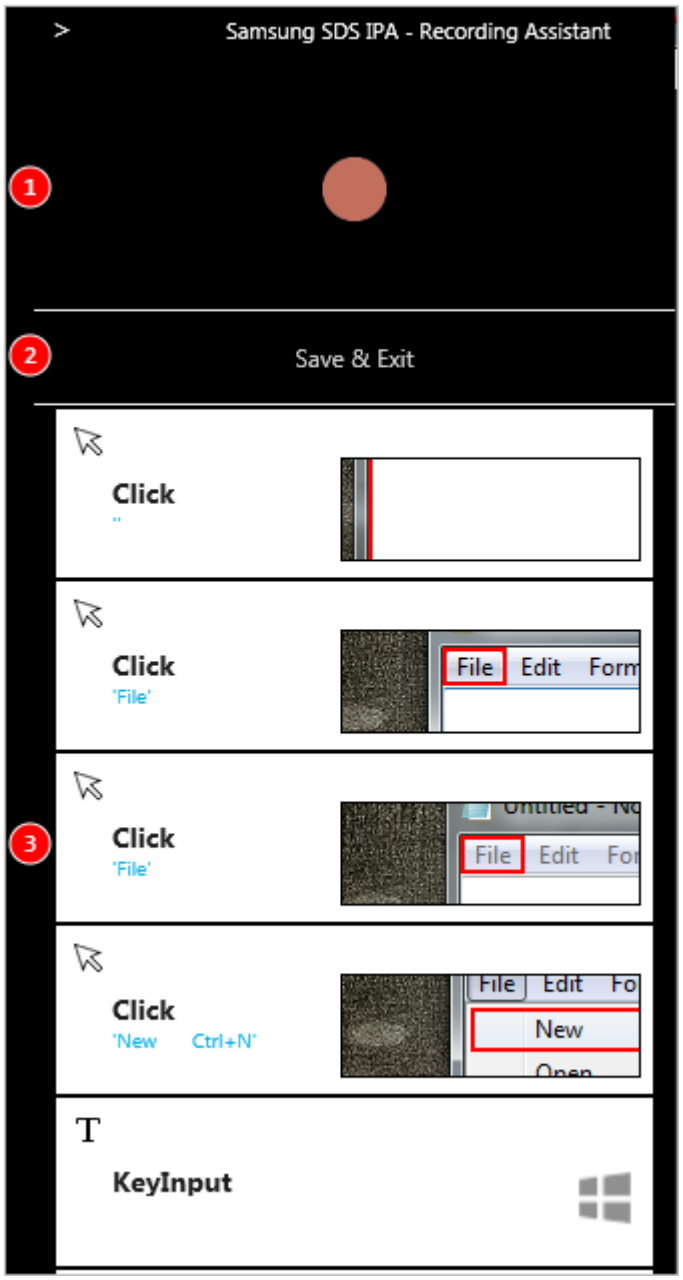
   Once the recording starts, the recording screen is displayed on the right-side of the screen like below.



No.	Description
❶	Pauses recording. Once recording stops, a recording screen opens and the recorded event list appears.
❷	Records actions on the mouse or keyboard as events and displays the notification window on the bottom-right corner of the screen.

3. Click pause ( ) if the action is complete or to pause recording.

   The recorded event list will appear.



No.	Description
❶	Starts recording again.
❷	Ends recording. Once recording ends, the IPA Designer screen is displayed.
❸	Shows the event list recorded by the user.

4. To finish recording, click **Save & Exit** on the event list screen.

The recorded action is added to the flow chart.

If you are using a 64-bit version software, make sure to install the 64-bit version of the driver. The 32-bit version of the driver is not compatible with the software.

4.2 Designing with the library

You can design a project with the library.

Adding an activity

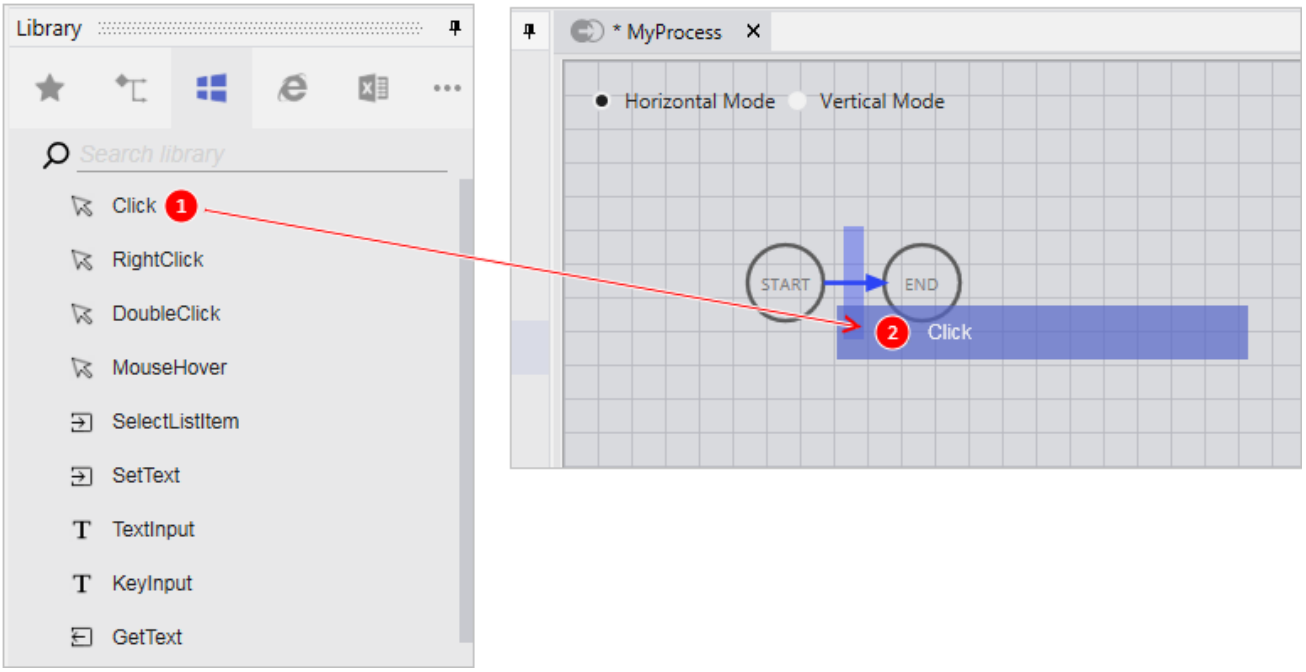
An activity refers to an action that create a flowchart.
You can add an activity by dragging a library you want to the flowchart on the design screen.

You can only add certain libraries of the **FlowControl** items, such as **If** or **For**, to the link. You cannot add it to the activity group or between activities.

Adding an activity to the link

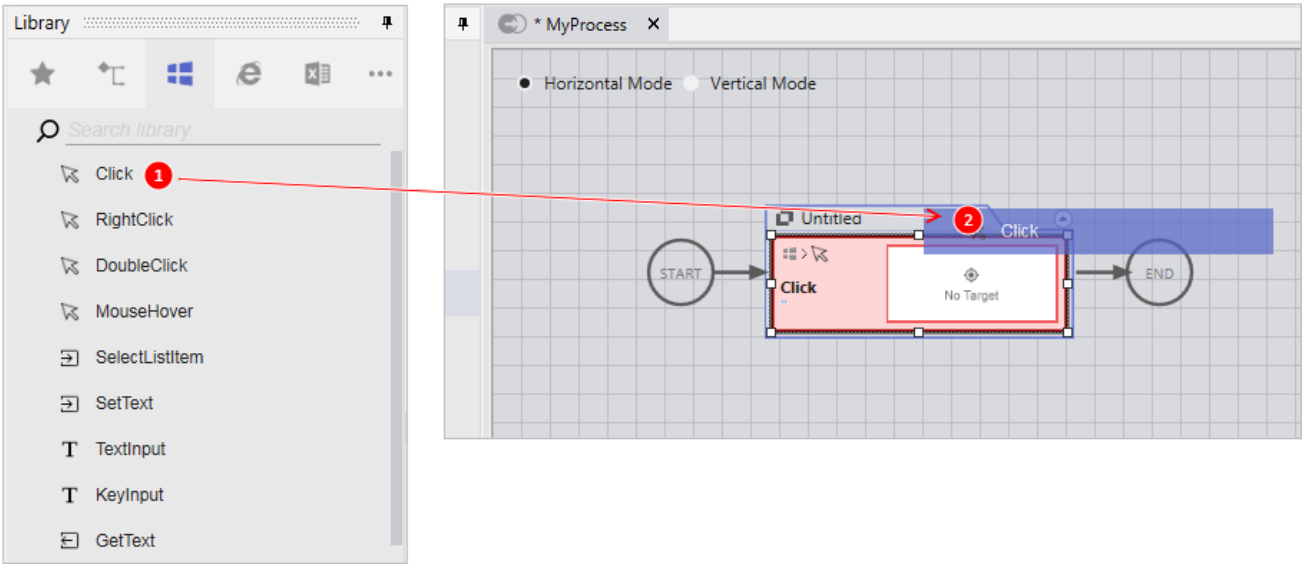
A link refers to the part that connects the activities. On the design screen, a link is displayed as an arrow in between the activities.
Select a library to add from the library screen and drag to the link you want to add it to. You can add

the activity by dropping the library when a blue rectangle appears.



Adding an activity to activity group

An activity group is a unit that combines many activities into one action.
Select a library to add from the library screen and drag it to the title area of the activity group you want to add it to. You can add the activity by dropping the library when the color of the activity group changes to blue. The activity will be added to the end of the activity group.



Adding an activity between groups

You can add an activity between activities in an activity group.
Select a library to add from the library screen and drag it between the activities of the activity group you want to add it to. You can add the activity by dropping the library in between activities when a blue rectangle appears.

Editing an activity

Deleting an activity

Select an activity to delete and click on the toolbar. Or you can click press the Delete key on the keyboard.

Copying an activity 1

You can copy an activity using the toolbar or the keyboard shortcut key.

1. Select an activity to copy.
2. Click on the toolbar or press Ctrl+C on the keyboard.
3. Select a location to paste.
4. Click on the toolbar or press Ctrl+V on the keyboard.

Copying an activity 2

You can copy an activity using the drag-and-drop action.

1. Select an activity to copy.
2. Hold down the right mouse button and drag the activity to the location to paste.
   Or you can hold the Ctrl key on the keyboard and move the cursor to the location to paste.

Moving an activity 1

You can move an activity using the toolbar or the keyboard shortcut key.

1. Select an activity to move.
2. Click on the toolbar or press Ctrl+X on the keyboard.
3. Select a location to paste.
4. Click on the toolbar or press Ctrl+V on the keyboard.

Moving an activity 2

You can move an activity using the drag-and-drop action.
Hold down the right mouse button and drag the activity to the location to paste.

Adding a task

You can add a task to another task just like an activity.
Select a task to add and drag to the link you want to add it to. You can add the task by dropping the task when a blue rectangle appears.

• A task can be added only to a link only and cannot be included in the activity group.
• A task cannot add itself as a task.
• Deleting, copying, and moving a task can be done using the same method as an activity.

Selecting multiple activities

You can select multiple activities to copy, move, or delete at once.
Drag the node to select using the mouse. Hold down the Ctrl key and click on the nodes for multiple selection.

• You can also make multiple selections by holding down the Shift key on the keyboard. However, you can select activities only in the same activity group.

> • If you press Ctrl+A key on the keyboard, you can select the entire activity group of the currently selected activity or select all nodes in the flowchart at once.

Inverting selection

You can invert the selection to select the activities that were unselected.
Click on the toolbar.

Deactivating

For automation, you can select to exclude specific activities.
Select more than one activity to exclude and click on the toolbar.

Folding an activity group

When the number of added activities increase, you can fold the activity group to see a condensed format. Once folded, the title and the number of activities in the group are displayed.
Click in the upper-right corner of the activity group.
To fold more than one activity group, select multiple activity groups and click on the toolbar.

Learn more about the library

A library is classified into app, web, Excel, SAP, image recognition, system, and flow control items depending on the target app or feature. Select an item to use from the library screen and enter the name of the library to search.

App and Web libraries

You can control app or web screens using the features provided by the **App** library and the **Web** library. These libraries identify the UI object on the screen using the UI automation technology provided by Windows and find the target UI specified by the user to perform the specified actions.

The features provided by the App and Web libraries are as follows:

- UI object recognition related features (Find, FindElement, RunAction, WaitAppear, WaitDisappear, IsExist, IsVisible, etc.)
- Mouse action features (MouseEvent, Click, RightClick, DoubleClick, MouseHover, SelectListItem, etc.)
- Keyboard action features (KeyboardEvent, SetText, TextInput, KeyInput, GetText, etc.)
- Execution and close features (OpenApp, ActivateApp, CloseApp, CloseAll, OpenWindow, CloseWindow, OpenBrowser, CloseBrowser, etc.)
- Email features (POP3,SMTP) (SendMail, ReceiveMail, DownloadMailAttachment, etc.)
- Other features (GetHierarchyInfo, etc.)
- WebDriver use features (Selenium, Headless) - beta

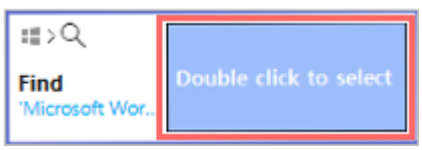
If the target app is developed using methods that are different to the ordinary methods, the **App** library may not operate as intended.

Find (Finding UI object)

Find searches for the UI object specified by the user.
This is default library of the UP related App library. For example, the **Click** library generates a mouse event of the **MouseEvent** library on the objects found using the **Find** library.

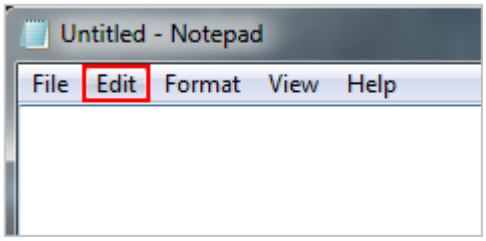
When searching for the target UI object, make sure to check that the UI object information recorded at the location of the mouse at the time of designating the target UI object matches the information of the target UI object. If the information does not match, you must find the target UI object using the method of moving down the upper path of the target UI object. If you are still unsuccessful in finding the target UI object using this method, collect all UI object information on the target app and compare it with the designated target UI object information to find the UI object.

1. On the node, double-click **No Target** > **Double click to select**.



2. Place the mouse cursor on the UI object.

A red border will appear.



3. To designate as the target a UI object, click the object once a red border appears.

The UI object is designated as the target UI object.

• When you move the mouse over a UI object that has a sub-object, an orange border will appear. To make a search for a target UI object without issues, choose a target UI object that does not have sub-UI objects. If it has sub-object, it may be difficult to find the target due to the difference in the recording time and the time of performing the automation. If you must designate a sub-UI object, keep in mind that the screen configuration may change during automation.

• If you must select the target UI object after pressing another UI object, or if you must designate the target UI object after moving to another screen, press the Ctrl key on the keyboard to pause the designation or press again to resume.

• When designating a target UI object, the object with a border displayed at the time of clicking the mouse becomes the search target. The position at the time of releasing the mouse button is recorded as relative coordinates based on the top left area of the target UI object. If the information of the target UI object is unclear, you can use this to designate a nearby object that can be specified and click on the mouse using the relative coordinates.

• If a pop-up window disappears or if the target app malfunctions due to the UI object border, you can hide the object borders by pressing the Alt Key on you keyboard once. Press again to see the borders again.

• If the border is displayed on a UI object that is different to the UI object with the mouse cursor, press the Shift key on the keyboard once to activate the **Search Around** option and try again. For more information, see the description for Search Around under Figure 4-1 Find properties screen.

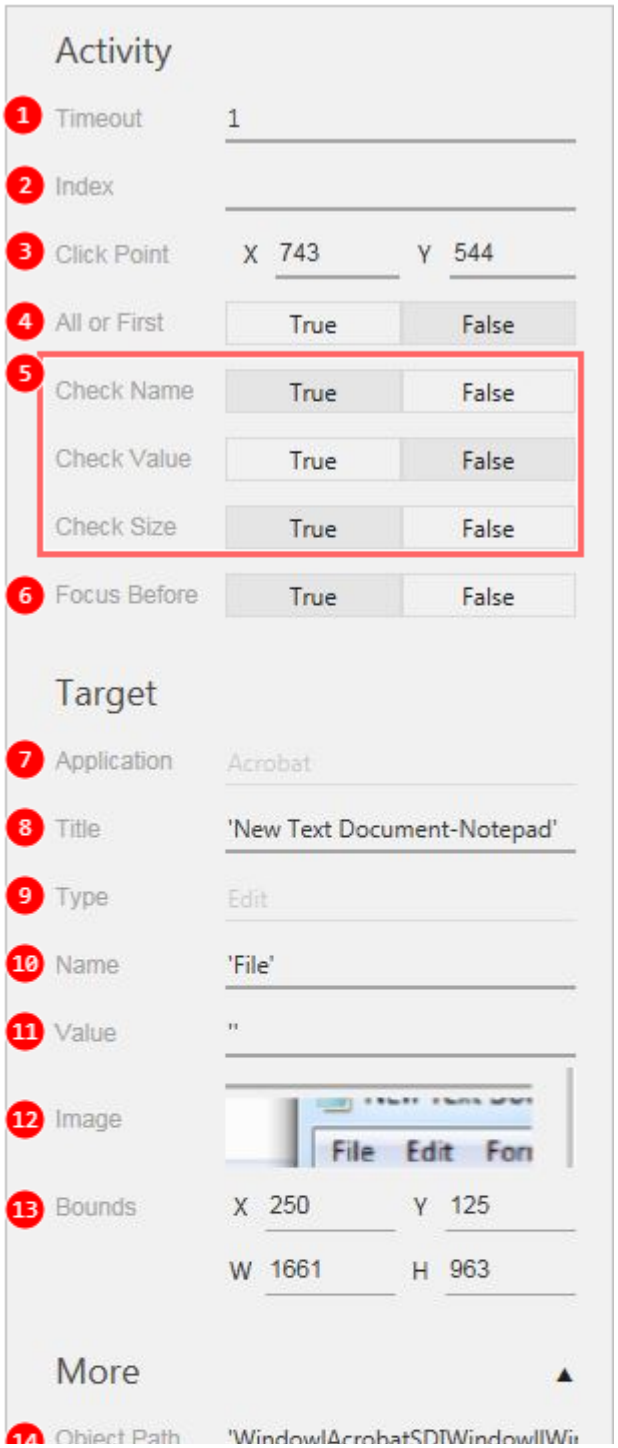
To check or change the target UI object, double-click the capture screen of the specified target UI object and click **Test** or **Reassign** from the "Find" pop-up window.

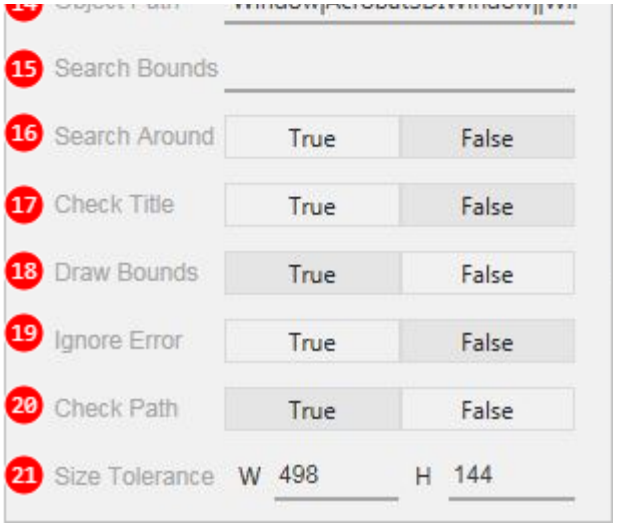
- **Test**: You can test for the designated target UI object. The red border will flicker for three seconds on the target UI object and will disappear again.

The test cannot be performed if a variable is entered in the object information on the properties screen. In this case, right-click the node and click **Run step** to execute this node only.

- **Reassign**: You can reassign the designated UI object.

Figure 4-1 **Find** properties screen



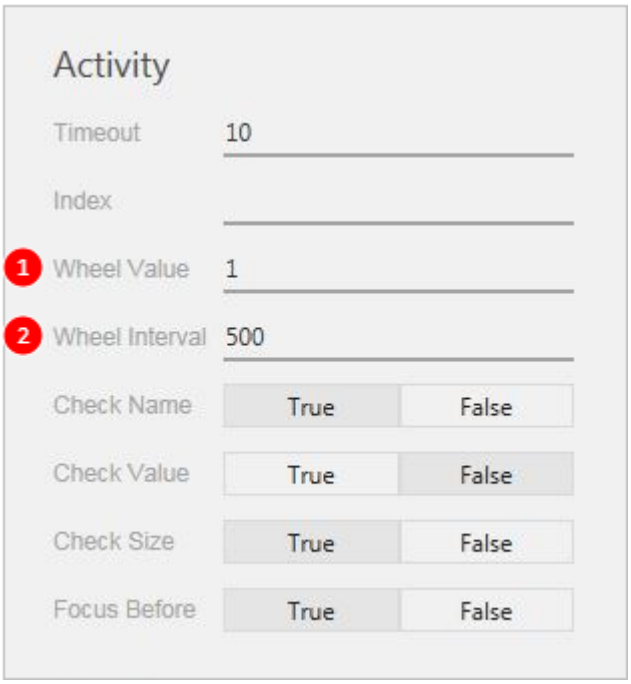


No.	Description
❶	Enter the duration of searching for the target UI object in seconds.
❷	Enter the index to identify the target UI object from the UI object list with identical information. If there is a transparent object around the target UI object that is invisible, or if the area of the object overlaps with a nearby UI object, the nearby object may be recognized instead of the target UI object. In this case, set to also scan the area of the surrounding object using the Search Around option.
❸	Enter the location to look for the UI object as coordinates based on the upper-left section of the UI object. This will be entered automatically if the object is designated directly with a mouse.
❹	Select **True** to get all target information that meet the condition, or select **False** to get only the information for the target discovered first. This will search the target UI object from the sub-level of the object entered in the Input field.
❺	Set whether to check the Name, Value, and Size when comparing UI objects.
❻	Set whether to focus on the UI object before generating an event. If the focus cannot be set on the target UI object, the focus is set to the main screen of the target app. If the target UI object is hidden by another app, the app is moved to the front. If there are multiple target apps running, the app at the front is scanned first. If the target UI object is not found, the app at the back is brought to the front for a scan.
❼	Shows the name of the target UI object app.
❽	Shows the title of the target UI object app. The title may be changed depending on the app status, so it is not essential for the title to match for UI object comparison. However, when there is a multiple detection of apps that are identical then the UI object that has the same title is selected.
❾	Shows the class type of the target UI object.
❿	Shows the name information of the target UI object. Mostly, a text string that can be represented on a screen will show but a value may not be displayed

No.	Description
	even if it exists depending on the app implementation method. For Name, Value, and Title, you can use a regular expression using the "\\:" keyword or enter a variable like this.SearchText. However, the node inspection feature is unavailable if a variable is entered.
⓫	Shows the value information of the target UI object. The text string that can be represented on a screen may be displayed on this item instead of the Name item depending on the app implementation method. Or, a value may not be displayed even if it exists.
⓬	Shows the image of the target UI object designated by the user.
⓭	Shows the position and size of the target UI object designated by the user as relative coordinates based on the top left section of the app.
⓮	Enter the path information of the target UI object designated by the user within single quotation marks. If the at (@) symbol is used, such as '@//Button', this will be considered as the XPath syntax. Then the entire UI structure of the designated target app will be generated in the XML format to search the target UI object information using the XPath syntax. For more information on the XML structure, see GetHierarchyInfo.
⓯	Select the specific area to search for, not the entire zone.
⓰	Set whether to check the nearby UI objects additionally for exact identification of the UI object.
⓱	Set whether to check the Title properties when comparing UI objects.
⓲	Set whether to show a bounding box when a UI object is found upon execution.
⓳	Set whether to continue a process or task even in the event of an error. Instead of writing a script using the **IsExist** library to click or ignore an error depending on whether an error exists, you can use the **Ignore Error** option.
⓴	Set whether to check the upper UI object instead of comparing the UI object on the screen when searching for the target UI object. In this case the search speed is improved but there may be a difficulty in locating the UI object if information such as Name, Value, Size, and Title, are not distinguished from other UI objects.
㉑	Enter the allowed margin of error when **Check Size** is set as **True**.

MouseScroll (Scrolling Mouse)

Figure 4-2 **MouseScroll** properties screen



No.	Description
❶	Enter the number of times the scroll will be taken down or up. (Process the positive number as scroll down and the negative number as scroll up.)
❷	Enter the scroll interval.

FindElement (Finding sub-UI object)

FindElement searches the target UI object from the sub-level of the object entered in the **Input** field.

RunAction (Generating events)

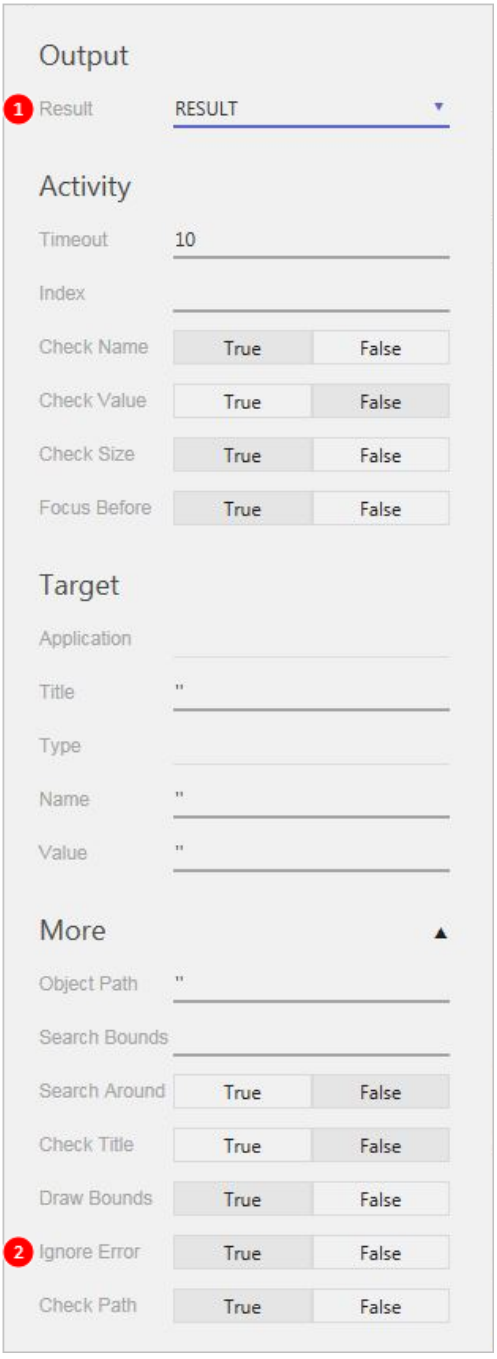
RunAction generates events, such as a click on the target UI object entered in the **Input** field, or searches nearby objects.

WaitAppear (Waiting until appeared)

WaitAppear waits until the target UI object appears on the screen.
This can be useful for changing the screen.

In case of having to wait for a long time, it is recommended that you add a Delay for the minimum waiting time before using the **WaitAppear** library. If **WaitAppear** library is used, the action may slow down or malfunction depending on the target app, because the UI object information of the target app is searched repeatedly.

Figure 4-3 **WaitAppear** properties screen

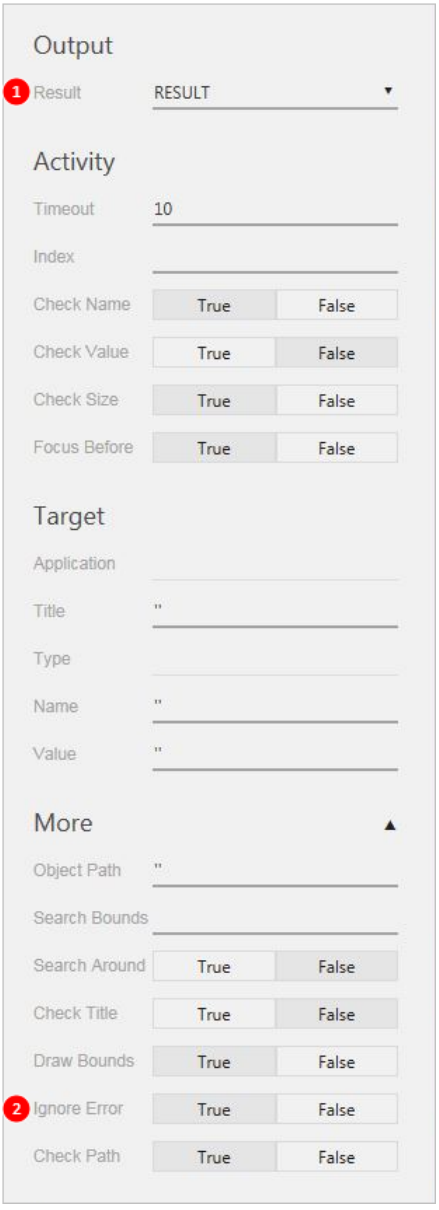


No.	Description
❶	Returns the discovery result of the target UI object as True or False.
❷	If set to **True**, False is returned in the **Result** if the target UI object is not discovered. If set to **False**, an exception is generated if the target UI object is not discovered.

WaitDisappear (Waiting until disappeared)

WaitDisappear waits until the target UI object disappears on the screen.
This can be useful for changing the screen.

Figure 4-4 **WaitDisappear** properties screen

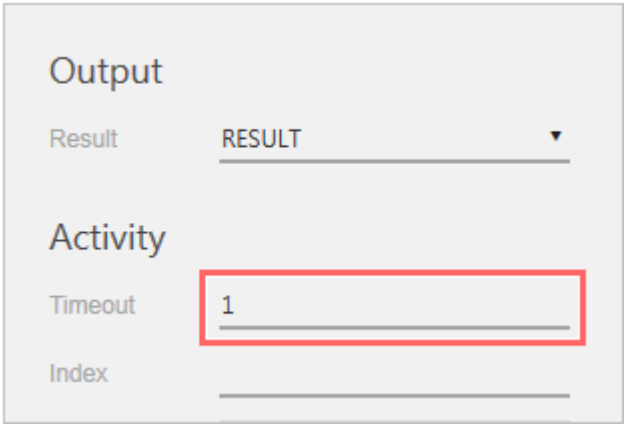


No.	Description
❶	Returns True if the target UI object disappears. Returns False if the target UI object does not disappear.
❷	If set to **True**, False is returned in the **Result** if the target UI object does not disappear. If set to **False**, an exception is generated if the target UI object does not disappear.

IsExist (Checking existence)

IsExist checks whether the target UI object is existent on the screen.

Figure 4-5 **IsExist** properties screen



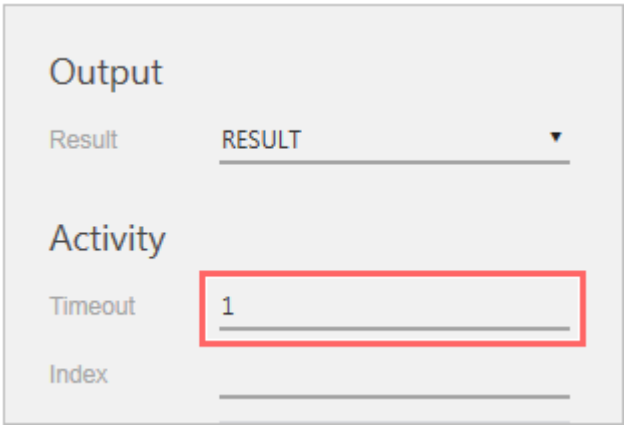
Returns True if the target UI object is found on the screen within the time entered in the **Timeout** field. Returns False if it is not found.

True is returned if the object information exists even if it not shown on the screen, such as if it is below the scroll or is hidden. In this case, the Bounds information of the object may be displayed as 0.

IsVisible (Checking visibility on screen)

IsVisible checks whether the target UI object is visible on the screen.

Figure 4-6 **IsVisible** properties screen



Check the Bounds information in addition to the **IsExist** library. Returns True if the Bounds information exists. True is also returned if Bounds information exists even if the object is hidden by another screen.

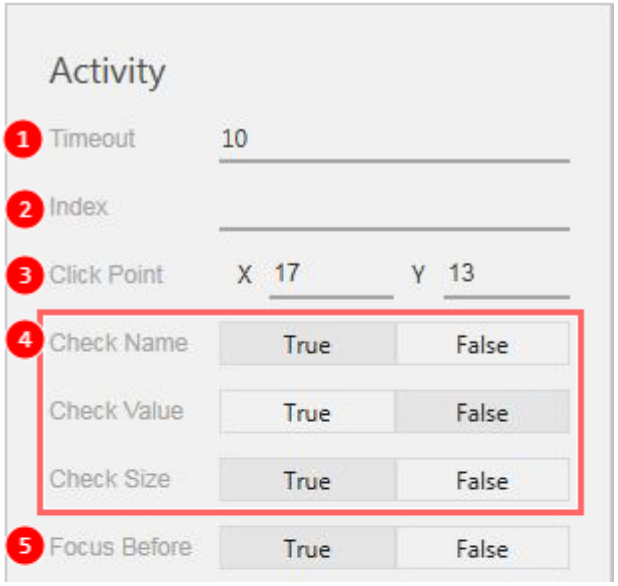
MouseEvent (Generating a mouse event)

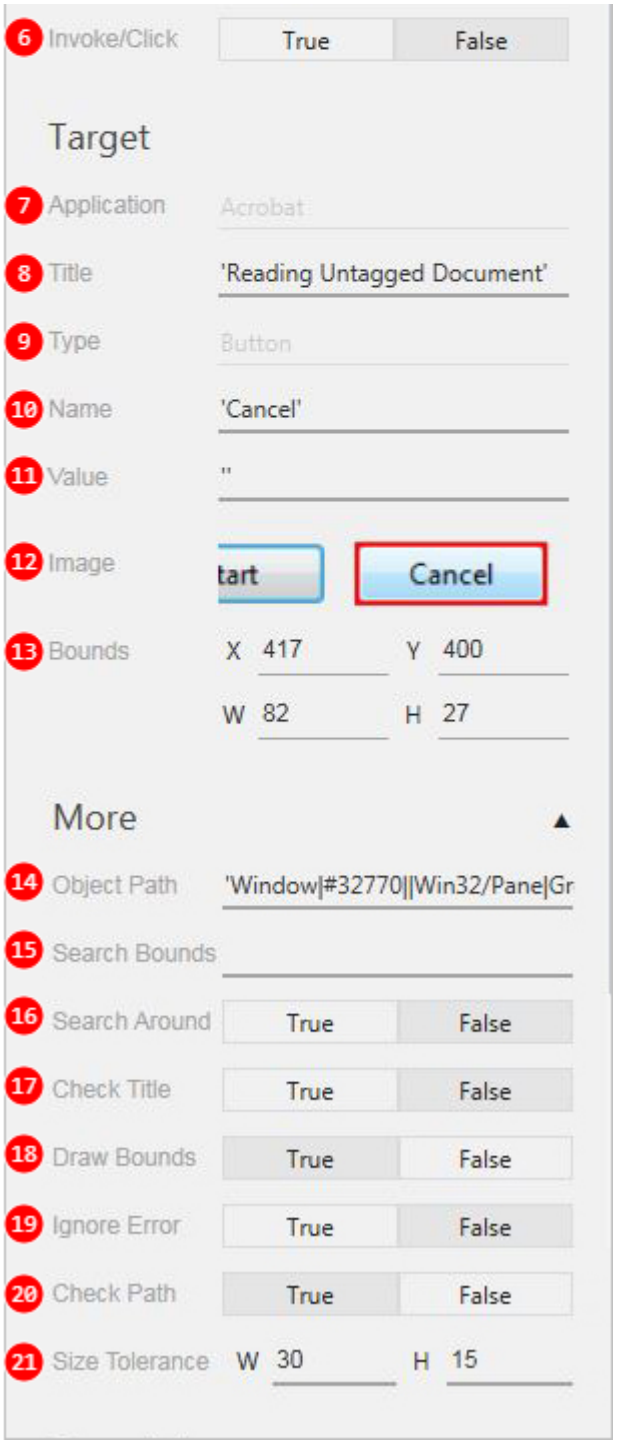
MouseEvent generates a mouse event such as a click on the designated location.

Click (Clicking a mouse)

Click left-clicks the target UI object.

Figure 4-7 **Click** properties screen





No.	Description
❶	Enter the duration of searching for the target UI object in seconds.
❷	Enter the index to identify the target UI object from the UI object list with identical information. If there is a transparent object around the target UI object that is invisible, or if the area of the object overlaps with a nearby UI object, the nearby object may be recognized instead of the target UI object. In this case, set to also scan the area of the surrounding object using the Search Around option.

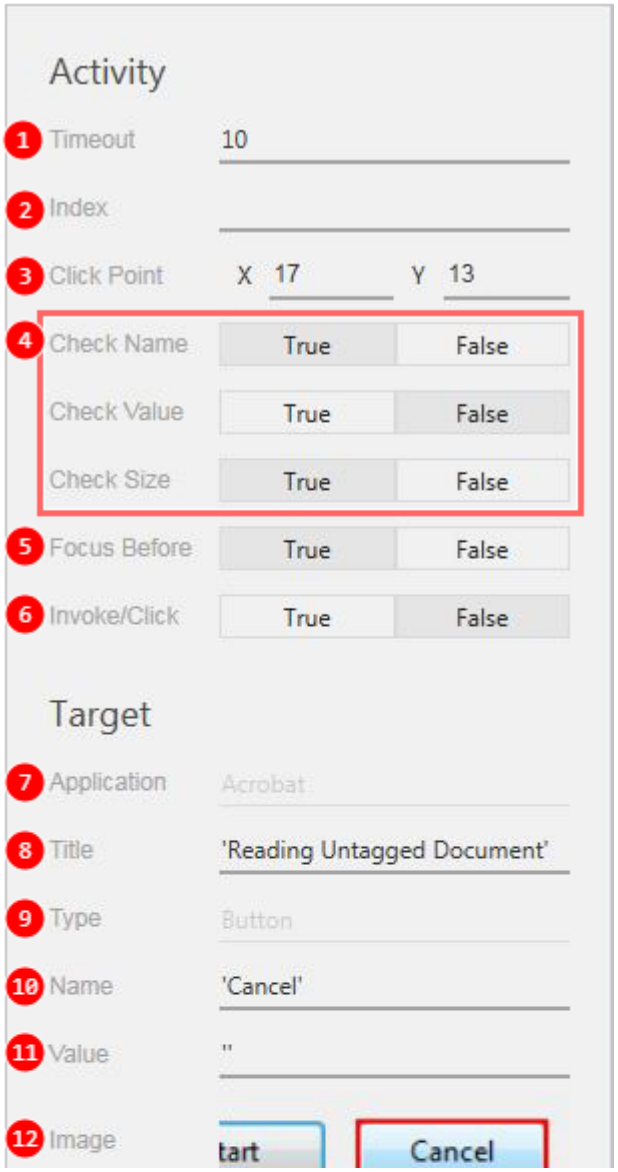
No.	Description
❸	Enter the location to click after finding the UI object as coordinates based on the top left section of the UI object. This will be entered automatically if the object is designated directly with a mouse.
❹	Set whether to check the Name, Value, and Size when comparing UI objects.
❺	Set whether to focus on the UI object before generating an event. If the focus cannot be set on the target UI object, the focus is set to the main screen of the target app. If the target UI object is hidden by another app, the app is moved to the front. If there are multiple target apps running, the app at the front is scanned first. If the target UI object is not found, the app at the back is brought to the front for a scan.
❻	Select whether to click or call the target UI object.
❼	Shows the name of the target UI object app.
❽	Shows the title of the target UI object app. The title may be changed depending on the app status, so it is not essential for the title to match for UI object comparison. However, when there is a multiple detection of apps that are identical then the UI object that has the same title is selected.
❾	Shows the class type of the target UI object.
❿	Shows the name information of the target UI object. Mostly, a text string that can be represented on a screen will show but a value may not be displayed even if it exists depending on the app implementation method. For Name, Value, and Title, you can use a regular expression using the "\\:" keyword or enter a variable. However, the node inspection feature is unavailable if a variable is entered.
⓫	Shows the value information of the target UI object. The text string that can be represented on a screen may be displayed on this item instead of the Name item depending on the app implementation method. Or, a value may not be displayed even if it exists.
⓬	Shows the image of the target UI object designated by the user.
⓭	Shows the position and size of the target UI object designated by the user as relative coordinates based on the top left section of the app.
⓮	Enter the path information of the target UI object designated by the user within single quotation marks.
⓯	Select the specific area to search for, not the entire zone.
⓰	Set whether to check the nearby UI objects additionally for exact identification of the UI object.
⓱	Set whether to check the Title properties when comparing UI objects.
⓲	Set whether to show a bounding box when a UI object is found upon execution.
⓳	Set whether to continue a process or task even in the event of an error. Instead of writing a script using the **IsExist** library to click or ignore an error depending on whether an error exists, you can use the **Ignore Error** option.

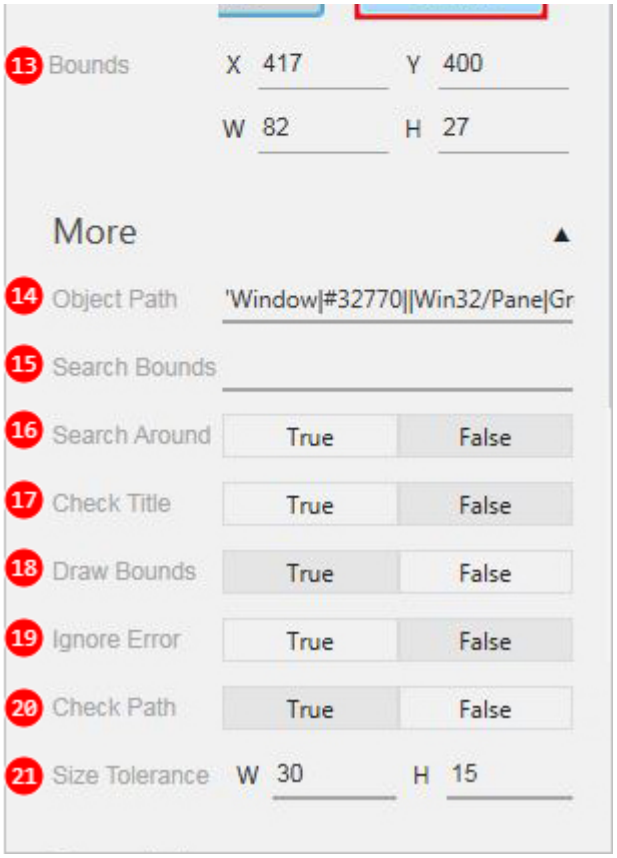
No.	Description
⓴	Set whether to check the upper UI object instead of comparing the UI object on the screen when searching for the target UI object. In this case, the search speed is improved, but there may be difficulty in locating the UI object if information, such as Name, Value, Size, and Title, are not distinguished from other UI objects.
㉑	Enter the allowed margin of error when **Check Size** is set as **True**.

RightClick (Clicking the right mouse button)

RightClick right-clicks on the target UI object.

Figure 4-8 **RightClick** properties screen





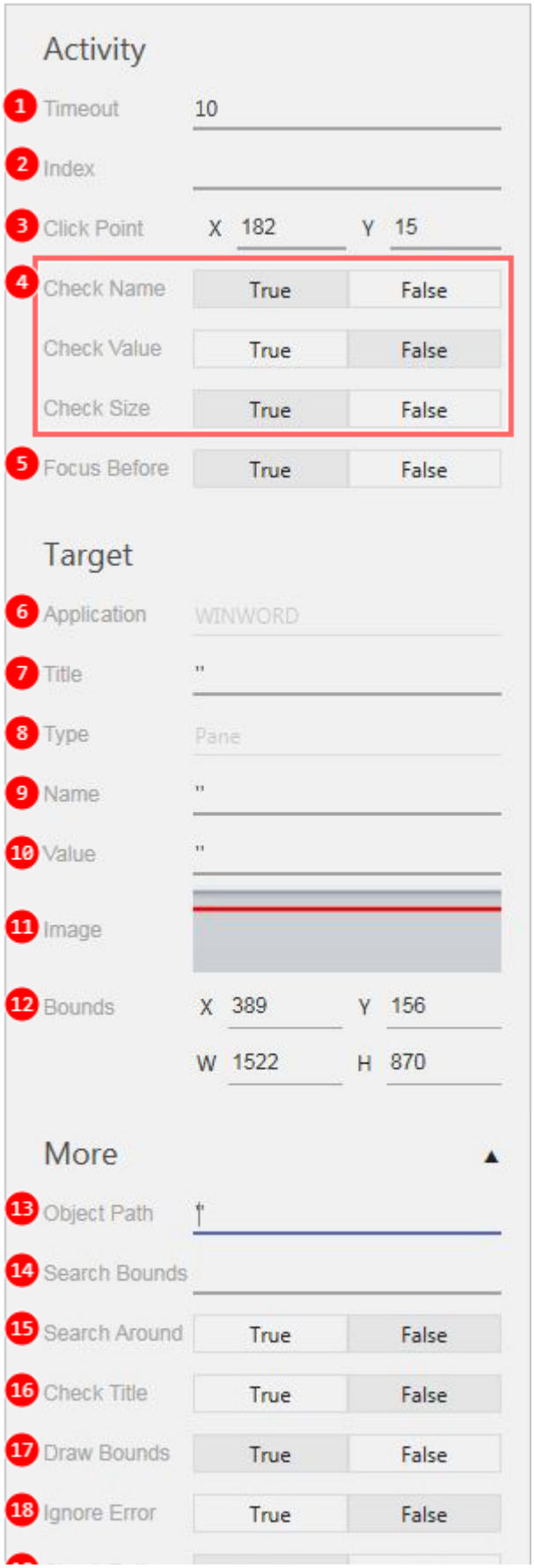
No.	Description
❶	Enter the duration of searching for the target UI object in seconds.
❷	Enter the index to identify the target UI object from the UI object list with identical information. If there is a transparent object around the target UI object that is invisible, or if the area of the object overlaps with a nearby UI object, the nearby object may be recognized instead of the target UI object. In this case, set to also scan the area of the surrounding object using the Search Around option.
❸	Enter the location to look for the UI object as coordinates based on the upper-left section of the UI object. This will be entered automatically if the object is designated directly with a mouse.
❹	Set whether to check the Name, Value, and Size when comparing UI objects.
❺	Set whether to focus on the UI object before generating an event. If the focus cannot be set on the target UI object, the focus is set to the main screen of the target app. If the target UI object is hidden by another app, the app is moved to the front. If there are multiple target apps running, the app at the front is scanned first. If the target UI object is not found, the app at the back is brought to the front for a scan.
❻	Select whether to click or call the target UI object.
❼	Shows the name of the target UI object app.
❽	Shows the title of the target UI object app.

No.	Description
	The title may be changed depending on the app status, so it is not essential for the title to match for UI object comparison. However, when there is a multiple detection of apps that are identical then the UI object that has the same title is selected.
❾	Shows the class type of the target UI object.
❿	Shows the name information of the target UI object. Mostly, a text string that can be represented on a screen will show but a value may not be displayed even if it exists depending on the app implementation method. For Name, Value, and Title, you can use a regular expression using the "\\:" keyword or enter a variable like this.SearchText. However, the node inspection feature is unavailable if a variable is entered.
⓫	Shows the value information of the target UI object. The text string that can be represented on a screen may be displayed on this item instead of the Name item depending on the app implementation method. Or, a value may not be displayed even if it exists.
⓬	Shows the image of the target UI object designated by the user.
⓭	Shows the position and size of the target UI object designated by the user as relative coordinates based on the top left section of the app.
⓮	Enter the path information of the target UI object designated by the user within single quotation marks.
⓯	Select the specific area to search for, not the entire zone.
⓰	Set whether to check the nearby UI objects additionally for exact identification of the UI object.
⓱	Set whether to check the Title properties when comparing UI objects.
⓲	Set whether to show a bounding box when a UI object is found upon execution.
⓳	Set whether to continue a process or task even in the event of an error. Instead of writing a script using the **IsExist** library to click or ignore an error depending on whether an error exists, you can use the **Ignore Error** option.
⓴	Set whether to check the upper UI object instead of comparing the UI object on the screen when searching for the target UI object. In this case the search speed is improved but there may be a difficulty in locating the UI object if information such as Name, Value, Size, and Title, are not distinguished from other UI objects.
㉑	Enter the allowed margin of error when **Check Size** is set as **True**.

DoubleClick (Double clicking)

DoubleClick double-clicks the target UI object.

Figure 4-9 **DoubleClick** properties screen


Activity
1 Timeout 10
2 Index
3 Click Point X 182 Y 15
4 Check Name True False
Check Value True False
Check Size True False
5 Focus Before True False
Target
6 Application WINWORD
7 Title ''
8 Type Pane
9 Name ''
10 Value ''
11 Image
12 Bounds X 389 Y 156
W 1522 H 870
More
13 Object Path
14 Search Bounds
15 Search Around True False
16 Check Title True False
17 Draw Bounds True False
18 Ignore Error True False

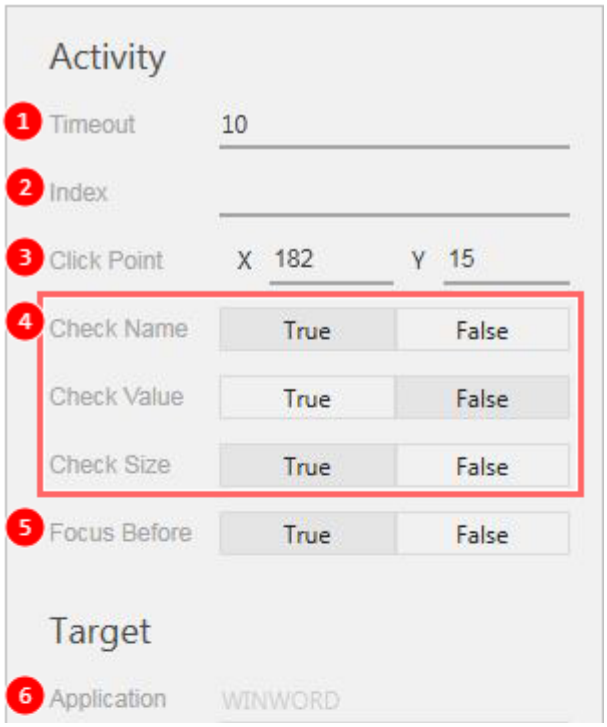
No.	Description
❶	Enter the duration of searching for the target UI object in seconds.
❷	Enter the index to identify the target UI object from the UI object list with identical information. If there is a transparent object around the target UI object that is invisible, or if the area of the object overlaps with a nearby UI object, the nearby object may be recognized instead of the target UI object. In this case, set to also scan the area of the surrounding object using the Search Around option.
❸	Enter the location to look for the UI object as coordinates based on the upper-left section of the UI object. This will be entered automatically if the object is designated directly with a mouse.
❹	Set whether to check the Name, Value, and Size when comparing UI objects.
❺	Set whether to focus on the UI object before generating an event. If the focus cannot be set on the target UI object, the focus is set to the main screen of the target app. If the target UI object is hidden by another app, the app is moved to the front. If there are multiple target apps running, the app at the front is scanned first. If the target UI object is not found, the app at the back is brought to the front for a scan.
❻	Shows the name of the target UI object app.
❼	Shows the title of the target UI object app. The title may be changed depending on the app status, so it is not essential for the title to match for UI object comparison. However, when there is a multiple detection of apps that are identical then the UI object that has the same title is selected.
❽	Shows the class type of the target UI object.
❾	Shows the name information of the target UI object. Mostly, a text string that can be represented on a screen will show but a value may not be displayed even if it exists depending on the app implementation method. For Name, Value, and Title, you can use a regular expression using the "\\:" keyword or enter a variable like this.SearchText. However, the node inspection feature is unavailable if a variable is entered.
❿	Shows the value information of the target UI object. The text string that can be represented on a screen may be displayed on this item instead of the Name item depending on the app implementation method. Or, a value may not be displayed even if it exists.
⓫	Shows the image of the target UI object designated by the user.
⓬	Shows the position and size of the target UI object designated by the user as relative coordinates based on the top left section of the app.

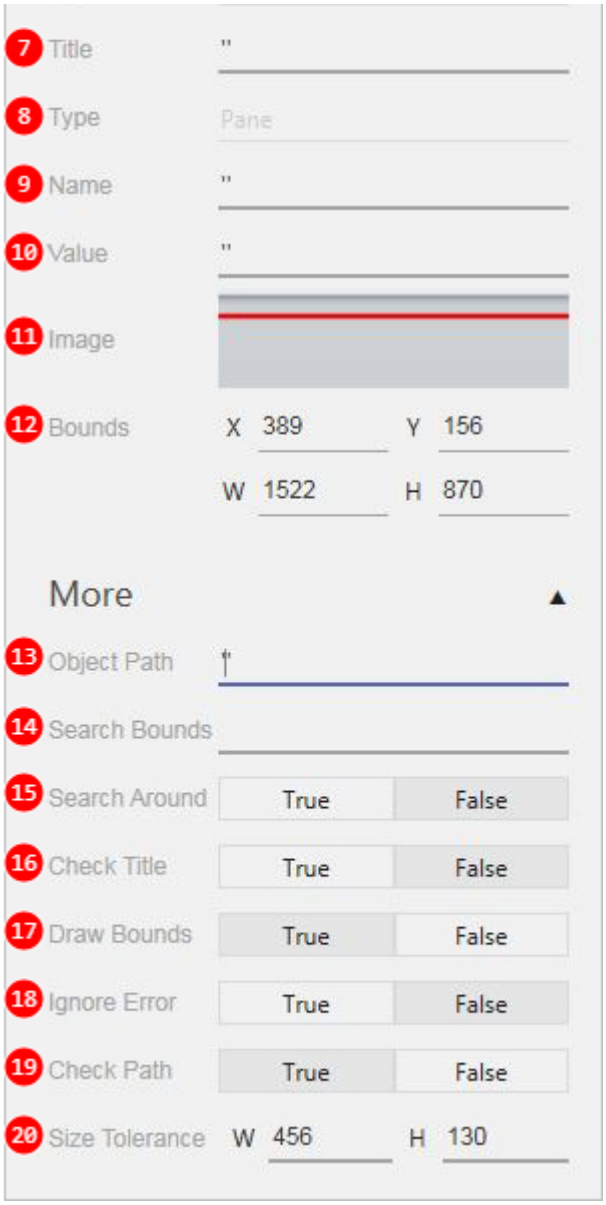
No.	Description
⓭	Enter the path information of the target UI object designated by the user within single quotation marks.
⓮	Select the specific area to search for, not the entire zone.
⓯	Set whether to check the nearby UI objects additionally for exact identification of the UI object.
⓰	Set whether to check the Title properties when comparing UI objects.
⓱	Set whether to show a bounding box when a UI object is found upon execution.
⓲	Set whether to continue a process or task even in the event of an error. Instead of writing a script using the **IsExist** library to click or ignore an error depending on whether an error exists, you can use the **Ignore Error** option.
⓳	Set whether to check the upper UI object instead of comparing the UI object on the screen when searching for the target UI object. In this case the search speed is improved but there may be a difficulty in locating the UI object if information such as Name, Value, Size, and Title, are not distinguished from other UI objects.
⓴	Enter the allowed margin of error when **Check Size** is set as **True**.

MouseHover (Hovering the mouse cursor)

MouseHover hovers the mouse cursor over the target UI object.

Figure 4-10 **MouseHover** properties screen





No.	Description
❶	Enter the duration of searching for the target UI object in seconds.
❷	Enter the index to identify the target UI object from the UI object list with identical information. If there is a transparent object around the target UI object that is invisible, or if the area of the object overlaps with a nearby UI object, the nearby object may be recognized instead of the target UI object. In this case, set to also scan the area of the surrounding object using the Search Around option.
❸	Enter the location to look for the UI object as coordinates based on the upper-left section of the UI object. This will be entered automatically if the object is designated directly with a mouse.
❹	Set whether to check the Name, Value, and Size when comparing UI objects.
❺	Set whether to focus on the UI object before generating an event.

No.	Description
	If the focus cannot be set on the target UI object, the focus is set to the main screen of the target app. If the target UI object is hidden by another app, the app is moved to the front. If there are multiple target apps running, the app at the front is scanned first. If the target UI object is not found, the app at the back is brought to the front for a scan.
❻	Shows the name of the target UI object app.
❼	Shows the title of the target UI object app. The title may be changed depending on the app status, so it is not essential for the title to match for UI object comparison. However, when there is a multiple detection of apps that are identical then the UI object that has the same title is selected.
❽	Shows the class type of the target UI object.
❾	Shows the name information of the target UI object. Mostly, a text string that can be represented on a screen will show but a value may not be displayed even if it exists depending on the app implementation method. For Name, Value, and Title, you can use a regular expression using the "\\:" keyword or enter a variable like this.SearchText. However, the node inspection feature is unavailable if a variable is entered.
❿	Shows the value information of the target UI object. The text string that can be represented on a screen may be displayed on this item instead of the Name item depending on the app implementation method. Or, a value may not be displayed even if it exists.
⓫	Shows the image of the target UI object designated by the user.
⓬	Shows the position and size of the target UI object designated by the user as relative coordinates based on the top left section of the app.
⓭	Enter the path information of the target UI object designated by the user within single quotation marks.
⓮	Select the specific area to search for, not the entire zone.
⓯	Set whether to check the nearby UI objects additionally for exact identification of the UI object.
⓰	Set whether to check the Title properties when comparing UI objects.
⓱	Set whether to show a bounding box when a UI object is found upon execution.
⓲	Set whether to continue a process or task even in the event of an error. Instead of writing a script using the **IsExist** library to click or ignore an error depending on whether an error exists, you can use the **Ignore Error** option.
⓳	Set whether to check the upper UI object instead of comparing the UI object on the screen when searching for the target UI object. In this case the search speed is improved but there may be a difficulty in locating the UI object if information such as Name, Value, Size, and Title, are not distinguished from other UI objects.
⓴	Enter the allowed margin of error when **Check Size** is set as **True**.

SelectListItem (Selecting a specific item from the list)

SelectListItem selects the target UI object by finding it from a combo box or list.
This can be used only if the target UI object is a ListItem type.

> Note that SelectListItem may not operate or the information may not change depending on the implementation method of the target app even if the target UI object is a ListItem type.

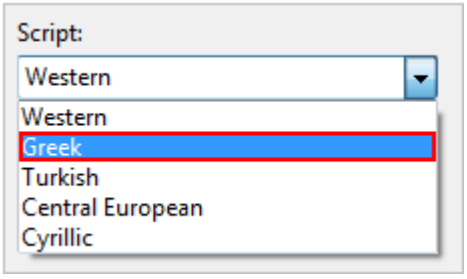
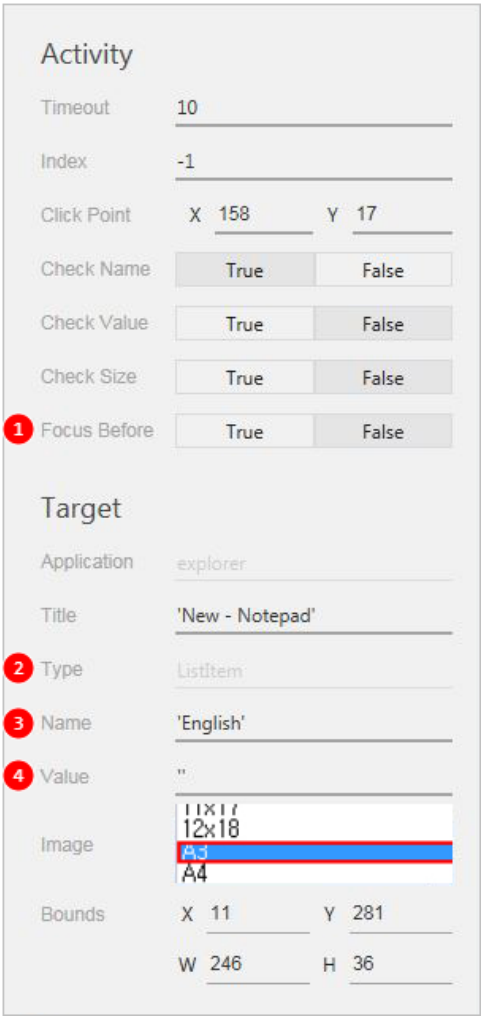


Figure 4-11 **SelectListItem** properties screen



No.	Description
❶	Set whether to focus on the UI object before generating an event. The default is set to **False** because an open combo box may be closed if focus set.
❷	Shows the type of the target UI object. The **SelectListItem** library can be used only for the **ListItem** type.
❸	Enter text of the item to select within single quotation marks.
❹	Depending on the target UI object, the text may be displayed in the **Value** field and not in the **Name**. If so, enter the text in the **Value** field.

GetListItems (Getting all items included on the list)

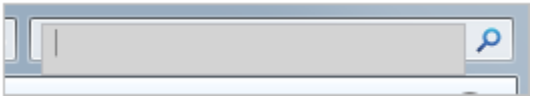
GetListItems returns the items on the same path as the target UI object from the list, tree, or menu.

KeyboardEvent (Generating a keyboard event)

KeyboardEvent generates a keyboard event regardless of the target UI object.

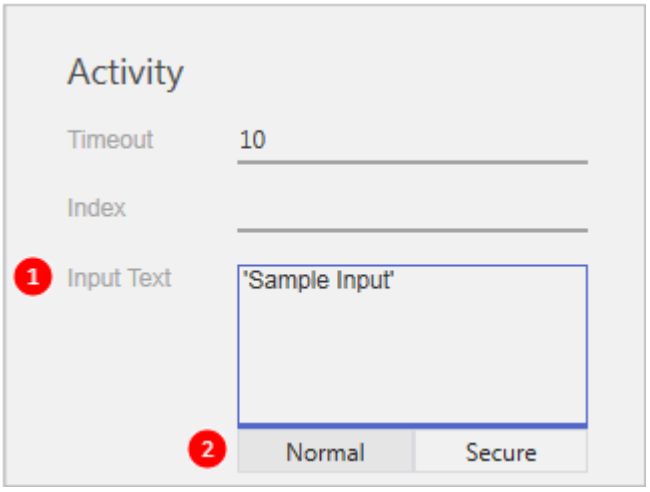
SetText (Inputting text)

SetText inputs text in the target UI object. This can be used only if the target UI object supports ValuePattern, like a text box, and is not read-only.



Select the target UI object and enter the text when the text box appears. You can enter multiple lines by pressing Shift+Enter on the keyboard.
Once you finish, press Enter to end.

Figure 4-12 **SetText** properties screen



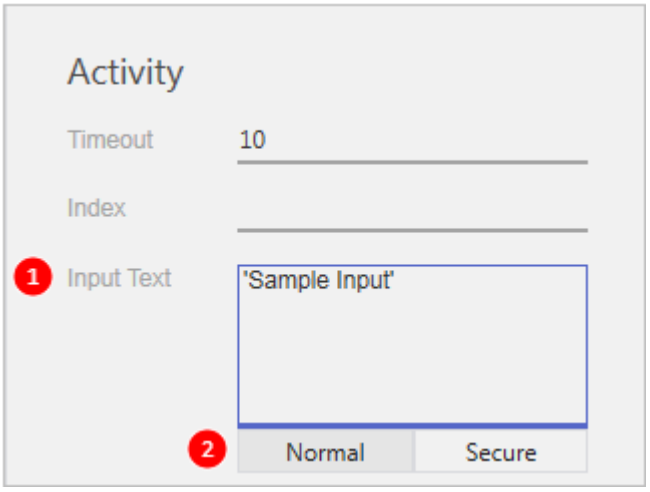
No.	Description
❶	Enter the text to input within single quotation marks. Write special characters such as line break or quotation marks using a backward slash "?" in accordance with to the JavaScript coding rule.
❷	Set whether to encrypt the entered text. Even if **Secure** is selected, the original text appears in the **Input Text** field and is encrypted only when saved to a file.

In case of using a project containing an encrypted string by copying to another PC, you cannot see the original text from another PC.

TextInput (Inputting text)

TextInput inputs text after finding the target UI object.
It uses the text input API of Windows to send the text to target UI objects one by one. You can also input text using the keyboard event API depending on the option settings.

Figure 4-13 **TextInput** properties screen



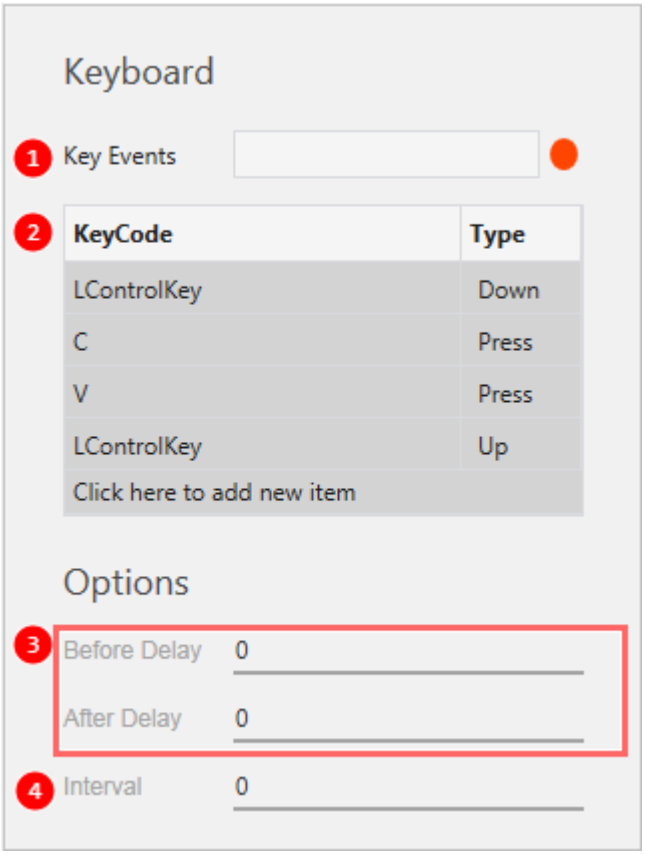
No.	Description
❶	Enter the text to input within single quotation marks. Write special characters, such as line break or quotation marks using a backward slash "?" in accordance with to the JavaScript coding rule.
❷	Set whether to encrypt the entered text. Even if **Secure** is selected, the original text appears in the **Input Text** field and is encrypted only when saved to a file.

• Text unsupported by Unicode may not be input.
• You can generate a keyboard event in between strings using the {{KeyCode}} type of keyword. For example, {{Enter}} is an action of pressing the Enter key and you can indicate the up and down function of the key such as {{+Enter}}or {{-Enter}}. For more information on the keyword to enter in between {{}}, go to the link below:
https://docs.microsoft.com/ko-kr/dotnet/api/system.windows.forms.keys?redirectedfrom=MSDN&view=netframework-4.7.2

KeyInput (Pressing the keyboard button)

KeyInput generates a keyboard event such as pressing or releasing keyboard buttons. This can be used even in case of pressing multiple buttons at the same time, such as Ctrl+Shift+C.
A keyboard event is generated on the UI object where the focus is currently set without designating the target UI object.

Figure 4-14 **KeyInput** properties screen

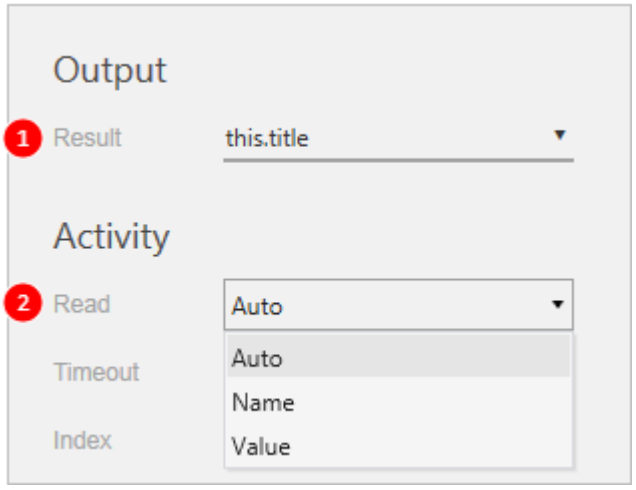


No.	Description
❶	Click ● to record a keyboard event.
❷	Shows the recorded keyboard events. If you record after selecting an event item, a new event is added above the selected item. If you press the Delete key on the keyboard after selecting an event, the event is deleted. Click **Click here to add new item**, then enter the values for **KeyCode** and **Type** to manually add a keyboard event. In the **Type** item, you can enter Press, Down, Up, Text, Variable, and Delay.
❸	Set the waiting time before and after generating a keyboard event.
❹	Set the interval of pressing the button. Depending on the target app, a shortcut key may not operate if the interval is too short.

GetText (Retrieving text)

GetText retrieves the text of the target UI object.
You can select the item to retrieve the text from either Name or Value.

Figure 4-15 **GetText** properties screen



No.	Description
❶	Specify the variable to allocate the text read from the target UI object.
❷	Select the text type to read the from the target UI object. If you select **Auto**, **Value** is read as a priority if the target UI object is Edit or Document type, and **Name** is read as priority for other cases.

GetBounds (Getting bound information)

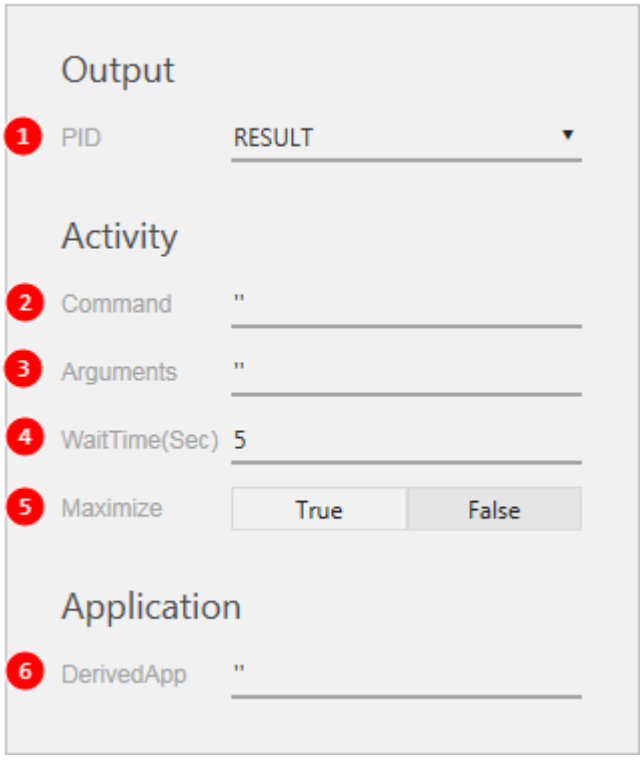
GetBounds gets the bound information of the target UI object. The result will be returned as strings in the form of x, y, width, and height. The returned strings can be used by converting them into numerical arrays using the eval function in Assign or ExecuteScript.

OpenApp (Launching app)

OpenApp launch a specified app. Once the app is launched, the process ID (PID) of the launched app is returned.

You can use the PID returned by **OpenApp** in **ActivateApp** and **CloseApp**.

Figure 4-16 **OpenApp** properties screen



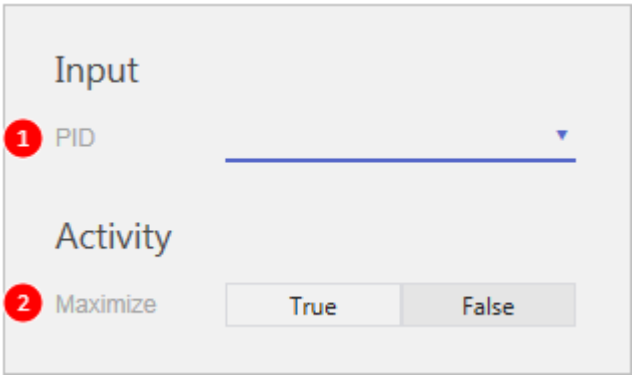
No.	Description
❶	Specify the variable to save the PID of the launched app.
❷	Enter the execution command of the app to launch. You may have to enter the entire path on the execution file depending on the app.
❸	Enter an execution argument that is required for launching the app. If there is no execution argument to send, enter an empty string (‘’).
❹	Set the duration of time to check whether executed properly after launching the app.
❺	Set whether to change into a maximized state after the app is launched.
❻	Specify the name of the app to be launched if an app that is not entered in the **Command** field is launched. If you enter null or false, execution status of the target app will not be checked.

PID of the app on the screen may be different from the PID returned after launching the app according to the characteristics of the target app such as being launched though a launcher app or not being launched delicately.
For example, if calc.exe is executed in Windows 10, calc.exe will close and calculator.exe is launched to appear on the screen. In this case, PID of the calc.exe specified in **OpenApp** will be returned on the **PID** field of **OpenApp** instead of the calculator.exe that appeared on the screen.

ActivateApp (Activating the app)

ActivateApp activates the app with the selected PID by moving the app to the very front.

Figure 4-17 **ActivateApp** properties screen

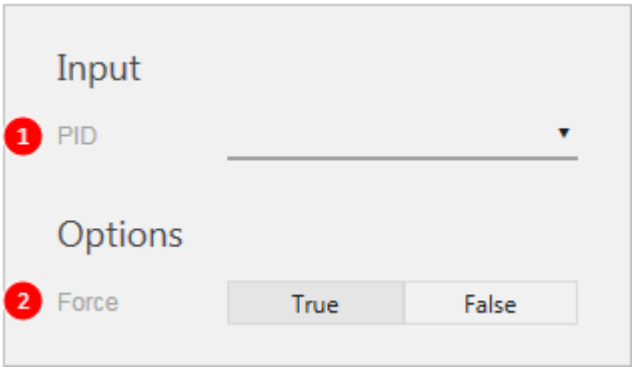


No.	Description
❶	Select PID of the app to activate.
❷	Select whether to maximize the app to activate.

CloseApp (Closing the app)

CloseApp closes the app with the selected PID.

Figure 4-18 **CloseApp** properties screen



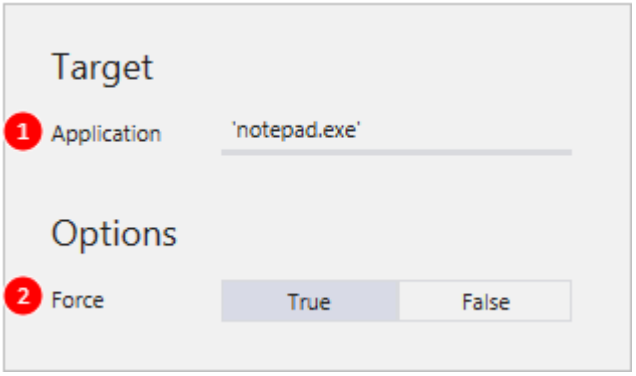
No.	Description
❶	Select PID of the app to close.
❷	If set to **True**, the app will be force closed regardless of the status, if it is not closed after the close command is sent.

No.	Description
	If set to **False**, the close command is sent to app only once. The app may or may not close depending on the app status. For example, when a command is sent to a Memo that is edited, the app will not be closed because a new pop-up is displayed asking if you want to save the edited file.

CloseAll (Closing the app)

CloseAll closes an app or all apps with the entered name.

Figure 4-19 **CloseAll** properties screen



No.	Description
❶	Enter the name of the app to close within the single quotation marks. To close all apps, enter an empty string (‘’).
❷	If set to **True**, the app will be force closed regardless of the status, if it is not closed after the close command is sent. If set to **False**, the close command is sent to app only once. The app may or may not close depending on the app status. For example, when a command is sent to a Memo that is edited, the app will not be closed because a new pop-up is displayed asking if you want to save the edited file.

OpenWindow (Opening a window)

OpenWindow opens the app, checks that the target UI object appears, and maximizes the app. You can utilize this for opening an app or launcher type which cannot be executed using the **OpenApp** library.

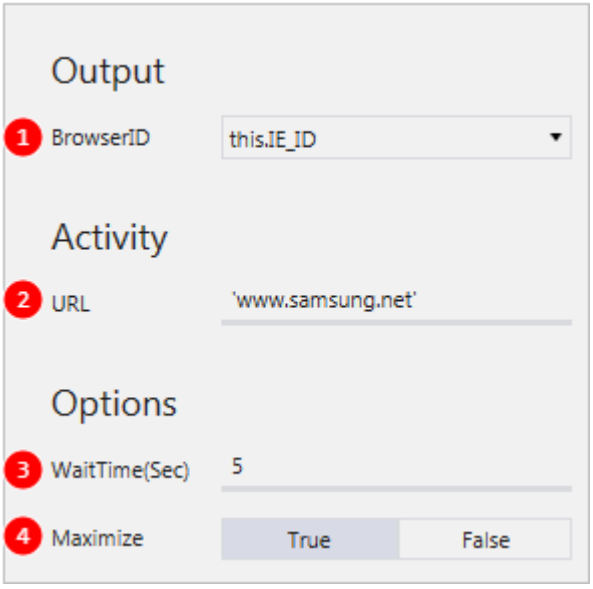
CloseWindow (Closing a window)

CloseWindow closes the window of the app to which the target UI object belongs to.

OpenBrowser (Opening a browser)

OpenBrowser opens Internet Explorer. Enter a URL to connect to.
Once executed, it returns the ID list of the browser. You can use the returned ID list in the **CloseBrowser**.

Figure 4-20 **OpenBrowser** properties screen



No.	Description
❶	Specify the variable to save ID of the launched browser.
❷	Enter the URL to connect to after the browser is launched within single quotation marks. To open an empty browser, enter an empty string ('').
❸	Set the duration of the waiting time until the browser is launched and appears on the screen.
❹	Set whether to maximize the browser window.

• A browser has one ID in most cases, but may have many IDs in some cases, such as if pop-up windows are opened.
• Unlike the PID of **OpenApp**, you cannot use the **CloseApp** because it is the ID of an app screen.

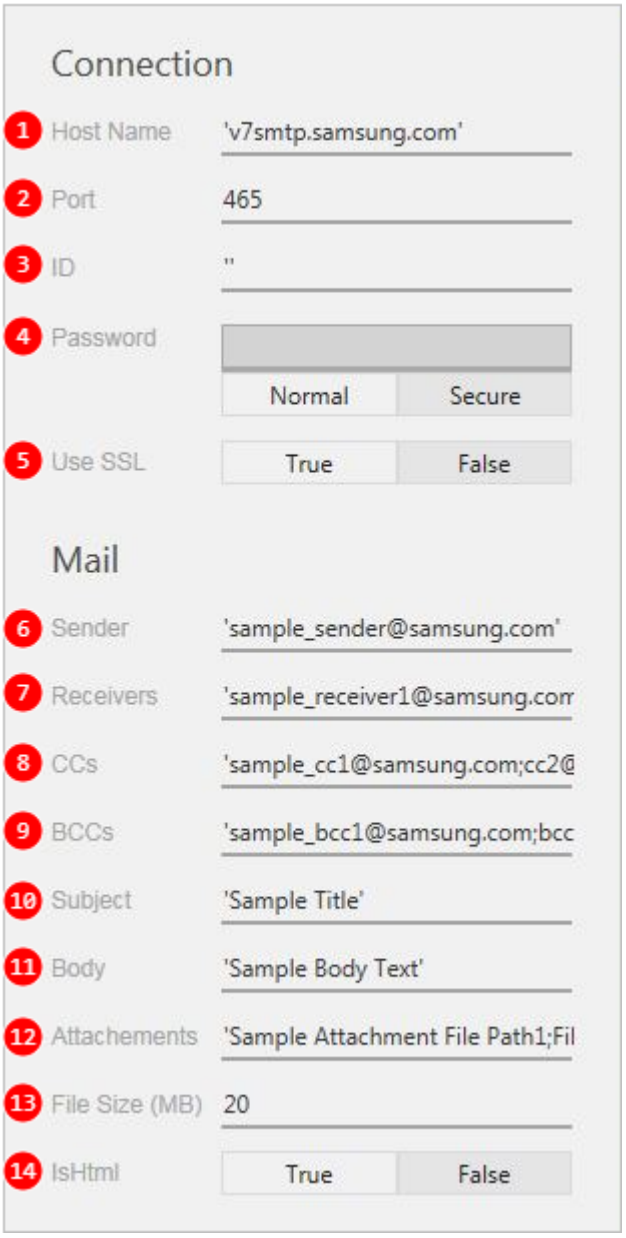
CloseBrowser (Closing a browser)

CloseBrowser closes the browser with the selected ID. You must save the ID list returned by the **OpenBrowser** to use this feature.

SendMail (Sending email)

SendMail sends an email.

Figure 4-21 **SendMail** properties screen

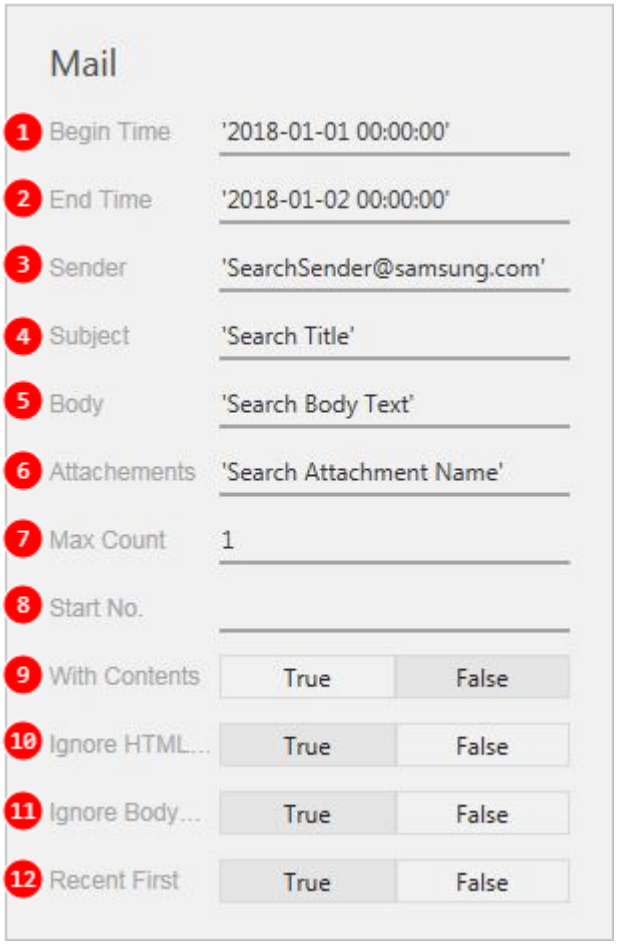


No.	Description
❶	Enter the host name of the email server (SMTP).
❷	Enter the port of the email server.
❸	Enter the ID of the sender.
❹	Enter the password of the sender.
❺	Set whether to use SSL.
❻	Enter the email address of the sender.
❼	Enter the mail address of the recipient or recipients. (Note: Multiple receivers can be designated by separating them with a semicolon, and this applies for both CCs and BCCs.)
❽	Enter the email addresses of carbon copy recipients.
❾	Enter the email addresses of blind carbon copy recipients.
❿	Enter the title of the email.
⓫	Enter the body of the email.
⓬	Enter the path of the attachment.
⓭	Set the size limit of the attachment. (Note: When sending an email, an error is returned when the attachment size greater than the limit.)
⓮	Set whether the email uses the html format.

ReceiveMail (Receiving email)

ReceiveMail receives an email and allocates to a variable.

Figure 4-22 **ReceiveMail** properties screen



No.	Description
❶	Set to receive emails sent after the entered time.
❷	Set to receive emails sent before the entered time. (If empty, checks until the current time.)
❸	Enter the email address to receive the emails sent from the email address.
❹	Enter the text to receive emails including the text in the subject.
❺	Enter the text to receive emails including the text in the body content. If entered, the search may be slowed.
❻	Enter the text to receive emails including the text in the attachments title. If entered, the search may be slowed.
❼	Enter the maximum number of received value. (All is set to 0.)
❽	Enter the receipt start number.
❾	Set whether to include the body and attachments list among the returned information.
❿	Set whether to ignore the HTML tags in the body content.
⓫	Set whether to ignore the attachments in the body among the returned information. (Set to **False** to receive the attachments in the body.)

No.	Description
⓬	Set whether to receive recent emails first.

DownloadMailAttachment (Saving email attachment)

DownloadMailAttachment saves the email attachment.

GetHierarchyInfo (Converting into XML format)

GetHierarchyInfo returns the information of the sub-object in XML format based on the specified object or app. You can use this for purposes, such as handling the information on the screen all at once.

The returned object information is expressed as an XMLDocument object. For more information about the XMLDocument, go to the link below:
https://docs.microsoft.com/ko-kr/dotnet/api/system.xml.xmldocument?view=netframework-4.7.2

Excel library

You can control Excel documents using the features provided by the **Excel** library.
Using the library, you can create new documents or open existing documents to edit and save.
The features provided by the Excel library are as follows:

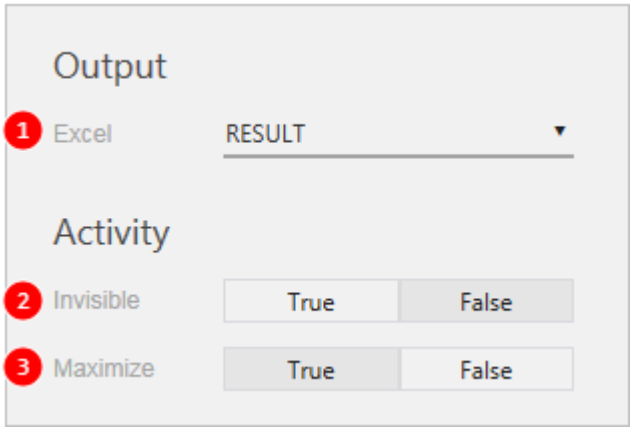
- Creating new Excel documents and worksheets (NewExcel, CreateExcel, CreateWorkSheet)
- Opening, activating, saving, closing, splitting and, getting Excel documents (OpenExcel, ActivateExcel, SaveExcel, CloseExcel, SplitExcel, GetActiveExcel, GetExcel, GetSheetName, GetAllSheetNames)
- Selecting worksheets and areas (SelectWorkSheet, SelectRange, SelectRow, SelectCol)
- Getting, reading, writing, and creating areas (GetActiveRange, GetDataRange, GetRowCount, GetColCount, SplitRange, MergeRange, ReadRange, WriteRange, MakeRange, ClearRange)
- Applying, releasing, and aligning filters (SetFilter, SetColorFilter, ReleaseFilter, SortData)
- Searching, writing, and replacing strings (SearchString, WriteString, ReplaceString)

- Copying, pasting, and deleting (CopyPaste, CopyWorkSheet, CopyRange, PasteRange, CopyWorkSheet, DeleteWorkSheet)
- Executing macro (ExecuteMacro)
- Adding, selecting, and deleting rows and columns (AddRow, AddCol, DeleteRow, DeleteCol)
- Importing and entering formula (ReadCellFormula, WriteCellFormula)

NewExcel (Opening new Excel document)

NewExcel opens a new Excel document.

Figure 4-23 **NewExcel** properties screen



No.	Description
❶	Specify the variable to allocate to the Excel document.
❷	Set whether to deactivate the Excel document.
❸	Set whether to maximize the Excel window.

CreateExcel (Creating a new Excel document)

CreateExcel creates a new document.

Figure 4-24 **CreateExcel** properties screen

No.	Description
❶	Enter the path to create the new Excel document and the filename.
❷	Set whether to overwrite the file if same filename already exists.

CreateWorkSheet (Adding a new worksheet)

CreateWorkSheet adds a new worksheet to the Excel document.

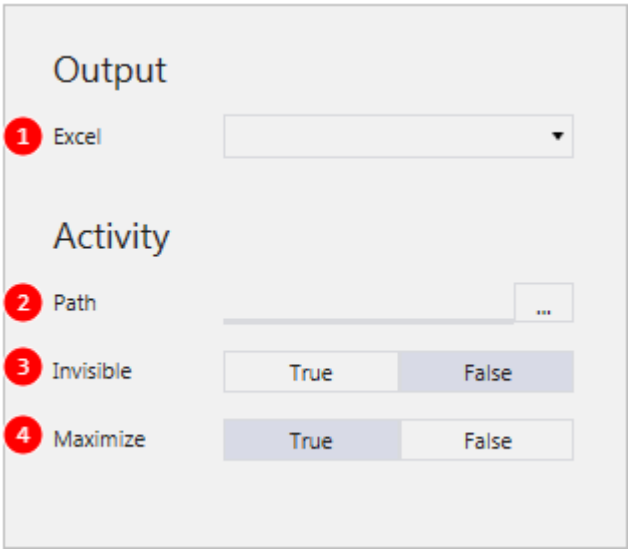
Figure 4-25 **CreateWorkSheet** properties screen

No.	Description
❶	Select the variable allocated with the opened Excel document.
❷	Enter name of the new worksheet to add within single quotation marks.
❸	Enter the location to create the new worksheet.

OpenExcel (Opening an Excel document)

OpenExcel opens an Excel document.

Figure 4-26 **OpenExcel** properties screen

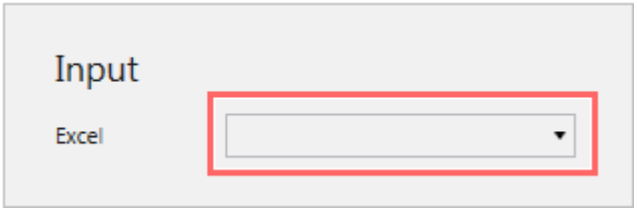


No.	Description
❶	Specify the variable to allocate to the open Excel document. You can use this in **CreateWorkSheet** and **SelectWorkSheet**, and more.
❷	Select the Excel file to open or enter the document path.
❸	Set whether to deactivate the Excel document.
❹	Set whether to maximize the Excel window.

ActivateExcel (Activating Excel window)

ActivateExcel activates the deactivated Excel document.

Figure 4-27 **ActivateExcel** properties screen



In the **Excel** field, select the variable allocated with the Excel document to activate.

SaveExcel (Saving Excel document)

SaveExcel saves an Excel document.

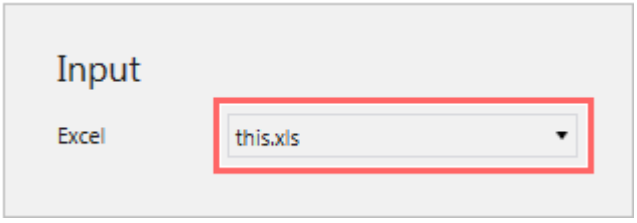
Figure 4-28 **SaveExcel** properties screen

No.	Description
❶	Select the variable allocated with the Excel document to save.
❷	To save as another name, select the path to save and enter a new filename.

CloseExcel (Closing Excel document)

CloseExcel closes an Excel document.

Figure 4-29 **CloseExcel** properties screen

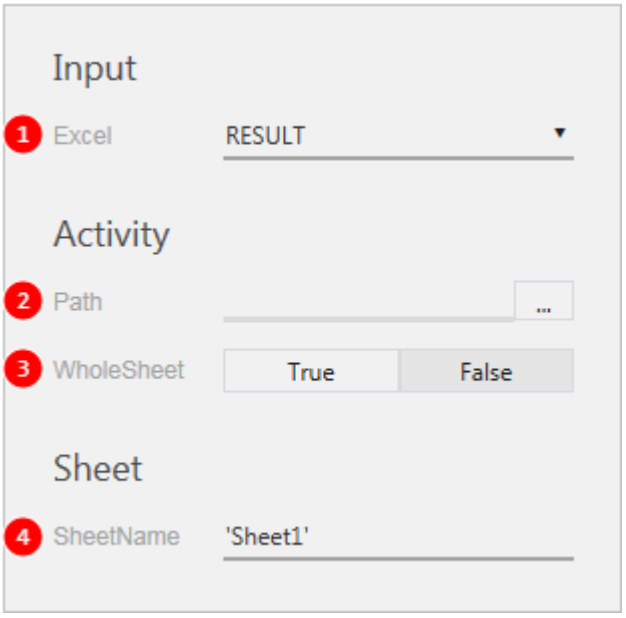


You can close the Excel document by specifying the variable allocated with the Excel document.

SplitExcel (Splitting an Excel document)

SplitExcel partially or entirely copies the worksheet of a personal Excel document that contains many worksheets to create a new Excel file.

Figure 4-30 **SplitExcel** properties screen

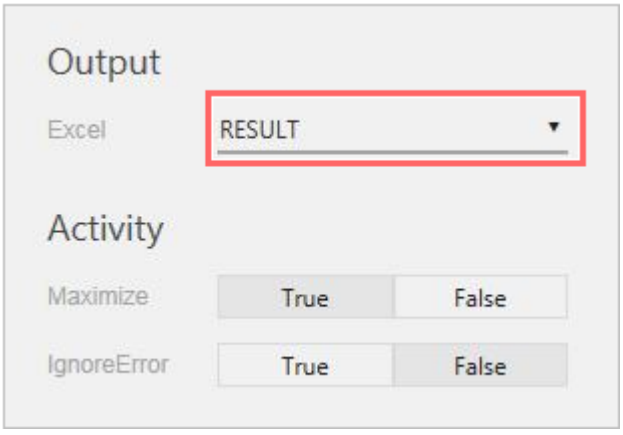


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Select the path to save the new Excel file.
❸	Select whether to copy all worksheets.
❹	Enter name of the worksheet to copy within single quotation marks.

GetActiveExcel (Getting an activated Excel document)

GetActiveExcel gets an activated Excel document and allocates to a variable.

Figure 4-31 **GetActiveExcel** properties screen

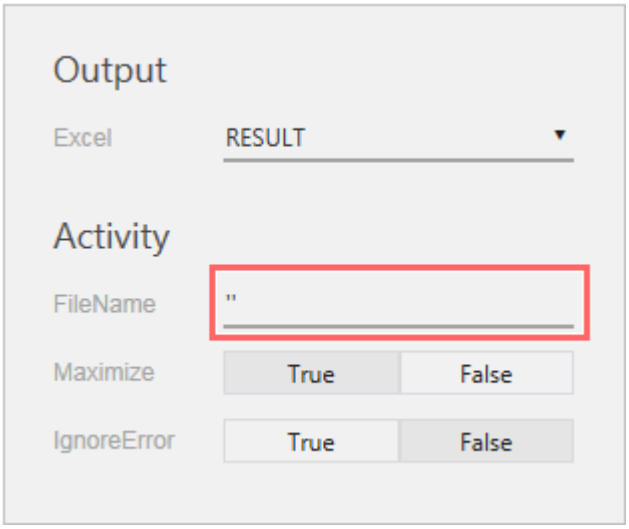


In the **Excel** field, select the variable to allocate the activated Excel document.

GetExcel (Getting an Excel document)

GetExcel gets an Excel document with the specified name from the executed Excel documents and allocates to a variable.

Figure 4-32 **GetExcel** properties screen

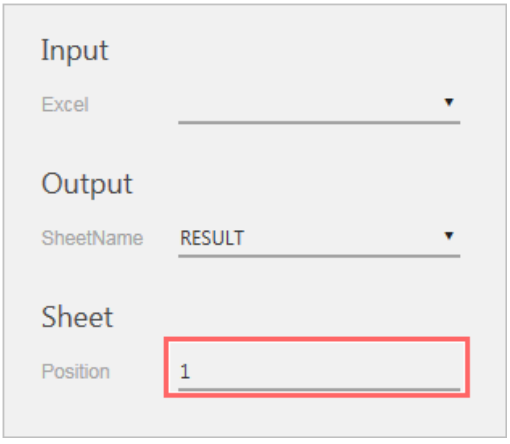


In the **FileName** field, enter the name of the Excel file to get within single quotation marks.

GetSheetName (Getting worksheet name)

GetSheetName gets the worksheet name from the entered position.

Figure 4-33 **GetSheetName** properties screen

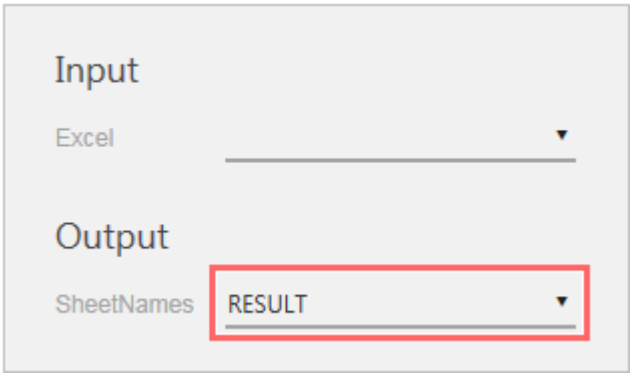


In the **Position** field, enter the location of the worksheet to get.

GetAllSheetNames (Getting all worksheets names of an Excel document)

GetAllSheetNames gets all worksheet names in an Excel document.

Figure 4-34 **GetAllSheetNames** properties screen

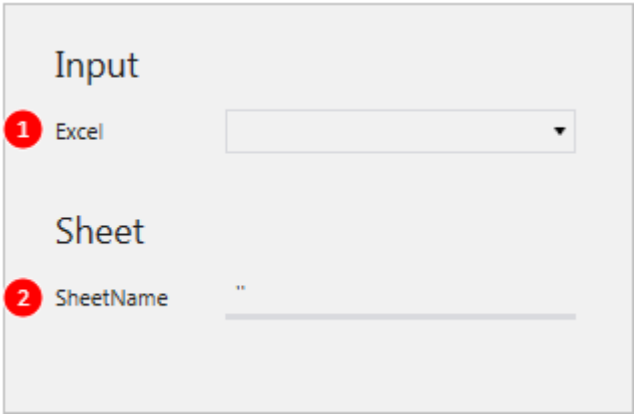


In the **SheetNames** field, select the variable to return the worksheet names as a string.

SelectWorkSheet (Selecting a specific worksheet)

SelectWorkSheet selects a specific worksheet from the Excel document.

Figure 4-35 **SelectWorkSheet** properties screen

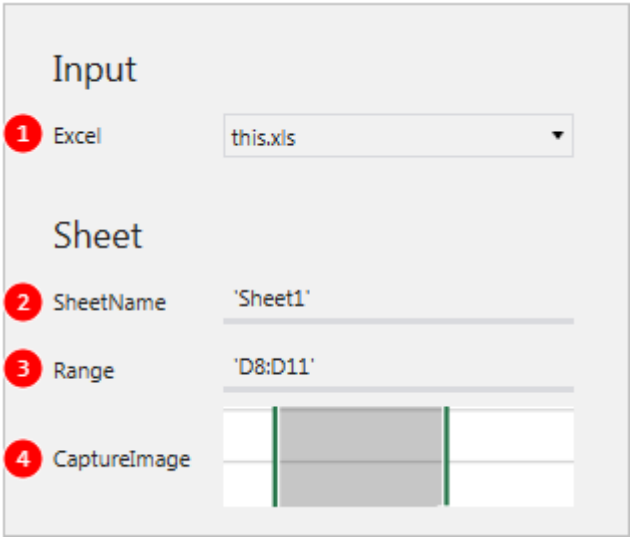


No.	Description
❶	Select the variable allocated with the opened Excel document.
❷	Enter the worksheet name to select within single quotation marks.

SelectRange (Specifying area)

SelectRange select an area from the Excel document.
On the node, double-click **No Target** > **Double click to select** and drag the area to select from the Excel document.

Figure 4-36 **SelectRange** properties screen

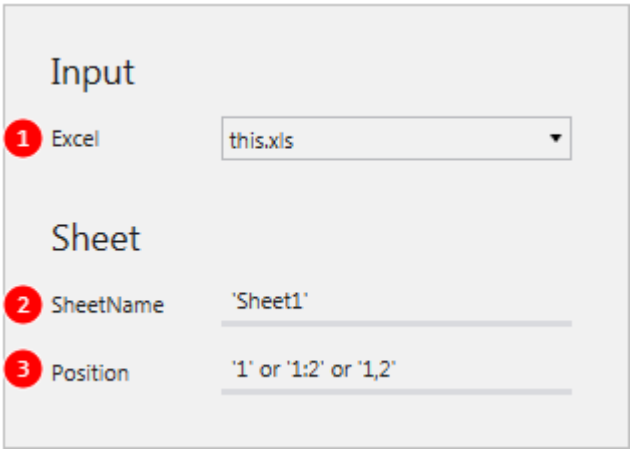


No.	Description
❶	Select the variable allocated with the Excel document containing the area to select.
❷	Enter the worksheet name containing the area to select within single quotation marks.
❸	Enter the range of the area to select within single quotation marks. For example, you can enter "A1" to select a specific column.
❹	Shows a captured screen if the area was directly selected by double-clicking **No Target**.

SelectRow (Selecting a row)

SelectRow selects a specific row.

Figure 4-37 **SelectRow** properties screen

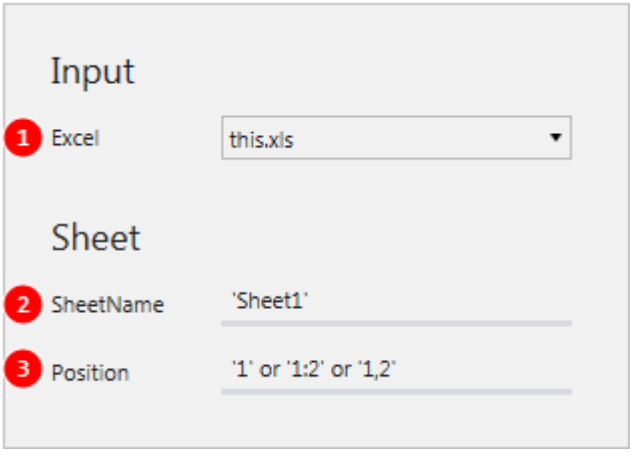


No.	Description
❶	Select the variable where the Excel document is allocated.
❷	Enter the worksheet name containing the row to select within single quotation marks.
❸	Enter the row number to select within single quotation marks. If you enter a colon ":" between the row numbers, all rows from the first row number to the second row number are selected. If you enter a comma "," between the row numbers, the specified row numbers are deleted.

SelectCol (Selecting column)

SelectCol selects a specific column.

Figure 4-38 **SelectCol** properties screen



No.	Description

No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the worksheet name containing the column to select within single quotation marks.
❸	Enter the column number to select within single quotation marks. If you enter a colon ":" between the column numbers, all columns from the first column number to the second column number are selected. If you enter a comma "," between the column numbers, the specified column numbers are selected.

GetActiveRange (Getting selected area)

GetActiveRange gets the selected area from the Excel document.

Figure 4-39 **GetActiveRange** properties screen

Specify a variable of the Excel document to get the selected area from.

DataArray reference method (Based on the retrieved range)
DataArray[0] → First row value
DataArray[0].Length → Number of first rows
DataArray[0][1] → First row value, second column value

GetDataRange (Getting data area)

GetDataRange gets the area where data is entered from the Excel document.

For example, the "A1:C3" area is allocated to the variable of **Range** field in the figure below.

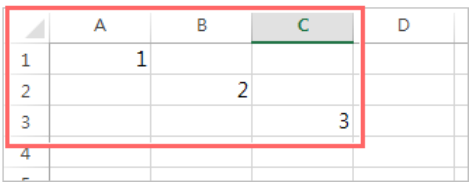
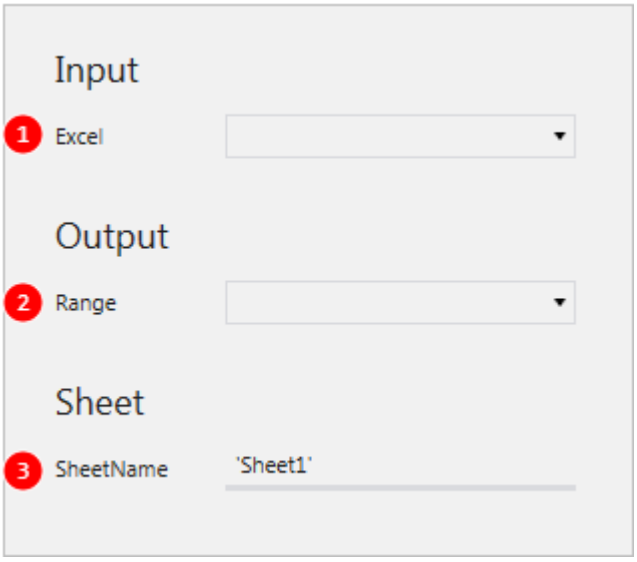


Figure 4-40 **GetDataRange** properties screen

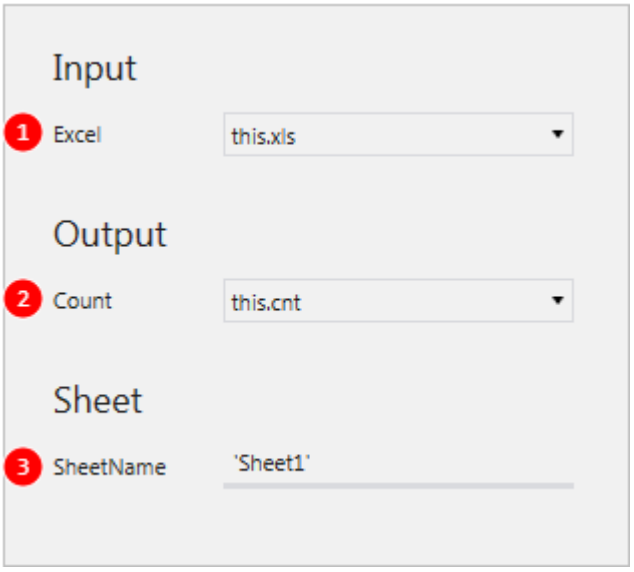


No.	Description
❶	Specify a variable of the Excel document to get the selected data area from.
❷	Specify the variable to allocate the imported data area.
❸	Enter the worksheet name with a data area to get within single quotation marks.

GetRowCount (Getting row count)

GetRowCount gets the number of rows.

Figure 4-41 **GetRowCount** properties screen

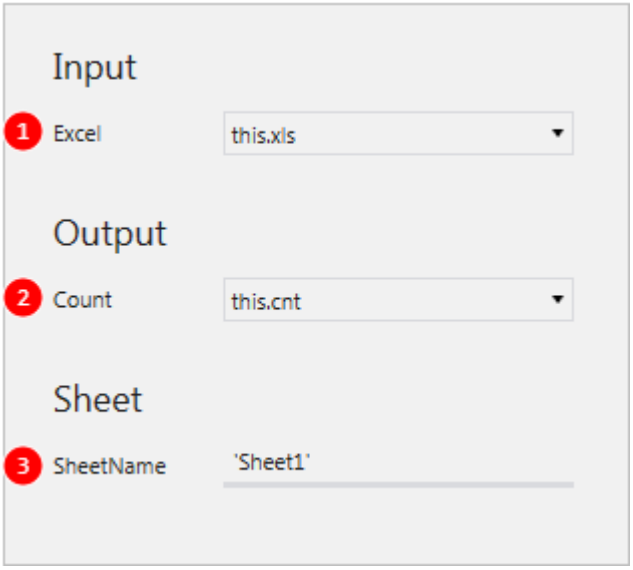


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Specify the variable to import row count.
❸	Enter the worksheet name to import row count within single quotation marks.

GetColCount (Getting Column Count)

GetColCount gets the column count.

Figure 4-42 **GetColCount** properties screen

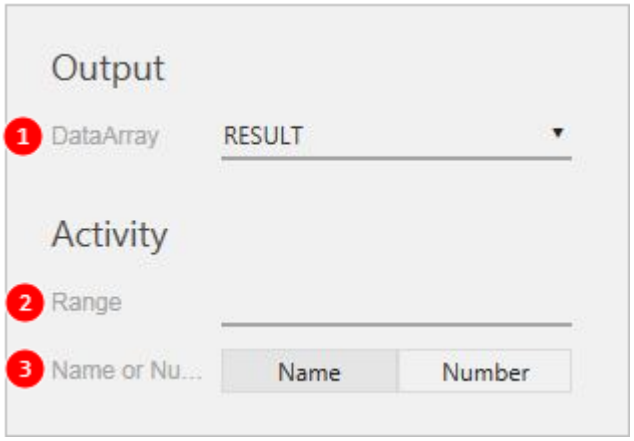


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Specify the variable to import column count.
❸	Enter the worksheet name to import the column count within single quotation marks.

SplitRange (Allocating by splitting rows and columns)

SplitRange splits rows and columns in the given area and allocates to the variable.

Figure 4-43 **SplitRange** properties screen



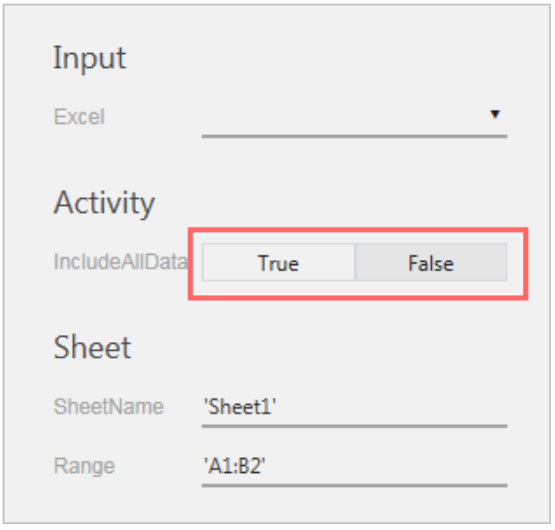
No.	Description
❶	Specify the variable to allocate the information of the split area.
❷	Enter the area to split.
❸	Set whether to receive the Column name as text or to receive it by converting it into numbers ('A' = 1).

If "A6:BB2" is entered, "A,6,BB,2" is allocated as array to the this.range variable.

MergeRange (Merging cell area)

MergeRange merges the cells of the selected area.

Figure 4-44 **MergeRange** properties screen

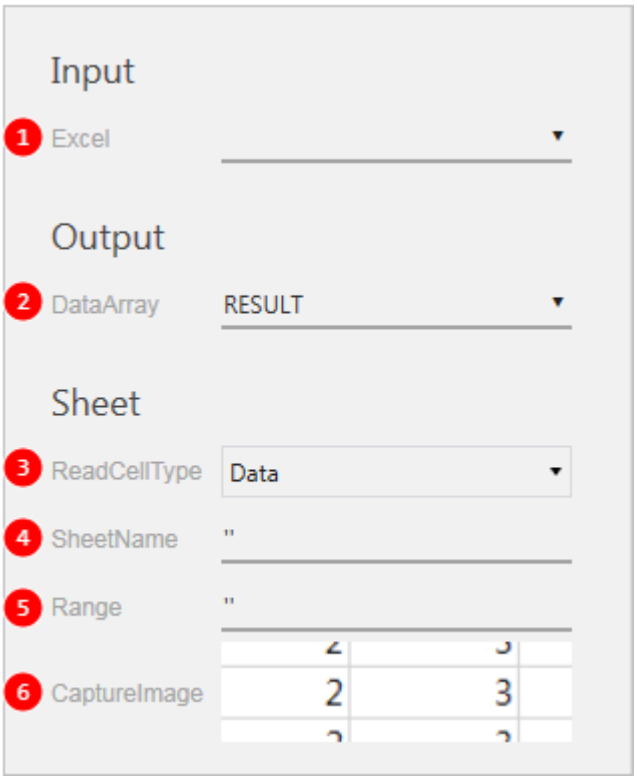


If you select **True** from the **IncludeAllData** field, data in all cells will be entered to the merged cell.
If you select **False**, only data from the first cell will be entered in the merged cell.

ReadRange (Reading-in selected area)

ReadRange reads in the selected area from the Excel document.
On the node, double-click **No Target** > **Double click to select** and drag the area to read-in from the Excel document.

Figure 4-45 **ReadRange** properties screen

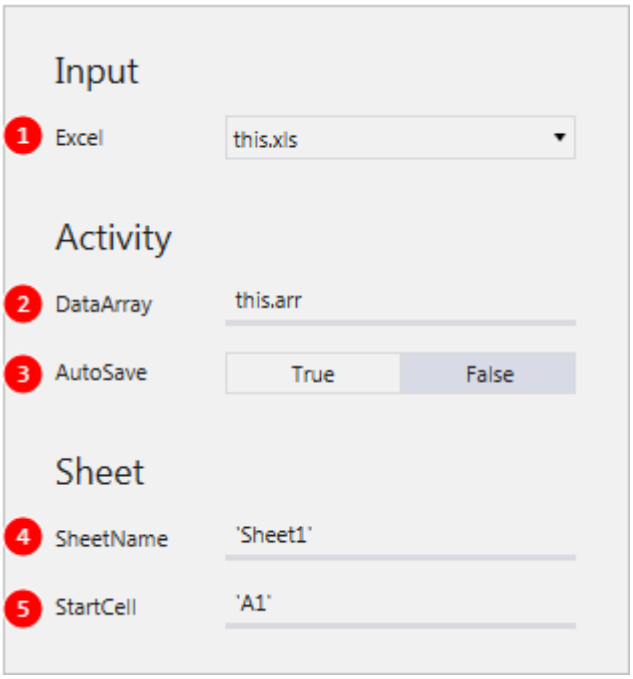


No.	Description
❶	Select the variable allocated with the Excel document containing the area to read-in.
❷	Specify the variable to allocate the read-in area.
❸	Select the type of information (Data, Value, Formula, or Text) to read-in.
❹	Enter the worksheet name containing the data area to read-in within single quotation marks.
❺	Enter the range of the area to select within single quotation marks. For example, you can enter "A1" to select a specific column.
❻	Shows a captured screen if the area was directly selected by double-clicking **No Target**.

WriteRange (Copying data area)

WriteRange copies the area where data is input from the Excel document and enters in the designated position.

Figure 4-46 **WriteRange** properties screen

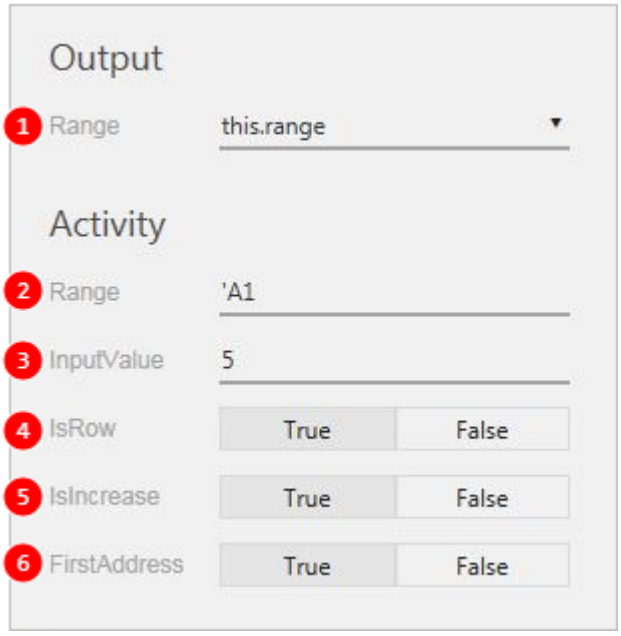


No.	Description
❶	Select the variable allocated with the Excel document to input data.
❷	Select the variable of the Excel document allocated with the data area to input.
❸	Set whether to automatically save after entering data.
❹	Enter the worksheet name to input data within single quotation marks.
❺	Enter the cell name in the upper-right side of the area to input data within single quotation marks.

MakeRange (Making area)

MakeRange creates an area and allocates to the variable.
This is used when it is difficult to enter the cell address due to the increased number of columns.

Figure 4-47 **MakeRange** properties screen



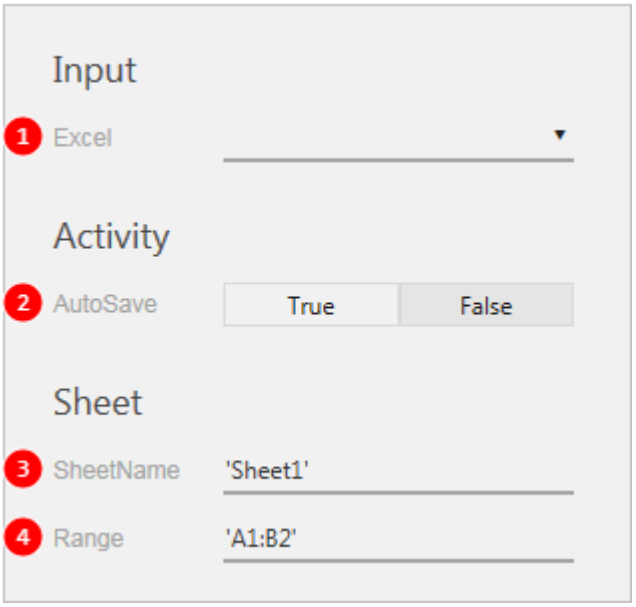
No.	Description
❶	Specify the variable to allocate the created area.
❷	Enter position to start creating the area within single quotation marks.
❸	Enter the number of rows or columns to increase or decrease based on the position of the **Range** field.
❹	Set **True** to change row and **False** to change column.
❺	Set **True** to increase and **False** to decrease.
❻	Select **True** to set the MakeRange target as the starting address if the Range is specified as an area and not as one cell.

"A6", which is the result from the above input will be assigned to the this.range variable.

ClearRange (Clearing area)

ClearRange deletes data from the specified cell area.

Figure 4-48 **ClearRange** properties screen



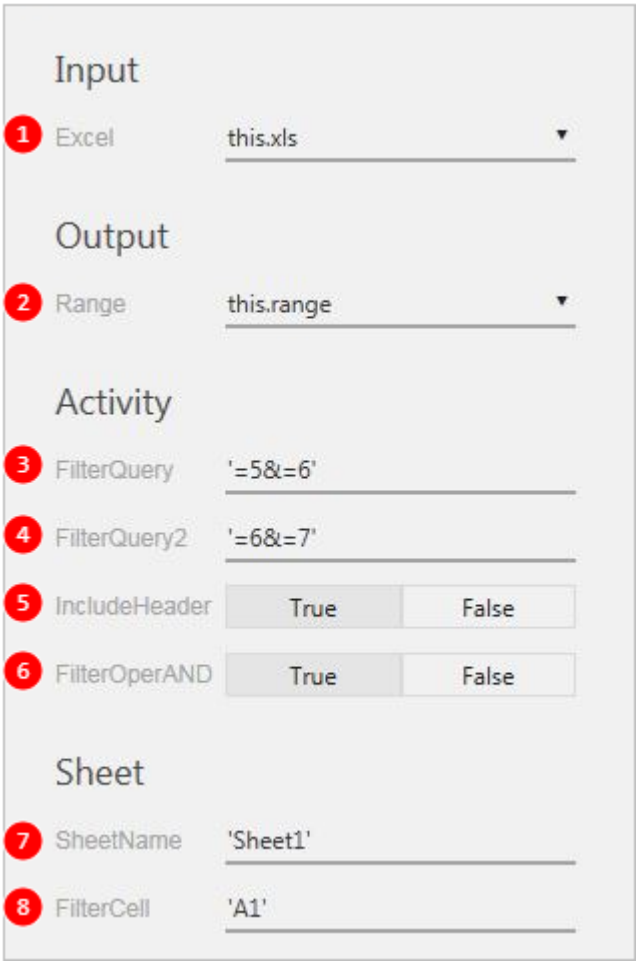
No.	Description
❶	Select the variable of the Excel document to delete.
❷	Set whether to save automatically after deleting data.
❸	Enter the worksheet name to delete data from within single quotation marks.
❹	Enter the cell area to delete date from within single quotation marks.

SetFilter (Specifying filter)

SetFilter specify a filter from an Excel document with filter. This can be used when there is a dropdown button shown in the figure below.

	A	B
	index	data
	11	a
	12	b
	13	c
	24	d
	25	e
	26	f
	37	g
	38	h
	39	i

Figure 4-49 **SetFilter** properties screen



No.	Description
❶	Select the variable allocated with the Excel document.
❷	Specify the variable to save the result.
❸	Enter the first query value to apply to the filter within single quotation marks.
❹	Enter the second query value to apply to the filter within single quotation marks.
❺	Set whether to include the header with filter when importing data.
❻	Select **True** to apply the first and second query value, and select **False** to apply the first or second query value.
❼	Enter the worksheet name to import data from within single quotation marks.
❽	Enter the cell position of the filter within single quotation marks.

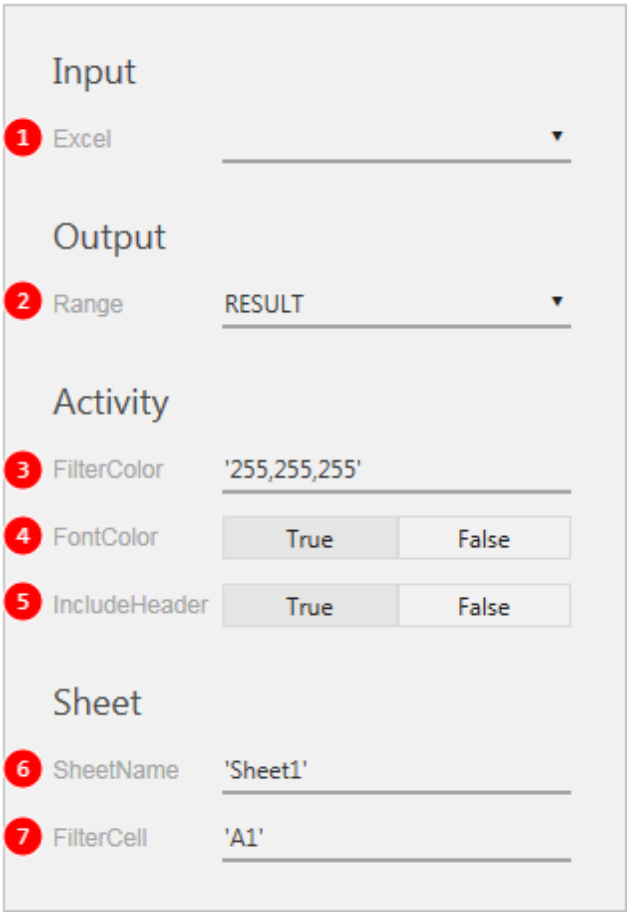
Learn more about filter query values:
'=test' → Filters the "test" string only.
'<>test' → Filters except the "test" string.

'=test*' → Filters strings that starts with "test."
'=*test' → Filters strings that end with "test."
'=*test*' → Filters all strings that include "test."
'<>*test*' → Filters all strings that do not include "test."
'=test??' → Filters the string that starts with "test" and has two more characters on the end.
'=??test' → Filters the string that ends with "test" and has two more characters on the front.
'=??test??' → Filters all strings that have two or more characters on both sides of "test".
'<>??test??' →Filters all strings that do not contain two or more characters on both sides of "test".
'>10' → Filters only the numbers higher than 10.
'>=10' → Filters only the numbers equal to or higher than 10.
'<10' → Filters only the numbers lower than 10.

SetColorFilter (Specifying color filter)

SetColorFilter specifies a filter by font or cell color.

Figure 4-50 **SetColorFilter** properties screen

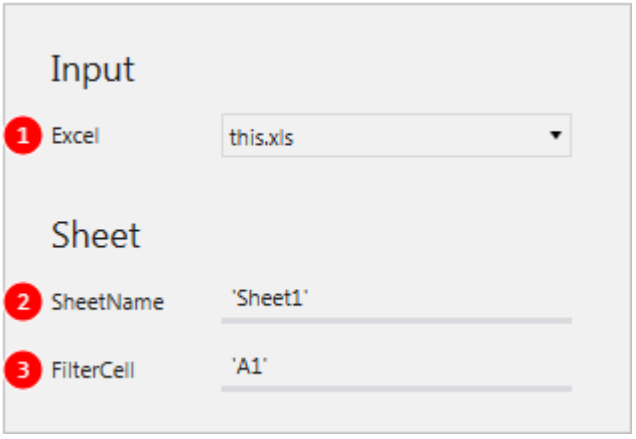


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Specify the variable to save the result.
❸	Enter the color value to specify the filter within single quotation marks.
❹	Set whether to specify a filter by font color.
❺	Set whether to include the header with filter when importing data.
❻	Enter the worksheet name to import data from within single quotation marks.
❼	Enter the cell position of the filter within single quotation marks.

ReleaseFilter (Releasing filter)

ReleaseFilter releases the filter which is set in the Excel document.

Figure 4-51 **ReleaseFilter** properties screen

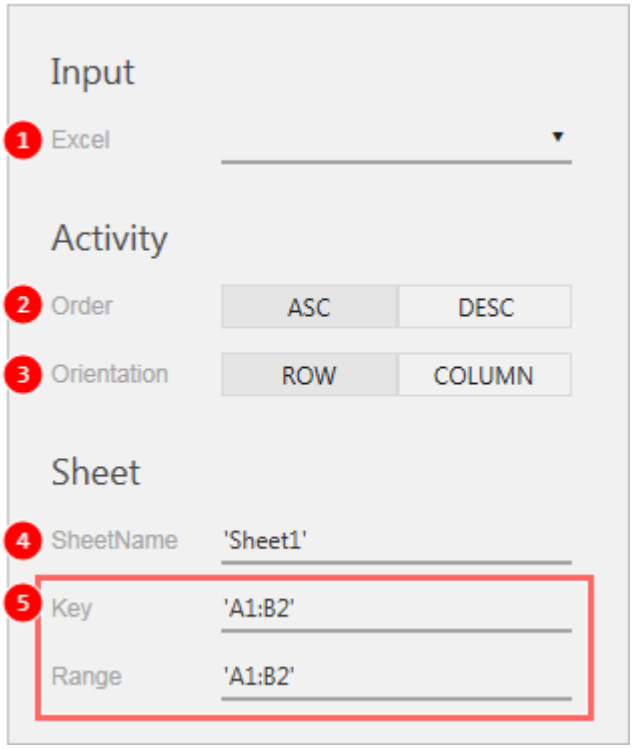


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the worksheet name containing a filter to release within single quotation marks.
❸	Enter the cell position of the filter within single quotation marks.

SortData (Sorting)

SortData can sort the specified column or row in ascending or descending order.

Figure 4-52 **SortData** properties screen

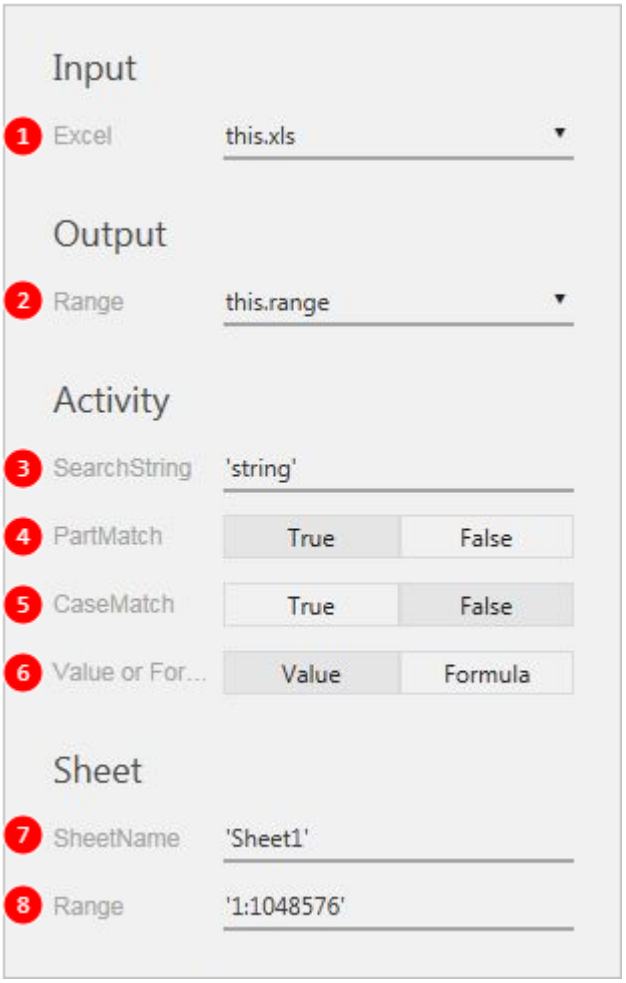


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Set whether to sort in ascending order or align in descending order.
❸	Select whether the data to sort is a row or column.
❹	Enter the worksheet name containing data to sort within single quotation marks.
❺	Enter the cell position containing data to sort within single quotation marks.

SearchString (Searching string)

SearchString searches for strings in an Excel document.
If the string is found, the position of string is returned.

Figure 4-53 **SearchString** properties screen

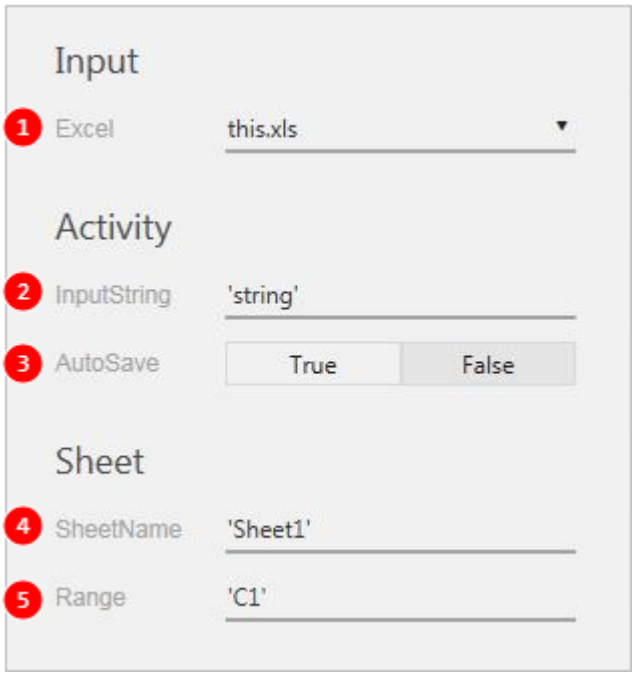


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Specify the variable to return the address of the searched string.
❸	Enter the string to search for within single quotation marks.
❹	Set whether to allow partial search.
❺	Set whether to use case-sensitive.
❻	Set whether to find a value or formula.
❼	Enter the worksheet name of the Excel document to search for the string within single quotation marks.
❽	Set the range of the Excel document to search the string. "1:1048576" is the default value and means that the search will be performed from row 1 to 1048576.

WriteString (Inputting string)

WriteString inputs a string in the cell which is set by the Excel document.

Figure 4-54 **WriteString** properties screen

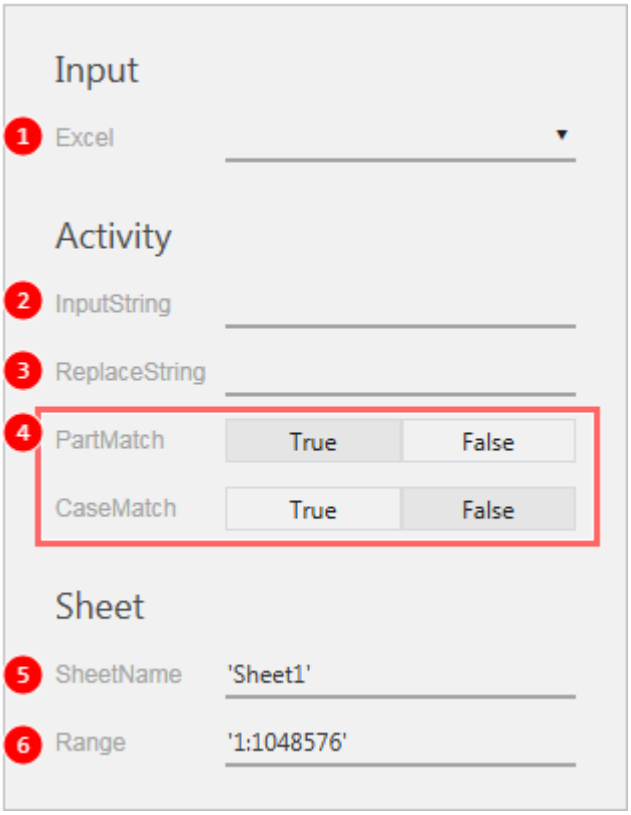


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the string to input within single quotation marks.
❸	Set whether to save automatically.
❹	Enter the worksheet name to input string within single quotation marks.
❺	Enter the cell address to input string within single quotation marks.

ReplaceString (Replacing string)

ReplaceString searches a string and replaces with another string.

Figure 4-55 **ReplaceString** properties screen

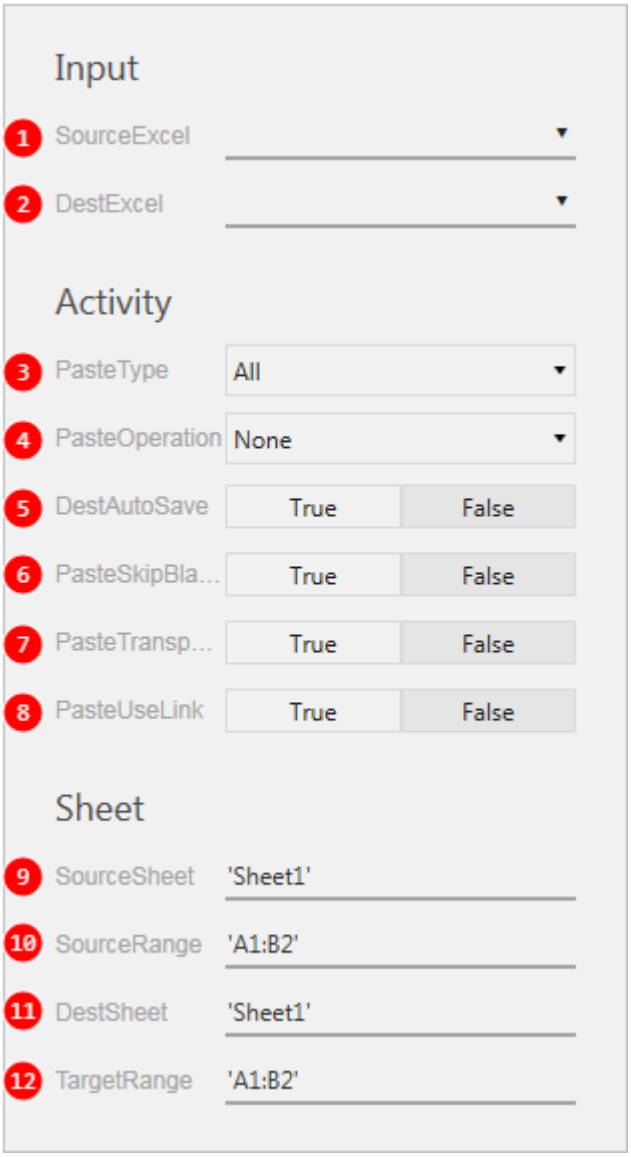


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter a string to search.
❸	Enter a string to replace.
❹	Select the matching method.
❺	Enter the worksheet name containing the string to replace within single quotation marks.
❻	Enter the cell address containing the string to be replaced within single quotation marks.

CopyPaste (Copying and pasting)

CopyPaste copies and pastes a specific area of Excel document.

Figure 4-56 **CopyPaste** properties screen



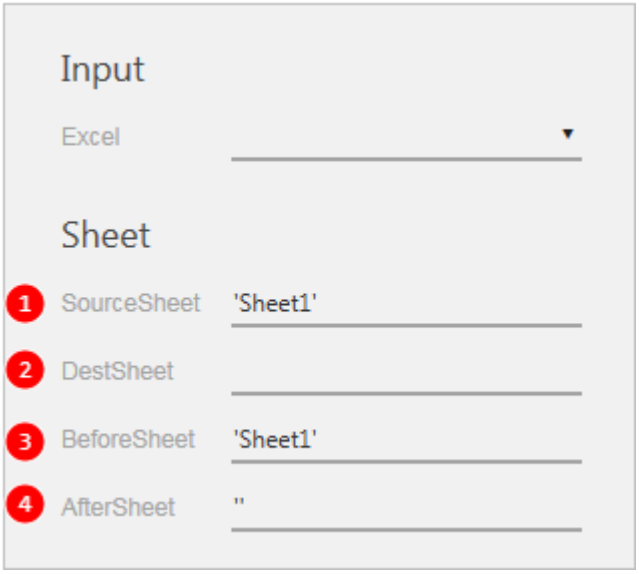
No.	Description
❶	Select the variable of the Excel document containing the area to copy.
❷	Select the variable of the Excel document to paste.
❸	Select the type of value to paste.
❹	Select the operation to paste.
❺	Set whether to save the pasted Excel document automatically.
❻	Set whether to paste the cell with content only.
❼	Set whether to paste by changing the row/column of the area to copy.
❽	Set whether to paste the link of the area to copy.
❾	Enter name of the worksheet containing the area to copy within single quotation marks.

No.	Description
⑩	Enter area of the Excel document to copy within single quotation marks.
⑪	Enter name of the worksheet to copy within single quotation marks.
⑫	Set whether to automatically save the Excel document to paste.

CopyWorkSheet (Copying specific worksheet)

CopyWorkSheet copies specific worksheet from the Excel document.

Figure 4-57 **CopyWorkSheet** properties screen



No.	Description
①	Enter name of the worksheet to copy within single quotation marks.
②	Enter name of the worksheet to paste the copied worksheet within single quotation marks.
③	Enter name of the worksheet right before the position of the worksheet to copy within single quotation marks.
④	Enter name of the worksheet right after the position of the worksheet to copy within single quotation marks.

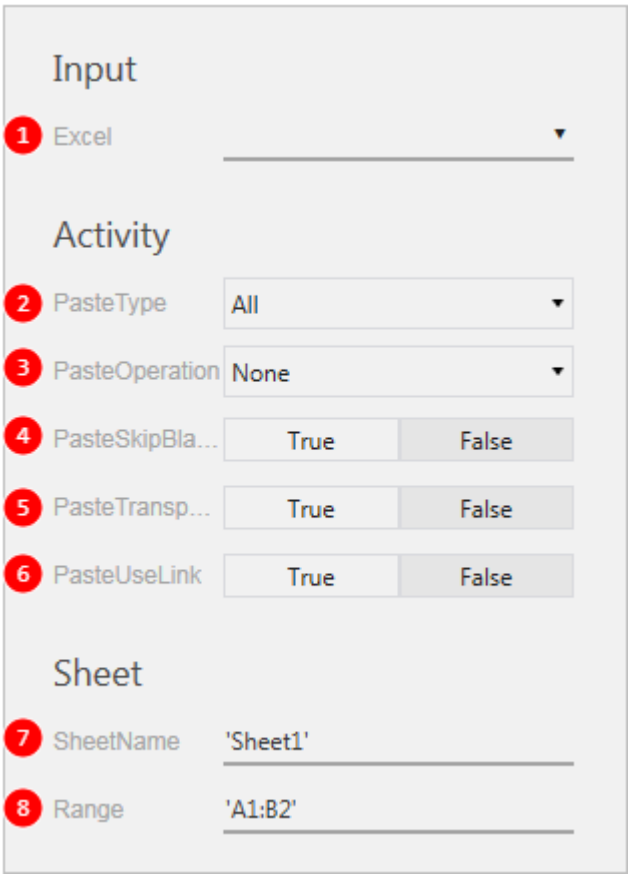
CopyRange (Copying an area)

CopyRange copies data of the specified area.

PasteRange (Pasting an area)

PasteRange pastes data of the copied area.

Figure 4-58 **PasteRange** properties screen

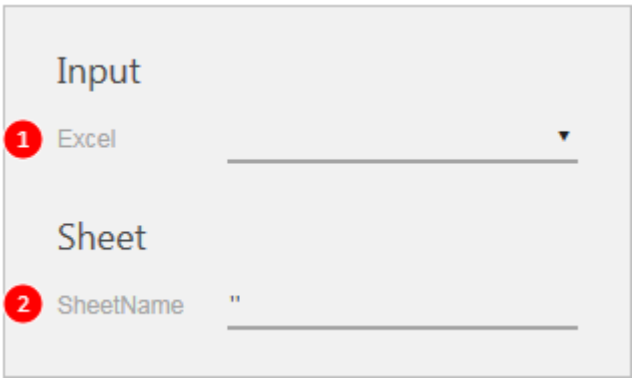


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Select the type of data to paste.
❸	Select the operation to paste.
❹	Set whether to paste the data with content only.
❺	Set whether to paste by changing the row/column of the copied area.
❻	Set whether to paste the link of the copied area.
❼	Enter name of the worksheet to copy within single quotation marks.
❽	Enter the position of the data to paste within single quotation marks.

DeleteWorkSheet (Deleting a worksheet)

DeleteWorkSheet deletes a specific worksheet from the Excel document.

Figure 4-59 **DeleteWorkSheet** properties screen

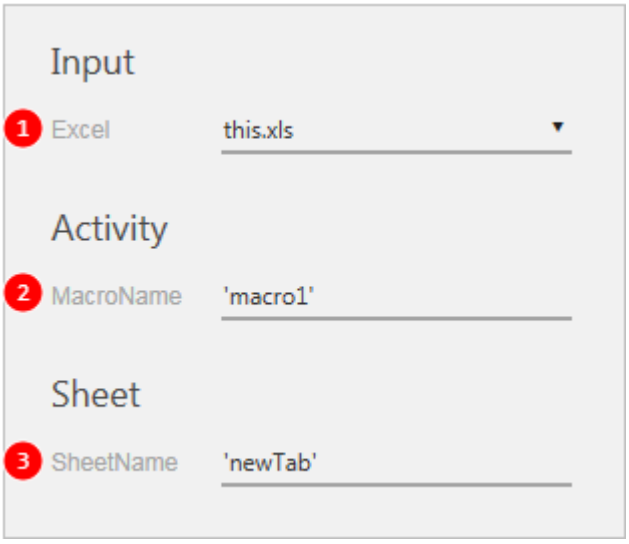


No.	Description
❶	Select the variable allocated with the opened Excel document.
❷	Enter the worksheet name to delete within single quotation marks.

ExecuteMacro (Executing macro)

ExecuteMacro executes macro that is saved in the Excel document.
The macro must be created in advance to use this feature.

Figure 4-60 **ExecuteMacro** properties screen



No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the macro name to execute within single quotation marks.
❸	Enter the worksheet name to execute the macro within single quotation marks.

AddRow (Adding rows)

AddRow adds rows to an Excel document.

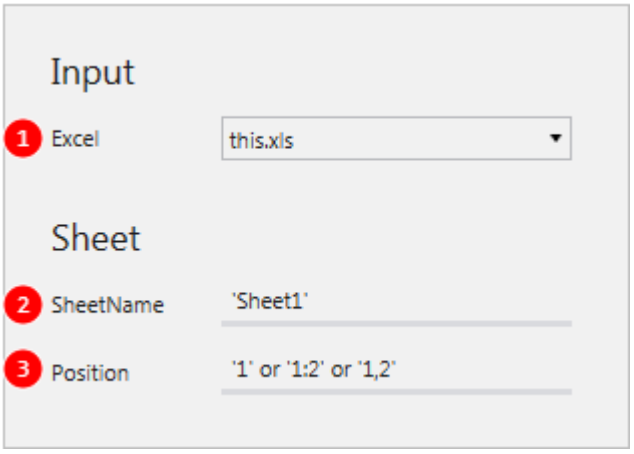
Figure 4-61 **AddRow** properties screen

No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the worksheet name to add rows to within single quotation marks.
❸	Enter a row number of the position to add the rows within single quotation marks. If you enter a colon ":" between the row numbers, all rows from the first row number to the second row number are added. If you enter a comma "," between the row numbers, the specified row numbers are added.

AddCol (Adding columns)

AddCol adds columns to an Excel document.

Figure 4-62 **AddCol** properties screen

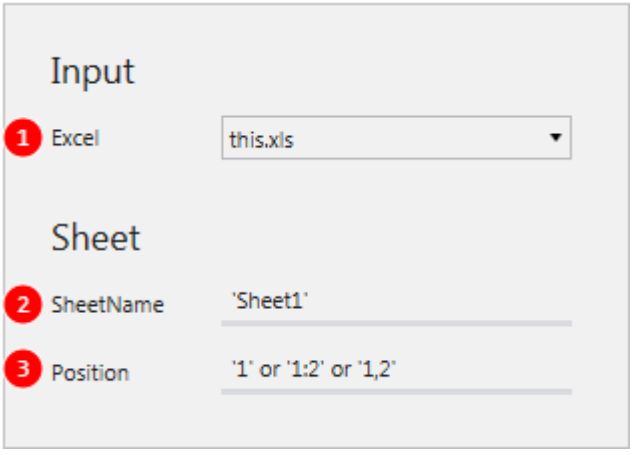


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the name of the worksheet to add columns to within single quotation marks.
❸	Enter the column number of the position to add the columns within single quotation marks. If you enter a colon ":" between the column numbers, all columns from the first column number to the second column number are added. If you enter a comma "," between the column numbers, the specified column numbers are added.

DeleteRow (Deleting rows)

DeleteRow deletes a specific row.

Figure 4-63 **DeleteRow** properties screen



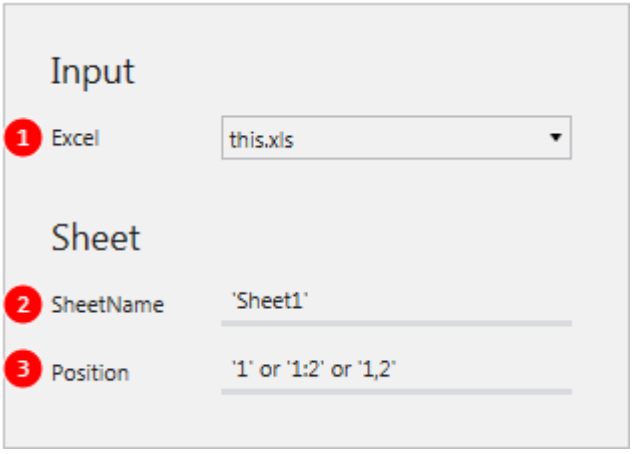
No.	Description

No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the name of the worksheet containing rows to delete within single quotation marks.
❸	Enter the row number to delete within single quotation marks. If you enter a colon ":" between the row numbers, all rows from the first row number to the second row number are deleted. If you enter a comma "," between the row numbers, the specified row numbers are deleted.

DeleteCol (Deleting columns)

DeleteCol deletes a specific column.

Figure 4-64 **DeleteCol** properties screen

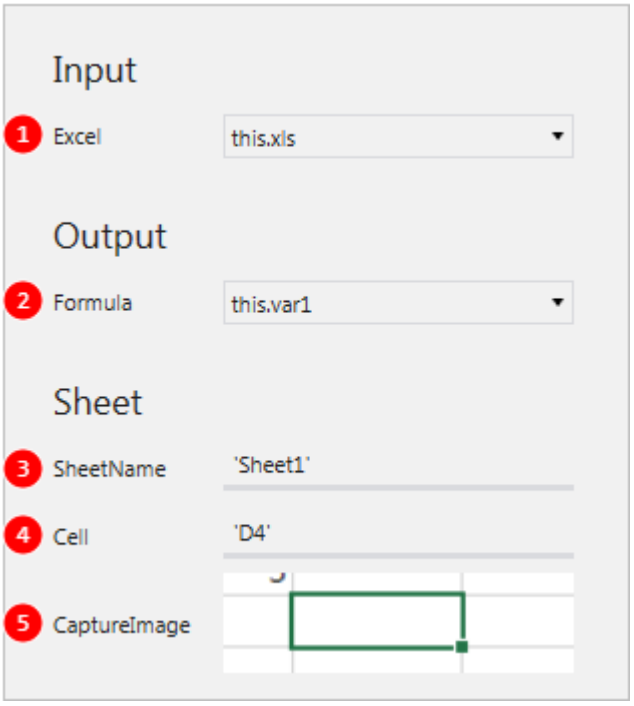


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter the name of the worksheet containing the column to delete within single quotation marks.
❸	Enter the text of the column to delete within single quotation marks. If you enter a colon ":" between the column numbers, all column from the first column number to the second column number are deleted. If you enter a comma "," between the column numbers, the specified column numbers are deleted.

ReadCellFormula (Importing input value of a cell)

ReadCellFormula imports a formula entered in a specific cell.

Figure 4-65 **ReadCellFormula** properties screen

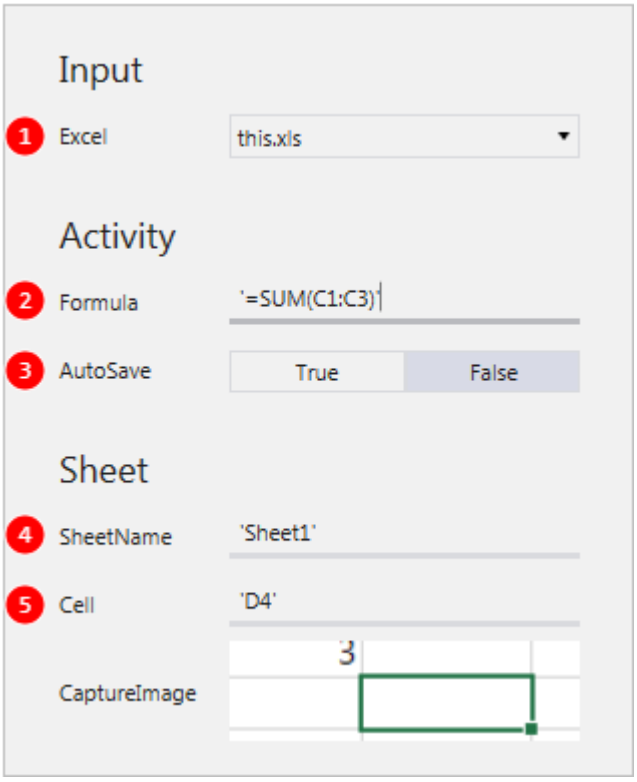


No.	Description
❶	Select the variable allocated with the Excel document.
❷	Specify the variable to allocate the formula that has been read-in.
❸	Enter the name of the worksheet to read-in the formula within single quotation marks.
❹	Enter the cell address to read-in the formula within single quotation marks.
❺	Shows a captured screen if the area was directly selected by double-clicking **No Target**.

WriteCellFormula (Inputting formula to a cell)

WriteCellFormula inputs formula to a specific cell.

Figure 4-66 **WriteCellFormula** properties screen



No.	Description
❶	Select the variable allocated with the Excel document.
❷	Enter a formula to input into a cell within single quotation marks.
❸	Set whether to save automatically.
❹	Enter the name of the worksheet to input formula within single quotation marks.
❺	Enter the cell address to input formula within single quotation marks.

SAP library

You can control SAP GUI using the features provided by the **SAP** library.
You can enter data into SAP GUI or import the entered data and change and handle the data.
The features provided by the SAP library are as follows:

- Mouse action features (Click, DoubleClick, RightClick, MouseHover)
- Keyboard action features (SetText, GetText)
- Features related to the target UI object recognition (IsExist, WaitSAPAppear, WaitSAPDisappear)
- Features related to the SAP Grid (GetGridColumnID, GetGridRowCount, GetGridColCount)
- Features related to the combo box (SelectComboItem)

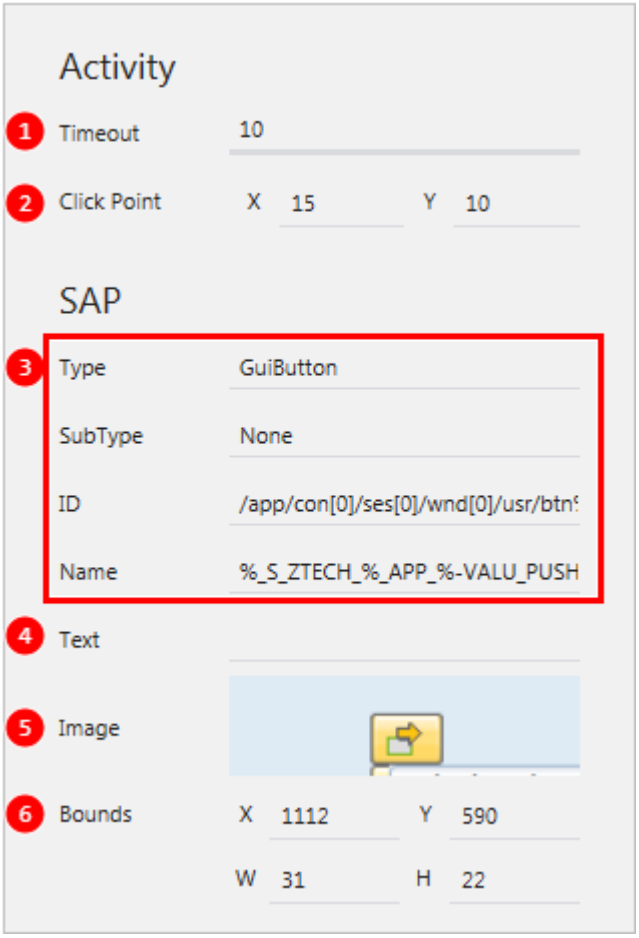
Click (Clicking)

Click finds and clicks the UI object designated by the user.

1. On the node, double-click **No Target** > **Double click to select**.
2. If you put the mouse over the UI object, a red border will appear. Click when a red border appears on the UI object to generate the click event.

The UI object is designated as the target UI object.

Figure 4-67 **Click** properties screen



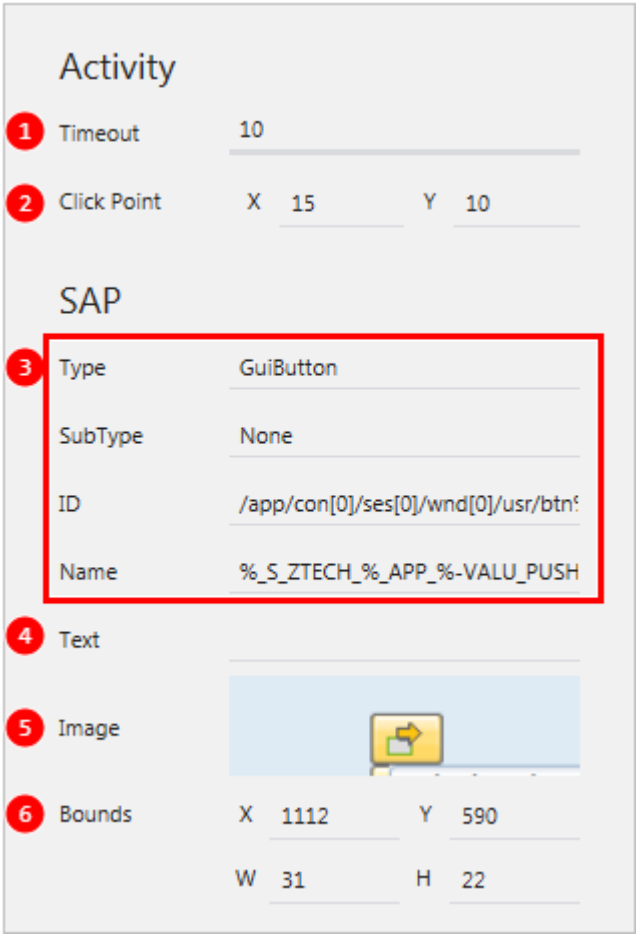
No.	Description
❶	Enter the duration to search for the target UI object in seconds.
❷	Enter the location to click after finding the target UI object as coordinates based on the top left section of the target UI object. This will be entered automatically if the object is designated directly with a mouse.
❸	Shows the type, sub-type, ID, and name information of the target UI object appears.

No.	Description
❹	Shows the text value of the target UI object.
❺	Shows the image of the target UI object designated by the user.
❻	Shows the position and size of the target UI object designated by the user as relative coordinates based on the screen.

DoubleClick (Double clicking)

DoubleClick double-clicks the target UI object.

Figure 4-68 **DoubleClick** properties screen



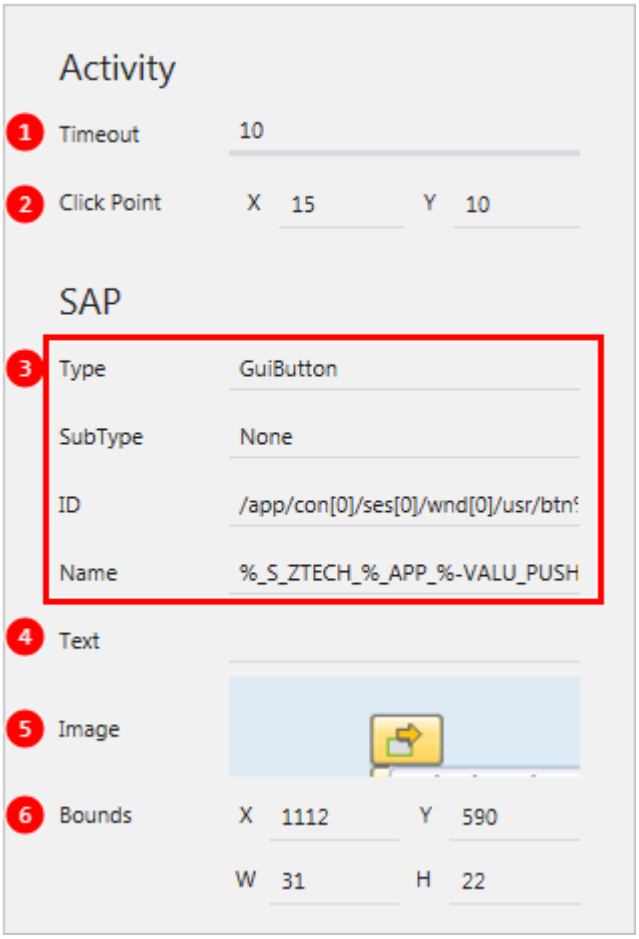
No.	Description
❶	Enter the duration to search for the target UI object in seconds.
❷	Enter the location to double-click after finding the target UI object as coordinates based on the upper-left section of the target UI object.

No.	Description
	This will be entered automatically if the object is designated directly with a mouse.
❸	Shows the type, sub-type, ID, and name information of the target UI object appears.
❹	Shows the text value of the target UI object.
❺	Shows the image of the target UI object designated by the user.
❻	Shows the position and size of the target UI object designated by the user as relative coordinates based on the screen.

RightClick (Clicking the right mouse button)

RightClick right-clicks on the target UI object.

Figure 4-69 **RightClick** properties screen



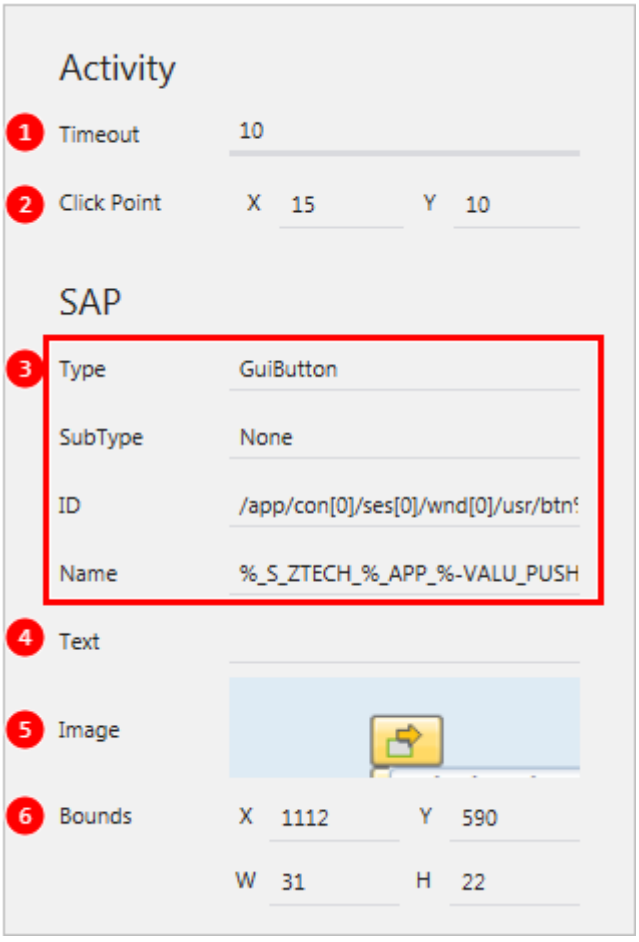
No.	Description
❶	Enter the duration to search for the target UI object in seconds.

No.	Description
❷	Enter the location to click after finding the target UI object as coordinates based on the top left section of the target UI object. This will be entered automatically if the object is designated directly with a mouse.
❸	Shows the type, sub-type, ID, and name information of the target UI object appears.
❹	Shows the text value of the target UI object.
❺	Shows the image of the target UI object designated by the user.
❻	Shows the position and size of the target UI object designated by the user as relative coordinates based on the screen.

MouseHover (Hovering the mouse cursor)

MouseHover hovers the mouse cursor over the target UI object.

Figure 4-70 **MouseHover** properties screen

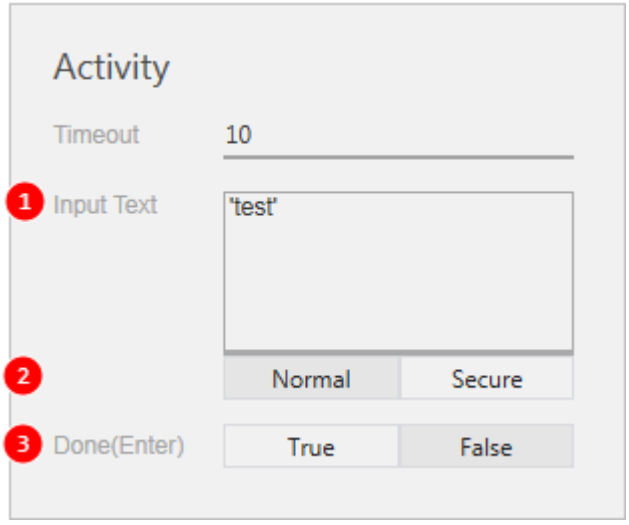


No.	Description
❶	Enter the duration to search for the target UI object in seconds.
❷	Enter the location to hover the cursor over after finding the target UI object as coordinates based on the upper-left section of the target UI object. This will be entered automatically if the object is designated directly with a mouse.
❸	Shows the type, sub-type, ID, and name information of the target UI object appears.
❹	Shows the text value of the target UI object.
❺	Shows the image of the target UI object designated by the user.
❻	Shows the position and size of the target UI object designated by the user as relative coordinates based on the screen.

SetText (Inputting text)

SetText inputs text in the target UI object. This can be used only if the target UI object supports ValuePattern, like a text box, and is not read-only.

Figure 4-71 **SetText** properties screen

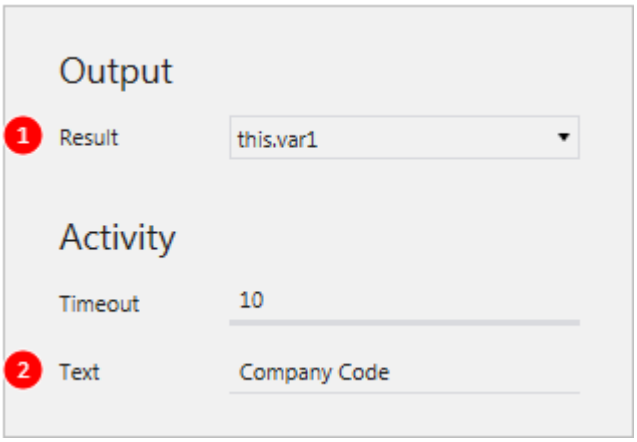


No.	Description
❶	Enter the text to input within single quotation marks.
❷	Set whether to encrypt the entered text.
❸	Set whether to press the Enter key after entering the text.

GetText (Retrieving text)

GetText retrieves the text of the target UI object.

Figure 4-72 **GetText** properties screen



No.	Description
❶	Specify the variable to allocate the text read from the target UI object.
❷	Shows the text that is read-in from the target UI object.

IsExist (Checking existence)

IsExist checks whether the target UI object is existent on the screen.
Returns True if the target UI object is found on the screen within the time entered in the **Timeout** field. Returns False if it is not found.

WaitSAPAppear (Waiting until appeared)

WaitAppear waits until the target UI object appears on the screen.
This can be useful for changing the screen.

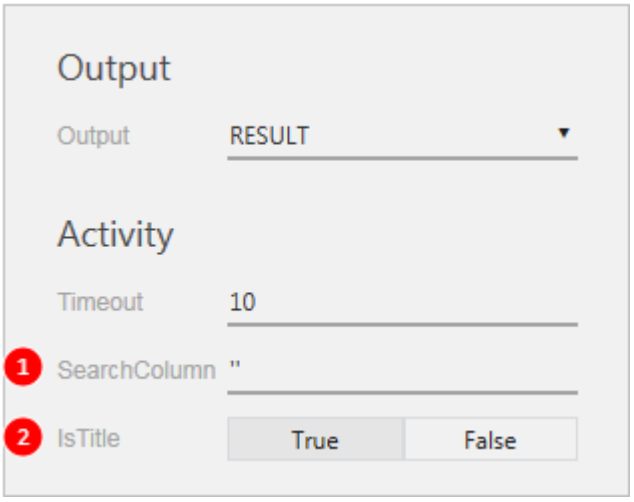
WaitSAPDisappear (Waiting until disappeared)

WaitDisappear waits until the target UI object disappears on the screen.
This can be useful for changing the screen.

GetGridColumnID (Getting the GridColumnID)

GetGridColumnID gets the Column ID of an area.

Figure 4-73 **GetGridColumnID** properties screen



No.	Description
❶	Enter the column name to get the ColumnID from.
❷	Set whether the designated target UI object is a title or not.

GetGridRowCount (Getting the row count)

GetGridRowCount gets the number of rows.

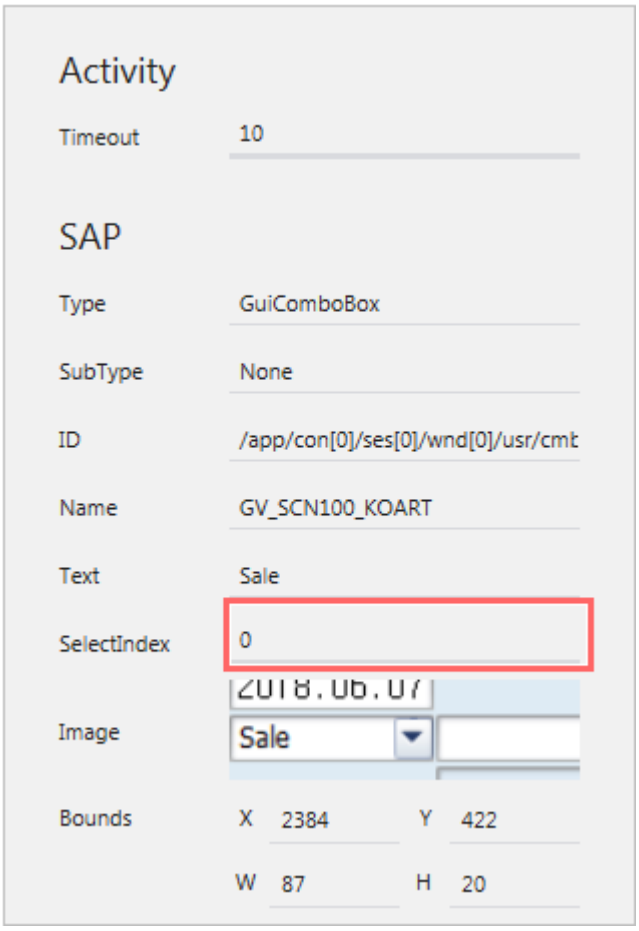
GetGridColCount (Getting the column count)

GetGridColCount gets the number of columns.

SelectComboItem (Selecting ComboItem value)

SelectComboItem selects a value of user's choice from the UI object in the form of a combo box.

Figure 4-74 **SelectComboItem** properties screen



In the **SelectIndex** field, enter index of the item to select.
The index starts from 0.

Image recognition library

You can specify an image as the target UI using the features provided by the **ImageRecognition** library. If a user captures an image, a library action is performed by recognizing the same UI factor as this image. You can perform actions, such as clicking a specific position of image or converting the characters inside an image into texts.

The features provided by the ImageRecognition library are as follows:

- Mouse action features (ImageClick, ImageHover, ImageRightClick, ImageDoubleClick)

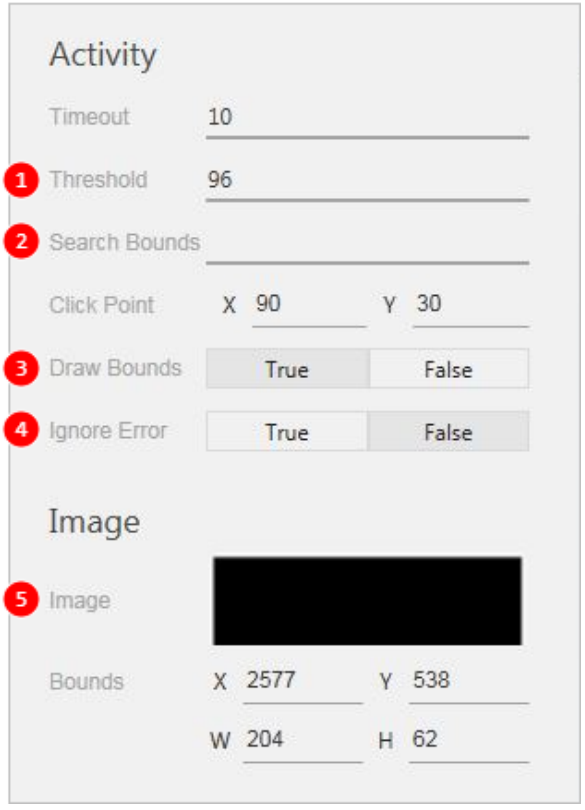
- Features related to OCR (GetImageText, GetImageTextInfo, GetImageTable, TextClick, TextDoubleClick, TextRightClick, TextHover)
- Features to check existence of image (IsExist, WaitImageAppear, WaitImageDisappear, MatchImages)
- Features on image capture (CaptureScreen, CaptureArea)
- Features related to image color (GetImageColor)

ImageClick (Clicking an image)

ImageClick clicks a specific position over an image.

1. On the node, double-click **No Target** > **Double click to select**.
2. You can capture the area by dragging over the desired area.
3. Select the location to click.

Figure 4-75 **ImageClick** properties screen

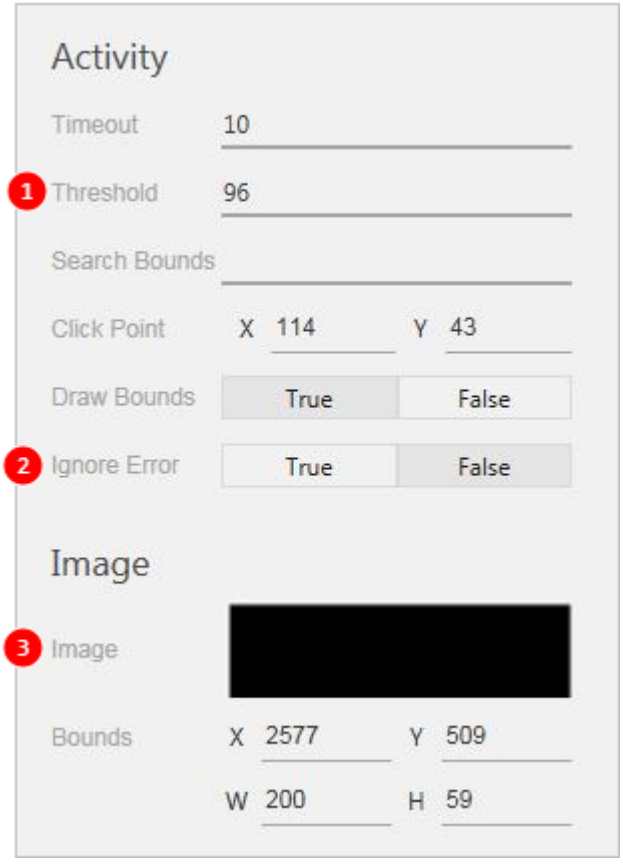


No.	Description
❶	Enter a value between 0-100 to set the similarity of images. The probability of not matching becomes higher as it gets closer to 0.
❷	Select the specific area to search for, not the entire zone.
❸	Select whether to draw a red border on the image area.
❹	Set whether to continue a process or task even in the event of an error.
❺	Shows the captured image. If you double-click the image, you can change the image settings using ImageEditor. • **Load**: Loads the image captured in the past. • **Crop**: Crops the captured image. • **Apply**: Reselects the location to click again. • **Reset**: Resets the changed settings.

ImageHover (Hovering mouse cursor over image)

ImageHover hovers mouse pointer above a specific location of the image.

Figure 4-76 **ImageHover** properties screen

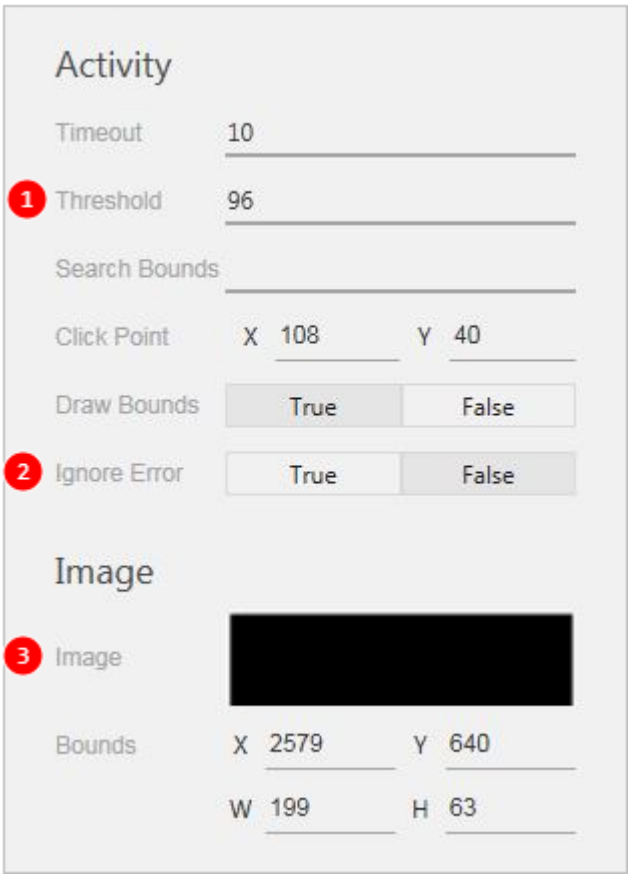


No.	Description
❶	Enter a value between 0-100 to set the similarity of images. The probability of not matching becomes higher as it gets closer to 0.
❷	Set whether to continue a process or task even in the event of an error.
❸	Shows the captured image. If you double-click the image, you can change the image settings using ImageEditor. • **Load**: Loads the image captured in the past. • **Crop**: Crops the captured image. • **Apply**: Reselects the location to click again. • **Reset**: Resets the changed settings.

ImageRightClick (Clicking image with right mouse button)

ImageRightClick right-clicks on a specific location of the image.

Figure 4-77 **ImageRightClick** properties screen



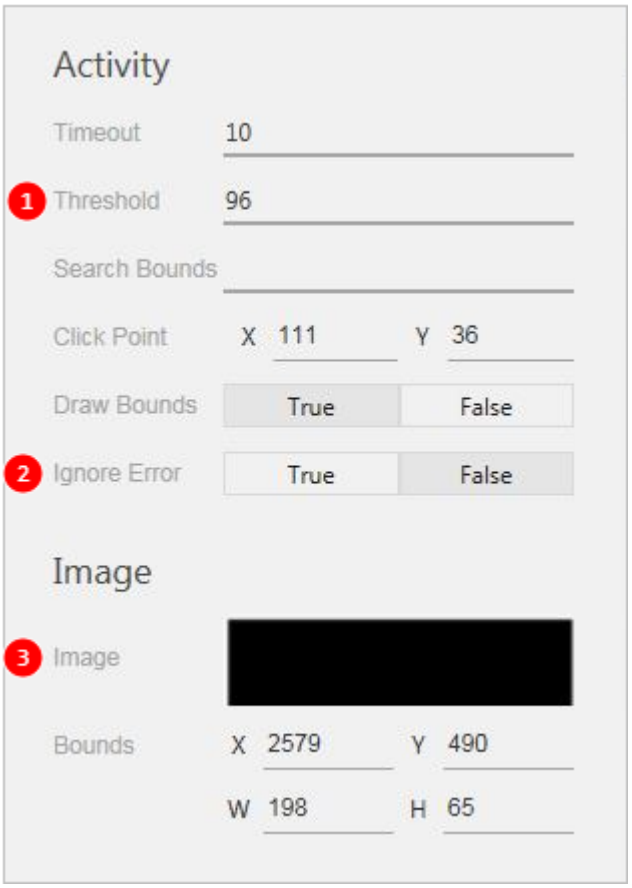
No.	Description

No.	Description
❶	Enter a value between 0-100 to set the similarity of images. The probability of not matching becomes higher as it gets closer to 0.
❷	Set whether to continue a process or task even in the event of an error.
❸	Shows the captured image. If you double-click the image, you can change the image settings using ImageEditor. • **Load**: Loads the image captured in the past. • **Crop**: Crops the captured image. • **Apply**: Reselects the location to click again. • **Reset**: Resets the changed settings.

ImageDoubleClick (Double-clicking an image)

ImageDoubleClick double-clicks on a specific location of the image.

Figure 4-78 **ImageDoubleClick** properties screen



No.	Description
❶	Enter a value between 0-100 to set the similarity of images. The probability of not matching becomes higher as it gets closer to 0.
❷	Set whether to continue a process or task even in the event of an error.
❸	Shows the captured image. If you double-click the image, you can change the image settings using ImageEditor. • **Load**: Loads the image captured in the past. • **Crop**: Crops the captured image. • **Apply**: Reselects the location to click again. • **Reset**: Resets the changed settings.

GetImageText (Getting text from image)

GetImageText converts the characters in an image into text.

Figure 4-79 **GetImageText** properties screen

GetImageTextInfo (Reading text information)

GetImageTextInfo returns the image information, such as height and width, by recognizing the image and reading in the text.

GetImageTable (Reading table text)

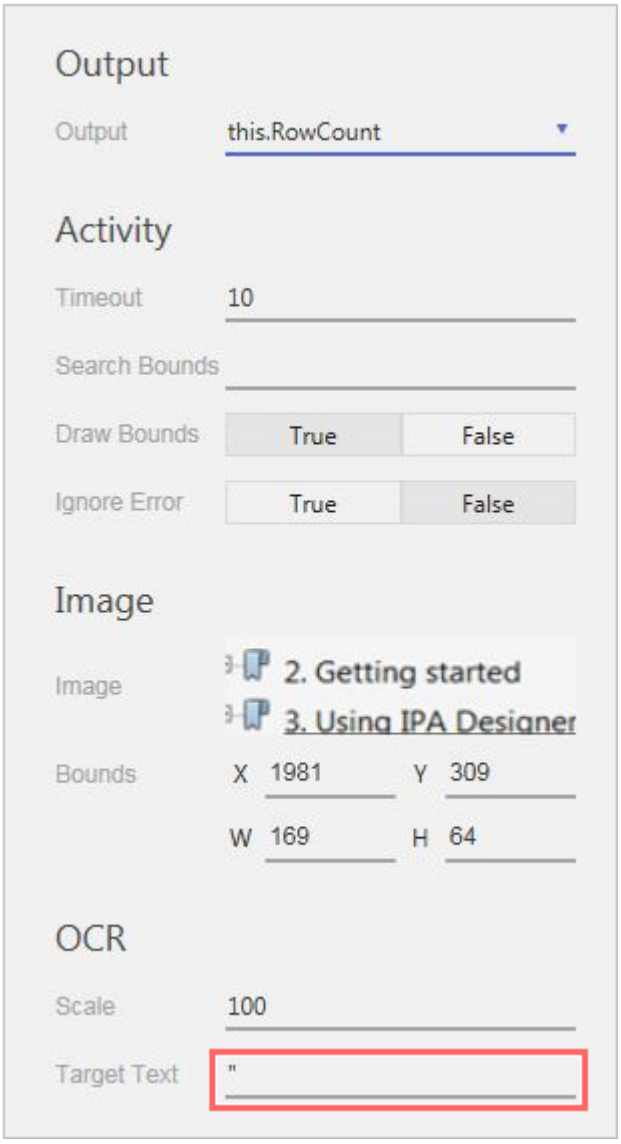
GetImageTable reads in text by recognizing tables in an image.

Figure 4-80 **GetImageTable** properties screen

TextClick (Clicking text)

TextClick finds specific text from an image and clicks.

Figure 4-81 **TextClick** properties screen

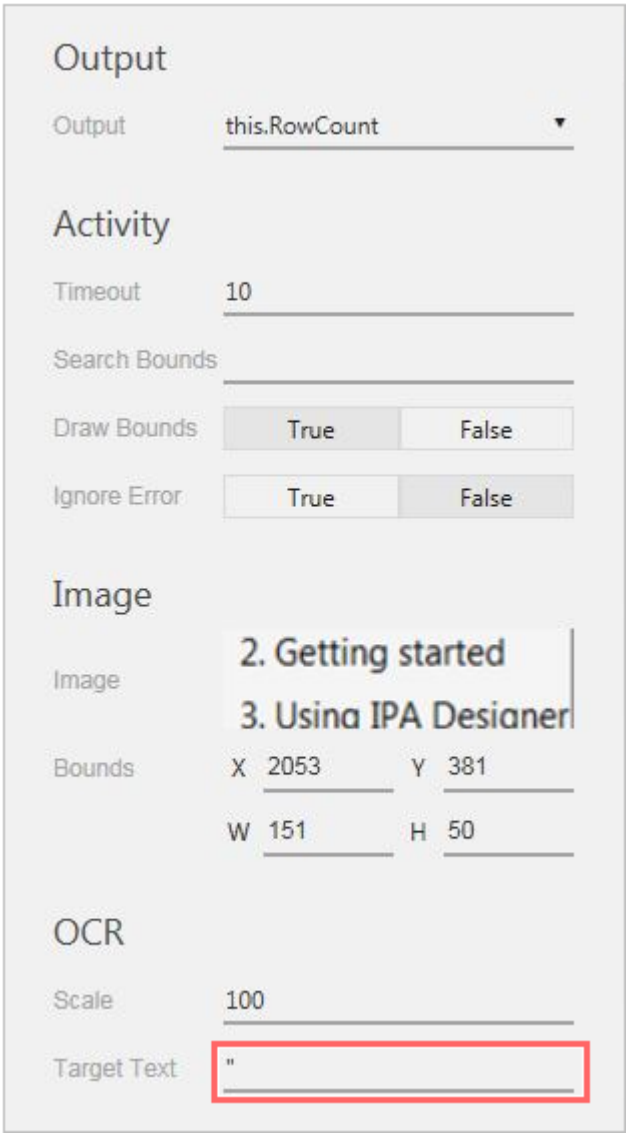


In the **Target Text** field, enter the text to click within single quotation marks.

TextDoubleClick (Double-clicking text)

TextDoubleClick finds specific text from an image and double-clicks.

Figure 4-82 **TextDoubleClick** properties screen

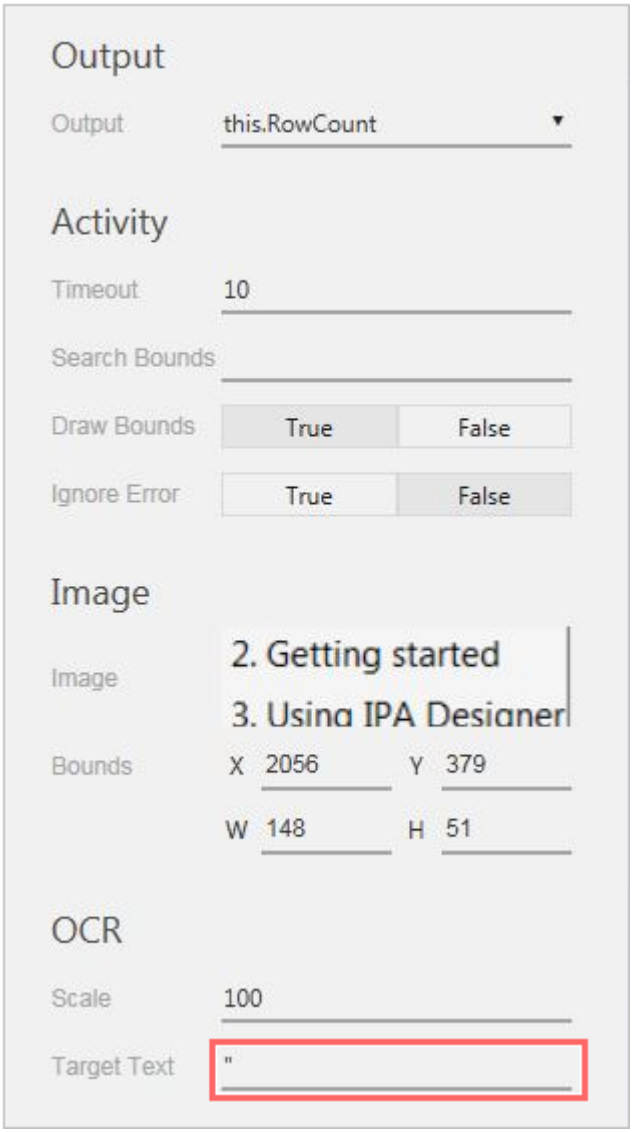


In the **Target Text** field, enter the text to select within single quotation marks.

TextRightClick (Right-clicking text)

TextRightClick finds specific text from an image and right-clicks.

Figure 4-83 **TextRightClick** properties screen

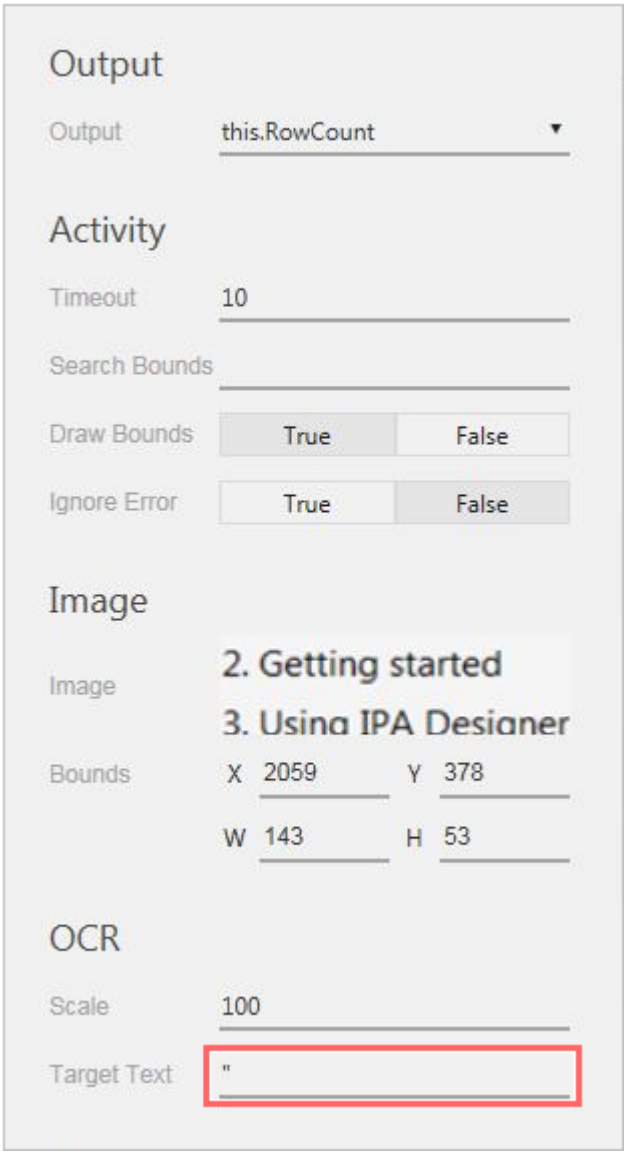


In the **Target Text** field, enter the text to click within single quotation marks.

TextHover (Hovering mouse cursor over image)

TextHover finds specific text from an image and places the mouse cursor.

Figure 4-84 **TextHover** properties screen

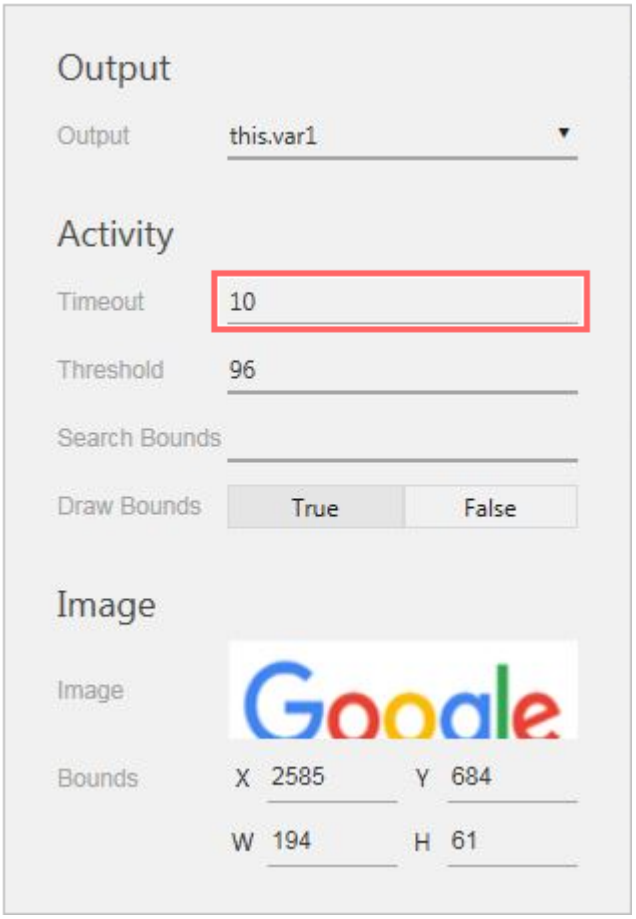


In the **Target Text** field, enter the text to hover the mouse cursor over within the single quotation marks.

IsExist (Checking existence)

IsExist checks whether a specific image exists on the screen.
Returns True if the image is found on the screen within the time entered in the **Timeout** field. Returns False if it is not found.

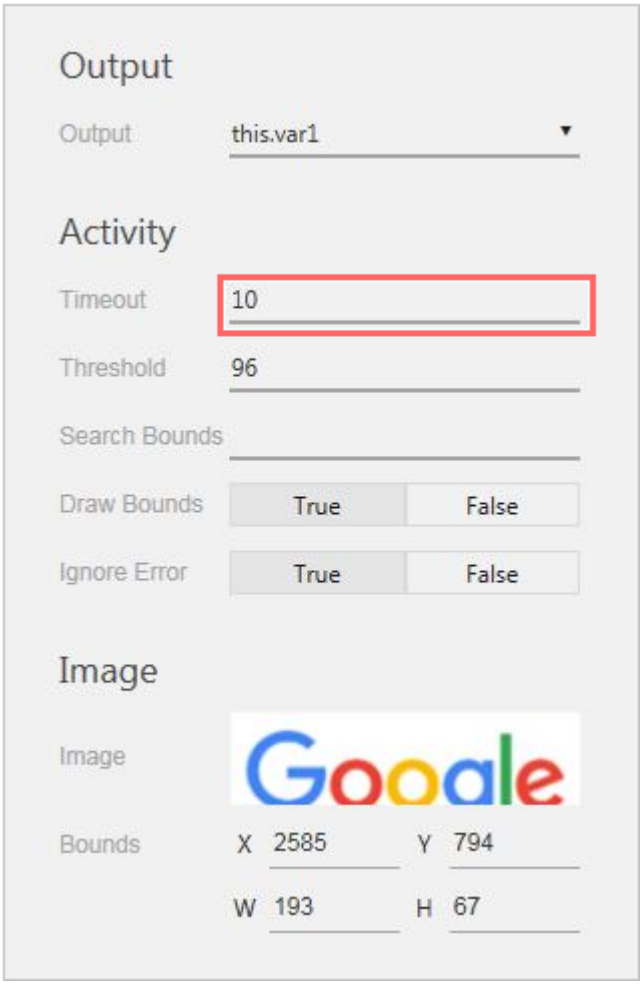
Figure 4-85 **IsExist** properties screen



WaitImageAppear (Waiting until image appears)

WaitImageAppear waits until a specific image appears on the screen.
False is returned if the duration exceeds the value entered in the **Timeout** field.

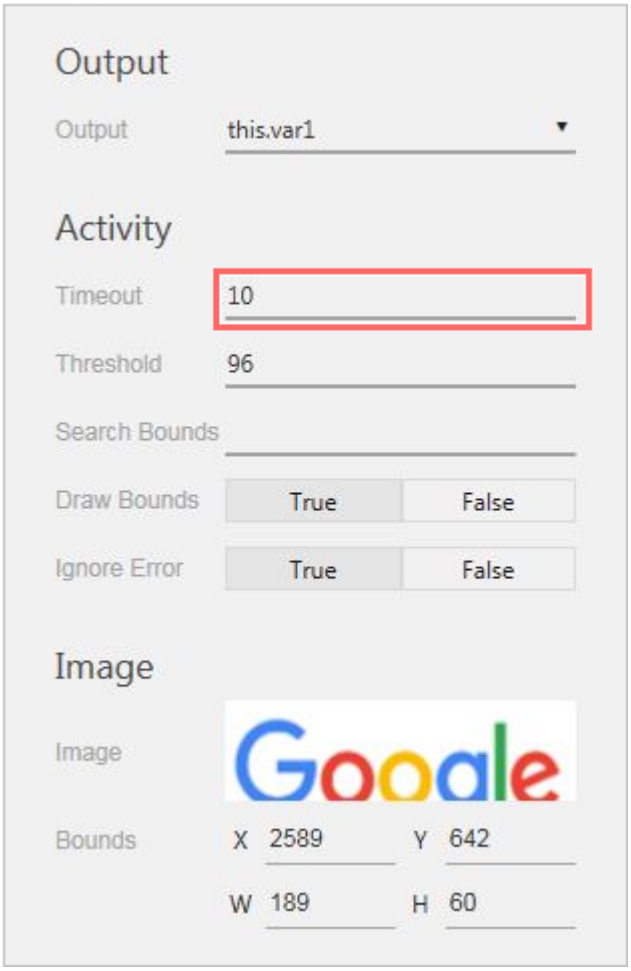
Figure 4-86 **WaitImageAppear** properties screen



WaitImageDisappear(Waiting until image disappears)

WaitImageDisappear waits until a specific image disappears on the screen.
False is returned if the duration exceeds the value entered in the **Timeout** field.

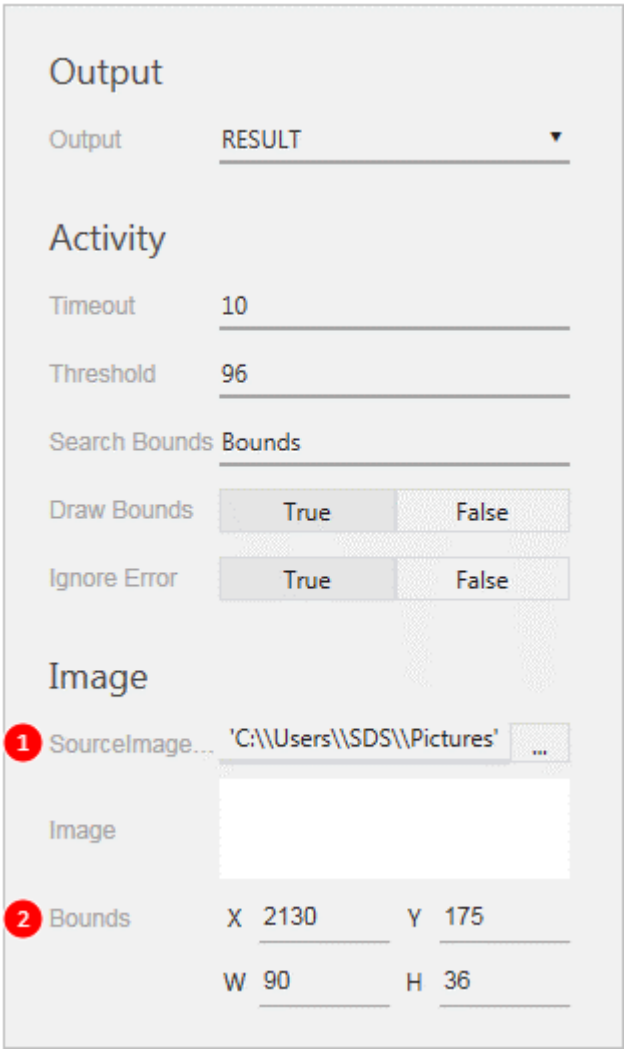
Figure 4-87 **WaitImageDisappear** properties screen



MatchImages (Matching images)

MatchImages returns a filename by finding an image file that matches with the image inside a specific folder within a specified area. An empty string is returned if there is no matching image.

Figure 4-88 **MatchImages** properties screen



No.	Description
❶	Select the path of the image file to compare.
❷	Specify the area of the image to compare.

CaptureScreen (Capturing full screen)

CaptureScreen captures the full screen of the current local PC.

Figure 4-89 **CaptureScreen** properties screen

You can select a path from the **CapturePath** field, and the captured image will be saved to this path.

CaptureArea (Capturing specific area)

CaptureArea captures a specified area of the PC screen.

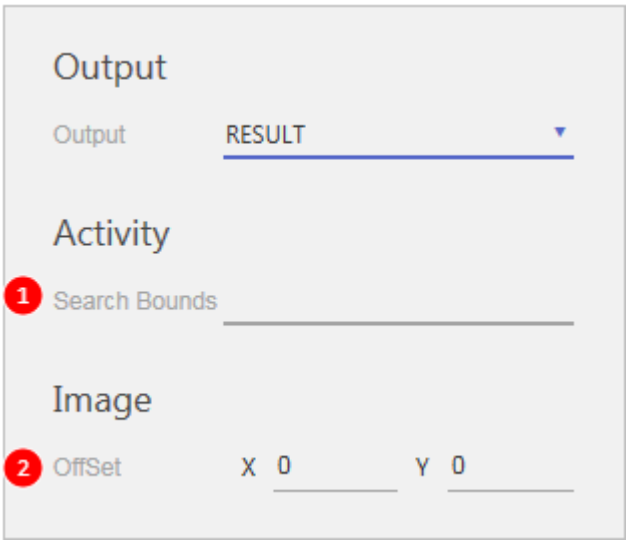
Figure 4-90 **CaptureArea** properties screen

You can select a path from the **CapturePath** field, and the captured image will be saved to this path.

GetImageColor (Getting image color)

GetImageColor gets the color from specified coordinates of an image.

Figure 4-91 **GetImageColor** properties screen



No.	Description
❶	Select the image to retrieve the color from.
❷	Enter the coordinates to retrieve the color from.

System library

You can control the system using the features provided by the **System** library.
The features provided by the System library are as follows:

- File control (CreateFile, ExecuteFile, CloseFile, CopyFile, DeleteFile, IsFileExist, GetFileList)
- Directory control (CreateDirectory, CopyDirectory, DelectDirectory, IsDirectoryExist, GetDirectoryList, GetSpecialFolder)
- Text file control (ReadTextFile, WriteTextFile)
- Display of input box and message box (InputBox, MessageBox)
- Features related to editing (GetClipboard, SetClipboard, ClipboardToFile, Copy(Ctrl+C), Cut(Ctrl+X), Paste(Ctrl+V), SelectAll(Ctrl+A), Screenshot(Alt+PrtSc))
- Program execution (ExecuteCmd)
- Calculation of elapsed time (TimeStamp)
- Features related to the registry (GetRegistryValue, SetRegistryValue)

CreateFile (Creating a file)

CreateFile creates file to a specified path.
You can create only the files of text format, such as TXT, XML, or HTML. You cannot create file formats prepared by specific apps, such as Excel or Word documents.

ExecuteFile (Executing a file)

ExecuteFile execute the specified file.
The app specified in the system is launched according to the extension of the specified file and the PID of launched app is returned.

CloseFile (Closing a file)

CloseFile closes the running file that was executed by using the PID returned from running the **ExecuteFile** library.

CopyFile (Copying a file)

CopyFile copies the specified file.

DeleteFile (Deleting a file)

DeleteFile deletes the specified file.

IsFileExist (Checking existence)

IsFileExist check whether the specified file exists.

GetFileList (Getting a file list)

GetFileList gets the file list of a specified folder.

CreateDirectory (Creating a directory)

CreateDirectory creates the specified directory.

CopyDirectory (Copying a directory)

CopyDirectory copies the specified directory.

DeleteDirectory (Deleting a directory)

DeleteDirectory deletes the specified directory.

IsDirectoryExist (Checking existence of a directory)

IsDirectoryExist checks whether the specified directory exists.

GetDirectoryList (Getting directory list)

GetDirectoryList gets the sub-directory list of the specified directory.

GetSpecialFolder (Getting user folder path)

GetSpecialFolder gets the path of user folder such as My Documents or Desktop.

ReadTextFile (Importing text file content)

ReadTextFile read the text file and imports as string.

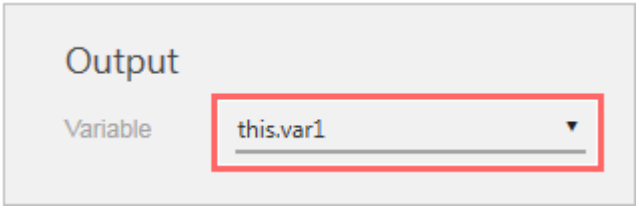
WriteTextFile (Writing text file)

WriteTextFile inputs and saves string in the text document of specified path.

InputBox (Displaying input box)

InputBox displays the input box where user can enter data on the screen.

Figure 4-92 **InputBox** properties screen



In the **Variable** field, specify the variable to save the input value.
If the user does not enter a value, the value of the specified variable is converted into a string and returned.

InputBox operates only if performed in the IPA Designer.
This can be used for debugging when writing a scenario.

MessageBox (Displaying message box)

MessageBox displays message box on the screen.

Figure 4-93 **MessageBox** properties screen

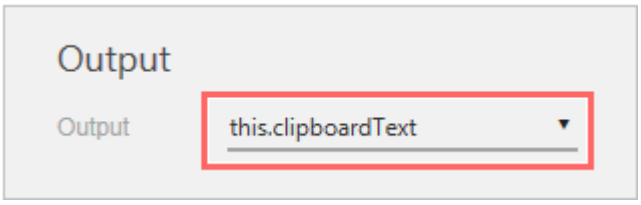
In the **Arguments** field, enter the value to display on the message box within single quotation marks.

The value output to the **MessageBox** operates only in case of performing in the IPA Designer.
This can be used for debugging when writing a scenario.

GetClipboard (Getting clipboard)

GetClipboard gets the value saved in the system clipboard.

Figure 4-94 **GetClipboard** properties screen



In the **Output** field, specify the variable to save the value read from the clipboard.

SetClipboard (Copying to clipboard)

SetClipboard saves the entered value to the system clipboard.
The saved value can be used for pasting feature because it has the same effect as Ctrl+C shortcut key on the keyboard.

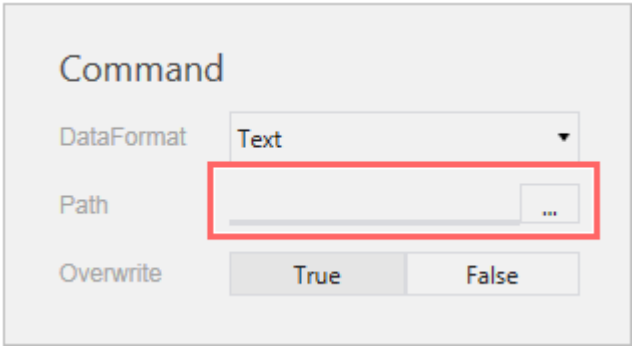
Figure 4-95 **SetClipboard** properties screen

In the **Input** field, enter value to save to the clipboard within single quotation marks.

ClipboardToFile (Writing clipboard content as file)

ClipboardToFile saves the contents of clipboard such as image or text as a file.

Figure 4-96 **ClipboardToFile** properties screen



In the **Path** field, select the path to save file.

Copy(Ctrl+C) (Copying)

Copy(Ctrl+C) generates a copy event on the screen that is currently focused on.
This has the same effect as Ctrl+C shortcut key on the keyboard.

Cut(Ctrl+X) (Cutting)

Cut(Ctrl+X) generates a cut event on the screen that is currently focused on.
This has the same effect as Ctrl+X shortcut key on the keyboard.

Paste(Ctrl+V) (Pasting)

Paste(Ctrl+V) generates a paste event on the screen that is currently focused on.
This has the same effect as Ctrl+V shortcut key on the keyboard.

SelectAll(Ctrl+A) (Selecting all)

SelectAll(Ctrl+A) generates a select all event on the screen that is currently focused on.
This has the same effect as Ctrl+A shortcut key on the keyboard.

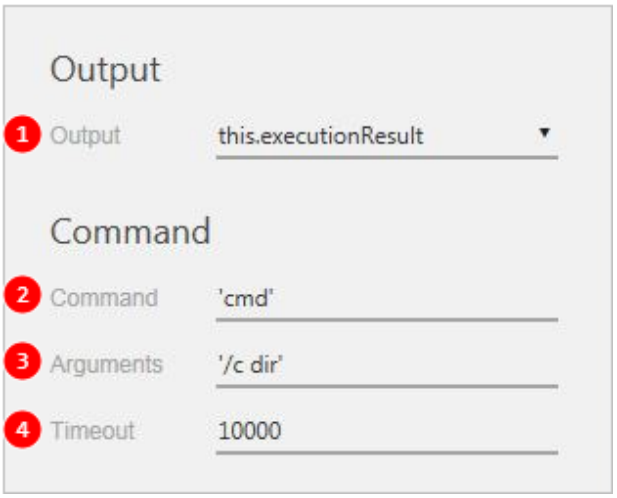
Screenshot(Alt+PrtSc) (Capturing screen)

Screenshot(Alt+PrtSc) captures the entire screen or the currently focused screen to save as a file.

ExecuteCmd (Executing program from background)

ExecuteCmd executes a program command from the background and return the execution result.
This will be executed in a state that is not visible on the screen.

Figure 4-97 **ExecuteCmd** properties screen



No.	Description
❶	Specify the variable to save the execution result of the target program. The converted value becomes the standard output string generated when carrying out a command.
❷	Enter a command to execute within single quotation marks.
❸	Enter the argument necessary for the execution command within single quotation marks.
❹	Enter the time to wait until the execution of command is complete (in milliseconds).

TimeStamp (Calculating elapsed time)

TimeStamp returns the elapsed time from the specified time to the current time in milliseconds.

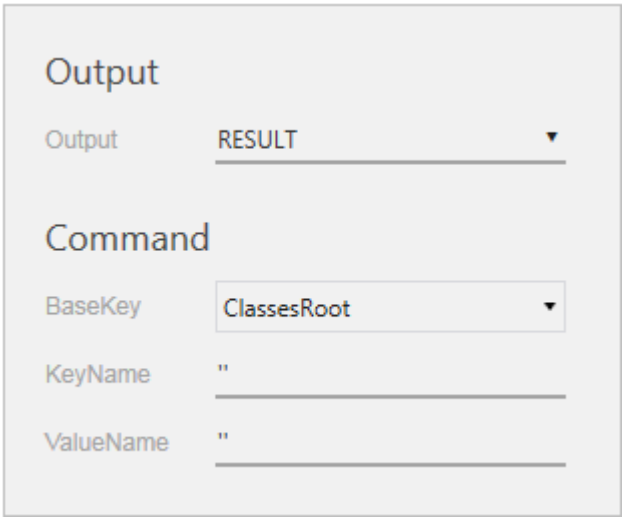
Figure 4-98 **TimeStamp** properties screen

No.	Description
❶	Set the variable to save elapsed time.
❷	Enter the start time to calculate elapsed time.

GetRegistryValue (Getting registry value)

GetRegistryValue gets the key value of the specified registry path.

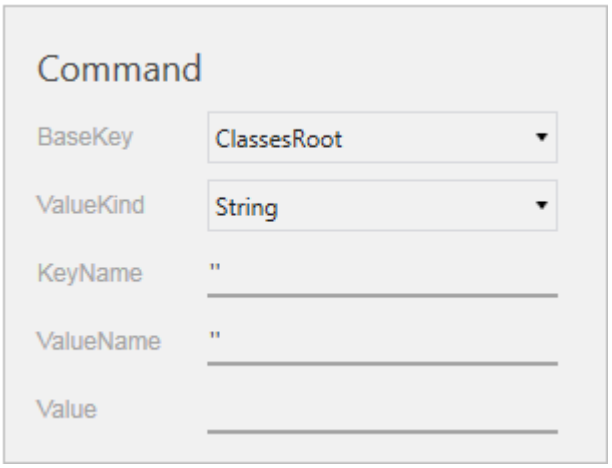
Figure 4-99 **GetRegistryValue** properties screen



SetRegistryValue (Inputting registry value)

SetRegistryValue enters key value in the specified registry path.

Figure 4-100 **SetRegistryValue** properties screen



FlowControl library

You can control process flows using the features provided by the **FlowControl** library.
You can perform repetitive tasks, set another task depending on the condition, or delay or end process flows using specific conditions.

The features provided by the FlowControl library are as follows:

- Decision-making (If)
- Repetition (For, Foreach, While)
- Assigning variables (Assign)
- Writing user scripts (ExecuteScript, ExecuteLibrary)
- Process control (Delay, TerminateFlow, TerminateAll, Break, Continue)
- Debugging and logging (Debugger, Log, DumpScreen)
- Other (RetryTask, Reboot, UploadResource, DownloadResource)

If (Decision-making)

You can create True or False branching statements that perform different actions depending on the conditions. The If/else statement of JavaScript is as follows:

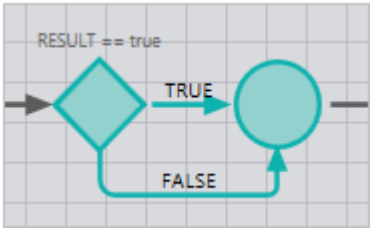
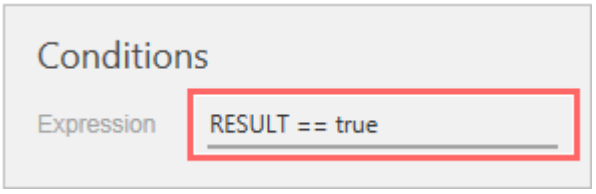


Figure 4-101 **If** properties screen



Enter conditional sentence in the **Expression** field.
You can enter many conditional sentences using operators such as && or ||.

For (Repetition)

You can create a loop. This is same as the for loop statement of JavaScript.

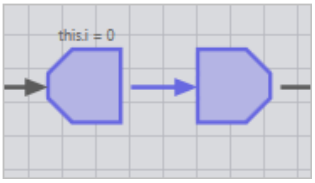
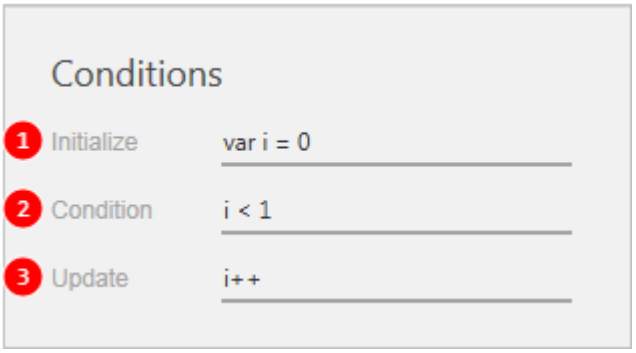


Figure 4-102 **For** properties screen



No.	Description
❶	Enter the initial value of index.
❷	Enter the condition of index.
❸	Enter the updated content of index.

Foreach (Importing list items in order)

You can create a loop to import list items in order. This is same as the foreach statement of JavaScript.

Figure 4-103 **Foreach** properties screen

No.	Description
❶	Enter variable of the list type with data.
❷	Enter the variable to allocate item one by one from the list type.
❸	Set whether to use the same purpose as the JavaScript syntax.

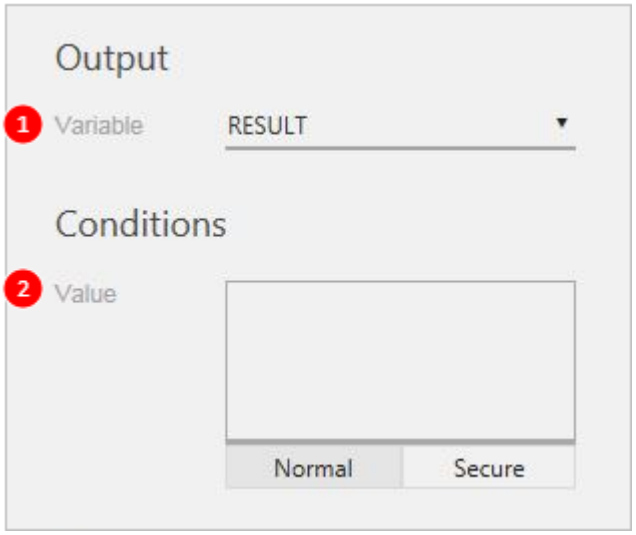
While (Performing for specific conditions)

You can create a loop to perform up to specific conditions. This is same as the while statement of JavaScript.

Assign (Assigning a variable)

You can assign a value to the variable.

Figure 4-104 **Assign** properties screen

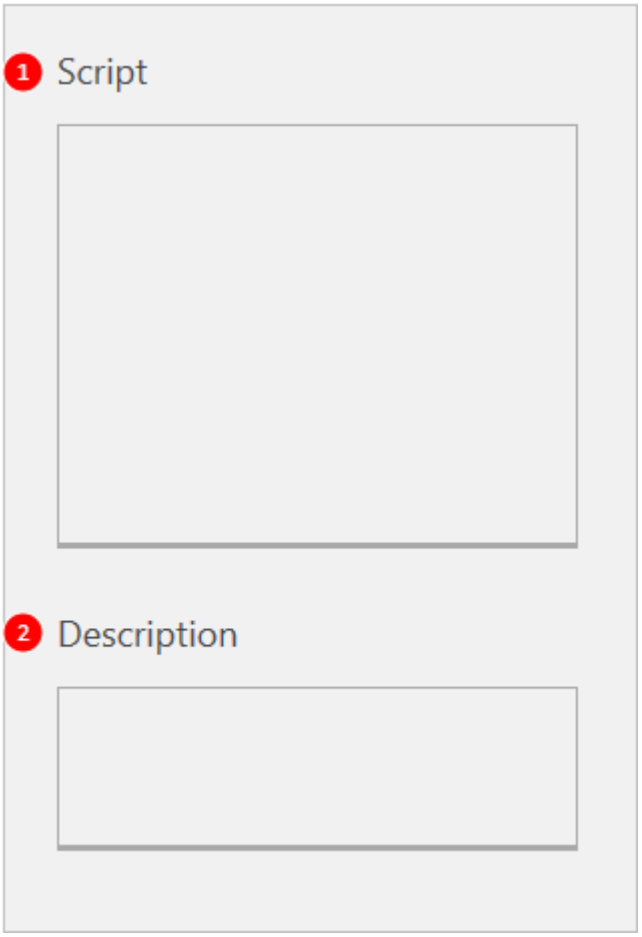


No.	Description
❶	Specify the variable to allocate value.
❷	Enter the value to allocate to the variable.

ExecuteScript (Inputting user-defined script)

You can enter a user-defined script. The script must be written using JavaScript grammar.

Figure 4-105 **ExecuteScript** properties screen



No.	Description
❶	Enter the user defined script. Double-click the text box to use ScriptEditor.
❷	Enter the description of script within single quotation marks.

ExecuteLibrary (Executing script function)

You can execute a script function that the user has written with JS file.

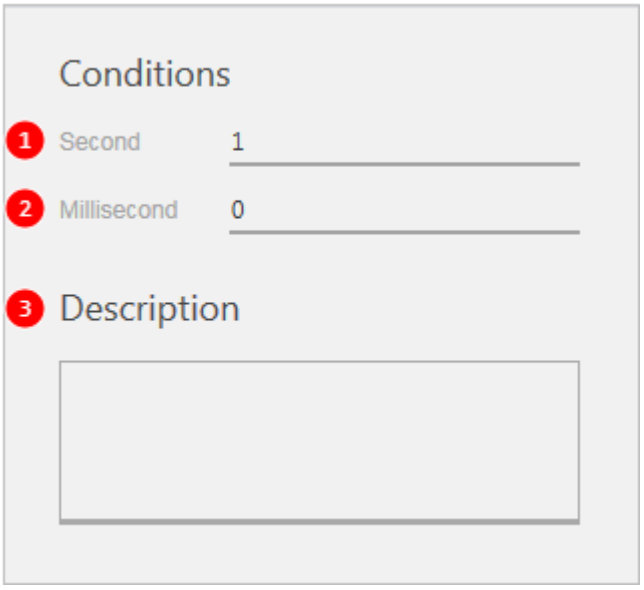
• Write the script in one of the following formats:
Script.file name = new function(){
this.function name = function (parameter list){}
}
• The default script functions are provided upon program installation. For more information, see the .js file in the Libraries folder in the program installation path.

> • Save the written script function file to the project folder of My Document after creating Libraries folder.

Delay (Delaying time)

You can delay as much as the time defined by the user.

Figure 4-106 **Delay** properties screen



No.	Description
❶	Enter time in seconds.
❷	Enter time in milliseconds.
❸	Enter description for the delay within single quotation marks.

TerminateFlow (Terminating flow)

You can terminate the flow that is operating. This is the same as the return statement in JavaScript.

TerminateAll (Terminating all)

You can terminate the processes that are operating. This is the same as the exit statement in

JavaScript.

Break (Breaking loop)

You can jump out of the loop at the time that the user has defined while in a loop. This is the same as the break statement in JavaScript.

Continue (Continuing loop)

You can move on to the next loop cycle at the time that the user has defined while in a loop. This is the same as the continue statement in JavaScript.

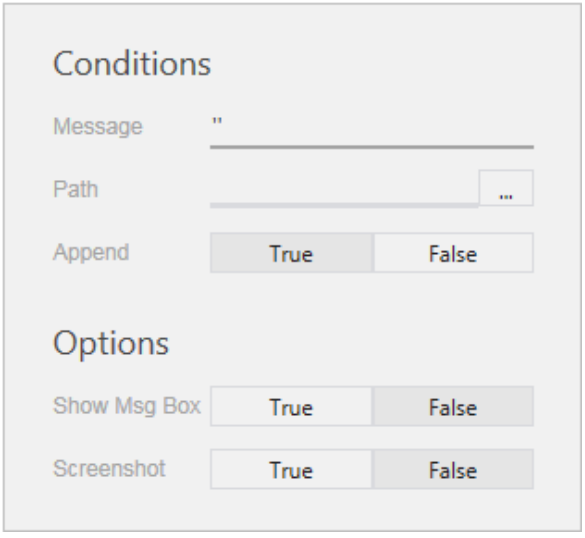
Debugger (Executing debug console)

You can execute a debug console. In the debug console, the user can input the variables and check the current progress.

Log (Outputting log)

You can output a user log to a specified file. Or it can be output in the form of a screenshot or message box.

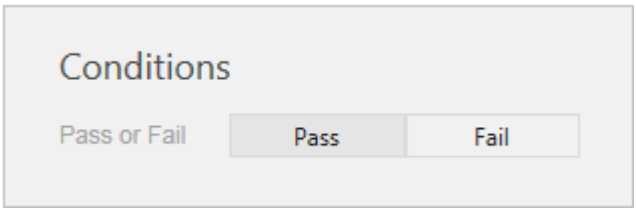
Figure 4-107 **Log** properties screen



DumpScreen (Dump screen shot)

Screen shot and Windows hierarchy information are saved under the Histories\Logs folder below the project folder. If saved as Fail, the information saved in the server is sent.

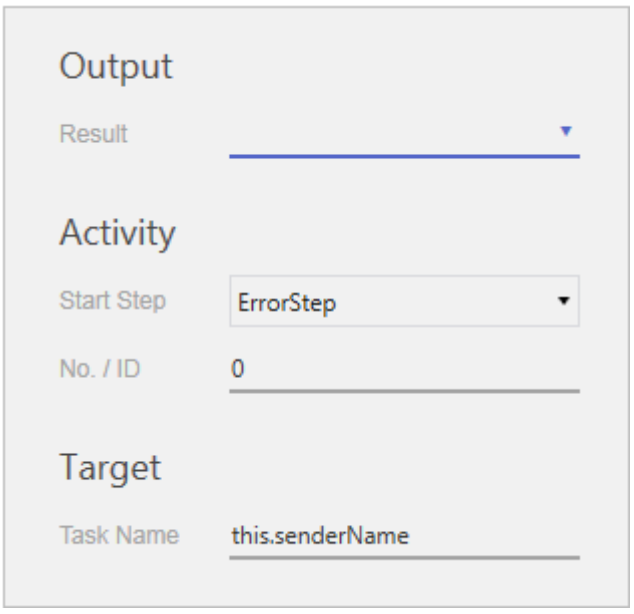
Figure 4-108 **DumpScreen** properties screen



RetryTask (Relaunching task)

When there is an error, you can restart the task currently in progress from the OnError event.

Figure 4-109 **RetryTask** properties screen



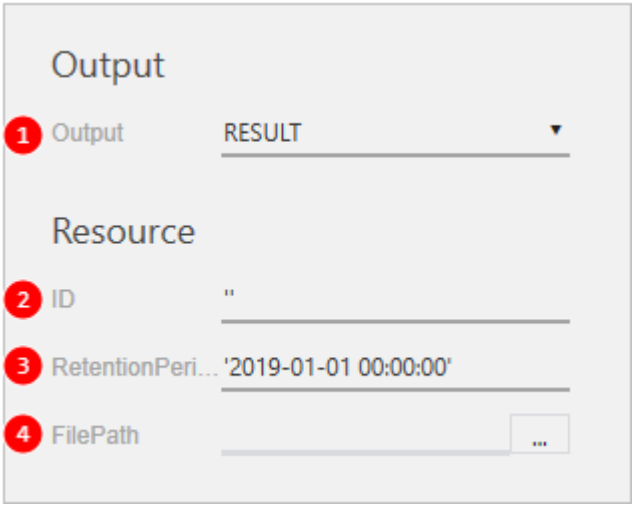
Reboot (Rebooting)

You can restart the computer.

UploadResource (Resource upload)

You can upload the resource (file) to the server. The resource ID is returned for the resource (file) uploaded to the server, and you can update or download the resource (file) again with this ID.
The resource ID can be automatically created or it can be entered by the user.
A resource (file) has a retention period and cannot be downloaded from the server after the retention period.

Figure 4-110 **UploadResource** properties screen

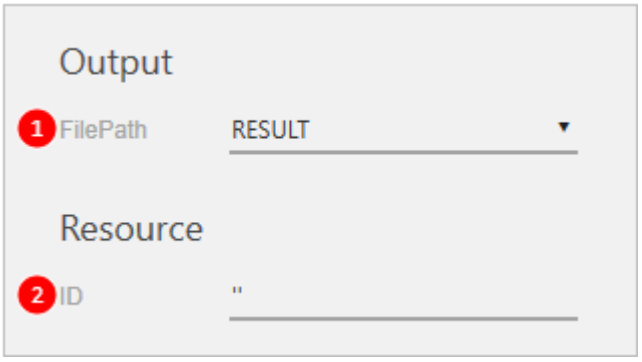


No.	Description
❶	Set to returns the ID registered in the server. If an ID was set, then the same ID will be received. If not, an ID created from the server will be returned.
❷	Enter the ID to be saved. (Overwrite)
❸	Enter the period to retain the file in the server. (until...)
❹	Enter the path of the file to be uploaded to the server.

DownloadResource (Resource download)

You can download resources (files) from the server. The resource ID of the resource to download must be entered to download.
The downloaded resource (file) is saved to the SharedResources folder inside the project folder. Once the download is complete, the file path is returned.

Figure 4-111 **DownloadResource** properties screen



No.	Description
❶	Enter the path of the downloaded file.
❷	Enter the ID of the file to download.

Knox Portal Library

The features provided by the Knox Portal Library can be used to use the link features of the Knox Portal REST API.
To use this library, the designer must be logged in.

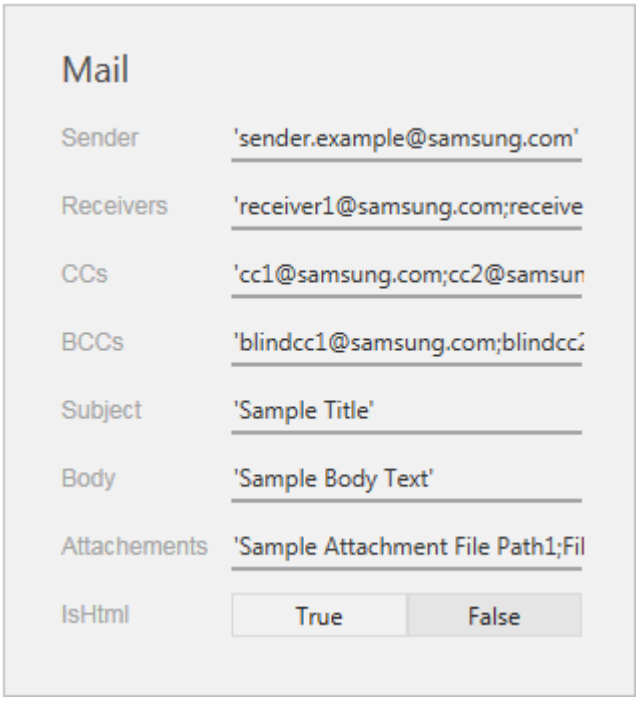
The features provided by the Knox Portal library are as follows:

- Sending email (SendMail)
- Submitting approvals and viewing approval statuses (SubmitApproval, GetApprovalStatus)
- Getting user information and department information (GetUserInfo, GetDeptInfo)
- PIMS features (CreateSchedule, DeleteSchedule, GetSchedules)

SendMail (Sending email)

The Knox Portal API is used to send emails.
Up to 200 recipients (including CCs and BCCs), 5 attachments, and a total of 10 MB are supported.

Figure 4-112 **SendMail** properties screen



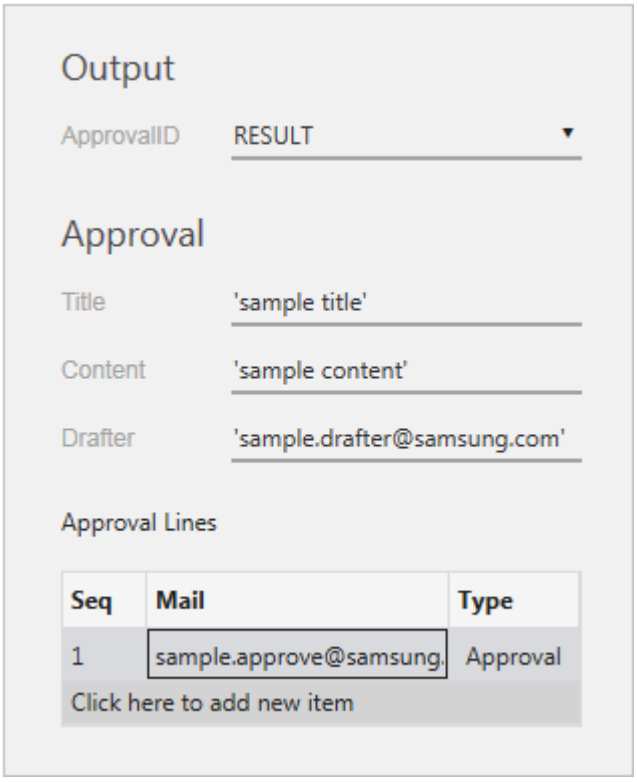
SubmitApproval (Submitting approval)

Submit an approval using Knox Portal API.

The following values can be selected when setting up the approval line. The order of the approval list must be identical for parallel consent and parallel approval.

- Approval
- Consent
- PostApproval
- ParallelConsent
- ParallelApproval
- Notification

Figure 4-113 **SubmitApproval** properties screen



GetApprovalStatus (Get approval status)

Get the approval status using Knox Portal API.

The current approval status is returned from the values below:

- FailedEncrypt (Encryption has failed)
- Encrypt (Encryption is being performed)
- Preliminary (Schedule has been submitted)
- Postponed (Approval is postponed)
- InProgress (Approval is in progress)
- Approved (Approval is complete and approved)
- Rejected (Approval is rejected)
- Canceled (Submission is canceled)
- PreApproved (Pre-approved status)
- PostApproved (Post-approved status)

Figure 4-114 **GetApprovalStatus** properties screen

GetUserInfo (Get user information)

Get employee information using KnoxPortal ID or email address.

The returned type is UserInfo and the members of the UserInfo class are as shown below.

- UserId (KnoxPortal ID)
- Name (Employee name)
- EngName (English name of employee)
- Department (Department name)
- EngDepartment (English name of department)
- DepartmentCode (Department code)
- EmployeeNo (Employee number)
- EngTitle (English title name)
- Title (Title name)
- Mail (Email address)
- Mobile (Mobile phone number)
- Telephone (Phone number)

Figure 4-115 **GetUserInfo** properties screen

GetDeptInfo (Get department information)

Get department information of an employee using KnoxPortal ID or email address.

The returned type is DeptInfo and the members of the DeptInfo class are as shown below.

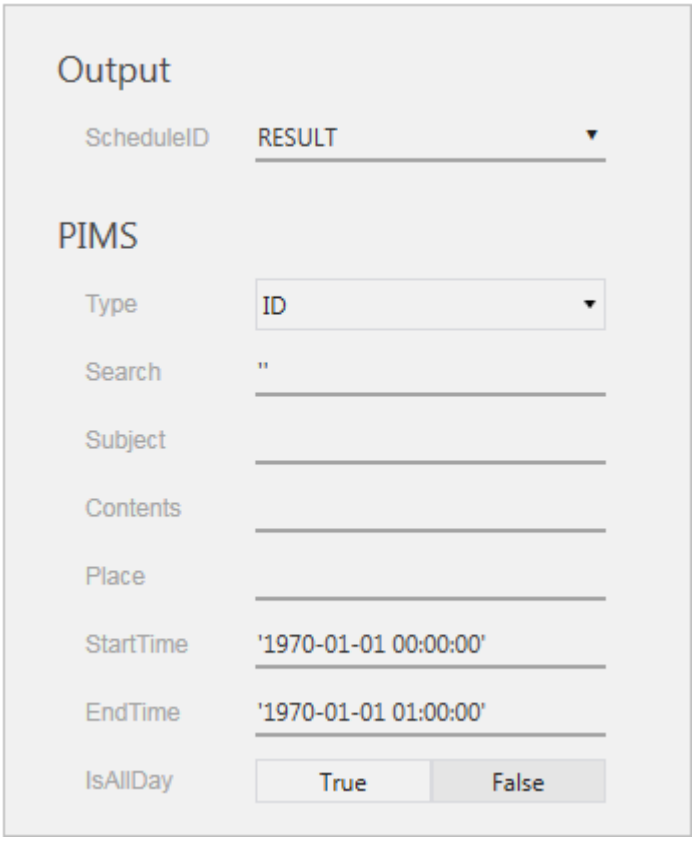
- DeptName (Department name)
- EngDeptName (English name of department)
- DeptCode (Department code)
- HighDeptName (Name of upper department)
- HighDeptName (English name of upper department)
- HighDeptCode (Upper department code)
- Company (Company name)
- EngCompany (English name of company)
- ManagerId (Knox ID of department head)
- ManagerName (name of department head)

Figure 4-116 **GetDeptInfo** properties screen

CreateSchedule (Create schedule)

Create the KnoxPortal PIMS schedule.
The schedule ID is returned if the schedule is created successfully.

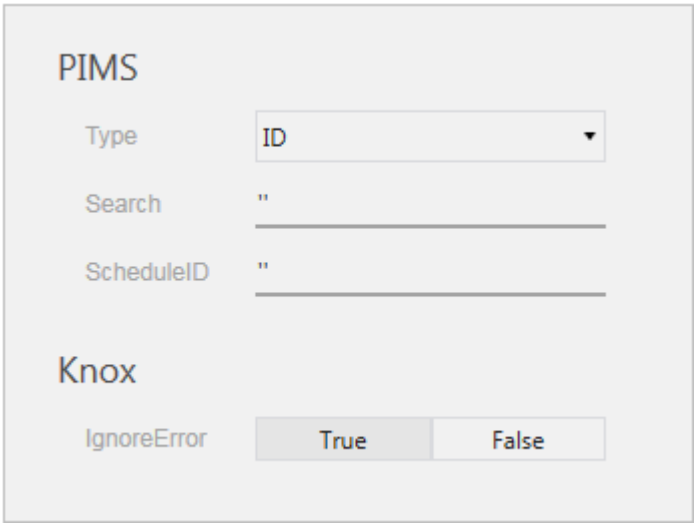
Figure 4-117 **CreateSchedule** properties screen



DeleteSchedule (Delete schedule)

Delete the KnoxPortal PIMS schedule.
The schedule ID to delete and user ID are required.

Figure 4-118 **DeleteSchedule** properties screen



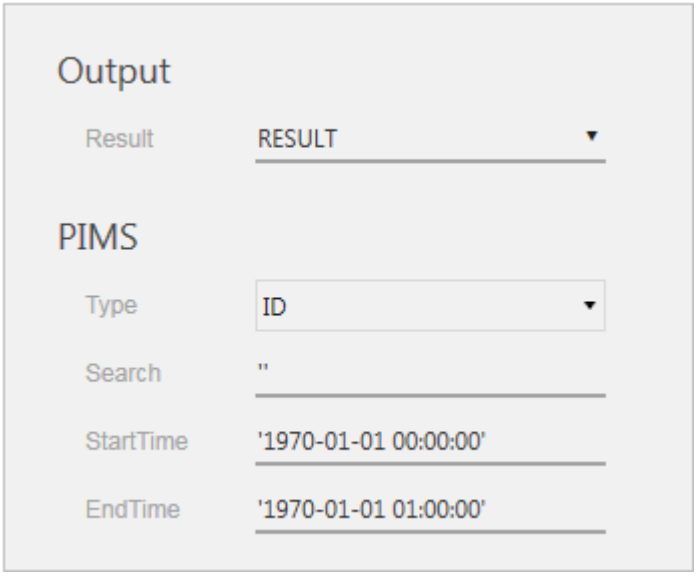
GetSchedulesInfo (Get schedule)

Get the KnoxPortal PIMS schedule.

The returned type is ScheduleInfo and the members of the ScheduleInfo class are as shown below.

- ScheduleId (Schedule ID)
- Subject (Title of schedule)
- Contents (Contents of schedule)
- StartTime (Starting time of schedule)
- EndTime (Ending time schedule)
- AllDayYn (All day schedule)
- Place (Place of schedule)
- Attendees (Attendees of schedule)

Figure 4-119 **GetSchedulesInfo** properties screen



Private Script Library

In case of designing a project by writing a script, you can use the bot objects and the default script library.

ToString (Object)

ToString returns information of value type variable, array, and two dimensional arrays as a string.

GetObjectInfo (Object)

GetObjectInfo returns the member data information of object type variables.

GetCount (Object)

GetCount returns the data count of array type objects.

Print (Message), Debug(), Info(), Warn(), Error()

Print outputs logs for each log severity.

CharToInt (String)

CharToInt returns the code value corresponding to the character.

IntToChar (Number)

IntToChar returns the character corresponding to the code value.

AddHostType (Name, TypeName)

AddHostType registers the class provided by C# to use in script.

CreateObject (TypeName)

CreateObject creates a COM object.

5.
Designing process flows

A process flow is a schematic diagram that shows the sequence and flow of handling many processes. The method of designing a process flow is similar to a project, with some differences. This chapter focuses on the differences in the design.

"Designing process flow" explains the following topics:

- Designing with processes
- Designing with the library

5.1 Designing with processes

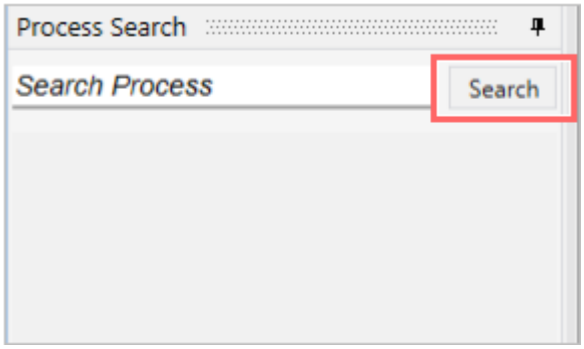
A process flow allows you to design by combining a number of processes published to the server.

Adding a process

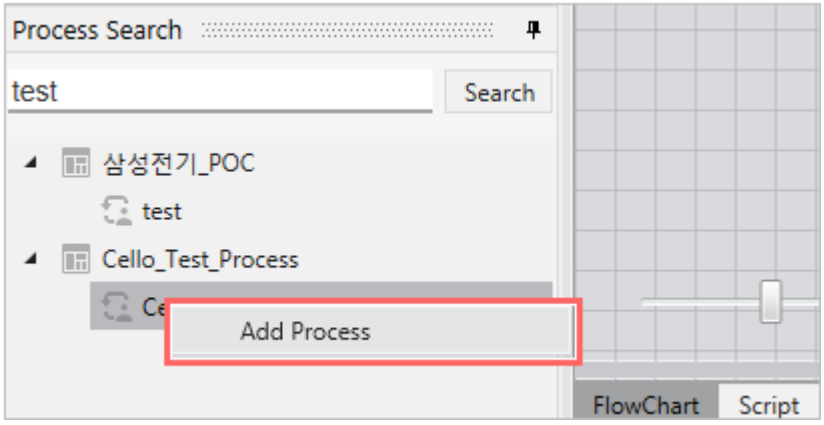
To design a process flow by searching for processes published on the server, complete the following steps below.

To search for a process, you must log in to the server first.

1. In the search bar of the process search screen, enter a search phrase, and click **Search**.The process containing the entered phrase and the project containing this process will appear.



2. Right-click on the process to add and click **Add Process**.



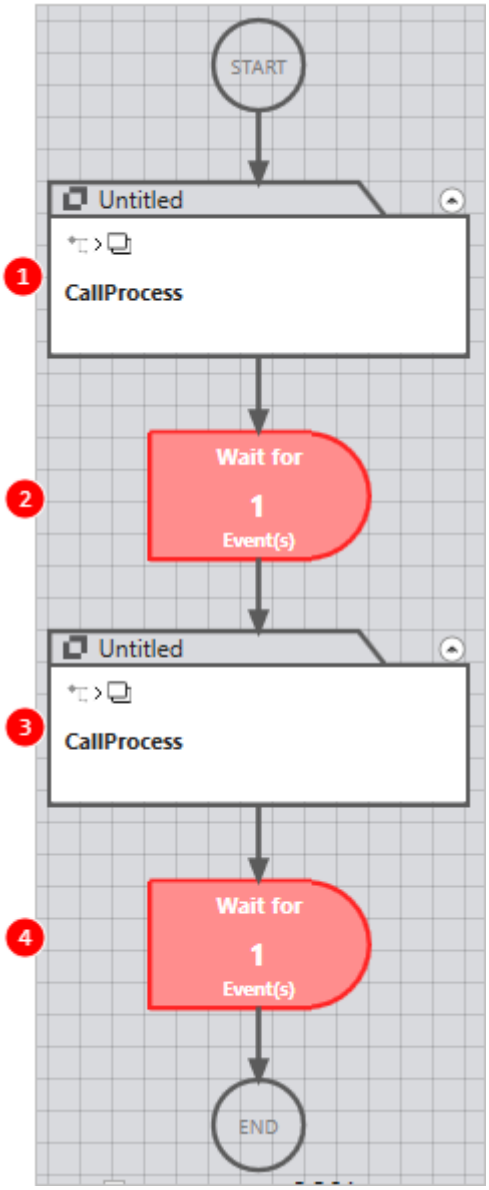
3. On the process flow library screen, search for **CallProcess**, and drag to add to the flowchart.
4. Enter the information of the process to call from the Properties screen.

The process will be added.

- Only the information of the process imported to the process flow will be searched.
- If there is input value or output value in the process, enter it in the Properties screen.

Adding waiting action

A process flow is an asynchronous method that is not performed in order. If a process must be performed in order, an action of waiting until the previous process ends must be applied.
For example, if a mail writing process must be performed after completing a login process, add **WaitForMultipleEvent** library between these two processes.



No.	Description
❶	Logging in
❷	Waiting for the completion of the "Logging in" process
❸	Writing email
❹	Waiting for the completion of the "Writing email" process

5.2 Designing with the library

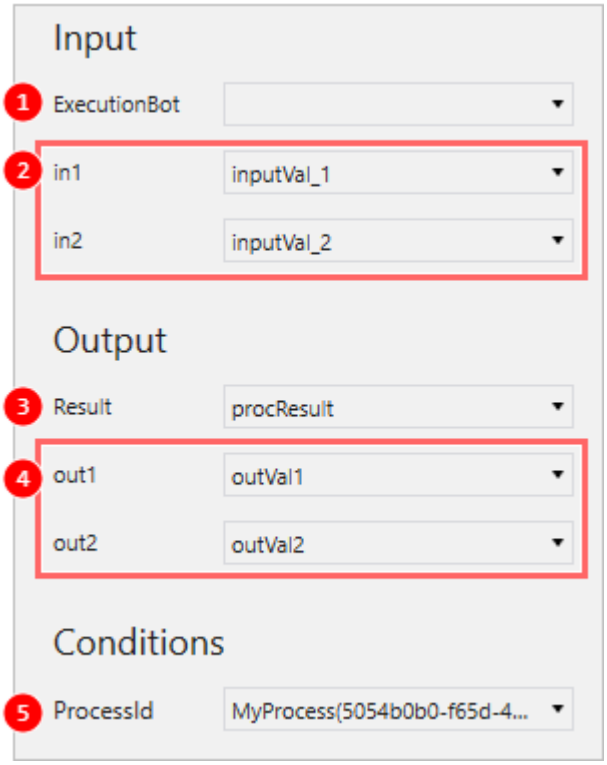
You can design a process flow using the library.
For information on how to add and edit activities using library, see "4.2 Designing with the library"

Learn more about the process flow library

CallProcess (Performing specific process)

CallProcess performs processes specified by the user. The process must be published in the server. If there is an input value to send or an output value to receive, this must be specified at the same time.

Figure 5-1 **CallProcess** properties screen



No.	Description
❶	Use when performing a process in the same bot as the process performed before. To use this feature, select the result variable of the process performed before or leave it blank if you do not want to use it.
❷	Specify the variable to send as an input value of the process to perform.
❸	Specify the variable to save information on the performed result of the process.
❹	Specify the variable to receive the output value from the process to perform.

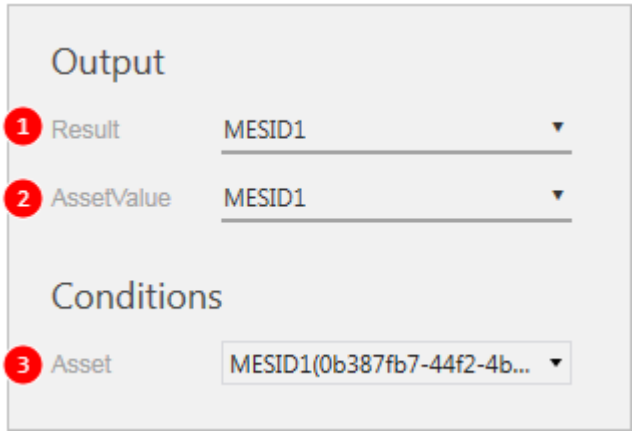
No.	Description
❺	Select ID of the process to perform. Only the process added through search appears on the list. Input and output values may change, depending on the selected process.

RequestAsset (Requesting asset)

Use in the process flow by requesting an asset in the server. For example, if a notice title template string for public use is registered as asset, you can receive the value through **RequestAsset**.

RequestAsset is used alone without having to add **WaitForMultipleEvents**.

Figure 5-2 **RequestAsset** properties screen



No.	Description
❶	Select the variable to save the requested asset object.
❷	Select the variable to save the requested asset object value. This is necessary when trying to use the asset object value which is not asset object from another library.
❸	Select the asset to request. Only the items added as public asset will appear on the list.

WaitForMultipleEvent (Waiting to perform an event)

In case of trying to perform the next action after one action is complete in a process flow performed as asynchronous method, you must use WaitForMultipleEvents. You can wait to complete one or more actions. An instance where at least one action gets completed and an instance where all actions get completed as options.

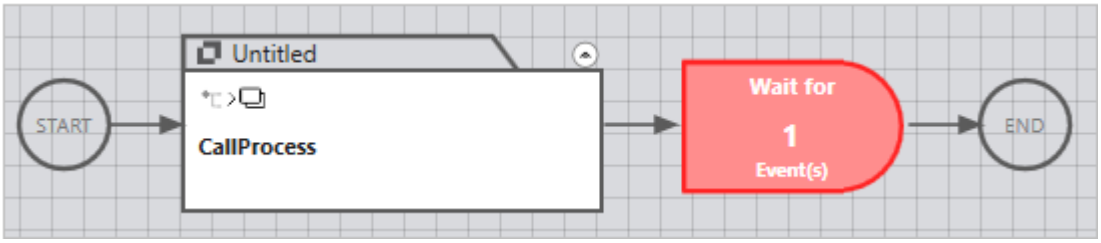
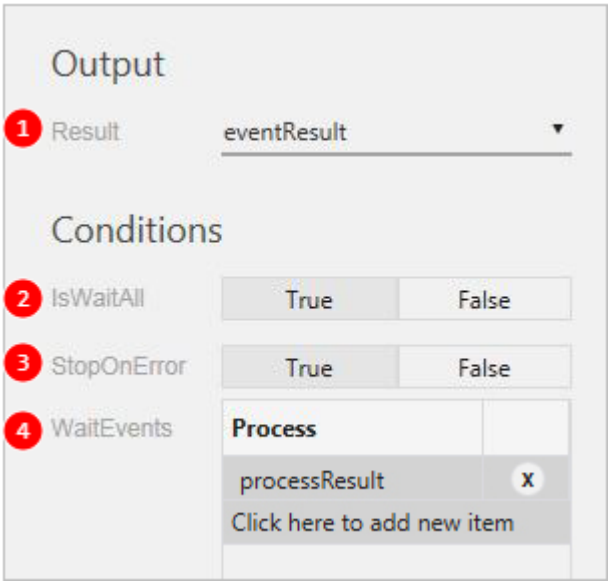


Figure 5-3 **WaitForMultipleEvents** properties screen

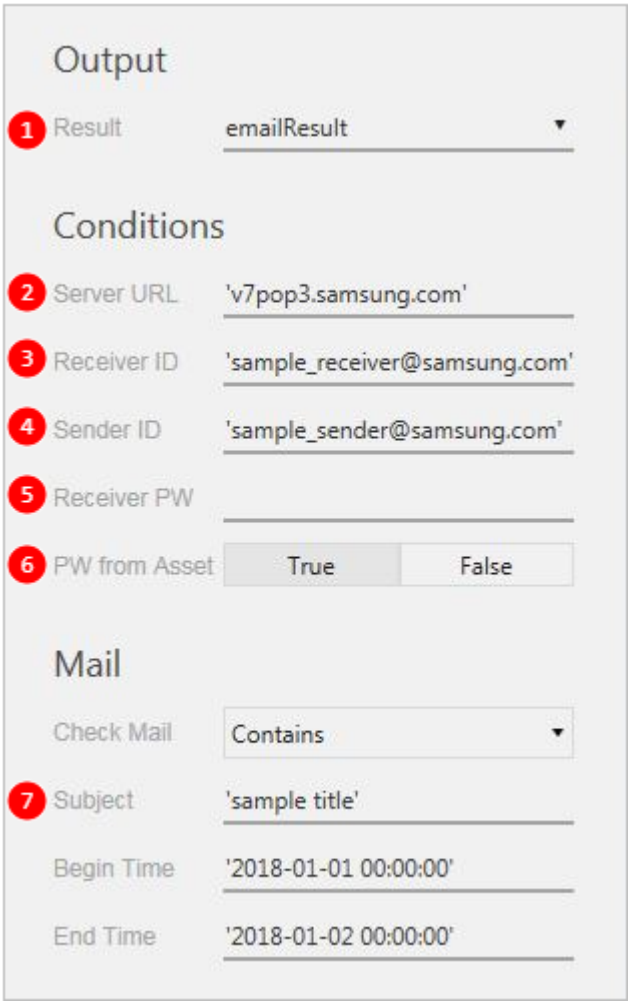


No.	Description
❶	Select the variable to save result information.
❷	Select whether to wait until all events are generated, or wait until at least one event is generated. For example, if set to False when waiting to perform five processes, proceed to the next process when one of the five processes is complete.
❸	Select whether to stop the process flow if any one of the waiting events fails.
❹	Select the event to wait for. Select the variable where the performed result of the event is saved. Click **Click here to add new item** to add the event to wait for. Click x to delete.

WaitEmail (Waiting to receive mail)

WaitEmail waits for the mail containing the information specified by the user and then performs a specific action when the mail arrives. You must enter the string information to include in sender and recipient account information and title.

Figure 5-4 **WaitEmail** properties screen

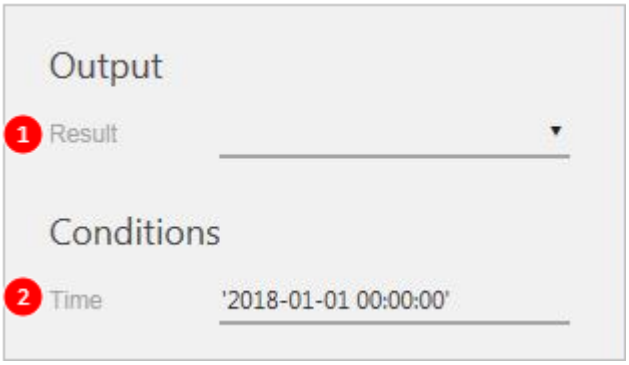


No.	Description
❶	Select the variable to save the result of waiting to receive mail.
❷	Enter the address of mail server within single quotation marks.
❸	Enter ID of the account to receive mail within single quotation marks.
❹	Enter ID of the account to send mail within single quotation marks.
❺	Enter password of the account to receive mail.
❻	Select the method to enter password of the account to receive mail.
❼	Enter the contents to include in the mail title within single quotation marks.

AssignDateTime (Setting the date and time information)

Assign the desired date and time information to the variable.

Figure 5-5 **AssignDateTime** properties screen

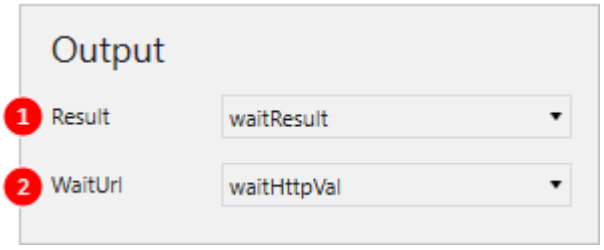


No.	Description
❶	Enter the variable to save time information.
❷	Enter time information to set.

WaitHttp (Waiting for URL connection)

WaitHttp waits for the temporarily created URL to be called. It is mainly performed for the account authentication of the user. Wait until the user connects to the URL sent via mail, messenger or SMS, etc.

Figure 5-6 **WaitHttp** properties screen

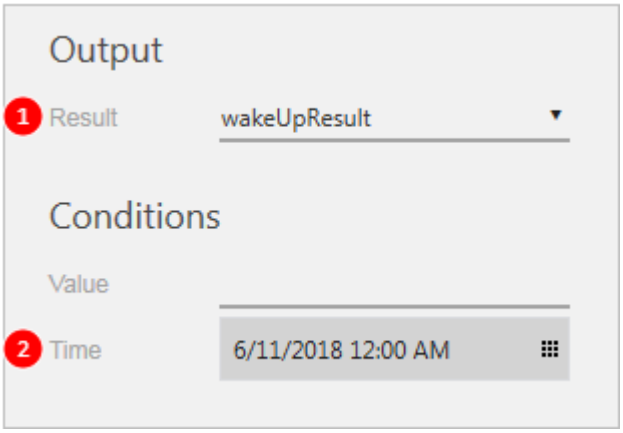


No.	Description
❶	Select the variable to save result object.
❷	Select the variable to save the value of result object. This is necessary to easily use from another library after saving the received URL value.

WakeUp (Waiting Until Specific Time)

Wait until the specific time specified by the user. For example, the user adds **WakeUp** before the **CallProcess** library and specifies time if a specific process is desired to be performed at 3 a.m. If the server requests to perform the process flow, the process is performed at 3 a.m. after waiting in this library stage.

Figure 5-7 **WakeUp** properties screen



No.	Description
❶	Select the variable to save result object.
❷	Set the time to perform action. The next action will be performed after waiting until it becomes the set time. You can select date and time after clicking ⁞⁞, or enter directly in the input field.

6.
Linking to server

You can publish a project or process flow you have worked on by connecting IPA Designer to the server. You can also edit or use by downloading a project or process flow published in the server.

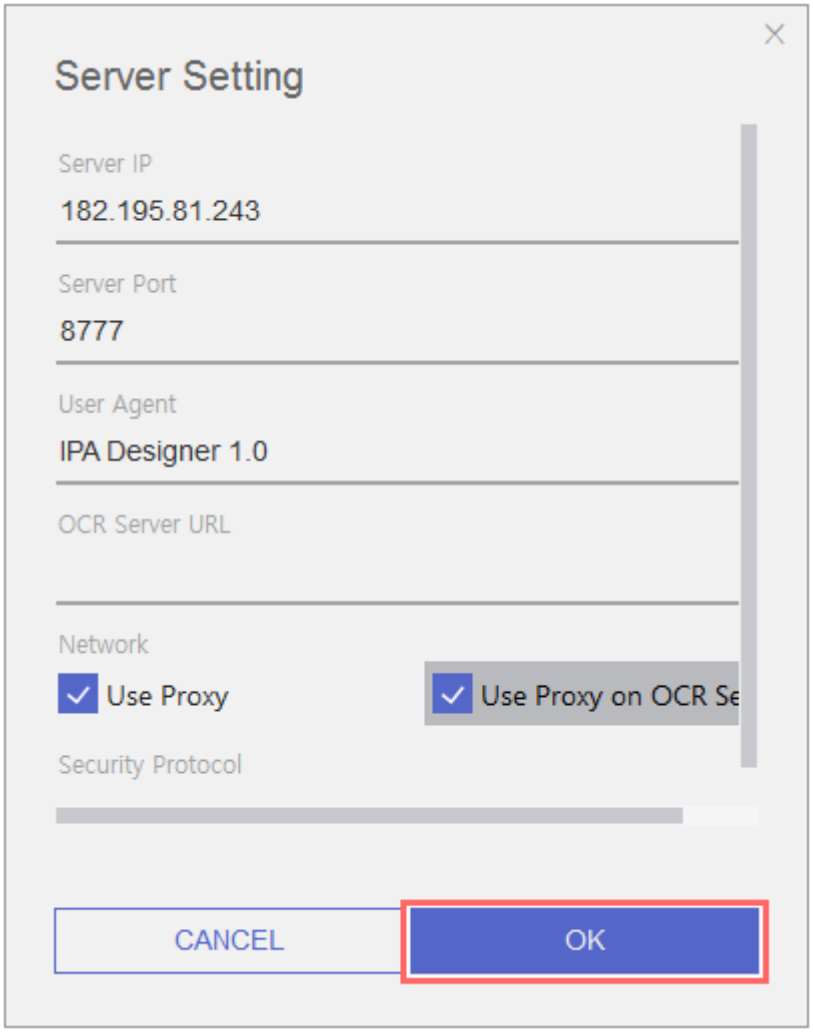
"Linking to server" explains the following topics:

- Preparing server linkage
- Activating and logging in
- Publishing project and process flow
- Importing project and process flow
- Using public resources

6.1 Preparing server linkage

To connect the server, you need a designer ID and a license key.

1. Generate a designer ID and a license key.
2. Create an account.
3. Click ☰ > **Settings** > **Server Setting**.
4. Enter **Server IP**/**Server Port** and click **OK**.



The server address will be set up.

6.2 Activating and logging in

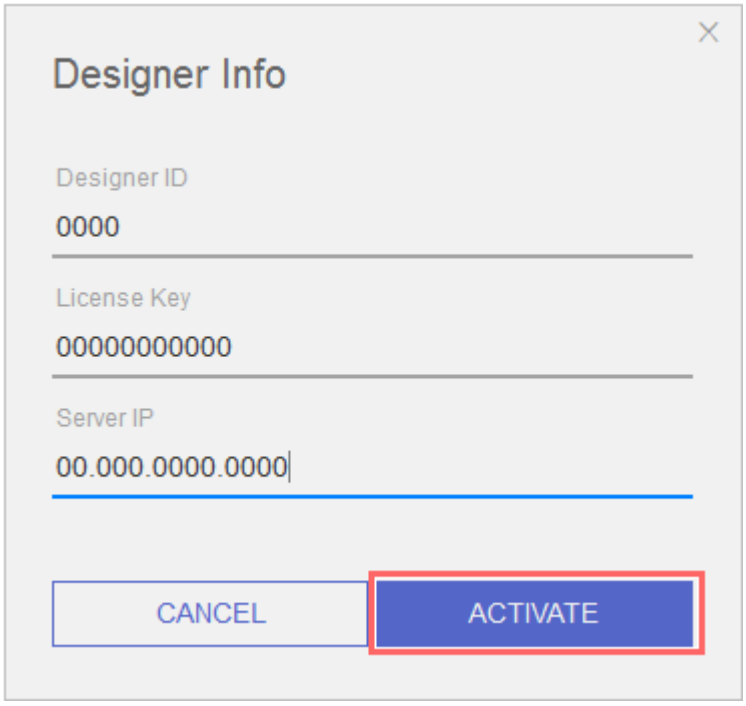
To connect the server, you must activate IPA Designer and log in.
A designer ID and a license key issued from the server are required.

Activating

If you are connecting to the server for the first time or have changed the designer ID/license key, activate the IPA Designer first.

1. Click ☰ > **Help** > **License**.
2. Enter the license key on the "Designer Info" pop-up and click **ACTIVATE**.

   The Designer ID will be entered automatically when the activation is complete.

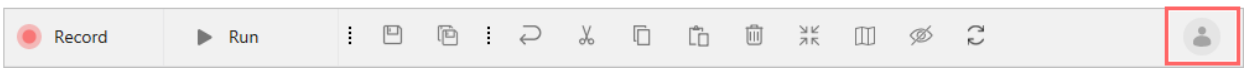


IPA Designer will be activated.

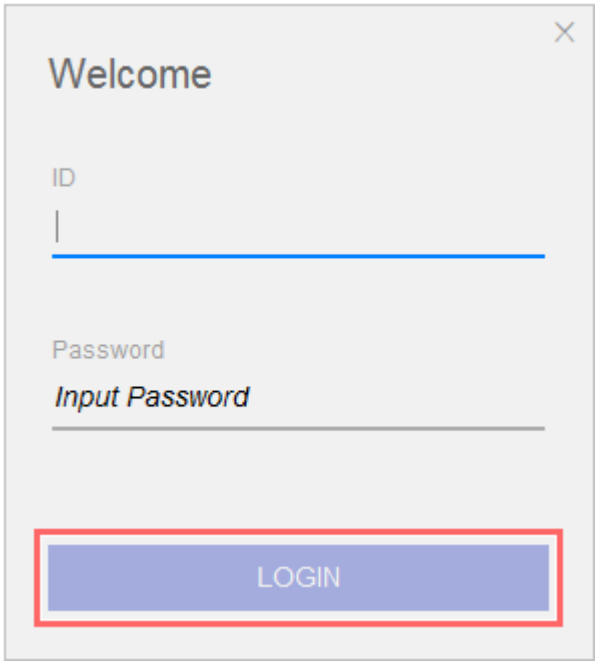
Logging in

Log in using the account issued from the server.

1. Click on the IPA Designer screen.



2. Enter ID/password generated from the server and click **LOGIN**.

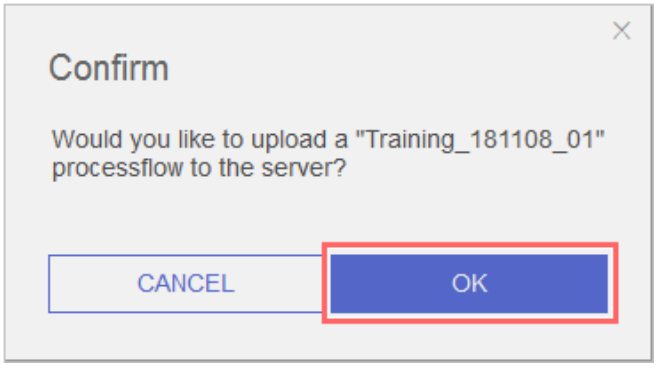


The login will be completed.

6.3 Publishing project and process flow

You can publish a project or process flow you are working on in IPA Designer to the server. The published project or process flow can be executed using a bot after activating from the server.

1. Click ☰ > **Publish**.
2. Click **OK** on the "Confirm" pop-up.

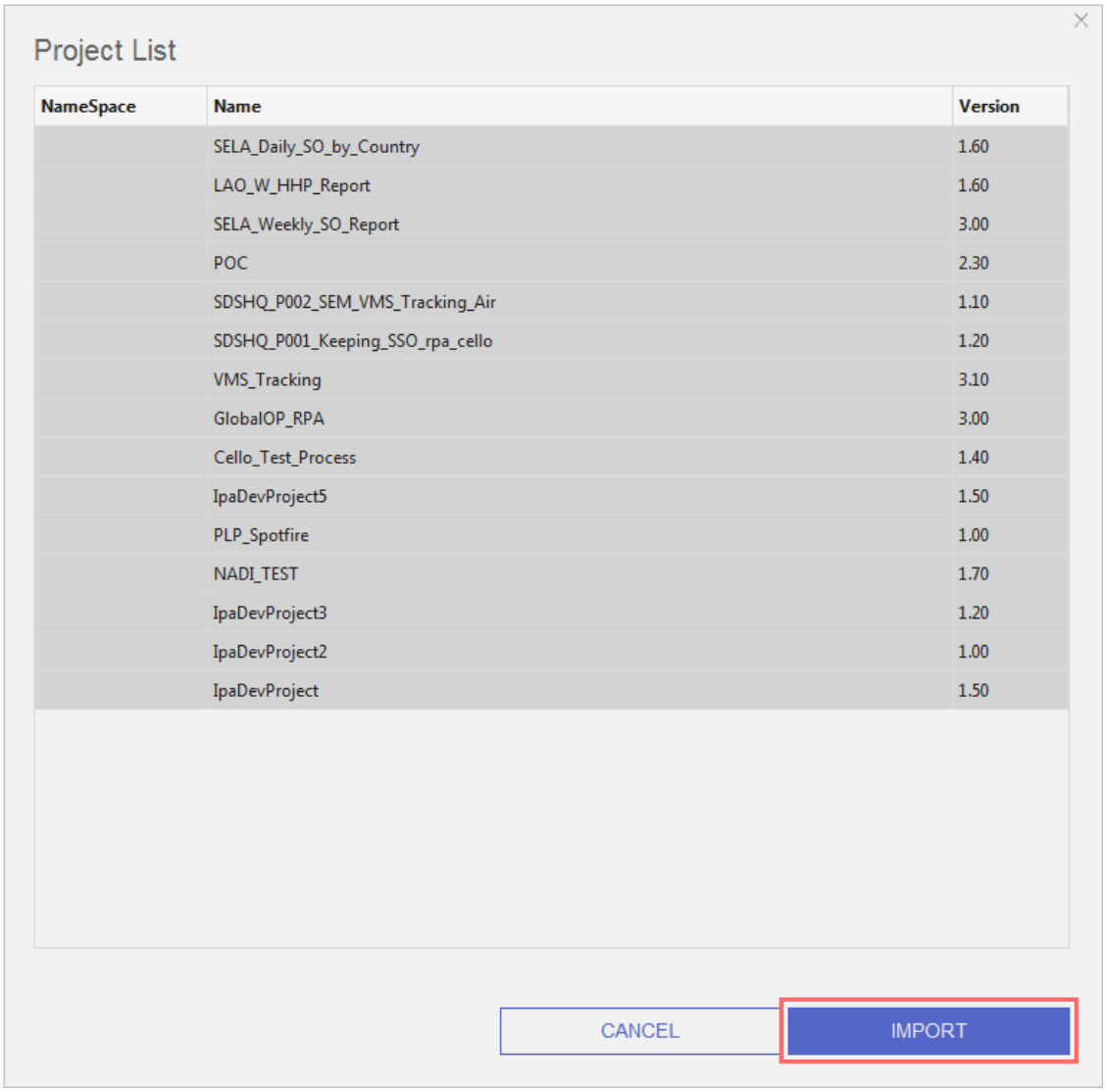


A project or process flow will be published to the server.

6.4 Importing project and process flow

You can import a project or process flow published in the server. Edit or execute the imported project or process flow in the local IPA Designer.

1. Click ☰ > **Import**.
2. Select **Project** or **ProcessFlow** and click **IMPORT**.
3. Select the imported project or process flow and click **IMPORT**.

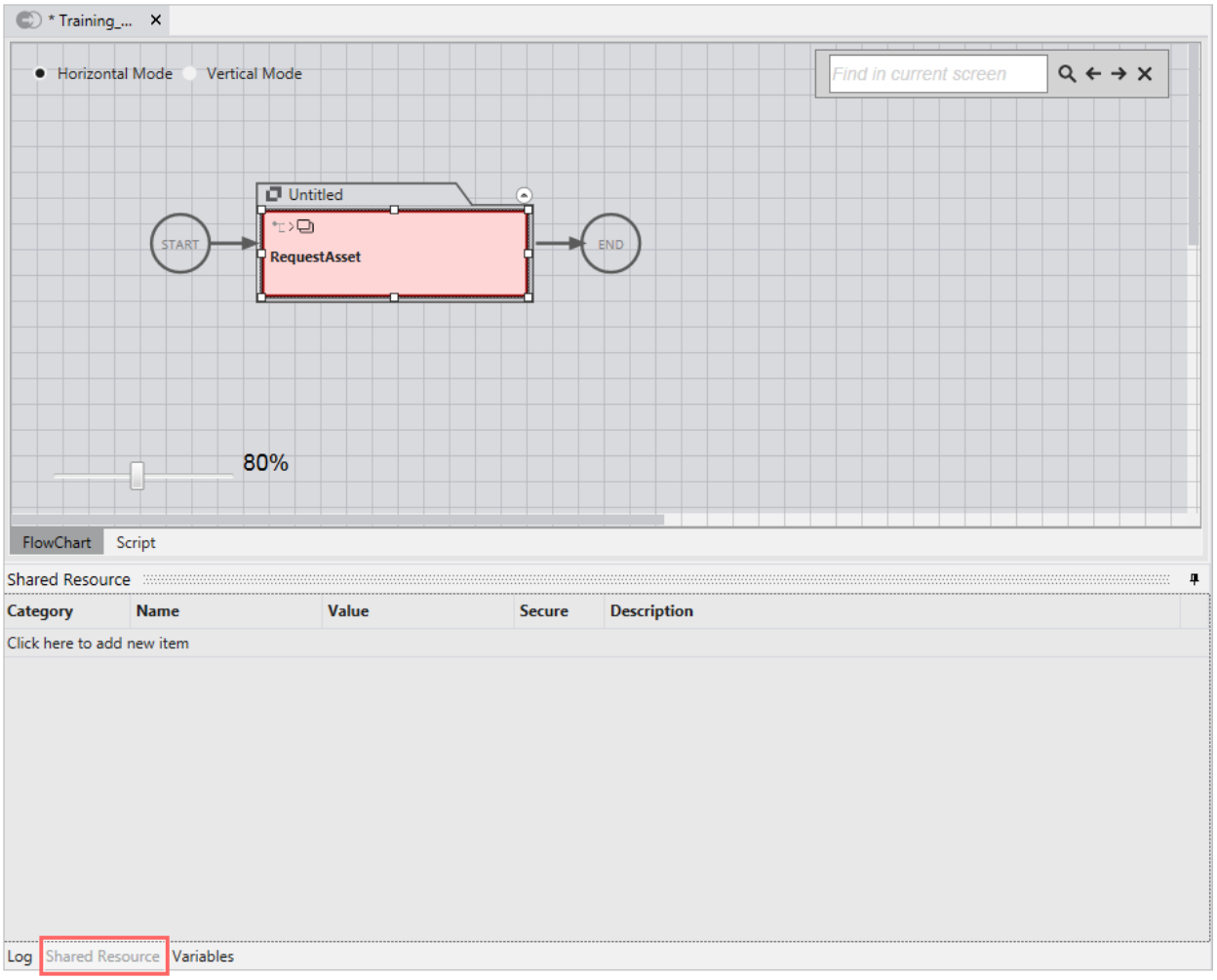


The imported project or process flow will be added.

6.5 Using shared resources

You can check the shared resources registered in the server and use by importing to the IPA Designer if necessary. Use the resources added to a project in the properties window of library.

1. Click **Shared Resource** to move to the shared resource screen.



2. Click **Click here to add new item**. The shared resource list added to the server will appear.
3. Select the resource to import from the shared resource list registered in the server and click **IMPORT**.

The resource will be added to the current project.

7.
Using IPA Bot

You can build a work automation environment using IPA Bot.

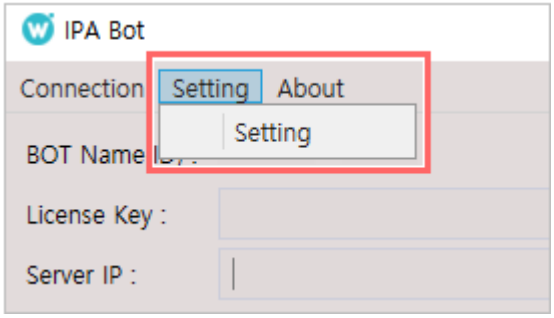
"Using IPA Bot" explains the following topics:

- Setting up use environment
- Running the bot

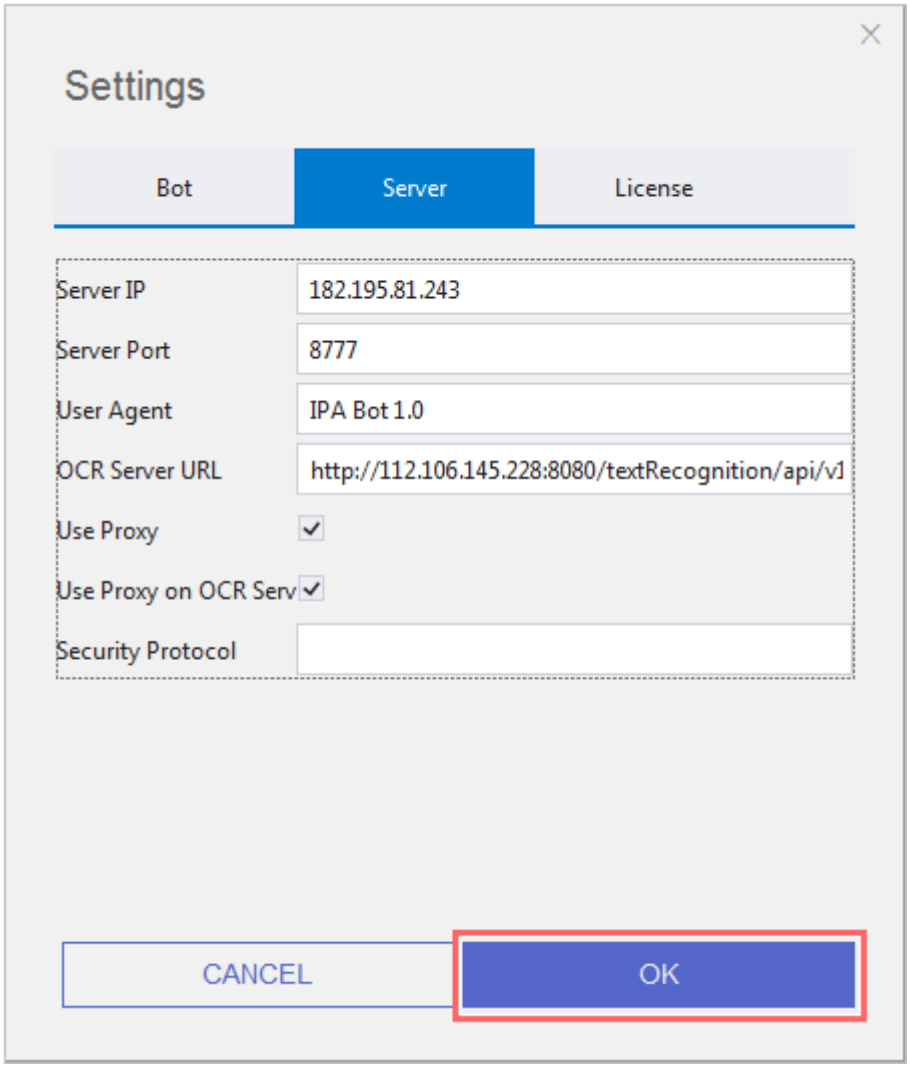
7.1 Setting up use environment

To use IPA Bot, you must generate a Bot ID and a license key from the server.

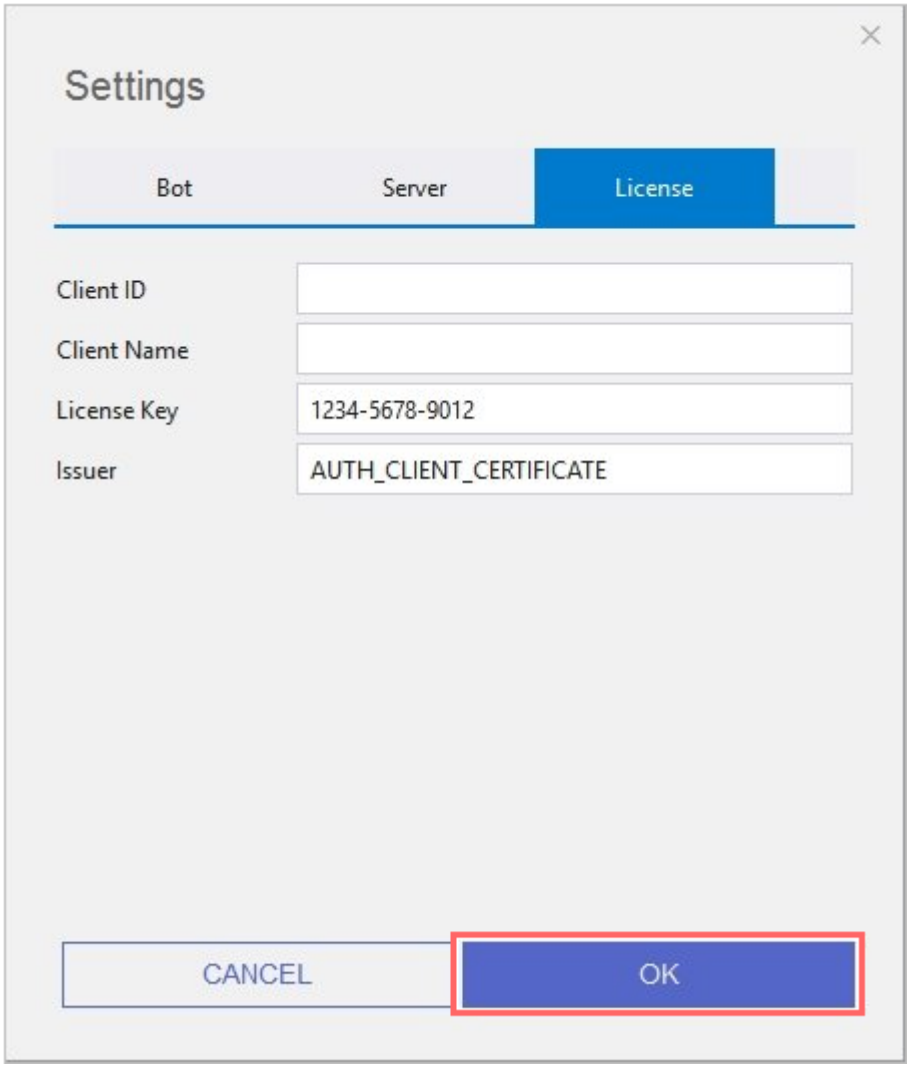
1. Click **Setting** > **Setting**.



2. Enter the server IP and server port in the **Server** tab and click **OK**.



3. Enter the license key from the **License** tab and click **OK**.

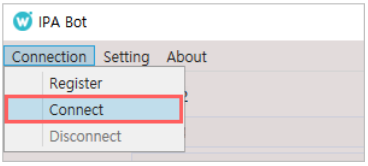


Setting up the use environment is complete.

7.2 Running the bot

You can run the bot after setting up a server IP, client ID and license key.

Click **Connection** > **Connect**.



The bot is started.

Appendix A.
Troubleshooting

This chapter describes the problems are often arise and troubleshooting.
"Troubleshooting," explains the following topics:

- Problems related to installation and registration
- Problems related to writing and executing automation scenario
- Problems related to linking servers

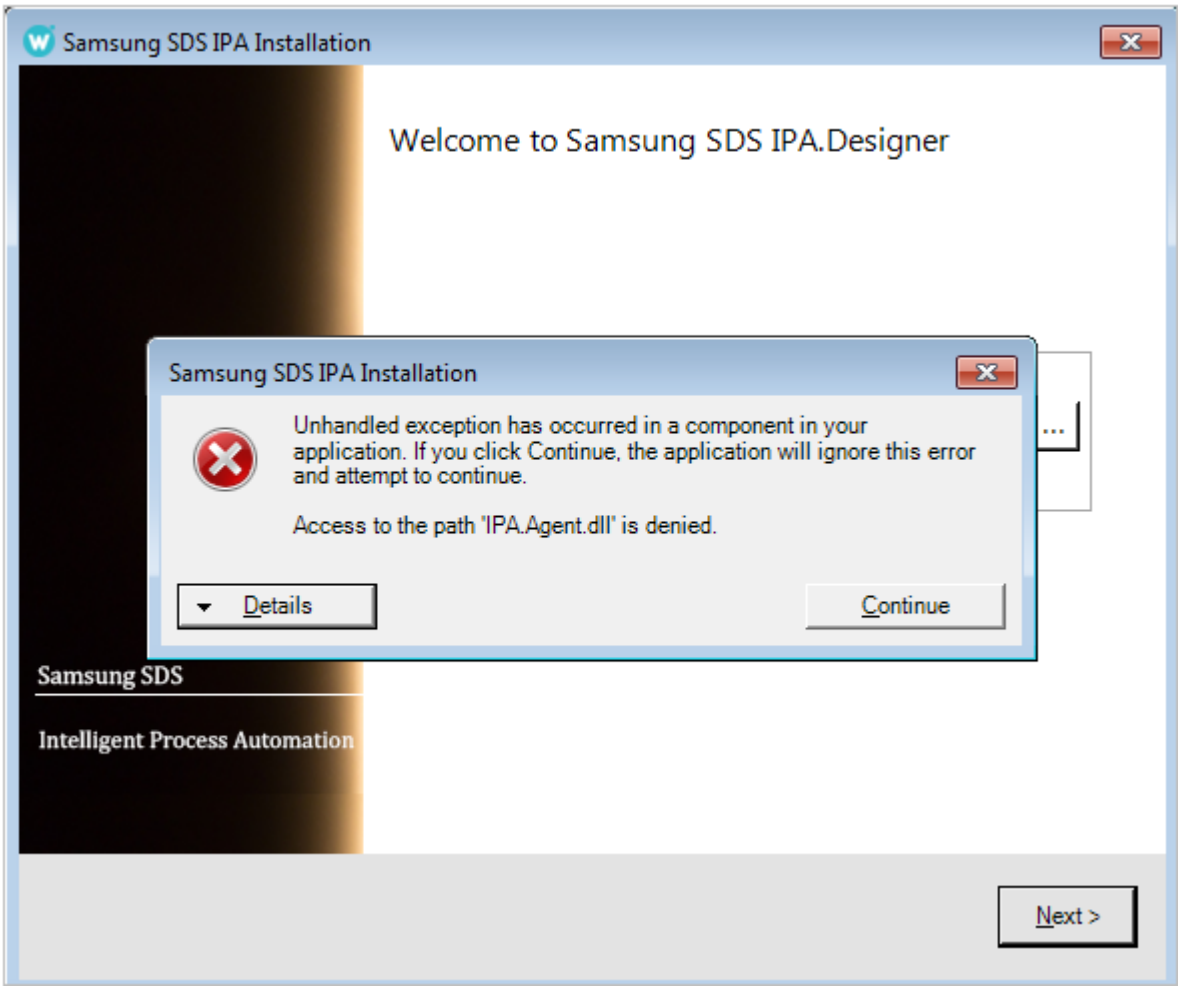
A.1 Problems related to installation and registration

> **Situation: IPA Designer and IPA Bot cannot be activated.**

Solution: IPA Designer and IPA Bot may not be activated if the certificate is not installed properly.
Install the auth_server.cer, AUTH_SERVER_CERTIFICATE.crt certificate again as shown below.

1. Click **Install Certificate**.
2. Click **Next** in the certificate importing wizard.
3. Select **Save all certificates to the repository below** and click **Find**.
4. Select the path to save certificate and click **OK**.
5. Click **Next**.
6. Confirm that the certificate is saved and click **Finish**.

> **Situation: The new version cannot be installed.**



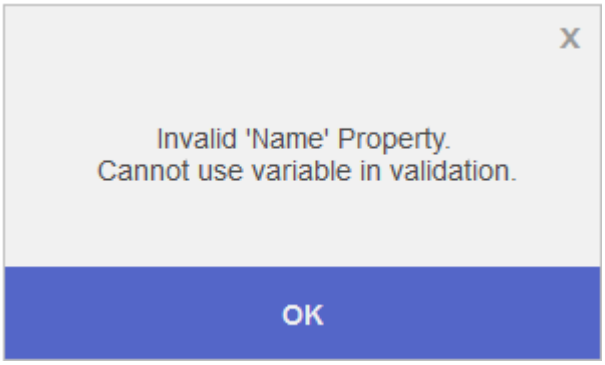
Solution: Install the new version after closing IPA Designer or IPA Bot of the previous version.

A.2 Problems related to writing and executing automation scenario

> **Situation: I want to click a sub-menu among app menus. If I press an upper menu to open a sub-menu, it is specified as a target UI object and the desired menu cannot be selected.**

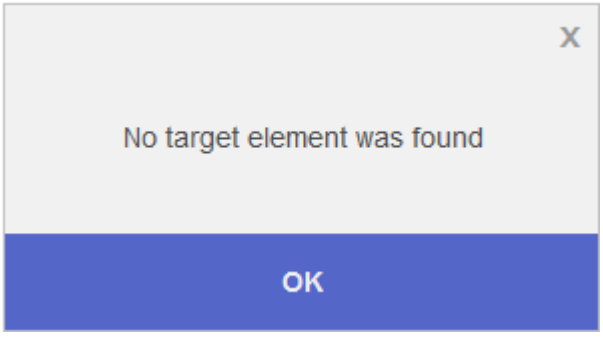
Solution: When having to select the target UI after clicking the UI object of the previous stage, such as a sub-item of an app menu or a context menu, the UI object clicked in the previous stage is specified as the target UI object. The same problem occurs when having to move to the relevant screen, because there is no UI object to specify on the screen.
In this case, you can stop specifying the target temporarily by pressing the Ctrl key on the keyboard. Then, you can start specifying the target again by pressing the Ctrl key again.

Situation: An error occurs when performing a target UI object test.



Solution: The property values that can be edited, such as **Name** or **Value** are handled the same way as a script code when creating a script.
Therefore, common strings must be entered inside the quotation marks and you may enter variables without quotation marks. The screen above shows an error when a variable is specified in the **Name** property. Because the value of the variable is decided at the time of execution, the value cannot be used during the test of the scenario writing stage.
Or if the quotation marks are omitted while inputting a string, an error may occur as the input value is recognized as a variable or script syntax.

Situation: The target UI object cannot be found.



Solution: Check **Name**/**Value** values and **Check Name**/**Check Value**/**Check Size** items of the target UI object on the properties screen. If the target UI object information of the recorded time and the performed time are different, the target UI object might not be found.
There are cases where the target UI does not properly provide the object information, such as in some apps including Chrome browser or websites using flash. In this case, you can specify the target UI with the image recognition method using the ImageRecognition library.

> **Situation: If each UI object is performed independently it works well, but I have problems when many UI objects are used consecutively.**

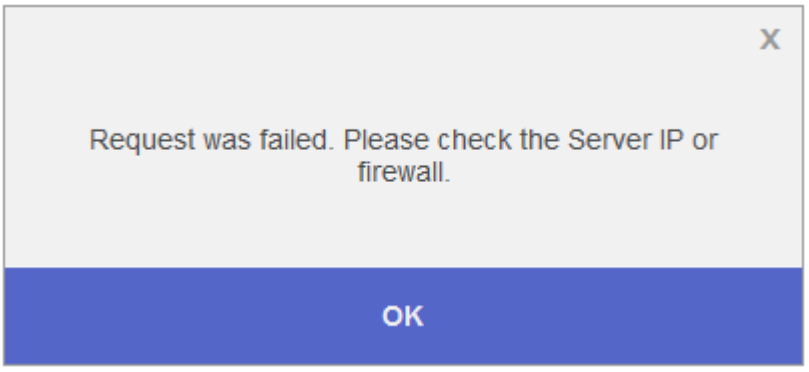
Solution: Some UI objects, such as combo box item or context menu may disappear from the screen if focus is set up. In this case, try again after setting **Focus Before** to **False**.
It might not be possible to get the UI object information properly at the time when screen is getting changed. Most of all, the possibility of error gets higher in case of being switched to the new window screen. In this case, add **Delay** library between each step to adjust the performing speed.

> **Situation: Error occurs if another task is called.**

Solution: After checking **Input** and **Output** fields from the properties screen of the task to call, check **Input** and **Output** fields of the RunTask properties screen where the error occurred. **Input**/**Output** variables of the target task and **Input**/**Output** properties of RunTask must match. If **Input** and **Output** variables of the task gets changed, they must be changed in the RunTask to call the task also.
Check if the task to call is saved. Tasks not saved will be executed using the script before editing.

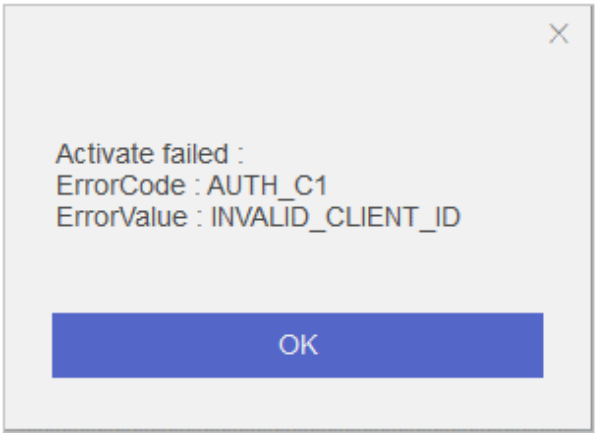
A.3 Problems related to linking servers

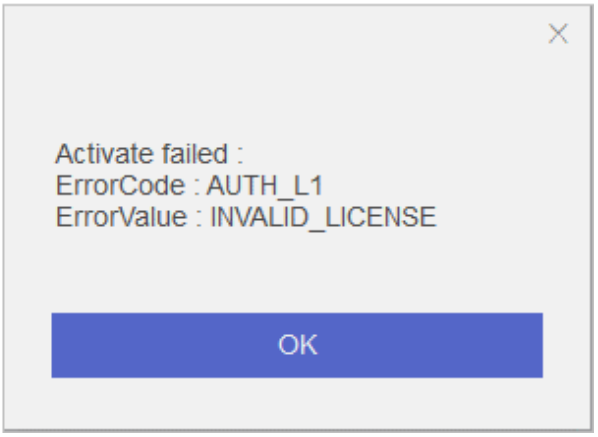
Situation: The server in not connected.



Solution: Check server IP or firewall.

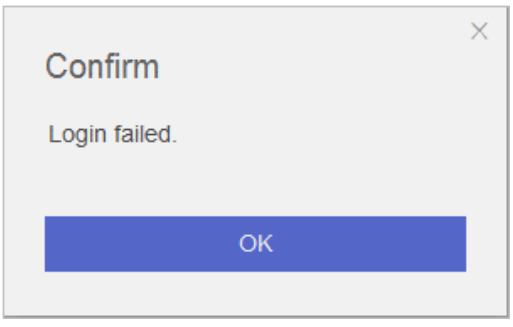
Situation: IPA Designer is not activated.





Solution: Check the client ID or license key.

Situation: I cannot log in.



Solution: Check the ID or password.

Appendix B.
FAQ

You can view frequently asked questions and answers.

Q. Is it possible to execute even if the resolution is different from the time of writing the scenario?

A. The mechanism is to search the target UI object using the information provided in Windows, SAP or Excel. Because the target UI is identified with an up-down relationship between UI objects, text information, and type information, there is no problem in performing basic actions even if the resolution is different.
However, the apps that change the overall screen configuration based on the resolution may not operate properly.

Q. Does a scenario written in Windows 7 also operate in Windows 10?

A. If the target apps are identical, they will operate the same way regardless of the Windows version. However, it may not operate properly if the target app just has the same name and function, but different version. For example, Paint for Windows 7 and the Paint for Windows 10 are not compatible, because they are not the same apps.

Q. Is it possible to use in a virtual OS?

A. It is possible to use in a virtual OS just like an ordinary PC.
However, you must be careful because the performance of a virtual OS is low in most cases. The scenario may not operate properly in an OS environment with low performance. In this case, you can use by adjusting time, such as adding a Delay step or adjusting speed from the settings screen.

Terminology