

Commands That Use Regex

WDG Automation Studio has a multitude of commands that use Regex, so to better embrace them, these commands will be separated into four categories:

Commands that use Regex as the primary resource

Starting with commands that use Regex as the primary feature, there are only three commands in this category, which are:

Get Text by Regular Expression (getRegex)

Gets the total or group correspondence of a regular expression, being able to specify the position, group name or group number.

(*) Get Text from Regular Expression

comment here

Input Parameters

Text* ?

Regular expression* ?

Options ?

Group Number ?

Group Name ?

Get by index ?

Index* ?

Output

Text gotten ?

Cancel Save

In this command, there are the following parameters:

- **Text (entry):** Text to be analyzed from which matches are obtained;
- **Regular expression (entry):** Regex to get the matches from the text.
- **Options (input):** Regex flags;
- **Group number (entry):** Regex group number;
- **Group name (entry):** Name of the Regex group;
- **Get Index (entry):** Option that allows you to search for the match by its index in case more than one match occurs in the text;
- **Index (entry):** Regex correspondence index in text;
- **Obtained text (output):** Obtained text that corresponds to the regular expression.

Get Table by Regular Expression (getRegexTable)

This command analyzes text based on a Regex, thus generating a table with all matches and their groups.

Create Table By Regular Expression

comment here

Input Parameters

Text * ?

Regular expression * ?

Options ?

Output

Table ?

Rows ?

Columns ?

Cancel Save

In this command, there are the following parameters:

- **Text (entry):** Text to be analyzed by Regex from which the table will be obtained;
- **Regular expression (entry):** Regex used to map table values;
- **Options (input):** Regex flags;
- **Table (output):** Table of data created from the matches in the text by Regex analysis, with following columns:
 - **Index:** Concerns the position in which the match begins in the regular expression;
 - **Value:** Relates to the match found using regular expression;
 - **N:** "N" represents a positive number greater than 1, related to partial matching, that is, the matches obtained by groups in regular expressions; **N_Index:** Corresponds to the index number of the positions of the groups obtained in the regular expression, in which the correspondence of the column "N" begins in the text.
- **Rows (output):** Number of rows from the obtained table ;
- **Columns (output):** Number of columns from the obtained table.

Search by Regular Expression (isMatch)

This command checks for matches in a text according to the Regex defined.

Find by Regular Expression

comment here

Input Parameters

Text* ?

Regular expression* ?

Options ?

Output

Occurrence ?

Cancel Save

In this command, there are the following parameters:

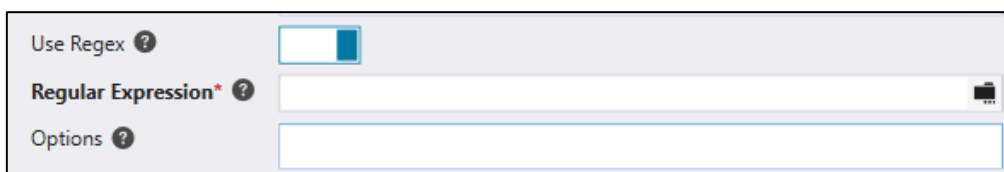
- **Text (entry):** Text analyzed to find regex matches;
- **Regular expression (entry):** Regex used to find matches in the text;
- **Options (input):** Regex flags;
- **Occurrence (output):** Boolean response indicating whether any Regex matches exist in the text.

Commands that use Regex as an assistant

On the commands that use Regex as an assistant, since there is a wide variety, we provide a list with all of them:

- Close Process (closeProcess);
- Check If Process Exists (ifProcess);
- Wait for Process (waitProcess);
- Find Table Cell Occurrences (findTableCell);
- Count Occurrences in Text (countTextOccurrences);
- Find Occurrences in Text (parseText);
- Replace Text (replaceText);
- Count Windows (countWindows);
- Find Window (findWindow);
- Find Windows (findWindows);
- Handle Open File Dialog (handleOpenFileDialog);
- Handle Save File Dialog (handleSaveFileDialog);
- Start or Associate Window (launchOrAttach);
- Wait for Window to Appear (waitWindow);
- Waits and Closes Window (watchAndCloseWindow);
- Connect to office application (officeAttach).

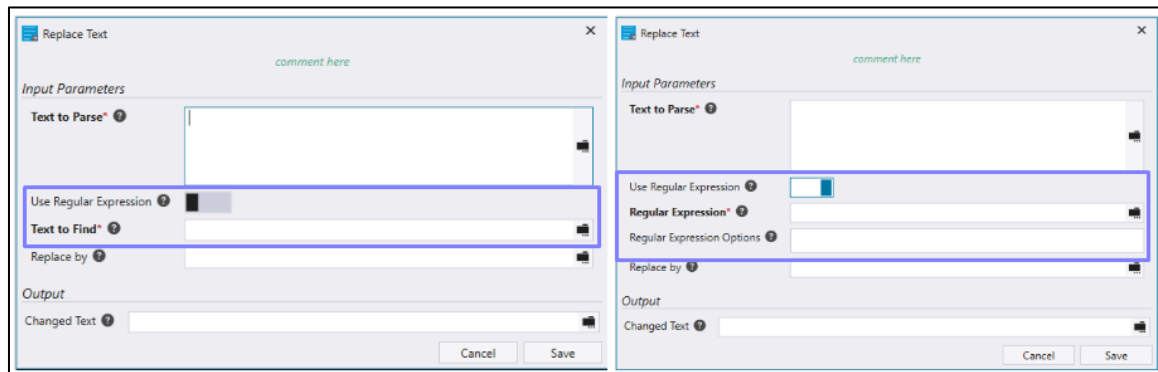
All of these commands have a parameter called "Use regular expression", which is responsible for enabling the use of a regular expression to assist the command's actions. When you enable this parameter, two new fields are shown, which are "Regular Expression" and "Options."



The image shows a configuration panel with three rows. The first row has the label 'Use Regex' followed by a question mark icon and a checked checkbox. The second row has the label 'Regular Expression*' followed by a question mark icon and a text input field with a small icon on the right. The third row has the label 'Options' followed by a question mark icon and a text input field.

- **Regular expression:** Regex used;
- **Options:** Regex flags.

For a better understanding, there will be an example with the "ReplaceText" command, responsible for overwriting the occurrences of a value in a text. At first, the occurrence is reported as a text, however, by activating the option "Use regular expression", such occurrence can be reported as regular expression, replacing all texts that match such expression.



Commands that use Regex as operator

Just as there is a wide variety of commands that use Regex as an assistant, there is also a wide variety of commands that use Regex as the operator, such as those listed below:

- Copy Collection Items If (collectionCopyIf);
- Count Item Occurrences If (collectionCountIf);
- Move Collection Item If (collectionMoveIf);
- Remove Items from Collection If (collectionRemoveIf);
- Else If (elseif);
- Run Subroutine If (gosubIf);
- Go to If (gotoIf);
- If (if);
- End Do While (until);
- When (when);
- While (while);
- Get Table of a Text File (textReadTable);
- Assert condition (assert);

- Imply (imply);
- Assign Value to Variable If (setVarIf).

Here Regex is used as an operator when the "Operator" parameter of a command is "Matches". This option tells us that the command should perform its action when something matches the defined regex.

To better exemplify, the "If" (if) command will be used, responsible for executing a block of code if a condition is met.

Note that in the "Left Operator" there is the word "mathing" and, when you mark the option "Operator" as "Matches", it is configured to perform the action (run the command block), if the contents of the "Left Operator" correspond to the Regex reported in the "Right Operator".

It is important to mention that when a Regex is used as an operator, you cannot use regex flags.

Commands that use Regex as a Comparator

Also, in WDG Automation Studio, there are cases where Regex is used as a comparator, with only two commands currently supported, which are "Get PDF Text by OCR" (extractPdfText) and "Click by OCR" (ocrClick). Such commands use Regex as a comparator when they have the option "Matches" selected in the "Comparison" parameter.

In the "Get PDF Text by OCR" (extractPdfText) command, Regex is used to perform a comparison within an area of a PDF in order to locate the value that corresponds to the defined regex, making the value found by the regular expression the anchor of the text extraction.

On the other hand, in the "Click by OCR" (ocrClick) command, Regex is used to perform a comparison across the user control interface in order to click the option that matches the defined Regex.

Click by OCR
 comment here

Input Parameters

Handle Error
 Safe Search
 OCR Provider* Google
 Comparison* Matches
 Language English
 Segmentation* Word
 Text* \bc\w*e\b
 Occurrence Type First occurrence
 Double Click
 Click on Position
 Selector* Name
 Name* File
 Region
 Invert Colors
 Update screen cache
 Element in Table
 Timeout

Output

Success
 Image
 Text
 Region

Cancel Save

It is important to note that, as with commands that use Regex as operators, commands that use Regex as a comparator also do not allow you to use Regex flags.