



Open2Test Web Test Automation Framework Scripting Standards for TestPartner

Version 1.0

September 2009

DISCLAIMER

Verbatim copying and distribution of this entire article is permitted worldwide, without royalty, in any medium, provided this notice is preserved.

TABLE OF CONTENTS

1. INTRODUCTION	5
1.1. Purpose	5
2. STANDARDS FOR KEYWORD SCRIPTING	6
2.1. Before Starting	6
2.2. Column Description	6
2.2.1. Automate (Column 'A')	7
2.2.2. Action (Column 'B')	7
2.2.3. Object (Column 'C')	10
2.2.4. Actionvalue1 (Column 'D')	12
2.2.5. Actionvalue2 (Column 'E')	16
2.2.6. Comments (Column 'F')	16
2.2.7. Delimiters.....	17
2.2.8. Variables.....	18
3. SEQUENCE OF KEYWORDS	19
3.1. Use of keyword 'Context'	19
3.2. Use of 'Conditional statements'	20

LIST OF TABLES

Table 1: Objects used in the Open Source Test Automation Framework..... **Error!**
Bookmark not defined.

TABLE OF FIGURES

Figure 1: Keyword Script and Calling the Framework.....	6
Figure 2: Column 'Automate'	7
Figure 3: Column 'Action'	8
Figure 4: Column 'Object'	10
Figure 5: Column 'Actionvalue1'	13
Figure 6: Column 'Actionvalue2'	16
Figure 7: Column 'Comments'	17
Figure 8: Keyword 'Context'	19

1. Introduction

1.1. Purpose

This document provides details about the various columns used, the keywords and their descriptions, along with some methodologies that need to be followed while using keywords for scripting.

2. Standards for Keyword Scripting

2.1. Before Starting

Before going into the details about the columns used for keyword scripting, the user should be familiar with what is known as the 'keyword script' and how to call the framework from the test script.

As shown in the figure below, the keyword script is the actual automation test script that corresponds to the manual test case. It is written in the Excel sheet and mapped to Active Data of the tool. The framework is called using the command 'Call Keyword_Driver(<TestScriptName>, <datasheet Name>)' from test script. Data sheet is an optional parameter to the Keyword_Driver.

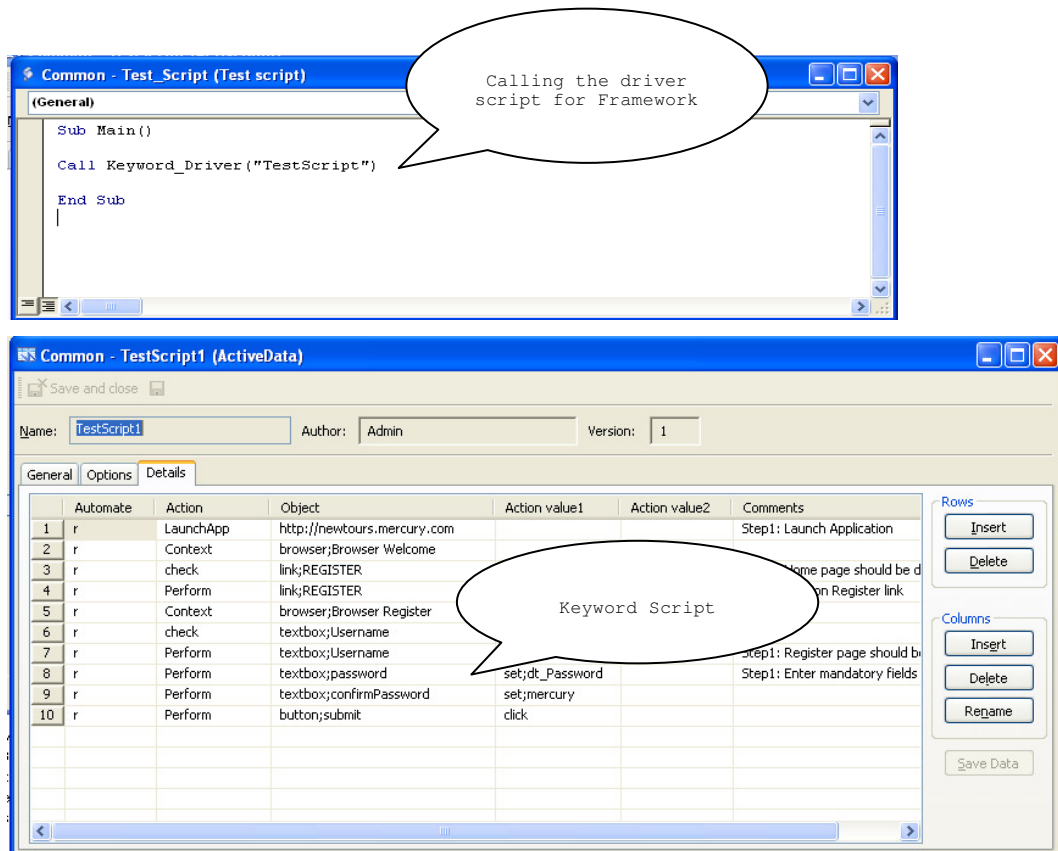


Figure 1: Calling the Framework and Keyword Script

2.2. Column Description

This section gives a description of the columns used for keyword scripting.

2.2.1. Automate (Column 'A')

The data in the 'Automate' column decides whether the current step in the test case is to be run (automated) or not. This column has the letter 'r,' which denotes that the current step should be run. If any step in the test case is not being, run then the corresponding row in the first column is to be left blank. The steps will run only based on the data in this column.

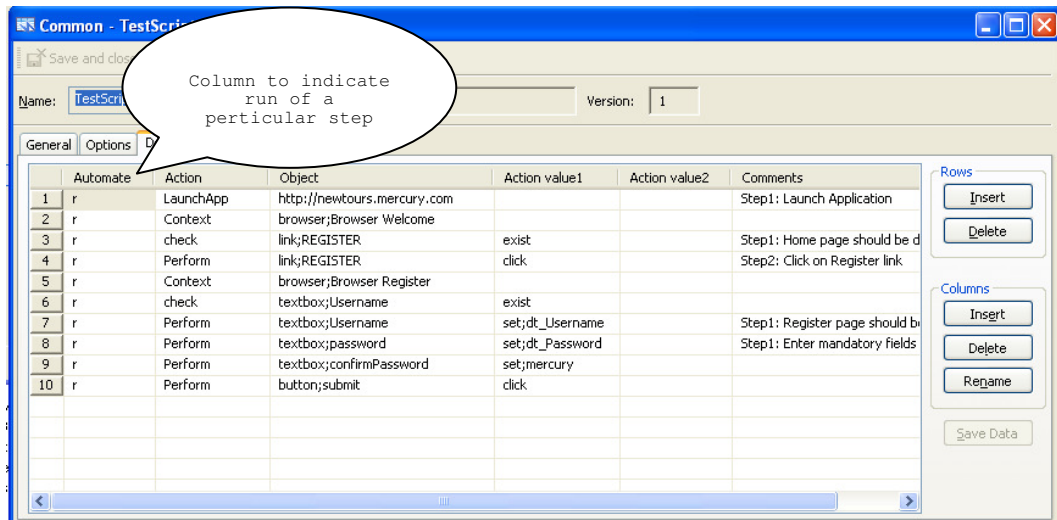


Figure 2: Column 'Automate'

2.2.2. Action (Column 'B')

The second column of the active data sheet is used to indicate the generic type of action being performed on the application under test (AUT). The action column is dedicated to different types of actions that are to be performed on a particular object.

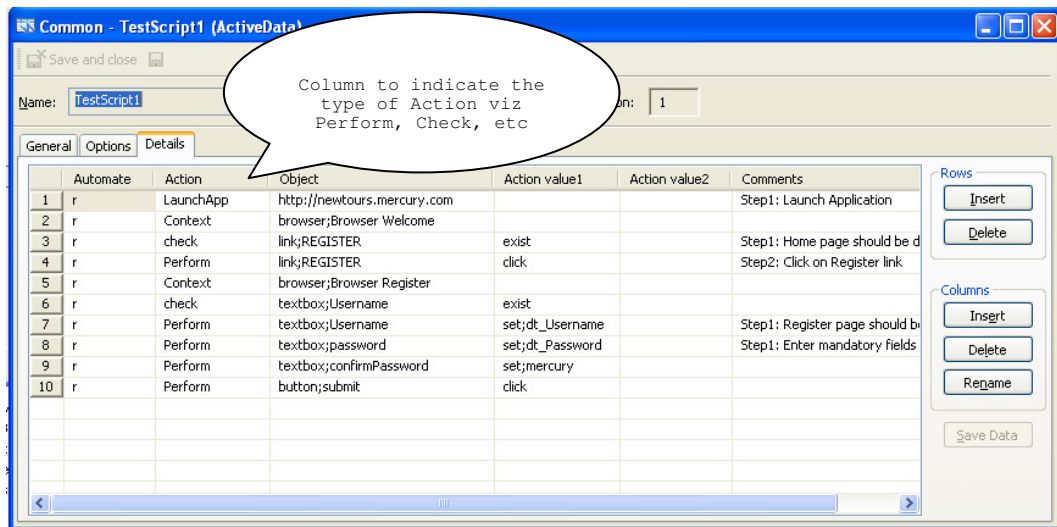


Figure 3: Column 'Action'

The keywords that can be used in this column are:-

1. **LaunchApp**

'LaunchApp' is used to launch the AUT. This keyword triggers the driver script to launch the application from a specified folder (the location specified in the third column).

2. **Context**

'Context' is used only on a window object. This keyword brings a particular window to the current context, so that any operation or checking can be performed on that particular window.

3. **Perform**

'Perform' is used to perform an operation on a particular object such as clicking on a button, closing an open window, typing some text in a textbox, etc. This keyword should be entered in the corresponding row in the second column if any such operations are to be performed.

4. **Check**

'Check' is used to check if the required property of a particular object is attained at runtime. This is a type of validation step (expected result).

5. **Condition**

'Condition' provides a feature for comparing two variables, checking properties, checking for the existence of windows, etc.

6. **CallFunction**

'CallFunction' is used to call any declared function that is used in a particular script. These functions should be declared in a different .vbs file.

7. **Storevalue**

'Storevalue' is used to store the property values of different objects in different dictionary variables. These dictionary variables can later be used as input parameters in various functions and also in scripts.

8. **PressKey**

'PressKey' is used to pass hot keys such as Enter, F3, F10, Ctrl-S, etc.

9. **Msgbox**

'Msgbox' is used for debugging to display the contents of a variable.

10. **Report**

'Report' is used for customized reporter events. It is displayed in the result sheet. The report can be of 2 types: i) Pass, ii) Fail.

11. **Strsearch**

'Strsearch' is used to search for a 'sub string' inside a 'main string'.

12. **Strreplace**

'Strreplace' is used for replacing a 'sub string' inside a 'main string' with a new 'sub string'.

13. **Strconcat**

'Strconcat' is used to concatenate any number of strings with each other.

14. **Wait**

'Wait' is used to place static waits in the keyword script.

15. **Arith**

'Arith' is used to perform the arithmetic operations on the variables.

16. **Assignvalue**

'Assignvalue' is used to assign dynamically generated values from the application to dictionary variables.

17. **Callaction**

'Callaction' is used to call reusable actions that are declared in the script.

18. **Loop**

'Loop' is used to loop a set of actions given in the test script.

19. **convert**

'Convert' is used to typecast from one data type to another.

20. **Function**

'Function' is used to perform FSO (File system Object) operations, such as creating a folder in a specified path, creating a file in a specified path, etc.



A detailed description of the keywords is provided in the Keyword Reference Document.

2.2.3. Object (Column 'C')

The third column of the active data sheet is used to indicate the object on which a particular type of action is to be performed. When the object is present in the object map, the object class and object name are specified in column C (as shown in example 1). The object column or column 'C' contain all the required details for an object (viz. Class to which the objects belong to and the object name) on which various operations and validations are to be performed.

Example 1:

Action	Object
Perform	Element;OK
Perform	Textbox; Lastname

In the above example, the object column indicates that some operation has to be performed on an object of class 'WebElement' having the name 'OK'. Similarly, in the next line some operation has to be performed on an object of class 'Textbox' having the name 'Lastname'.



The object and its name are usually separated by a delimiter ';' as shown in the above example. (Delimiters will be covered in a later topic).

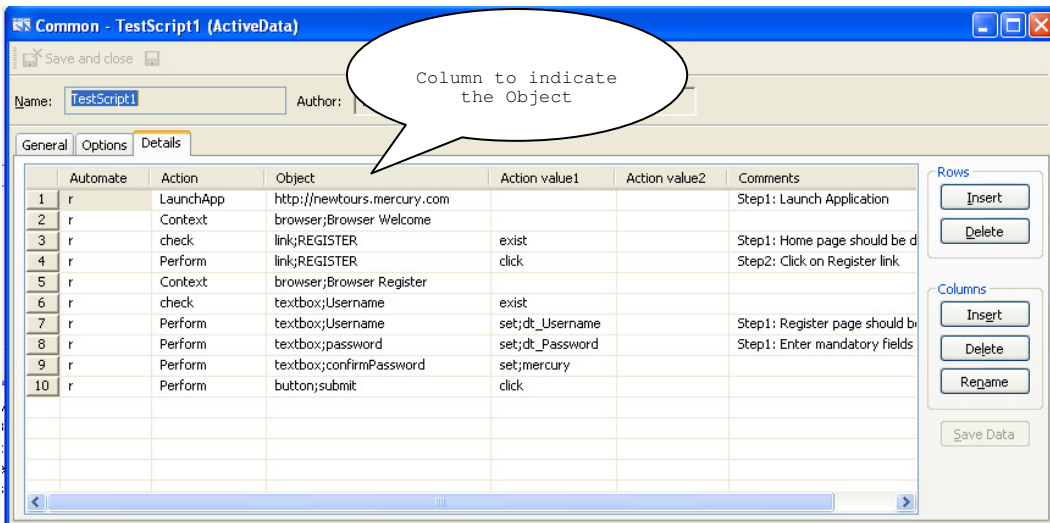


Figure 4: Column 'Object'

The objects that are commonly used are:

S1.No	Objects used in the Open Source Test Automation Framework	Window Object Class
1.	Browser	HTMLBrowser, MozillaBrowser
2.	Anchor	HTMLAnchor, MozillaAnchor
3.	Frame	HTMLFrame, MozillaFrame
4.	Window	Window
5.	Button	HTMLButton, MozillaButton
6.	Checkbox	HTMLCheckBox, MozillaCheckBox
7.	Combobox	HTMLComboBox, MozillaComboBox
8.	Textbox	HTMLEditBox, MozillaEditBox
9.	RadioButton	HTMLRadioButton, MozillaRadioButton
10.	Div	HTMLDiv, MozillaDiv
11.	Image	HTMLImage, MozillaImage
12.	Area	HTMLArea
13.	ListBox	HTMLListBox, MozillaListBox
14.	Table	HTMLTable, MozillaTable
15.	InputFileEditBox	HTMLInputFileEditBox,
16.	Label	HTMLLabel, MozillaLabel
17.	Span	HTMLSpan, MozillaSpan
18.	TD	HTMLTD, MozillaTD
19.	TextArea	HTMLTextArea, MozillaTextArea
20.	ViewLink	HTMLViewLink
21.	TabBar	IETabBar, MozillaTabBar
22.	TreeView	IETreeView, MozillaTreeView
23.	ListView	ListView
24.	ToolBar	ToolBar, MozillaToolBar
25.	ScrollBar	ScrollBar
26.	CalendarControl	CalendarControl
27.	ColorPicker	MozillaColorPicker
28.	ColorPickerButton	MozillaColorPickerButton
29.	Menu	MozillaMenu
30.	Object	MozillaObject
31.	Popup	MozillaPopup
32.	ProgressMeter	MozillaProgressMeter

33.	TabCtrl	MozillaTabCtrl
-----	---------	----------------

Table 1: Objects used in the Open Source Test Automation Framework



A detailed description of the keywords is given in the Keyword Reference Document.

2.2.4. Actionvalue1 (Column 'D')

The fourth column of the active data sheet indicates the specific action being performed on the object present in the AUT. It contains the details of all the operations or verifications that have to be performed on the objects listed in the 'Objects' column.

Consider the example of the object 'HTMLElement' with the name OK.

One of the actions that can be performed on an HTMLElement would be Click, so in column 4 the above operation is put in the keyword form as "CLICK".

Example 2: The keyword CLICK on an OK button is as follows:

Action	Object	Operation
Perform	Element;OK	Click

ACTION

If the user wants to **check** if the button is enabled before clicking, the syntax would be:

Action	Object	Operation
Check	Element;OK	Enabled:True

CHECKING

It would be the same if the user wants to check whether the object is disabled. The syntax would be:

Action	Object	Operation
Check	Element;OK	Enabled:False

CHECKING

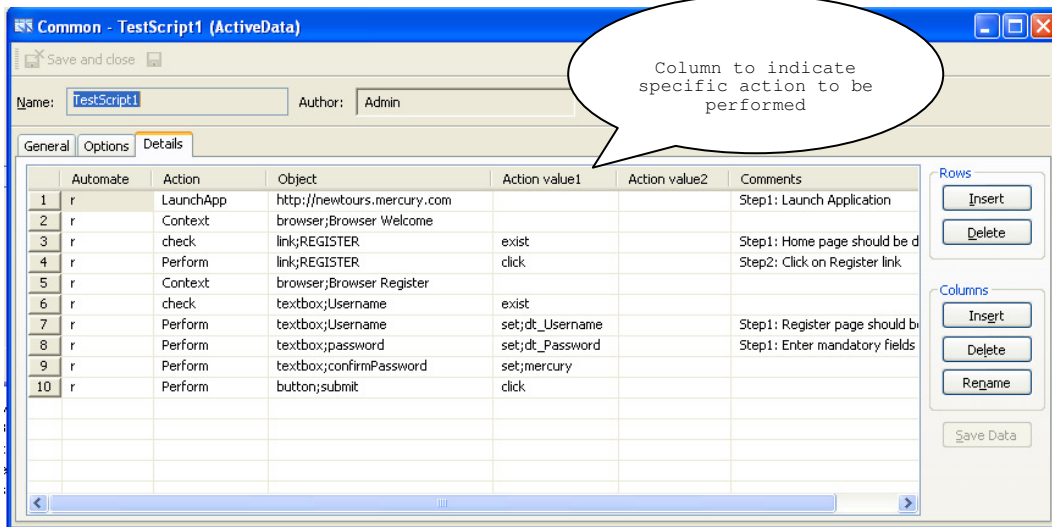


Figure 5: Column 'Actionvalue1'

The most commonly used keywords for specific actions that can be used with the generic keyword **'Perform'** written in Column 'Action' are:

1. **Click**

'Click' is used to perform the click operation on objects. It is used with the perform keyword in keyword scripts (e.g., clicking an HTMLButton).

2. **Close**

'Close' is used to perform the close operation on any open objects. It is used with the perform keyword in keyword scripts (e.g., closing a window, etc.).

3. **Maximize**

'Maximize' is used to perform the maximize operation on any open objects. It is used with the perform keyword in keyword scripts.

4. **Minimize**

'Minimize' is used to perform the minimize operation on any open objects. It is used with the perform keyword in keyword scripts.

5. **Restore**

'Restore' is used to perform the restore operation on any open objects. It is used with the perform keyword in keyword scripts.

6. **Select:<name/Item>**

'Select' is used to select an item from Combobox, Listbox, Treeview, Listview, and Tab. It is used with the perform keyword in keyword scripts.

7. **Selectindex:<index>**

'Selectindex' is used to select an item from a Tab or Combobox. It is used with the perform keyword in keyword scripts.

8. **Set:<Text>**
'Set' is used to assign a value to an edit field. It is used with the perform keyword.
9. **Type:<Value>**
'Type' is used to assign a value to an edit field. It is used with the perform keyword.
10. **Type:<Item>**
'Type' is used to select an item from Combobox. It is used with the perform keyword.
11. **Type:d_currenttime**
This sets the current system time to the edit field. It is used with the perform keyword in keyword scripts.
12. **Type:d_currentdate**
This sets the current system date to the edit field. It is used with the perform keyword in keyword scripts.
13. **Setdate:Date/Now/<Date>**
This sets the current system date (Date)/ current system date and time.
(Now)/specified date(<date>) to the calendar object. It is used with the perform keyword in keyword scripts.
14. **SetTime:Now/<Time>**
This sets the current system time (Now)/specified time(<Time>) to the calendar object. It is used with the perform keyword in keyword scripts.
15. **Set**
This is used to select a radio button. It is used with the perform keyword in keyword scripts.
16. **DoubleClick**
'DoubleClick' is used to perform the doubleclick operation on objects. It is used with the perform keyword in keyword scripts.
17. **Press:<name>**
This is a perform operation to click on the specified toolbar item.
18. **<conversiontype>:<variable name>:<format type>**
This is used to convert a variable from one data type to another.
19. **Collapse:<name>**
This is used to collapse the treeitem specified. It is used with the perform keyword in keyword scripts.
20. **SelectRange:<item name1>:<item name2>**
This is used to select the range of items in a listview.
21. **Create;<Folder Path/Name>/<File Path/Name>**
This is used to create a folder/file in the specified path.
22. **Delete;<Folder Path/Name>/<File Path/Name>**

- This is used to delete a folder/file in the specified path.
23. **Copy;**<Source Path/Name>;<DestinationFolder Path/Name>/<Source File Path/Name>;<Destination Folder Path>
This is used to copy a folder/file from the source to the destination path specified.
 24. **Move;**<Source Path/Name>;<DestinationFolder Path/Name>/<Source File Path/Name>;<Destination Folder Path>
This is used to move a folder/file from the source to the destination path
 25. **Write;**<File Path/Name>;<The value to be entered>
This is used to write the file with the data mentioned in the specified path.
 26. **Read;**<File Path/Name>;<Variable to store data from file>
This is used to read the contents of a mentioned file and store the values in the specified variable.
 27. **Append;**<File Path/Name>;<text to be appended to file>
This is used to append the data specified with the data contained in the file.
 28. **SQLMultipleCapture;**<query>
This is used for capturing multiple values from the database.

The most commonly used keywords for specific actions that can be used with the generic keyword **Check** written in Column 'Action' are:

1. **Checked:**<True/False>
This is a check operation that is used to verify whether a checkbox is checked or not.
2. **Enabled:**<True/False>
This is a check operation that is used to verify whether the given window or object is enabled or not.
3. **Exist:**<True/False>
This is a check operation that is used to verify whether the window or object whose name is specified exists or not.
4. **Focused:**<True/False>
This is a check operation that is used to verify whether the object is focused or not.
5. **ItemCount:**<Item>
This is a check operation that is used to verify the number of items present or not in a specified object.
6. **Text:**<text/#Variable_Name>
This is a check operation that is used to verify whether the required text is present or not in the object.
7. **Prop_name:**<variable_name>
This is used to store the property value in the specified variable. It is used with the storevalue keyword.
8. **Windowtext:** <Text>:<True/False>

This is a check operation that is used to verify whether a windows text is present or not in the window object.

9. Tabexist:<Tabitemname>

This is a check operation that is used to verify whether the tab item specified is present or not.

10. ItemExist:<Item name>

This is a check operation that is used to verify whether an item is present or not in the window object.



A detailed description of the keywords is provided in the Keyword Reference Document.

2.2.5. Actionvalue2 (Column 'E')

The fifth column of the active data sheet may be used to store the values returned from specific functions (e.g., user-defined functions).

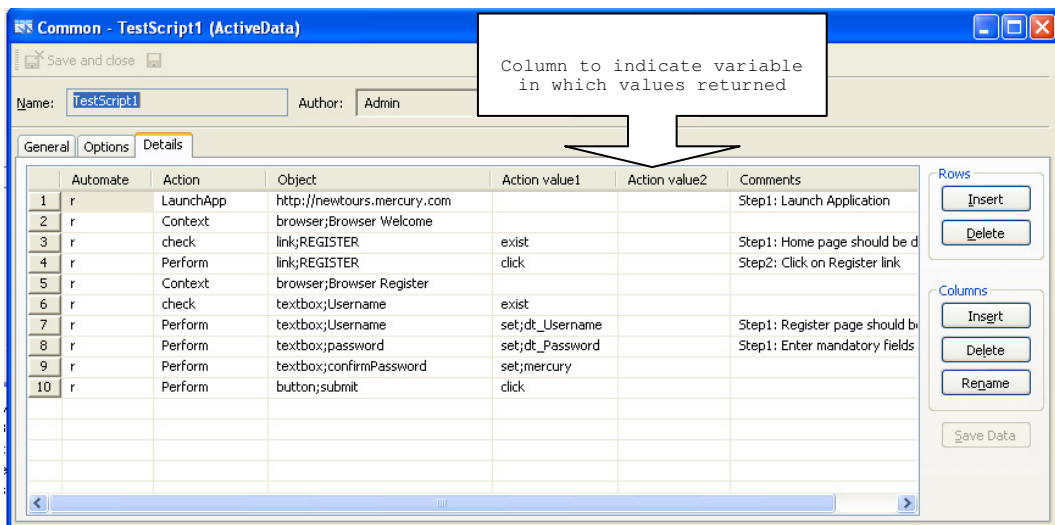


Figure 6: Column 'Actionvalue2'

2.2.6. Comments (Column 'F')

The 'Comments' column is used to enter generic information about the current step being run. It provides a better understanding of the steps being performed in the particular test script and also helps to map the test script to the manual test case.

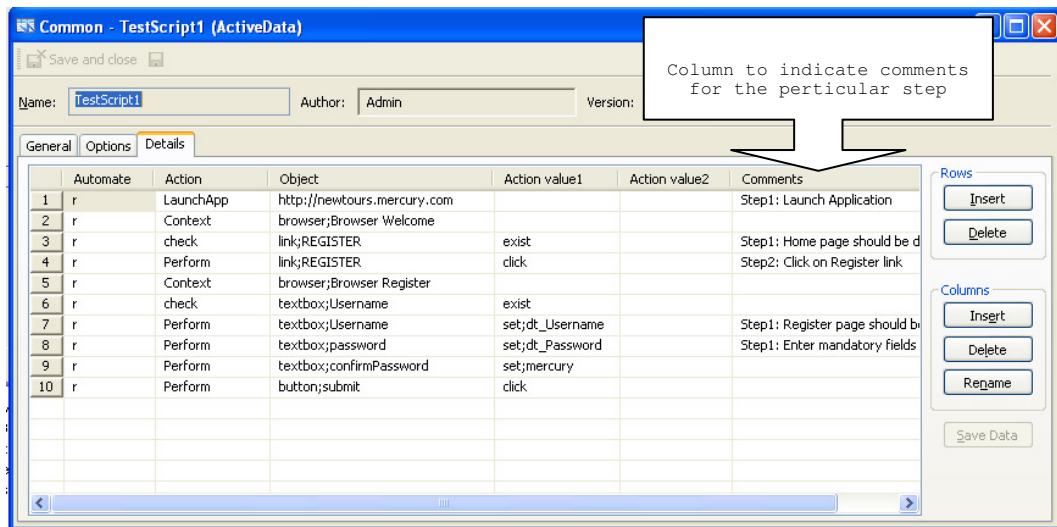


Figure 7: Column 'Comments'

2.2.7. Delimiters

Delimiters are any string characters used to identify the sub-string limits. Delimiters are generally used with the Split function, which is used to split the input in to different substrings.

When a delimiter is omitted, the space character (" ") is assumed to be a delimiter.

Purpose of using delimiters:

The main purpose of using delimiters in this framework is to break down the input values to different strings and take them as keywords to perform any operation concerned with that object.

Delimiters Used in this framework:

The most important point to keep in mind while scripting using the keyword-driven approach is to place separators or delimiters between two keywords. Delimiters that are used in the framework are:

- : (colon)
- ; (semi colon)
- :: (double colon)

Understanding the usage of delimiters:

There are four columns involved in the keyword-driven approach. The role of delimiters comes in the 'Objects' column (column 3) and 'Actionvalue1' column (column 4).

'Objects' column (column 3):

This column is used to define the class and the name of the object. The delimiter used in this column separates the class of the object and the name of the object with a semi-colon ';'.

Example:

Textbox; <textboxname>

'ActionValue1' column (column 4):

This column usually provides the details of the operations that need to be performed on the object. The delimiter used to separate the property and the property values in this column is a colon ':'.
Example:

```
Selectindex: <index>
```

To specify the actual1 and actual2 values for report, the delimiter that is used is a double colon '::'.
Example:

```
<event>;<exp1>:<exp2>::<act1>:<act2>
```

2.2.8. Variables

- To store a value in a variable, an environment variable is used.

Example:

Assignvalue	strName;Smith	
-------------	---------------	--

Here in the variable 'strName', the value 'Smith' is stored.

- To store the property value of an object, a dictionary variable is used.

Example:

Storevalue	Textbox;<textbox name>	Prop_name:<varName>
------------	------------------------	---------------------

Here, the value in the textbox is stored to a variable 'varName'

- To input a value to a field from a variable, the variable should be preceded by '#'.
Example:

Perform	Textbox;<textbox name>	Set:#varName
---------	------------------------	--------------

Here, the value stored in varName is typed into the textbox.

To define a variable, certain standards need to be followed. For example, for a variable to store a string value it should be appended with "Str" ex.StrVarName. Similarly, for integer, it should be appended with "int" and for Boolean should be appended with "bln".

3. Sequence of Keywords

While scripting using keywords, some keywords have to be written in combination with other keywords. This section deals with those methodologies.

3.1. Use of keyword 'Context'

The keyword 'context' has to be used whenever the AUT screen changes. Example:

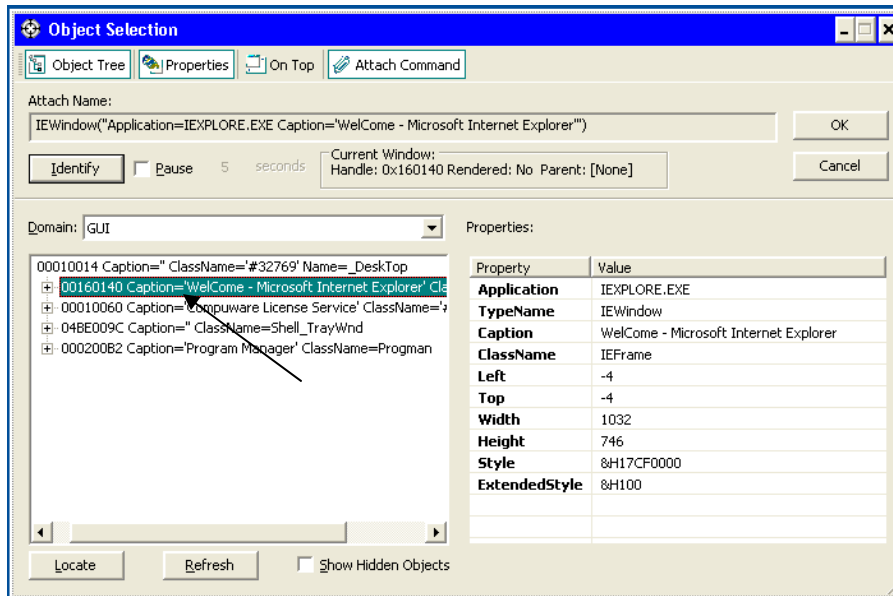


Figure 8: Keyword 'Context'

If the object 'EditBox' has to be used in the script, then the preceding row should have the context set to the previous object in the hierarchy.

Therefore, the combination to be used while performing an action on the object 'EditBox' is:

Context	Browser;Caption='Welcome - Microsoft Internet Explorer'	
Perform	Textbox;Username	SetText:User1

If we have to use another object on the same page, the context need not be set again.

Context	Browser;Caption='Welcome	
---------	--------------------------	--

	- Microsoft Internet Explorer'	
Perform	Textbox;Username	SetText:User1
Perform	Textbox;Password	SetText:pass123

3.2. Use of 'Conditional statements'

If the user is implementing an If - Else conditional statement, then the keyword is followed by a semi-colon ';' and the values that indicate the start row and the end row should be separated by a semi-colon ';'.
Example:

Condition	<var1>;comparator;<var2>	startrow;endrow
-----------	--------------------------	-----------------

If the condition mentioned is 'True,' execution starts from the startrow and would end at the endrow specified. If the condition specified is 'False,' there would be no effect in the script and the execution would continue as normal.

When an operation has to be used, then two conditional statements have to be used together to satisfy the 'and' condition

Condition	<var1>;comparator;<var2>	startrow;endrow
Condition	<var2>;comparator;<var3>	startrow;endrow

Therefore, this effectively implies that an 'and' operation is being performed.

COPYRIGHT

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.