Open2Test Powerbuilder Test Automation Framework for QTP – Scripting Standards

Version 1.0

April 2011

DISCLAIMER

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

TABLE OF CONTENTS

List of Tables

Tables of Figures

1. Introduction

1.1. Purpose

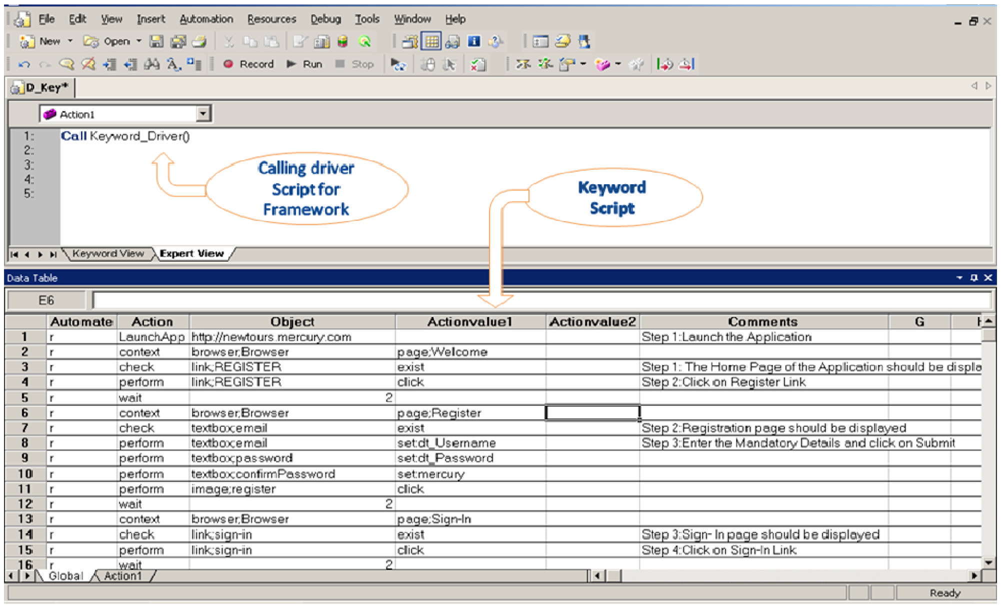
This document provides details about the various columns used during scripting, the keywords and their descriptions, along with methodologies to be followed while scripting using keywords.

2. Standards for Keyword Scripting

2.1. Getting Started

Before learning about the columns used for keyword scripting, the user should know 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 global sheet of the tool. In the 'Expert View' of the tool, the framework is called using the command 'Call Keyword_Driver()'.



```
Call Keyword_Driver()
```

E6

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign- In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

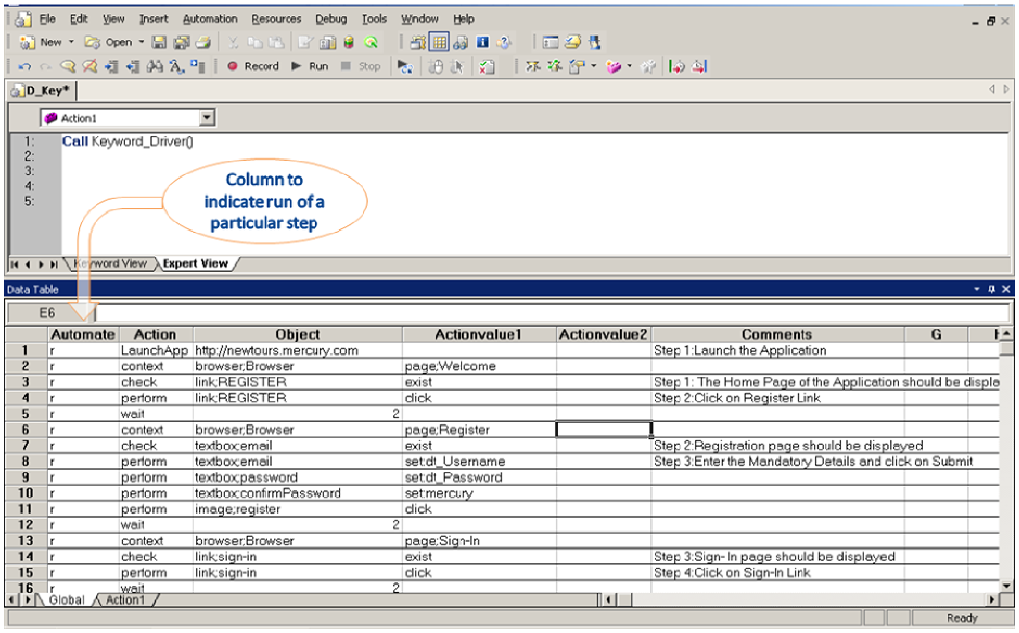
Figure 1: Keyword Script and Calling the Framework

2.2. Column Description

This section describes the columns used for keyword scripting.

2.2.1. Automate (Column 'A')

The data in the 'Automate' column identifies 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. All the steps will run based on the data in this column.


File Edit View Insert Automation Resources Debug Tools Window Help
New Open Record Run Stop
D_Key*
Action1
Column to indicate run of a particular step
Keyword View
Expert View
Data Table
Global
Action1
Ready

```
Call Keyword_Driver()
```

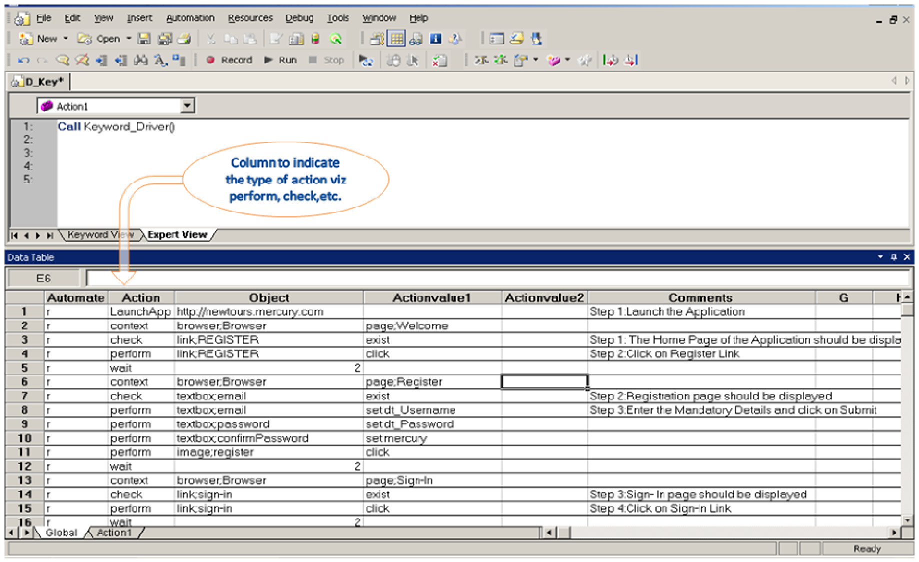
E6

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign- In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

Figure 2: Column 'Automate'

2.2.2. Action (Column 'B')

The second column of the global sheet indicates the generic type of action being performed on the application under test (AUT). The Action column is dedicated to different types of actions that will be performed on a particular object.


File Edit View Insert Automation Resources Debug Tools Window Help
New Open Record Run Stop
D_Key*
Action1
Column to indicate the type of action viz perform, check,etc.
Keyword View
Expert View
Data Table
Global
Action1
Ready

```
Call Keyword_Driver()
```

E6

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign- In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

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 either from a specified folder (the location of which is specified in the third column) or, if the application is already synchronized with QuickTest Professional (QTP), then this automatically launches the application from the location specified in QTP.

2. **Context**

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

3. **Perform**

   'Perform' is used to perform an operation on a particular object (ex: clicking on a button, closing a Window, or typing some text in a textbox). 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' is used to compare two variables, check properties for the existence of windows, etc.

6. **CallFunction**

   'Call Function' is used to call any declared function 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 environment variables. These environment variables can later be used as input parameters in various functions and also in scripts.

8. **PressKey**

   'Press Key' 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 results can include: i) Pass, ii) Fail, iii) Done, iv) Warning.

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 one other.

14. **Wait**

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

15. **Arith**

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

16. **Assignvalue**

'Assignvalue' is used to assign dynamically generated values from the application to environment 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 datatable.

19. **Convert**

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

A detailed description of the keywords is provided in the QTP Open Source Test Automation Framework Keywords for PowerBuilder document.

2.2.3. Object (Column 'C')

The third column of the global 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 repository, the object class and object name are specified in column C (as shown in Example 1). However, if the object is not added to the object repository, descriptive programming can be used by specifying any property and its value (as shown in Example 2). The object column, or column 'C', contains 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	Button;OK
Perform	Textbox; Lastname

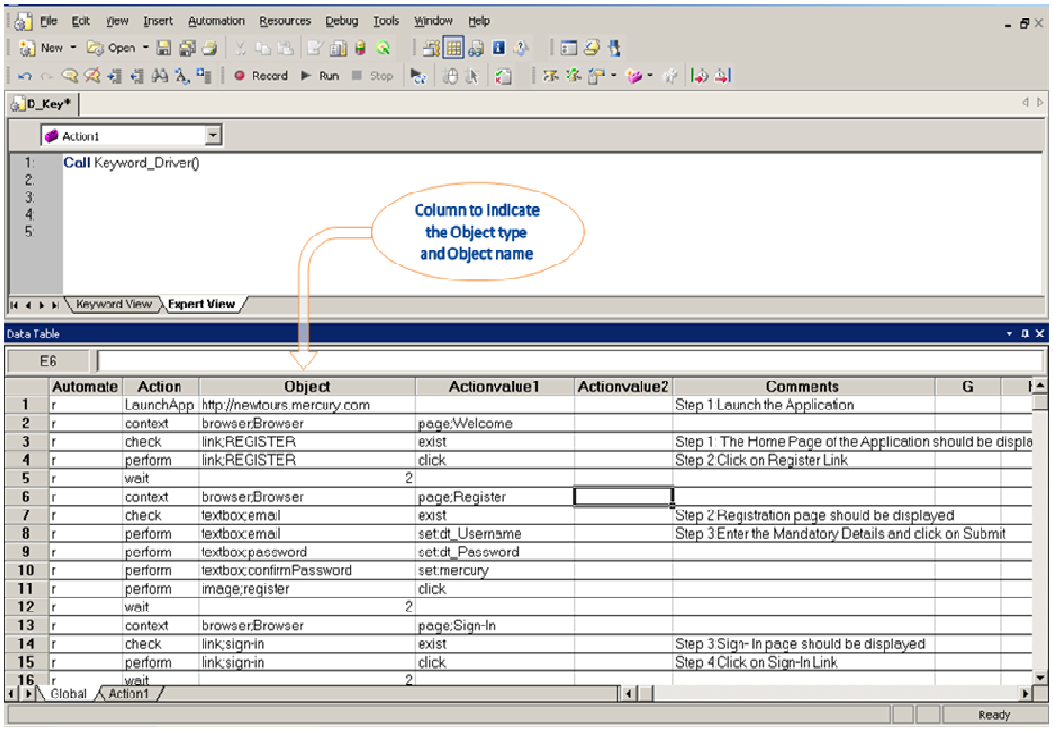
In the above example, the object column indicates that some operation has to be performed on an object of class 'PbButton' having the name 'OK'. Similarly, in the next line some operation has to be performed on an object of class 'PbEdit' having the pbname 'Lastname'.

Example 2:

Action	Object
Perform	Button;text:=OK
Perform	Textbox;pbname:=Lastname

In the above example, the object is not added to the object repository. Some operation has to be performed on an object of class PbButton having a property 'text', the value of which is 'OK'. Similarly, some operation has to be performed on an object of class PbEdit having a property 'pbname', the value of which is 'Lastname'.

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



```
Call Keyword_Driver()
```

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign-In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

Figure 4: Column 'Object'

Commonly used objects include:-

Sl.No	Objects used in the Open Source Test Automation Framework for PowerBuilder	PowerBuilder Object Class
1.	Window	PbWindow
2.	DataWindow	PbDataWindow

3.	WinButton	WinButton
4.	Button	PbButton
5.	WinEdit	WinEdit
6.	WinMenu	WinMenu
7.	WinScrollBar	WinScrollBar
8.	Scrollbar	PbScrollbar
9.	Checkbox	PbCheckBox
10.	Combobox	PbCombobox
11.	Textbox	PbEdit
12.	List	PbList
13.	Listview	PbListview
14.	Radio button	PbRadioButton
15.	Tabstrip	PbTabstrip
16.	Toolbar	PbToolbar
17.	Treeview	PbTreeview
18.	Object	PbObject
19.	Dialog	Dialog

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

2.2.4. ActionValue1 (Column 'D')

The fourth column of the global sheet is used to indicate the specific action being performed on the objects present in the AUT. It contains the details of all the operations or verifications to be performed on the objects listed in the objects column.

Consider the example of the object 'PbButton' with logical name 'OK'.

One of the actions that can be performed on a PbButton would be Click.

Therefore, 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	Button;OK	Click

ACTION

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

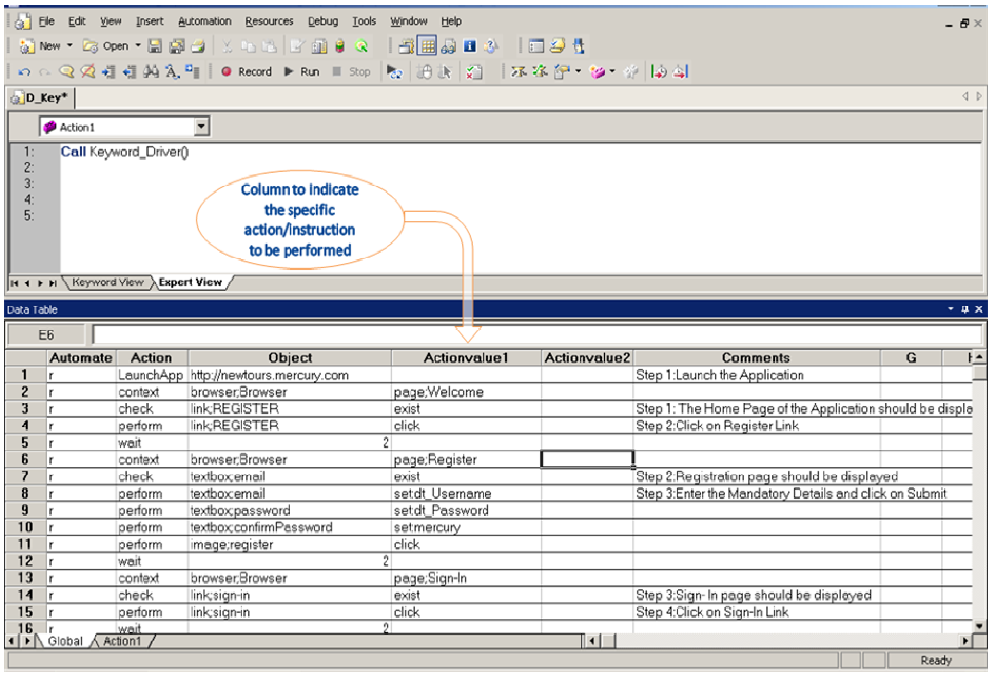
Action	Object	Operation
Check	Button;OK	Enabled:True

CHECKING

Similarly, if the user wants to check whether the object is disabled, the syntax would be

Action	Object	Operation
Check	Button;OK	Enabled:False

CHECKING



```
Call Keyword_Driver()
```

E6

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign- In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

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 (ex. clicking a Button).

2. **Close**

   'Close' is used to perform the close operation on any open objects. (ex. closing a window or dialog box).

3. **Select:<name>**

   'Select' is used to select an item from Combobox or List.

4. **Selectindex:<index>**

   'Selectindex' is used to select an item from a List or Treeview.

5. **Set:<Text>**

   'Set' is used to assign a value to an edit field.

6. **Set:d_currenttime**

   'Set:d_currenttime' sets the current system time in the edit field.

7. **Set:d_currentdate**

   'Set:d_currentdate' sets the current system date in the edit field.

8. **Set:d_d; <value to be added/subtracted>**

   This function adds or subtracts the value specified to the current system date and sets the edit field to a given value.

9. **Set:d_m; <value to be added/subtracted>**

   This function adds or subtracts the value specified to the current system month and sets the edit field to a given value.

10. **Set:d_y; <value to be added/subtracted>**

    This function adds or subtracts the value specified to the current system year and sets the edit field to a given value.

11. **Set:<On/Off>**

    This function is used to select or deselect a Radio button

12. **Submit**

    This function is used to submit the value entered in a Textbox, Combobox, or Radio button, etc.

13. **Deselect:<name>**

    This function is used to deselect a selected item in a list.

14. **Extendselect:<name>**

    This function is used to select more than one item from a list.

15. **set:env_<Environment Name>**

    This function is used to assign the value, which is stored in environment variable, to the edit field.

16. **set:dt_Parameter**

    This function is used to assign the value, which is given in the external test data sheet, to the edit field.

17. **set:#<variable>**

    This function is used to assign the value, which is stored in the variable, to the edit field.

18. **set:p_<parameter>**

    This function is used to assign the value, which is stored in the Input Parameter, to the edit field.

19. **setcelldataindex:<#varrownumber>:<#varcolumnnumber>:<value to be entered in the cell>**

    This function is used to set the contents of a cell of datawindow identified by the <#varrownumber> and <#varcolumnnumber> to the specified text. #varrownumber and #varcolumnnumber are variables that contain value for row number and column number.

20. **setcelldataindex:<rownumber>:<internalcolumnnumber>:<value to be entered in the cell>**

    This function is used to set the contents of a cell of datawindow identified by the <rownumber> and <internalcolumnnumber> to the specified text. <internacolumnnumber> takes into account the hidden columns. E.g. if there is a hidden column to the left of the first visible column, the internal column number of the first visible column is 2.

21. **setcelldataindex:<rownumber>:<visiblecolumnnumber>:<value to be entered in the cell>**

    This function is used to set the contents of a cell of datawindow identified by the <rownumber> and <visiblecolumnnumber> to the specified text.

22. **setcelldataname:<rownumber>:<visiblecolumnname>:<value to be entered in the cell>**

    This function is used to set the contents of a cell of datawindow identified by the <rownumber> and <visiblecolumnname> or the column header to the specified text.

23. **setcelldataname:<rownumber>:<internalcolumnname>:<value to be entered in the cell>**

    This function is used to set the contents of a cell of datawindow identified by the <rownumber> and <internalcolumnname>.

24. **activatecell:<row>:<column>**

    This function is used to double-click on the cell of datawindow specified by the <row> and <column>.

25. **selectrow:<rownumber> or <#varrownumber>**

    This function is used to click on the cell of datawindow specified by the <rownumber> or the variable <#varrownumber> thereby selecting the whole row.

26. **selectrow:<val1>:<val2>:<val3>…**

    This function is used to click on the cell of datawindow specified by the <rownumber> or the variable <#varrownumber> thereby selecting the whole row.

27. **selectcellindex:<#varrownumber>:<#varcolnumber>**

    This function is used to click on the cell of datawindow specified by the <#varrownumber> and <#varcolumnnumber>. #varrownumber and # varcolumnnumber are variables that contain values for row number and column number.

28. **selectcellindex: <rownumber>:<internalcolumnnumber>**

    This function is used to click on the cell of datawindow specified by the <row> and <internalcolumnnumber>.

29. **selectcellname:<rownumber>:<internalcolumnname>**

    This function is used to click on the cell of datawindow specified by the <rownumber> and <internalcolumnname>.

30. **selectcellname:<rownumber>:<visiblecolumnname>**

    This function is used to click on the cell of datawindow specified by the <#row> and <#column>.

31. **<conversiontype>:<variable name>:<format type>**

This function is used to convert a variable from one data type to another.

32. **Create;<Folder Path/Name>/<File Path/Name>**

This function is used to create a folder/file in the specified path.

33. **Delete;<Folder Path/Name>/<File Path/Name>**

This function is used to delete a folder/file in the specified path.

34. **Copy;<Source Path/Name>;<DestinationFolder Path/Name>/<Source File Path/Name>;<Destination Folder Path>**

This function is used to copy a folder/file from the source to the destination path specified.

35. **Move;<Source Path/Name>;<DestinationFolder Path/Name>/<Source File Path/Name>;<Destination Folder Path>**

This function is used to move a folder/file from the source to the destination path.

36. **Write;<File Path/Name>;<The value to be entered>**

This function is used to write the file with the data mentioned in the specified path.

37. **Read;<File Path/Name>;<Variable to store data from file>**

This function is used to read the contents of a mentioned file and store the values in the specified variable.

38. **Append;<File Path/Name>;<text to be appended to file>**

This function is used to append the data specified with the data contained in the file.

39. **DBObjectName:OutputCheckPointName**

This function is used for capturing multiple values from the database.

DBObjectName is the name of the DB Object to be present in the object repository and Output Checkpoint is the name of the checkpoint placed inside, where many output values are captured.

40. **Descriptive Programming**

Descriptive programming can also used with the above object types when the object is not added to the object repository.

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

1. **Selection:<item name>**

This is a Check operation that is used to verify whether the desired item is selected in the 'Combobox'.

2. **Checked:<Name>**

This is a Check operation which is used to verify whether a Radio button in a PbRadiobutton group whose name is specified is checked or not.

3. **Checked:<True/False>**

This is a Check operation that is used to verify whether a Checkbox is checked or not.

4. **Enabled:<True/False>**

   This is a Check operation that is used to verify whether the given PowerBuilder object is enabled or not.

5. **Exist:<True/False>**

   This is a Check operation that is used to verify whether the PowerBuilder object whose name is specified exists or not.

6. **Focused:<True/False>**

   This is a Check operation that is used to verify whether the object is focused or not.

7. **Visible:<True/False>**

   This is a Check operation that is used to verify whether the specified object is visible or not.

8. **ItemsCount:<Item>**

   This is a Check operation that is used to verify the number of items in a Combobox.

9. **Text:<text/#Variable_Name>**

   This is a Check operation that is used to verify whether the required text is present in the object.

10. **rowcount:<row count value to be checked for>**

    This function is used to verify if the <row count value to be checked for> matches the actual row count of a datawindow and prints the result.

11. **columncount:<column count value to be checked for>**

    This function is used to verify if the <column count value to be checked for> matches the actual column count of a datawindow and prints the result.

12. **celldata:<rownumber>:<columnnumber>:<value to be checked>**

    This function is used to check if the <value to be checked> matches the actual value in the cell identified by <rownumber> and <columncount> and prints the result.

13. **celldataindex:<#varrownumber>:<#varcolumnnumber>:<value to be checked>**

    This function is used to check if the <value to be checked> matches the actual value in the cell identified by the variables <#varrownumber> and <#varcolumncount> and prints the result.

14. **celldataindex:<rownumber>:<internalcolumnnumber>:<value to be checked>**

    This function is used to check if the <value to be checked> matches the actual value in the cell identified by the cell with <rownumber> and <internalcolumnnumber> as row and column and prints the result.

15. **celldataindex:<rownumber>:<visiblecolumnnumber>:<value to be checked>**

    This function is used to check if the <value to be checked> matches the actual value in the cell identified by the cell with <rownumber> and <visiblecolumnnumber> as row and column and prints the result.

16. **celldataname:<rownumber>:<internalcolumnname>:<value to be checked>**

This function is used to check if the <value to be checked> matches the actual value in the cell identified by the cell with <rownumber> and <internalcolumnname> as row and column and prints the result.

17. **celldataname:<rownumber>:<visiblecolumnname>:<value to be checked>**

This function is used to check if the <value to be checked> matches the actual value in the cell identified by the cell with <rownumber> and <visiblecolumnname> as row and column and prints the result.

18. **prop_name:<variable_name>**

This function is used to store the property value in the specified variable. It is used with the 'Storevalue' keyword.

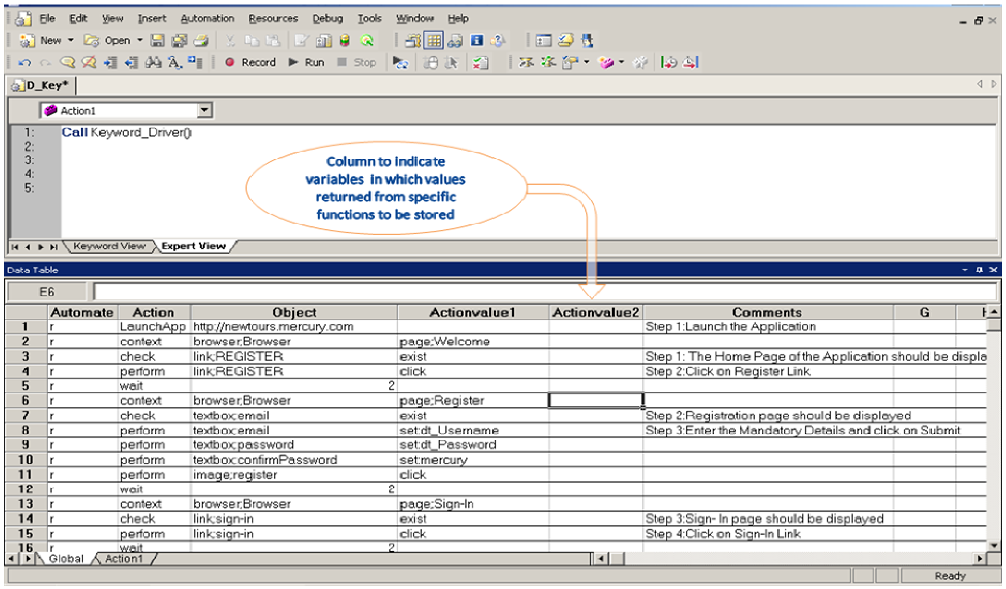
19. **text:<text to search>**

This is a Check operation that is used to verify whether a string is present or not in the table.

A detailed description of the keywords is provided in the QTP Open Source Test Automation Framework Keywords for PowerBuilder document.

2.2.5. ActionValue2 (Column 'E')

The fifth column of the global sheet may be used to store the values that are returned from specific functions (ex: user-defined functions).


File Edit View Insert Automation Resources Debug Tools Window Help
New Open Record Run Stop
D_Key*
Action1
Keyword View Expert View
Column to indicate variables in which values returned from specific functions to be stored
Data Table
E6
Global Action1
Ready

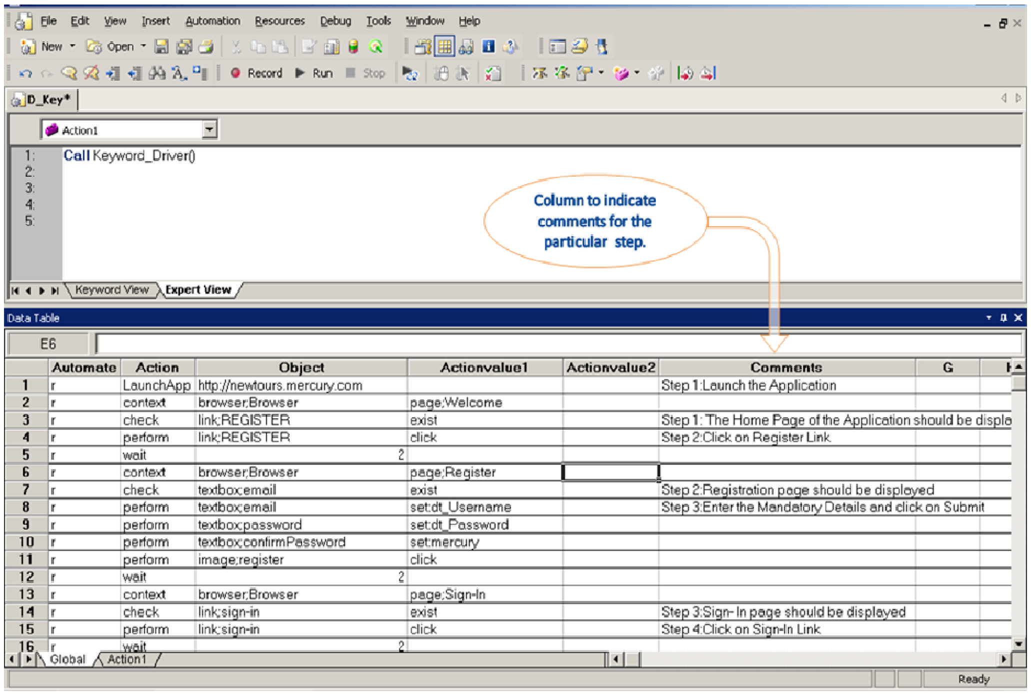
```
Call Keyword_Driver()
```

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign- In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

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.



```
Call Keyword_Driver()
```

E6

	Automate	Action	Object	Actionvalue1	Actionvalue2	Comments	G
1	r	LaunchApp	http://newtours.mercury.com			Step 1:Launch the Application	
2	r	context	browser;Browser	page;Welcome			
3	r	check	link;REGISTER	exist		Step 1: The Home Page of the Application should be displa	
4	r	perform	link;REGISTER	click		Step 2:Click on Register Link	
5	r	wait	2				
6	r	context	browser;Browser	page;Register			
7	r	check	textbox;email	exist		Step 2:Registration page should be displayed	
8	r	perform	textbox;email	set:dt_Username		Step 3:Enter the Mandatory Details and click on Submit	
9	r	perform	textbox;password	set:dt_Password			
10	r	perform	textbox;confirmPassword	set:mercury			
11	r	perform	image;register	click			
12	r	wait	2				
13	r	context	browser;Browser	page;Sign-In			
14	r	check	link;sign-in	exist		Step 3:Sign- In page should be displayed	
15	r	perform	link;sign-in	click		Step 4:Click on Sign-In Link	
16	r	wait	2				

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

Purpose of Delimiters:

Delimiters 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:

When scripting using the keyword-driven approach, use separators or delimiters appropriately between two keywords. Delimiters that are used in the framework are:

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

Using Delimiters:

There are four columns involved in the keyword-driven approach. The role of delimiters comes in the 'Objects' column (column 3) and the 'Operations' 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 to separate the class of the object and the name of the object is a semi-colon ';'.

Example:

Textbox; <textboxname>

'ActionValue1' column (column 4):

This column usually provides 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 child objects present in a window or dialog box, use the double colon '::' delimiter.

Example:

window;<name> :: datawindow;<name>

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, an environment 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.

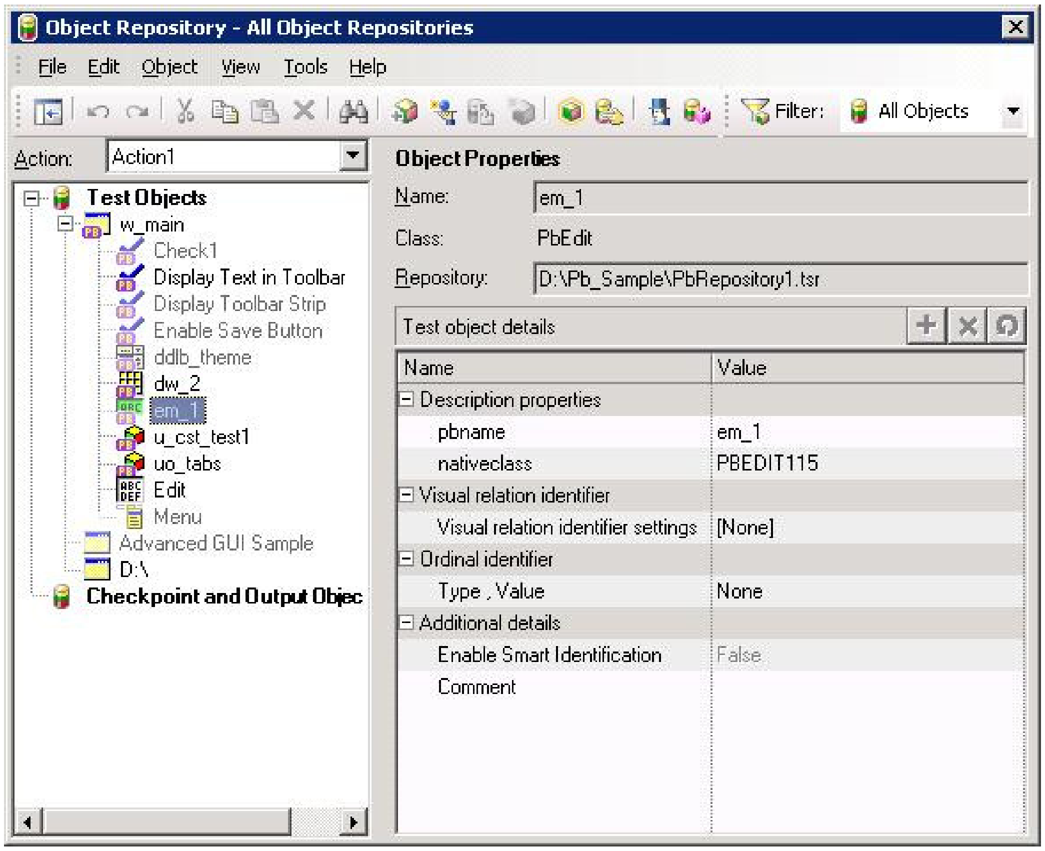
Defining a variable requires the use of certain standards. For example, for a variable to store a string value it should be appended with “Str” ex.StrVarName. Similarly, an integer should be appended with “int” and Boolean should be appended with “bln” .

3. Sequence of Keywords

While scripting using keywords, some keywords have to be written in combination with other keywords. This section deals with methodologies that should be followed while scripting using keywords.

3.1. Use of the Keyword ‘Context’

The keyword ‘Context’ has to be used whenever the AUT screen changes. Example:


Object Repository - All Object Repositories
File Edit Object View Tools Help
Filter: All Objects
Action: Action1

Test Objects
- w_main
  - Check1
  - Display Text in Toolbar
  - Display Toolbar Strip
  - Enable Save Button
  - ddlb_theme
  - dw_2
  - em_1
  - u_cst_test1
  - uo_tabs
  - Edit
  - Menu
- Advanced GUI Sample
- D:\

Checkpoint and Output Objec

Object Properties

Name: em_1

Class: PbEdit

Repository: D:\Pb_Sample\PbRepository1.tsr

Test object details

Name	Value
Description properties	
pbname	em_1
nativeclass	PBEDIT115
Visual relation identifier	
Visual relation identifier settings	[None]
Ordinal identifier	
Type , Value	None
Additional details	
Enable Smart Identification	False
Comment	

Figure 8: Keyword ‘Context’

If the object ‘em_1’ 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 ‘em_1’ is :

Context	Window; <w_main>	
Perform	Textbox;em_1	Set:Smith

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

Context	Window; <window name>	Window; <window name>
Perform	Textbox;em_1	Set:Smith
Perform	Checkbox;Check1	Set:ON

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.

Two conditional statements must be used together to satisfy the 'and' condition.

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

This 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.