Open2Test Web Test Automation Framework for TestPartner

Version 1.0

September 2009

TABLE OF CONTENTS

1. Purpose of the Document

The purpose of this document is to describe Open Source Test Automation Framework code in detail.

1.1. Scope

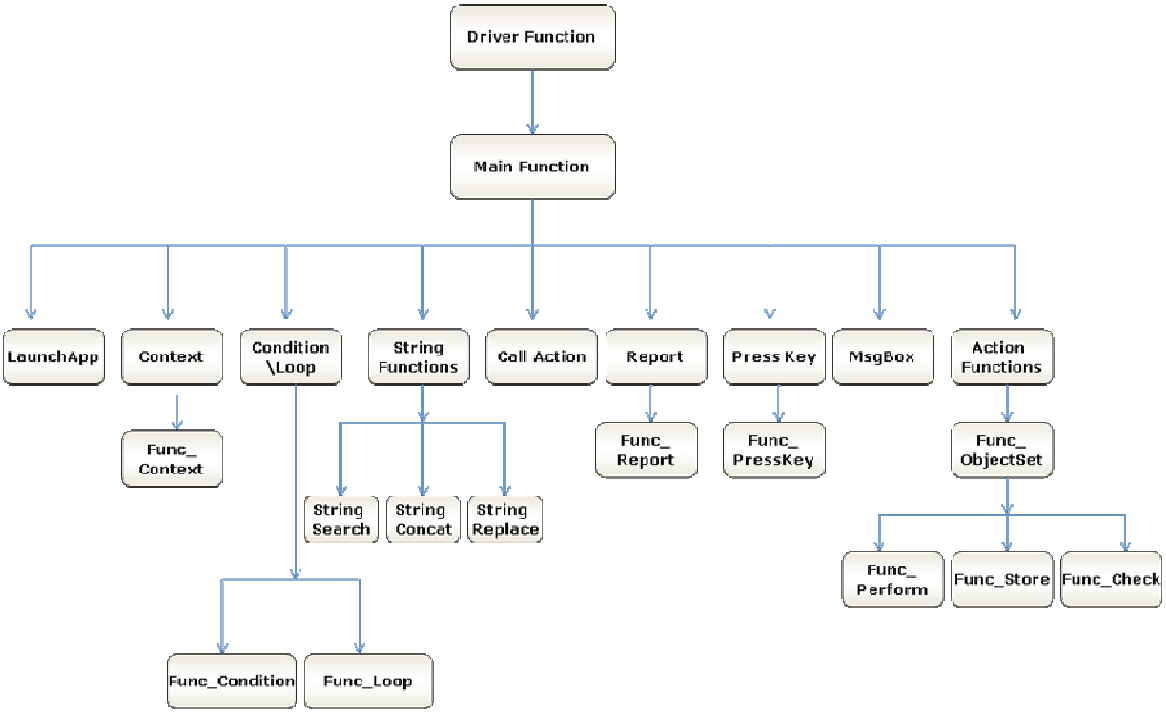
The scope of this document is to provide details about Open Source Test Automation Framework code.

1.2. Overview

This document provides details about:

- Framework Architecture
- Driver Function
- Action Functions
- Common Functions
- User-Defined Functions

2. Framework Code Structure



3. Driver Functions

3.1. Keyword Driver Function

Name of the function: Keyword_Driver()

Description: This function is used to call the main framework.

Parameters:

a) Script: This parameter contains the Active Data name of the keyword test script to be executed.

b) Testdata (Optional): This parameter contains the Active Data name of a datasheet to be passed along with the test script.

Assumptions: Keyword Test Script and Test Data are loaded as an Active Data asset in Testpartner (TP).

Variables:

a) rowCount: This variable is used to store the number of rows in the Keyword Test Case sheet.

b) rowCountDataSheet: This variable is used to store the number of rows in the Test Data sheet.

c) intRowCount: This variable is used to store the iteration count for looping through the rows of Keyword TC.

Functionality:

- The Driver function reads the values in the first column of the active data sheet.
- Whenever the value is 'r,' it calls the main function (Keyword_Web).
- When 'r' is not present in any of the cells in the first column, it will skip the row and read the value in the next row.
- If the global sheet is empty then it reports fail, stating "Script is not present in the global sheet."

3.2. Main Function

Name of the function: Keyword_Web()

Description: This is the main function, which interprets the keywords and performs the desired actions. All the keywords used in the data table are processed in this function.

Parameters: NA

Assumptions: The automation script is present in the active data sheet of Testpartner.

Variables:

a) initial – This variable is used to store the value in the second column.

b) objName – This variable is used to store the value in the third column.

c) arrAction() – This array is used to store the object type and name.

d) objPerform – This variable is used to store the value present in the fourth column.

e) arrKeyValue() – This array is used to store two values in the fourth column, separated with delimiter ":".

f) arrKeyIndex() – This array is used to store all the values in the fourth column, separated by delimiter ":"

g) strIexplorePath: This variable stores the path at which exe of IE has been stored.

h) strMozillaPath: This variable stores the path at which exe of Mozilla has been stored.

i) RetVal: This variable is used to store the return value of the functions LaunchApp and Common Functions.

j) strCellData: This variable is used to store the value of the 'Object' column for 'LaunchApp' keyword.

Functionality:

- The Main function reads the values in the second, third, and fourth columns in the data sheet and stores the values in the variables. (For more details please see the variables section).
- Based on the value in the variable initial (second column), the Main function calls different functions.
- If the value is other than 'perform,' 'storevalue,' or 'check,' the Main function calls the respective functions. For example, if the value is 'report,' it will call Func_Report().
- If the value is 'perform,' 'storevalue,' or 'check,' the Main function calls the function Func_ObjectSet() to set the object and then it calls the respective functions. For example, if the value is 'perform,' the Main function will call Func_Perform() to perform required actions on the Application Under Test (AUT).
- Refer to the framework code structure diagram for a better understanding of the function calls.

4. Action Functions

4.1. Perform Function

Name of the function: Func_Perform()

Description: This function performs the set of actions on the required object in the AUT.

Parameters:

a) Object – This is the object on which the specified operation needs to be performed.

b) arrObj – This parameter holds the type of the object and the object name on which the action has to be performed.

c) arrKeyValue – This parameter identifies the operation that needs to be performed on the object.

d) arrKeyIndex – This additional parameter is required to identify the object on which the operation needs to be performed. It holds the value of the specific action type and the value to be used.

e) intRowCount – This parameter holds the count of the current row in the data table.

Assumptions: Context is set on the current object where the action has to be performed.

Variables:

a) Dbconn : This variable is used to store the database connection object.

b) Dbrs: This variable is used to store the result set of database operation performed.

c) CaptureValue: This variable is used to store the loop count of SQL Multi Capture keyword.

d) Intarray(): This variable is used to store the integer array value.

e) IntX, intY: These variables are used to store the x, y coordinates of the mouse.

f) SQLvalueCapture: This variable is used to store the value captured by SQL Query.

g) boolaction: This variable is used to store the result of perform action.

Functionality:

Based on the values in arrObj(0), Func_Perform() performs different actions on objects.

If the value is:

1. Window

Based on the values in arrKeyValue(0) the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. close: Performs close operation on the parent object

b. activate: Activates the window object

c. maximize: Maximizes the window object

d. minimize: Minimizes the window object

e. restore: Restores the window object to its previous size

f. attach: Attaches or sets focus to a specific window

g. click: Simulates a mouse click on the specified window object.

h. Doubleclick: Simulates a double mouse click on a Window object.

i. menuselect:<item1~item2..>: Selects a specified menu item from a window.

j. menuselecttype:<index_no>:<value>: Selects a specified menu item from a window depending on the index_no and value.

k. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on a Window object at the specified X and Y coordinate.

l. mouseup:<X_Value>:<Y_Value>: Simulates a mouse up action on a Window object at the specified X and Y coordinate.

m. mousemove:<X_Value>:<Y_Value>: Simulates a mouse move action on a Window object at the specified X and Y coordinate.

n. rightclick: Right clicks on the specified Window object.

o. Rightclickoncoordinate:<X_Coordinate>:<Y_Coordinate> : Right clicks on the specified X and Y coordinates.

p. Setfocus: Brings the specified window object in focus.

q. Textselect:<value>: Selects the specified text in window object.

r. Type:<value>: Types the mentioned text on the specified window object.

s. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

2. Anchor

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

c. Rightclick: Right clicks on the specified object.

d. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate specified in arrKeyValue(1) on the specified object.

e. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

3. Area

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

c. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

d. mouseup:<X_Value>:<Y_Value>: Simulates a mouse up action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

e. mousemove:<X_Value>:<Y_Value>: Simulates a mouse move action on an item name, specified in arrKeyIndex(1), on the object.

f. doubleclick: Performs the double-click operation on the current object.

g. Rightclick: Right clicks on the specified object.

h. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate specified in arrKeyValue(1) on the specified object.

i. Textselect:<value>: Selects the specified text in the object.

j. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

k. Type:<value>: Types the mentioned text on the specified object.

4. Browser

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Activate: Activates the browser object.

b. Activate Parent: Activates the parent browser of the object.

c. attach: Sets the focus to the current object.

d. click: Performs click operation on the current object.

e. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

f. doubleclick: Performs the double-click operation on the current object.

g. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

h. Type:<value>: Types the mentioned text on the specified object.

i. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

j. mouseup:<X_Value>:<Y_Value>: Simulates a mouse up action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

k. mousemove:<X_Value>:<Y_Value>: Simulates a mouse move action on an item name, specified in arrKeyIndex(1), on the object.

l. Rightclick: Right clicks on the specified object.

m. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate specified in arrKeyValue(1) on the specified object.

n. Setfocus: Brings the specified object in focus

5. Button, ColorPickerButton

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

6. Calendar

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Setdate:now: Sets the current date and time in the specified calendar object.

b. Setdate:date: Sets the current date in the specified calendar object.

c. Setdate:<date>: Sets the mentioned date in the specified calendar object.

d. click: Performs click operation on the current object.

e. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

f. attach: Sets the focus to the current object.

g. Rightclick: Right clicks on the specified object.

h. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

7. Checkbox

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. doubleclick: Performs the double-click operation on the current object.

c. check: Checks the current checkbox.

d. uncheck: Unchecks the current checkbox.

8. ColorPicker

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. doubleclick: Performs the double-click operation on the current object.

c. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

d. Setcolor:<value>: Sets the specified color in color picker object.

9. Combobox

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. Type:<value>: Types the mentioned text on the specified object.

c. Textselect:<value>: Selects the specified text in the object.

d. attach: Sets the focus to the current object.

e. Select:<item_name>: Selects the specified item from the combobox.

f. Selectindex:<item_index>: Selects the specified item index from the combo box.

10. Div

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. attach: Sets the focus to the currently attached object.

c. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

d. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y co-ordinate, specified in arrKeyIndex(1), on the object.

e. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y coordinate, specified in arrKeyIndex(1), on the Object.

f. mouseup:<X_Value>:<Y_Value>: Simulates a mouse up action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

g. mousemove:<X_Value>:<Y_Value>: Simulates a mouse move action on an item name, specified in arrKeyIndex(1), on the object.

h. Rightclick: Right clicks on the specified object.

i. Textselect:<value>: Selects the specified text in the object

j. Type:<value>: Types the mentioned text on the specified window object.

k. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate specified in arrKeyValue(1) on the specified object.

l. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

11. Frame, td

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. attach: Sets the focus to the currently attached object.

c. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

d. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

e. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y coordinates, specified in arrKeyIndex(1), on the object.

f. mouseup:<X_Value>:<Y_Value>: Simulates a mouse up action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

g. mousemove:<X_Value>:<Y_Value>: Simulates a mouse move action on an item name, specified in arrKeyIndex(1), on the object.

h. Rightclick: Right clicks on the specified object.

i. Textselect:<value>: Selects the specified text in the object.

j. Type:<value>: Types the mentioned text on the specified window object.

k. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate, specified in arrKeyValue(1), on the specified object.

l. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

m. Setfocus: Brings the specified window object in focus.

12.Image

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. attach: Sets the focus to the currently attached object.

c. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

d. Rightclick: Right clicks on the specified object.

e. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate, specified in arrKeyValue(1), on the specified object.

f. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

g. doubleclick: Performs the double-click operation on the current object.

13. Inputfileeditbox, textbox

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. type: Types the value specified in arrKeyValue(1) in the current object.

b. set: Sets the value specified in arrKeyValue(1) in the current object.

c. click: Performs click operation on the current object.

d. clear: Clears the text in the specified text box control.

e. doubleclick: Performs the double-click operation on the current object.

f. attach: Sets the focus to the currently attached textbox.

g. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified on arrKeyValue(1).

h. mousedown:<X_Value>:<Y_Value>: Simulates a mouse down action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

i. mouseup:<X_Value>:<Y_Value>: Simulates a mouse up action on the X and Y coordinate, specified in arrKeyIndex(1), on the object.

j. mousemove:<X_Value>:<Y_Value>: Simulates a mouse move action on an item name, specified in arrKeyIndex(1), on the object.

k. Rightclick: Right clicks on the specified object.

l. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

m. Rightclick: Right clicks on the specified object.

n. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinates, specified in arrKeyValue(1), on the specified object.

o. Setfocus: Brings the specified window object in focus.

14. Label

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

c. Rightclick: Right clicks on the specified object.

d. Rightclickoncoordinate:<X_Value>:<Y_Value>: Right clicks on the X and Y coordinate, specified in arrKeyValue(1), on the specified object.

e. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

f. doubleclick: Performs the double-click operation on the current object.

g. Textselect:<value>: Selects the specified text in the object.

h. Type:<value>: Types the mentioned text on the specified window object.

15. Listbox

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Select:<itemname>:Selects the specified item from the listbox object.

b. Selectindex:<itemindex>: Selects the specified index item of the listbox object.

c. Selectrange:<itemname1>:<itemname2>: Selects the range of the item starting from itemname1 through itemname2.

d. Selectrangeindex:<itemindex1>:<itemindex2>: Selects the range of the items starting with index1 through index2.

e. Extendselect:<itemname>: Extend selects the specified item of listbox object.

f. Extendselectindex:<itemindex>: Extend selects the specified item index of the list object.

g. Activateitem:<itemname>: Activates the specified item of the listbox.

h. click: Performs click operation on the current object.

i. attach: Sets the focus to the currently attached object.

j. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

k. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

l. Textselect:<value>: Selects the specified text in the object.

m. Type:<value>: Types the mentioned text on the specified window object.

n. Rightclick: Right clicks on the specified object.

16. ListView

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Select:<itemname>:Selects the specified item from the list view object.

b. Selectindex:<itemindex>: Selects the specified index item of the list view object.

c. Selectrange:<itemname1>:<itemname2>: Selects the range of the item starting from itemname1 through itemname2.

d. Selectrangeindex:<itemindex1>:<itemindex2>: Selects the range of the items starting with index1 through index2.

e. Extendselect:<itemname>: Extend selects the specified item of list view object.

f. Extendselectindex:<itemindex>: Extend selects the specified item index of the list object.

g. Activateitem:<itemname>: Activates the specified item of the list view.

h. doubleclick: Performs the double-click operation on the current object.

i. Clickitem: Clicks on the specified list view object.

j. attach: Sets the focus to the currently attached object.

k. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

l. Rightclick: Right clicks on the specified object.

m. Textselect:<value>: Selects the specified text in the object.

n. Selectrightclick:<item_name>: Right clicks on the specified list view object.

17. Menu

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. doubleclick: Performs the double-click operation on the current object.

b. attach: Sets the focus to the currently attached object.

c. click: Performs click operation on the current object.

d. Type:<value>: Types the mentioned text on the specified window object.

e. Select:<itemname>: Selects the specified item from the list view object.

18. Object, Popup

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. doubleclick: Performs the double-click operation on the current object.

b. attach: Sets the focus to the currently attached object.

c. click: Performs click operation on the current object.

19. ProgressMeter

Based on the values in arrKeyValue(0), it performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

20. Radiobutton

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. Set: Sets the value of the specified radio button.

21. Random

Assigns the random number to the specified variable.

22. Scrollbar

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Set:<position>: Sets the scrollbar in the specified position.

23. Span

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Click: Clicks on the specified span object.

24. Split

split;<variable>^<delimiter>^: Splits the variables using the <delimiter> and stores the elements(e1,e2) of the split array into the variable(v1,v2) respectively.

25. Tabbar

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Select:<itemname>: Selects the specified item from the tabbar object.

b. Selectindex:<itemindex>: Selects the specified index item of the tabbar object.

c. Close:<tabname>: Closes the specified tab bar object.

d. Click: Simulates the click action in the specified tab bar object.

e. Selectrightclick:<item_name>: Right clicks on the specified tab bar object.

f. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

g. attach: Sets the focus to the currently attached toolbar object.

26. Table

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Click: Simulates the click action in the specified table object.

b.getcelldata:<row_num>:<col_num>: Retrieves celltext specifying the row number and column number and stores it in the variable specified in the fifth column for a table.

c. rownum;<rowval1>--<colno>: Stores the value in the fifth column in the variable strParam and calls the function Func_getRowNum().

d. tableSearch;<colname1>;<rowval1>::colname2>;<rowval2>--<[no of columns]>: Stores the value in the fifth column in the variable strParam. It splits the value in strParam with ';' as the delimiter and calls the function Func_tablesearch().

27. Textarea

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. type: Types the value specified in arrKeyValue(1) in the current object.

b. click: Performs click operation on the current object.

c. clear: Clears the text in the specified text box control.

d. doubleclick: Performs the double-click operation on the current object.

e. attach: Sets the focus to the currently attached textbox.

f. set: Sets the value specified in arrKeyValue(1) in the current object.

g. Setfocus: Brings the specified window object in focus.

h. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

28. Toolbar

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. Press<button>: Performs a 'Press' operation on the toolbar button specified.

b. select:<full_path_of_menu_item>: Selects the specified menu item from the toolBar.

c. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the specified X and Y coordinate.

d. doubleclick: Doubleclicks on the specified toolbar object.

e. Rightclick: Right clicks on the specified toolbar object.

f. popupmenu:<menuitem>: Selects a menu item from a shortcut (context) menu.

29. Treeview

Based on the values in arrKeyValue(0), the function performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. expand:<node_name>: Expands the node specified in arrKeyValue(1).

b. collapse:<node_name>: Collapses the node specified in arrKeyValue(1.

c. select:<node_name>: Selects the node specified in arrKeyValue(1) using the Select method

d. attach: Sets the focus to the currently attached Treeview object.

e. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the specified X and Y coordinate specified in arrKeyValue(1).

f. doubleclick: Double clicks on the specified Treeview object.

g. Rightclick: Right clicks on the specified Treeview object.

h. popupmenu:<menuitem>: Selects a menu item specified in arrKeyIndex(1) from a shortcut (context) menu.

i. selectrightclick:<item_name>: Right clicks on the specified item name specified in arrKeyIndex(1).

j. selecttreebutton:<item_name> : Simulates clicking on the directory list item name's button, specified in arrKeyValue(1), to expand or collapse the directory list.

k. Textselect:<value>: Selects the specified text in the object.

12. ProgressMeter

Based on the values in arrKeyValue(0), it performs different actions on objects. If the value in the variable arrKeyValue(0) is:

a. click: Performs click operation on the current object.

b. clickoncoordinate:<X_Value>:<Y_Value>: Simulates a mouse click on the X and Y coordinate specified in arrKeyValue(1).

If the value of arrObj(0) is not among the above listed, then the function will check for arrObj, which holds the value in the third column.

If the value of arrObj(0) is:

i) sqlexecute:

- Creates DB object dbConn
- Executes the query using execute method of DB object by passing strSQL variable as an argument
- Closes the DB object

ii) sqlvaluecapture:

- Calls the function Func_gfQuery() by passing the argument arrObj(1)
- Stores the value returned by the function in the variable in the fourth column of data table

iii) sqlcheckpoint:

- Sets the current row in the action sheet to 1
- Sets the TO property of the connection string
- Changes the DB Objects source(SQL) statement
- Executes the DB checkpoint

iv) sqlmultiplecapture:

- Sets the current row in the action sheet to 1
- Sets the TO property of the connection string
- Changes the DB Objects source(SQL) statement
- Executes the DB output checkpoint
- Returns value from the function stored in the variables specified in the output checkpoint

4.2. Store value Function

Name of the function: Func_Store()

Description: This function is used to store any property of a particular object into the specified variable.

Parameters:

a) Object – This is the object on which the specified operation needs to be performed.

b) arrObj – This parameter holds the type of the object from which the property has to be stored and the object name.

Assumptions: Context is set on the current object where the action has to be performed.

Variables:

a) strPropName – This variable is used to store the name of the property.

b) arrPropSplit – This array holds the property name and variable name.

Functionality:

- This function stores the value of the required object property in the specified variable.
- Using the function 'Func_ParamCheck()', system checks whether the number of parameters mentioned in the 'Action Value' column corresponds to the required number of parameters of the mentioned object for the Store function.
- If the number of parameters mentioned in Action Value column differs from the required number of parameters, the system exits out of the function.

- If the number of parameters mentioned in the Action Value column matches the required number of parameters, the system proceeds with the Store function.
- Based on the values in arrPropSplit(0) the function performs different actions. If the value is :

  i) itemscount: "items count" RO property of the object is stored in the mentioned variable.

  ii) focused: "Focus" property of the object is stored in the mentioned variable.

  iii) text: "Text" property of the object is stored in the mentioned variable.

  iv) focusitem: "Focusitem" property of the object is stored in the mentioned variable.

  v) checked: "Checked" property of the object is stored in the mentioned variable.

  vi) position: "Position" property of the object is stored in the mentioned variable.

  vii) multiline: "Multline" property of the object is either stored in the mentioned variable or compared with the expected value.

  viii) readonly: "Readonly" property of the object is stored in the mentioned variable.

  ix) exist: "exist" property of the object is stored in the mentioned variable.

  x) selectedtext: "Selectedtext" RO property of the object is stored in the mentioned variable.

  xi) enabled: "Enabled" RO property of the object is stored in the mentioned variable.

  xii) selectioncount: "selected items count" RO property of the object is stored in the mentioned variable.

  xiii) getcelldata: "GetCellText" method of the object is invoked by passing row number and column number variables as arguments. The return value is stored in the variable in the fifth column in the data table (arrPropSplit(1)).

  - If any of the above properties is not valid for any of the objects, then an error is thrown stating that the property is not supported.

- If the value in arrPropSplit(0) does not have any of the values listed above, then arrPropSplit(0) ROProperty of the object is stored in the value in the variable arrPropSplit(1).

4.3. Check Function

Name of the function: Func_Check()

Description: This function is used to check whether the value of the object's property is as expected.

Parameters:

a) Object – This is the object on which the specified operation needs to be performed.

b) arrObj – This parameter holds the type of the object and the object name on which the action has to be performed.

c) arrKeyValue – This parameter identifies the operation that needs to be performed on the object.

d) arrKeyIndex – This additional parameter is required to identify the object on which the operation needs to be performed. It holds the value of the specific action type and the value to be used.

e) intRowCount – This parameter holds the count of the current row in the data table.

Assumptions: Context is set on the current object where the action has to be performed.

Variables:

a) ActualValue: This variable stores the actual value of the object property.

b) ExpectedValue: This variable stores the expected value of the object property, fetched from Keyword Test Step.

c) reportStepPass: This variable stores the report pass value.

d) reportStepFail: This variable stores the report fail value.

e) strStatus: This variable stores the status of the report.

f) iStatus: This variable stores the status as Pass/Fail.

g) CheckDataIndex: This variable stores the index of the search string.

Functionality:

- This function checks whether the value of the required object property is as expected or not.
- Using the function 'Func_ParamCheck()', the system checks whether the number of parameters mentioned in the 'Action Value' column corresponds to the required number of parameters of the mentioned object for the Check function.
- If the number of parameters mentioned in the Action Value column differs from the required number of parameters, control exits out of the function.
- If the number of parameters mentioned in the Action Value column matches the required number of parameters, the system proceeds with the Check function.
- Based on the values in arrPropSplit(0) the function performs different checks. If the value is :
    a) Windowtext: Checks whether the specified window text is as expected or not.
    b) Enabled: Checks whether the specified object is enabled or not.
    c) Readonly: Checks whether the specified object is read only or not.
    d) Exists: Checks whether the specified object exists or not.
    e) Visible: Checks whether the specified object is visible or not.

f) Text: Checks whether the text of the object is as expected or not.

g) Innertext: Checks whether the inner text of the specified object is as expected or not.

h) Rowcount: Checks whether the row count of a particular table is as expected or not.

i) itemscount: Checks whether the itemcount is as expected or not.

j) Itemexist: Checks whether the specified item exists or not.

k) Itempresent: Checks whether the mentioned item is present in the specified object.

l) Textexist: Checks whether the specified text exists on the mentioned object.

m) Checked: Checks whether the mentioned object is checked or not.

n) Focused: Checks whether the mentioned object is in focus or not.

o) Ordercheck: Checks whether the mentioned item is present in the order as expected or not.

p) Tablesearch: Checks for the presence of a particular row and column in a specified table.

q) ColumnCount: Checks whether the value of the particular row's column count is as expected or not.

If any of the above properties is not valid for any of the objects, then an error is thrown stating that the property is not supported.

- If the actual value matches expected value, system would report the check as 'Passed.'
- If the actual value differs with the expected value, system would report the check as 'Failed.'

5. Functions for setting object

5.1. Function for setting the context:

Name of the function: Func_Context()

Description: This function is used to set the focus on the object on which some action has to be performed.

Parameters:

a) arrObj – This parameter holds the type of the object and the object name on which the action has to be performed.

b) intRowCount – This parameter holds the count of the current row in the data table.

Assumptions: The current object exists.

Variables:

a) curBrowser – This variable is used to store the type of the browser, given under 'ActionValue2' column.

Functionality:

- Based on the values in arrObj(0), Func_Context() sets the context on objects. If the value is:

1. Window

   Based on the value present in arrObj(1) the function sets context on the specified Window.

2. Browser

   Based on the value present in arrObj(1) the function sets context on the specified browser.

- By default or if the 'ActionValue2' column contains an entry 'ie', the system performs the action in Internet Explorer. If the 'ActionValue2' column contains the value 'mozilla', system performs the action in the Mozilla browser.

5.2. Function for setting the object:

Name of the function: Func_ObjectSet()

Description: This function is used to set the child object on which some action is to be performed.

Parameters:

a) arrObj – This parameter holds the class of the object and the object name on which the context has to be set.

b) intRowCount – This parameter holds the count of the current row in the data table.

Assumptions: The AUT is already up and running.

Variables:

a) curObjClassName – This variable is used to store the exact object name.

Functionality:

- The function will check for the value in the variable arrObj(0). If the value is “split,” “random,” “sqlvaluecapture,” “sqlexecute,” “sqlcheckpoint,” or "sqlmultiplecapture," the function will generate a fail report.
- If the value of arrObj(0) is :

  i) window: Sets the window name with the mentioned attach name as the current object

  ii) Anchor: Sets the anchor with the mentioned attach name as current object.

  iii) Area: Sets the area with the mentioned attach name as the current object.

  iv) browser: Sets the browser with the mentioned attach name as the current object

  v) listbox: Sets the listbox with the mentioned attach name as current object.

  vi) toolbar: Sets the toolbar with the mentioned attach name as current object.

  vii) button: Sets the button with the mentioned attach name as current object.

  viii) treeview: Sets the treeview with the mentioned attach name as current object.

  ix) Checkbox: Sets the checkbox with the mentioned attach name as current object.

  x) Div: Sets the div with the mentioned attach name as the current object.

  xi) Frame: Sets the frame with the mentioned attach name as the current object.

  xii) Inputfiletextbox: Sets the InputFileEditBox with the mentioned attach name as the current object.

  xiii) label: Sets the label with the mentioned attach name as the current object.

  xiv) listview: Sets the list view with the mentioned attach name as the current object.

  xv) menu: Sets the menu with the mentioned attach name as the current object.

  xvi) object: Sets the Mozilla object with the mentioned attach name as the current object.

  xvii) textbox: Sets the textbox with the mentioned attach name as the current object.

  xviii) checkbox: Sets the checkbox with the mentioned attach name as the current object.

  xix) radiobutton: Sets the radio button with the mentioned attach name as the current object.

  xx) combobox: Sets the combo box with the mentioned attach name as the current object.

  xxi) calendar: Sets the calendar with the mentioned attach name as the current object.

xxii) scrollbar: Sets the scrollbar with the mentioned attach name as the current object.

xxiii) Image: Sets the image with the mentioned attach name as the current object.

xxiv) Td: Sets the table data with the mentioned attach name as the current object.

xxv) Span: Sets the span with the mentioned attach name as the current object.

xxvi) Textarea: Sets the text area with the mentioned attach name as the current object.

xxvii) Viewlink: Sets the view link with the mentioned attach name as the current object.

xxviii) Colorpicker: Sets the Mozilla Colorpicker with the mentioned attach name as the current object.

xxix) Colorpickerbutton: Sets the Mozilla Colorpickerbutton with the mentioned attach name as the current object.

xxx) tabbar: Sets the tab bar with the mentioned attach name as the current object.

xxxi) table: Sets the table with the mentioned attach name as the current object.

xxxii) Popup: Sets the Mozilla popup with the mentioned attach name as the current object.

xxxiii) Progress Meter: Sets the Mozilla Progress meter with the mentioned attach name as the current object.

- By default or if the 'ActionValue2' column contains an entry 'ie', the system performs the action in Internet Explorer. If the 'ActionValue2' column contains the value 'mozilla,' the system performs the action in the Mozilla browser.

6. Reporting and Error-Handling Functions

6.1. Reporting Function:

Name of the function: Func_Report()

Description: This function is used for generating the customized report with specified user inputs through the keyword.

Parameters: None

Assumptions: NA

Variables:

a) reportobj – This variable is used to store the contents of the third column of the current row in the global sheet.

b) reportcon – This variable is used to store the status of the report (Pass/Fail).

c) reportconOne – This variable is used to store the actual message of the report.

d) reporterZero – This variable is used to store the expected message of the report.

e) expmess– This variable is used to store the concatenated expected message.

f) actmess – This variable is used to store the concatenated actual message.

g) reporterOne – This variable is used to store the split value of reportcon.

Functionality:

- 'reportobj' is split with delimiter ";" and is stored in the array 'reportcon.
- 'reportcon(0)' holds the status of the report.
- 'reportcon(1)' is split with delimiter "::" and is stored in the array 'reportconOne.'
- 'reportconOne(0)' is split with the delimiter ":" and is stored in the 'reporterZero.'
- 'reporterZero' holds the expected message.
- 'reportconOne(1)' is split with the delimiter ":" and is stored in the 'reporterOne.'
- 'reporterOne' holds the actual message.
- Based on the values in the 'reportcon(0)' the function will generate different reports. If the value is :

  i. Pass:

     1. Generates a report with status as Pass, expected message as 'reporterZero', and actual message as 'reporterOne'

  ii. Fail:

1. Generates a report with status as Fail, expected message as ‘reporterZero’, and actual message as ‘reporterOne’

6.2. Error-Handling Function:

Name of the function: Func_Error()

Description: This function is used to capture errors generated at runtime.

Parameters: NA

Assumptions: NA

Variables: NA

Functionality:

This function is used to capture the error generated at runtime. It generates a failure report with the error message.

7. String and Regular Expression Functions

7.1. Function for string operations:

Name of the function: Func_StringOperations()

Description: This function is used for all string operations.

Parameters:

a) strCriteria – This variable holds the value of the second column of the data table.

Assumptions: None

Variables:

a) arrSplit – This array is used to store the elements from the third column of the datatable after splitting with ";" delimiter.

b) strMainString – This variable is used to store the main string (arrSplit(0)).

c) strSubString – This variable is used to store the substring(arrSplit(1)).

d) intLen – This variable is used to store the length of the array "arrSplit".

e) ReturnVal – This variable is used to store the return value.

Functionality:

- Based on the values in strCriteria, the function performs different actions. If the value is:

  i) strsearch:

  1. Searches for the substring (strSubString) in the main string (strMainString).
  2. Stores the position of the substring in the return value variable (ReturnVal).

  ii) strconcat:

  1. Concatenates the main string (strMainString) and the substring (strSubString).
  2. Stores the concatenated string in the return value variable (ReturnVal).

  iii) strreplace:

  1. Searches for the substring (strSubString) in the main string (strMainString) and replaces it with strString (arrSplit(2)).
  2. Stores the replaced main string in the return value variable (ReturnVal).

- After the ReturnVal variable is updated, the value in the ReturnVal variable is stored in the variable specified in the fourth column of the data table.

7.2. Function for retrieving value from a variable:

Name of the function: GetValue()

Description: This function is used to retrieve the value from any variable.

Parameters:

strCellData – This parameter holds the name of the variable.

Assumptions: None

Variables:

a. arrSplitCheckData – This variable is used to store the elements after the value is split with "_" delimiter.

b. strParamName – This variable is used to store the 2nd element of the array 'arrSplitCheckData'.

Functionality:

The function first checks if the variable name, specified in strCellData, starts with a "#","env_" or "dt_".

If the variable starts with:

a. "#"- The function truncates the "#" from the variable name and returns the value of the environment variable.

b. "env_"- The function truncates "env_" from the variable name and returns the value of the environment variable.

c. "dt_" – The function truncates "dt_" from the variable name and reads the value of the variable from the specified data sheet (name of the data sheet to be passed as an additional parameter in the Keyword_driver function).

7.3. Function for Press Key Operations:

Name of the function: Func_presskey()

Description: This function is used to retrieve the value from any variable.

Parameters:

a) arrObj – This variable holds the value of the third column in the data table.

b) WshShell – This is the object created for shell scripting.

Assumptions: NA

Variables: NA

Functionality:

- WshShell object is created.
- Based on the values in arrObj(0), different values are passed as arguments to the SendKeys method of the created shell scripting object. Shell object is set to 'nothing'.

7.4. Condition function

Name of the function: Func_Condition()

Description: This function is used to evaluate the expression according to the inputs given in the keyword script.

Parameters:

a) intRowCount – This holds the value of the current row number of the data table.

Assumptions: NA

Variables:

a) iFlag– This variable is used to set the flag.

b) cndSplit – This variable is used to store the value of the fourth column of the global sheet.

c) startRow – This variable is used to store the start row for the condition.

d) endRow – This variable is used to store the end row for the condition.

e) cstrCellData – This variable is used to store the condition to be checked.

f) varOne – This variable is used to store the first element to be evaluated.

g) varTwo – This variable is used to store the second element to be evaluated.

Functionality:

- cndSplit array is stored with values in the fourth column of the data table after splitting with the delimiter ";".
- The variable startRow is assigned with the value of the first item in the array cndSplit.
- The variable endRow is assigned with the value of the second item in the array cndSplit.
- cstrCellData array is stored with values in the third column of the data table after splitting with the delimiter ";".
- The variable varOne is assigned with the value of the first item in the array cstrCellData.
- The variable varTwo is assigned with the value of the third item in the array cstrCellData.
- Then the condition in the cstrCellData(1) is evaluated and the intRowCount is assigned with the value in startRow if the condition is true. Otherwise, intRowCount is assigned with the value in the endRow if the condition is false.

7.5. Loop Function

Name of the function: Func_loop()

Description: This function is used to repeat a set of statements for a specified number of times.

Parameters:

a) variable – This holds the query variable that has to be converted.

b) strconverttype – This holds the data type into which the variable is to be converted.

Assumptions: If the number of times to be looped is not specified, by default this number is taken as the number of active rows in the Action1 sheet of the data table.

Variables:

a) arrloopData – This variable is used to store the start row and end row values.

b) intcntr – This variable is used to store the loop count.

c) Counter – This variable is used to store the count value.

d) endRow1 – This variable stores the end row for looping.

e) loopRowCount – This variable stores the current loop count

f) intDataCounter – This variable stores the current data counter.

Functionality:

- arrloopData is stored with the values after splitting the value in the third column of the data table with ";" as delimiter.
- The variable intcntr is assigned with the value in the fourth column in the data table.
- The value in the variable intcntr is converted into an integer and stored in the variable counter.
- The value in the variable intcntr is stored into the variable Counter.

This function recursively calls the Keyword_Web function (Main function) 'n' times. Here 'n' is the value present in the variable Counter.

7.6. Function for Querying Database:

Name of the function: Func_gfQuery()

Description: This function is used to query the database.

Parameters:

a) strSQL – This holds the query that needs to be executed.

Assumptions: The connection string is specified and a connection is established with the database.

Variables:

a) dbConn – This variable is used to store the database connection object.

b) dbRS – This variable holds the results of the database operation performed.

c) connectionString – This variable stores the connection string for the database.

d) dbUID – This variable is used to store the user name to connect to the database.

e) dbPWD – This variable is used to store the password to connect to the database.

f) dbServer – This variable is used to store the database server name.

g) dbHost – This variable is used to store the database host name.

h) dbDRIVER – This variable is used to store the database driver name.

Functionality:

- Creates DB object dbConn
- Calls the method open for the dbConn object, passing the environment value of the variable connection String as argument
- Calls the execute method by passing the variable strSQL as an argument
- Returns the items retrieved from the database after querying using the execute method
- Closes the object and sets dbConn to 'Nothing'

7.7. Function for Arithmetic Operations:

Name of the function: Func_arith()

Description: This function is used to perform arithmetic operations.

Parameters:

a) strX – This variable is used to store the input values specified in the keyword script.

b) strY – This variable is used to store the output value of the function in a variable specified in the keyword script.

Assumptions: NA

Variables:

a) arrSplit1 – This variable is used to store the arithmetic equation.

b) intz – This variable is used to store the flag return value.

Functionality:

- This function will search for '#' in variable strX. If '#' is not present then it will call the eval function, passing the variable strX as an argument. The return value is assigned to the environment variable in the fourth column of data sheet.
- If '#' is present, then the function will search for '+', '*', '/', or '–' in the variable strX.
- If the Value in strX is :

  i) '+': strSplit array is stored with the values in strX after splitting with the delimiter '+'. Then it will retrieve the environment values if they start with '#'. Then the strX variable is replaced with values in the variables strSplit(0) & strSplit(1). Then it will call the eval function by passing the variable strX as an argument. The return value is assigned to the environment variable in the fourth column of the data sheet

  ii) '*': strSplit array is stored with the values in strX after splitting with the delimiter '*'. Then the function will

retrieve the environment values if they start with '#'. Then the strX variable is replaced with values in the variables strSplit(0) & strSplit(1). Then the function will call the eval function by passing the variable strX as an argument. The return value is assigned to the environment variable in the fourth column of the data sheet.

iii) '/': strSplit array is stored with the values in strX after splitting with the delimiter '/'. Then the function will retrieve the environment values if they start with '#'. Then the strX variable is replaced with values in the variables strSplit(0) & strSplit(1). Then it will call the eval function by passing the variable strX as argument. The return value is assigned to the environment variable in the fourth column of the data sheet.

iv) '-': strSplit array is stored with the values in strX after splitting with the delimiter '-'. Then the function will retrieve the environment values if they start with '#'. Then the strX variable is replaced with values in the variables strSplit(0) and strSplit(1). Then it will call the eval function by passing the variable strX as an argument. The return value is assigned to the environment variable in the fourth column of the data sheet.

7.8. Function for Converting Data Types:

Name of the function: Func_Convert()

Description: This function is used to query the database.

Parameters:

a) variable - This parameter holds the variable that has to be converted.

b) strconverttype - This holds the data type into which the variable is to be converted.

Assumptions: NA

Variables:

a) strObject- This variable is used to store the variable to be converted.

b) arrConvert - This variable is used to store elements such as the conversion type and variable it is to be stored in.

c) strObjectOne - This variable is used to store the converted value.

Functionality:

- This function searches for '#' in the variable strObject.
- If '#' is present, the strObjectOne variable will be assigned with the environment value of strObject; otherwise, the strObjectOne variable will be assigned with the value of the variable strObject.
- It will store the values in the variable strconverttype after splitting the delimiter ':' into the array arrConvert.
- Based on the values in arrConvert(0), the function will perform different actions. If the value is:

i) 'date': It will convert the value in the strObjectOne variable based on the format available in arrConvert(2).

ii) 'roundto': It will round the value in the variable strObjectOne and store it in the environment value of arrConvert(1).

iii) 'lcase': It will convert the value in strObjectOne into lowercase and store it in the environment value of arrConvert(1).

iv) 'ucase': It will convert the value in strObjectOne into uppercase and store it in the environment value of arrConvert(1).

v) 'cstr': It will convert the datatype of the value in strObjectOne into the string datatype and store the converted value in arrConvert(1).

vi) 'ascii': It will convert the value in strObjectOne into ASCII value and store the converted value in arrConvert(1).

vii) 'trim': It will remove the extra spaces in the value in the variable strObjectOne.

viii) 'len': It will return the length of the value in the variable strObjectOne and store the returned length in the variable arrConvert(1).

7.9. Function to check the input parameters

Name of the Function: Func_ParamCheck ()

Description: This function is used to verify whether the expected number of parameters is provided under the 'Action Value' column of Keyword TC for Perform, Check, and Store keywords.

Parameters:

a) arrObj – This parameter holds the type of the object and the object name on which the action has to be performed.

b) arrKeyValue – This parameter identifies the operation that needs to be performed on the object.

c) arrKeyIndex – This additional parameter is required to identify the object on which the operation needs to be performed. It holds the value of the specific action type and the value to be used.

d) intRowCount – This parameter holds the count of the current row in the data table.

Assumptions: NA

Variables:

a) Invalid Number: This variable denotes whether the expected number of parameters is provided under the 'Action Value' column of Keyword TC.

Functionality:

- System compares the provided number of parameters with the defined number of parameters of the corresponding function.
- If the provided number of parameters differs from the expected number of parameters, the system stores the value '0' in the 'Invalid Number' variable.

- If the value of the 'Invalid Number' variable is '0', the system fails the user check with the message stating "Invalid no of parameters entered."

8. Table Operation Functions

8.1 Function for Table Search:

Name of the function: Func_tablesearch()

Description: This function is used to search for the particular row and column in the table based on the two search criteria entered in the keyword script for perform keyword. It is also used to check for text present in a table based on the two search criteria entered in the keyword script for check keyword.

Parameters:

a) object – Holds the type of the object and the object name on which action should be performed or checked

b) strSearch – Holds the search criteria entered in the keyword script (e.g. <colname1>;<colValue2>::<colname1>;<colValue2>--5)

c) strOutVarOne – Stores the number of the output row when the 'perform' keyword is used

d) strOutVar2 – Stores the number of the output column when the 'perform' keyword is used.

Assumptions: NA

Variables:

a) intCheck – Stores the row number in the table where the text is present.

b) arrCol – Stores the number of columns.

c) strSearch – Stores the value present in the fourth column.

Functionality:

• Based on the values in arrKeyValue(0), the Func_tablesearch() function performs different actions.

8.2 Function for Retrieving Row Number:

Name of the function: Func_getRowNum ()

Description: This function is used to retrieve the row number in which specified text is present in table. This function is used for 'rownum' keyword.

Parameters:

a) object – Holds the type of the object and the object name on which action should be performed or checked.

b) strSearch – Holds the search criteria entered in the keyword script (e.g., <colname1>;<colValue2>::<colname1>;<colValue2>--5).

c) strReturnVal – Stores the output row number for perform keyword.

Assumptions: NA

Variables:

a) arrCol – Stores the number of columns

b) strSearch – Stores the value present in the fourth column

Functionality:

- Based on the values in arrKeyValue(0), the Func_getRowNum() function performs different actions.

9. Common Functions

9.1 Function for FSO Operations

Name of the function:

Func_CommonFunctions(strType,strDetailsOne,StrDeatailsTwo,intRowCount)

Description: This function is used to perform a set of operations using the file system objects.

Parameters:

a) strType –This holds the type of object being used for FSO such as File or Folder.

b) strDetailsOne & StrDetailsTwo – These variables holds the details to be used while using FSO.

c) intRowCount – This holds the current row count in the data table.

Assumptions: NA

Variables:

a) strFuncType – This variable is used to store the type of object(Ex:File,Folder) to be used.

b) strFuncDetailsOne – This variable is used to store the details of the object (Ex:Folder name,Folder Path).

c) strFuncDetailsTwo – This variable is used to store the Actionvalue2 details of the script sheet

Functionality:

- This function assigns the value in the variable strType to the variable strFuncType.
- This function assigns the value in the variable strDetails to the variable strFuncDetails.
- Based on the values in the variable strFuncType, this function performs different actions. If the value is :

i) folder: It will call the function Func_Folder() while passing the variable strFuncDetailsOne as argument.

ii) file: It will call the function Func_File() while passing the variable strFuncDetailsOne as argument.

iii) exportxml: It will call the function Func_ExportXML() while passing the variables strFucnDetailsOne and strFuncDetailsTwo as arguments.

iv) deletexml: It will call the function Func_DeleteXML() while passing the variable strFuncDetailsOne as an argument.

v) callcheck: It will call the function Func_CallCheck() while passing the variable strFuncDetailsOne as an argument.

9.2 Function for Folder Operations

Name of the function: Func_Folder()

Description: This function is used to work on folders using FSO.

Parameters:

a) pCellData – This holds the details to be used while using FSO.

Assumptions: NA

Variables:

a) arrFolderpath – This variable is used to store the elements of the folder path separated by delimiter "\".

b) intFolderloc – This variable is used to store the location of the folder name.

c) DestFolder – This variable is used to store the destination folder.

d) Foldername – This variable is used to store the folder name.

e) oFSO – This variable is used to store the created object.

f) arrCellData – This variable is used to store the details of the operation to be performed

g) oFolder – This variable is used to store the details of the object created.

Functionality:

- The value in the variable pCellData is split with the delimiter ";" and is stored in the array arrCellData.
- Based on the values in the arrCellData(0) the function will perform different actions. If the value is:

i) create:

- If the folder with the name arrCelldata(1) is present, then a report is generated stating that the folder already exists.
- If the folder is not present, then a new folder with the name in arrCellData(1) is created.

ii) delete:

- If the folder with the name arrCelldata(1) is not present, then a report is generated stating that the folder is not present.
- If the folder is present, then the folder is deleted.

iii) copy:

- If the folder with the name arrCelldata(1) is not present, then a report is generated stating that the folder does not exist.
- If the folder is present, it is copied to the location present in the variable arrCelldata(2).

iv) move:

- If the folder with the name arrCelldata(1) is not present, then a report is generated stating that the folder does not exist.
- If the folder is present, it is moved to the location present in the variable arrCelldata(2).

9.3 Function for File Operations

Name of the function: Func_File()

Description: This function is used for working with files by using FSO.

Parameters:

a) pCellData – This holds the details to be used while using FSO.

Assumptions: NA

Variables:

a) arrFilepath – This variable is used to store the file path.

b) DestFile – This variable is used to store the destination file name.

c) strFilename – This variable is used to store the file name to be used.

d) intFileLoc – This variable is used to store the location of the file name.

e) iFSO – This variable is used to store the created object.

f) oFile – This variable is used to store the details of created object.

g) arrCellDataOne – This variable is used to store the details of the operation to be performed.

h) intf – This variable is used for looping.

i) strMess – This variable is used to store the string, which has to be written into a file.

Functionality:

- The value in the variable pCellData is split with the delimiter ";" and is stored in the array arrCelldataOne.
- Based on the values in the arrCellDataOne(0), the function will perform different actions. If the value is:

i) create:

  - If the file with the name arrCellDataOne(1) is present, then a report is generated stating that the file already exists.
  - If the file is not present, a new file with the name in arrCellDataOne(1) is created.

ii) delete:

  - If the file with the name arrCellDataOne(1) is not present, then a report is generated stating that the file is not present.
  - If the file is present, then the file is deleted.

iii) copy:

  - If the file with the name arrCelldataOne(1) is not present, then a report is generated stating that the file does not exist.
  - If the file is present, it is copied to the location present in the variable arrCelldataOne(2).

iv) move:

- If the file with the name arrCelldataOne(1)is not present, then a report is generated stating that the file does not exist.
- If the file is present, it is moved to the location present in the variable arrCelldataOne(2).

v) write:

- If the file with the name arrCelldataOne[1] is not present, then a report is generated stating that the file does not exist.
- If the file is present, then required text is written in the file.

vi) read:

- If the file with the name arrCelldataOne(1) is not present, then a report is generated stating that the file does not exist.
- If the file is present, then required line is read from the file.

vii) append:

- If the file with the name arrCelldataOne(1) is not present, then a report is generated stating that the file does not exist.
- If the file is present, then it opens the file in append mode(i.e., write = true).

9.4 Function for Exporting XMLs

Name of the function: Func_ExportXML()

Description: This function is used to export data and store it in XML format.

Parameters:

a) strDetailsOne – This holds the details to be used while exporting in XML format.

b) sPath – This holds the path where the XML file has to be stored.

Assumptions: NA

Variables:

a) arrDocSplit – This variable is used to store the elements to be exported to XML and the document name.

b) strDocName – This variable is used to store the document name.

c) arrElementSplit – This variable is used to store the variables and the tag names to be exported to XML.

d) arrElementName – This variable is used to store the current variable and its tag names to be exported to XML.

e) oDoc – This variable is used to store the XML object.

Functionality:

- The value in the variable strDetailsOne is split with the delimiter ";" and is stored into the array arrDocSplit.
- XML file Object oDoc is created.
- The XML file with the name in the variable 'docname' is created by using the method CreateDocument.
- All the data present in arrDocSplit(1) is exported into the above created file by using the AddChildElementByName method.
- The XML file is saved in the path available in the variable 'sPath'.
- oDoc object is set to Nothing.

9.5 Function for Deleting XMLs

Name of the function: Func_DeleteXML()

Description: This function is used to delete XML files.

Parameters:

a) sPath – this holds the path in which the XML file is present.

Assumptions: NA

Variables:

a) dFileObj – This variable is used to store the XML file to be used.

b) dFSO – This variable is used to store the XML object.

Functionality:

- The File System Object 'dFSO' object is created.
- The XML file in the path specified in the variable 'sPath' is accessed using the GetFile method.
- Using the delete method, the above file is deleted.

  dFSO object is set to Nothing.

9.6 Function for Callchecks for functions.

Name of the function: Func_CallCheck()

Description: This function is used if the user wants to check standard checks.

Parameters:

b) CheckToExecute– This holds the name of the standard check to be executed.

Assumptions: The check is already created and saved in the "Check" section of Testpartner.

Variables:

Functionality:

This function is used to execute all the standard checks provided by Testpartner (E.g., Bitmap check, content check, property check, text check, etc.).

10 User-Defined Functions

10.1 Function for 'CallFunction' Keyword

Name of the function: Func_FunctionCall()

Description: This is the template for users to create their own functions.

Parameters:

a) strFunName – The function name to be used.

b) StrParameters – The parameters to be used with the called function.

Assumptions: NA

Variables:

a) arrActionParam – This variable is used to store the Action Parameters.

b) intActionParamCount – This variable is used to store the number of parameters.

c) strInfo – This variable is used to store all the parameters to be used in the functions.

d) inta – This variable is used for looping.