



# Open2Test Mainframe Test Automation Framework for QTP

Version 1.0

June 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**

<b>1. PURPOSE OF THE DOCUMENT</b> .....	<b>4</b>
1.1. Scope .....	4
1.2. Overview .....	4
<b>2. FRAMEWORK CODE STRUCTURE</b> .....	<b>5</b>
<b>3. DRIVER FUNCTIONS</b> .....	<b>6</b>
3.1. Keyword Driver Function .....	6
3.2. Main Function .....	6
<b>4. ACTION FUNCTIONS</b> .....	<b>7</b>
4.1. Perform Function .....	7
4.2. Store Function .....	8
4.3. Check Function .....	9
<b>5. FUNCTIONS FOR SETTING OBJECT</b> .....	<b>10</b>
5.1. Function for setting the context: .....	10
5.2. Function for setting the object: .....	<b>Error! Bookmark not defined.</b>
<b>6. REPORTING AND ERROR-HANDLING FUNCTIONS</b> .....	<b>11</b>
6.1. Reporting Function: .....	11
6.2. Error-handling Function: .....	12
<b>7. STRING AND REGULAR EXPRESSION FUNCTIONS</b> .....	<b>13</b>
7.1. Function for string operations: .....	13
7.2. Function for Regular Expression Test: ..	<b>Error! Bookmark not defined.</b>
7.3. Function for Regular Expression Match: ..	<b>Error! Bookmark not defined.</b>
7.4. Function for Regular Expression Replace: .....	<b>Error! Bookmark not defined.</b>
<b>8. TABLE OPERATION FUNCTIONS</b> .....	<b>ERROR! BOOKMARK NOT DEFINED.</b>
8.1. Function for Table Search: .....	<b>Error! Bookmark not defined.</b>
8.2. Function for Retrieving Row Number: ...	<b>Error! Bookmark not defined.</b>
<b>9. COMMON FUNCTIONS</b> .....	<b>14</b>
9.1. Function for Retrieving Variables: ....	<b>Error! Bookmark not defined.</b>
9.2. Function for Press Key Operations: ....	<b>Error! Bookmark not defined.</b>
9.3. Function for Dynamic Wait: .....	14
9.4. Function for Arithmetic Operations: .....	14
9.5. Function for Querying Database: .....	<b>Error! Bookmark not defined.</b>
9.6. Function for Converting Data types: ...	<b>Error! Bookmark not defined.</b>

9.7.	Function for Importing Keyword Script from Excel Sheet .....	15
9.8.	Function for FSO Operations .....	<b>Error! Bookmark not defined.</b>
9.9.	Function for Folder Operations .....	<b>Error! Bookmark not defined.</b>
9.10.	Function for File Operations .....	<b>Error! Bookmark not defined.</b>
9.11.	Function for Exporting XMLs .....	<b>Error! Bookmark not defined.</b>
9.12.	Function for Deleting XMLs .....	<b>Error! Bookmark not defined.</b>
<b>10.</b>	<b>REUSABLE FUNCTIONS .....</b>	<b>16</b>
10.1.	Function for Calling Reusable Actions .....	16
<b>11.</b>	<b>CONDITION AND LOOPING FUNCTIONS .....</b>	<b>17</b>
11.1.	Condition function .....	17
11.2.	Loop Function .....	17
<b>12.</b>	<b>DEBUG FUNCTIONS .....</b>	<b>ERROR! BOOKMARK NOT DEFINED.</b>
12.1.	Function for Debugging: .....	<b>Error! Bookmark not defined.</b>

## 1. Purpose of the Document

The purpose of this document is to describe the Open2Test Test Automation Framework code in detail.

### 1.1. Scope

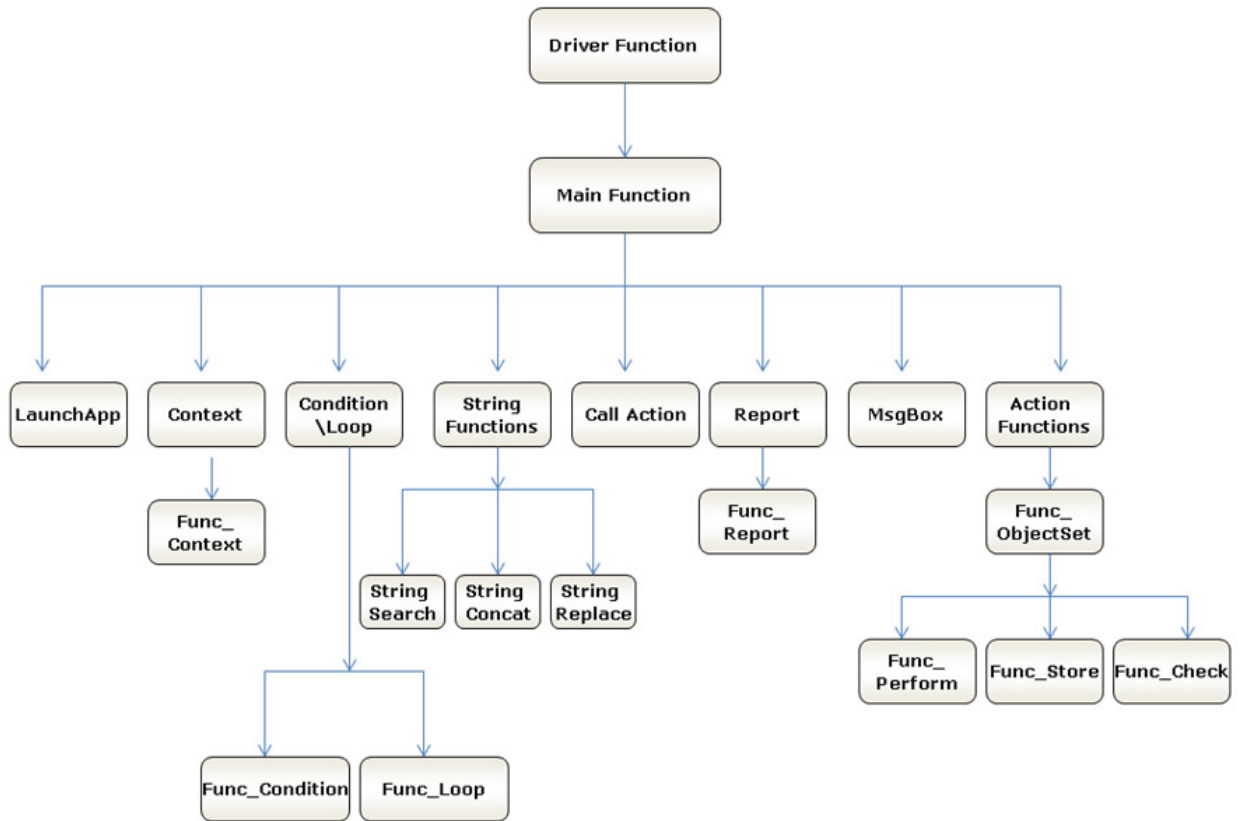
The scope of this document is to provide details about Open2Test Test Automation Framework code and its architecture and functions.

### 1.2. Overview

This document provides details about:

- Framework Architecture
- Driver Functions
- Action Functions
- Reusable 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:** NA

**Assumptions:** The Automation Script is present in the Global Sheet of QTP.

**Variables:**

- a) intRowCount - Loops through all the data table rows
- b) intDataCounter - Stores the iteration count for looping
- c) intSheet - Checks whether a keyword script is present in the Global Sheet

**Functionality:**

- Reads the values in the first column of the Global Sheet.
- Whenever the value is 'r', it calls the main function (Keyword\_MF).
- When 'r' is not present in any of the cells in the first column, it skips the row and reads 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\_MF()

**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 Global Sheet of QTP.

**Functionality:**

- Reads the values in the second, third, and fourth columns in the data sheet and perform action
- Based on the value in the 2<sup>nd</sup> column, keyword\_MF() calls different functions
- 3<sup>rd</sup> column value refers to object name in Object repository
- 4<sup>th</sup> column value refers to a value and it will be mapped to either Test Data file or given value in Global Sheet

## 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 AUT.

**Parameters:** NA

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

**Variables:**

- a) curObject - Stores the values from Column 3 value of data sheet after splitting it with delimiter ";"
- b) curObjClassName - Stores the object name from the curObject variable
- c) curObjPerform - Stores the data in the fourth column of the data sheet
- d) dCellData - Stores the split array of the fourth column of the data sheet
- e) envCheck - Stores left most 3 characters of value from dCellData(0)

**Functionality:**

- Based on the values in dCellData(0) Func\_Perform () performs different actions. If the value is:
  - i) set: Checks the value in curObject(0). If the value is 'field' then the value in dCellData(1) is set based on value in a variable envCheck. If envCheck is 'dt\_' then value will be retrieved from Test Data file and if value is 'env' then value will be retrieved from Environment variable.
  - ii) close: Checks for the value in the variable curObject(0). If the value is "Window" then it will perform a close operation on the parent object.
  - iii) Activate: Checks for the value in the variable curObject(0). If the value is "Window" then it will perform a Activate operation on the parent object.
  - iv) Click: Check the value in curObject(0). If the value is 'SetCursorPos' then the cursor will be set to current object and it will be clicked
  - v) PF1 - PF24: Checks for the value in the variable curObject(0). If the value is "SendKey" then performs sending function keys operations based on value in dCellData(0). The object must be in focus for this operation.
  - vi) Enter: Checks for the value in the variable curObject(0). If the value is "SendKey" then performs 'Enter' operation.
  - vii) If the value of curObjPerform(0) is not among the above listed then test will be terminated

## 4.2. Store Function

**Name of the function:** Func\_Store ()

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

**Parameters:**

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

**Variables:**

- a) strPropName - Stores the name of the property
- b) arrPropSplit - Holds the property and variable names
- c) intGRowNum - Stores the row number
- d) intGColNum - Stores the column number

**Functionality:**

- Based on the values in arrPropSplit(0), it performs different actions. If the value is :
  - i) itemscount: "items count" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - ii) enabled: "disabled" RO property of the object is captured and converted to Boolean data type. Then negation of the value retrieved is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - iii) columncount: Columncount property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - iv) rowcount: Rowcount property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - v) filename: "file name" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - vi) imagetype: "image type" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - vii) defaultvalue: "default value" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - viii) maxlength: "max length" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - ix) allitems: "all items" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - x) selectiontype: "select type" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - xi) exist: "exist" property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).

- xii) selectioncount: "selected items count" RO property of the object is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
  - xiii) getcelldata: "GetCellData" method of the object is invoked by passing intGRowNum (Row No) and intGColNum (Column No) variables as arguments. The return value is stored in the variable in the fourth column in the data table (arrPropSplit(1)).
- 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 for all the checking operations to be performed on the AUT.

**Parameters:** None

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

**Variables:**

checkObject - Stores the value in third column

checkObjClass - Stores the object class name from split value of checkObject

checkObjClassName - Stores the object name from split value of checkObject

checkVal - Stores the value from fourth column

dCheckData - Stores the split value in an array from checkVal variable

**Functionality:**

- Based on the values in dCheckData it performs different actions. If the value is:
  - i) exist: If the dCheckData(1) value is true, then the function checks whether the current object exists or not. If it exists, the Func\_Check() function will generate a pass report. Otherwise it will generate a fail report. If the dCheckData(1) value is false, the function checks whether the current object exists or not. If it exists, it will generate a fail report. Otherwise it will generate a pass report. This exist functionality works on different object classes like 'Window', 'Screen', 'Field'

## 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 full hierarchical path for the object on which some action is to be performed.

**Parameters:** None

**Assumptions:** AUT is already up and running.

**Variables:**

- a) curContext - Stores the contents (Object Class & Object Name) of the third column of the current row in the Global Sheet
- b) curWnd - Stores the Object name from the variable curContext(1)

**Functionality:**

- Based on the values on curContext(0), the Func\_Context () function will set the context on different objects. If the value is:
  - i) Window: Sets the current context of object with "Window" Class with the name in the variable curWnd
  - ii) Screen: Sets the curContext(1) object with "Screen" Class with the name in the variable curWnd

## 6. Reporting and Error-handling Functions

### 6.1. Reporting Function:

**Name of the function:** Func\_Report ()

**Description:** This function is used for generating a customized report with specified user inputs through the use of keywords.

**Parameters:** None

**Assumptions:** NA

**Variables:**

- a) reportobj - Stores the contents of the third column of the current row in the Global Sheet
- b) reportcon - Stores the status of the report (Pass/Fail)
- c) reportcon1 - Stores the actual message of the report
- d) reporter0 - Stores the expected message of the report
- e) expmess - Stores the concatenated expected message
- f) actmess - Stores the concatenated actual message

**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 'reportcon1'.
- 'reportcon1(0)' is split with the delimiter ":" and is stored in the 'reporter0'.
- 'reporter0' holds the expected message.
- 'reportcon1(1)' is split with the delimiter ":" and is stored in the 'reporter1'.
- 'reporter1' holds the actual message.
- Based on the values in the 'reportcon(0)', the Func\_Report () function will generate different reports. If the value is :
  - i) Pass:
    - Generates a report with status as Pass, expected message as 'reporter0', and actual message as 'reporter1'
  - ii) Fail:
    - Generates a report with status as Fail, expected message as 'reporter0', and actual message as 'reporter1'
  - iii) Done:
    - Generates a report with status as Done, expected message as 'reporter0', and actual message as 'reporter1'

iv) Warning:

- Generates a report with status as Warning, expected message as 'reporter0', and actual message as 'reporter1'

## 6.2. Error-handling Function:

**Name of the function:** Func\_Error ()

**Description:** This function is used to capture the error generated at runtime.

**Parameters:** NA

**Assumptions:** NA

**Variables:**

- a) strError - This variable is used to store the value present in the fifth column of the current row in the data sheet.

**Functionality:**

This function checks for the Err.Number after processing each keyword line. When ever the error number is not equal to '0', it will generate a fail report. It also checks for the strError variable, which holds the value of the fifth column of the current row in the data table. Whenever the value is 'onfailureexit' and the value of the variable keyword is '1' , the Func\_Error () function will exit the test.

## 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 - Stores the elements from the third column of the data table after splitting with the ";" delimiter
- b) strMainString - Stores the main string (arrSplit(0))
- c) strSubString - Stores the sub string(arrSplit(1))
- d) intLen - Stores the length of the array "arrSplit"
- e) ReturnVal - Stores the return value

**Functionality:**

- Based on the values in strCriteria, Func\_StringOperations () performs different actions. If the value is:
  - i) strsearch:
    1. Searches for the sub string (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 sub string (strSubString)
    2. Stores the concatenated string in the return value variable (ReturnVal)
  - iii) strreplace:
    1. Searches for the sub string (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.

## 8. Common Functions

### 8.1. Function for Wait:

**Name of the function:** Func\_Wait ()

**Description:** This function is used for synchronization with the application.

**Parameters:** NA

**Assumptions:** NA

**Variables:** NA

**Functionality:**

- A function check for the wait time provided in Column 3 in Global spreadsheet

### 8.2. Function for Arithmetic Operations:

**Name of the function:** Func\_arith ()

**Description:** This function is used to perform addition (+) and subtraction (-) operations.

**Parameters:**

- a) strX - This variable is used as 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 the data sheet.
- If '#' is present then the function will search for '+' or '\*' or '/' 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) and strSplit(1). Then the Func\_arith () 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.

- ii) `'*'`: strSplit array is stored with the values in strX after splitting with the delimiter `'*'`. Then the Func\_arith () 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.
- iii) `'/'`: strSplit array is stored with the values in strX after splitting with the delimiter `'/'`. Then the Func\_arith () 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.
- iv) `'-'`: strSplit array is stored with the values in strX after splitting with the delimiter `'-'`. Then the Func\_arith () 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.

### 8.3. Function for Importing Keyword Script from Excel Sheet

**Name of the function:** Func\_ImportData ()

**Description:** This function is used to import test data at runtime.

**Parameters:** NA

**Assumptions:** The required file is present and it is an Excel sheet.

**Variables:**

- a) strPath - Stores the Path with .xls
- b) strDataPath - Locates and stores the full Path
- c) strSheetName - Stores the sheet name to be imported

**Functionality:**

- This function will store the value of the fourth column in the variable 'strSheetName'.
- Using the ImportSheet method, the Excel sheet with the name in the variable 'strSheetName' in the Excel work book in the path specified by the variable 'strDataPath' is imported to the Action1 sheet.

## 9. Reusable Functions

### 9.1. Function for Calling Reusable Actions

**Name of the function:** Func\_CallAction ()

**Description:** This function is used to call a Re-usable Action.

**Parameters:**

- a) strData - Holds the name of the reusable action.
- b) strInfo - Holds the parameters for the reusable action.

**Assumptions:** NA

**Variables:**

- a) arrParam - Stores the parameters to be passed to the reusable action
- b) strActionName - Stores the reusable action name

**Functionality:**

- This function checks for the value in the variable strInfo.
- If the value is Null then the Func\_CallAction () function will call the RunAction method by passing variable 'actionName' and value 'oneIteration' as arguments (without passing parameters).
- If the value is not Null then the Func\_CallAction () function will split the value in the variable strInfo with the delimiter ':' and store it in array 'paramSplit'.
- Based on the number of items in array paramSplit, this function performs different actions.
- For example, if the number of items in paramSplit is 4, then 4 parameters are passed while calling the RunAction method.

## 10. Condition and Looping Functions

### 10.1. Condition function

**Name of the function:** Func\_Condition ()

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

**Parameters:NA**

**Assumptions:** NA

**Variables:**

- a) cndSplit - Stores the value of the third or fourth column(for Else statements) of the Global Sheet
- b) startRow - Stores the start row for the condition
- c) endRow - Stores the end row for the condition
- d) resumeRow - Stores the resume row for the condition

**Functionality:**

- cndSplit array is stored with values in the third or 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.

### 10.2. 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: N/A**

**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 - Stores the start row and end row values
- b) intRN - Starting row of data in test data sheet
- c) intRC - Stores the loop count from Test Data sheet

**Functionality:**

- arrloopData is stored with the values after splitting the value in the third column of the data table with “;” as a delimiter
- The variable intRC is calculated value from the test data sheet and it will count number of rows of data, so loop will be iterated for that many times

- This function recursively calls Keyword\_Mainframes function (Main function) 'n' times. Here, 'n' is the value present in the row count.

---

**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.*