Open2Test Test Automation Framework

Version 3.0

Jan 2014

TABLE OF CONTENTS

1. PREFACE 3

2. OPEN2TEST FRAMEWORK OVERVIEW 4

2.1. Introduction to Open2Test 4

2.2. Framework Features 4

2.3. Open2Test Benefits 5

3. FRAMEWORK ARCHITECTURE 6

3.1. Framework Architecture 6

3.1.1. Driver Script 6

3.1.2. Function Library 7

3.1.3. External Test Data & Variables 7

3.1.4. Object Repository 7

3.1.5. Reporting 7

4. FLOW DIAGRAM OF THE OPEN2TEST TEST AUTOMATION FRAMEWORK 8

5. STRUCTURE OF THE SCRIPT 9

5.1. Columns 9

5.2. Role of Delimiters 10

5.3. Variables 11

5.4. Sequence of keywords 11

1. Preface

The aim of Test Automation is to get increased efficiency of the resources, test coverage, and quality & reliability of the software and also to reduce the overall cycle time.

While there are several frameworks that provide support for automated software testing, this document introduces one particularly effective type called the **Open2Test.**

In this framework, the discrete functional business events that make up an application are described using keywords. This approach fosters code reusability, optimum utilization of the tool, and greater productivity.

2. Open2Test Framework Overview

2.1. Introduction to Open2Test

A framework imposes a structure that contains libraries, reusable components, and related documentation as a unit that supports a given objective/solution. There are various frameworks available for test automation. Open2Test uses the keyword-driven approach.

Keyword-based approach is based on the idea that the discrete functional business events that make up any application can be described using short text descriptions (English like keywords). Using these keywords, testers begin to build a common library of keywords that can be used to create test scripts/scenarios.

This framework allows testers to develop test cases using Microsoft Excel. When the test is executed, the framework processes the keywords and calls functions associated with the keywords in order to perform the defined actions on the application under test (AUT).

With this framework in place, applications can be automated without starting to develop a framework from scratch. Testers use the keywords for the creation of test scripts and create extra application-specific keywords if necessary.

2.2. Framework Features

In addition to standard features such as performing operations and verifications on the objects, the framework includes other sophisticated features, such as:

Feature	Description
Variables	Creation and usage of variables (local and global) within the scope of the test script
Conditions	Conditions can be implemented to handle different flows in the script
Data driven	External test data sets can be imported into the test script and used across the flow
Reports	Customized reporter logs
Call functions/actions	Common functions or actions can be called using keywords.
Miscellaneous	Keywords for performing keyboard operations, Date-time functions.
Exceptions	Runtime errors are handled and reported in the framework

2.3. Open2Test Benefits

Benefit	Description
No scripting skills required	The testers will use English like keywords and so there is no real necessity to know the underlying scripting language of the test automation tool for creation of test scripts.
Increased productivity	Since it is keyword based and the processing code is already written, the number of lines and the complexity is reduced which reduces effort to create test scripts.
High fidelity scripts	Keywords usage induces code quality which is easy to read and understand.
Maintenance	Ease of maintenance
Enables High ROI	Increased productivity in the creation of scripts and ready-to-use framework availability along with other benefits enables high and quick ROI.

3. Framework Architecture

3.1. Framework Architecture

The framework consists of the following sub-components, namely:

- Driver Script
- Function Library
- External Test Data & Variables
- Object repository

Apart from the above, the Test Automation Script holds the step by step scenario which is to be tested on the Application Under Test written using the Open2Test Keyword combinations.

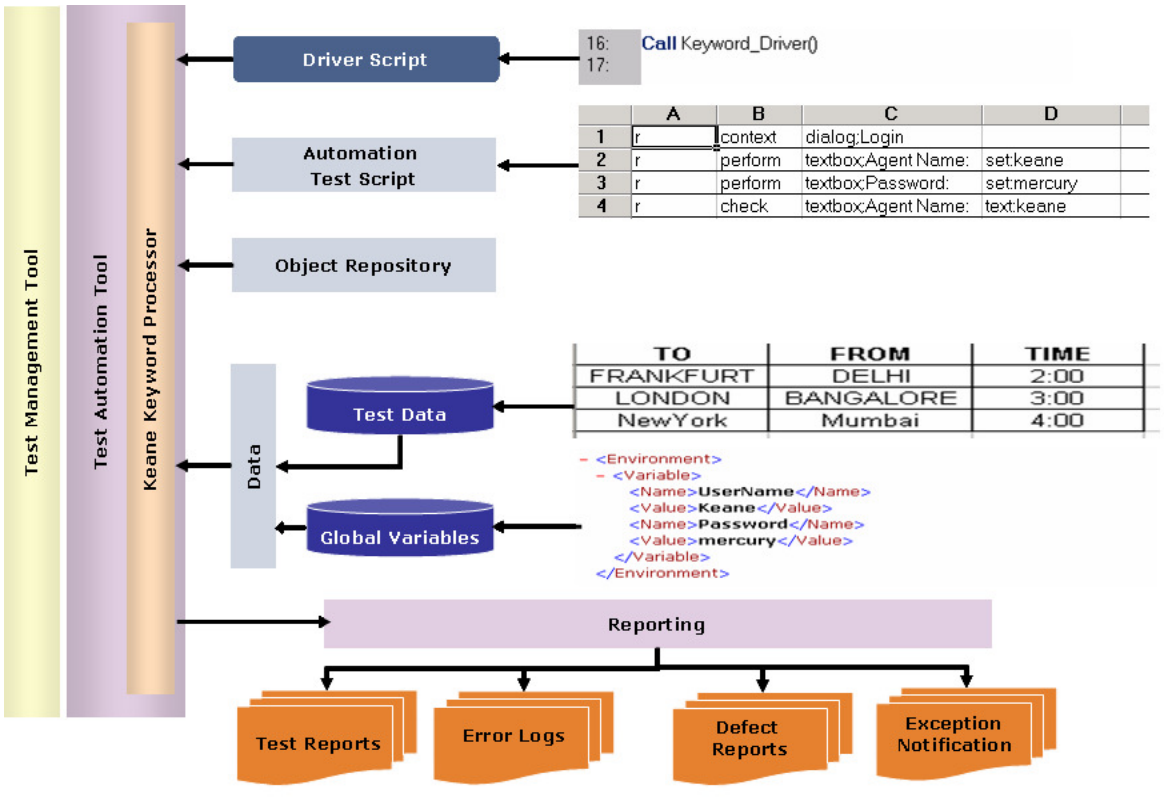


Figure 1: Framework Architecture

3.1.1. Driver Script

The driver script (DS) drives the scenarios. This takes care of parsing the keywords in the test script in conjunction with the objects, if any, to perform the actions on the application under test.

3.1.2. Function Library

The function library holds all the function definitions of the framework.

3.1.3. External Test Data & Variables

External test data is given as inputs to the test scripts for use along with the operations on the application using different set of data. The Test data are referred using the prefix "dt_".

Variables (Global and Local) can be used across the test script to store run time values and used again in further steps based on need.

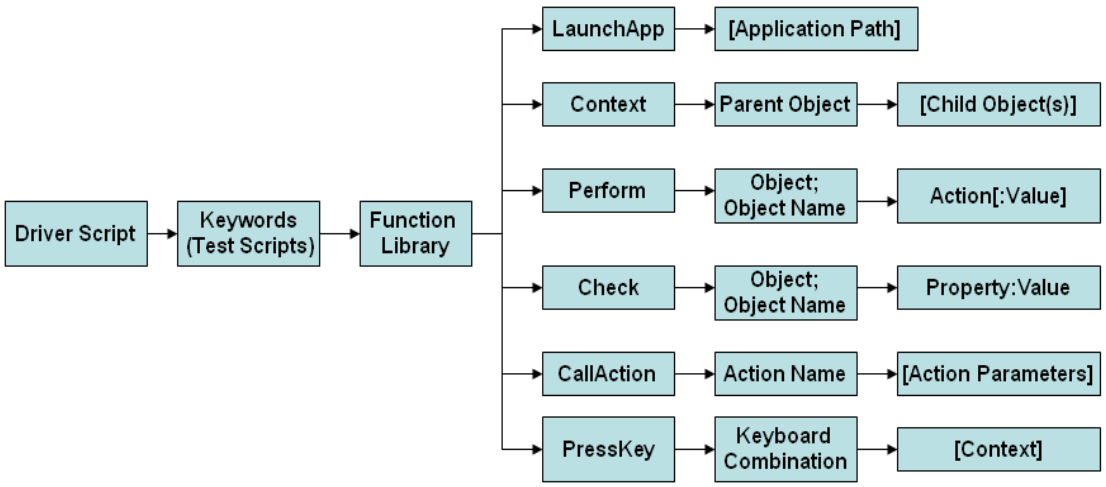
3.1.4. Object Repository

The Test Automation tool (like QTP) learns the interface of an application by using the properties and names of the application's objects and these are stored in the Object Repository. These object related information (usually the names) is used to refer to the objects during the flow of a scenario.

3.1.5. Reporting

After execution of the test script, it is important that a report is generated with the results of the execution. QTP generates a default report and in addition Open2Test can also create a HTML based report file for all operations in the test script.

4. Flow Diagram of the Open2Test Test Automation Framework



Note :

The values within [ ] are optional, usage depends on the scenario

Figure 2: Flow Diagram

5. Structure of the script

5.1. Columns

The Keyword script is the actual test automation script based on the given scenario that corresponds to a manual test case. The actions that are to be performed based on the test scenario written line by line using the keyword and method/action combinations as excel sheets. The Open2Test test automation script involves 6 columns which are:

Column number	Column name	Description
1	Run	Usually takes 'r' or '<empty>' as the value. Used for letting the framework know whether to run that row or skip. If the value is 'r', the keyword driver will process further columns of that row.
2	Keyword	This holds the keywords that correspond to the type of action to be performed. Some of the keywords that could be used here are perform, launchapp, loop, check, wait etc. The keyword driver will use the remaining columns as parameters for the keyword provided.
3	Objectdetails	Used for mentioning the objects of the application upon which the action is to be performed. But this could vary for some of the keywords. When this is used for mentioning the objects, it uses ';' as a delimiter for the object type and object name. For example, Window;Login where Window is the object type and Login is the object's name.
4	Action	This is used to indicate the specific action that is to be performed on the object specified in the 'Objectdetails' column. For example, Click, Set, Submit etc. If there are parameters that need to be passed, it is mentioned with the delimiter ':'.
5	Action2	Could be used to handle return values from a user defined function.
6	Comments	Free form text to attach an meaningful text for that row in the script. This is optional.

The following simple example gives you illustration of how a Open2Test based test automation script would look like:

Sample Test steps

1. Open the application.
2. Set the username
3. Set the password
4. Click on Login

Objects
Window – Login
Textbox – Username
Textbox – Password
Command button – Login

Run	Keyword	Objectdetails	Action	Action2	Comments
r	Launchapp	C:\myapp\app.exe			Launching the application
r	Context	Window;Login			Setting the context on the login window
r	Perform	Textbox;username	Set:myname		Setting the username
r	Perform	Textbox;password	Set:mypasswd		
r	Perform	Button;login	Click		Performing click

As you would have observed in the above example, the keyword uses the objectdetails as the action columns for parameter values (launchapp takes the application location. Context and the perform uses object information). It is also important to note that context keyword should be used once for every new parent object that is used in the script. This tells the underlying script about the current active window/browser/page under which the related objects in the subsequent lines of scripts are available.

For the list of keywords and its usage, refer to the keywords documentation.

5.2. Role of Delimiters

Open2Test uses three kinds of delimiters ; (semi colon), :(colon), ::(double colon) in various columns. These are used to separate the entities provided in the column to be used in the scripts.

; (semi colon)

The column 3 (object details) uses the semi colon to separate the object type and the object name.

For example:

Textbox;username

Here ‘Textbox’ denotes the object type and ‘username’ denotes the object’s name. In essence, it means ‘username of the type Textbox’ as a whole.

: (colon)

The column 4 (Action) uses the colon to separate the method/property with that of the value/parameter.

For example:

Set:myname

Here 'Set' denotes the method/action (for the object mentioned in column 3 and 'myname' is the parameter to that method/action. In essence, it means 'set myname' to the object mentioned in the previous column.

:: (double colon)

The column 4 (Action) uses the double colon to specify the child objects present in a window, browser or dialog box.

For example:

page;<name> :: window;<name> :: Dialog;<name>

To specify the optional parameters to be used for certain keywords, use the double hyphen '**--**'delimeter.

For example:

TableSearch:<colname1>;<rowval1>::<colname2>;<rowval2>--<[no of columns]>

5.3. Variables

To store a value in a variable, the following can be used.

r	assignvalue	strName;Smith			

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

To store the property value of an object :

r	storevalue	Textbox;username	<prop_name>:<var_name>		

To input a value from a variable:

r	Perform	Textbox;username	Set:#varname		

If a datatable value is to be referred, it will be dt_varname instead of #varname.

5.4. Sequence of keywords

While scripting using keywords, some keywords have to be written in combination with other keywords.

The keyword 'Context' has to be used whenever the AUT screen changes. Let us assume that after successful login, the AUT screen changes to Home page. Then the script should include the new context before the objects of the new screen are referred.

Run	Keyword	Objectdetails	Action	Action2	Comments
r	Launchapp	C:\myapp\app.exe			Launching the application
r	Context	Window;Login			Setting the context on the login window
r	Perform	Textbox;username	Set:myname		Setting the username
r	Perform	Textbox;password	Set:mypasswd		
r	Perform	Button;login	Click		Performing click
r	context	Window;Home			Setting context to the current active screen
r	perform	Button;new	Click		Click on a button in the current active screen.