Open2Test Test Automation Framework for OpenScript – Implementation Guide (Web)

Version 1.0

January 2010

DISCLAIMER

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

TABLE OF CONTENTS

1. Purpose of the Document

This document provides an overview of the prerequisites and settings required to implement the Open2Test Test Automation Framework using Oracle OpenScript.

2. Open2Test Test Automation Framework for OpenScript - Features

The Open2Test Test Automation Framework for Oracle OpenScript is a comprehensive Keyword-Driven Framework that can be used to execute most of the user actions that can be performed on the Application under Test (AUT).

Along with such standard features as performing operations and verifications on the objects, a number of other sophisticated features are also included:

- **OOPS Concepts**: As Java is the scripting language for OpenScript, this framework has been designed to incorporate the powerful features of OOPS concepts.
- **Web_Framework Class**: The Open2Test Test Automation Framework has been created as a class, whose instance needs to be created in calling script for framework execution.
- **Usage of Variables**: Variables can be defined, manipulated, and used across the generated test script.
- **Browser Compatibility**: Supports automation of test scripts in both Internet Explorer (version 6.0 and above) and Mozilla Firefox.
- **Conditional Checking:** Conditional constructs like 'if' can be implemented using keywords to handle different flows based on conditions.
- **Data-Driven Testing**: This framework supports data-driven testing by importing data from external data sheets, which can be used in the required fields during execution.
- **User-Defined Functions**: User-defined functions can be included in the framework to automate the application-specific functions.
- **Looping/Iteration**: Required test steps/test scripts can be looped/iterated for the required number of times using the loop keyword.
- **Common Functions**: Operations related to files and folders can be performed using the 'Functions' keyword.
- **Exception Handling**: Run-time errors are effectively handled and reported using this framework.

3. Framework Implementation in OpenScript

The Keyword-Driven Framework is an application-independent framework that performs all possible actions and verifications on an object. Hence, the code for the same object can be used across different applications.

In the keyword-driven approach, the entire script is developed with keywords. The script is developed in a spreadsheet that is interpreted by the main driver script, which then uses the function library to execute the complete script.

3.1. Object Library

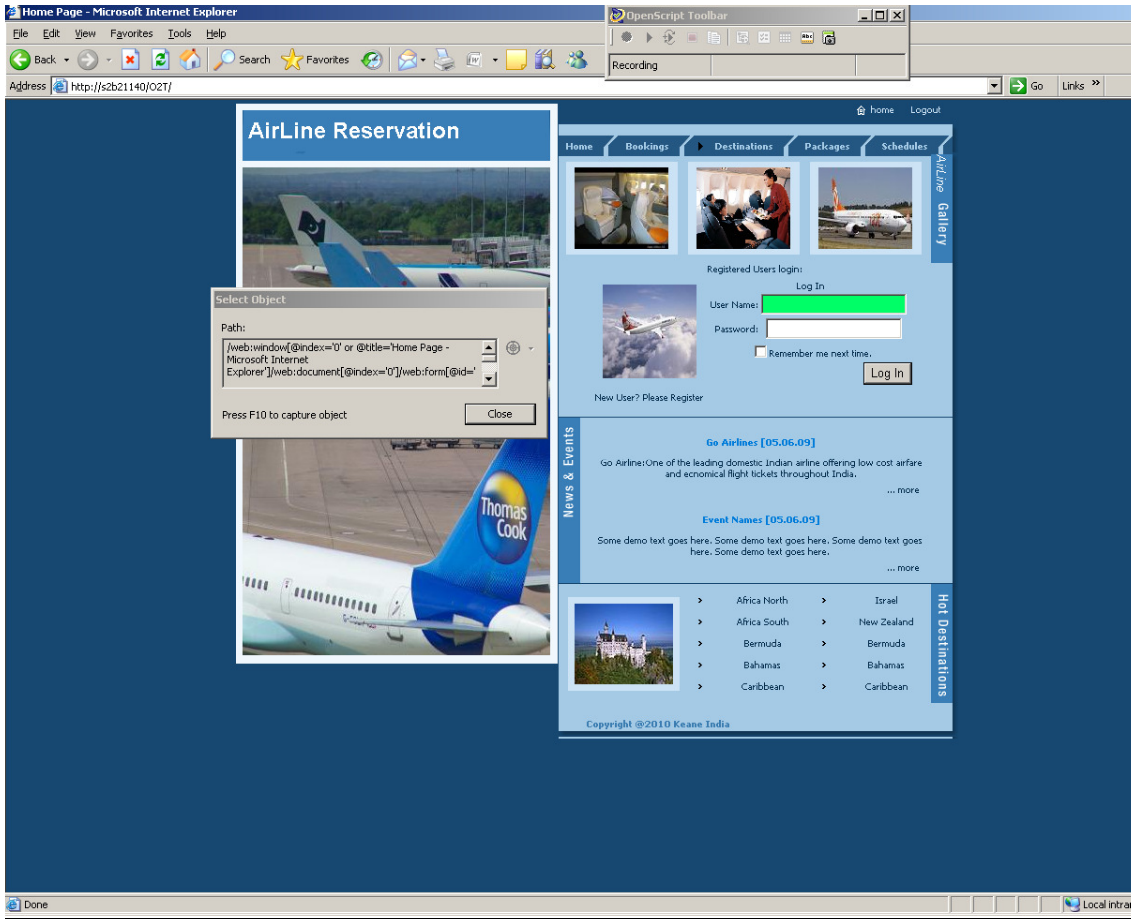
For the Open2Test Test Automation Framework, the Object Library of the AUT acts as an object reference. In OpenScript, the Object Library can be created in two ways:

1. Using Inspect Path.
2. Through OpenScript Recording.

3.1.1. Using Inspect Path

DOM Path of the required Web element can be captured manually in OpenScript and can be saved in the required file of the Object Library. To capture the DOM Path manually:

1. Select 'Script → Inspect Path' from the menu bar.
2. System will display the 'Select Object' window and will launch the browser.
3. Load the required Web application in the browser and click on the required web element.
4. DOM Path of the highlighted element will be displayed in 'Select Object' window.
5. Press 'F10'.
6. System will capture the DOM Path of the selected Web element.
7. Select 'Save to Object Library' menu from the 'Select Object' window.
8. Provide the required alias name and save the element's path in the required Object Library.



Select Object

Path:

```
/web:window[@index='0' or @title='Home Page -
Microsoft Internet
Explorer']/web:document[@index='0']/web:form[@id='
```

Press F10 to capture object

Close

Go Airlines [05.06.09]

Go Airline:One of the leading domestic Indian airline offering low cost airfare and ecnomical flight tickets throughout India.

... more

Event Names [05.06.09]

Some demo text goes here. Some demo text goes here. Some demo text goes here. Some demo text goes here.

... more

Africa North	Israel
Africa South	New Zealand
Bermuda	Bermuda
Bahamas	Bahamas
Caribbean	Caribbean

Copyright @2010 Keane India

<u>Note:</u> While creating the Object library, always choose the option 'Specified Location' in the 'Save In' dropdown of 'Add Object Library' window.

3.1.2. Through OpenScript Recording

Object Library can also be created while recording the test flow in OpenScript. To create an Object Library while recording:

1. Create a new Web functional test script in OpenScript.

2. Right click on 'Initialize' node and select the option 'Add → Other'.

3. From the 'Add' window, select 'General → Load Object Library' option and click on the 'OK' button.

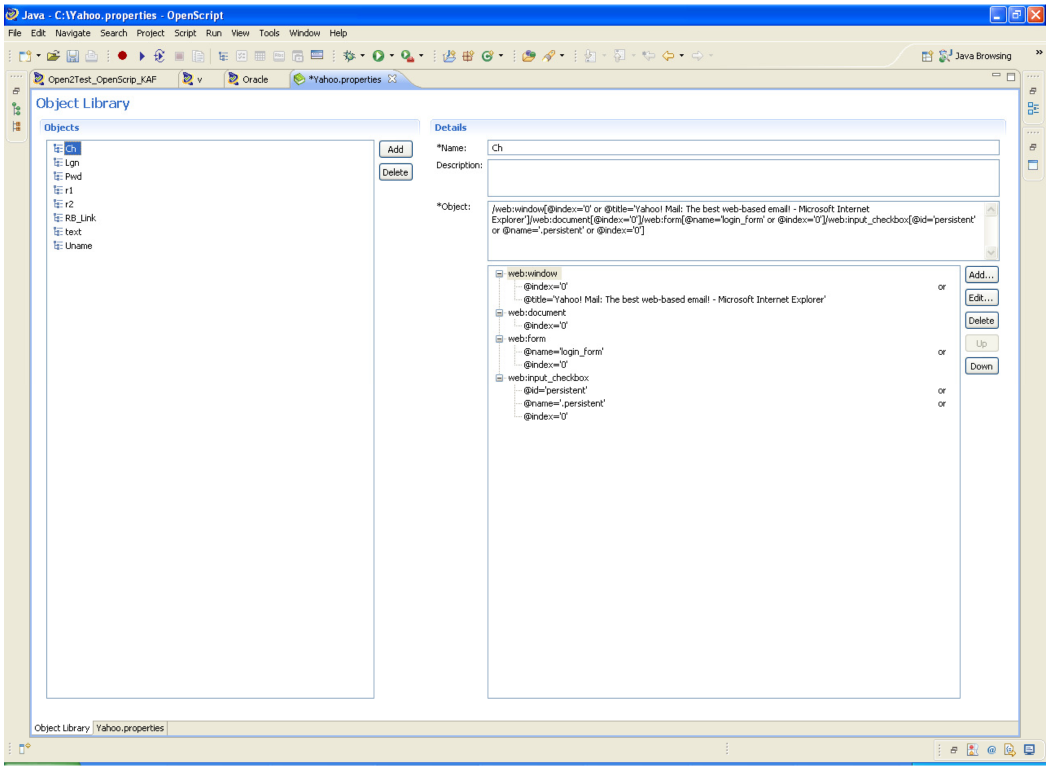
4. In the 'Add Object Library' window, choose the option 'Specified Location' from the 'Save In' dropdown.

5. Browse for the required directory and create a new empty .properties file.

6. Record the required test flow in OpenScript.

7. Save the script.

8. OpenScript would have captured the DOM Path of all the web elements encountered during the test flow recording. Use 'File -> Open Object Library' menu to verify.



*Name: Ch

Description:

*Object: /web:window[@index='0' or @title='Yahoo! Mail: The best web-based email! - Microsoft Internet Explorer']/web:document[@index='0']/web:form[@name='login_form' or @index='0']/web:input_checkbox[@id='persistent' or @name='.persistent' or @index='0']

```
web:window
    @index='0'                                                      or
    @title='Yahoo! Mail: The best web-based email! - Microsoft Internet Explorer'
web:document
    @index='0'
web:form
    @name='login_form'                                              or
    @index='0'
web:input_checkbox
    @id='persistent'                                                or
    @name='.persistent'                                             or
    @index='0'
```

3.2. Keyword Reference Document

The KeyWord Reference Document (KRD) will be used as a standard reference for scripting purposes. The scripting can be done for all possible actions and verifications using the KRD. KRD defines the syntax to be used in the Keyword Test Case for framework implementation. It lists the syntax for all possible actions and verifications that can be performed using Open2Test Test Automation Framework.

3.3. Keyword Test Case

Open2Test Test Automation Framework supports Keyword Test Case in .csv file format. Basically, Keyword Test Case acts as an input source for the framework, which describes the actions to be performed on the AUT as test steps.

Keyword Test Case contains following input source columns:

a) To be Automated: This column decides whether the particular test step of Keyword Test Case needs to be automated or not. If this column contains a label 'r', that particular test step would be automated; otherwise, the corresponding test step would be skipped during execution.

b) Keyword: This column identifies the kind of action that would be performed on the AUT. Action could be Perform, Check, StoreValue, Condition, Loop etc. Please refer to the Keyword Reference Document for a detailed list of actions.

c) Object: This column defines the object on which the particular action needs to be performed. The object column predominantly has two parameters - Object Type and Object Path, separated by the delimiter ';'.

   Object Type denotes the type of the object (browser, button, checkboxes, etc.).

   Except for the objects Browser and Dialog boxes, for every other object, the Object Path should correspond to the alias name of the Object Path defined in Object library.

   For Browser, the Object Path corresponds to the text displayed on the title of the corresponding browser. Wild card characters can also be submitted with title text.

   For Dialog boxes, the Object Path corresponds to the text displayed on the corresponding dialog boxes.

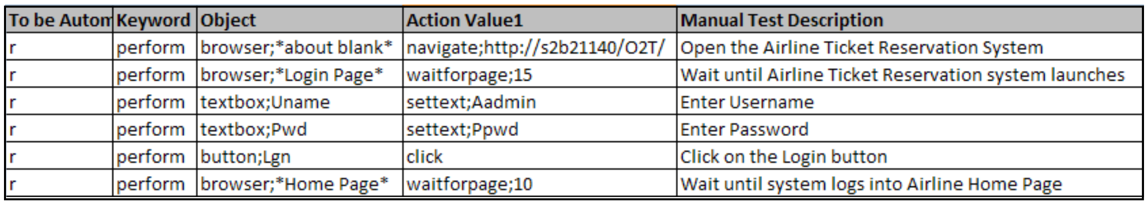
d) Action Value: This column defines the type of action that needs to be performed on AUT. Action value column predominantly has two parameters - Action Type and Action value, separated by the delimiter ';'.

   Action type denotes the type of actions like settext, check, selecttext, multiselecttext, etc.

   Action value denotes the value to be set or selected on the particular object.

   Action value column also denotes the attribute value to be captured on the object and the variable at which the attribute value is to be stored.

e) Manual Step Description: This column describes the action that is performed on the corresponding step. The text provided in this column will be displayed in Results View, which helps in detailed error reporting.



To be Autom	Keyword	Object	Action Value1	Manual Test Description
r	perform	browser;*about blank*	navigate;http://s2b21140/O2T/	Open the Airline Ticket Reservation System
r	perform	browser;*Login Page*	waitforpage;15	Wait until Airline Ticket Reservation system launches
r	perform	textbox;Uname	settext;Aadmin	Enter Username
r	perform	textbox;Pwd	settext;Ppwd	Enter Password
r	perform	button;Lgn	click	Click on the Login button
r	perform	browser;*Home Page*	waitforpage;10	Wait until system logs into Airline Home Page

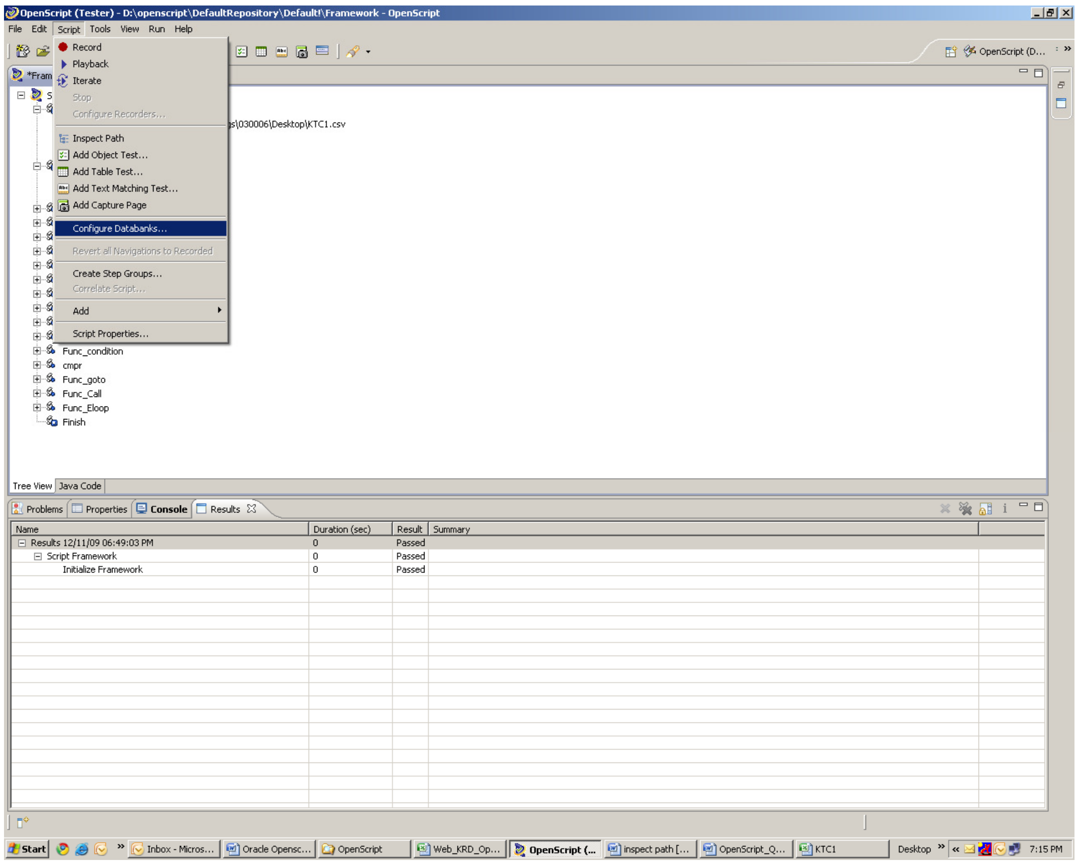
Figure 2: Sample Script

3.4. Databank - Parameterization

In OpenScript, parameterization can be done using the Databank feature. The user needs to configure the .csv file containing test data with the test script to use external test data in OpenScript.

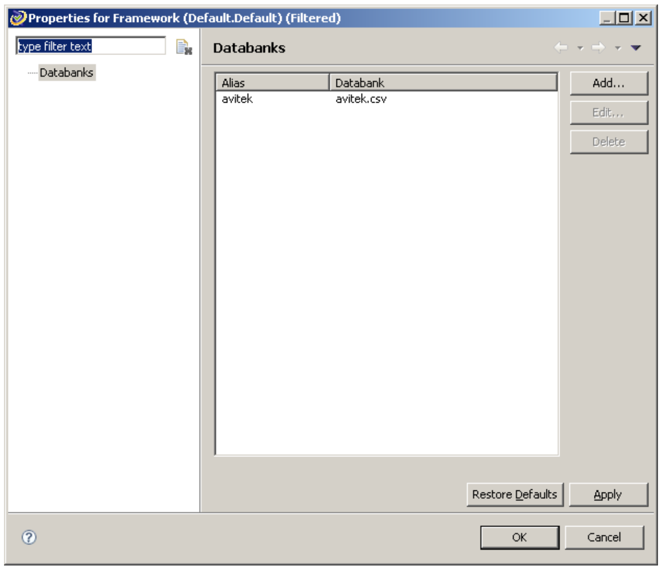
Open2Test Test Automation Framework also supports parameterization with .csv files. To use parameterized test data in the framework, follow the steps below:

1. Select 'Script -> Configure Databanks..' from the menu bar.



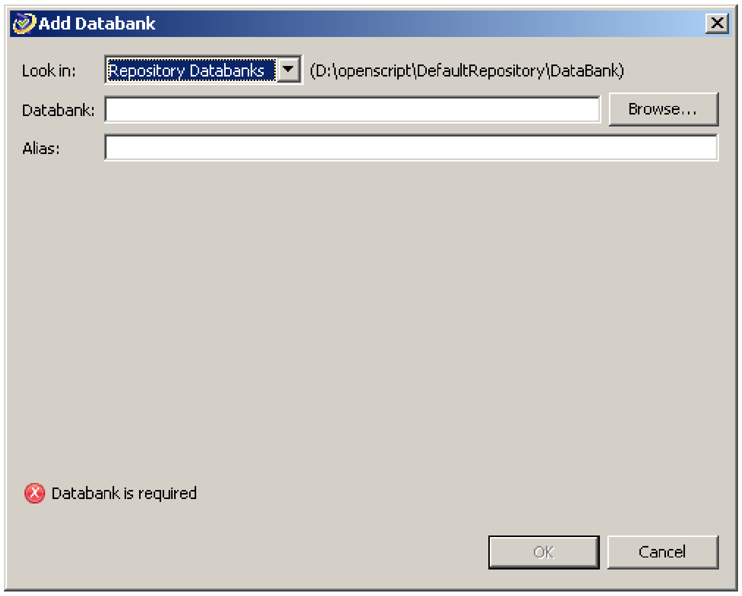
Name	Duration (sec)	Result	Summary
Results 12/11/09 06:49:03 PM	0	Passed	
Script Framework	0	Passed	
Initialize Framework	0	Passed	

2. Click on the Add button in the 'Properties for Framework' dialog box. The system will open the 'Add Databank' dialog box.



Alias	Databank
avitek	avitek.csv

3. Click on the Browse button and point the .csv file containing external test data and provide the required ‘Alias’ name.



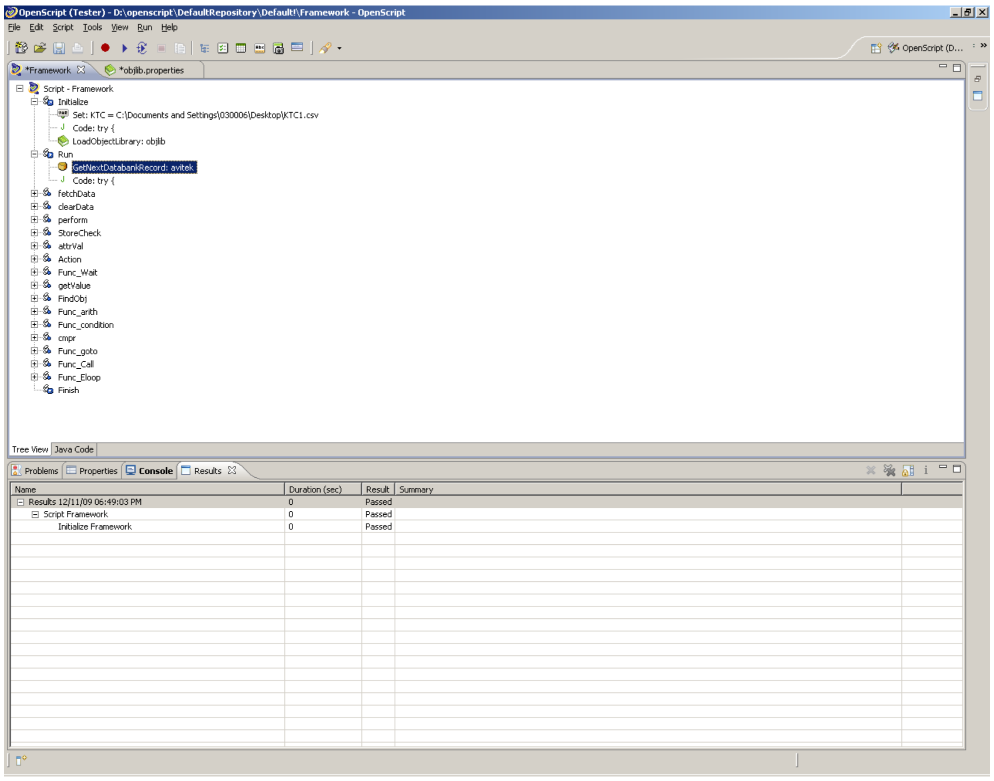
Look in: Repository Databanks (D:\openscript\DefaultRepository\DataBank)

Databank:

Alias:

Databank is required

4. Once added, the loaded .csv file will be configured as Databank for the script.


OpenScript (Tester) - D:\openscript\DefaultRepository\Default!\Framework - OpenScript
File Edit Script Tools View Run Help
*Framework
*objlib.properties
Tree View
Java Code
Problems
Properties
Console
Results

```
Script - Framework
  Initialize
    Set: KTC = C:\Documents and Settings\030006\Desktop\KTC1.csv
    Code: try {
    LoadObjectLibrary: objlib
  Run
    GetNextDatabankRecord: avitek
    Code: try {
  fetchData
  clearData
  perform
  StoreCheck
  attrVal
  Action
  Func_Wait
  getValue
  FindObj
  Func_arith
  Func_condition
  cmpr
  Func_goto
  Func_Call
  Func_Eloop
  Finish
```

Name	Duration (sec)	Result	Summary
Results 12/11/09 06:49:03 PM	0	Passed	
Script Framework	0	Passed	
Initialize Framework	0	Passed	

<u>Note:</u> Alias name provided for the databank file should be passed as one of the parameters while creating the instance of 'Framework' class. Refer Call to Framework section for more details.

<u>Note:</u> In Keyword Test Script, syntax 'dt_' should precede the name of the required Databank column name to use the value of the corresponding column in test execution.

<u>Note:</u> Keyword 'GetNextDB' can be used in Keyword Test Script to use the next row value of the Databank column.

3.5. Call to Framework

As OpenScript uses Java as the scripting language, Open2Test Test Automation Framework has been created as a separate class whose instance needs to be created in the calling script for framework execution.

To create the instance of the Web_Framework class in the calling script, the first package 'com.Open2Test.Oscript' (containing the classes Web_Framework, UDF and CommonFn) needs to be exported as specified in the document 'Open2Test Test Automation Framework for OpenScrit – Quick Start Guide'.

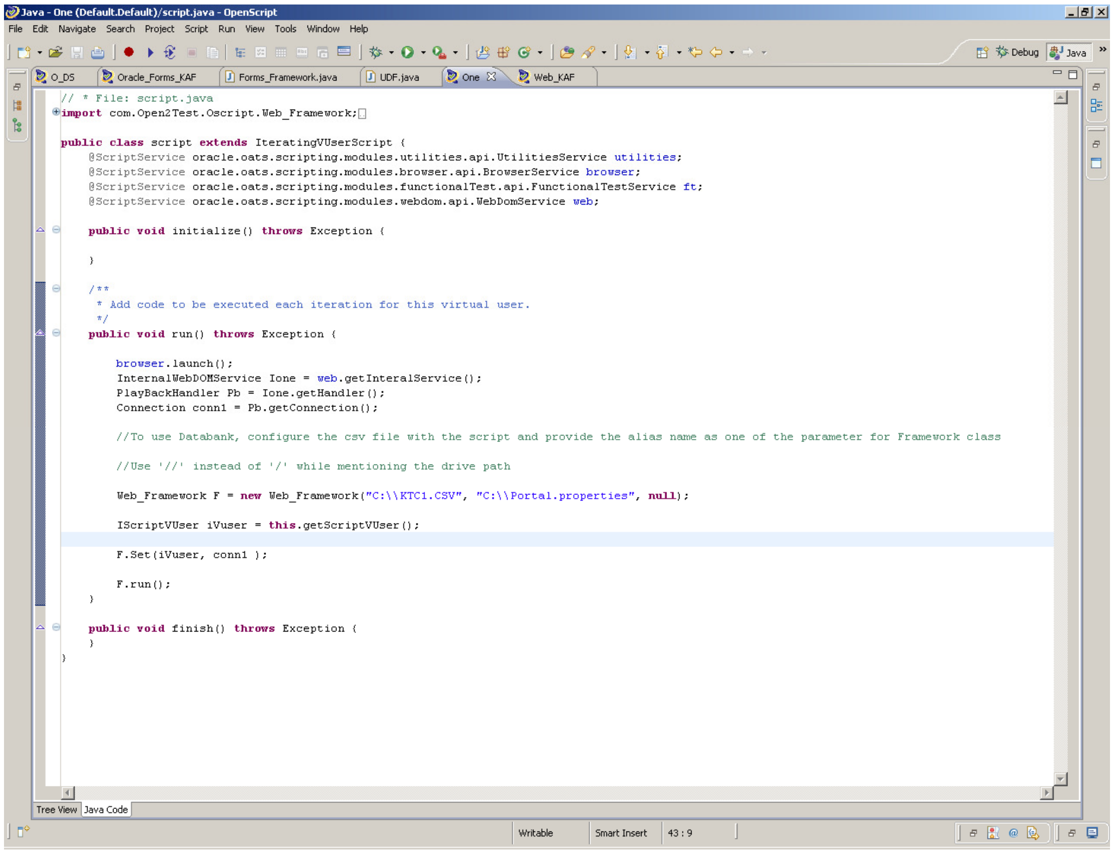
Then a new Web functional test script needs to be created in OpenScript and a source code, available in the notepad file 'Driver Class' (downloaded from Open2Test.org), needs to be pasted in the created script.

Web_Framework class requires three parameters as input for creating an instance:

1. Drive path of Keyword Test Case
2. Drive path of Object Library
3. Alias name of Databank (configured with the script).

Of the above mentioned parameters 'Alias name of the Databank' is optional.If a test script doesn't use databank in its execution, value 'null' can be passed.

<u>Note:</u> Exported Web_Framework Class needs to be referenced in the calling script using the 'Manifest.MF' file. Please refer to the Quick Start Guide for more details.

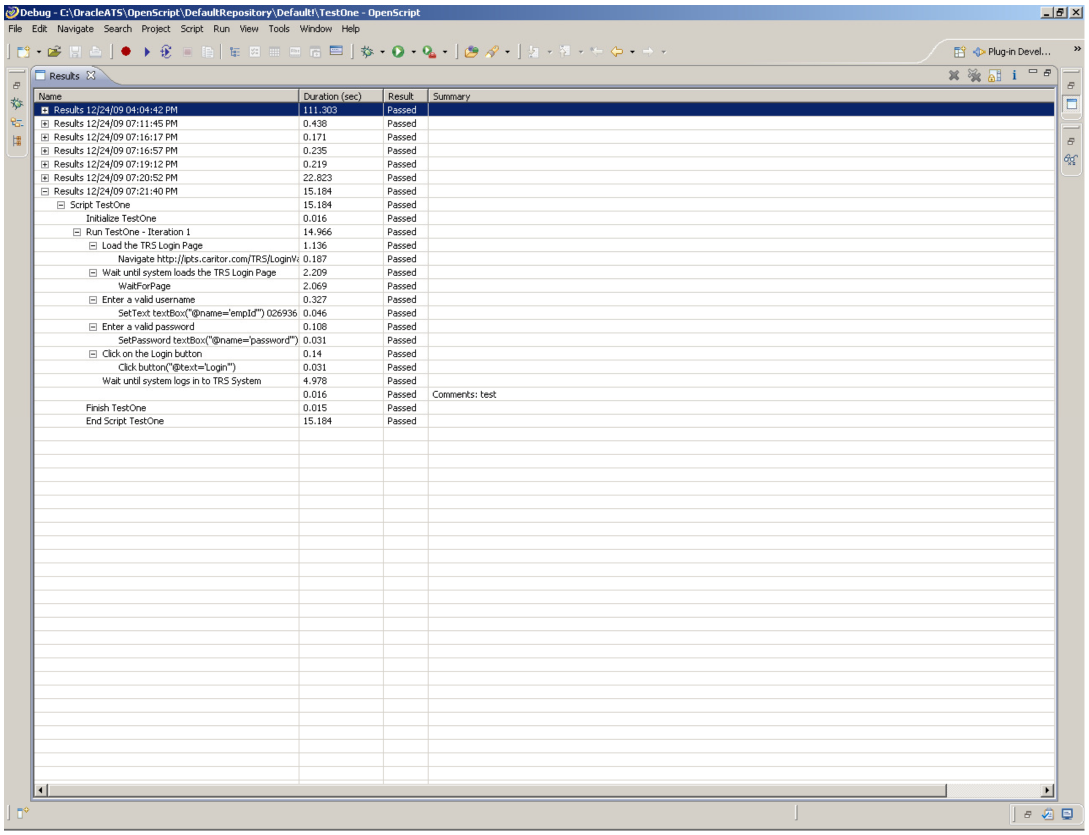

Java - One (Default.Default)/script.java - OpenScript
File Edit Navigate Search Project Script Run View Tools Window Help
Debug Java
O_DS Oracle_Forms_KAF Forms_Framework.java UDF.java One Web_KAF
Tree View Java Code
Writable Smart Insert 43 : 9

```java
// * File: script.java
import com.Open2Test.Oscript.Web_Framework;

public class script extends IteratingVUserScript {
    @ScriptService oracle.oats.scripting.modules.utilities.api.UtilitiesService utilities;
    @ScriptService oracle.oats.scripting.modules.browser.api.BrowserService browser;
    @ScriptService oracle.oats.scripting.modules.functionalTest.api.FunctionalTestService ft;
    @ScriptService oracle.oats.scripting.modules.webdom.api.WebDomService web;

    public void initialize() throws Exception {

    }

    /**
     * Add code to be executed each iteration for this virtual user.
     */
    public void run() throws Exception {

        browser.launch();
        InternalWebDOMService Ione = web.getInteralService();
        PlayBackHandler Pb = Ione.getHandler();
        Connection conn1 = Pb.getConnection();

        //To use Databank, configure the csv file with the script and provide the alias name as one of the parameter for Framework class

        //Use '//' instead of '/' while mentioning the drive path

        Web_Framework F = new Web_Framework("C:\\KTC1.CSV", "C:\\Portal.properties", null);

        IScriptVUser iVuser = this.getScriptVUser();

        F.Set(iVuser, conn1 );

        F.run();
    }

    public void finish() throws Exception {
    }
}
```

Figure 1: Call to Framework

3.6. Test Results for a Keyword-Driven Script

Test execution results can be viewed and analyzed as soon as the run session ends. To verify the test results, open the results pane and expand the 'Run' node of the test result. Detailed Test Results containing Manual Step Description will be displayed.


Debug - C:\OracleATS\OpenScript\DefaultRepository\Default!\TestOne - OpenScript
File Edit Navigate Search Project Script Run View Tools Window Help
Plug-in Devel...
Results

Name	Duration (sec)	Result	Summary
Results 12/24/09 04:04:42 PM	111.303	Passed	
Results 12/24/09 07:11:45 PM	0.438	Passed	
Results 12/24/09 07:16:17 PM	0.171	Passed	
Results 12/24/09 07:16:57 PM	0.235	Passed	
Results 12/24/09 07:19:12 PM	0.219	Passed	
Results 12/24/09 07:20:52 PM	22.823	Passed	
Results 12/24/09 07:21:40 PM	15.184	Passed	
Script TestOne	15.184	Passed	
Initialize TestOne	0.016	Passed	
Run TestOne - Iteration 1	14.966	Passed	
Load the TRS Login Page	1.136	Passed	
Navigate http://ipts.caritor.com/TRS/LoginV	0.187	Passed	
Wait until system loads the TRS Login Page	2.209	Passed	
WaitForPage	2.069	Passed	
Enter a valid username	0.327	Passed	
SetText textBox("@name='empId'") 026936	0.046	Passed	
Enter a valid password	0.108	Passed	
SetPassword textBox("@name='password'")	0.031	Passed	
Click on the Login button	0.14	Passed	
Click button("@text='Login'")	0.031	Passed	
Wait until system logs in to TRS System	4.978	Passed	
	0.016	Passed	Comments: test
Finish TestOne	0.015	Passed	
End Script TestOne	15.184	Passed	

Figure 6: Results in OpenScript

4. References

Oracle OpenScript Help documentation

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.