Selenium Open Source Test Automation Framework Implementation Guide

Version 1.0

September 2009

DISCLAIMER

TABLE OF CONTENTS

1. Purpose of the Document

This document provides an overview of the prerequisites and settings required to implement the keyword-driven framework in Selenium RC

2. Framework Implementation in Selenium

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.

2.1. Test Settings for Keyword-Driven Scripting

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.

a. The test suite, test script, object repository and report folder location should be mentioned in Selenium_Utility excel.

File\Folder Name	Location
Test Suite	C:/Innovez/Selenium_Automation/Test_Suite.xls
Test Script	C:/Innovez/Selenium_Automation/Test_Scripts/
Object Repository	C:/Innovez/Selenium_Automation/Test_Data/Object_Repository.xls
Summary Report	C:/Innovez/Selenium_Automation/Test_Reports/
Screen Shot Report	C:/Innovez/Selenium_Automation/Test_Reports/ScreenShot_Report/
Detailed Report	C:/Innovez/Selenium_Automation/Test_Reports/Detailed_Report/

Note: The selenium utility excel file should be placed in

C:\Documents and Settings\Mantis\Demo-Selenium\Selenium_Utility.xls

b. Collect the properties of objects and define it in Object_Repository.xls like below

ObjectName	ObjectIdentification	ObjectType
Username	username	TextBox
Password	password	TextBox
Login	//input[@value='Login']	Button
Manage	link=Manage	link
Manage Projects	link=Manage Projects	link

c. **Define the test scripts in the test suite Excel like below**

Run	Test Driver
r	Mantis_Create Project
r	Mantis_Report Issue
r	Mantis_View Issue and Delete Project
r	Mantis_Fail
r	Call Tariff Addition and Amendment

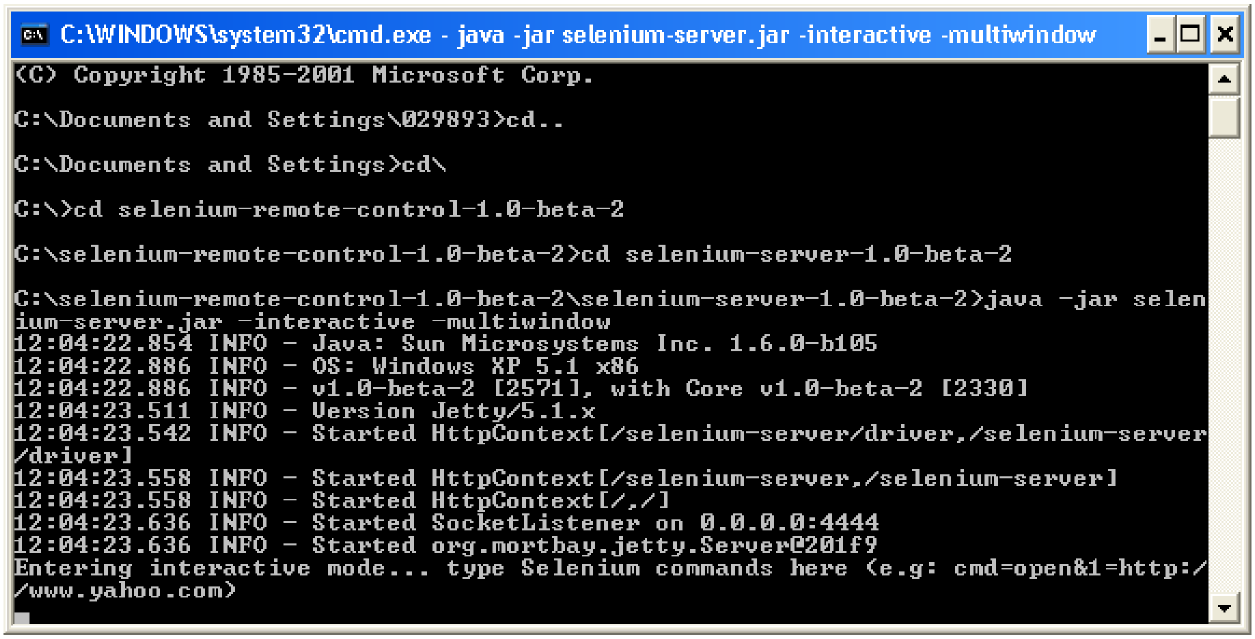
d. **Write the keywords for the test script like below**

Step	Operation	Object	Action
r	callaction	C:\Documents and Settings\Mantis\Demo-Selenium\Innovez\Selenium_Automation\Test_Scripts\Login Action.xls	
r	perform	link;Manage Projects	click
r	wait	3	
r	perform	Button;Create New Project	click
r	perform	Textbox;Project Name	set:Selenium

3. Selenium RC – Server up

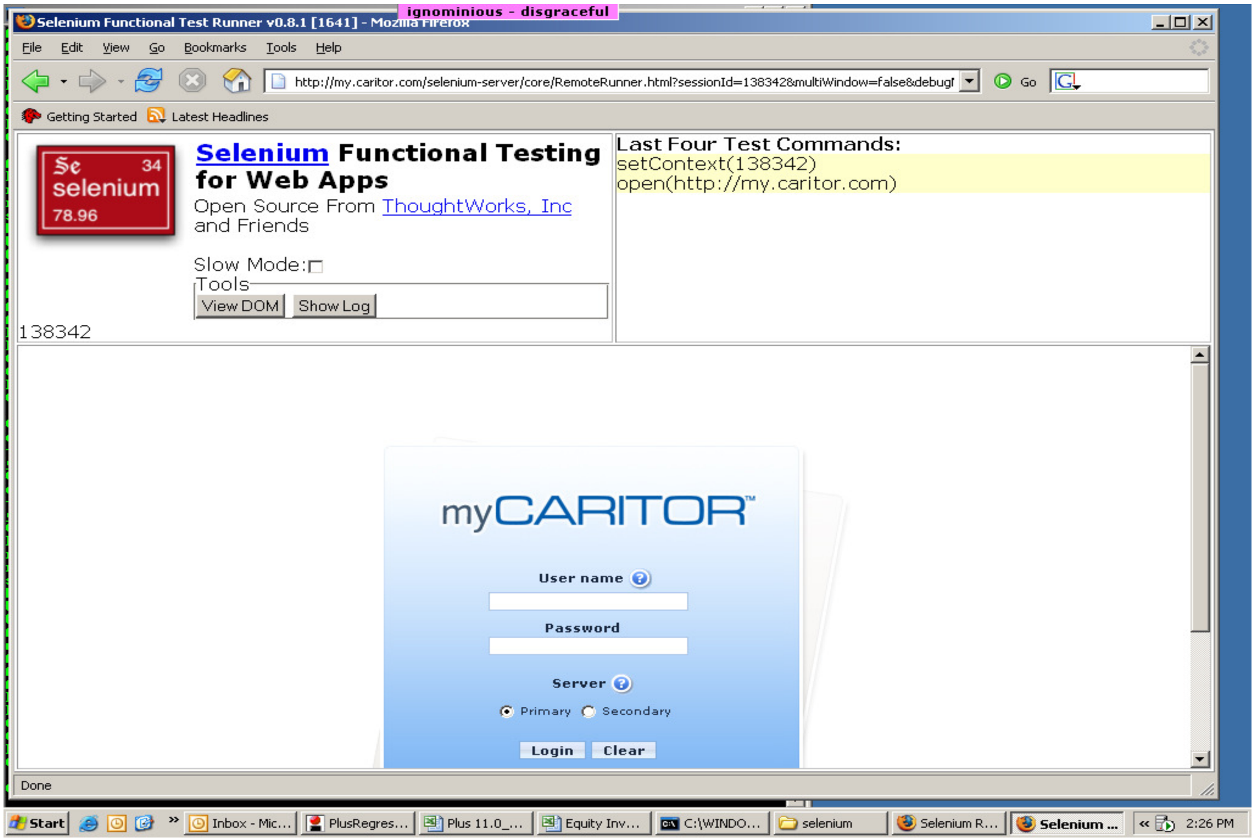
- Starting the server:

  java –jar selenium-server.jar –interactive


C:\WINDOWS\system32\cmd.exe - java -jar selenium-server.jar -interactive -multiwindow

```
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\029893>cd..

C:\Documents and Settings>cd\

C:\>cd selenium-remote-control-1.0-beta-2

C:\selenium-remote-control-1.0-beta-2>cd selenium-server-1.0-beta-2

C:\selenium-remote-control-1.0-beta-2\selenium-server-1.0-beta-2>java -jar selen
ium-server.jar -interactive -multiwindow
12:04:22.854 INFO - Java: Sun Microsystems Inc. 1.6.0-b105
12:04:22.886 INFO - OS: Windows XP 5.1 x86
12:04:22.886 INFO - v1.0-beta-2 [2571], with Core v1.0-beta-2 [2330]
12:04:23.511 INFO - Version Jetty/5.1.x
12:04:23.542 INFO - Started HttpContext[/selenium-server/driver,/selenium-server
/driver]
12:04:23.558 INFO - Started HttpContext[/selenium-server,/selenium-server]
12:04:23.558 INFO - Started HttpContext[/,/]
12:04:23.636 INFO - Started SocketListener on 0.0.0.0:4444
12:04:23.636 INFO - Started org.mortbay.jetty.Server@201f9
Entering interactive mode... type Selenium commands here (e.g: cmd=open&1=http:/
/www.yahoo.com)
```

Figure 1: Selenium RC server up and running


ignominious - disgraceful
Selenium Functional Test Runner v0.8.1 [1641] - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://my.caritor.com/selenium-server/core/RemoteRunner.html?sessionId=138342&multiWindow=false&debugf
Go
Getting Started Latest Headlines
Se 34 selenium 78.96
Selenium Functional Testing for Web Apps
Open Source From ThoughtWorks, Inc and Friends
Slow Mode:□
Tools
View DOM
Show Log
myCARITOR™
User name
Password
Server
● Primary ○ Secondary
Login
Clear
Done
Start
Inbox - Mic...
PlusRegres...
Plus 11.0_...
Equity Inv...
C:\WINDO...
selenium
Selenium R...
Selenium ...
2:26 PM

```
138342
```

Last Four Test Commands:

```
setContext(138342)
open(http://my.caritor.com)
```

Figure 2: Selenium RC running – Embedded Browser

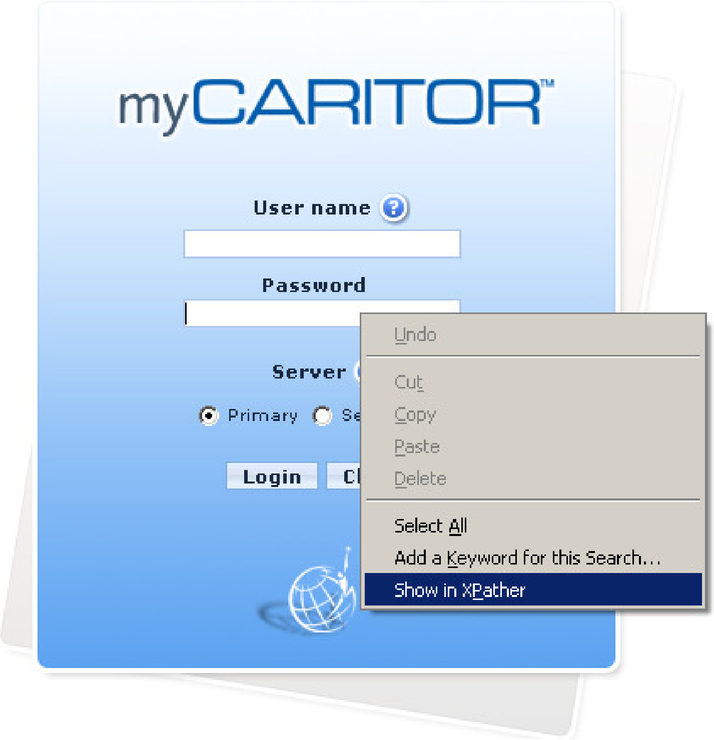
4. Managing Object Repository

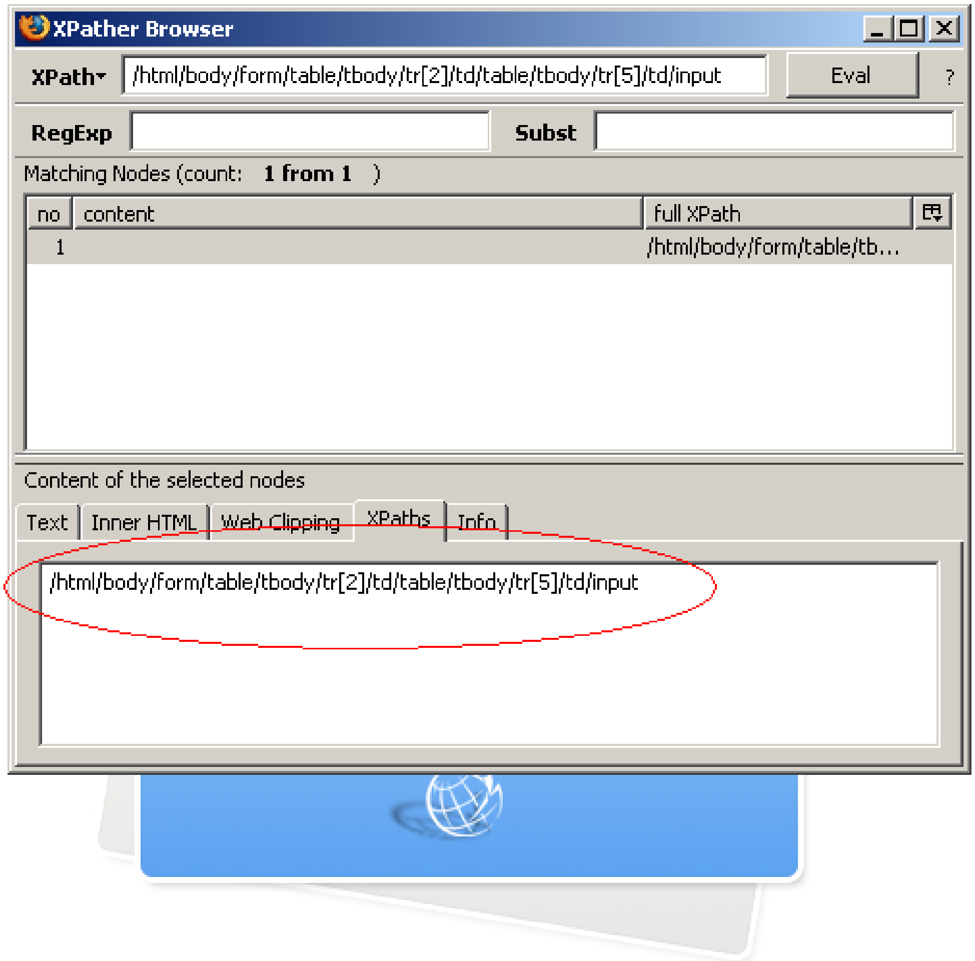
Selenium must learn the interface of an application to be able to work with it. It does this by learning the application's objects and their corresponding property values and storing these object descriptions in an object repository file. There are two types of object repositories: the shared object repository, and the per-action object repository.

The same object repository file can be used for multiple tests if the tests include the same objects. Object information that applies to many tests is kept in one central location and the read-only copy of the repository is associated with the tests.

Here the object repository is maintained in the form of Excel sheets. XPath is also installed as a plug-in to Firefox. To identify the object properties, open Firefox browser, right click on the object and select Show in XPather. The XPather Browser window opens and XPath value is used as the object with logical names.

For Internet Explorer, make use of its add-in 'Developer Too' to get the unique id or text for the object.





XPath: /html/body/form/table/tbody/tr[2]/td/table/tbody/tr[5]/td/input

Matching Nodes (count: **1 from 1**)

no	content	full XPath
1		/html/body/form/table/tb...

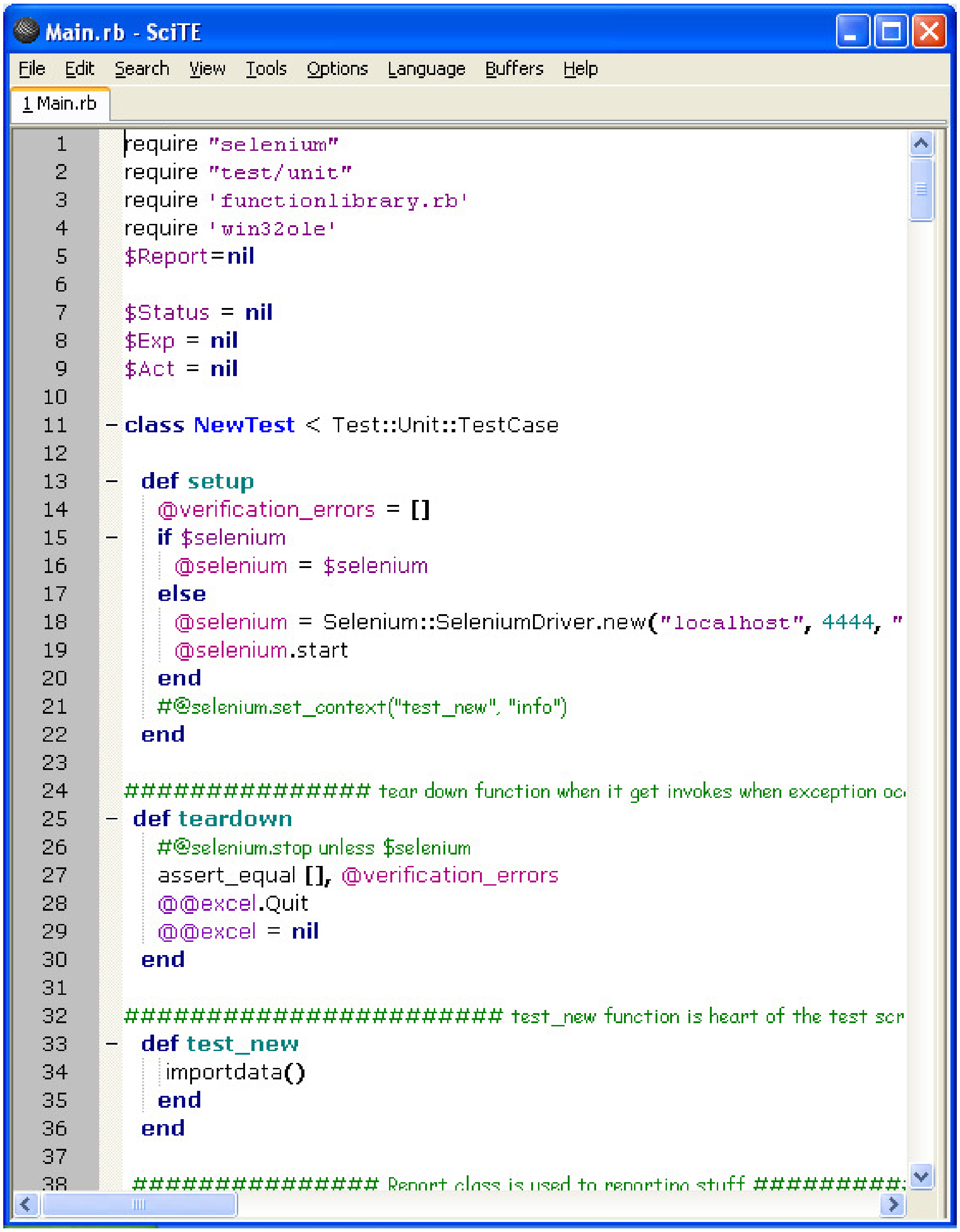
/html/body/form/table/tbody/tr[2]/td/table/tbody/tr[5]/td/input

Figure 3: XPather – Object Identification

5. Call to Framework

The Main.rb should be opened in the SciTE. Press F5 and this will call the framework file associated with the test and perform the actions by interpreting the keywords specified in the data table.

Instead of pressing F5, we are able to run the main.rb by command ruby <main.rb>

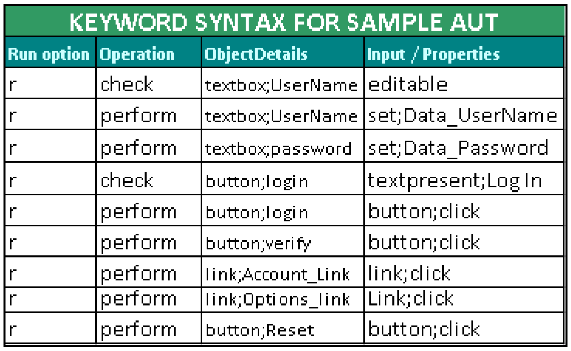


```
require "selenium"
require "test/unit"
require 'functionlibrary.rb'
require 'win32ole'
$Report=nil

$Status = nil
$Exp = nil
$Act = nil

class NewTest < Test::Unit::TestCase

  def setup
    @verification_errors = []
    if $selenium
      @selenium = $selenium
    else
      @selenium = Selenium::SeleniumDriver.new("localhost", 4444, "
      @selenium.start
    end
    #@selenium.set_context("test_new", "info")
  end

################ tear down function when it get invokes when exception oc
 def teardown
    #@selenium.stop unless $selenium
    assert_equal [], @verification_errors
    @@excel.Quit
    @@excel = nil
  end

######################## test_new function is heart of the test scr
  def test_new
     importdata()
    end
  end

 ################ Report class is used to reporting stuff ##########
```

Figure 4: Call to Framework

6. Usage of Keywords

The keywords should be entered in Sheet1 of Microsoft Excel placed in the Test Scripts folder. The syntax for the keywords can be found in the Selenium Keyword Reference Dictionary Document. Below is an example of a simple keyword-driven scripting.



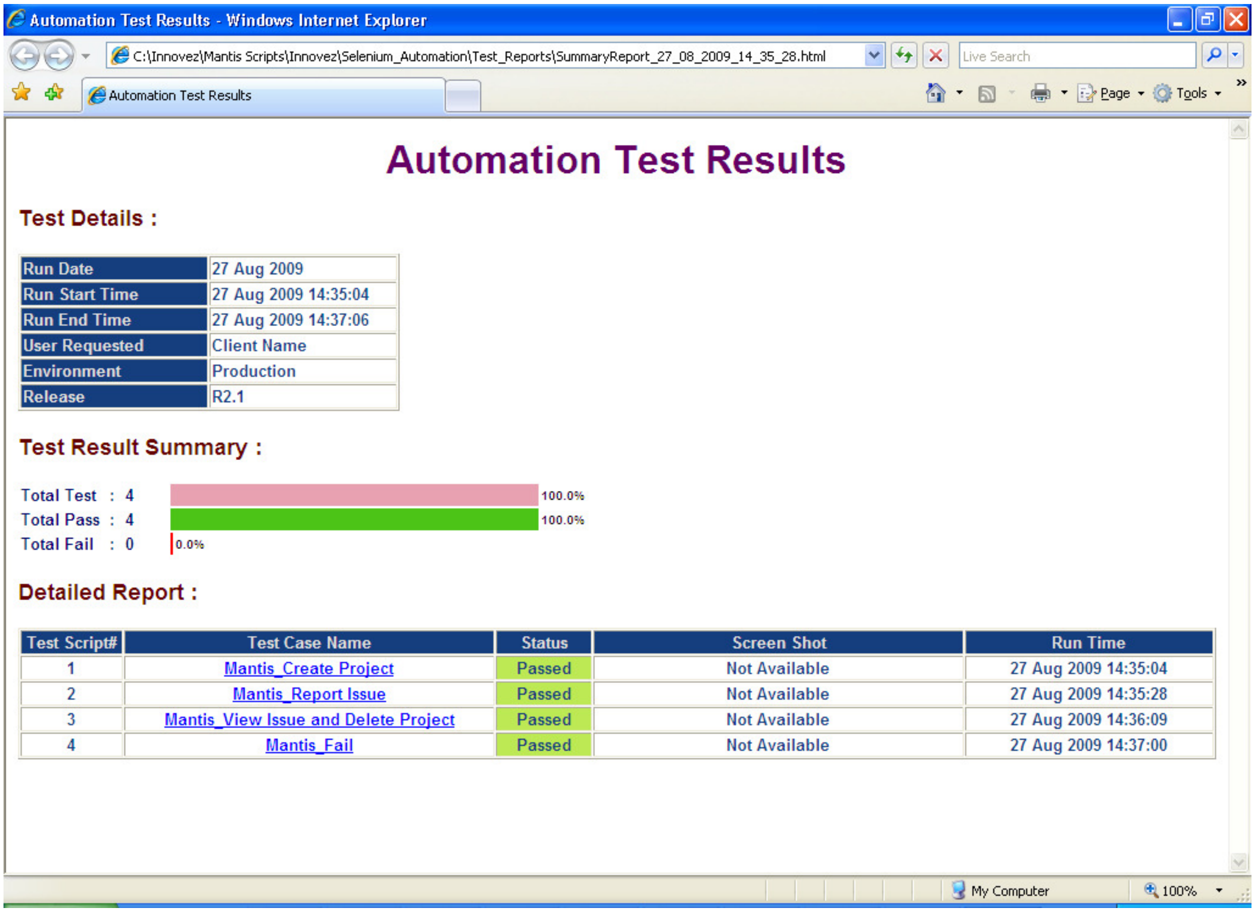
KEYWORD SYNTAX FOR SAMPLE AUT

Run option	Operation	ObjectDetails	Input / Properties
r	check	textbox;UserName	editable
r	perform	textbox;UserName	set;Data_UserName
r	perform	textbox;password	set;Data_Password
r	check	button;login	textpresent;Log In
r	perform	button;login	button;click
r	perform	button;verify	button;click
r	perform	link;Account_Link	link;click
r	perform	link;Options_link	Link;click
r	perform	button;Reset	button;click

Figure 5: Using the Keyword

7. Test Results for a Keyword-Driven Script

Test execution results can be viewed and analyzed as soon as the run session ends. To access the test results, go to the Test_Reports folder customized using the Test automation framework. Two folders will be available: one showing the summary report for Test suite execution, and another folder Detailed_Report displaying the detailed step-wise test results for the each test script. A screenshot will be available for the failure scripts under ScreenShot_Report.


Automation Test Results - Windows Internet Explorer
C:\Innovez\Mantis Scripts\Innovez\Selenium_Automation\Test_Reports\SummaryReport_27_08_2009_14_35_28.html
Live Search
Automation Test Results
Page
Tools
My Computer
100%

Automation Test Results

Test Details :

Run Date	27 Aug 2009
Run Start Time	27 Aug 2009 14:35:04
Run End Time	27 Aug 2009 14:37:06
User Requested	Client Name
Environment	Production
Release	R2.1

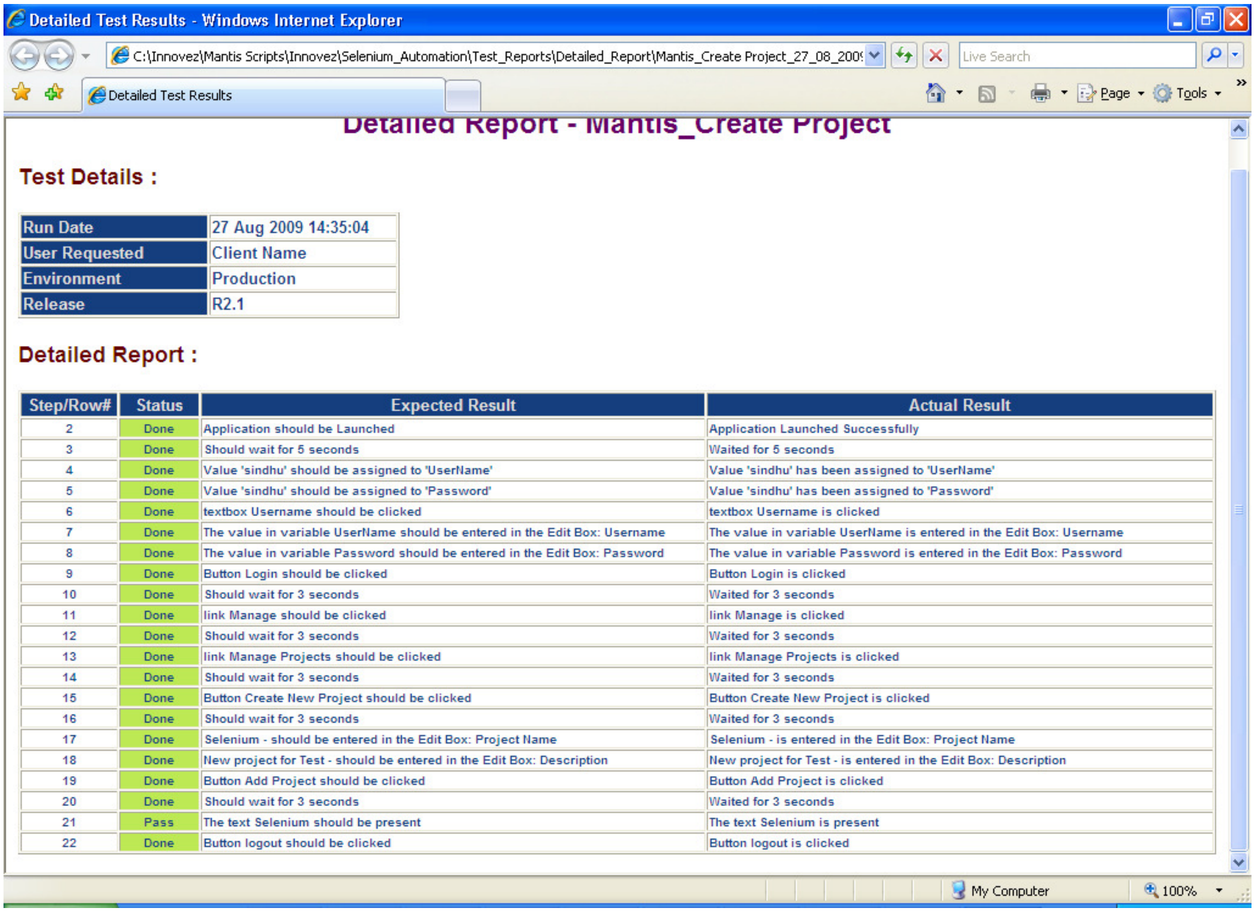
Test Result Summary :

Total Test : 4 100.0%
Total Pass : 4 100.0%
Total Fail : 0 0.0%

Detailed Report :

Test Script#	Test Case Name	Status	Screen Shot	Run Time
1	Mantis_Create Project	Passed	Not Available	27 Aug 2009 14:35:04
2	Mantis_Report Issue	Passed	Not Available	27 Aug 2009 14:35:28
3	Mantis_View Issue and Delete Project	Passed	Not Available	27 Aug 2009 14:36:09
4	Mantis_Fail	Passed	Not Available	27 Aug 2009 14:37:00

Figure 6: Test Results summary for a Test suite


Detailed Test Results - Windows Internet Explorer
C:\Innovez\Mantis Scripts\Innovez\Selenium_Automation\Test_Reports\Detailed_Report\Mantis_Create Project_27_08_200
Live Search
Detailed Test Results
Page
Tools
My Computer
100%

Detailed Report - Mantis_Create Project

Test Details :

Run Date	27 Aug 2009 14:35:04
User Requested	Client Name
Environment	Production
Release	R2.1

Detailed Report :

Step/Row#	Status	Expected Result	Actual Result
2	Done	Application should be Launched	Application Launched Successfully
3	Done	Should wait for 5 seconds	Waited for 5 seconds
4	Done	Value 'sindhu' should be assigned to 'UserName'	Value 'sindhu' has been assigned to 'UserName'
5	Done	Value 'sindhu' should be assigned to 'Password'	Value 'sindhu' has been assigned to 'Password'
6	Done	textbox Username should be clicked	textbox Username is clicked
7	Done	The value in variable UserName should be entered in the Edit Box: Username	The value in variable UserName is entered in the Edit Box: Username
8	Done	The value in variable Password should be entered in the Edit Box: Password	The value in variable Password is entered in the Edit Box: Password
9	Done	Button Login should be clicked	Button Login is clicked
10	Done	Should wait for 3 seconds	Waited for 3 seconds
11	Done	link Manage should be clicked	link Manage is clicked
12	Done	Should wait for 3 seconds	Waited for 3 seconds
13	Done	link Manage Projects should be clicked	link Manage Projects is clicked
14	Done	Should wait for 3 seconds	Waited for 3 seconds
15	Done	Button Create New Project should be clicked	Button Create New Project is clicked
16	Done	Should wait for 3 seconds	Waited for 3 seconds
17	Done	Selenium - should be entered in the Edit Box: Project Name	Selenium - is entered in the Edit Box: Project Name
18	Done	New project for Test - should be entered in the Edit Box: Description	New project for Test - is entered in the Edit Box: Description
19	Done	Button Add Project should be clicked	Button Add Project is clicked
20	Done	Should wait for 3 seconds	Waited for 3 seconds
21	Pass	The text Selenium should be present	The text Selenium is present
22	Done	Button logout should be clicked	Button logout is clicked

Figure 7: Detailed Test Results for a Test script

8. Resource Files

The resource files such as Framework Files, Shared Object Repositories, and Environment Variables that are associated with the test can be stored in the user's machine and can be obtained during test execution.

Function libraries of the file type .rb, environment variables of file type .excel, and object repositories of file type .excel can be placed in the folder specified in the Utility Excel.

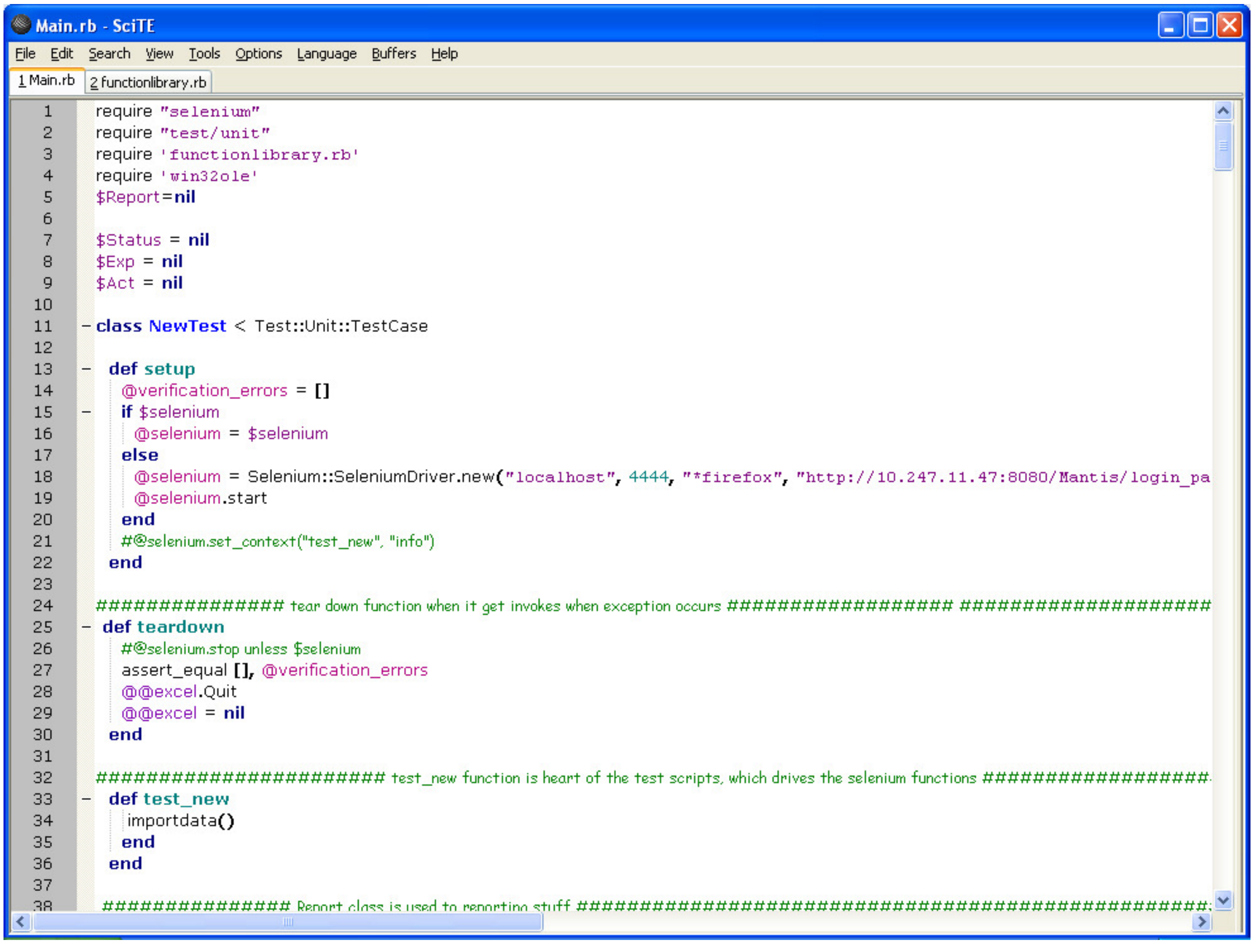
File\Folder Name	Location
Test Suite	C:/Innovez/Selenium_Automation/Test_Suite.xls
Test Script	C:/Innovez/Selenium_Automation/Test_Scripts/
Object Repository	C:/Innovez/Selenium_Automation/Test_Data/Object_Repository.xls
Summary Report	C:/Innovez/Selenium_Automation/Test_Reports/
Screen Shot Report	C:/Innovez/Selenium_Automation/Test_Reports/ScreenShot_Report/
Detailed Report	C:/Innovez/Selenium_Automation/Test_Reports/Detailed_Report/

Note: The selenium utility Excel file should be placed in

C:\Documents and Settings\Mantis\Demo-Selenium\Selenium_Utility.xls

Library files

1. Main.rb
2. Functionlibrary.rb



```
require "selenium"
require "test/unit"
require 'functionlibrary.rb'
require 'win32ole'
$Report=nil

$Status = nil
$Exp = nil
$Act = nil

class NewTest < Test::Unit::TestCase

  def setup
    @verification_errors = []
    if $selenium
      @selenium = $selenium
    else
      @selenium = Selenium::SeleniumDriver.new("localhost", 4444, "*firefox", "http://10.247.11.47:8080/Mantis/login_pa
      @selenium.start
    end
    #@selenium.set_context("test_new", "info")
  end

################ tear down function when it get invokes when exception occurs ################### #####################
 def teardown
    #@selenium.stop unless $selenium
    assert_equal [], @verification_errors
    @@excel.Quit
    @@excel = nil
  end

######################## test_new function is heart of the test scripts, which drives the selenium functions ###################
  def test_new
     importdata()
    end
  end

 ################ Report class is used to reporting stuff ######################################################
```

Figure 8: Resource Files