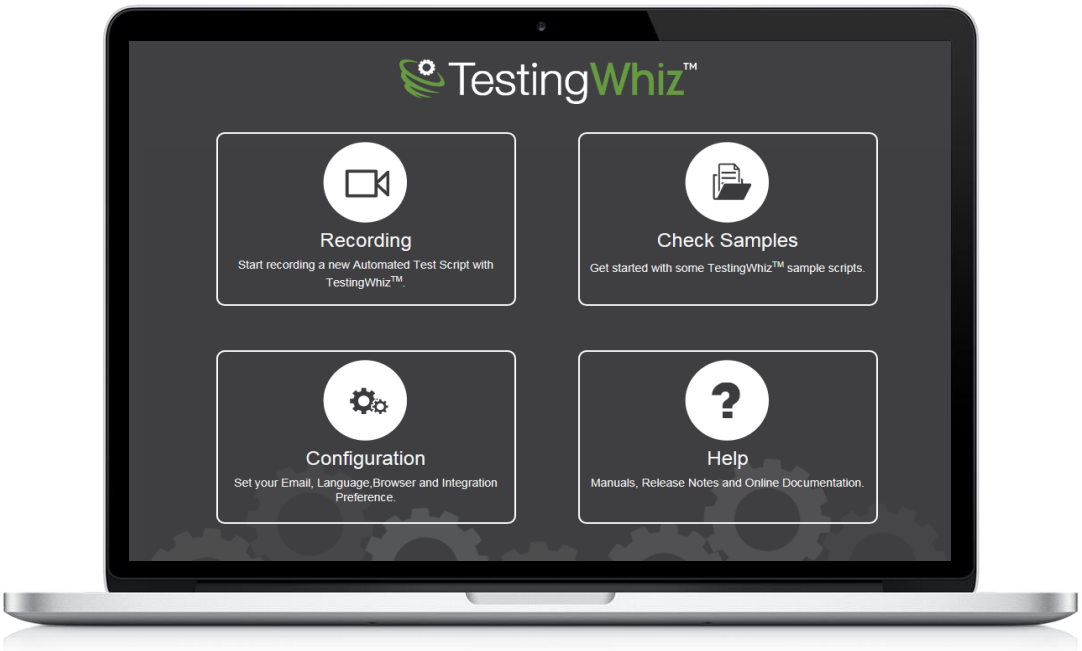
TestingWhiz

Code Less, Test More



TESTINGWHIZ

USER MANUAL

TestingWhiz Version: 6.0.0

Document Created: Nov 14, 2017

Table of Contents

1 GENERAL INFORMATION

1.1 Target Audience

This manual is intended to help QA engineers and software testers automate the testing of web and cloud-based applications using TestingWhiz. It can also be used by Business Analysts, Project Managers, Test Leads and other Stakeholders who are involved in testing activities for analysis and estimation purposes.

1.2 System Requirements

Operating System: Windows XP/ Windows Vista/ Windows 7/ Windows 8/Windows 10

Processor: Intel Pentium 4 or later

RAM: 2 GB (4 GB Recommended)

Free Disk Space: 500 MB

Java Version: JRE 8 or later

1.3 Platform Support

TestingWhiz can be used to create Automation Test Scripts only on Windows. However, the test scripts created using TestingWhiz can be executed on different operating systems like Mac and Linux. For information on how TestingWhiz Automation Tests can be run on MAC or Linux, please contact us.

1.4 Browser Support

TestingWhiz supports

- Internet Explorer: Version 9, 10 and 11
- Mozilla Firefox: Version 50 to Version 54
- Google Chrome: Version 55 to Version 58
- Edge: Version 25
- Headless Version 2

User can use any of the above browsers as a default browser to execute the Automation Test Scripts created using TestingWhiz.

Refer Section - **Configuration** to learn how to set the default browser.

Note: TestingWhiz also supports HTML5 partially.

1.5 Mobile Support

TestingWhiz also comes with a Mobile support to perform mobile and web testing on Android and iOS devices. Mobile test execution can be done by connecting the real device with the system or via Simulator.

Android Device Support

TestingWhiz provides mobile Web and Native test execution on Android Mobile Phones & Tablets. Currently it supports the following Android versions:

- Gingerbread (2.3)
- Honeycomb (3.0)
- Ice Cream Sandwich (4.0)
- Jelly Bean (4.3)
- KitKat (4.4)
- Lollipop (5.0)
- Marshmallow (6.0)
- Nougat 7.0

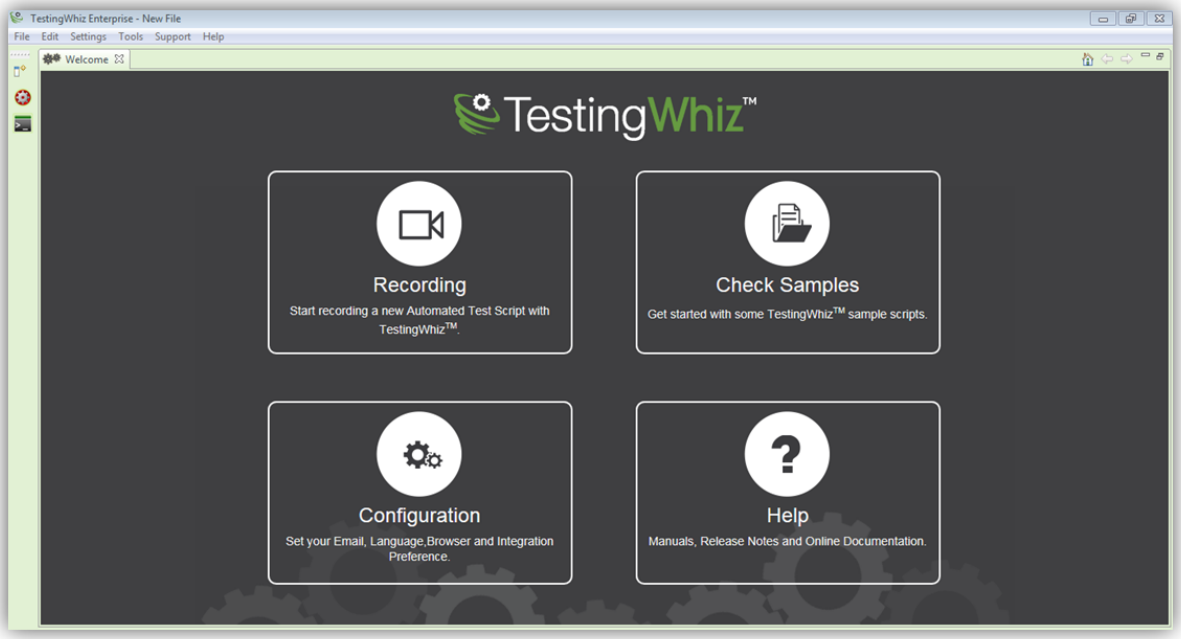
iOS Device Support

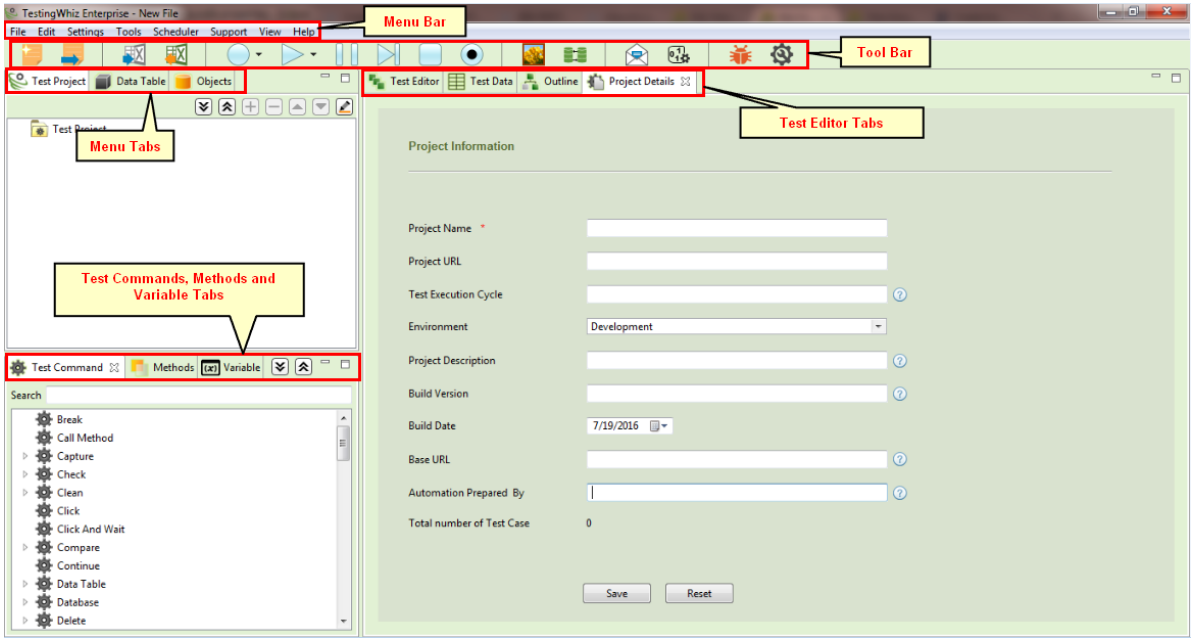
TestingWhiz provides mobile web test execution on iOS devices like iPhones and iPads. Currently it supports the following iOS versions:

- iOS 7.1
- 8.x
- 9.0, 9.1, 9.2

2 UNDERSTANDING TESTINGWHIZ

TestingWhiz has a straightforward user-interface for quick, effective and trouble-free testing.

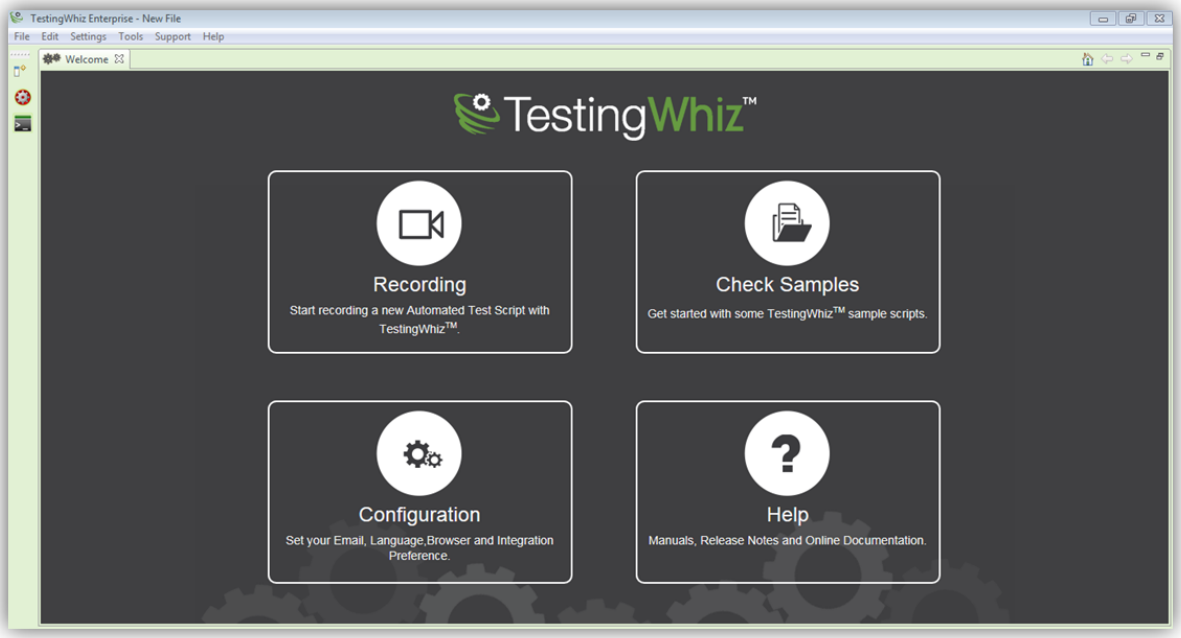




2.1 Welcome Screen

When a user launches TestingWhiz tool, a Welcome Screen appears highlighting four functions with a brief explanation of each function. User can simply click on the function name (Recording, Configuration, Check Samples or Help) to perform it.

- **Recording**: To record test case execution steps.
- **Configuration**: To set configuration preferences (Set up default browser, language, etc.)
- **Check Samples:** To open a list of sample scripts.
- **Help**: To get help for using TestingWhiz.

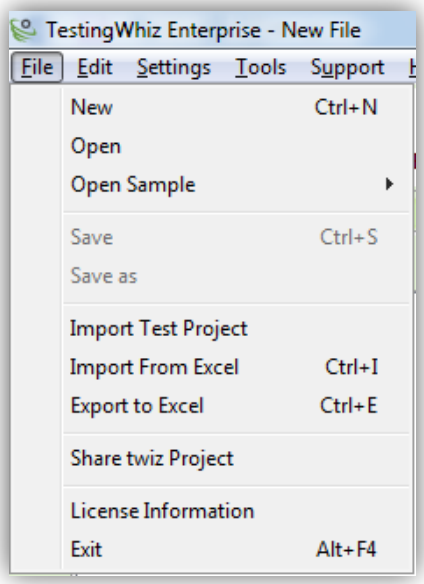


2.2 Menu Bar

Menu Bar contains important functions of TestingWhiz in a drop-down format. It provides instant access to different tasks and actions along with short-cuts for seamless test project execution.

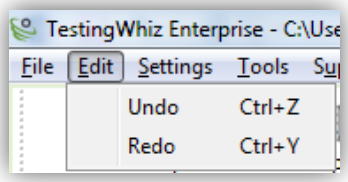
2.2.1 File

Open File menu to perform the following functions:



New	Click New to create a New Test Project
Open	Click Open to open an existing Project/Script (.twiz file)
Open Sample	Click Open Sample to view a list of sample script
Save	Click Save to save the Test Project
Save As	Click Save As to save the existing file with a new name and at the new location
Import Test Project	Click Import Test Project to import Test Suite/Cases/Data table/Methods to existing test project
Import from Excel	Click Import from Excel to import existing scripts from Excel file to TestingWhiz
Export to Excel	Click Export to Excel to save and export existing test scripts from TestingWhiz to Excel file
Share twiz Project	Click Share twiz Project to email existing .twiz Project/Script
License Information	Click License Information to view details of the license – type, start date, end date, edition, etc.
Exit	Click Exit to exit TestingWhiz application

2.2.2 Edit

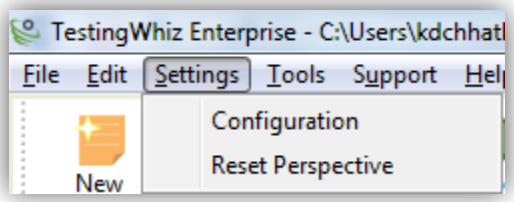


Use of Edit menu provides the following options:

Undo	Click Undo to undo/reverse the last step(s)
Redo	Click Redo to redo/repeat the last step(s)

2.2.3 Settings

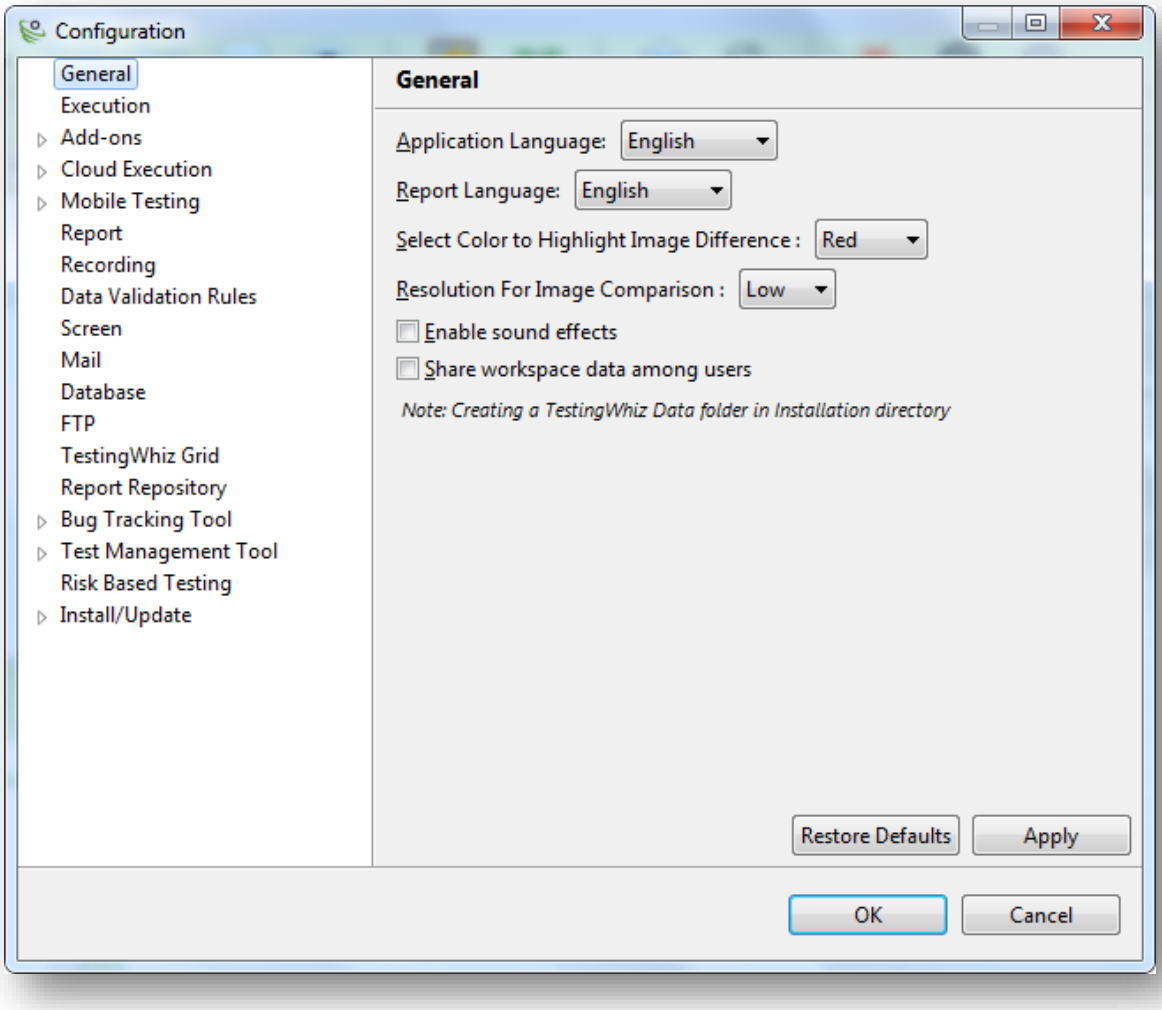
Use settings to set preferences and defaults in TestingWhiz



2.2.3.1 Configuration

Click Configurations to set up General preferences, Execution, Mobile Web Testing, Screen, Mail, Recording Rules, Database configuration, TestingWhiz Grid, Bug Tracking Tool and Test Management Tool etc.

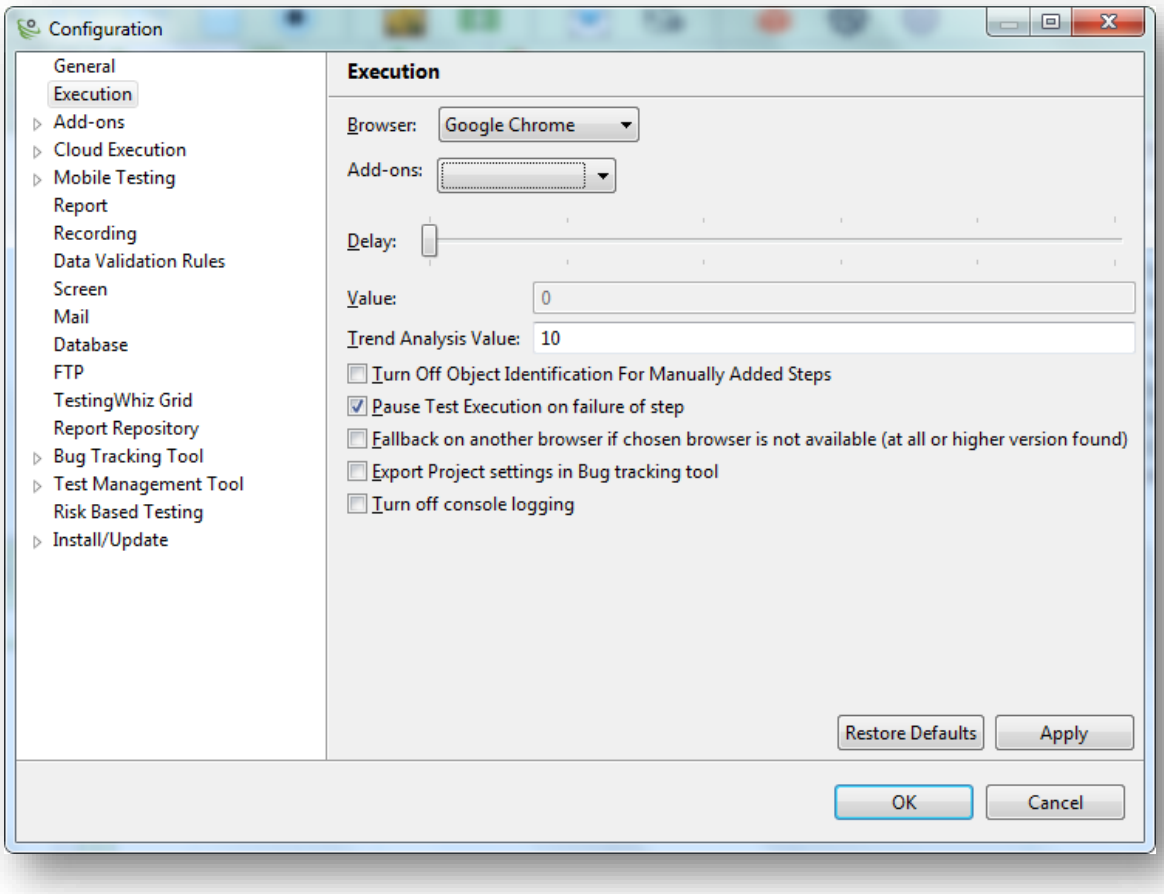
I. General: Set up General Preferences.



Application Language	Select default language to write application test cases – English, Français, Duetsch, Nederlands, Español, & Italiano. [**Note:** *Changes in language will be effective after Restart.*]
Report Language	Select default language to generate test report – English, Français, Duetsch, Nederlands, Español, & Italiano. [**Note:** *Changes in language will be effective after Restart.*]
Select Color to Highlight Image Difference	Select default color to highlight image difference – Red, Green & Yellow. [**Note:** *Image difference color will be highlight in "Image Comparison Report" after completion of test execution.*]
Resolution For Image Comparison	Select resolution of Image Comparison– Low or High. [**Note:** *Resolution for Image Comparison will be highlight in "Image Comparison Report" after completion of test execution.*]

Enable sound effects	Tick this option to enable or disable sound effect.
Share work space data among users	Tick this option to share work space data among other users.
Restore Defaults	Click Restore Defaults to reverse to default settings.
Apply	Click Apply to confirm and save the settings.

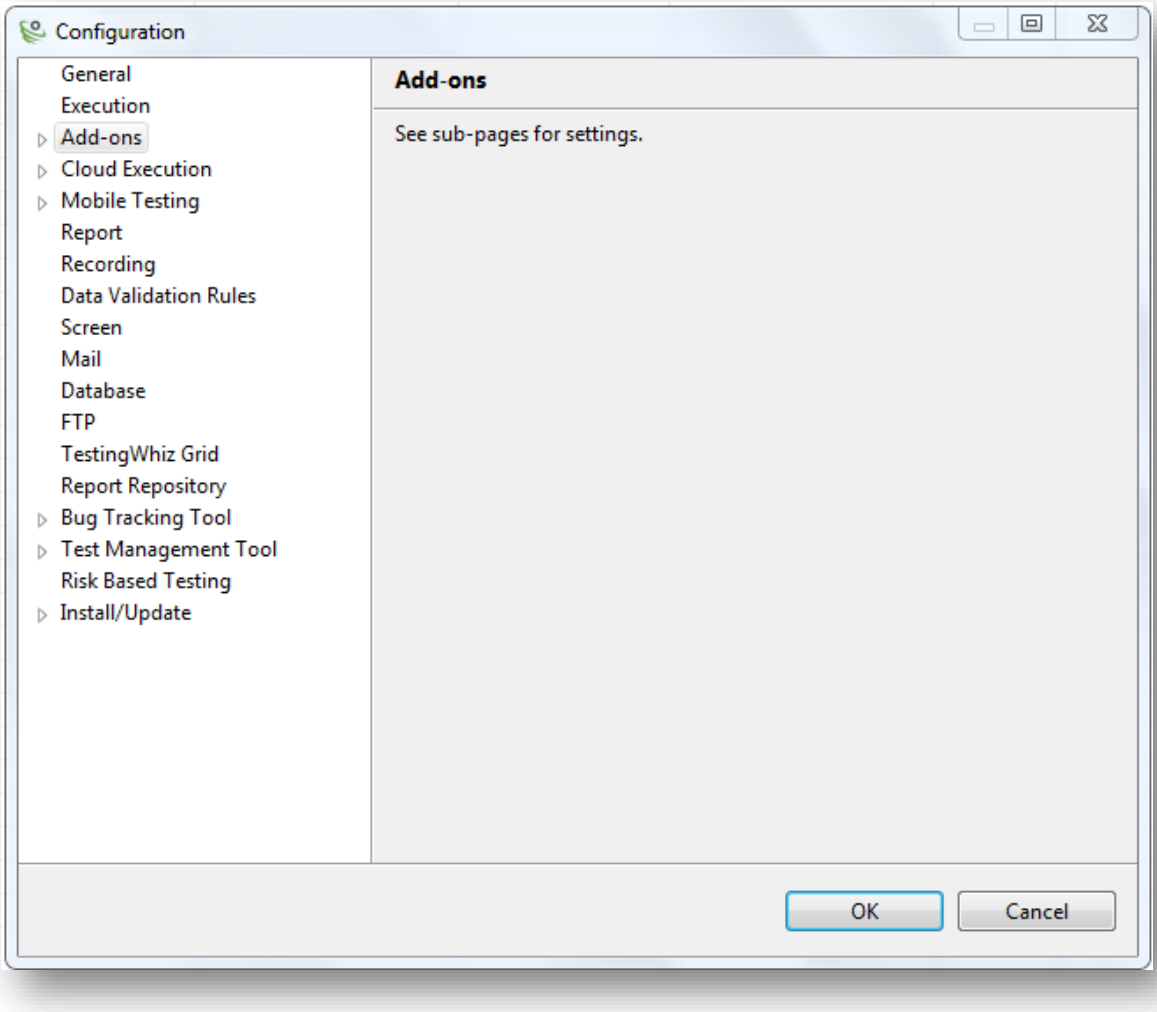
II. Execution: Set up Execution Preferences.



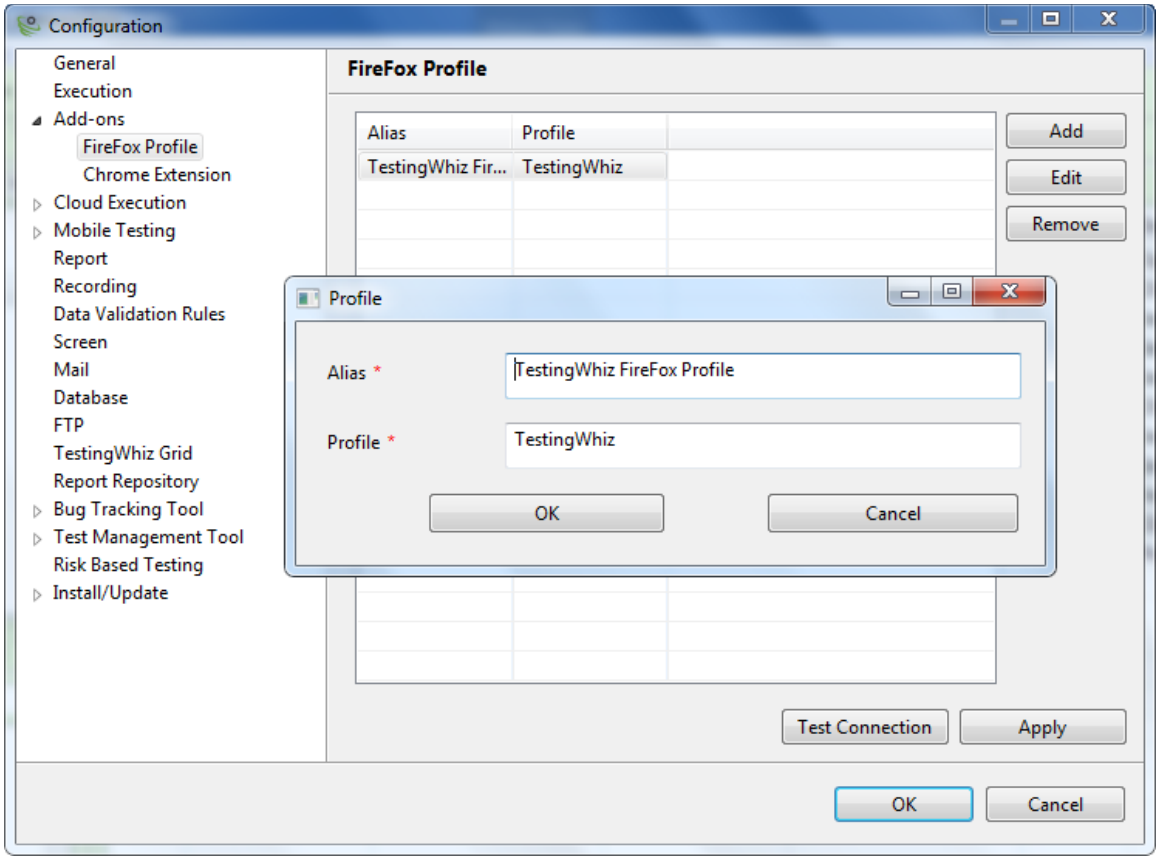
Browser	Select a default browser from the following: • Web Browsers – Internet Explorer, Mozilla Firefox, Google Chrome, Edge • Mobile – Android, iOS • Cloud – BrowserStack, Sauce Labs • Headless Execution
Add-ons	Select a default Add-on from the drop-down if any.

Delay Value	Set up speed to execute/play the test script. [**Note:** *Value will be displayed in Milliseconds.*]
Trend Analysis Value	Set up the maximum bars in Trend Analysis column. [**Note:** *By default, value appears as 10.*]
Turn Off Object Identification	Tick this option to turn off Object Eye feature for manually added steps.
Pause Test Execution	Tick this option to pause the test execution. [**Note:** *Allows user to select correct object at the time of test execution, also helps to handle dynamic object.*]
Fall back on another browser	Tick this option to switch to another browser if default browser is not compatible. [**Note:** *View details in execution logs in executed repor.t*]
Export Project details in Bug Tracking Tool	Tick this option to export project details into the bug tracking tool while opening a new bug ticket.
Turn Off Console Logging	Tick this option to turn off the console logging while execution. Only failed logs, will be displayed. This will improve your execution performance.
Restore Defaults	Click Restore Defaults to reverse to default settings.
Apply	Click Apply to confirm and save the settings.

III. Add-on: Setup Add-ons for Firefox & Chrome Extension Automation

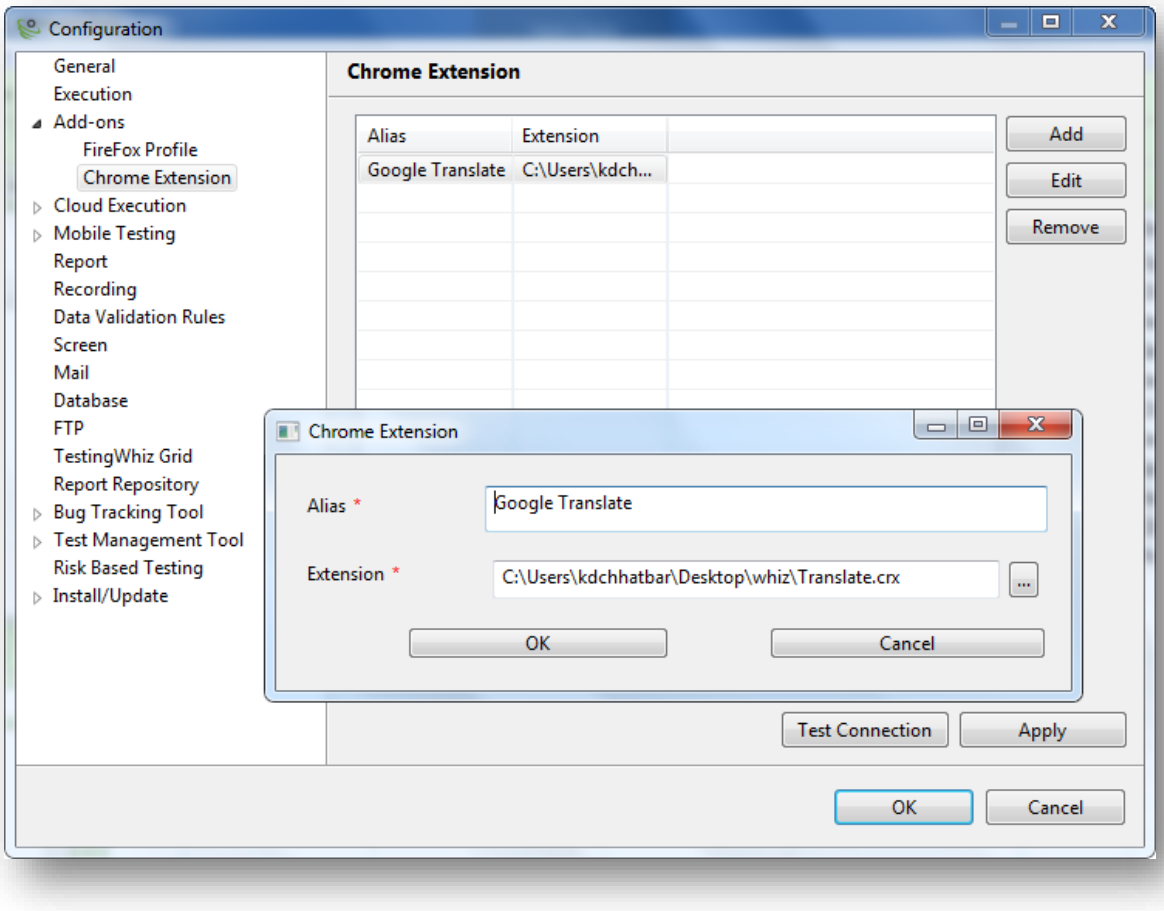


A. Firefox Profile Setup



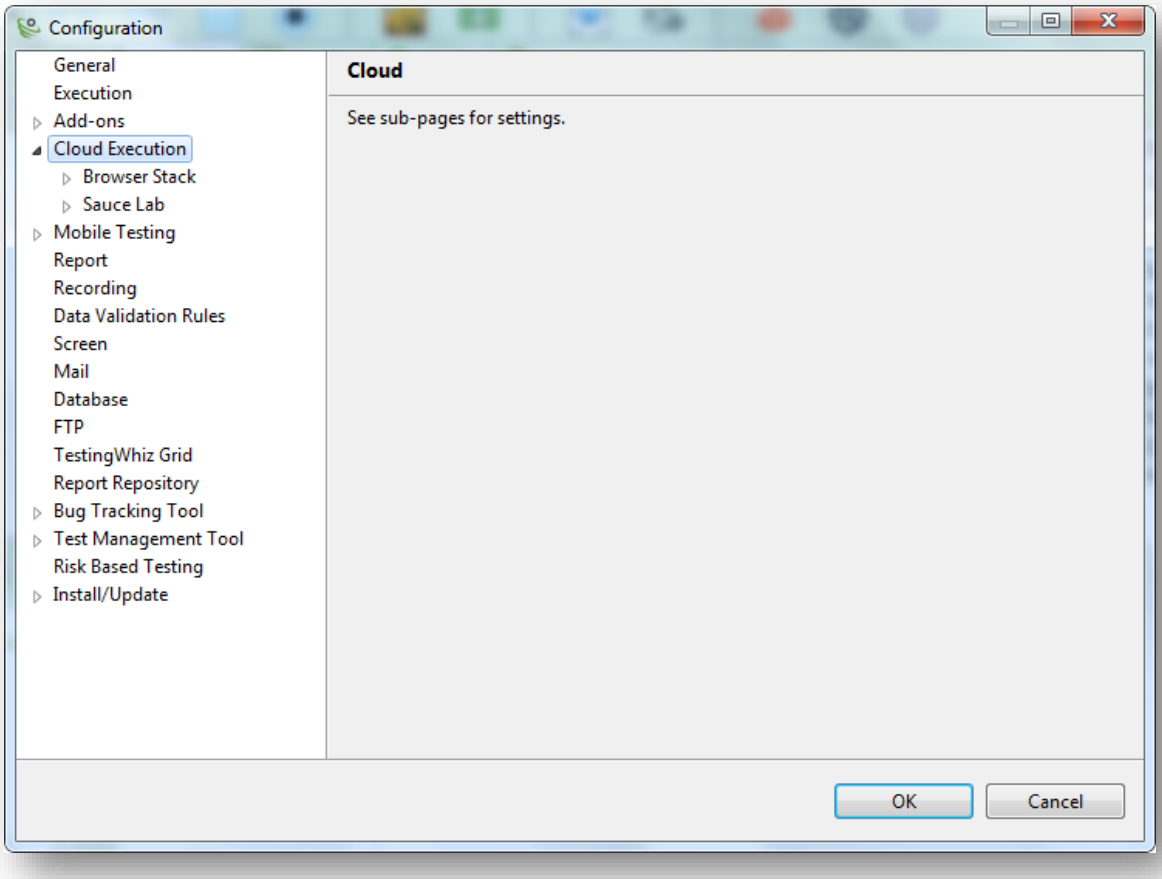
Alias	Enter the Alias for FireFox Profile.
Profile	Enter the Profile Name used for creating FireFox Profile.
Add	To add a new FireFox Profile.
Edit	To edit an existing FireFox Profile.
Remove	To remove an existing FireFox Profile.
Test Connection	Click Test Connection to test the connection with the FireFox Profile.
Apply	Click Apply to confirm and save the settings.

B. Chrome Extension Setup

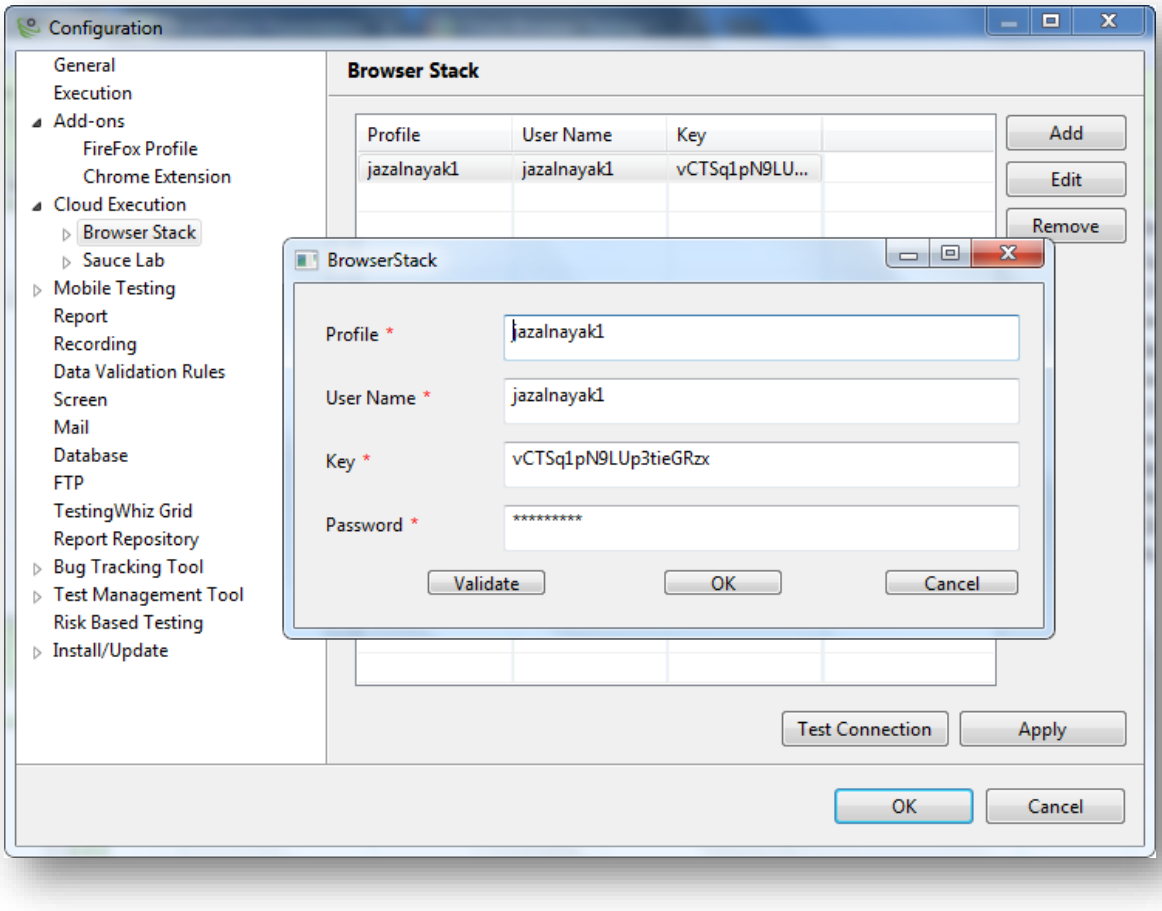


Alias	Enter the Alias for Chrome Extension.
Extension	Enter the Extension of the respective Extension.
Add	To add a new Chrome Extension.
Edit	To edit an existing Chrome Extension.
Remove	To remove an existing Chrome Extension.
Test Connection	Click Test Connection to test the connection with the Chrome Extension.
Apply	Click Apply to confirm and save the settings.

IV. Cloud Execution: Setup BrowserStack OR Sauce Labs Test Execution Preferences

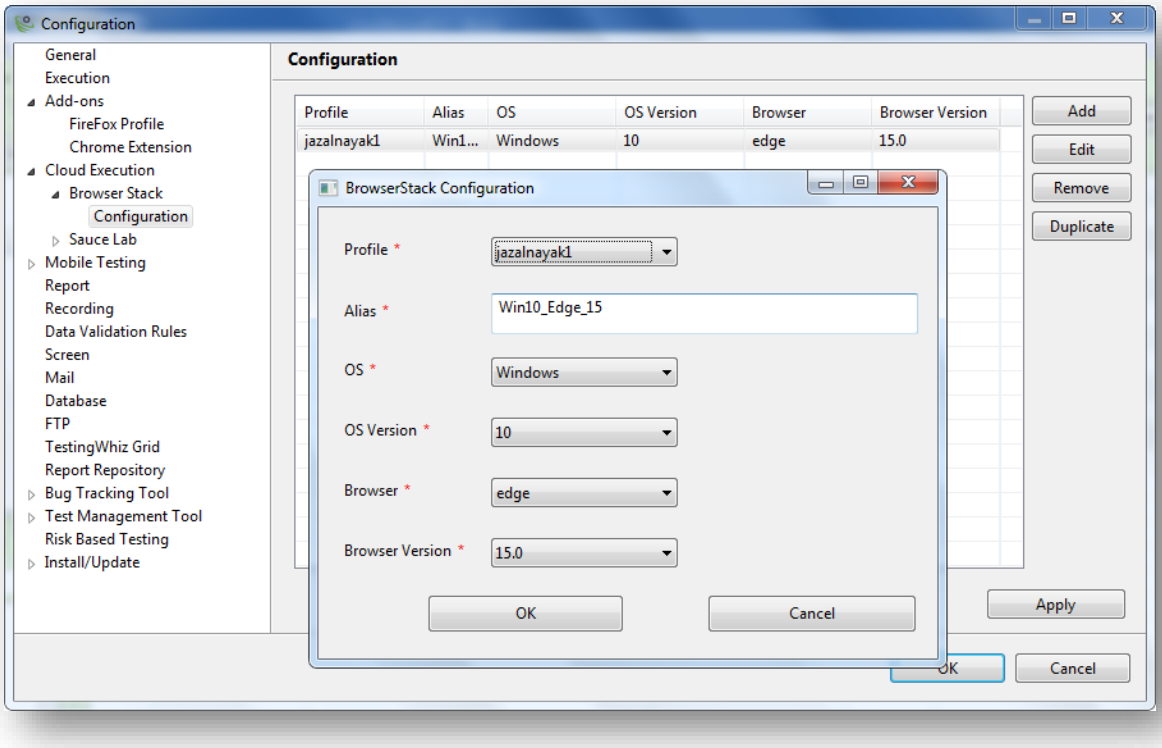


A. *BrowserStack Setup*



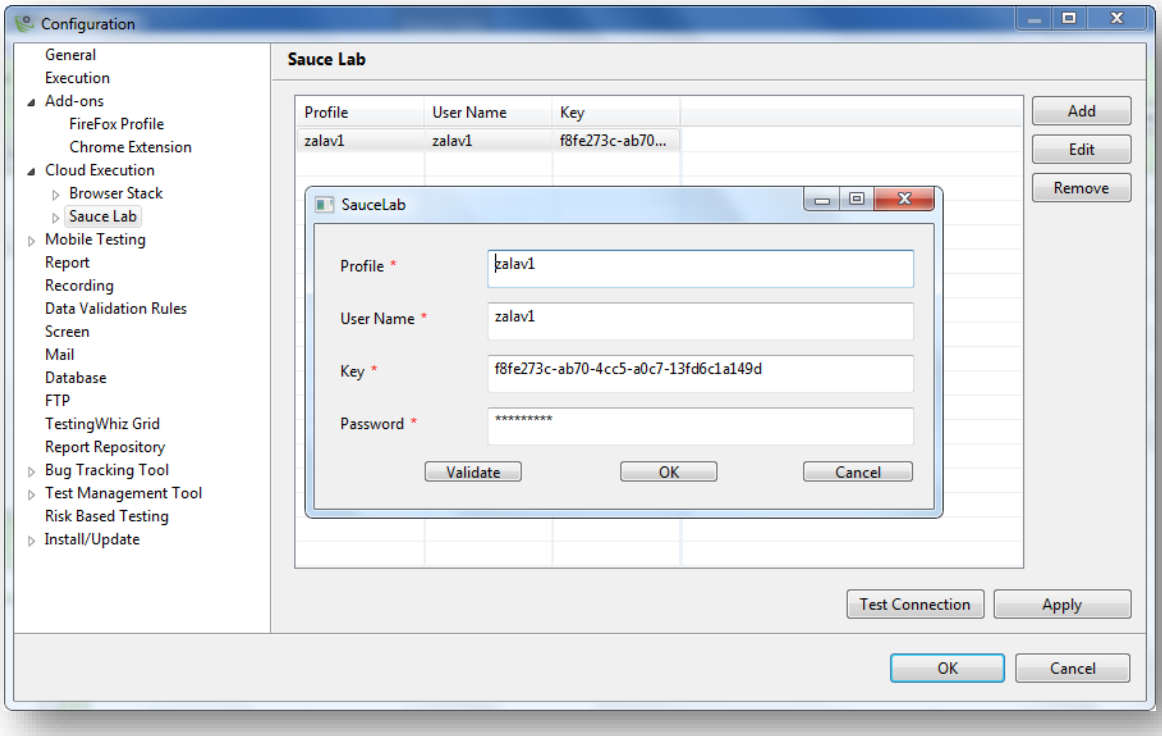
Profile	Enter the Profile name as you per your interest.
Username	Enter the Username of the BrowserStack account.
Key	Enter the key provided by BrowserStack.
Password	Enter the Password of BrowserStack account.
Validate	This button will validate the credentials of BrowserStack account.

i. BrowserStack Configuration



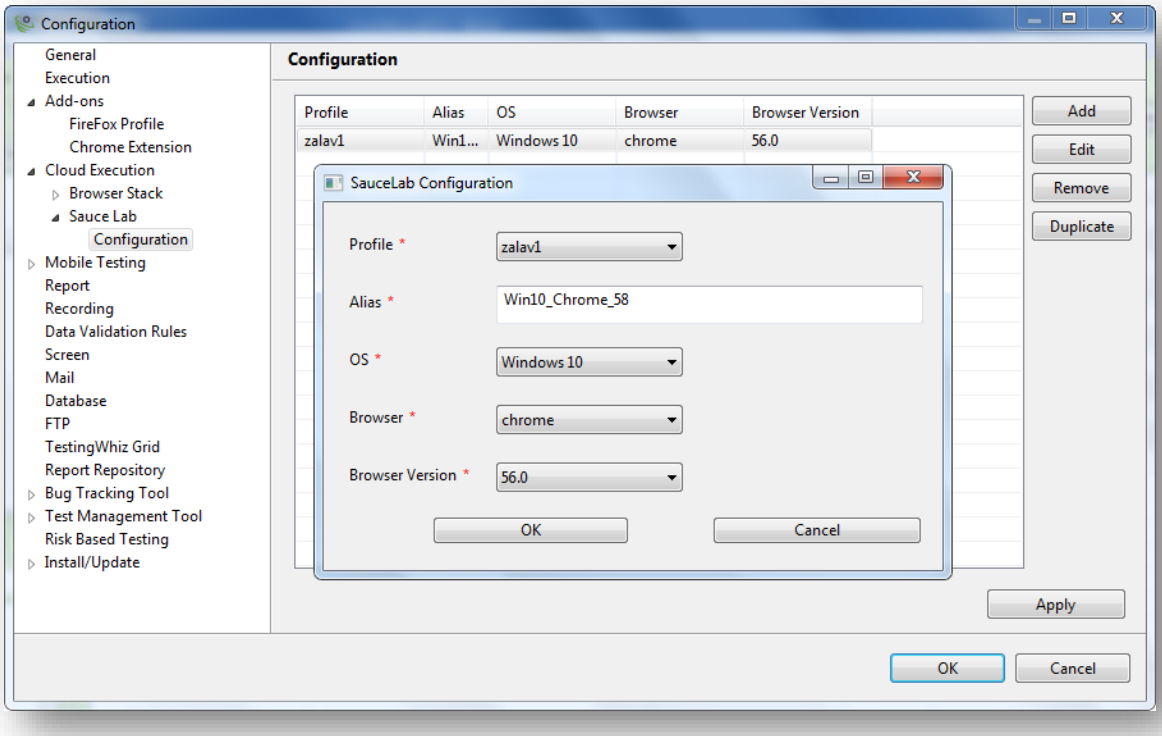
Profile	Select the BrowserStack profile from the dropdown list.
Alias	Enter Alias for BrowserStack account.
OS	Select the OS from the dropdown.
OS Version	Select the OS version from the dropdown.
Browser	Select the Browser from the dropdown.
Browser Version	Select the version of the Browser from the dropdown.

B. Sauce Labs Setup



Profile	Enter the Profile name as you per your interest.
Username	Enter the Username of the Sauce Labs account.
Key	Enter the key provided by Sauce Labs.
Password	Enter the Password of Sauce Labs account.
Validate	This button will validate the credentials of Sauce Labs account.

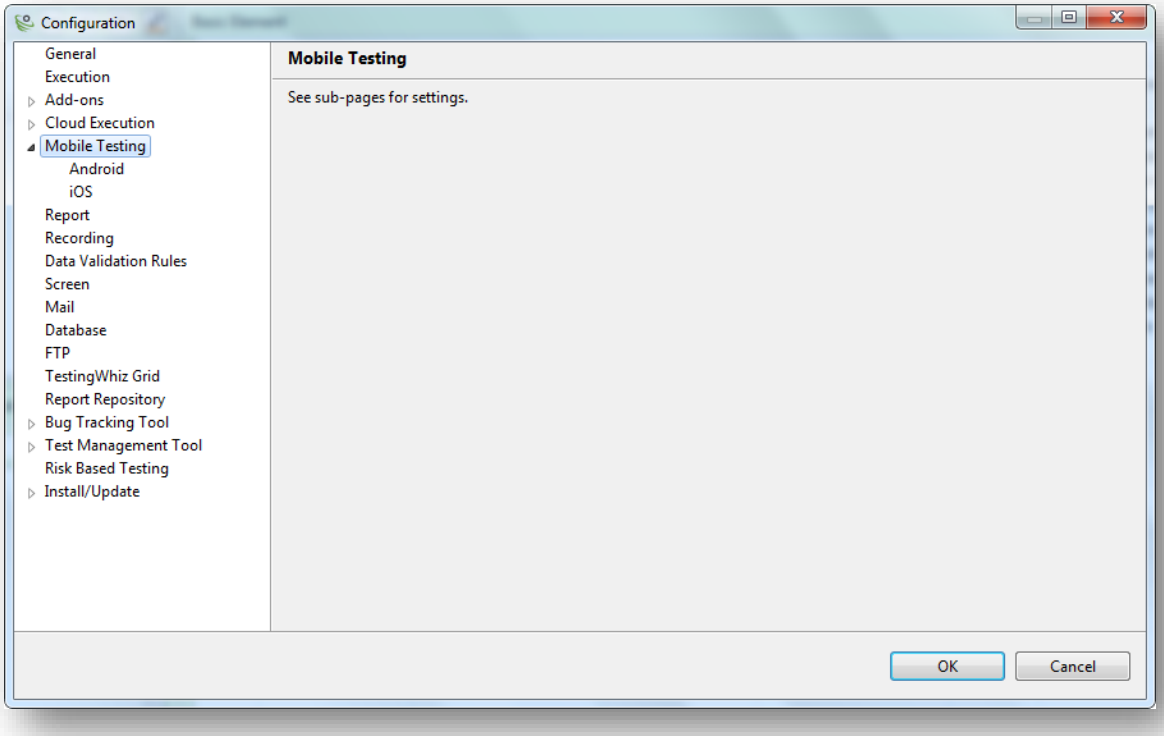
i. Sauce Labs Configuration



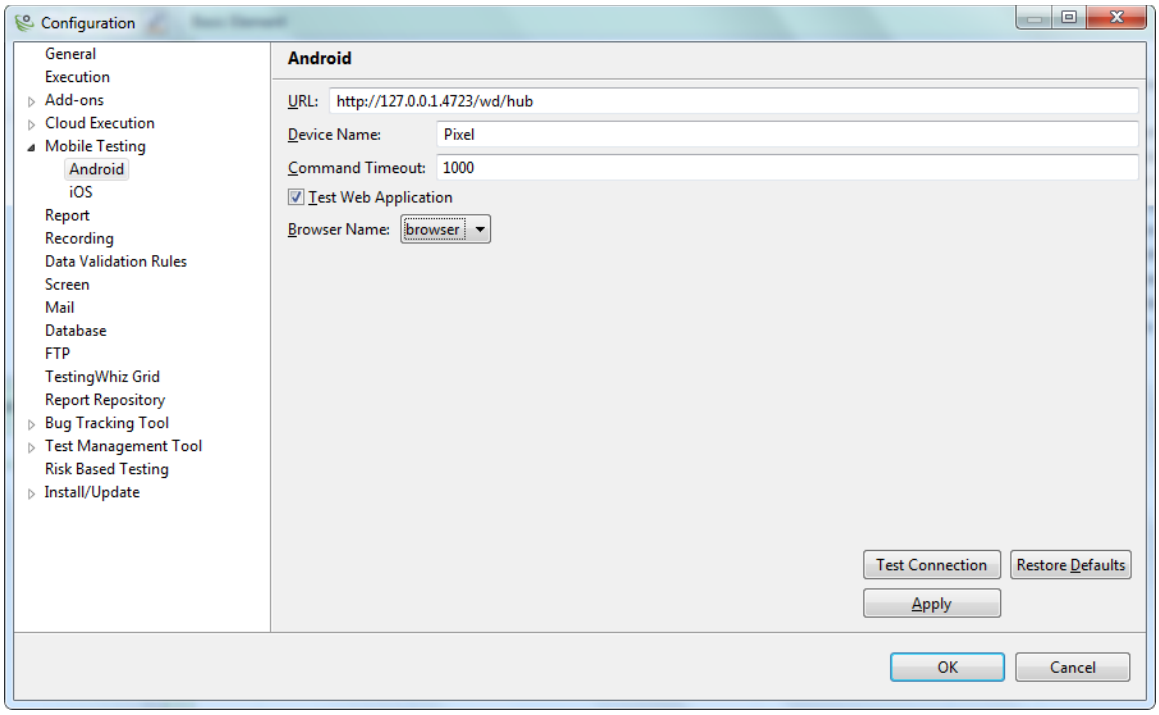
Profile	Select the Sauce Labs profile from the dropdown list.
Alias	Enter Alias for Sauce Labs account.
OS	Select the OS from the dropdown.
Browser	Select the Browser from the dropdown.
Browser Version	Select the version of the Browser from the dropdown.

V. Mobile Testing: Set up Android or iOS Test Execution Server Preferences.

Select the platform between Android and iOS for mobile web testing.

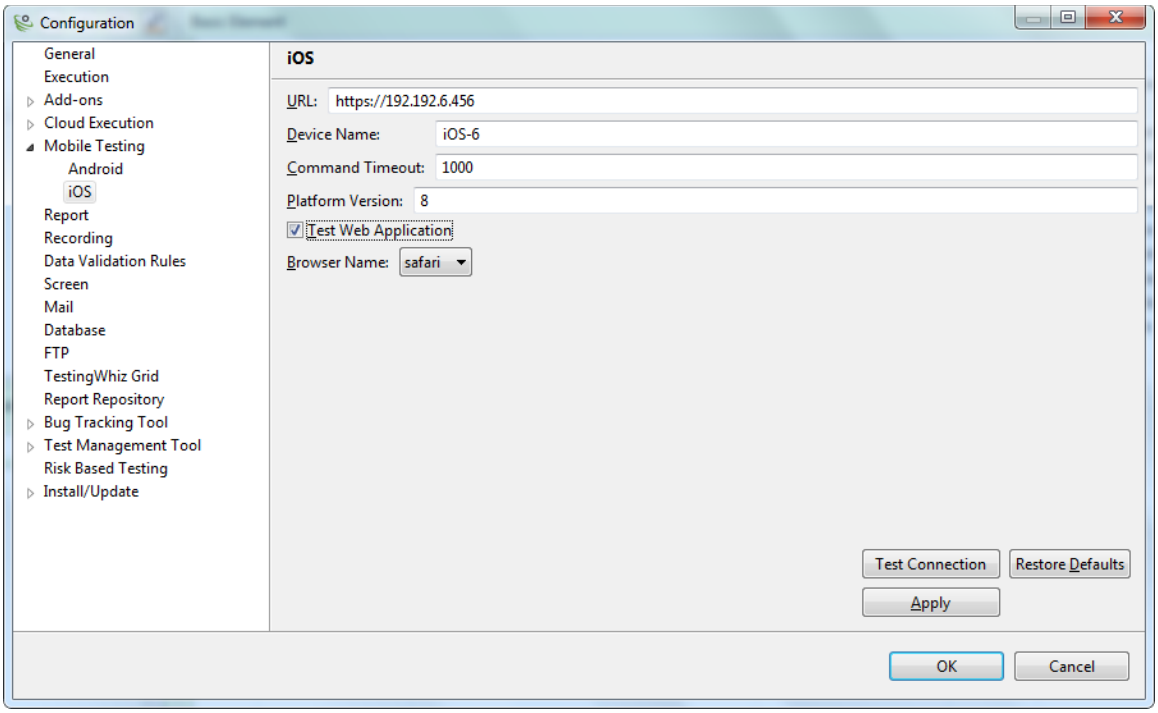


A. Mobile Web Testing for Android



URL	Enter the Server URL – URL of an Appium server which is connected to Android device.
Device Name	Enter the device name in case of simulation testing.
Command Timeout	Enter the time in milliseconds to test the connection of Android device.
Test Web Application	Tick this option to test web application in the Android device.
Browser	Select a browser from the drop-down.
Test Connection	Click Test Connection to test the connection with the Appium Server.
Restore Defaults	Click Restore Defaults to default settings.
Apply	Click Apply to configure and save the settings.

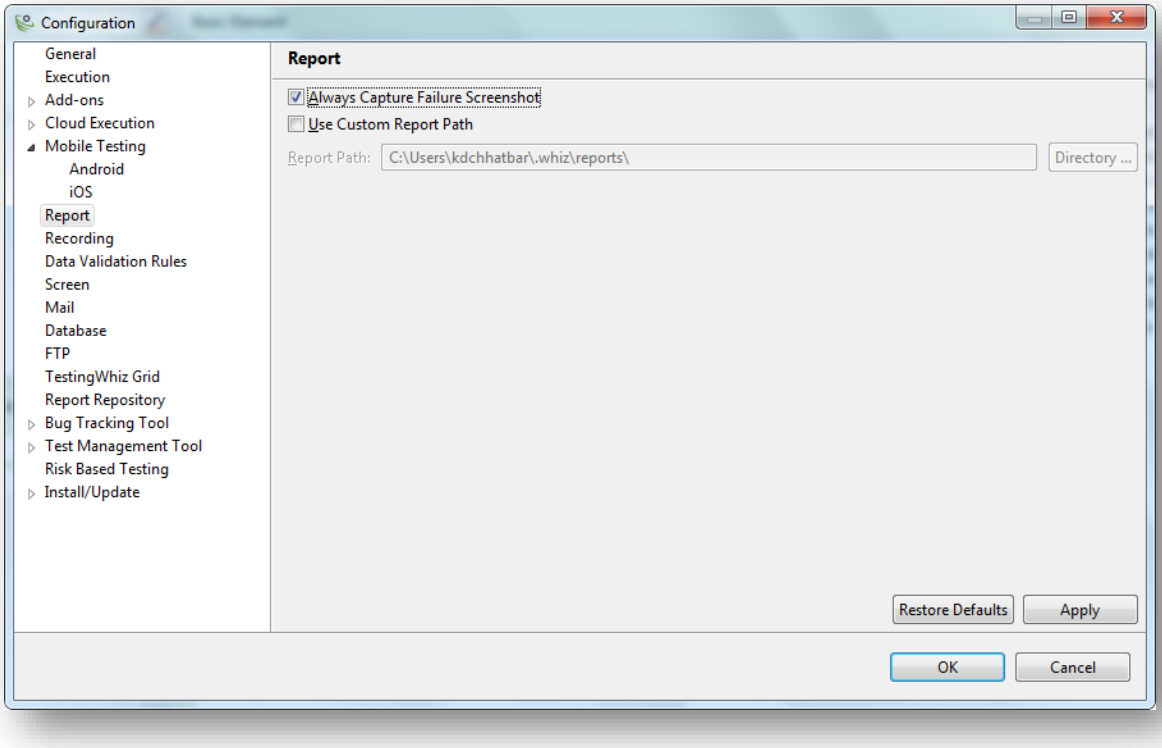
B. Mobile Web Testing for iOS



URL	Enter the URL of an Appium Server which is connected to iOS device.
Device Name	Enter the device name in case of simulation testing.
Command Timeout	Enter the time in milliseconds to test the connection of iOS device
Platform Version	Enter the platform version of the iOS device.

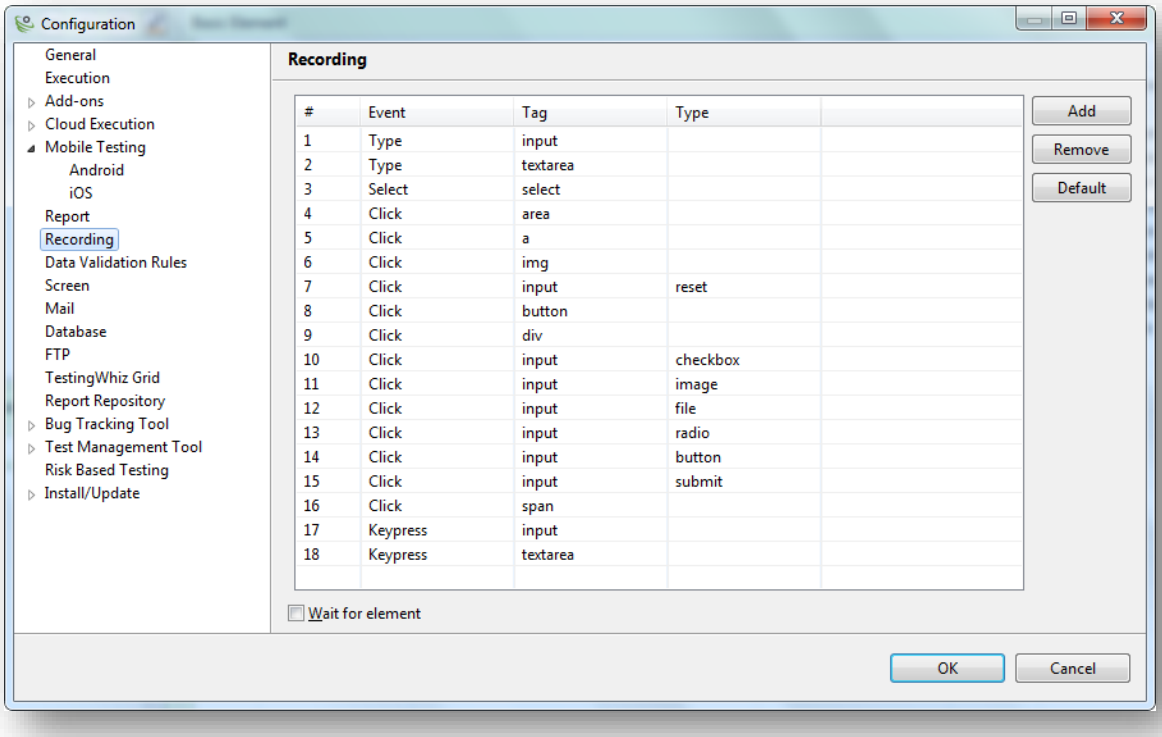
Test Web Application	Tick the checkbox to test the Web Application in iOS device.
Browser	Select a browser from the drop-down.
Test Connection	Click Test Connection to test the connection with the Appium Server.
Restore Defaults	Click Restore Defaults to default settings.
Apply	Click Apply to configure and save the settings.

VI. Report: Set Up Execution Report Preferences.



Always Capture Failure Screenshot	Tick this option to capture screenshots of the failed test cases
Use Custom Report Path	Tick this option to store reports at any other custom path of your choice rather than the default location
Restore Defaults	Click Restore Defaults to reverse to default settings
Apply	Click Apply to confirm and save the settings

VII. Recording: Set up Recording Rules to be Performed While Recording Test Scripts using Internal as well as External Browsers.



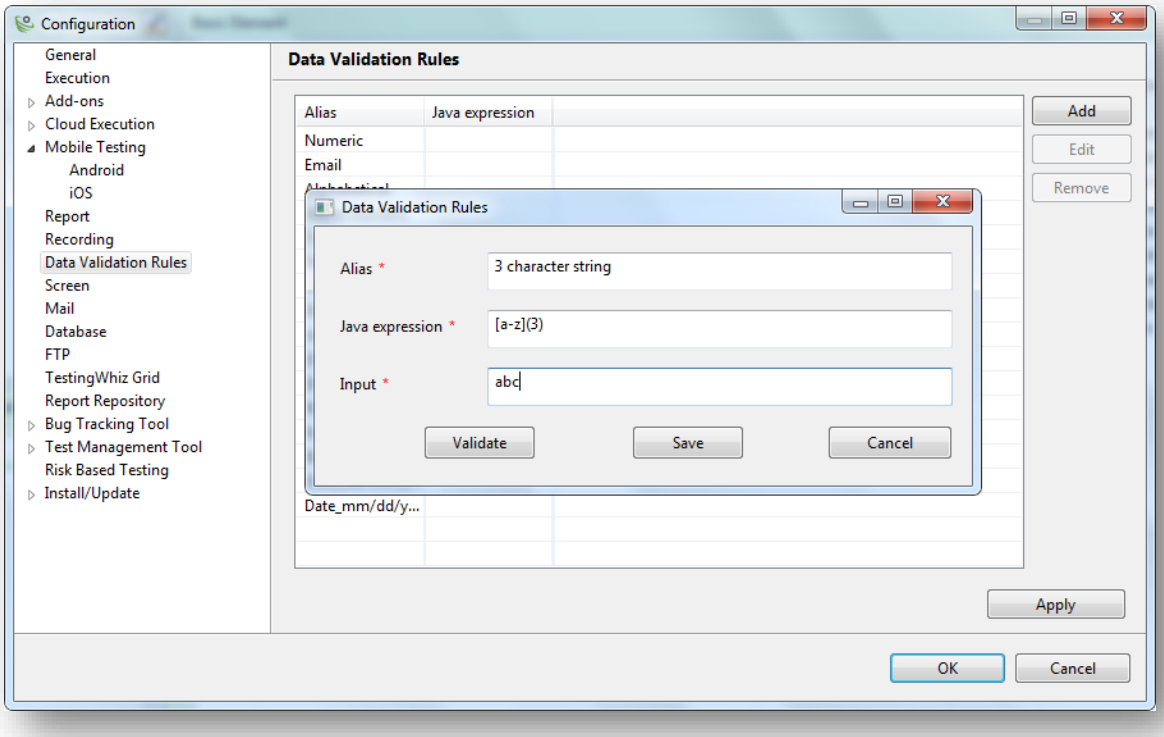
Add	Click Add to insert more recording rules to suit the recording behavior before generating scripts with Record and Playback feature
Remove	Click Remove to delete a particular or a set of recording rules from the existing rules to suit the recording behavior requirement
Default	Click Default to get the default list of events (recording rules)

Refer Section – **Record to Create Test Script Using Internal Browser** to learn more about Recording using Internal Browser.

Refer Section - **Record to Create Test Script Using External Browser** to learn more about Recording using External Browser.

[**Note:** *By default, TestingWhiz provides 18 Events (recording rules).*]

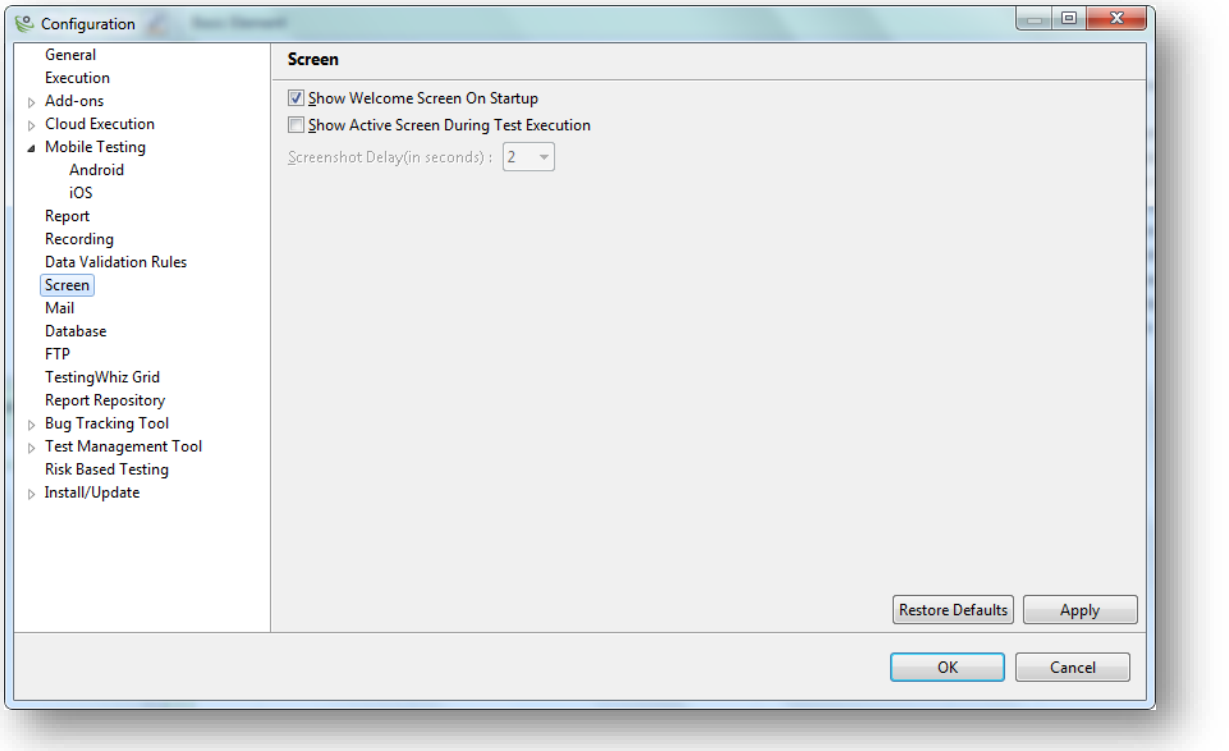
VIII. Data Validation Rules: Set up Data Validation Rules for The Process of Data Cleansing.



[**Note:** *By default, TestingWhiz provides 9 Alias for data validation.*]

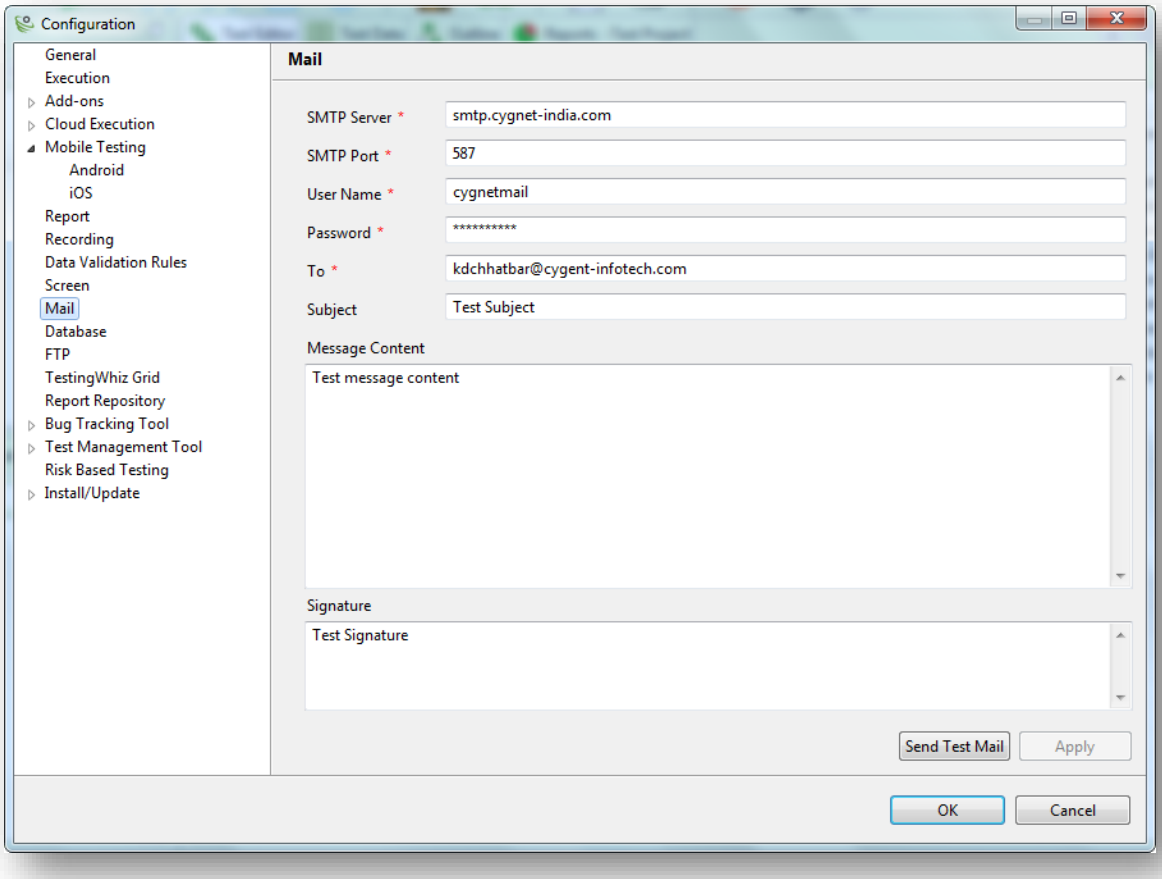
Add	Click Add to insert more Alias, Java Expression and Input for data validation.
Edit	Click Edit to edit Alias, Java Expression and Input for data validation.
Remove	Click Remove to remove a particular or a set of Alias and related Java Expression and Input to suit the requirement for data validation.

IX. Screen: Set up Display Preferences.



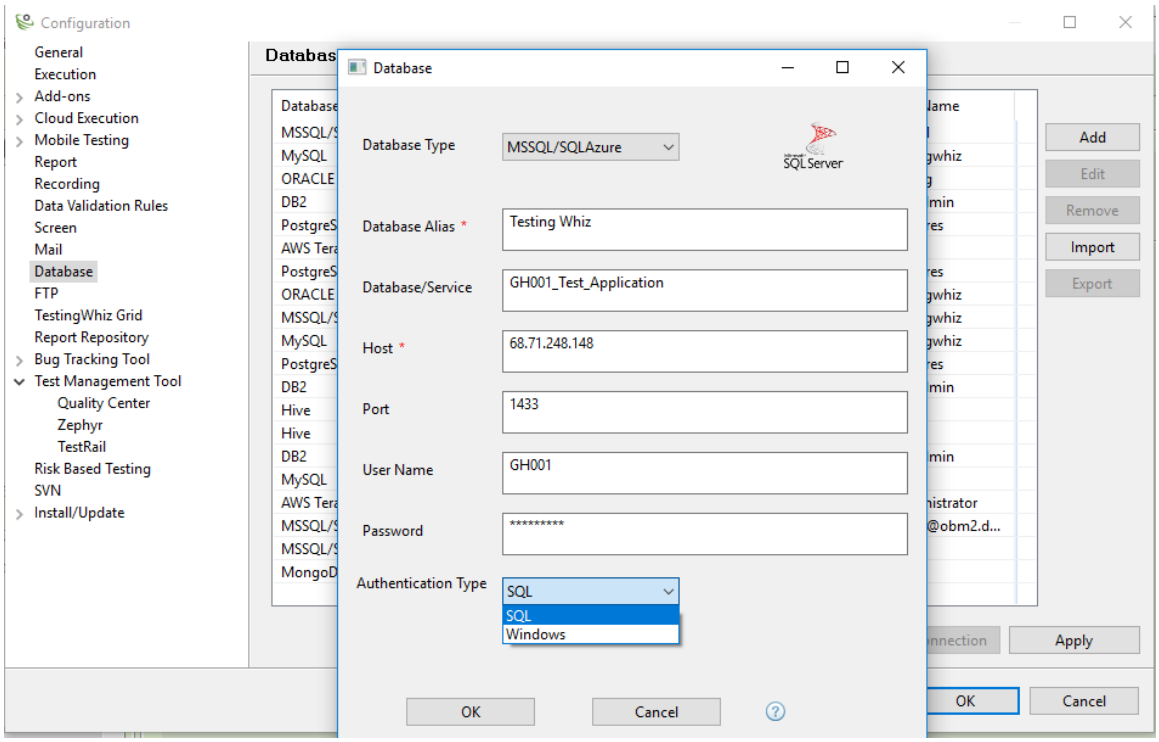
Show Welcome Screen	Tick this option to view Welcome Screen on every start-up of the application.
Show Active Screen	Tick this option to view Active Screen or current screen during test execution.
Restore Defaults	Click Restore Defaults to reverse to default settings.
Apply	Click Apply to confirm and save the settings.

X. Mail: Configure Mail Account with TestingWhiz to Send Test Reports through Mail.



SMTP Server	Enter SMTP Server (Outgoing Mail Server) details of the User's Email Address.
SMTP Port	Enter SMTP Port details to authorize User's Email Address.
Username & Password	Enter Users Server credential - Username & Password to configure Email Address in TestingWhiz.
To	Enter Recipient's Email Address.
Subject	Enter Subject of the Email.
Message Content	Enter Message Content (Optional).
Signature	Enter Signature (Optional).
Send Test Mail	Click Send Test Mail to test whether the Email has been configured correctly or not (Optional).
Apply	Click Apply to confirm and save the settings.

XI. Database: Configure Database to Fetch Data Directly into the Data table and Run Raw Queries.

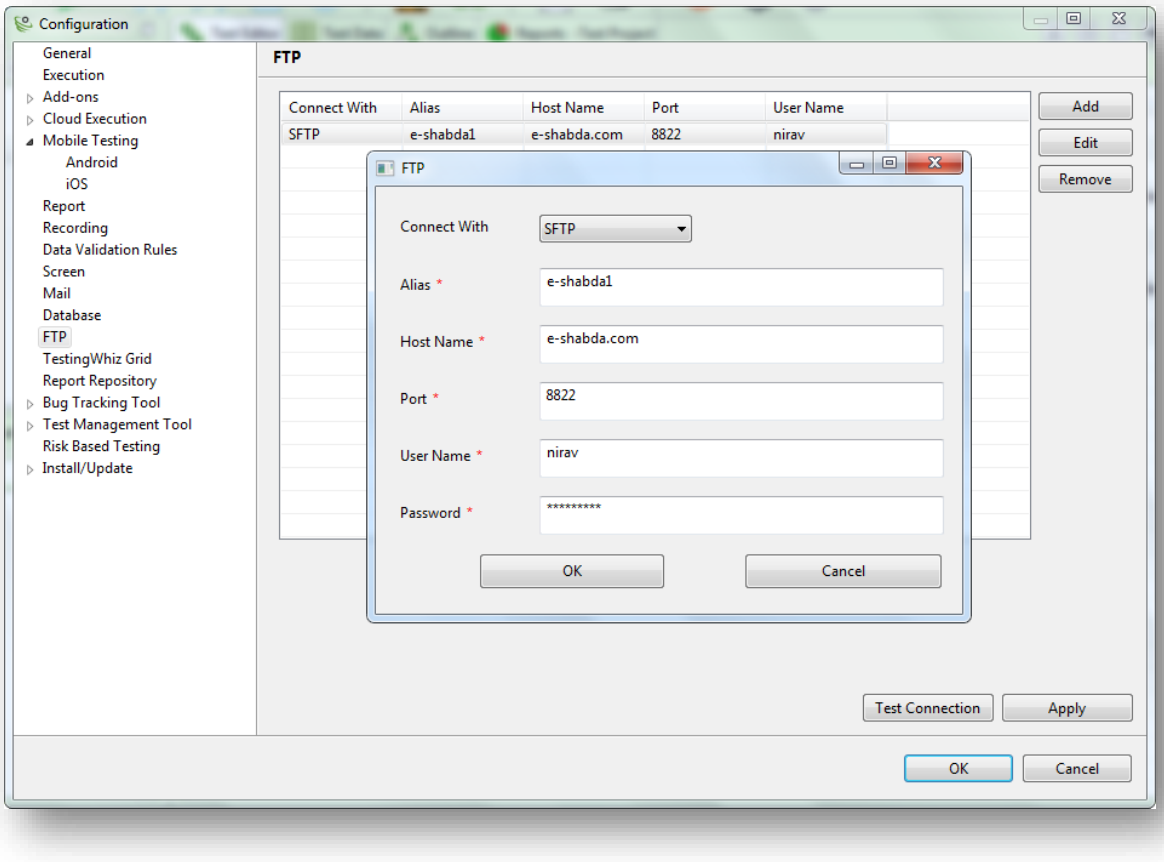


Database Type	Click Add and select database type from the drop-down – MySQL, MSSQL, ORACLE, DB2, PostgreSQL, AWS Teradata or Hive.
Database Alias	Enter the Database Alias.
Database/Service	Enter the Database Name.
Host	Enter the location (IP address) where the database is to be hosted.
Port	Enter the Port of the Database.
User name & Password	Enter the User name & Password to authorize the Database.
Authentication Type	TestingWhiz also provides Windows Authentication login method supported by MSSQL/SQL Azure.
Edit	Click Edit to edit items in the Database.
Remove	Click Remove to remove particular items from the Database.
Import	Click Import to integrate Database Connection configuration in Testing Whiz.
Export	Click Export to save the Database configuration so that you don't have to repeat the connection setting procedure again.

Test Connection	Click Test Connection to test the connection with the Database.
Apply	Click Apply to configure and save the settings.

Note: Windows Authentication support has been provided for MSSQL/SQL Azure database only.

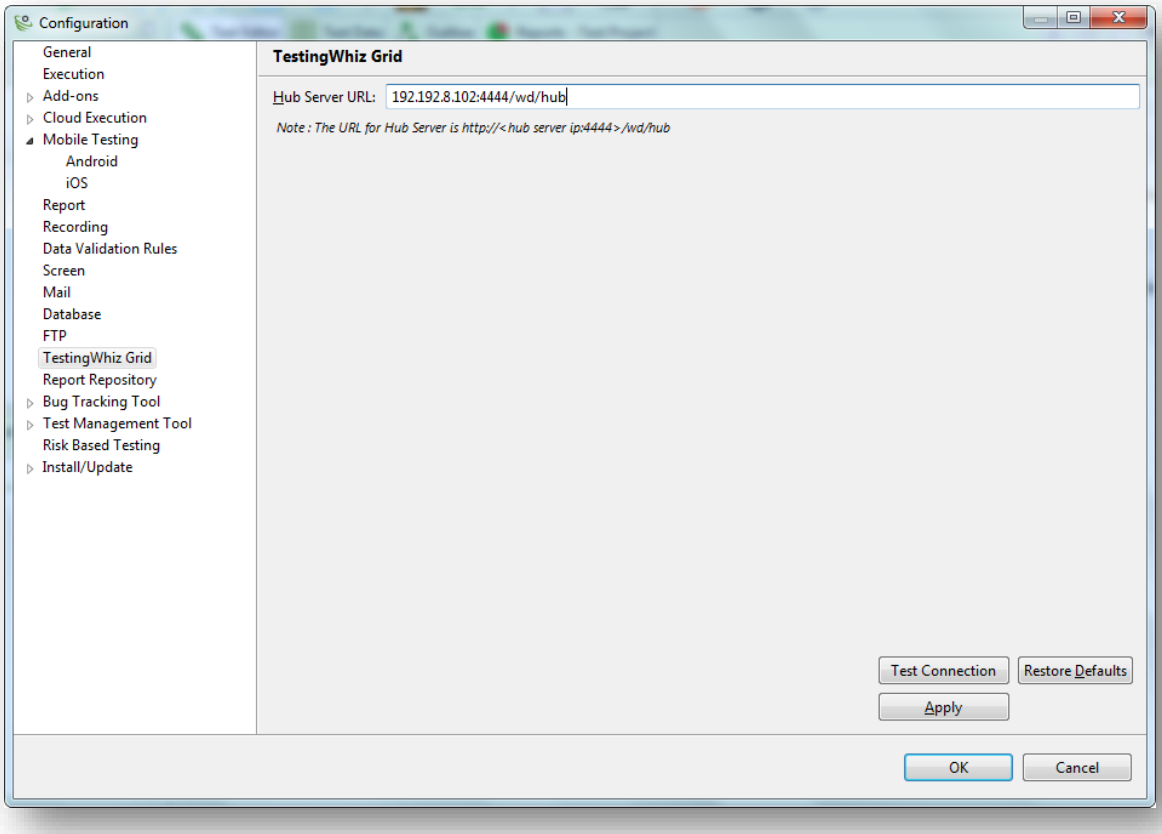
XII. FTP: Set up FTP to Upload Files to Server.



Connect With	Click Add and select FTP type from the drop-down – FTP or SFTP.
Alias	Enter the FTP Alias.
Host Name	Enter the location where the FTP is to be hosted.
Port	Enter the Port of the FTP.
Username & Password	Enter the Username & Password to authorize the FTP.
Edit	Click Edit to edit items in the FTP.
Remove	Click Remove to remove particular items from the FTP.

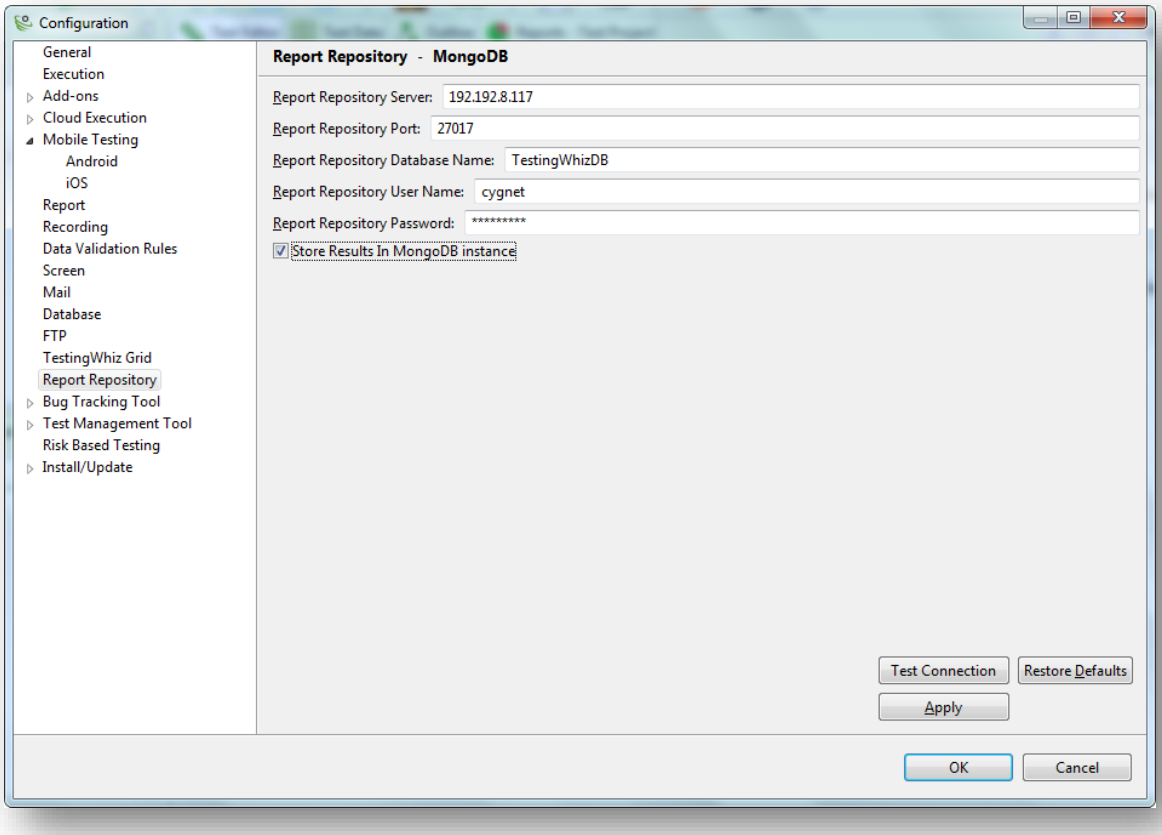
Test Connection	Click Test Connection to test the connection with the FTP.
Apply	Click Apply to configure and save the settings.

XIII. TestingWhiz Grid: Set Up TestingWhiz Grid to Distribute the Test Execution across Multiple Machines and Reduce the Execution Time.



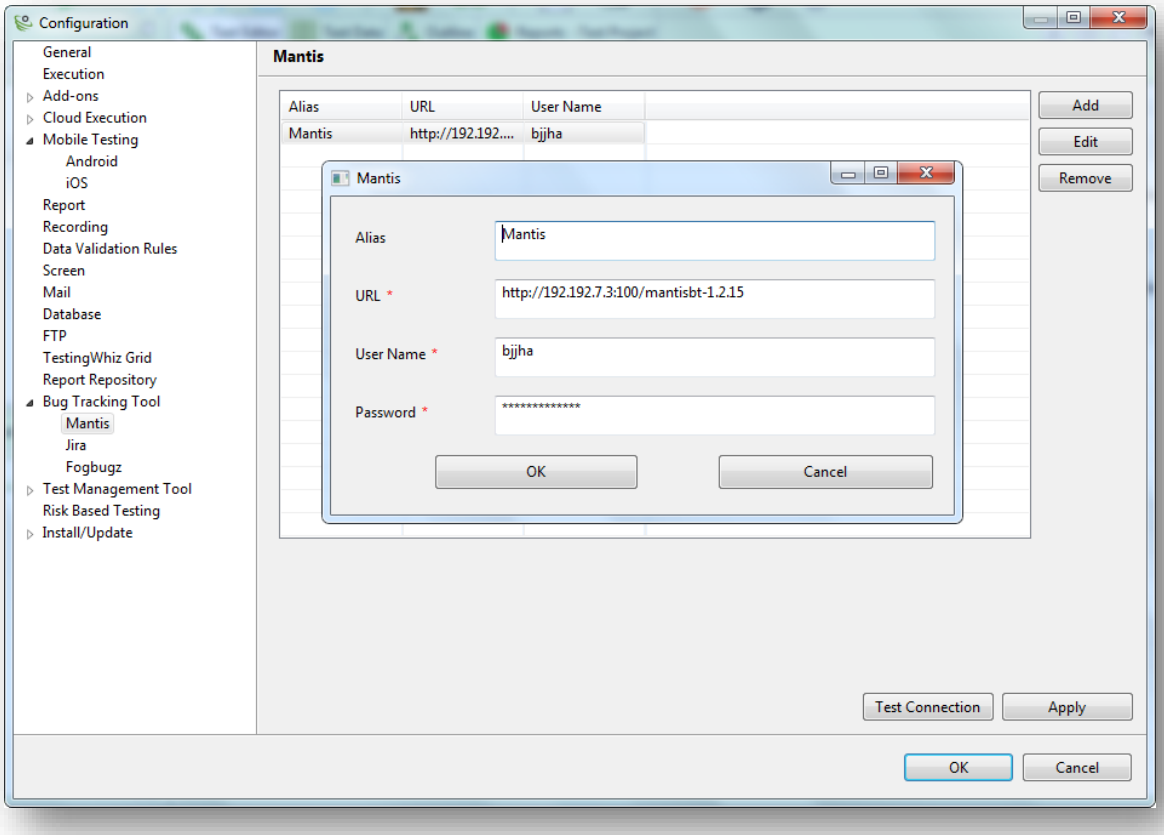
Hub Server URL	Enter Hub Server URL – URL of a centralized server/main machine which is connected with other machines.
Test Connection	Click Test Connection to test the connection with the Hub Server URL.
Restore Defaults	Click Restore Defaults to default settings.
Apply	Click Apply to configure and save the settings.

XIV. Report Repository: Set-up Report Repository to Store the Execution Reports in MongoDB Database



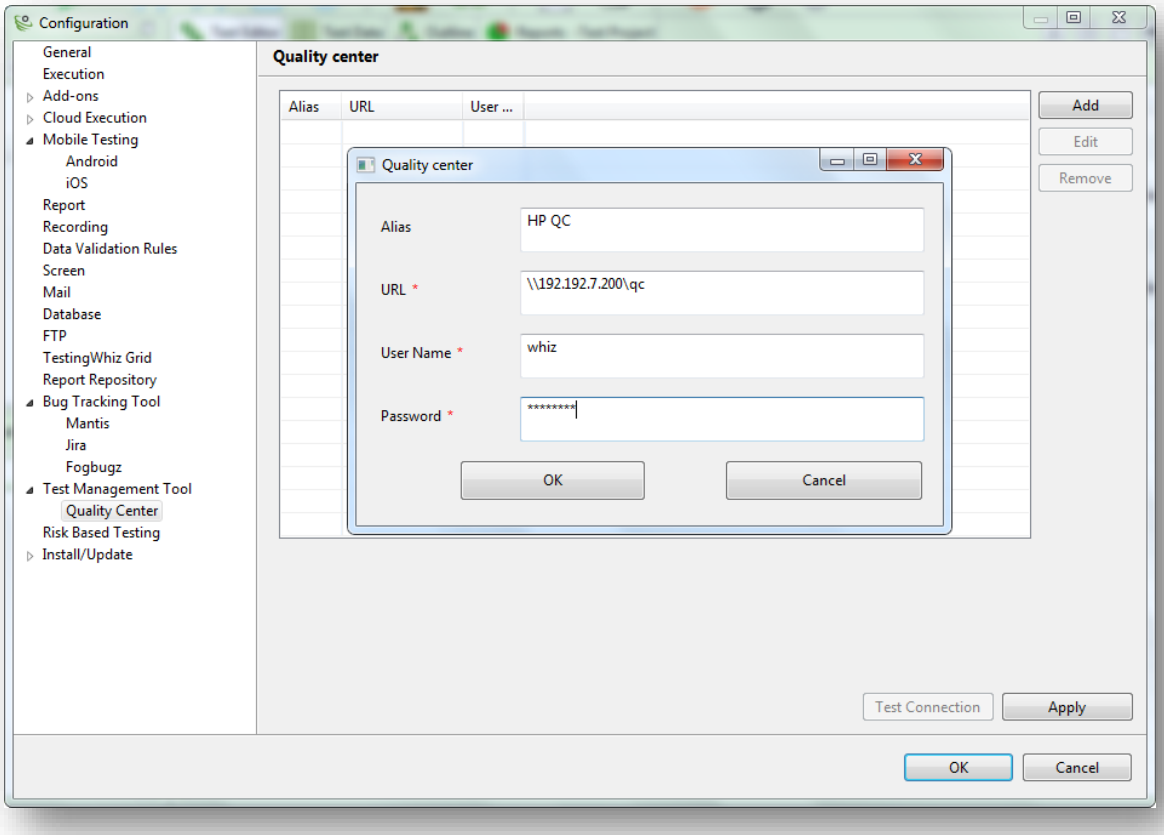
Report Repository Server	Enter the Server details.
Report Repository Port	Enter the Port number.
Report Repository Database Name	Enter the Database Name.
Report Repository Username	Enter the Username of Database.
Report Repository Password	Enter the Password of Database.
Test Connection	Click Test Connection to test the connection with the Database Server.
Restore Defaults	Click Restore Defaults to default settings.
Apply	Click Apply to configure and save the settings

XV. Bug Tracking Tool: Configure Bug Tracking Tool Accounts to Post Bugs Directly from TestingWhiz.



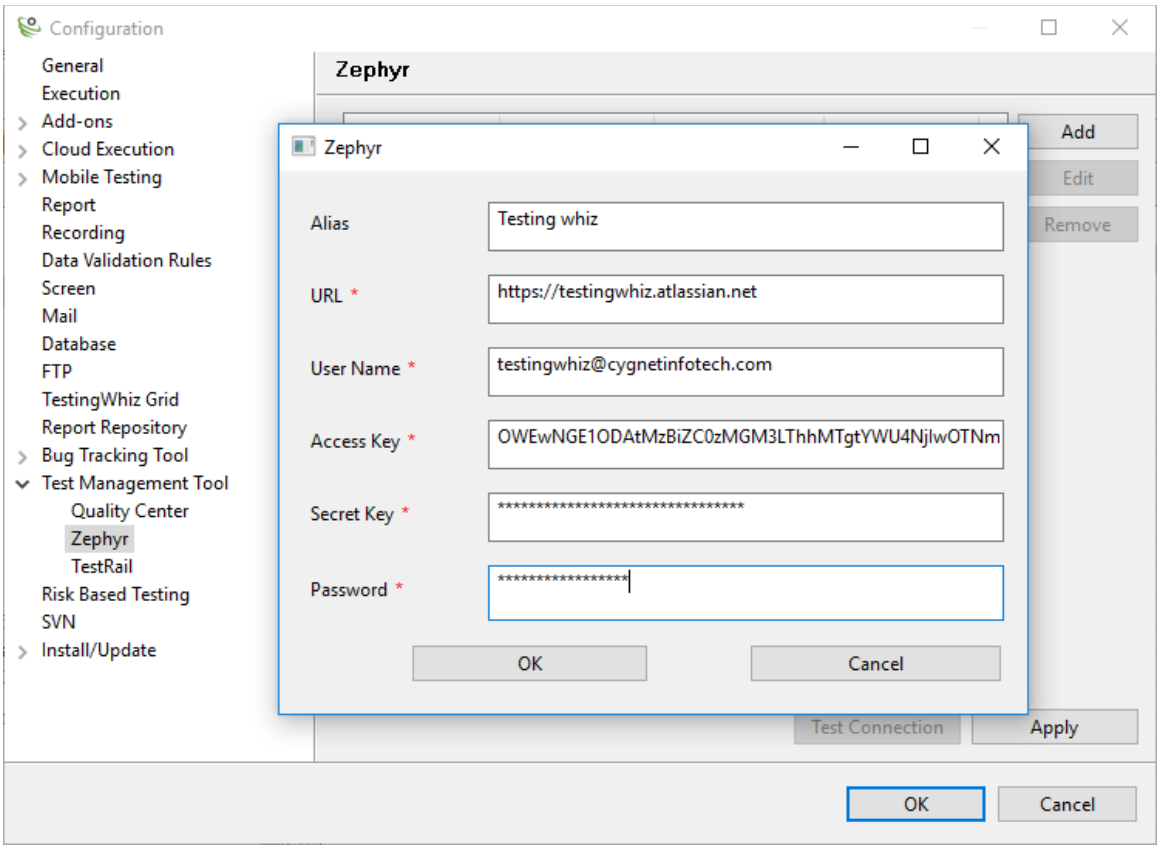
Select Bug Tracking Tool	Select the Bug Tracking Tool – Jira, Mantis or FogBugz.
Alias	Click Add and enter Alias of the Bug Tracking Tool.
URL	Enter URL of the Bug Tracking Tool.
Username & Password	Enter Username & Password to establish connection with the Bug Tracking Tool.
Test Connection	Click Test Connection to test reporting with the selected Bug Tracking Tool.
Apply	Click Apply to configure and save the settings.

XVI. Test Management Tool: Configure Test Management Tool (Quality Center) with TestingWhiz.



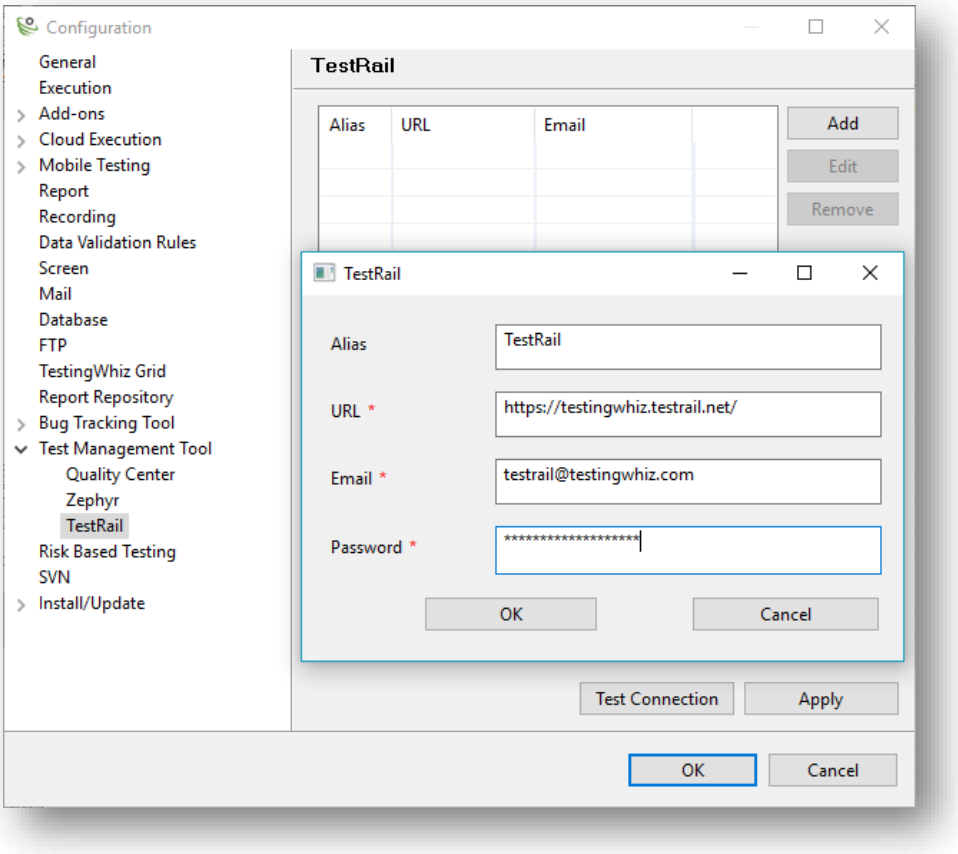
Alias	Click Add and enter the Alias name of the Test Management Tool.
Quality Centre URL	Enter URL of the HP QC.
Username & Password	Enter Username & Password to authorize the Test Management Tool.
Test Connection	Click Test Connection to test the connection with the Test Management Tool.
Apply	Click Apply to configure and save the settings.

Configure Test Management Tool (Zephyr) with TestingWhiz.



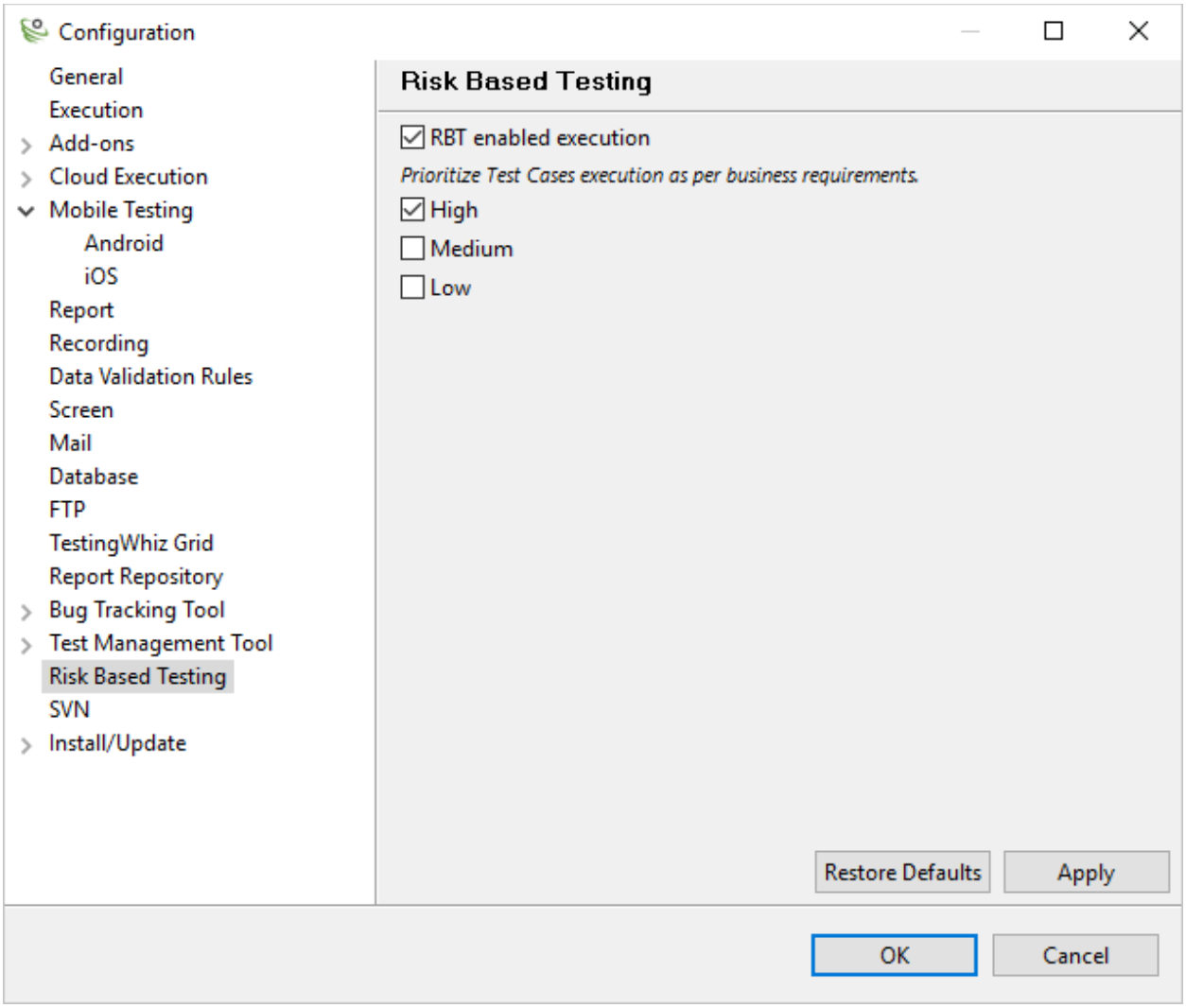
Alias	Click Add and enter the Alias name of the Test Management Tool.
Zephyr URL	Enter URL of the Zephyr for Jira .
User name & Password	Enter User name & Password to authorize the Test Management Tool.
Access Key	Enter the access key (You can found them by logging in to your JIRA cloud instance, browse to *Tests (top menu bar) > Importer > API Keys*.)
Secret Key	Enter the secret key (You can found them by logging in to your JIRA cloud instance, browse to *Tests (top menu bar) > Importer > API Keys*.)
Apply	Click Apply to configure and save the settings.
Test Connection	Click Test Connection to test the connection with the Test Management Tool.

Configure Test Management Tool (Test Rail) with TestingWhiz.



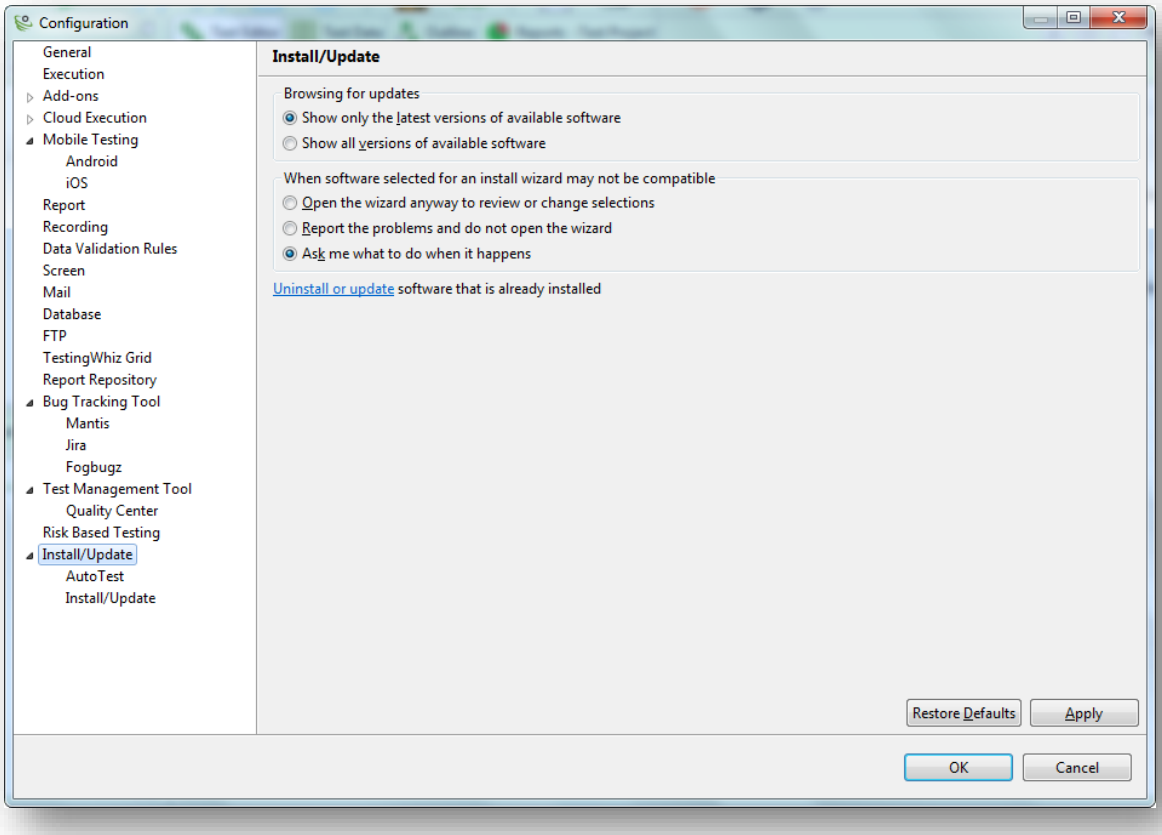
Alias	Click Add and enter the Alias name of the Test Management Tool.
Test Rail URL	Enter URL for TestRail.
Email & Password	Enter Email & Password to authorize the Test Management Tool.
Test Connection	Click Test Connection to test the connection with the Test Management Tool.
Apply	Click Apply to configure and save the settings.

XVII. Risk Based Testing: Configure settings for executing Risk Based Testing.



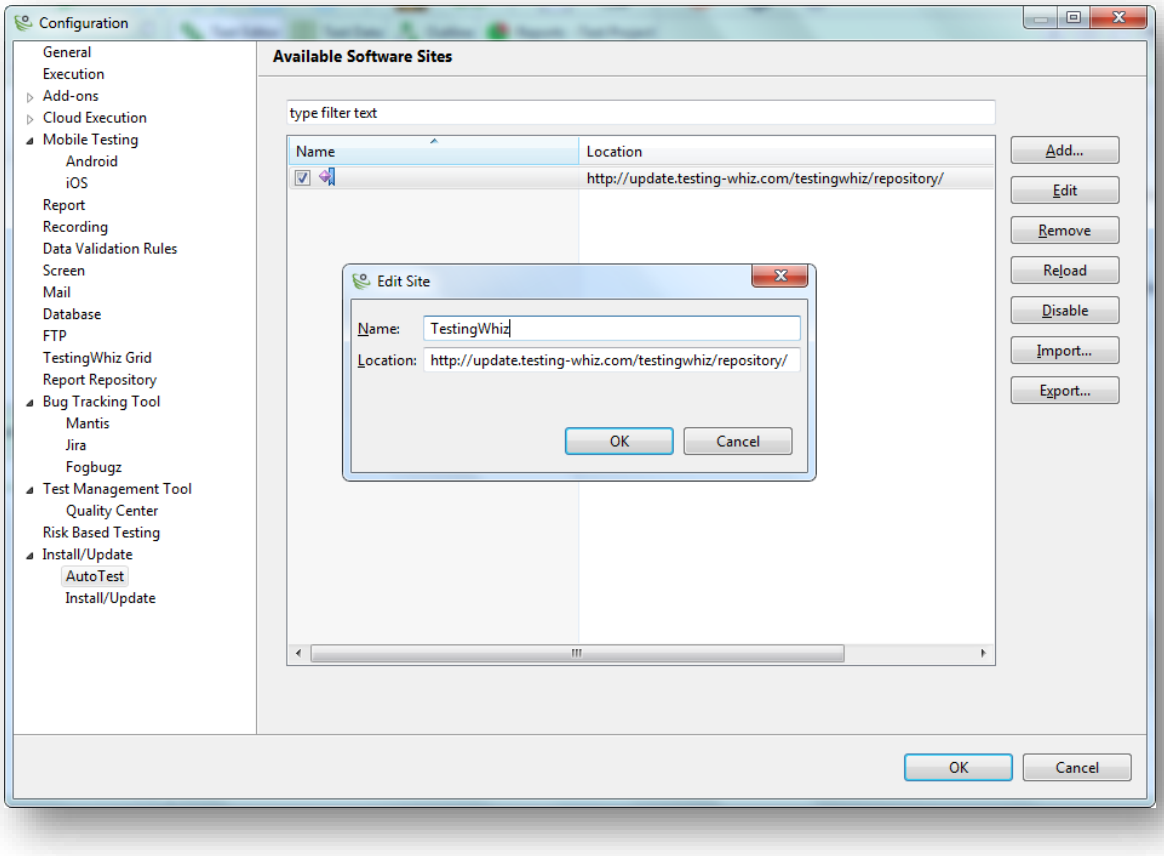
RBT enabled execution	Tick this option to enable Risk Based Testing.
Prioritize Test Cases	Select options among High, Medium and Low to prioritize Test Case execution with Risk Based Testing.

XVIII. Install/Update: Configure Settings Related to Installation & Updates of TestingWhiz.



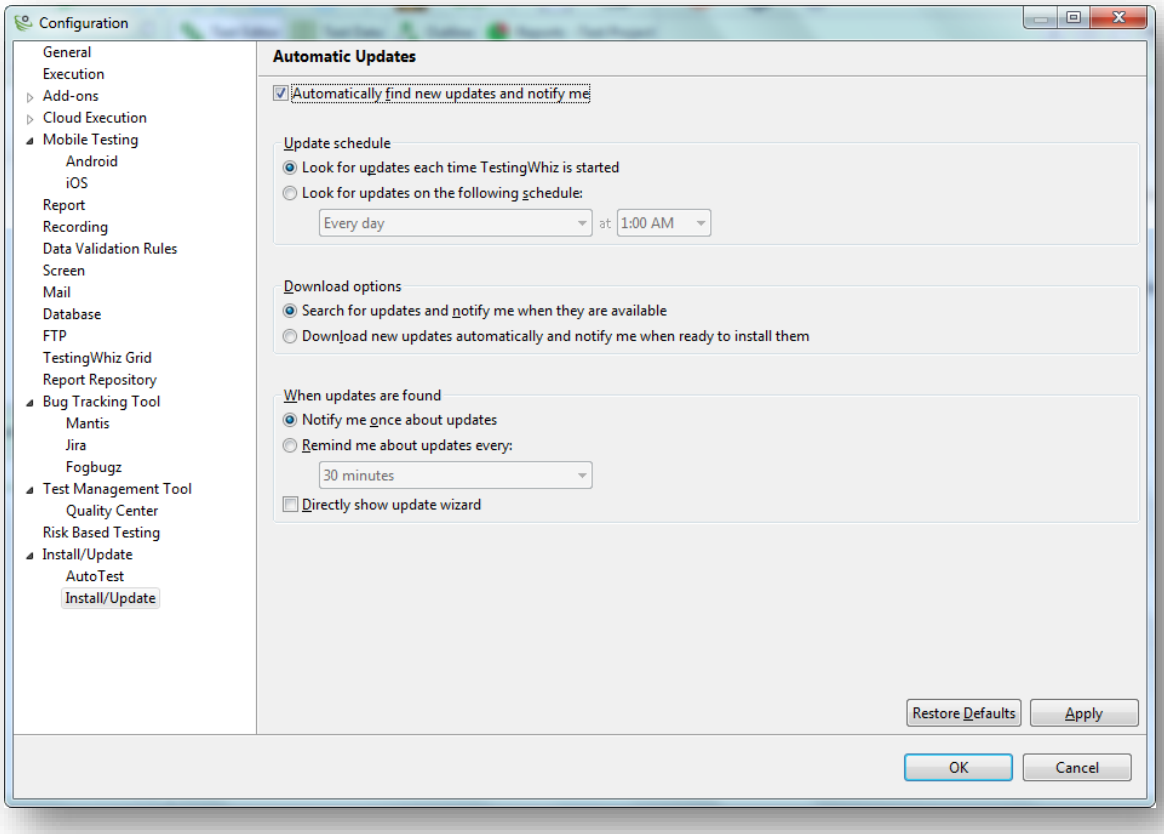
Browsing for Updates	Select option to show only latest versions of the available software OR show all versions of the available software while browsing for updates.
Software Compatibility for an Install Wizard	Select option to either: Open Wizard to review/change selections OR Report only problems OR Ask me what to do when the software is not compatible for the install wizard.

A. AutoTest: Available Software Sites



Add	Enter Name of the available software and location of the file from Local or Archive folder.
Edit	Click Edit to change the name and location of the already added software site.
Remove	Click Remove to remove existing software site.
Reload	Click Reload to reload the software site in TestingWhiz.
Enable/Disable	Click Enable/Disable buttons to check or uncheck existing software site.
Import	Click Import to integrate software sites in TestingWhiz.
Export	Click Export to save the list of software sites.

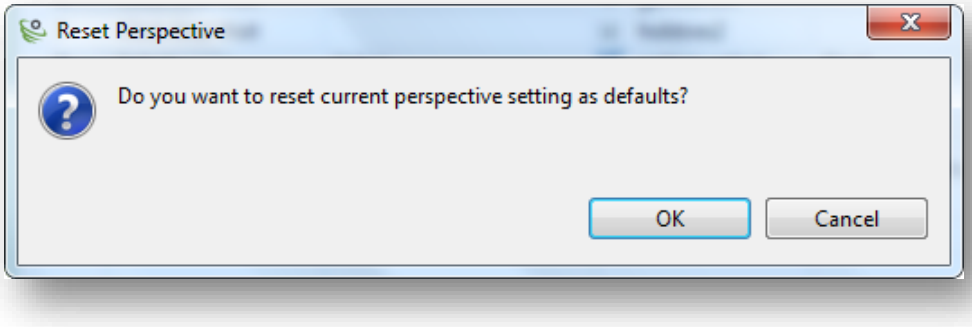
B. Install/Update: Automatic Updates



Find New Updates	Tick this option to automatically search for new updates.
Update Schedule	Select option to either look for updates each time TestingWhiz is started OR specify the day and time to look for updates.
Download Options	Select option to either search and notify for new updates OR download updates automatically.
When updates are found	Select option to either notify when updates are found OR set timings to remind for updates automatically.
Restore Defaults	Click Restore Defaults to reverse to default settings.
Apply	Click Apply to configure and save settings.

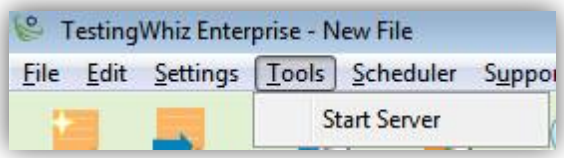
2.2.3.2 Reset Perspective

Click Reset Perspective to restore the default settings.



2.2.4 Tools

Use Tools to perform the following functions:

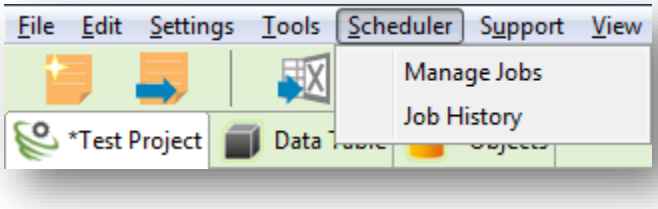


2.2.4.1 Start Server

Click Server Start to execute on Jenkins.

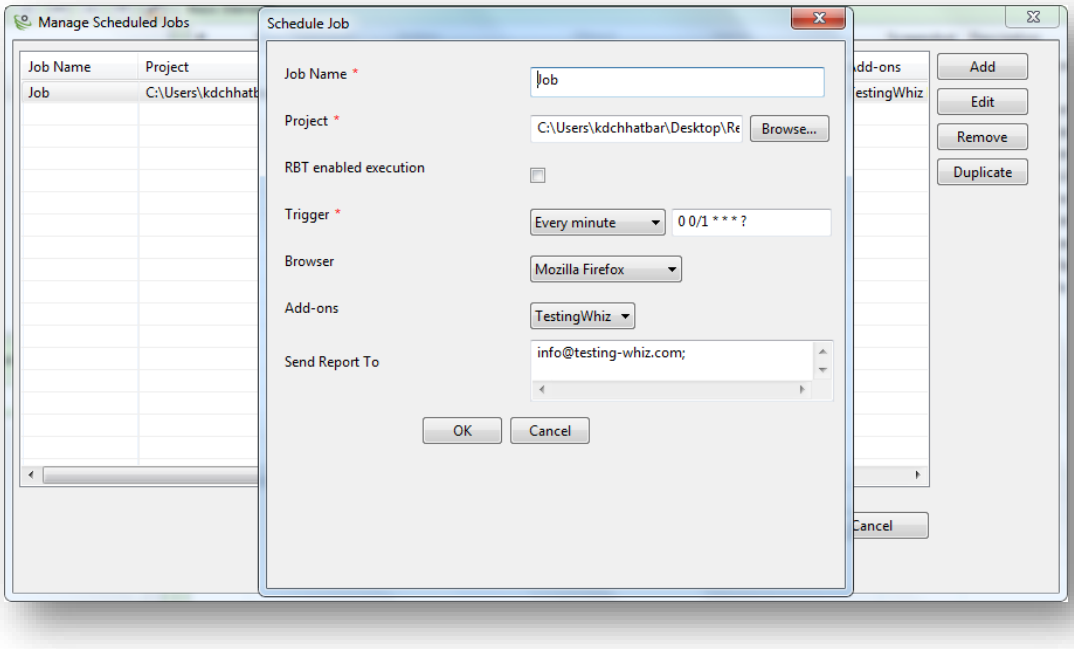
[**Note:** *To avail Jenkins integration functionality on your TestingWhiz, email at sales@testing-whiz.com.*]

2.2.5 Scheduler



2.2.5.1 Manage Jobs

Click Scheduler > Manage Jobs to schedule and manage jobs in TestingWhiz.



Add: Click Add to Schedule a Job(s) and enter the details.

Job Name	Enter the Name of the job to be scheduled.
Project	Enter the path/location of the Test Script(s) to be scheduled.
RBT enabled execution	Tick this option to perform Risk-based Testing of the selected Test Script.
Trigger	Select the time to Trigger a job schedule. [**Note:** *User can customize Trigger Time based on Minute, Hour, Day, Week and Month.*]
Browser	Select the Browser to run the Test Script(s).

Add-ons	Select the Add-ons of the respective Browser.
Send Report To	Enter the email addresses to Send Report to individuals. [**Note:** *This function will work only after a user has set Mail Preferences.*]
Active	When the checkbox is checked, the scheduled job will be active and executed. Otherwise, it will be kept as a record.

Edit: Click Edit to edit the details of the scheduled job(s).

Remove: Click Remove to remove specific job(s) from the list.

Duplicate: Click Duplicate to copy a scheduled job.

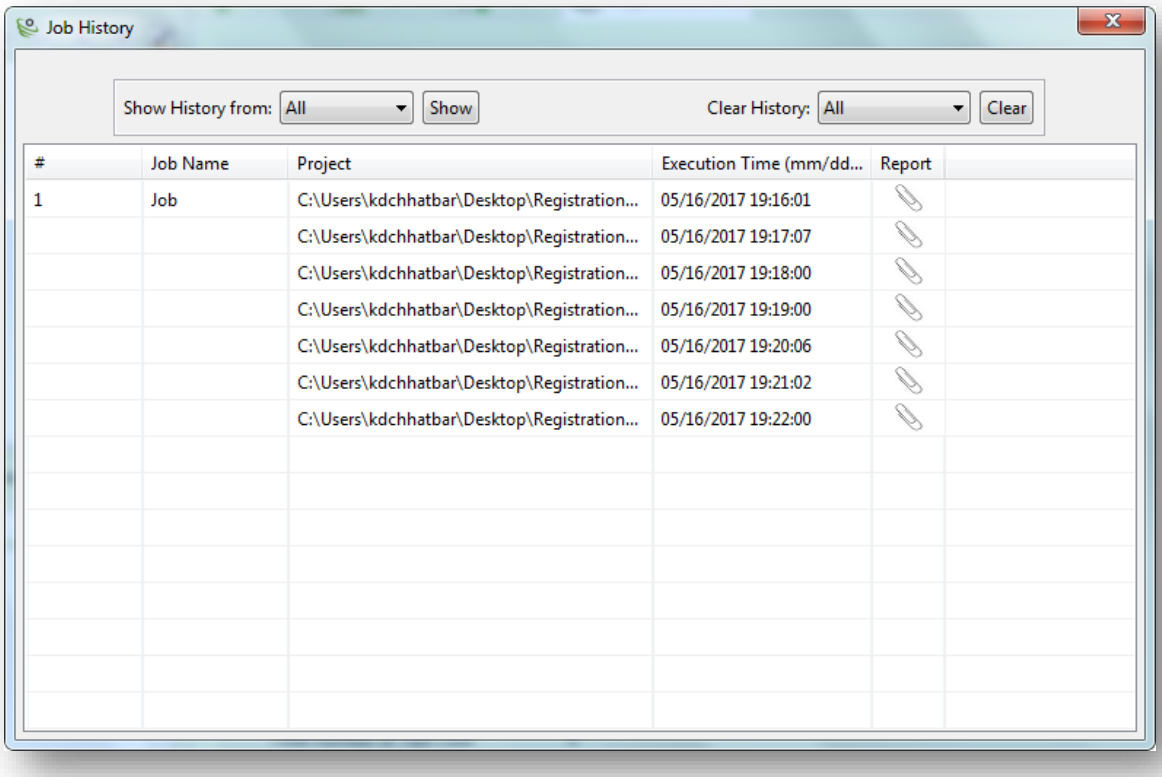
Save: Click Save to save the details of the scheduled job.

Trend Analysis: It facilitates to view the Trend of scheduled jobs.

2.2.5.2 Job History

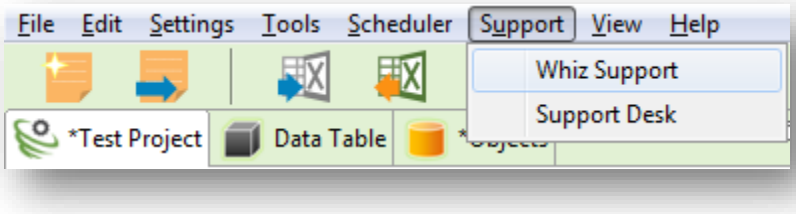
Click Job History to view a history of the scheduled + executed jobs.

Select from the drop-down: All, Today, Last 7 days, Last 15 days, Last 30 days, Last 90 days. Similarly, a user can clear history of the scheduled + executed jobs by selecting the period from the drop-down list.



2.2.6 Support

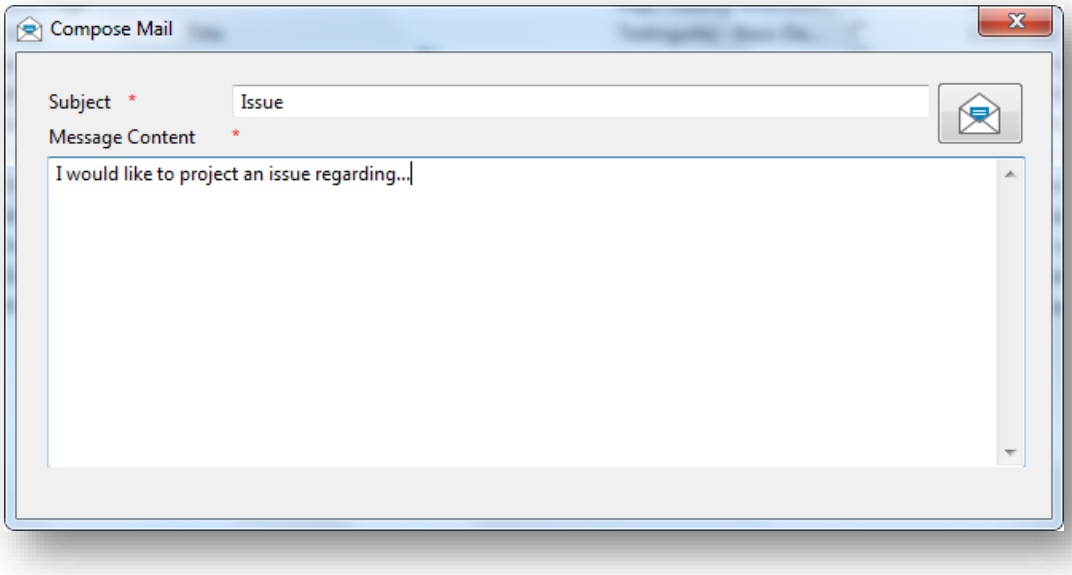
Click Support Tab to get personalized support from TestingWhiz Support Team related to the tool usage and queries.



2.2.6.1 Whiz Support

Click Whiz Support to directly email issues, queries and concerns related to TestingWhiz to the Support Team. Enter the Subject and Message Content. Once finished, click  to send the e-mail.

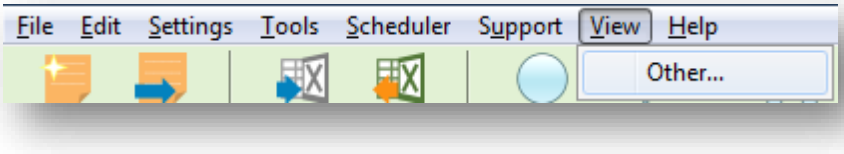
[**Note:** *Mailings through Whiz Support will work only after the Mail server connection has been established.*]



2.2.6.2 Support Desk

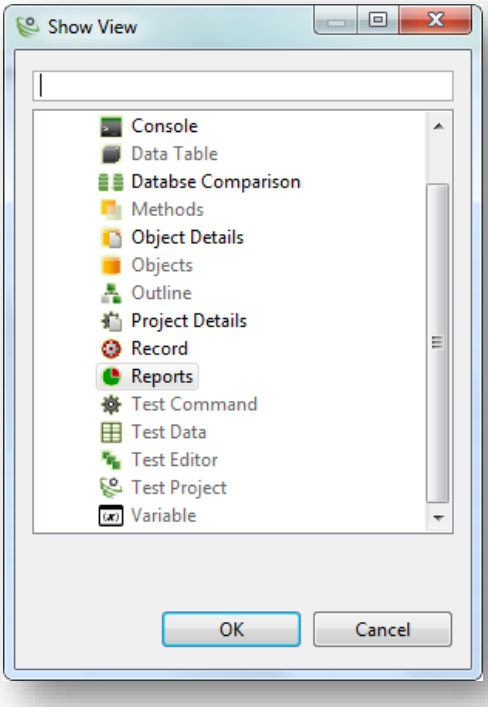
Clicking on Support Desk will redirect a user to TestingWhiz Support Web Page. From this web page, a user can get answers from the community forums, submit requests to the Support Desk and check updates & announcements.

2.2.7 View



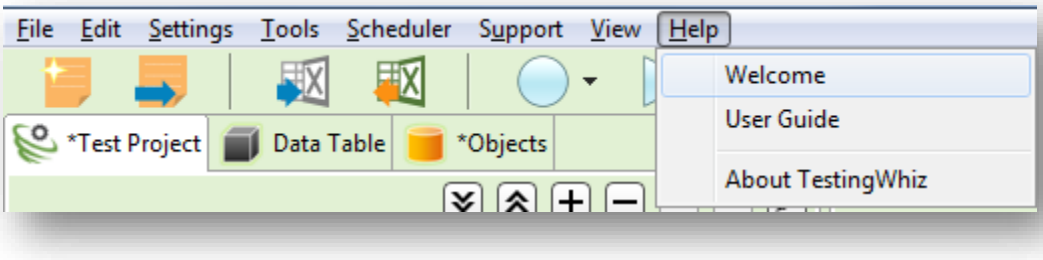
2.2.7.1 Others

Use the Quick Access Icons to get instant access to Menu items, Recording function and Console.



2.2.8 Help

Use Help to access following functions:



2.2.8.1 Welcome

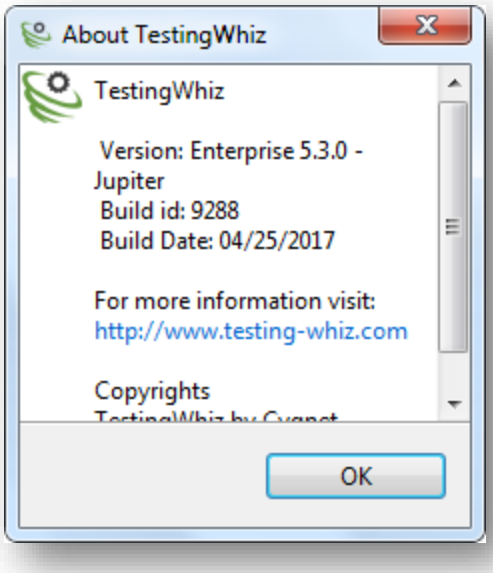
Click Welcome to go to TestingWhiz Welcome Page.

2.2.8.2 User Guide

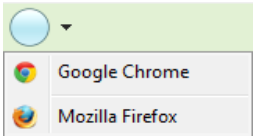
Click User Guide to open and view the TestingWhiz User Manual in PDF Format.

2.2.8.3 About TestingWhiz

Click on About TestingWhiz to avail information related to TestingWhiz application version, build date, etc.

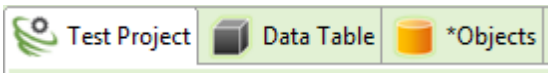


2.3 Tool Bar

Features	Description
	To create a New Test Project in TestingWhiz. OR Press [CNTR+N] to create a New Test Project
	To open an Existing Project. OR Press [CNTR+O] to create a New Test Project
	To import the Script(s) from the Excel file to TestingWhiz. [**Note:** *Available only after user has created script(s) in Excel.*]
	To export recorded or automated Test Script(s) to Excel.
	To start recording Test Case(s). [**Note:** *The color will change to Red*  *, which indicates that the recording is in progress.]* Click  to start recording test case(s) in external browser (Google Chrome). Click  to stop recording of Test Case(s).
	To execute a Test Case. Click down arrow to select the browser or platform for execution. [**Note:** *If there is no Test Script present in the Test Project, this button will remain disabled.*]
	To go to the Next Step during Test Execution. [**Note:** *This button will be enabled only if a test script execution is paused / is at Toggle BreakPoint.*]
	To Pause the Test Execution process. [**Note:** *This button will be active only when a test is being executed.*]

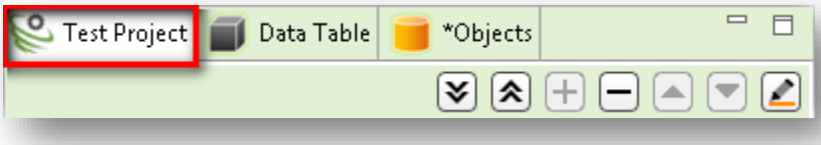
	
	To Stop the Test Execution process. [**Note:** *This button will be active only when a test is being executed.*]
	To record on-screen objects, we can use the visual recorder. [**Note:** The color will change to Red  which indicates that the recording is in progress.] Click on  to stop the visual recorder.
	To view Image Comparison report. [**Note:** *This feature will be enabled only after completion of Image Comparison.*]
	To view Database Comparison report. [**Note:** *This feature will be enabled only after completion of Database Comparison.*]
	To mail Test Report(s). [**Note**: *A window to enter the Email Address(es), Subject and the Message Content will appear.*] [**Note:** *This feature will function only if a user has set Email preferences in the Configuration section.]*
	To generate test data on based on standard rules or own Java regular expression [**Note:** *A window will be shown to enter the Name of the Data Table, the number of Data Combinations, Field Name, Test Data type*
	To log issues (if any) in the Bug Tracking Tool during test execution. [**Note:** *A window to select the Bug Tracking Tool will appear.*] [**Note:** *This feature will function only if a user has set Bug Tracking Tool credentials in the Configuration section.*]
	To integrate Test Projects, Defects, etc. with the Test Management Tool. [**Note:** *This feature will function only if you have set Test Management Tool preferences in the Configuration section.*]
	To Log Off TestingWhiz [***Note:*** *This feature is available only for Floating License Users.*]

2.4 Menu Tabs



2.4.1 Test Project

Use Test Project tab to create, delete and move Test Suite(s) and Test Case(s).

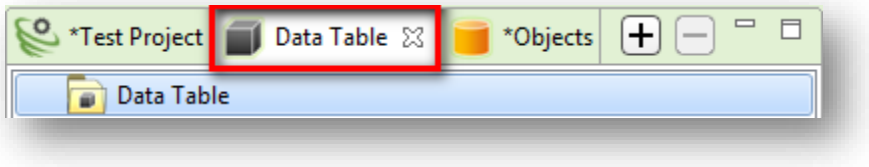


Function	Description
	To expand all the Test cases in one particular Test Suite.
	To hide all the Test cases in one particular Test Suite.
	To add new Test Suite(s) and Test Case(s).
	To delete existing Test Suite(s) or Test Case(s).
	To move up a particular Test Suite(s) or Test Case(s).
	To move down a particular Test Suite(s) or Test Case(s).
	To highlight a particular Test Suite or Test Case.

Refer Section – **Steps to Create New Project** to learn more.

2.4.2 Data Table

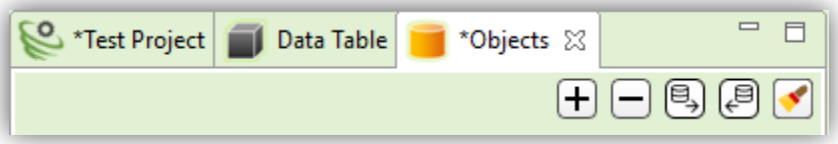
Use Data Table to add/import multiple data sets and input values for executing Test Case(s).



Function	Description
	To add new Data Table for inserting Test Data.
	To delete a particular Data Table containing Test Data.

2.4.3 Objects

Objects displays the repository of Test Objects.



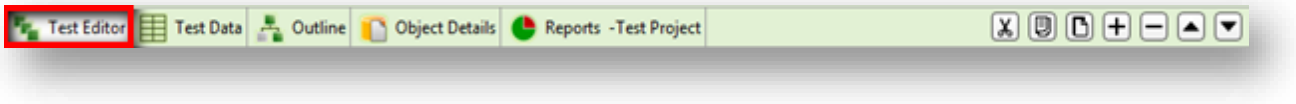
Features	Description
	To add new Object Properties
	To delete a particular Object Properties
	Export all objects to a Database
	Export objects from a Database
	To clean up existing Test Objects.

Refer Section – **Object Repository** to learn more.

2.5 Test Editor Tabs

2.5.1 Test Editor

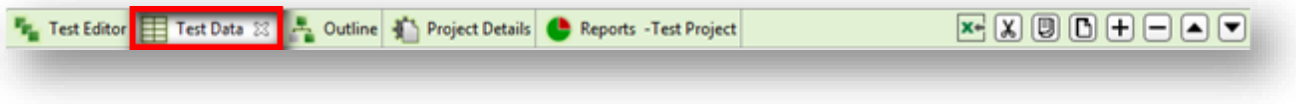
Use Test Editor to add and manage Test Steps.



Function	Description
	To cut particular Test Step(s) created under a specific Test Case.
	To copy particular Test Step(s) created under a specific Test Case.
	To paste already cut/copied Test Step(s).
	To add new Test Step(s) within a particular Test Case.
	To delete existing Test Step(s) within a particular Test Case.
	To move up a particular Test Step.
	To move down a particular Test Step.

2.5.2 Test Data

Use Test Data to render values of Data Table.



Function	Description
	To import data from Excel file to Data Table
	To cut particular Data Table Value/Test Step.
	To copy particular Data Table Value/Test Step.
	To paste already cut/copied Data Table Value/Test Step.
	To add new Data Table Value/Test Step.
	To delete a particular Data Table Value/Test Step.

	To move up a particular Data Table Value/Test Step.
	To move down a particular Data Table Value/Test Step.

2.5.2.1 Process to Import Data from Excel:

Step 1: Click to Import data from Excel file to Data Table.

Step 2: Select Excel File to be imported

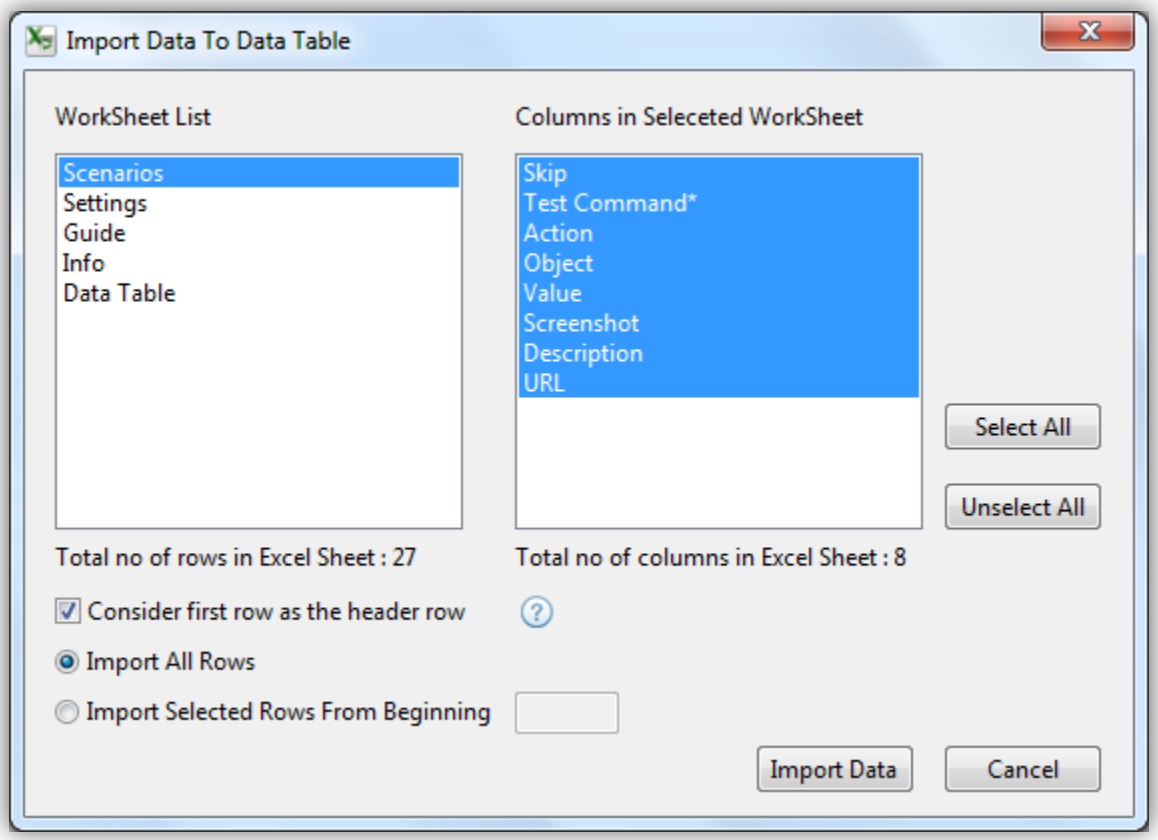
Step 3: Select any one Worksheet from Worksheet list of Excel File.

Step 4: Select all columns or multiple columns or single columns from the Column List of Selected worksheet from List.

Step 5: Tick option to Consider first row as the header row

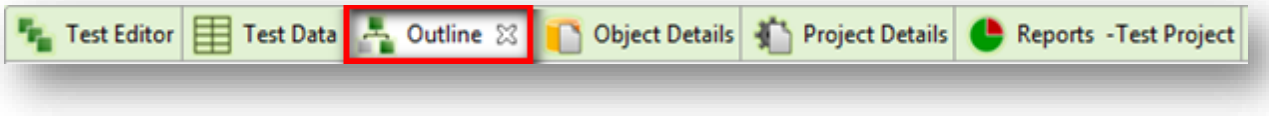
Step 6: Choose option between 'Import All Rows' and 'Import Selected Rows From Beginning'

Step 7: Click "Import Data" button as per the user selection all records are imported into selected placeholder.



2.5.3 Outline

Use Outline to view the Test Case in Data Flow Diagram.



Refer Section – **Outline View/Visual Presentation View** to learn more

2.5.4 Object Details

Use Object Details to add, edit and manage Object Properties of Test Objects

Refer Section – **Object Repository** to learn more

2.5.4.1 Process to Manage Object Properties

Step 1: Add/Edit Alias of the Test Object

Step 2: Add/Edit ID of Test Object

Step 3: Add/Edit Name of Test Object

Step 4: Add/Edit Tag Type of Test Object

Step 5: Add/Edit Inner HTML

Step 6: Enter Value of Test Object

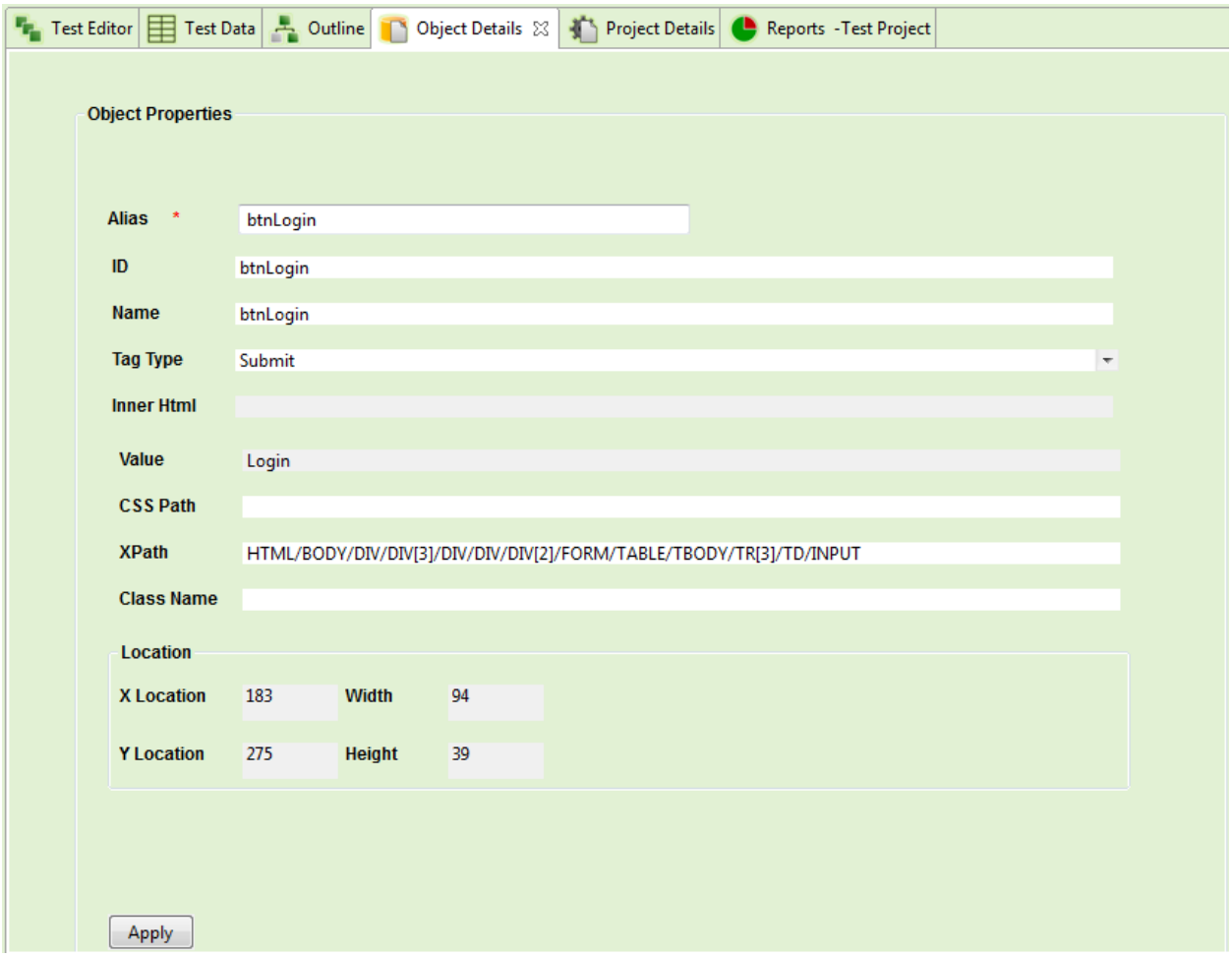
Step 7: Enter CSS Path of Test Object

Step 8: Enter XPath of Test Object

Step 9: Add/Edit Class Name of Test Object

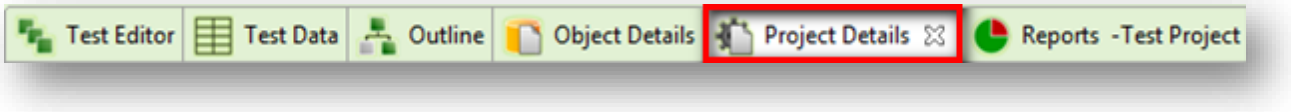
Step 10: Manage X & Y Location of Test Object

Step 11: Click Apply to save changes



2.5.5 Project Details

Use Project Details Tab to Add, Manage and View Test Project Details



[Note: *Project Details Tab will be displayed only when a user clicks on the Test Project Folder as shown above.*]

2.5.5.1 Process to Add & Manage Test Project Details

Step 1: Enter Project Name

Step 2: Enter Project URL

Step 3: Enter Number of Test Execution Cycle

Step 4: Enter Environment Details

Step 5: Enter Project Description

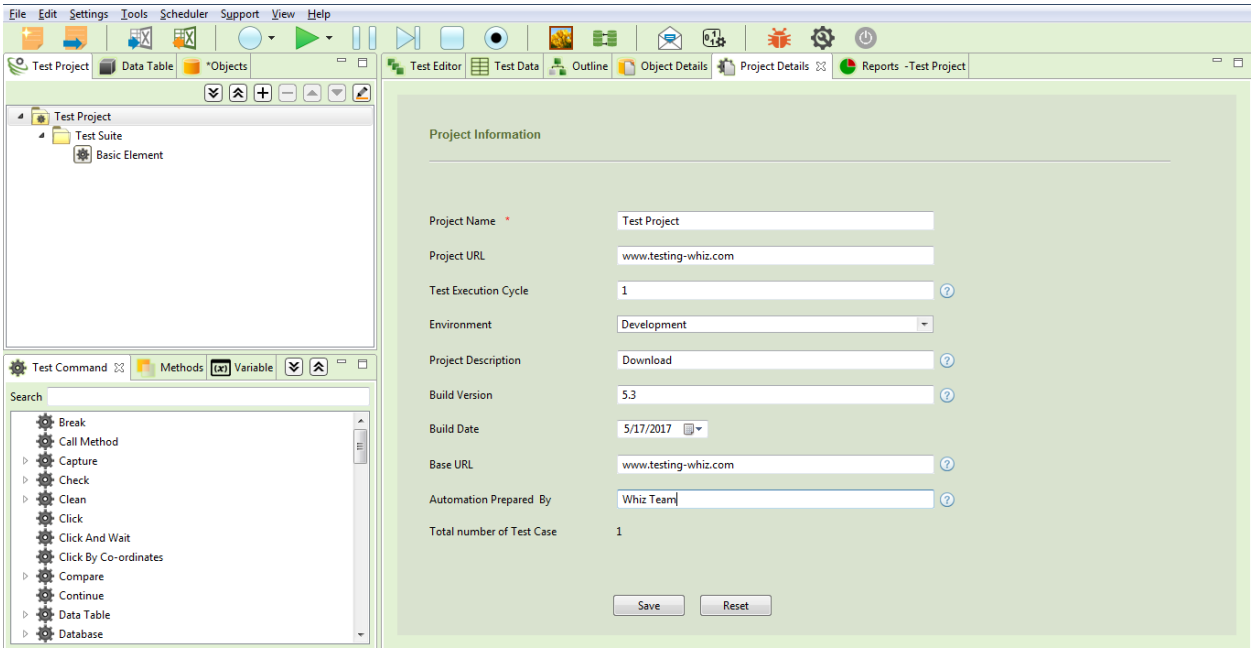
Step 6: Enter Build Version

Step 7: Select Build Date

Step 8: Enter Base URL

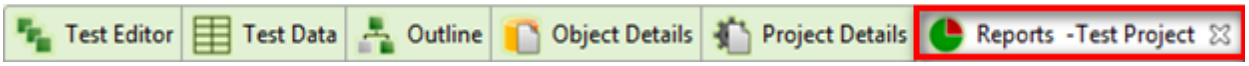
Step 9: Enter Details of Automation Prepared By

Step 10: Click Save to save the Project Details

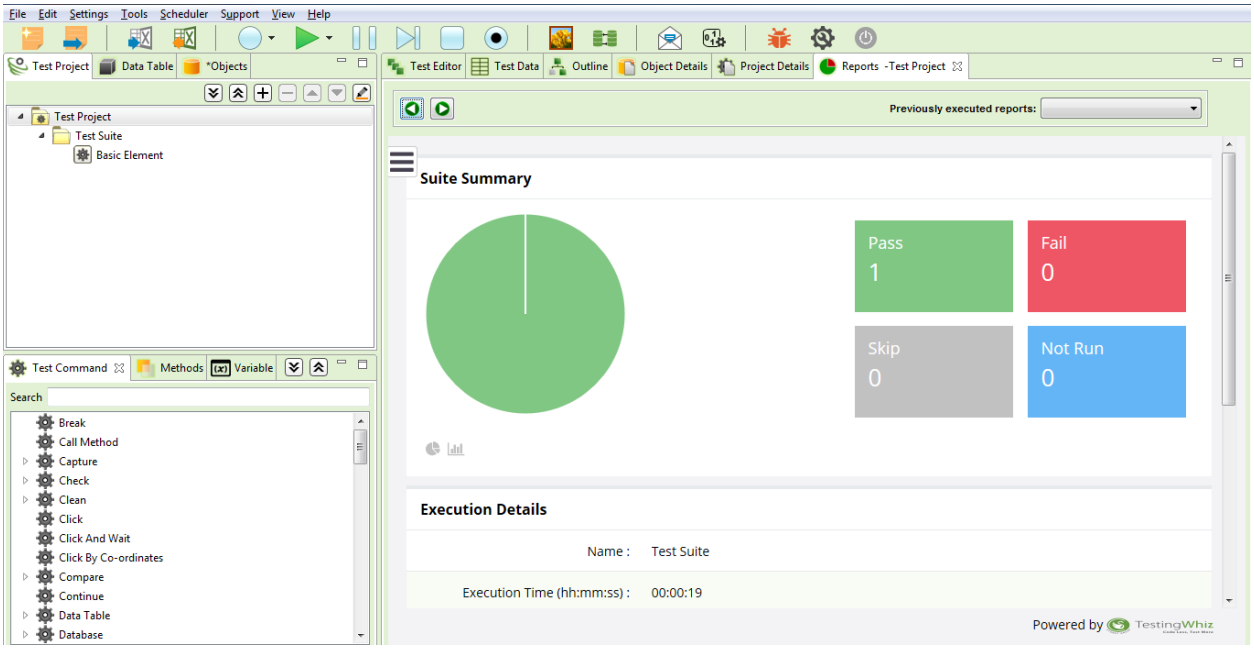


2.5.6 Reports

Use Reports tab to view the reports; i.e., percentage of passed or failed Test Case(s).



Use Backward  and Forward  buttons to View alternate Reports.

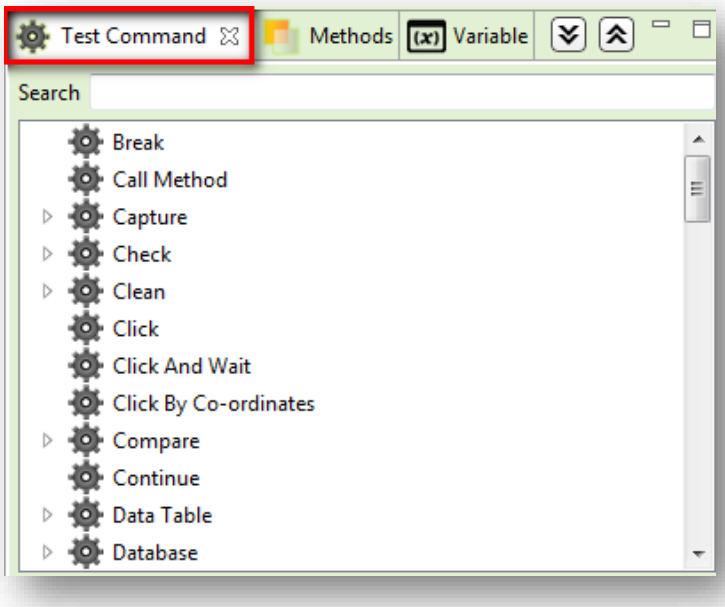


[**Note:** *Reports Tab will be enabled only after a Test Case has been executed; if not, this function will not be visible in the Test Editor Tab.*]

2.6 Test Command, Methods and Variable Tab

2.6.1 Test Command

Use Test Command Tab to view the list of available commands defined in the system.

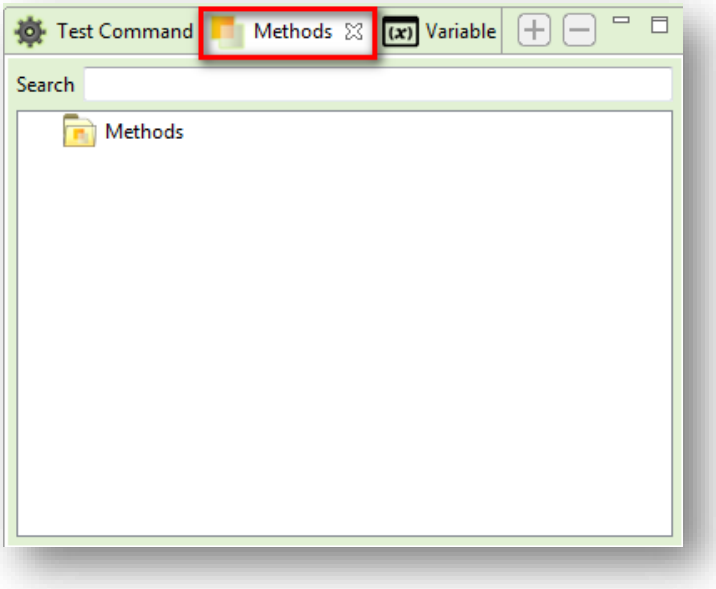


Function	Description
	To expand all the functions of a selected commands.
	To collapse all the expanded functions of a selected commands.

For more details on Test Command, kindly refer chapter LIST OF TEST COMMANDS & CORRESPONDING ACTIONS.

2.6.2 Methods

Use Methods Tab to add and apply methods of choice in the test execution process:

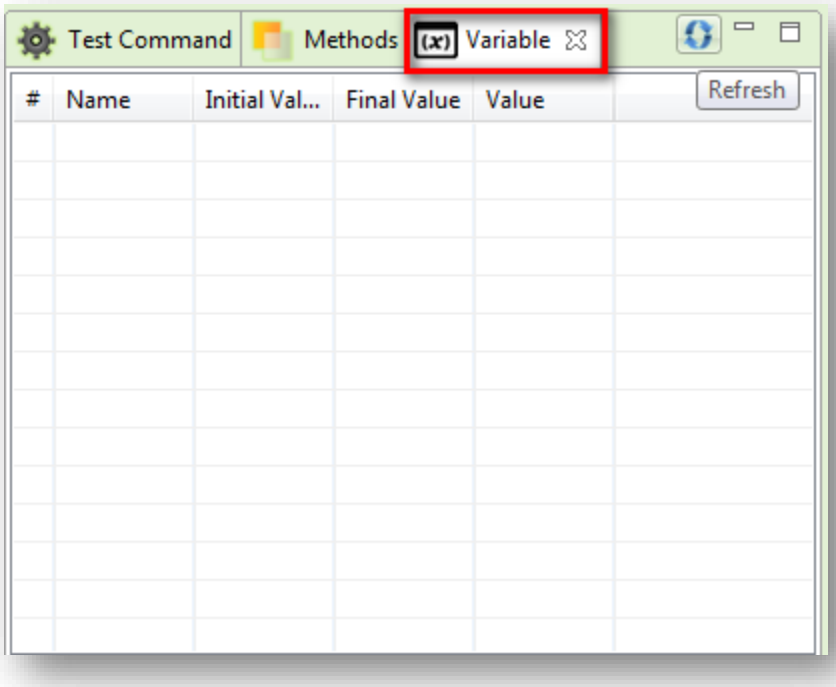


Function	Description
	To create new Method.
	To delete a particular Method.

Refer Section - **Methods** for more details on Methods.

2.6.3 Variable

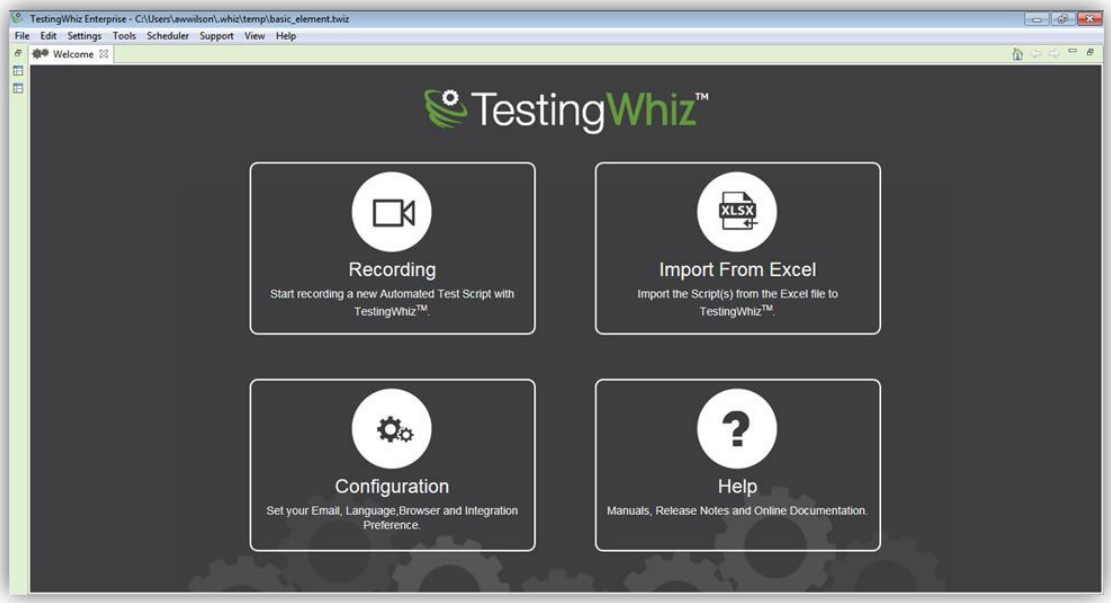
Use Variable Tab to view the value of variables used to Perform Command with type - Local & Global Variable.



Refer Section – **Perform** Command in Test Command Section to view more.

3 LEARNING TO CREATE & MANAGE TEST PROJECTS, TEST CASES & TEST SCRIPTS

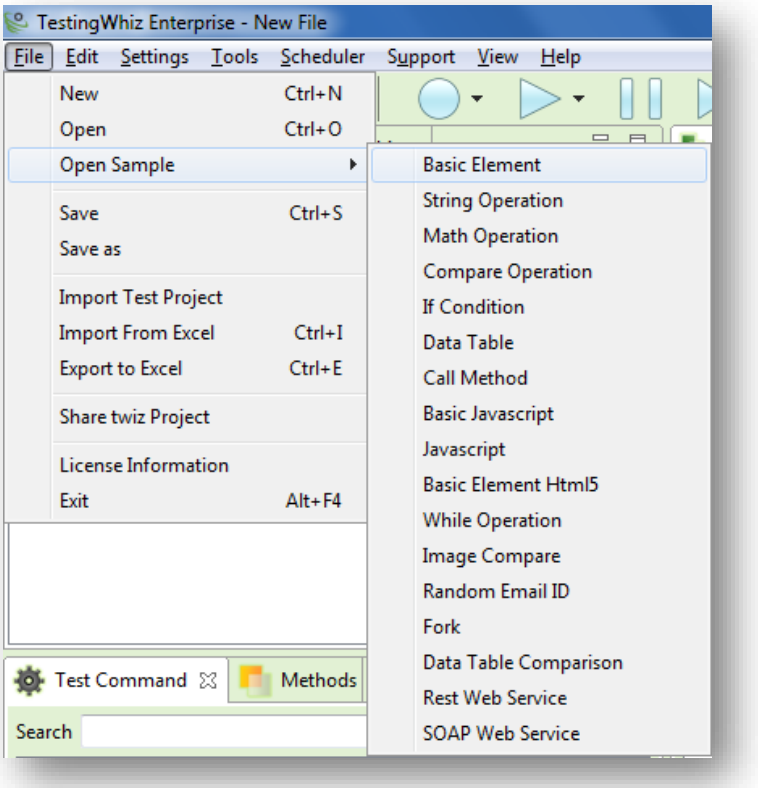
TestingWhiz works on the principle of codeless testing. It allows users to build test scripts without presupposing programming or technical knowledge. The naming conventions of Test Commands and Actions are provided in a simple and functional language to help a user create and understand Automation Test Scripts easily.



3.1 Learn from a Sample Test Case

TestingWhiz includes several sample test cases to help a user get acquainted with the process.

To view a Sample Case, click on the **Open Sample** in **File** menu and select the type of the Test Case.



3.2 Process to Create & Manage Test Project, Test Suite & Test Suite

3.2.1 Steps to Create New Project

Here's the step-by-step process to create and manage Test Projects, Test Suites & Test Scripts in TestingWhiz

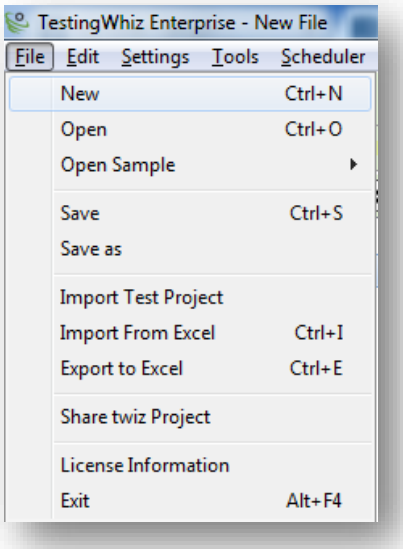
3.2.1.1 Add a New Project

Click 

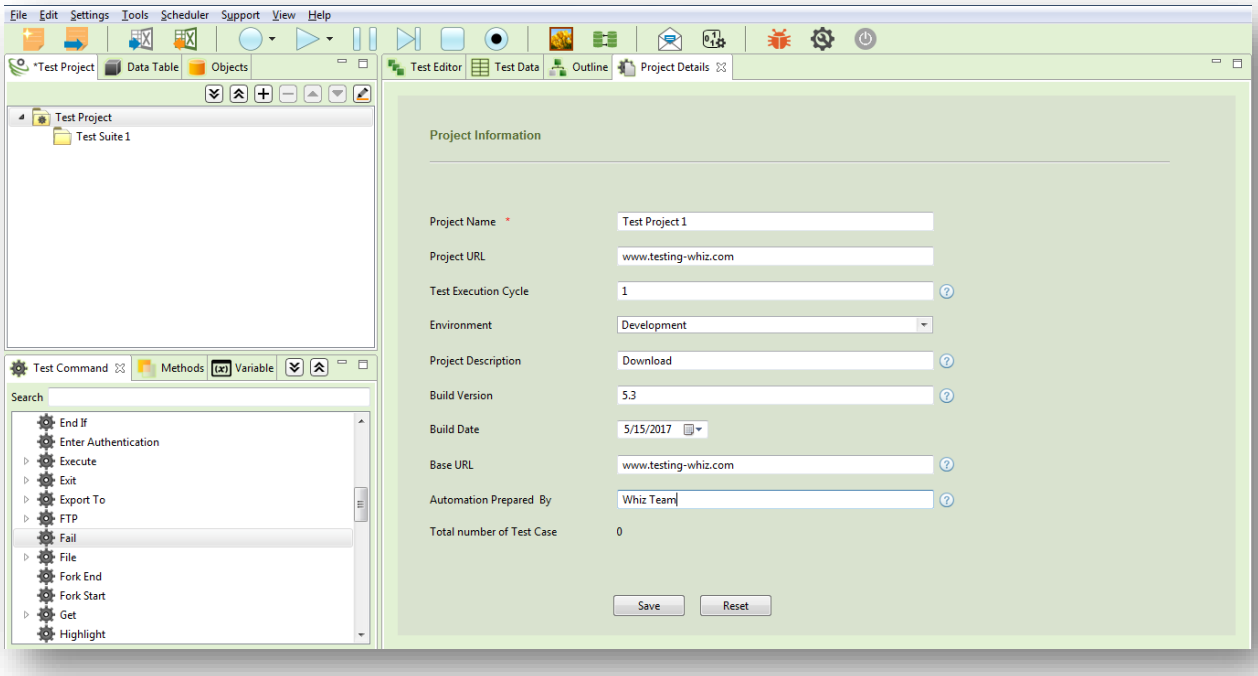
from the Tool Bar.

OR

Click on **File** and then click **New**.



Enter the Project Information in the form as shown below and click **Save**.



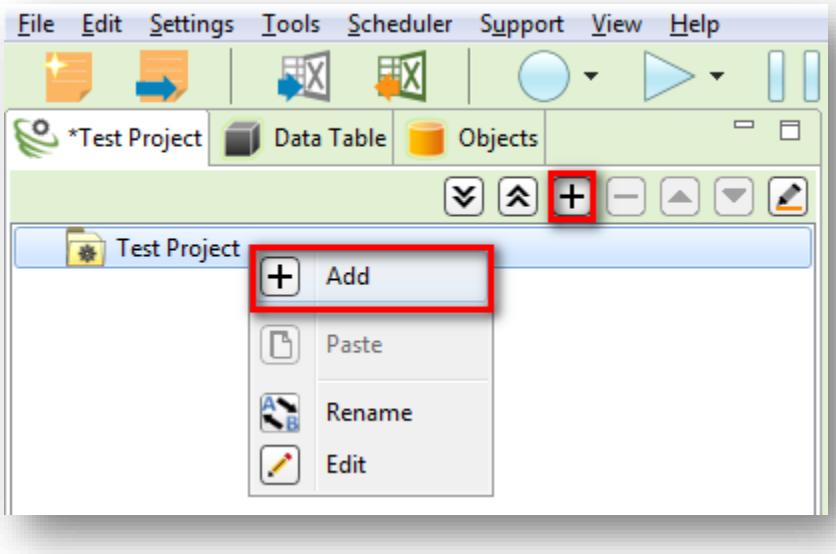
3.2.2 Steps to Add & Manage Test Suite under Test Project

3.2.2.1 Add a New Test Suite

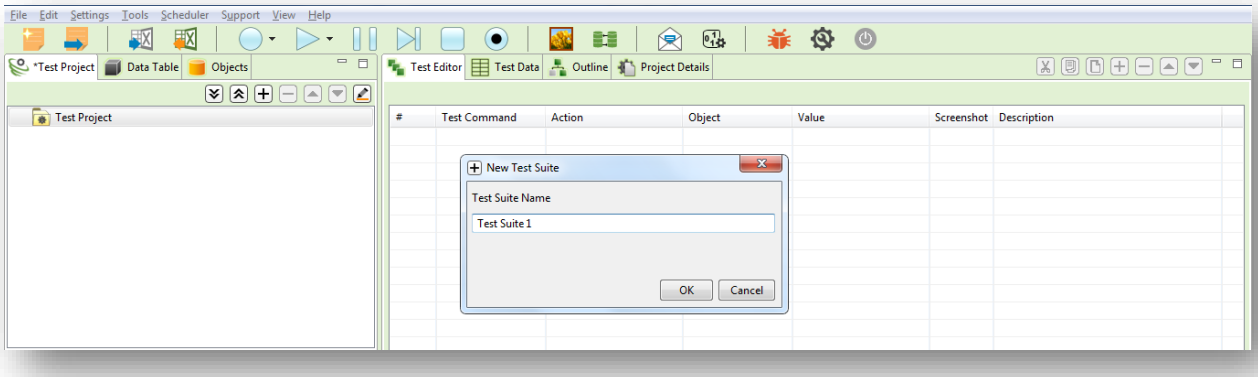
Select the Test Project as created in Step **3.2.1.1** and click  button to add a **New Test Suite** within the Test Project.

OR

Right Click on Test Project and click **Add**.



Give an appropriate name of the **Test Suite** in the pop-up as shown below and click **OK**.

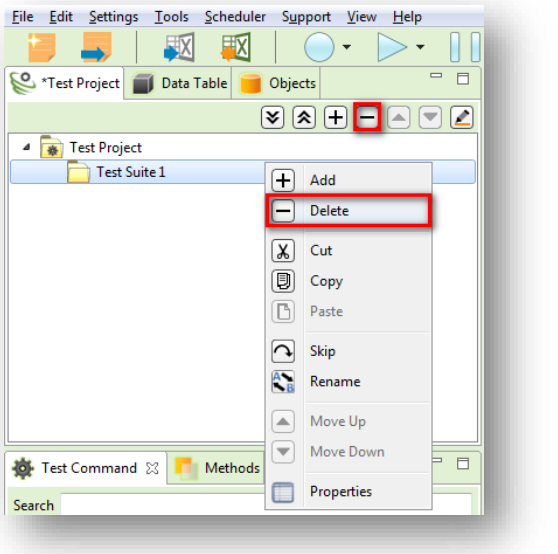


3.2.2.2 Delete a Test Suite

Select a Test Suite and click  to delete that Test Suite.

OR

Right Click on a particular Test Suite and click **Delete**.

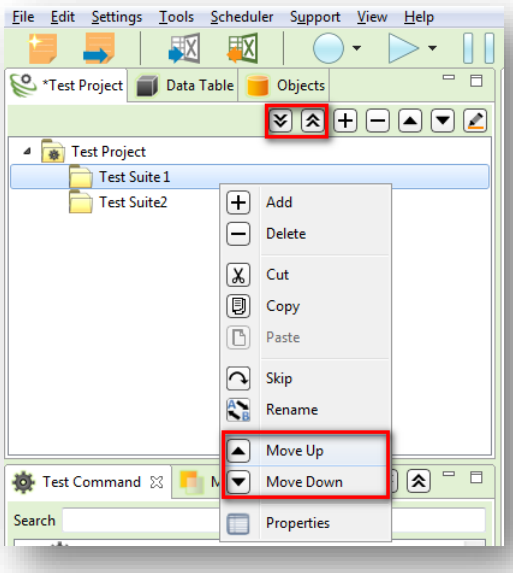


3.2.2.3 Move Up or Move Down a Test Suite

Select a Test Suite and click  or  respectively to Move Up or Move Down that particular Test Suite.

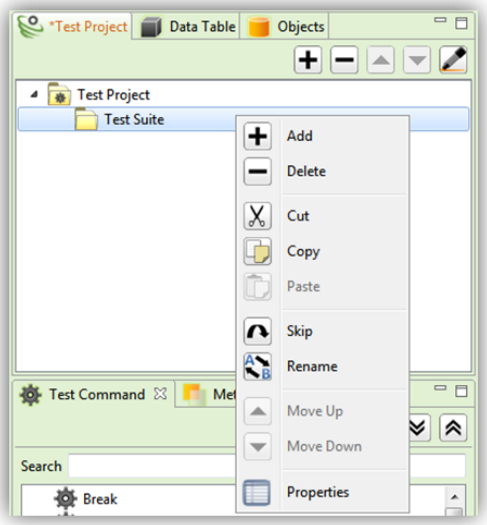
OR

Right Click on Test Suite and click **Move Up** or **Move Down**.



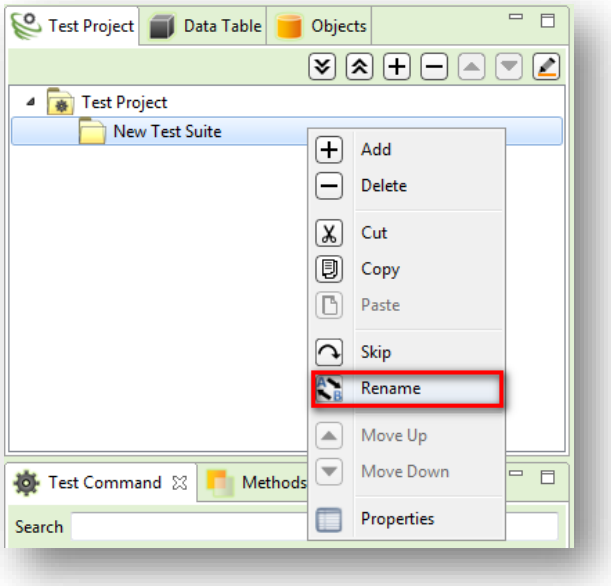
3.2.2.4 Cut, Copy, Skip a Test Suite

Select a Test Suite and right click on it to Cut, Copy, Skip or Rename that Test Suite.



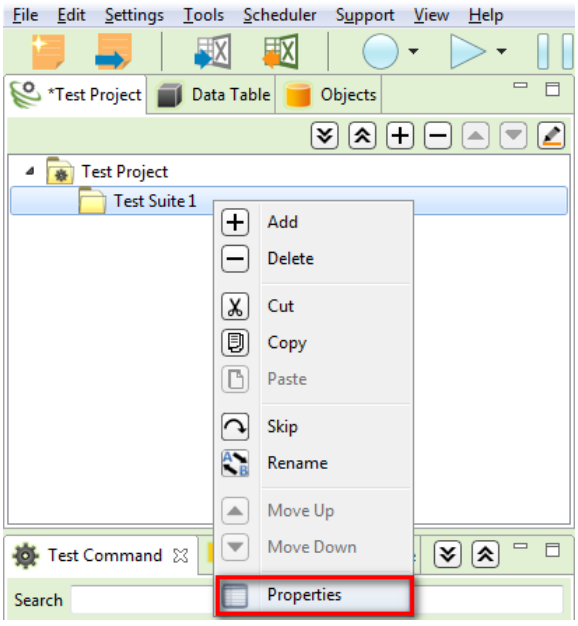
3.2.2.5 Rename a Test Suite

Right click on a Test Suite and click Rename. Alternately, Press F2 to rename the selected Test suite.

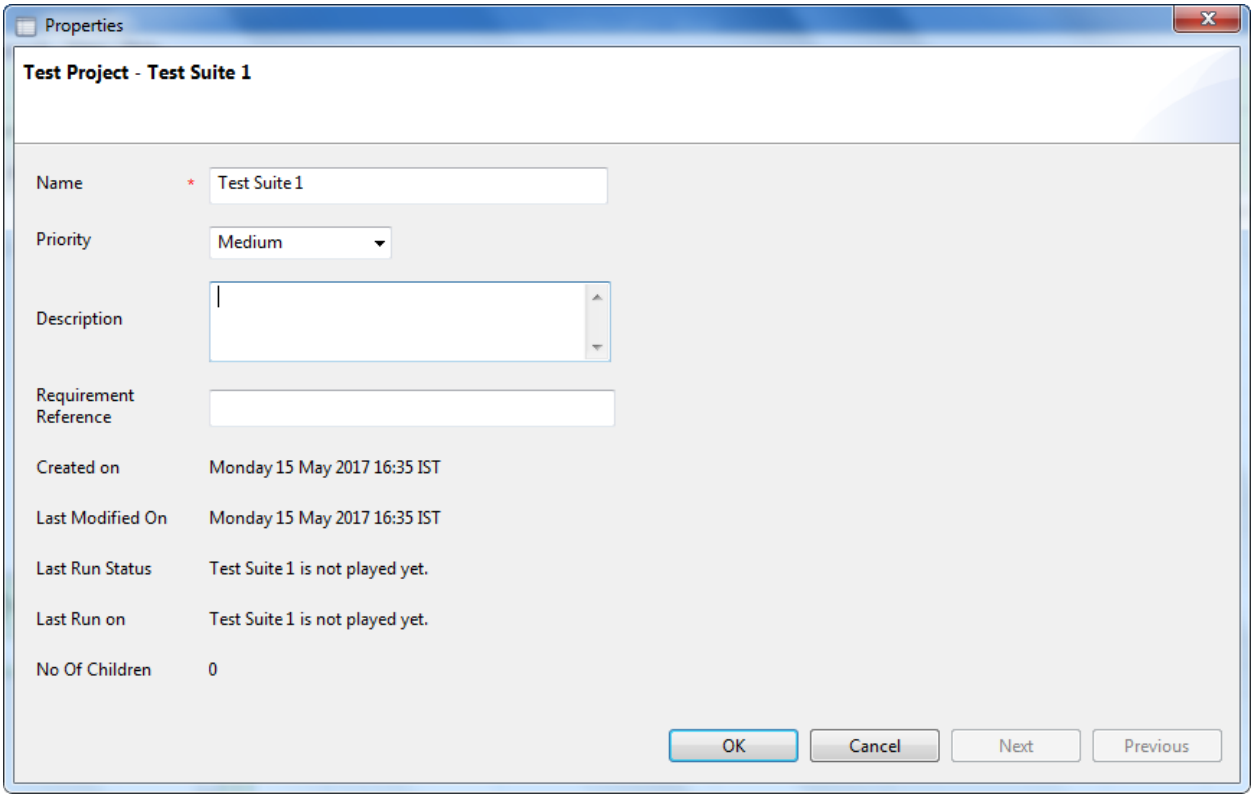


3.2.2.6 Describe Properties of a Test Suite

Select a Test Suite. Right click on it and select Properties.



[Note: *A Dialog Box as shown below will pop-up which will allow a user to describe Test Suite Name, Priority, Description (Optional) and Requirement Reference (Optional). User can also refer Properties Dialog Box to check details like* ***Created Date***, ***Last Modified Date***, ***Last Run Date*** *and* ***Status*** *of a particular Test Suite.*]


Properties
Test Project - Test Suite 1
Name
*
Test Suite 1
Priority
Medium
Description
Requirement Reference
Created on
Monday 15 May 2017 16:35 IST
Last Modified On
Monday 15 May 2017 16:35 IST
Last Run Status
Test Suite 1 is not played yet.
Last Run on
Test Suite 1 is not played yet.
No Of Children
0
OK
Cancel
Next
Previous

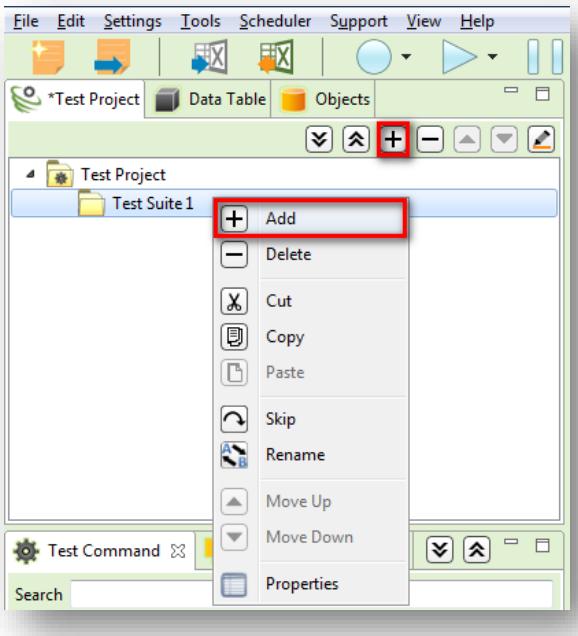
3.2.3 Steps to Add & Manage Test Cases & Test Scripts

3.2.3.1 Add a Test Case

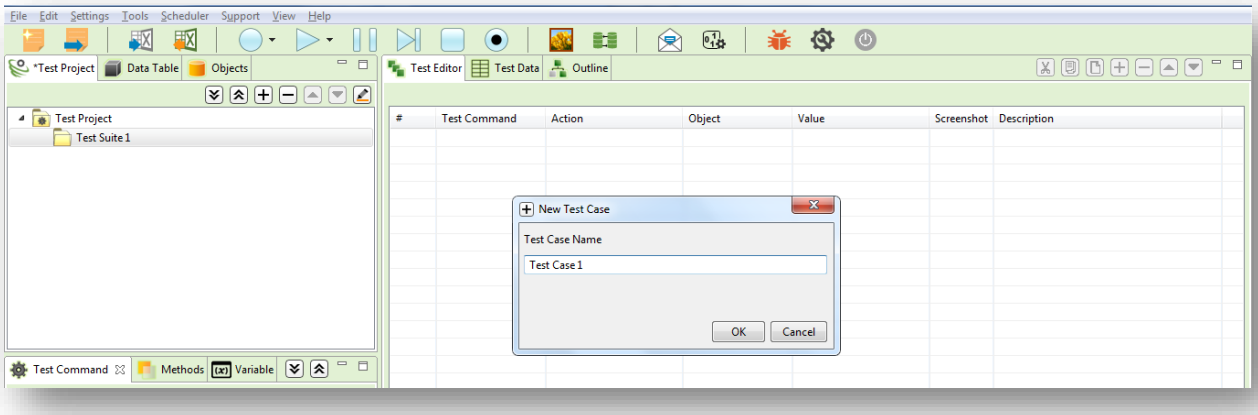
Select the Test Suite as created in Step **3.2.1.1** and click  to create a New **Test Case** within that Test Suite.

OR

Right click on the Test Suite and click **Add**.

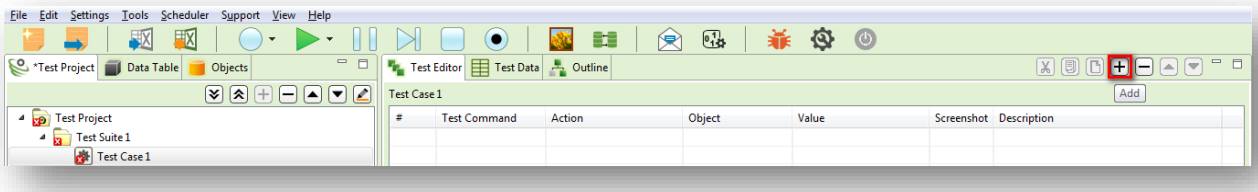


Give an appropriate name of the **Test Case** in the pop-up as shown below and click **OK**.



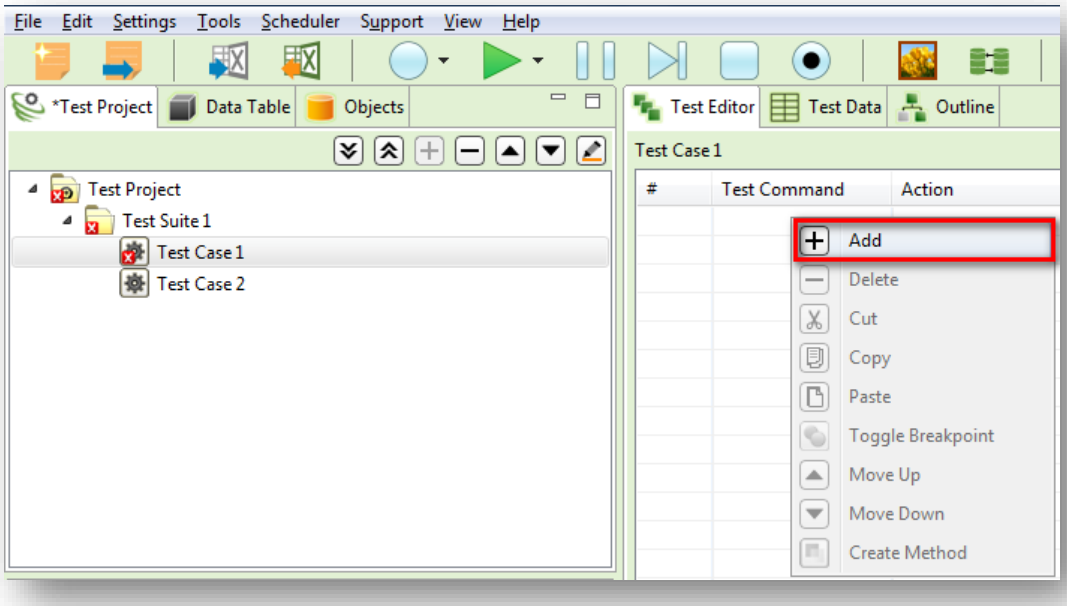
3.2.3.2 Create Test Steps

Select a Test Case and click  button from the **Test Editor Section** to add Test Step.



OR

Right click on the blank area of Test Editor and click **Add**.



After adding a Test Step, add the necessary information to the Test Step in the following process:

Step 1: Add a Test Step.

Step 2: Select Test command to perform from the drop-down list.

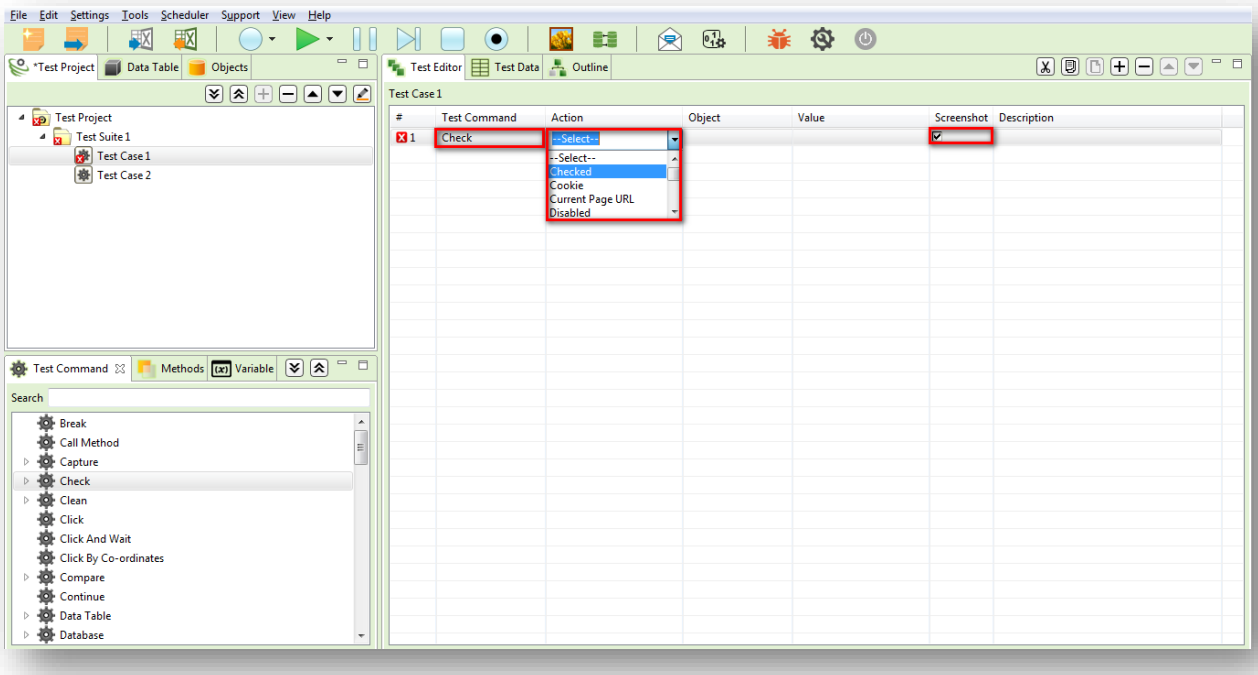
Step 3: Select necessary Action, Object and Input Value in the corresponding row, depending on the testing scenario.

Step 4: Check/uncheck the option of the screenshot.

Step 5: Enter Description if necessary.

Step 6: Add the Next Test Step.

Step 7: Repeat the process in the same manner to create a complete Test Script.



[**Note:** *Series of Test Steps is called Test Script.*]

[**Note:** *User can also create Test Scripts by Importing Test Script or by using the Recording function.*]

Refer Section - **Import Test Script** and **Record Test Script** to learn more.

Sample Test Script to Log into Makemytrip.com without Password and Verify the Text

Step 1: Create Test Case under Test Suite.

Step 2: Name the Test Case **'Login'**.

Step 3: Add a Test Step and input serial number.

Step 4: Select the **'Open Page'** Test command.

Step 5: Add **'https://support.makemytrip.com/customersupports.aspx'** as Value in the corresponding row.

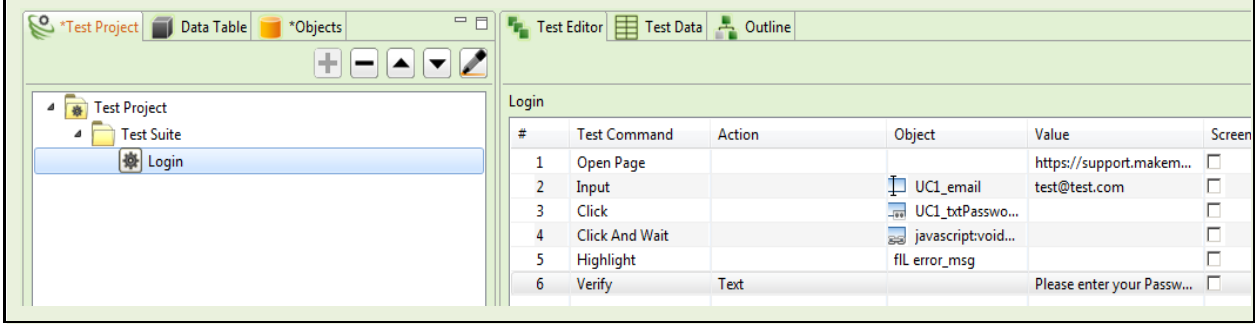
Step 6: Select Test commands as **'Input'**, Object **'UC1_email'** and add Value as '**test@test.com**'.

Step 7: Select Test commands as **'Click'** and add Object as **'plainpassword'** (plainpassword will attempt to login without password).

Step 8: Select Test commands as **'Click And Wait'** and Object as **'javascript:void(0);**

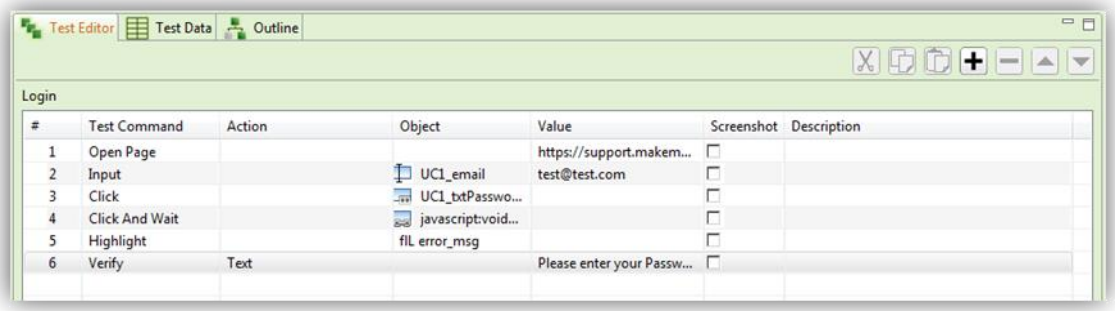
Step 9: Select Test commands as '**Highlight**' and Object as **'errormsg_password'** (this step will highlight error message for not entering the password while logging in).

Step 10: Select Test commands as **'Verify'** with Action as **'Text'** and place Value as **'Please enter your Password'**.



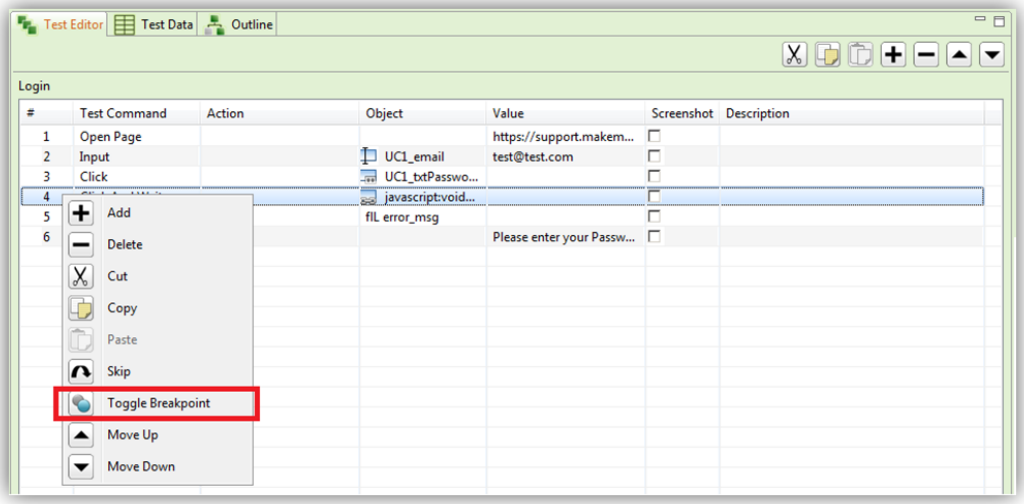
3.2.3.3 Manage Test Steps

Delete, Cut, Copy or **Move** the Test Steps by using the buttons on the **Test Editor** OR by right clicking on the Test Steps.



3.2.3.4 Add Toggle BreakPoint to a Test Step

Right click on a Test Step to put **Toggle BreakPoint** to that step.



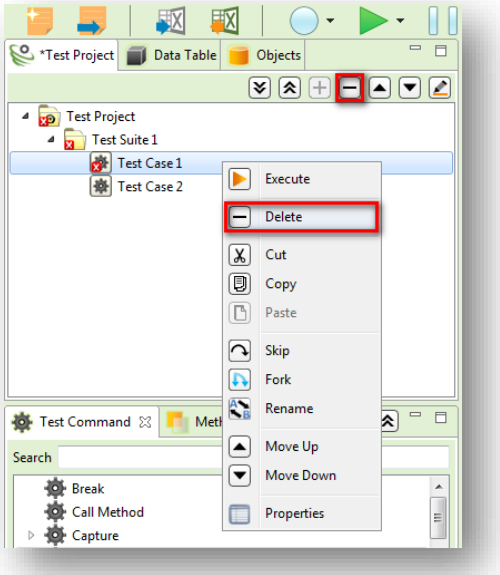
[Note: *This function will automatically pause the execution once it reaches the pre-defined Test Step. This kind of function is useful when a website requires some manual input/intervention during execution. For e.g., Inserting a Captcha Code while filling a form in some websites.*]

3.2.3.5 Delete a Test Script

Select a Test Script and click  to delete that Test Case.

OR

Right Click on a Test Script and click **Delete**.

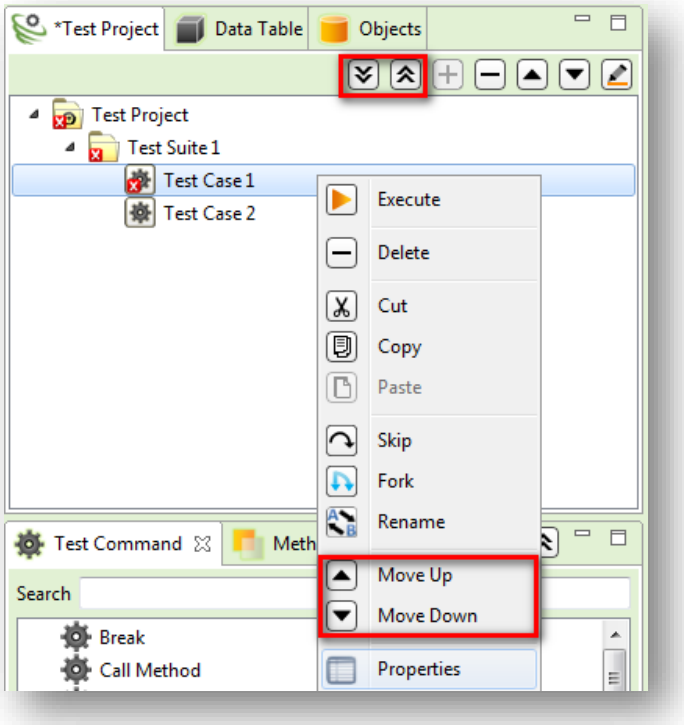

*Test Project
Data Table
Objects
Test Project
Test Suite 1
Test Case 1
Test Case 2
Execute
Delete
Cut
Copy
Paste
Skip
Fork
Rename
Move Up
Move Down
Properties
Test Command
Search
Break
Call Method
Capture

3.2.3.6 Move Up or Move Down a Test Script

Select a Test Script and click  or  respectively to Move Up or Move Down that particular Test Script.

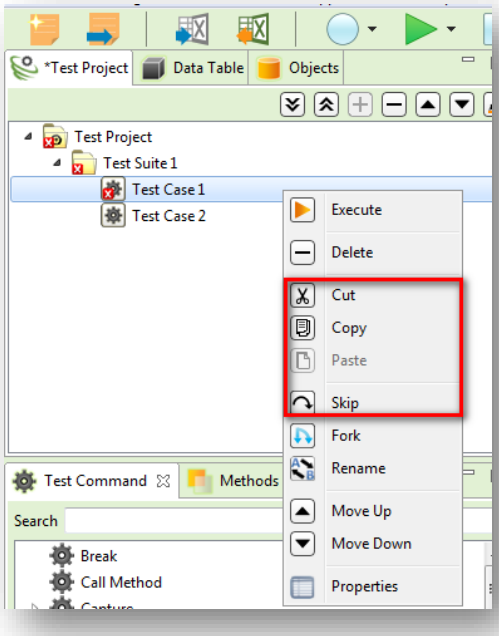
OR

Right Click on and click **Move Up** or **Move Down**.



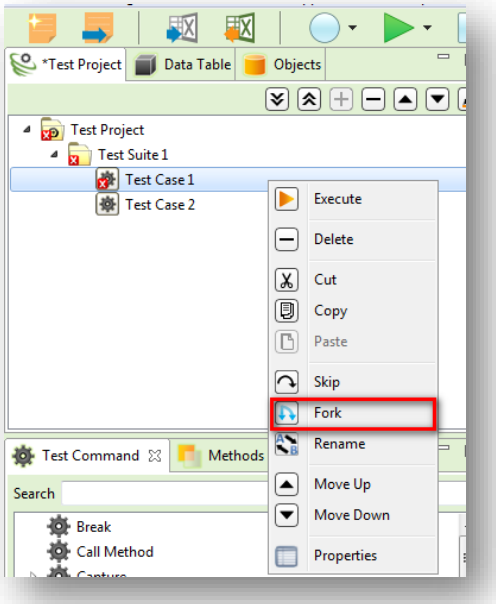
3.2.3.7 Cut, Copy and Skip a Test Script

Select a Test Script and right click on it to Cut, Copy, or Skip that Test Script.



3.2.3.8 Fork

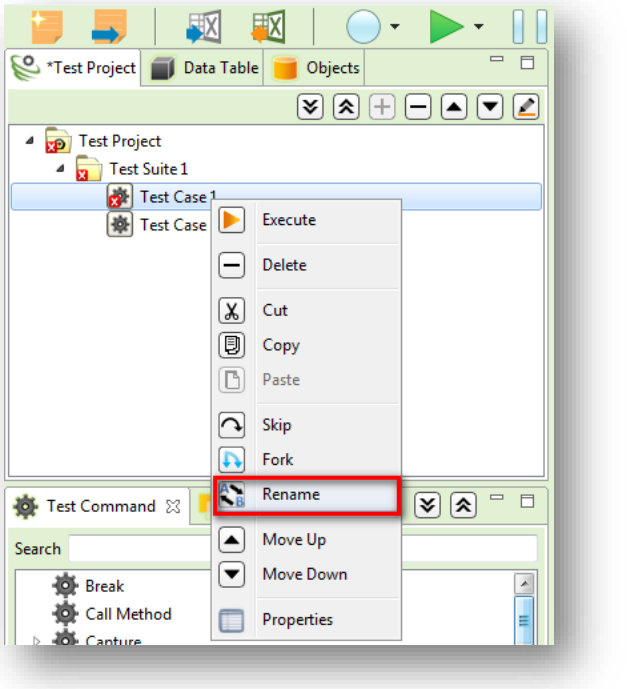
Select a Test Case and right click on it to Fork. Fork function allows users to execute Test Cases on multiple browsers simultaneously from the same instance.



[***Note***: *Default multiple browser opens from the instance i.e. Mozilla Firefox.*]

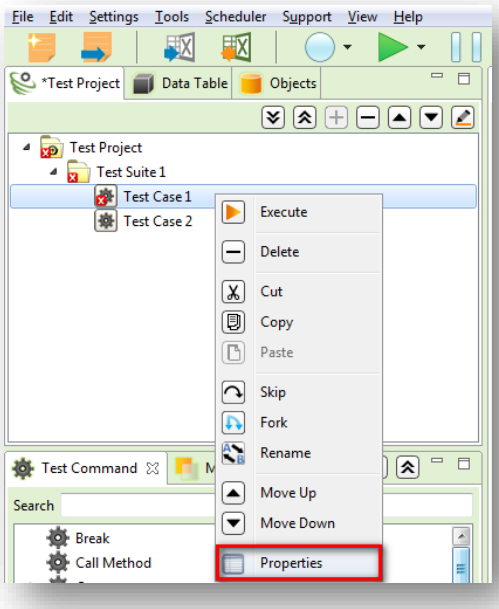
3.2.3.9 Rename a Test Script

Right click on a Test Script and click Rename. Alternately, Press F2 to rename the selected Test Script.

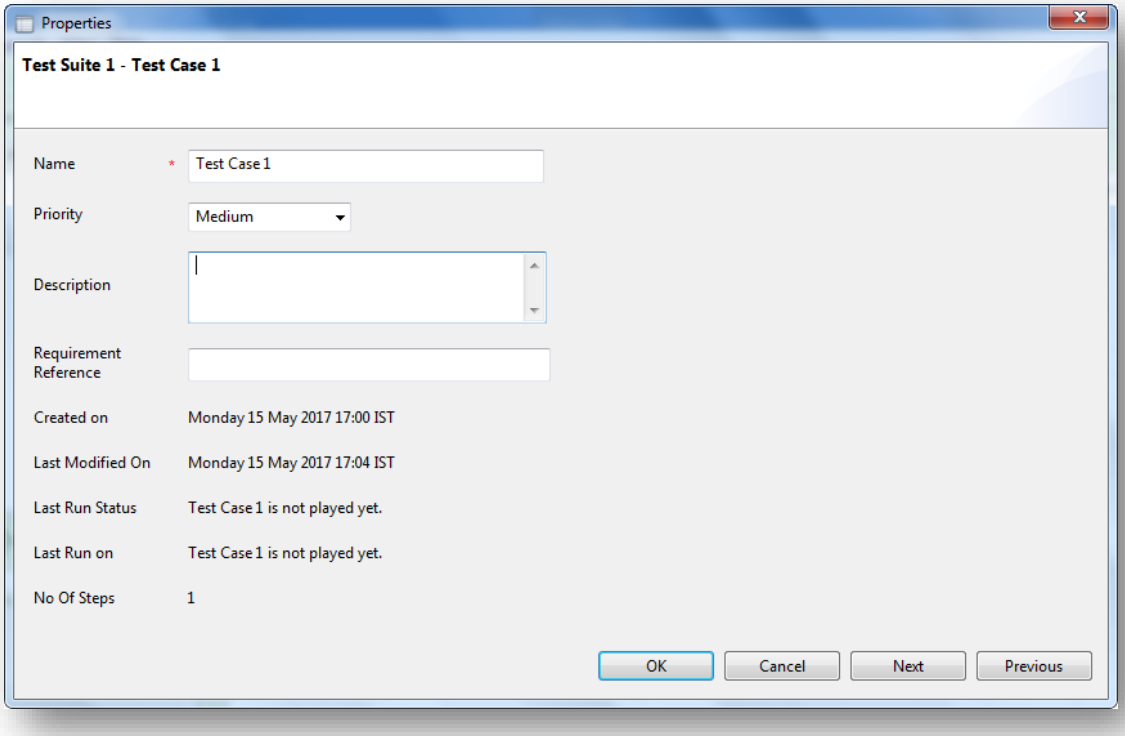


3.2.3.10 Describe Test Script details in Properties

Select a Test Script. Right click on it and select Properties.



[Note: *A Dialog Box as shown below will pop-up which will allow a user to describe Test Script Name, Description (Optional) and Requirement Reference (Optional). User can also refer Properties Dialog Box to check details like* ***Created Date***, ***Last Modified Date***, ***Last Run Date*** *and* ***Status*** *of a particular Test Script.*]

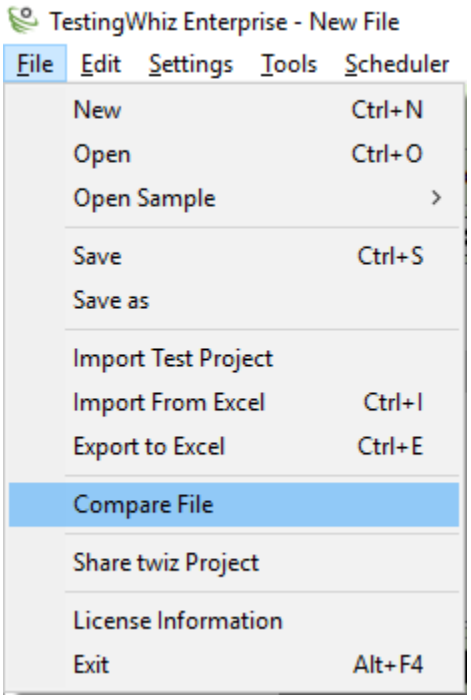


3.3 Compare File Utility

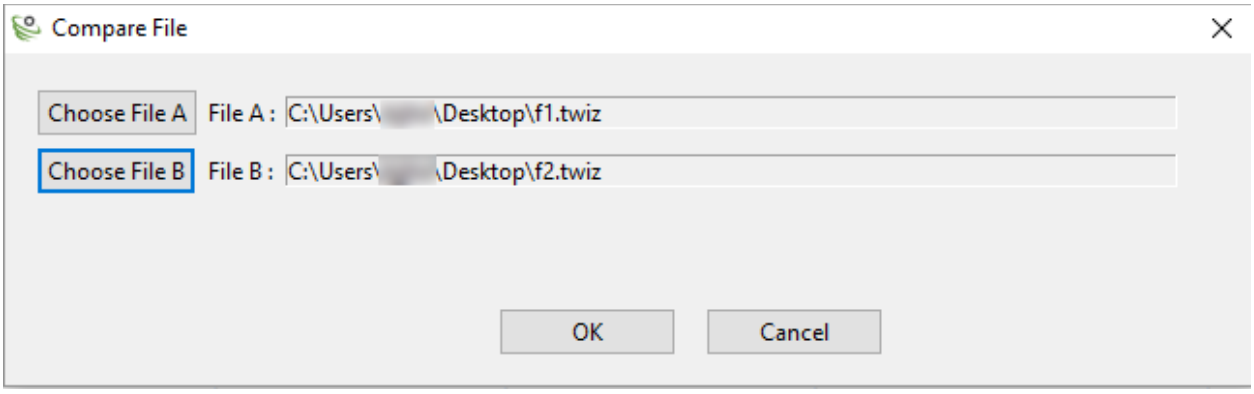
Compare file utility allows users to compare Test Project/Test Suites/Test Cases of two TestingWhiz Project files. Side-by-side comparison window can be utilized to analyze the difference between two test cases. It also allows users to copy a Test Step/Test Case from one file to another, and make changes accordingly in their desired file.

Steps to compare twiz files:

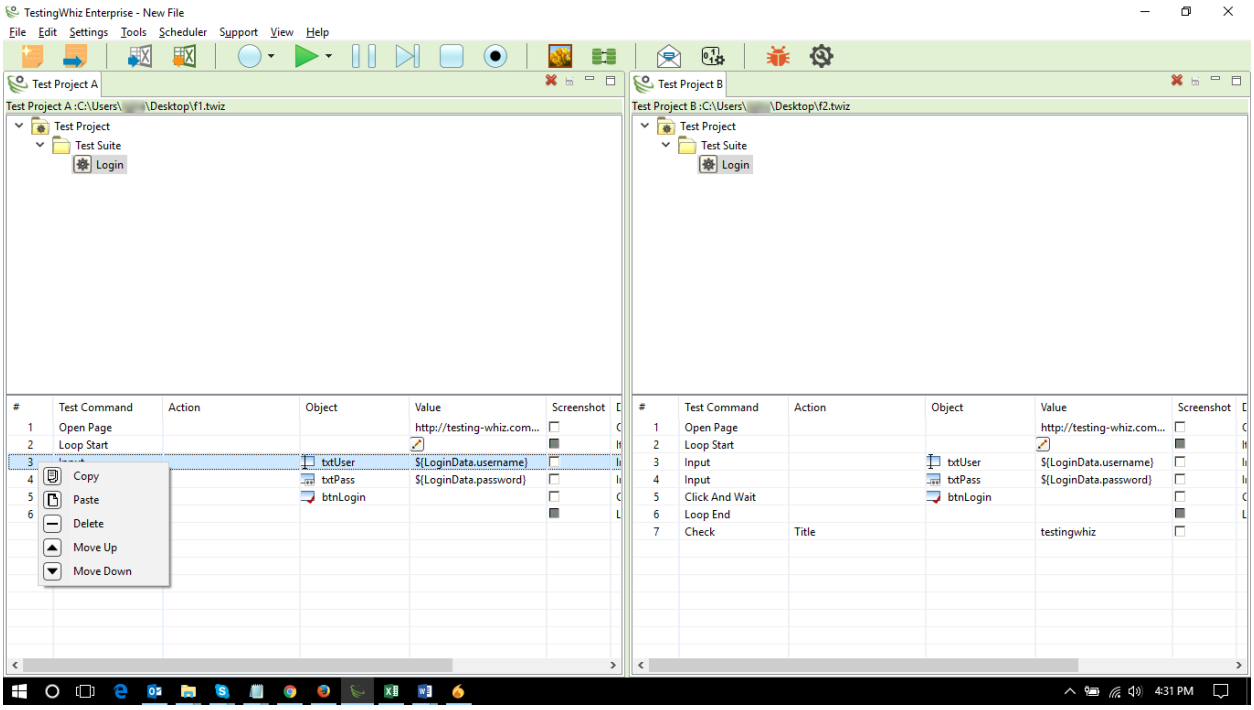
1. Click on **Compare File** from **File** menu of TestingWhiz.



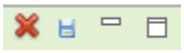
2. Browse files for comparison and click on **OK** as follows:



3. Users can analyze and make changes in their respective files if needed from the File comparison window as follows:



4. Click on the Save button of the respective file to save the changes made.



5. Click on close button after completing the changes.

4 PROCESS OF CREATING, EXECUTING, REPORTING & MANAGING TESTS IN TESTINGWHIZ

4.1 Create, Record and Import Automation Test Scripts

There are three methods of generating Test Scripts. Let us understand each of these three ways of generating Test Scripts.

1. Creating Manually
2. Recording (Using Internal Browser and External Browser)
3. Importing

4.1.1 Create Test Automation Scripts Manually

To execute a test, create a Test Script under Test Suite.

Refer Section – **Steps to Add & Manage Test Scripts** to understand the process of creating Test Scripts.

4.1.2 Record to Create Test Script Using Internal Browser

User can record test scripts to avoid creating scripts manually. Follow these simple steps to generate a test script by recording:

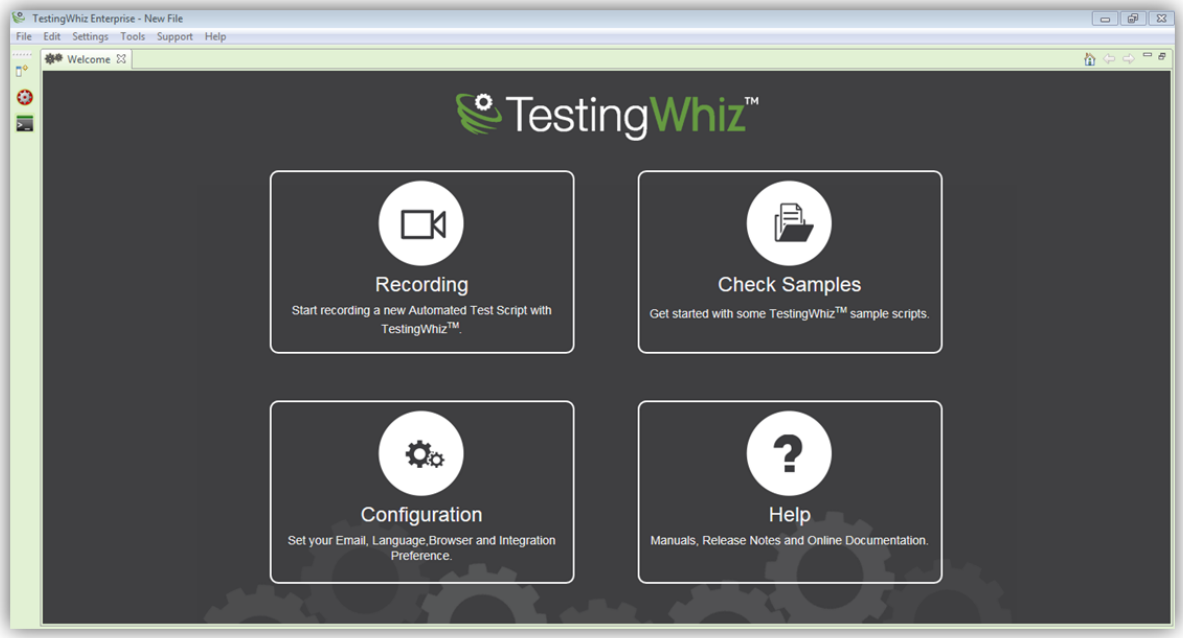
4.1.2.1 Switch On Recording

Click  on the Tool Bar to switch on Recording on the browser of preference.

[**Note:** *The moment a user clicks on the Record button, it will turn* Red  .]

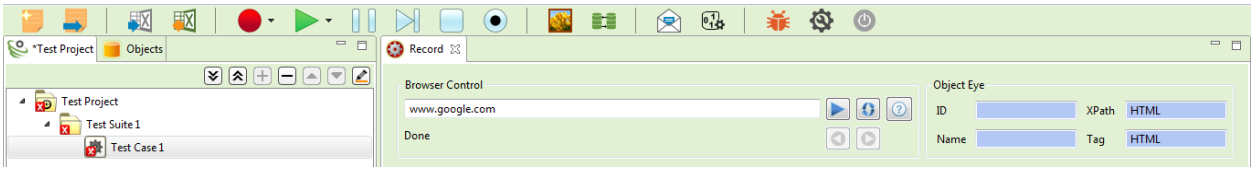
OR

Click **Recording** button on the **Welcome screen**.



4.1.2.2 Enter Browser Control

Enter the URL of the website to test (For e.g. www.google.com).



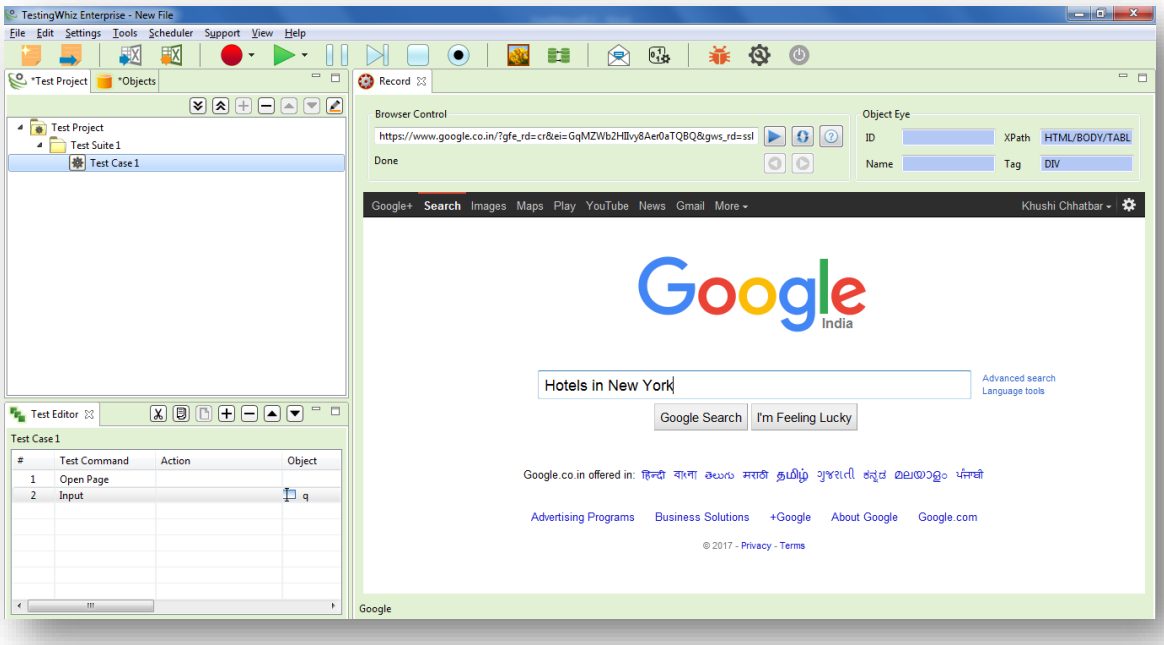
4.1.2.3 Start Recording

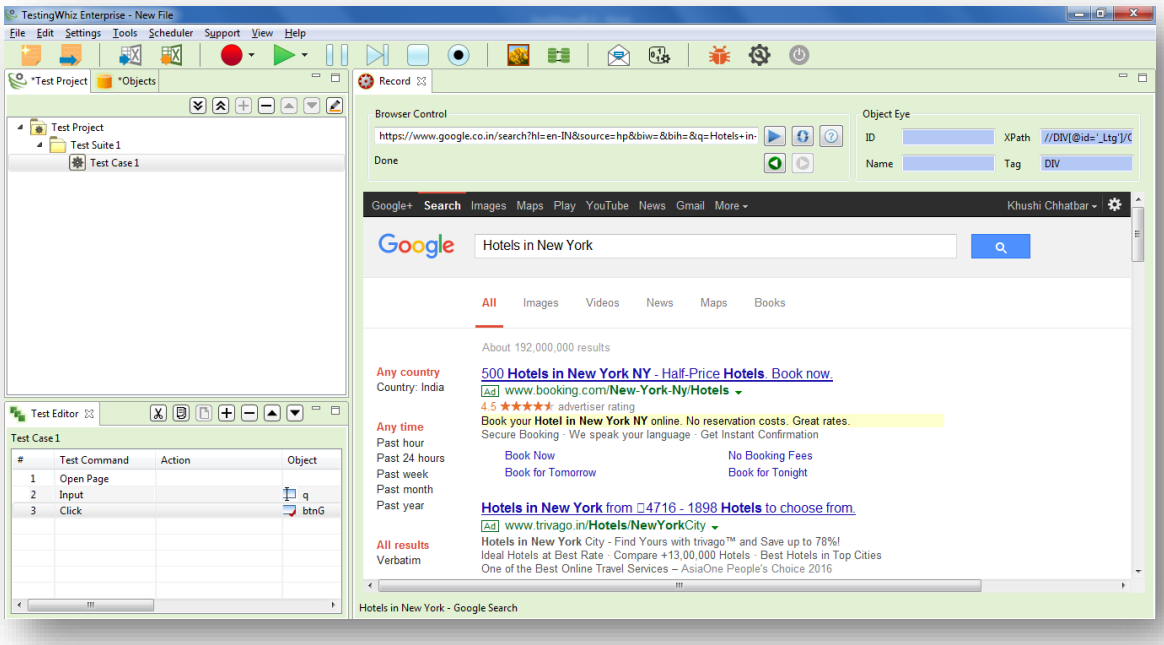
Click  near the Browser Control to begin the execution process. TestingWhiz will open the website in the Test Editor section.

Perform the required function like search, fill contact form, etc. (For e.g., Search '**Hotels in New York**').

Test scripts will start getting generated simultaneously in the bottom left section as user performs function.

Perform the necessary process further to complete the recording process.





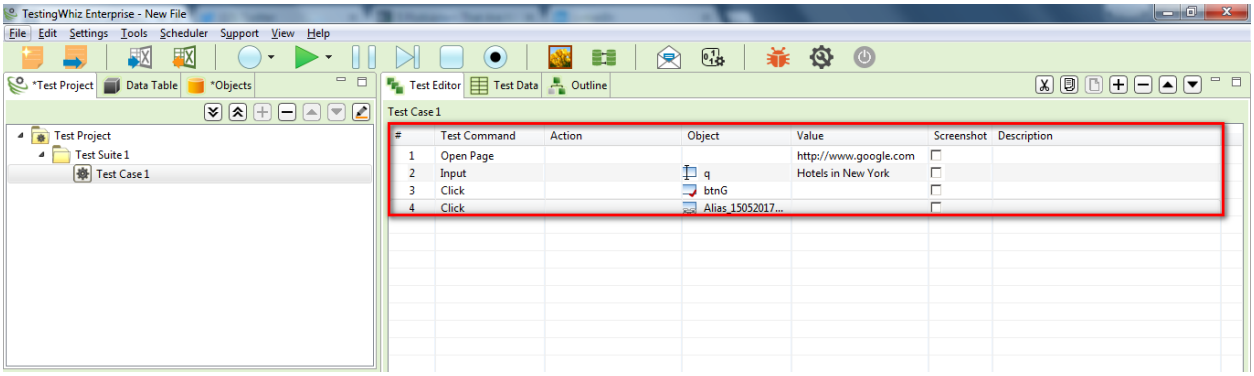
[Note: *While recording the test steps, a user can any time click back*  *or forward*  *buttons to add previous and next steps in the test script respectively.*]

4.1.2.4 Switch off the Recording

Click on  button to stop the recording once a user reaches the desired step(s).

The tool will automatically generate the script based on the steps followed during the recording process.

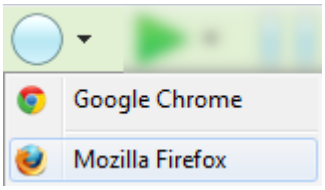
Here is the example of the test script generated by recording the process of searching '***Hotels in New York***' in www.google.com.



4.1.3 Record to Create Test Script Using External Browser

User can also generate Test Scripts by recording using external browser. Follow these simple steps to record using external browser.

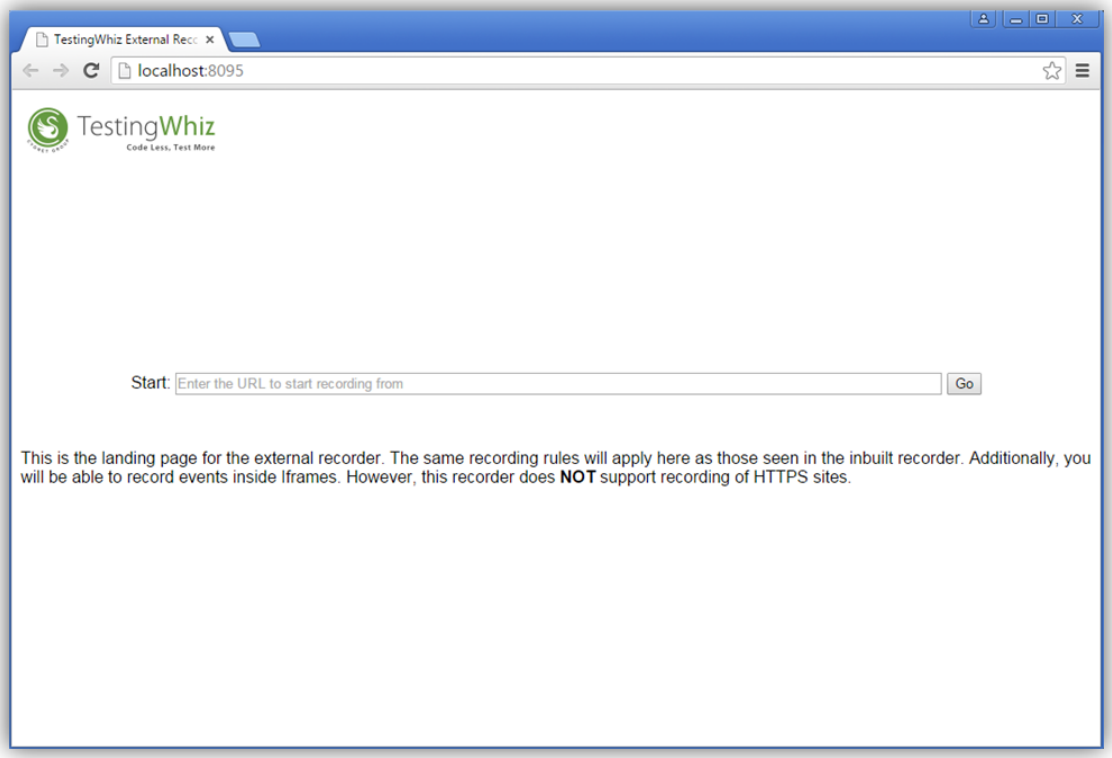
4.1.3.1 Switch On Recording

Click 

the Drop-down below the Record button to switch on recording.

[**Note:** *The moment a user switches on the Recording, it will turn* Red  .]

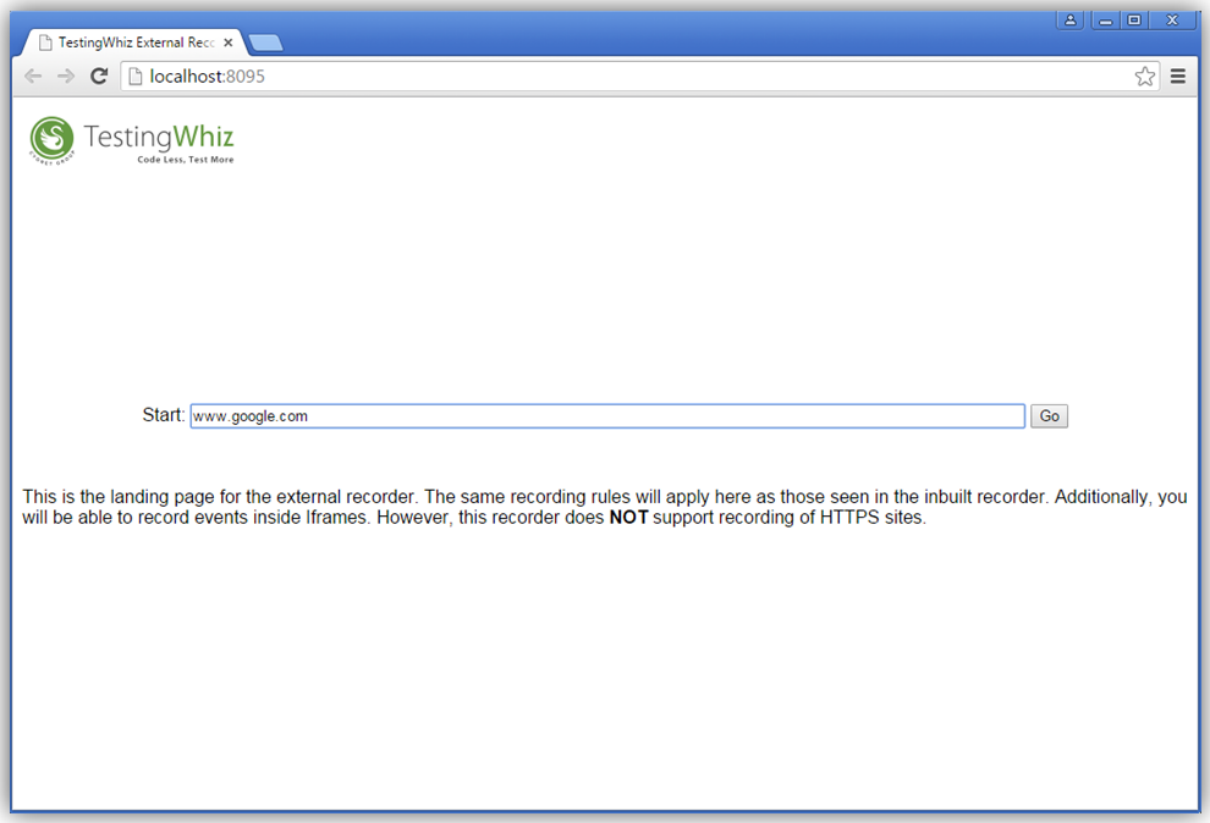
[**Note:** *A new window will get opened on Google Chrome.*]


TestingWhiz External Rec
localhost:8095
TestingWhiz
Code Less, Test More
Start: Enter the URL to start recording from
Go
This is the landing page for the external recorder. The same recording rules will apply here as those seen in the inbuilt recorder. Additionally, you will be able to record events inside Iframes. However, this recorder does NOT support recording of HTTPS sites.

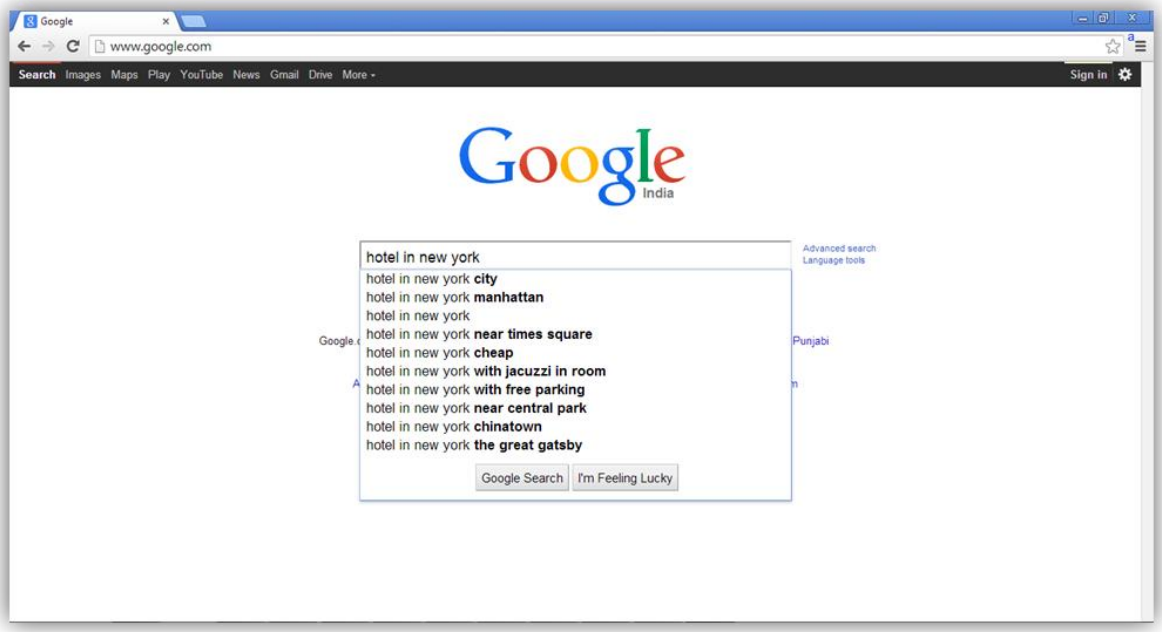
4.1.3.2 Enter URL

Enter the URL of the website to test (For e.g. www.google.com).

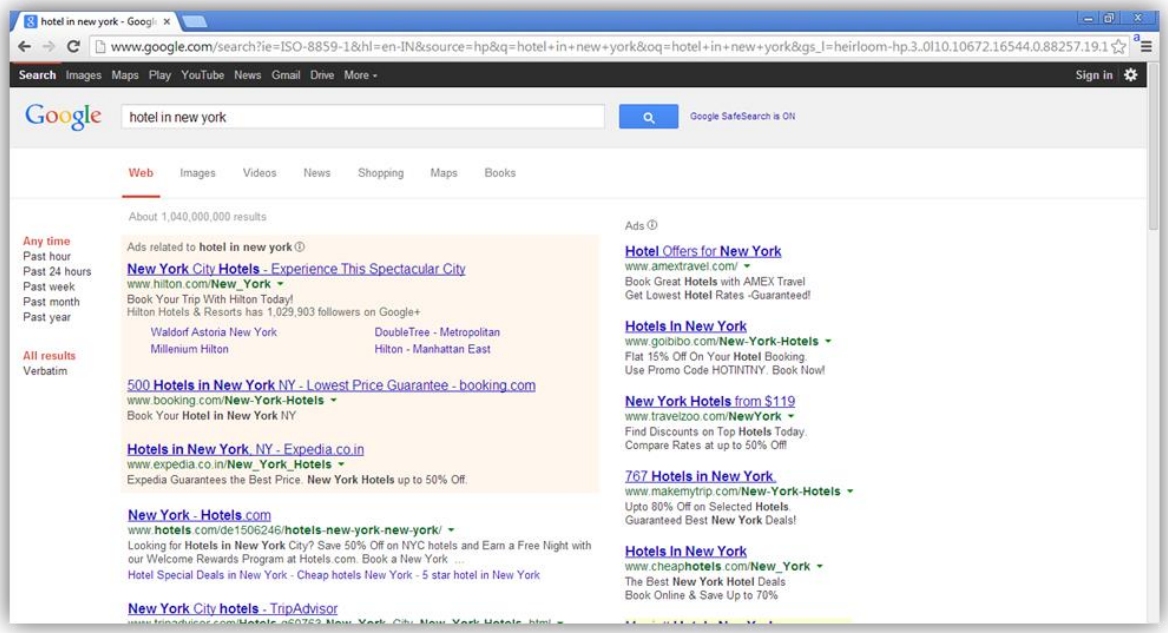
Click Go near the Browser Control to begin the recording process. The website will be open the website in the external browser window.



Perform the required function like search, fill contact form, etc. (For e.g., Search '**Hotels in New York**').



Test Script will start getting generated in the bottom left section as you perform function. (For e.g. Search 'Hotels in New York').



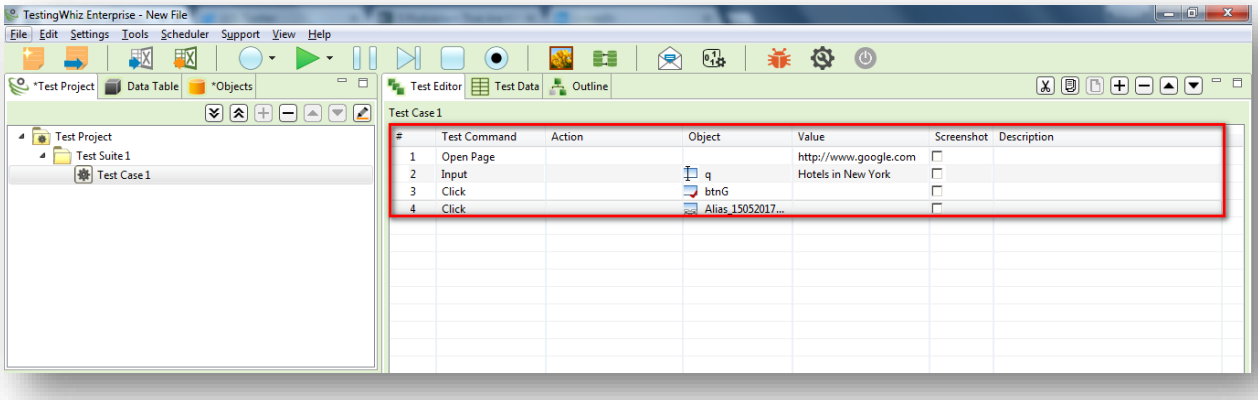
Perform the necessary process further to complete the recording process.

4.1.3.3 Switch off the Recording

On reaching the desired point of recording the test steps, click on  button to stop the recording.

The tool will automatically generate the script based on the steps followed during recording process.

In this case, we browsed 2 pages of Google search results for the Hotels in New York. Here's the script generated:



4.1.4 Record to Create Test Script using Visual Recorder

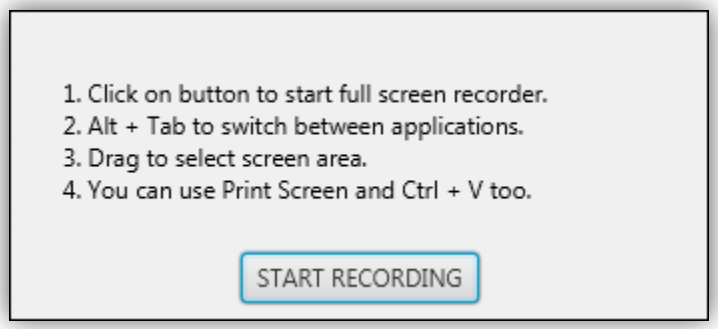
User can generate Test Script by recording using Visual Recorder.

4.1.4.1 Switch on Recording

Click  to switch on recording.

[**Note:** The moment a user switches on the Recording, it will turn red ]

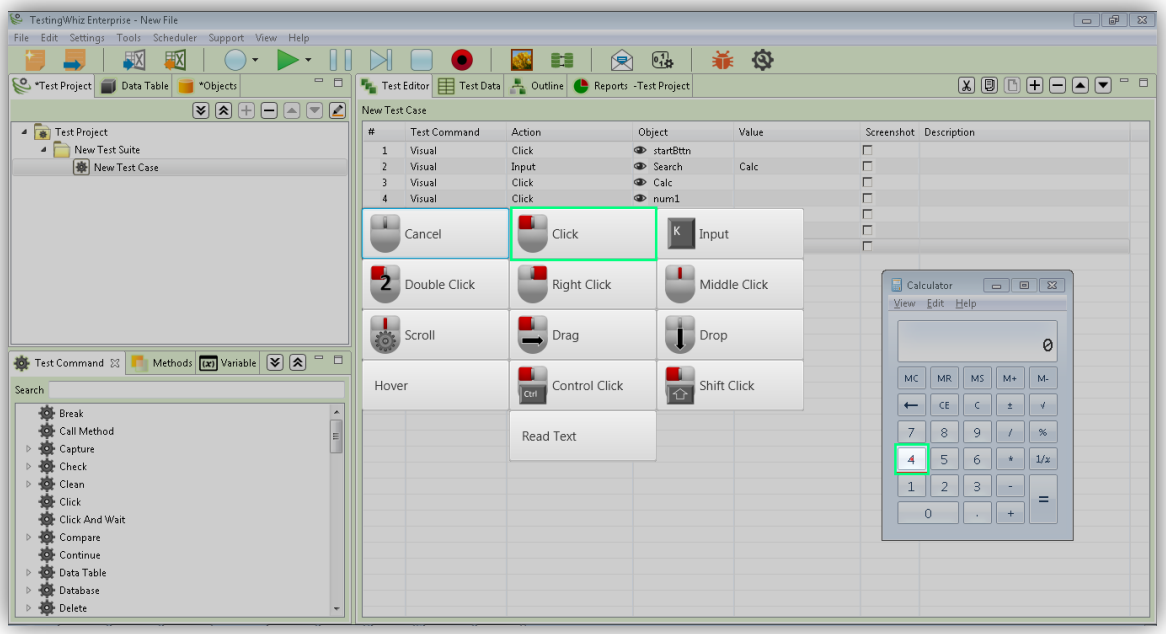
Clicking on the Visual Recorder opens up the dialog box.



4.1.4.2 Start Recording

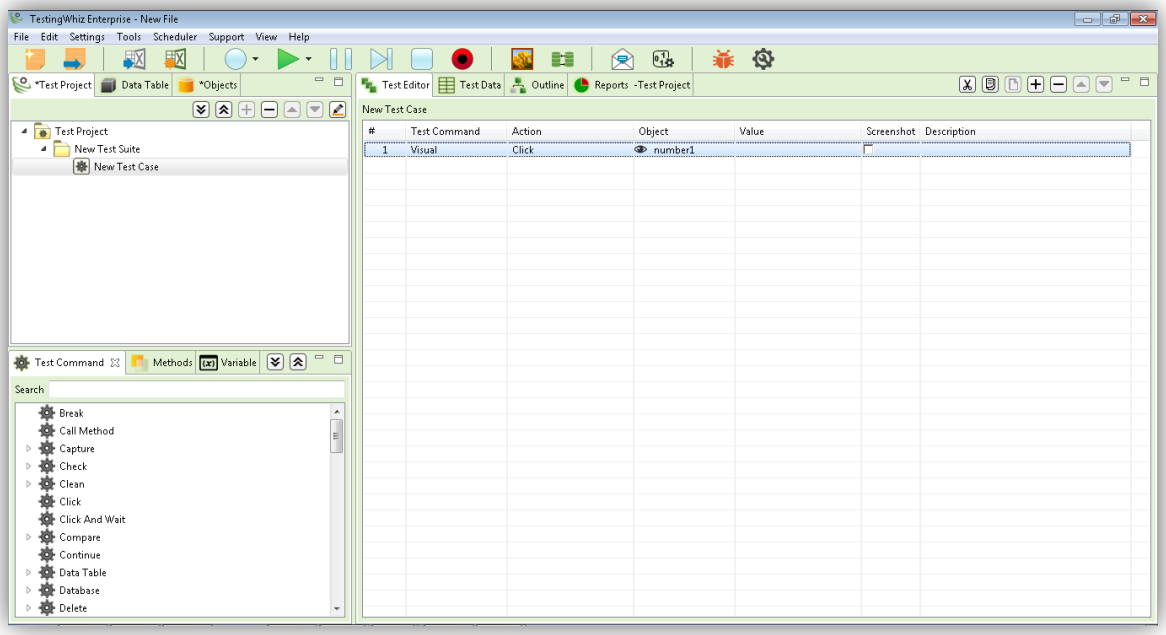
On clicking "Start Recording", the visual recorder will start recording the screen.

For e.g. Select an area on calculator to record. That will open the options for action. To click on



number 4, we have to select "Click" action.

Test script will be generated automatically.



Perform the necessary process further to complete the recording process.

4.1.4.3 Switch off the Recording

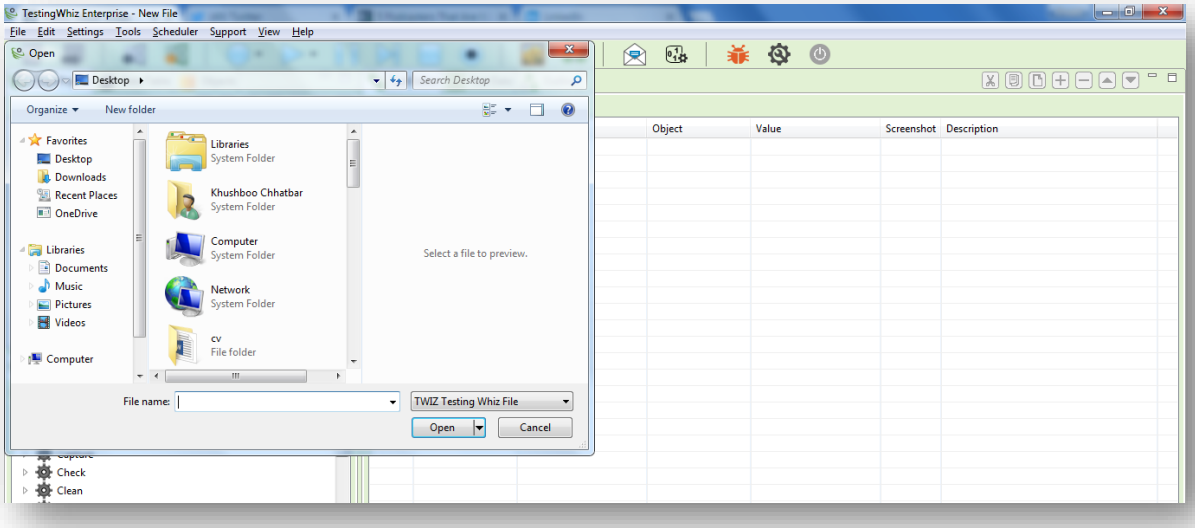
Click on  button to stop the recording once the user has finished the desired scenario. The tool will automatically generate all the test steps according to the scenario as shown in the above image.

4.1.5 Import Test Script

Import Test Script is another feature that lets a user build test automation scripts. User can import script file in **Excel** or .twiz format by following below mentioned steps.

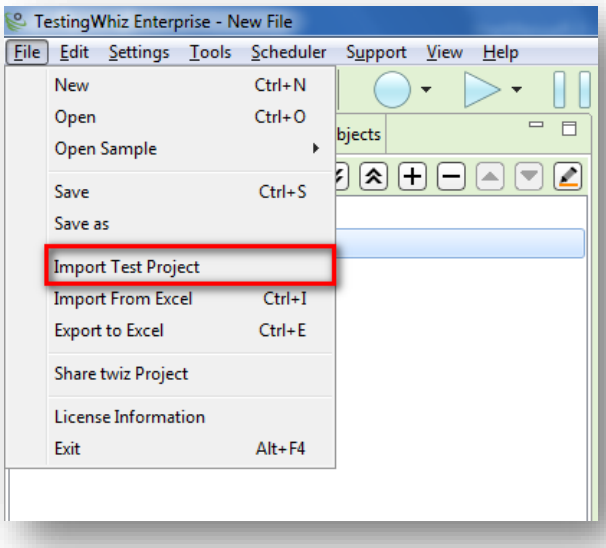
4.1.5.1 Import Test Project

Click  from the Tool Bar to import Test Script stored in .twiz file.



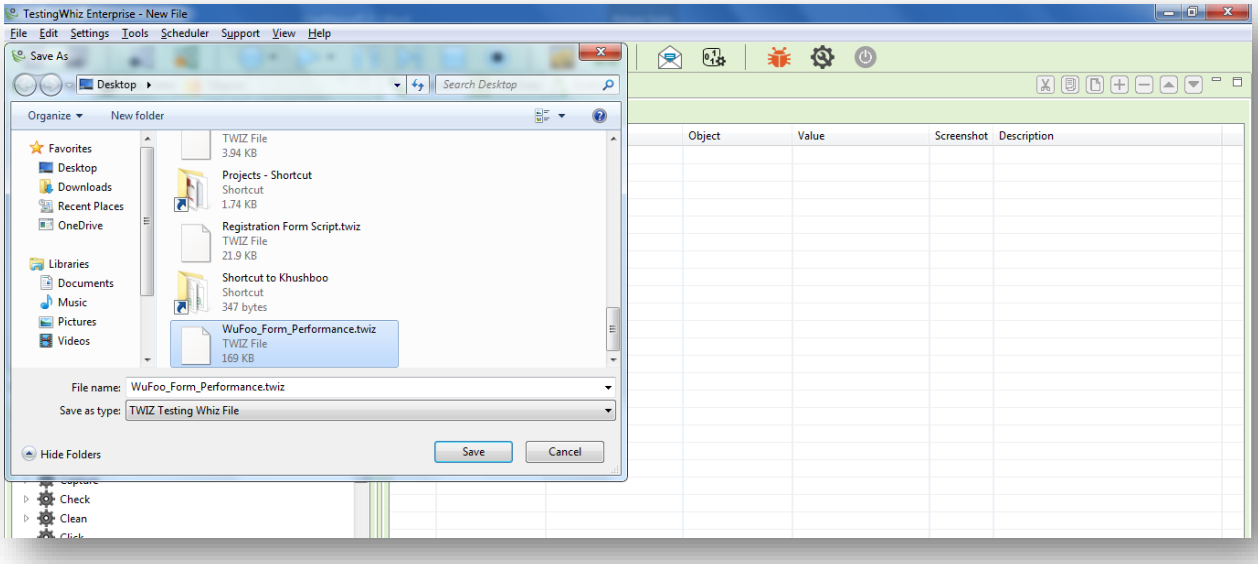
OR

Open File menu and click **Import Test Project**.



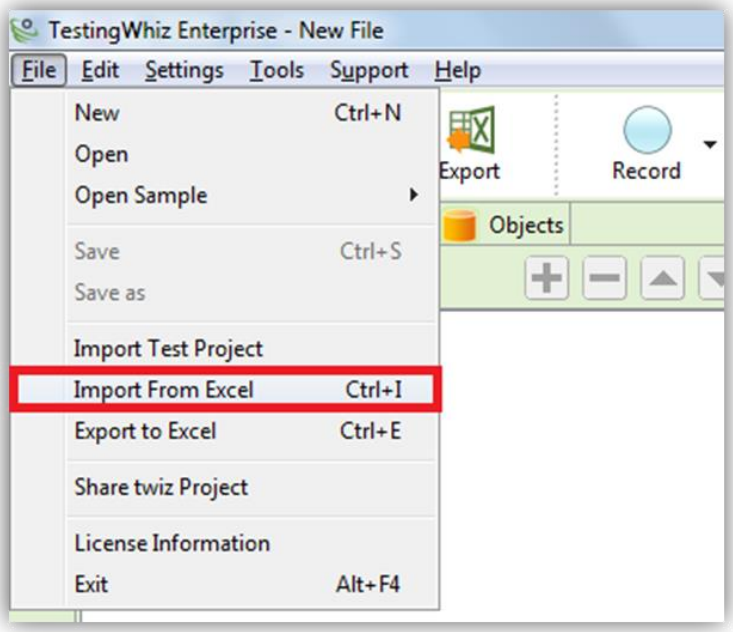
4.1.6 Import from Excel

Click  from the Tool Bar to import Test Script saved as .xlsx or .xls.



OR

Open File menu and click **Import from Excel**.



4.2 Execute Test Script

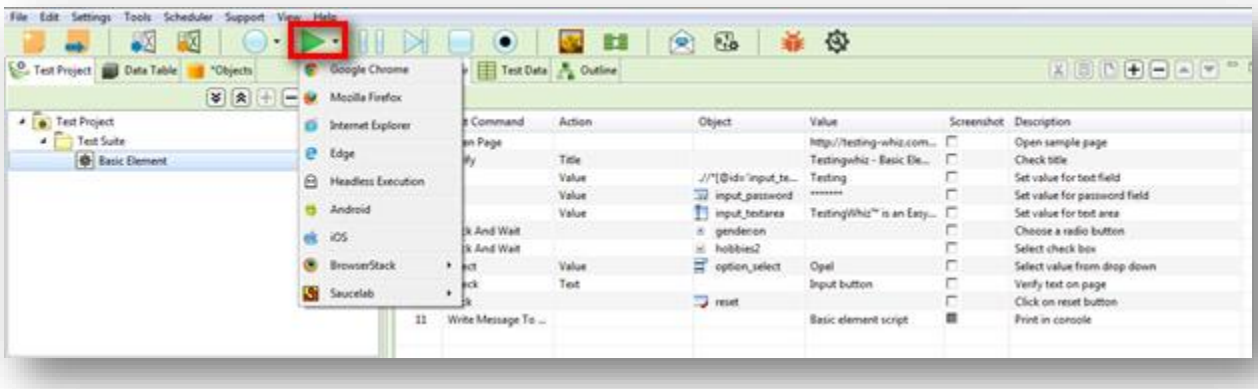
4.2.1 Select Browser

Click down arrow in the  to select the browser to execute the created/imported/recorded Test Script(s).

[**Note:** *If a user selects a browser that doesn't exist in the system, and if fallback browser flag from* ***Settings>Configuration>Execution*** *is selected then, TestingWhiz will fall back to another browser and execute the Test Script.*]

4.2.2 Run Test Script

Click  from the Tool Bar to execute a Test Script.

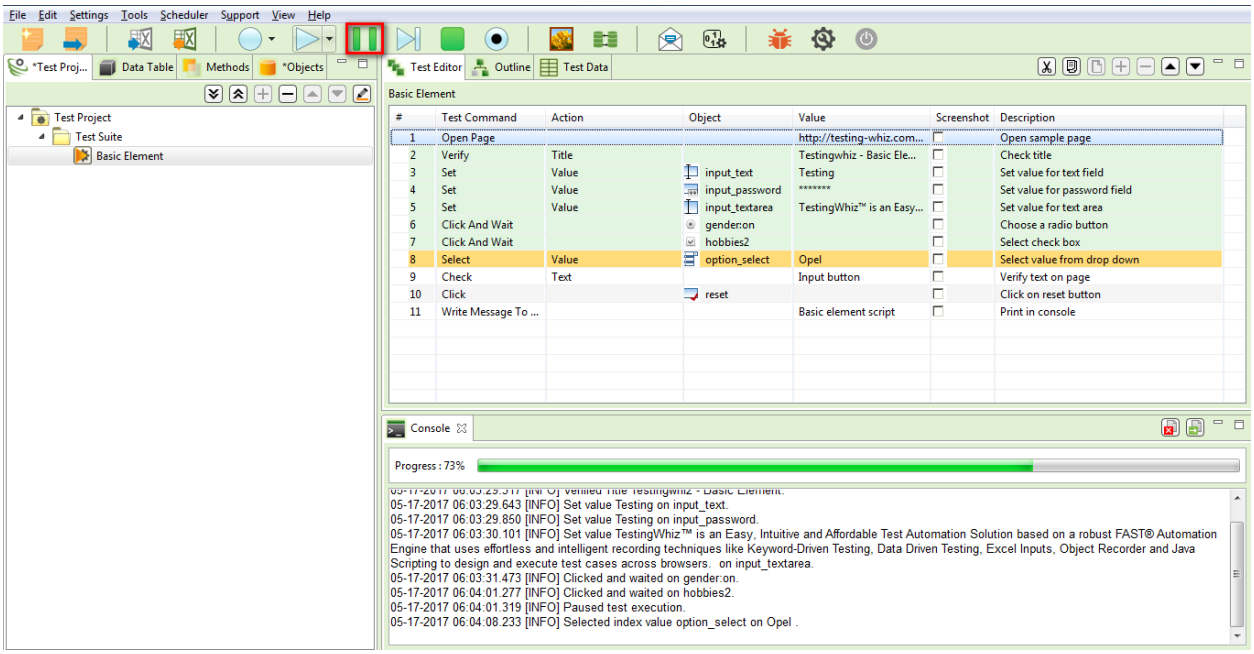


[**Note:** *Test Script will be executed in the default browser selected by the user.*]

[**Note:** *During execution, TestingWhiz will display the Active Screen of the website being tested.]*

4.3 Pause Test Execution

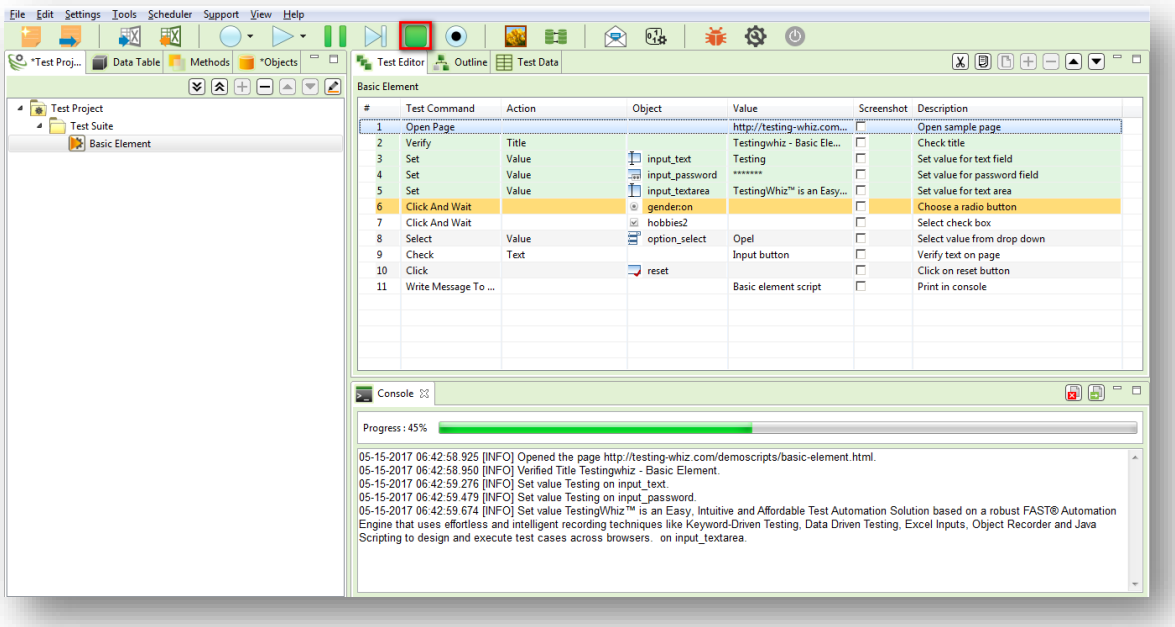
Click on  to Pause live execution of a Test Script.



[**Note**: *Click on Pause button only when it has turned Green.*]

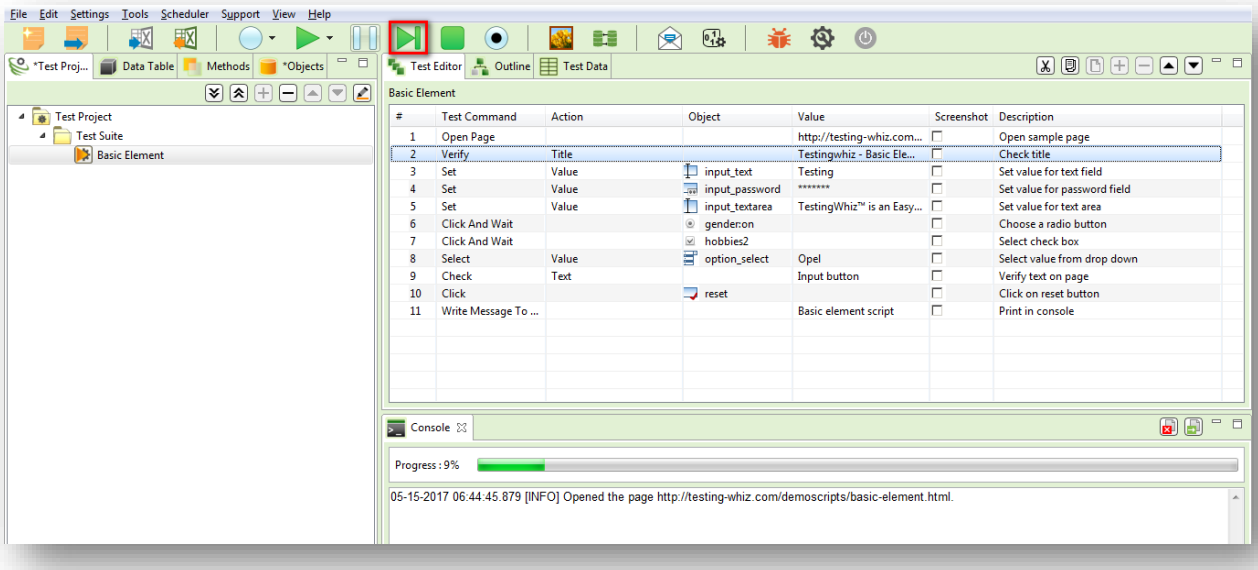
4.4 Stop Test Execution

Click on  to stop test execution.



4.5 Move to Next Step

Click  to move to the next step if any of the step contains Toggle Breakpoint from where the execution has been paused automatically.

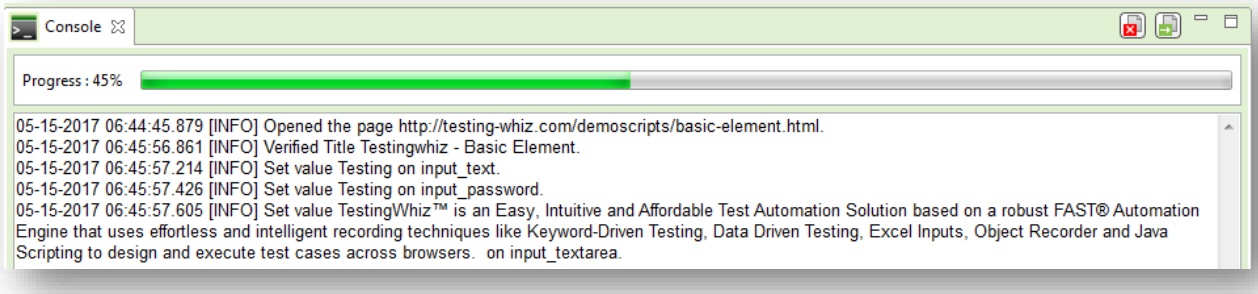


[**Note:** *Step 5 in the above screen contains Toggle Breakpoint where the execution has been paused automatically. Click on Next button will resume the testing to execute next step.*]

Refer Section – **Pause Test Execution** & **Toggle BreakPoint** to learn more.

4.6 Check Progress and Execution Log

Check the progress and test execution log in the Console section exactly below the Test Editor.

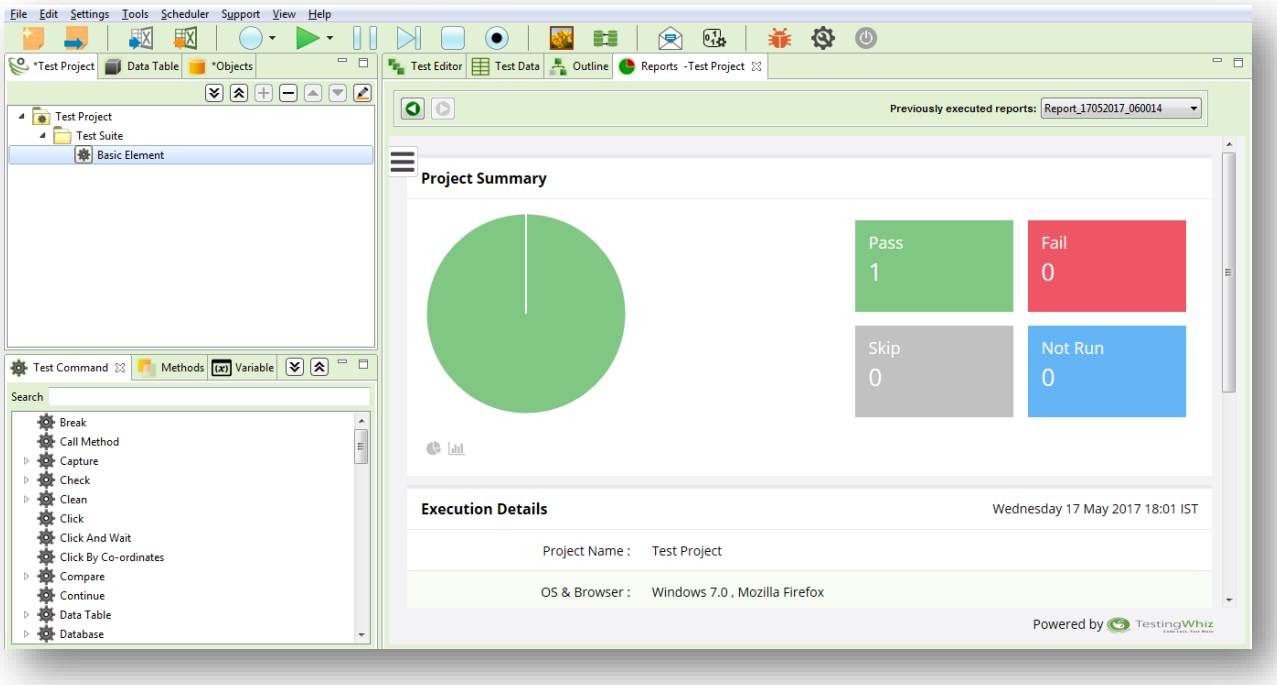


4.6.1 Clear or Export Logs

Clear the summary/log of execution using  button or Export the log using  from the Console tab.

4.7 Test Report

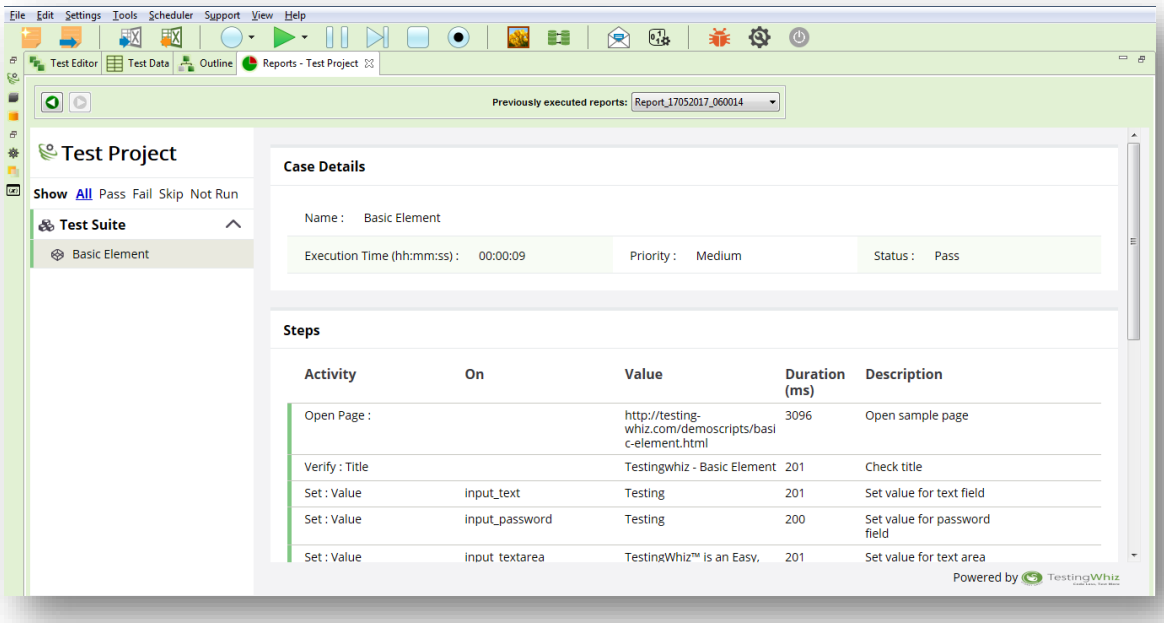
At the end of the testing cycle, a report will be generated which will contain the test results along with the execution log. Test Report describes the actions performed and the results of the action.

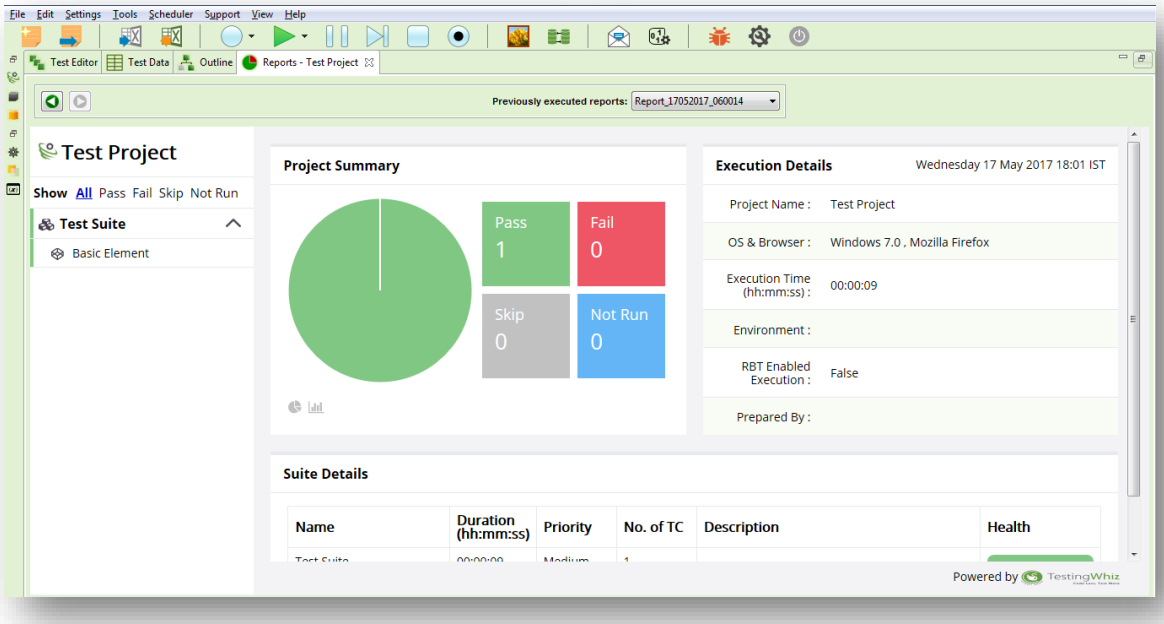


4.7.1 Analyze Report

Scroll towards the right or click on the maximize button  to view a detailed report of the test execution.

Click **Pass/Fail/Skip/Not Run** to view complete details of the test execution along with the time taken to complete each step.



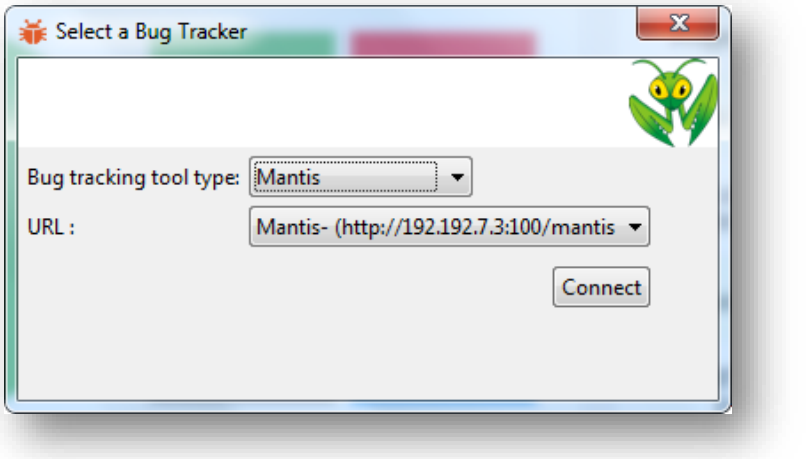


[Note: *The test reports will be stored in "C:\Users\<username>\.whiz\reports" on the user's machine.*]

4.8 Log a Defect

If a test case fails, log a defect using Bug Tracking Tool.

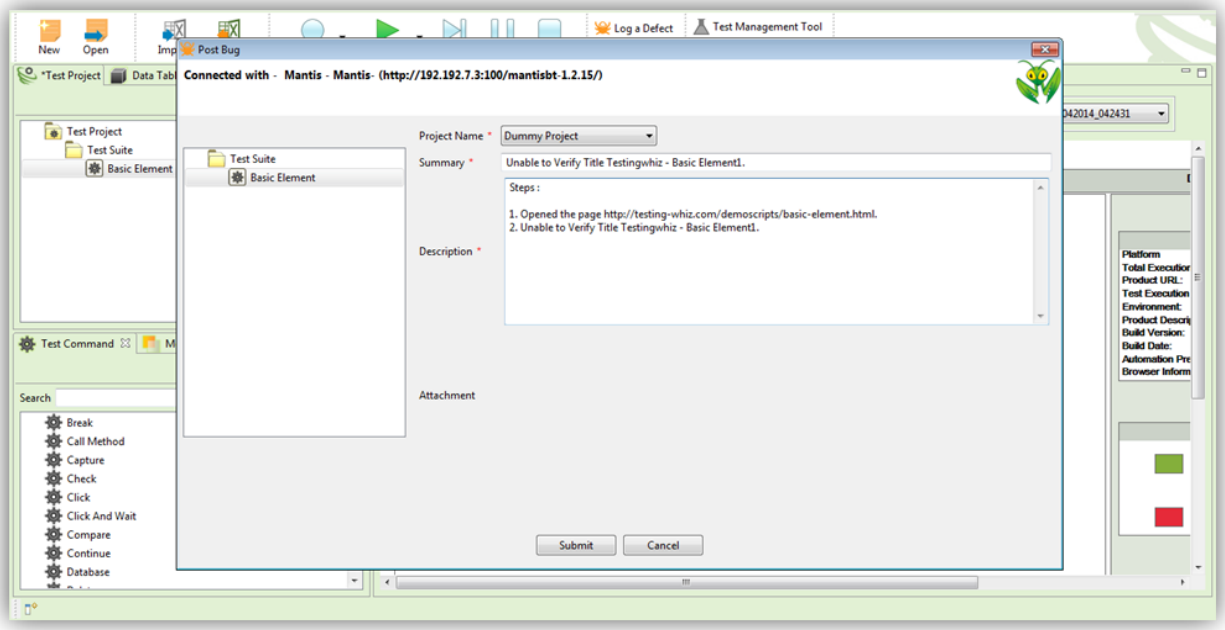
Step 1: Click  from the Tool Bar. Select the Bug Tracking Tool and URL and click **Connect**.



Step 2: After the connection, has been successful, select the Project Name from the drop-down on the right side.

Step 3: Now select the respective Test Case from the left pane.
[**Note:** *Details will be automatically populated in the form.*]

Step 4: Click **Submit**. A Ticket Number will be generated and the defect will be registered in the respective Bug Tracking Tool.



[**Note:** *This feature will function only if a user has set Bug Tracking Tool credentials in the Configuration section.*]

[**Note:** *After a user submits the defect once, the Submit button will become disabled so as to prevent the user from submitting the same defect again as a duplicate.]*

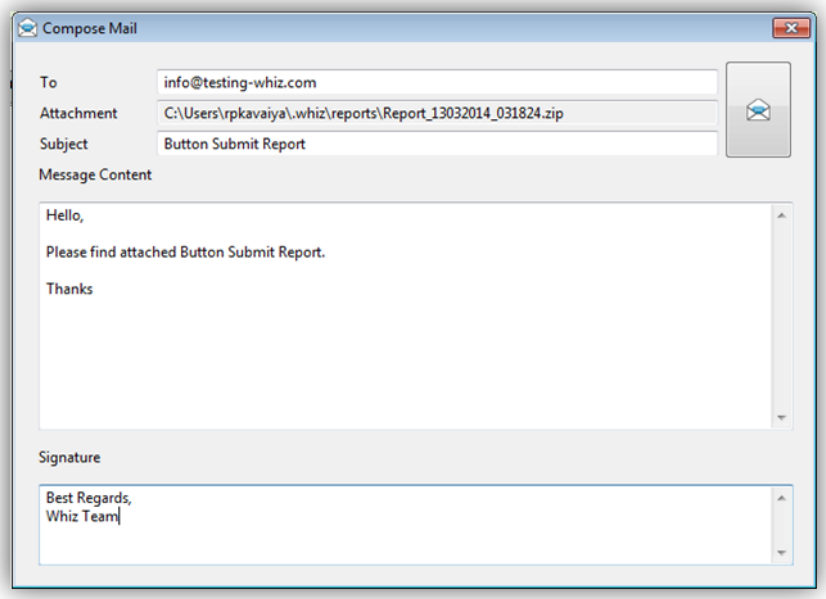
4.9 Email Report

Email complete report of the executed test cases using the Email Report feature.

Step 1: Click  from the Tool Bar.

Step 2: Enter recipient's email id in the '**To**' field.

Step 3: Enter **Subject** and **Message Content** and click  to send the report.



[**Note:** *This feature will function only if a user has set Email Preferences in the Configuration section.*]

5 KEYWORD-DRIVEN & DATA DRIVEN TESTING IN TESTINGWHIZ

5.1 Keyword-Driven Testing

Using Keyword-driven testing approach, TestingWhiz separates much of the programming work of Test Automation from the actual Test Design. Testers or Test designers can write the test cases based on a set of keywords into a table. The test is executed using a driver that reads the keywords and executes the corresponding codes.

5.1.1 Setting up Keyword-Driven Test Script

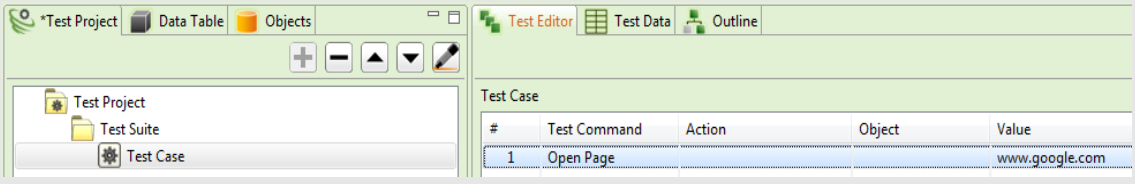
Step 1: Create a Test Suite under Test Project.

Step 2: Add a Test case.

Step 3: Select Test Command from the available test commands as a keyword which associates with the action to be performed.

Step 4: Add Value in the corresponding cell to perform the function.

Step 5: Complete the Test Script as per the steps mentioned in section **3.2.3**.

Sample Test Script to open – www.google.com
Step 1: Create a Test Suite under Test Project. **Step 2:** Add a Test case. **Step 3:** Go to Test command and pick the option '**Open Page**' from the dropdown. **Step 4:** Put www.google.com in '**Value**' column. 

5.2 Data-Driven Testing

Testing a particular module for various valid/invalid combinations of data sets is a vital requirement before QA can provide sign off for a particular test suite. Testing a module with positive, negative and random data set consumes time and effort. TestingWhiz lets a user test an

application with a different set of input values and ensures that the application works as expected. This is particularly useful while running quick regression cycles.

5.2.1 Setting up Data-Driven Test Script

Step 1: Click on Data table tab

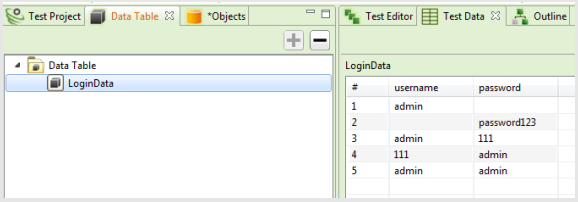
Step 2: Add a new data table

Step 3: Specify the name of the data table

Step 4: Add the fields and the default input values to perform the test

Step 5: Add the data to the fields created manually or by importing data from an Excel file using the import button

Step 6: Once the fields and the default input values are defined, create a Test Script in the Test Project Section with a Test command that fetches data from the Data table

Sample Data Driven Test Script to Log in www.testing-whiz.com with different usernames and passwords
Create Data in Data table.      **Step 1:** Add a new Data table in Data table tab. **Step 2:** Name the Data table as **'Login Data'**. **Step 3:** Add Fields as **'Username'** and **'Password'**. **Step 4:** Add different sets of usernames and passwords as data, manually or by importing an Excel file.

Create Test Script to Call/Fetch data created in the Data table.

Step 5: Add a Test Case and select Test command as **'Open Page'** and add Value as '**http://testing-whiz.com/demoscripts/place-holder.html**'.

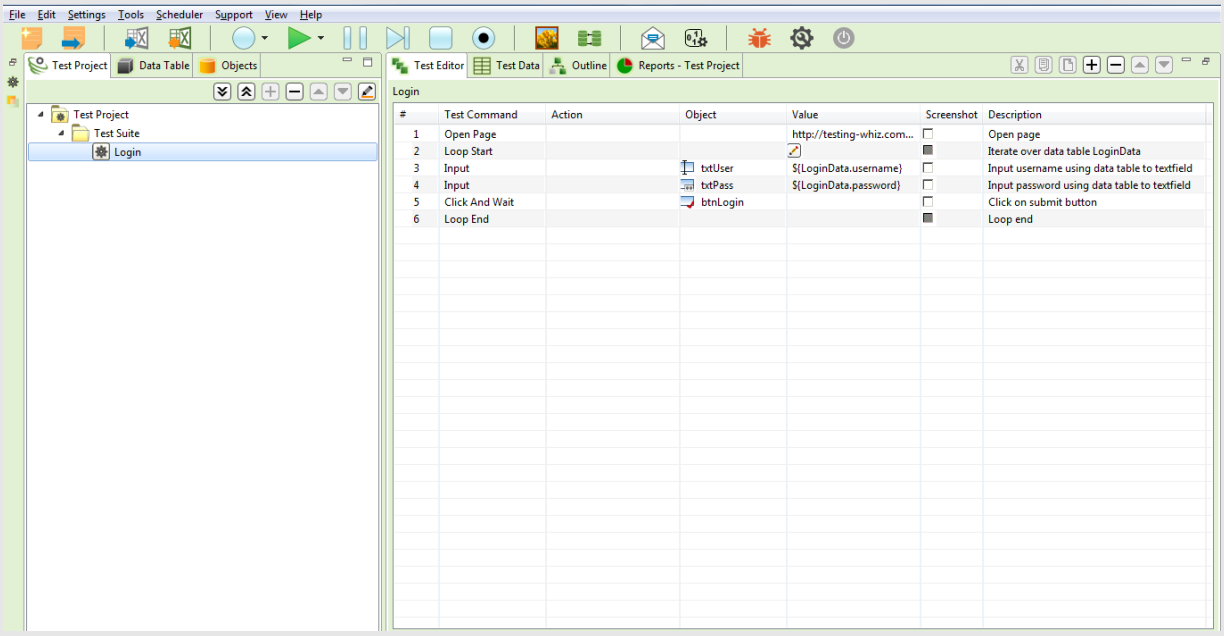
Step 6: Add next step and select **'Loop Start' or 'Loop End' or 'Loop Continue' or 'Loop Break'** as Test command to loop the process of login with different usernames and passwords. Add Value as **'Login Data'** (from Data table) to fetch all the usernames and passwords stored in the Data table.

Step 7: Select Test command as **'Input'**, add Object as '**txtPass'** and insert **Value** as '**${LoginData.username}**'.

Step 8: Select Test command as **'Input'**, add Object as '**txtPass'** and insert **Value** as '**${LoginData.password}**'.

Step 9: Select Test command as **'Click And Wait'** and Object as **'btnLogin'** (To click and wait for login after each combination of usernames and passwords).

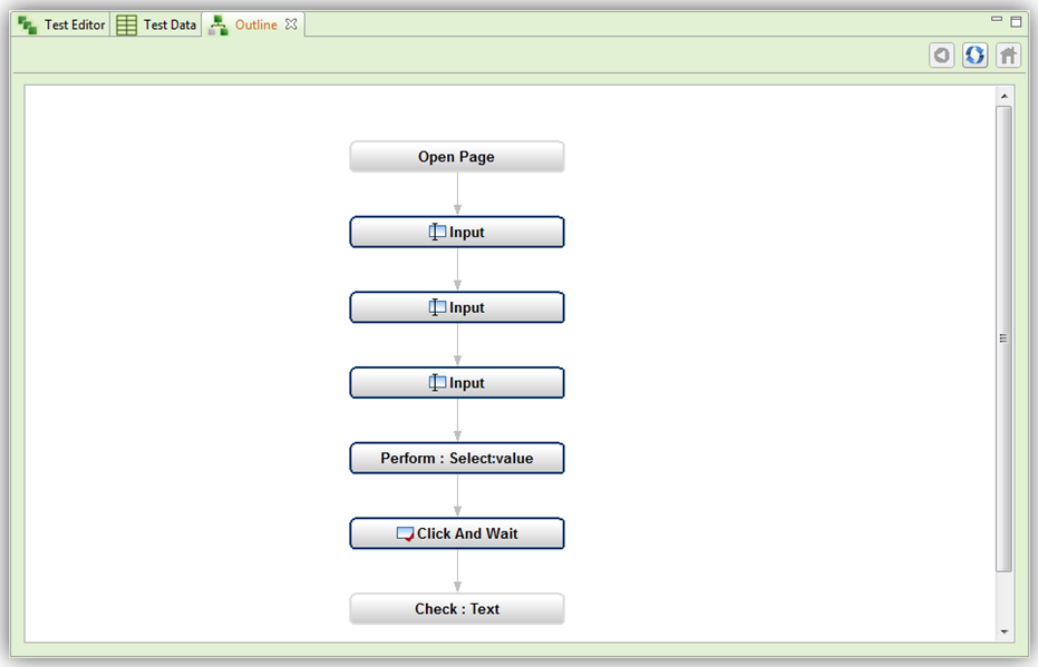
Step 10: Select Test command as **'Loop End'** (To try logging in with various usernames and passwords until the login is successful).



6 IMPORTANT FUNCTIONS OF TESTINGWHIZ

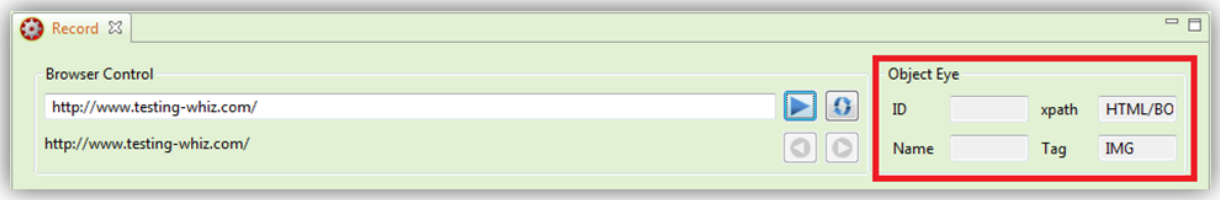
6.1 Data Flow Diagram View/Outline View

TestingWhiz provides a unique representation of the Test Step(s) with **Outline** which encapsulates the complexity of the test case by displaying all the steps of a test case through a flow chart. Click on any of the steps opens its sub steps and its respective flow chart.



6.2 Object Eye

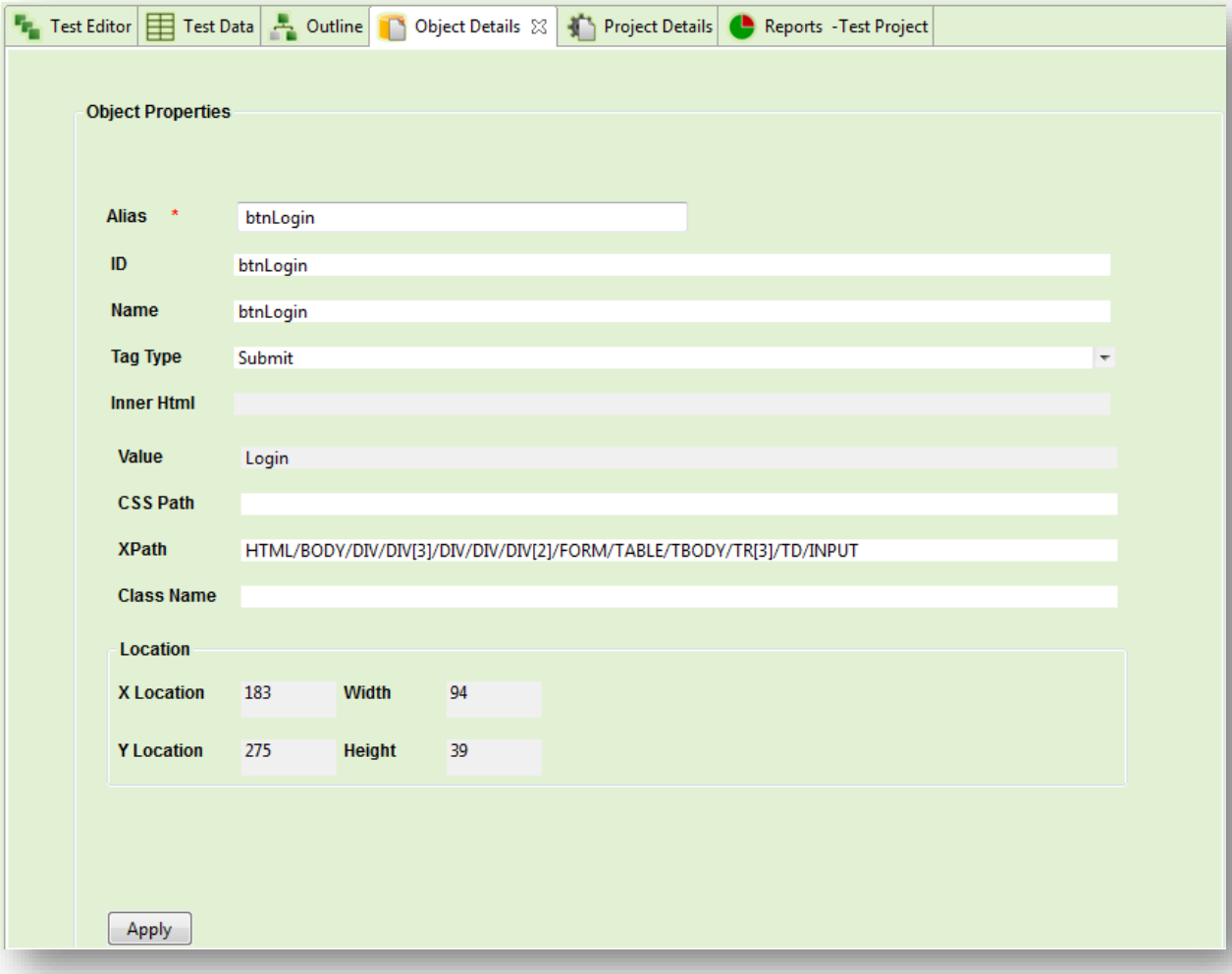
TestingWhiz features Object Eye which allows a user to view the properties of an object during test recording process. Details like the ID, Name, XPath and Tag of the object are displayed based on the selection at the time of recording the test steps.



6.3 Object Repository

6.3.1 Object Properties

Object Repository feature tracks and stores the objects and properties of the Test Script(s) that have been captured at the time of recording test steps. It keeps track of the object as per the modules followed by a user at the time of recording the test steps. The objects are displayed URL wise.



Following are the attributes of an Object that are displayed in the Object Repository, according to URLs:

Alias	Alias by default displays the **Object Id** of the selected object which is utilized in the test step grid. [**Note**: *A user can change the Alias name. It is advisable to provide user-friendly names to test scripts for easy maintenance.*]
Id	Displays the **Id** of the selected object. [**Note**: *A user can change the Id. It is advisable to provide user-friendly names to test scripts for easy maintenance.*]
Name	Displays the **Name** of the selected object. [**Note**: *A user can change the Name. It is advisable to provide user-friendly names to test scripts for easy maintenance.*]
Tag Type	Displays the **Name of the Control** that has been selected during the test case execution.
Inner HTML	Displays the **Inner Html** of the page.
Value	Displays the **Text Inserted** in the selected object.
CSS Path	Displays the location of the object through CSS structure.
XPath	Displays the **XPath** of the object through XML structure.
Class Name	Displays **Class Name** of the selected object. [**Note**: *A user can change the class name. It is advisable to provide user-friendly names to test scripts for easy maintenance.*]
Location	Displays X and Y **Locations** of the selected object along with its width and height.

[Note: *User can delete an Object from Repository using right click option.*]

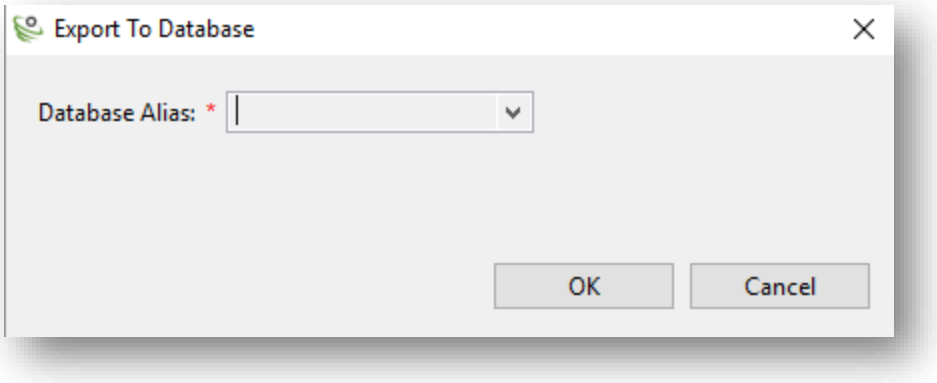
[**Note:** *The system displays only those objects that have been used by a user.*]

6.3.2 Exporting Objects to the Database

Users can collaborate between themselves by sharing their object repositories to a common database and utilize the frequently used objects between themselves.

Steps to export objects from TestingWhiz to database:

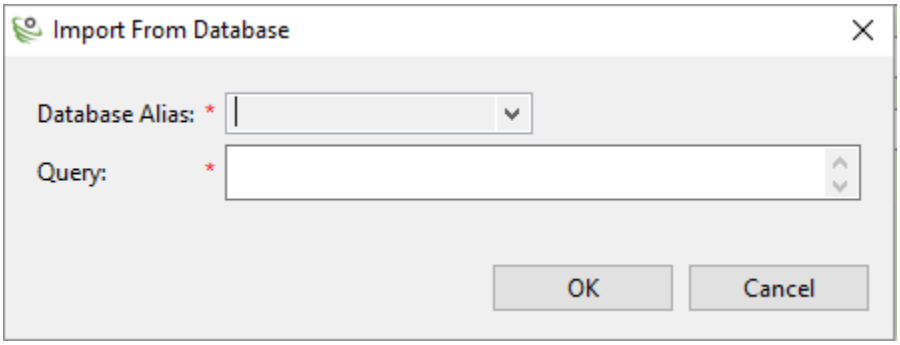
1. Navigate to Object Repository panel.
2. Click on  icon to open up **Export to Database** window as follows:



3. Select the **Database Alias** from the dropdown.
4. Click on **OK** button to export all the objects to the selected database from the dropdown.
5. All the objects are stored in the **ObjectRepository** named table in the respective database.

6.3.3 Importing Objects from a Database

1. Navigate to Object Repository panel.
2. Click on  icon to open up **Export to Database** window as follows:



3. Select Database alias, write a query and click **OK** button to import objects from the specified database.

6.4 Methods

TestingWhiz provides a feature of grouping functions as Methods so that a user can use/execute that method in the Test Case/Script multiple times. This reduces the code size, saves time and increases maintainability of the Test Scripts.

The Methods function is highly useful if multiple Test Scripts include the same set of functions, or functions that are frequently used.

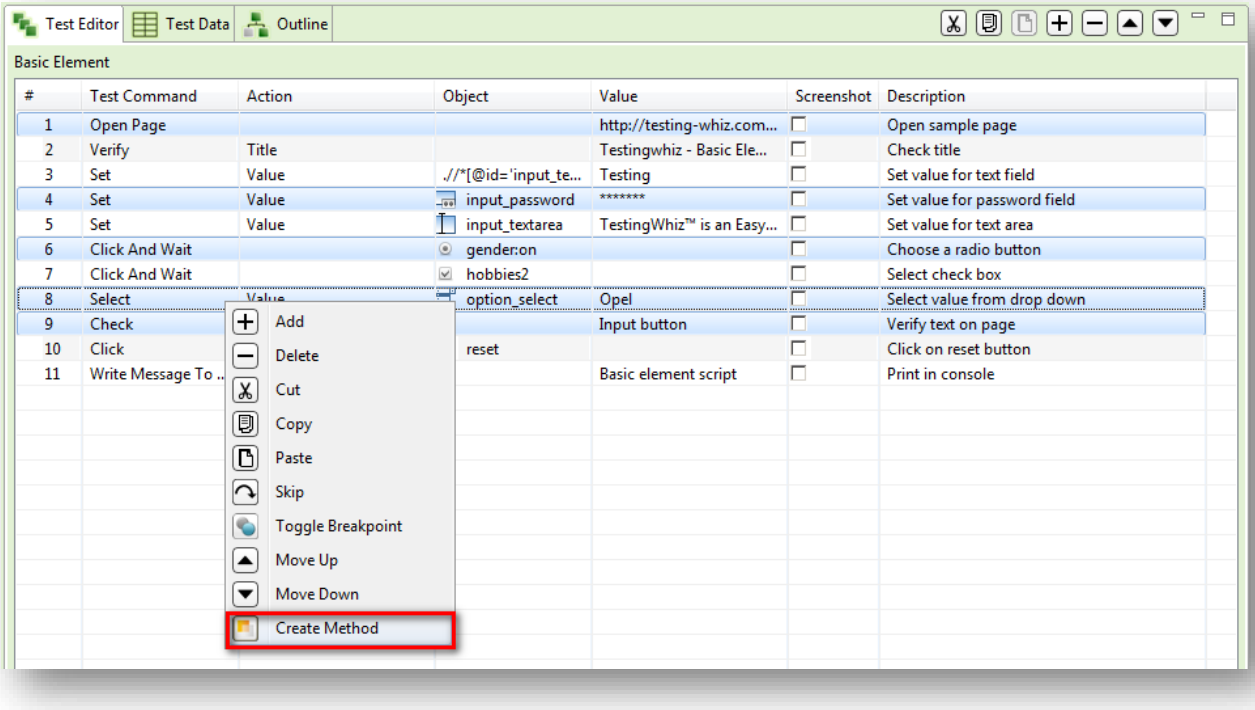
6.4.1 Process of Creating and Calling Method

Here's a complete process of creating and calling Methods:

Let us take an example of the process of Login into TestingWhiz application.

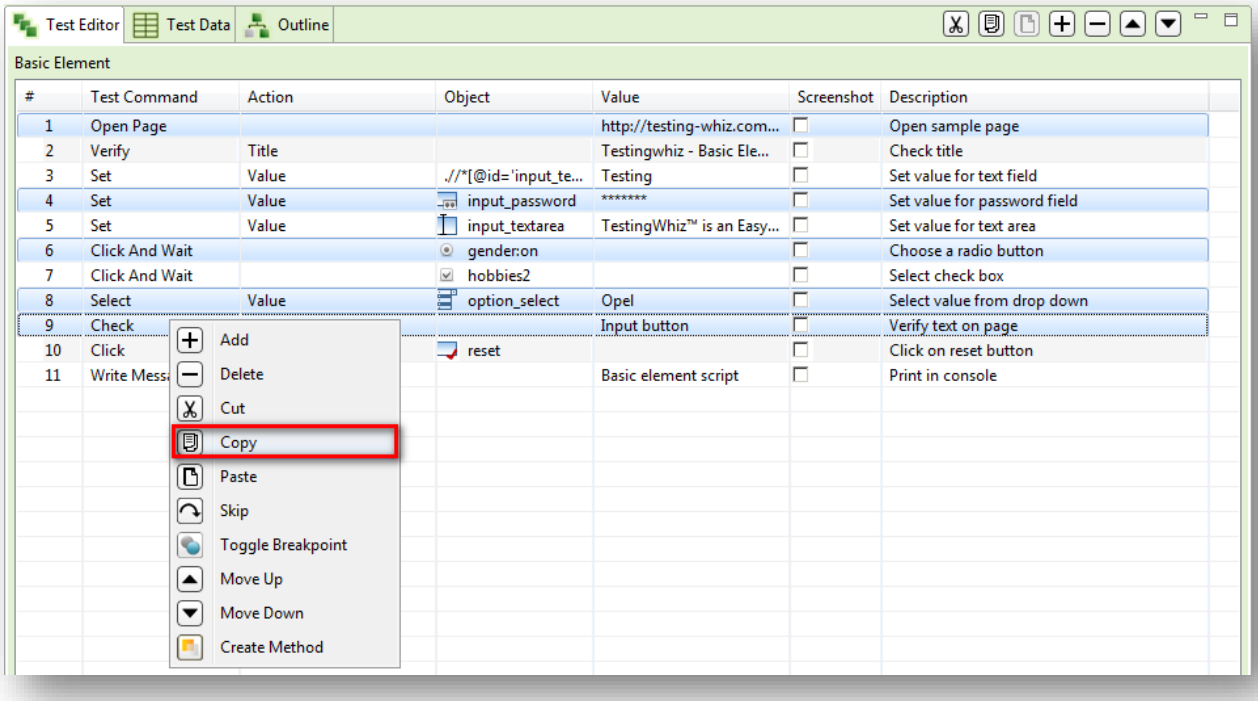
Step 1: Select the Test Steps from the existing Test Scripts to group as a single Method.

Step 2: Right click and select **Create Method**  from the context menu.



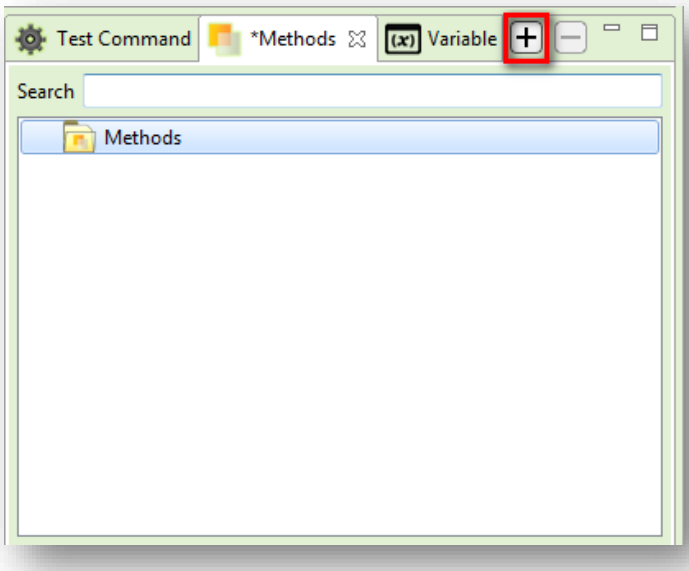
OR

Step 2: Press **Copy**  to copy the select Test Steps from the Test Editor

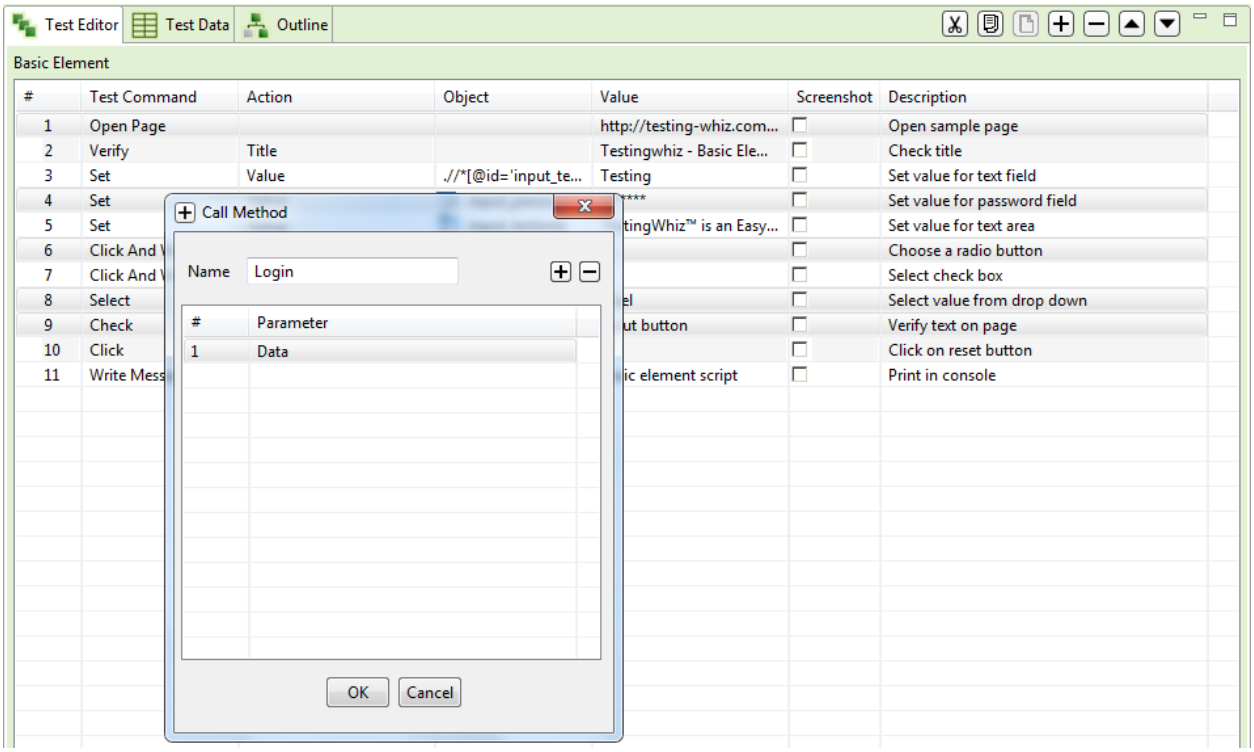


OR

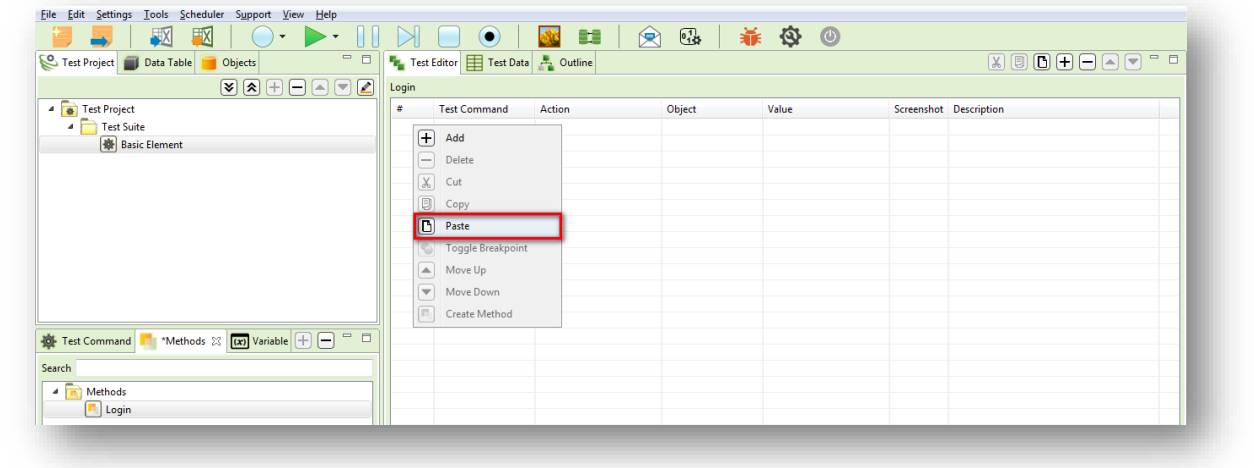
Step 2: Go to Methods Tab and click  to add a new Method.



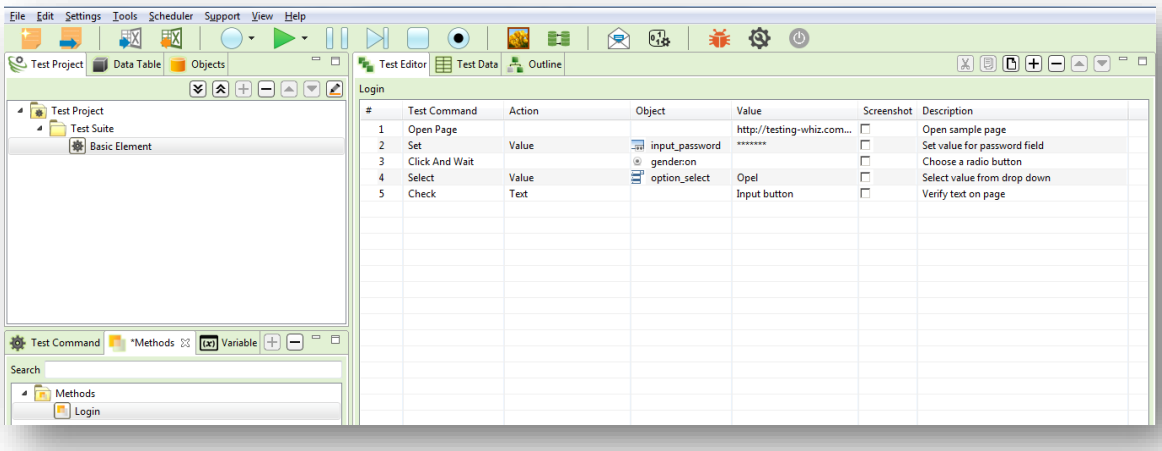
Step 3: Enter the **Name of the Method** - For e.g., 'Login' and the **Parameter Value** – For e.g., 'Data'.



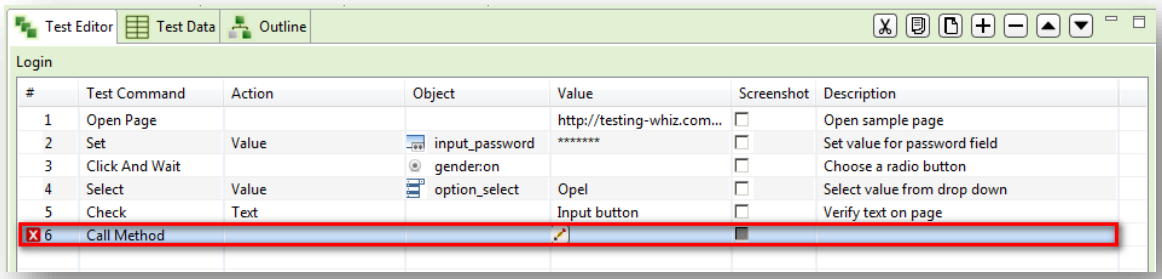
Step 5: Paste the Test Steps to newly created method. (In case a user has copied the Test Cases)



Selected Test Steps will appear under the newly created Method.

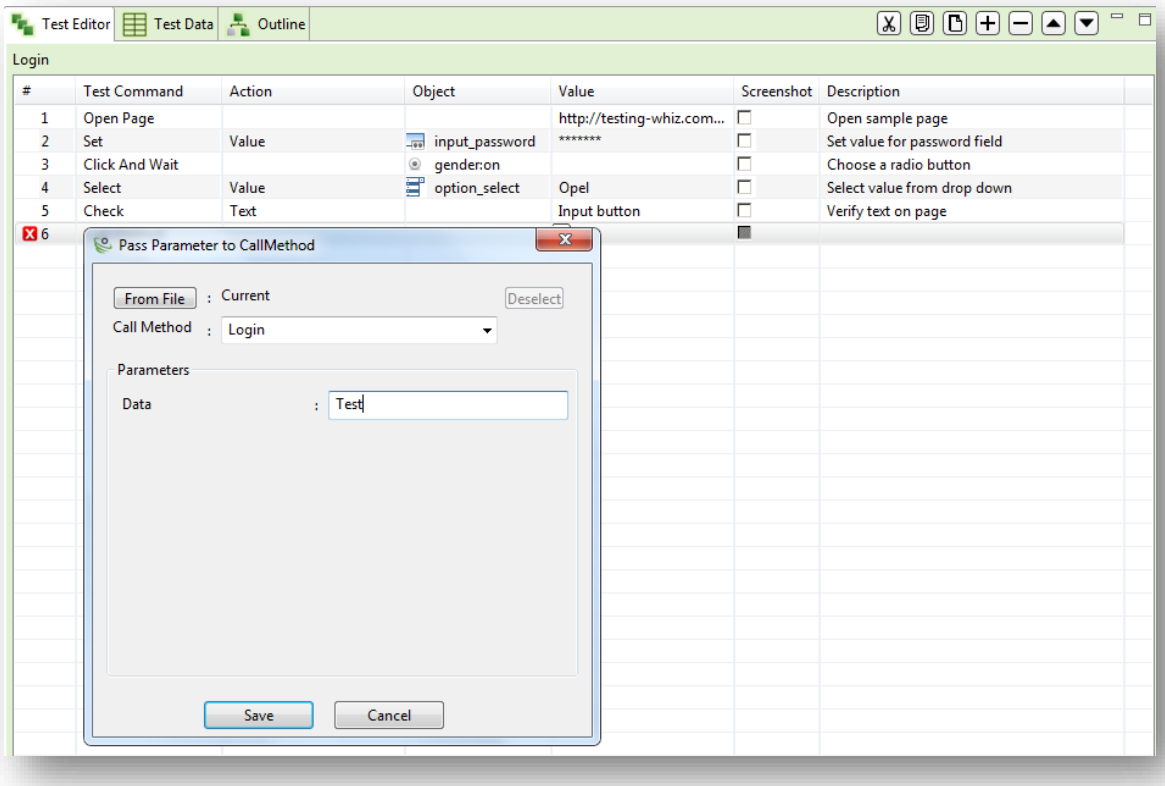


Step 7: Add a new Test Step and select Test command as '**Call Method**'.



Step 8: Click **Value** in the corresponding cell and select Call Method from the drop-down of available methods – in this case select '**Login**'.

Step 9: Enter Parameter Value and click **Save**.



Step 10: Add further steps to the Script if necessary.

Step 11: Execute the Test Script.

[**Note:** *User can add any number of Methods.*]
[**Note:** *Selecting a Call Method step and pressing F3 would open up the respective method.*]
[**Note:** *A user can also call method from any other .twiz file.*]

6.5 Image Comparison

TestingWhiz helps users to compare two images and record the difference at pixel level. TestingWhiz automatically converts a particular webpage into an image to carry out the comparison.

User can perform pixel by pixel Image Comparison in the following ways:

1. Image to Image Comparison
2. Image to URL Comparison
3. URL to URL Comparison

[**Note**: *TestingWhiz only supports Image File formats – 'png, jpg, bmp and gif'.*]

[**Note**: *TestingWhiz only supports Image comparison of same file extension.*]

6.5.1 How Image Comparison Works

Image Comparison functionality allows a user to capture images from the specified path in test commands and resize the captured images to *600x600*. Targeted images shows the difference in (%) value. Image comparison functionality will only compare the color between two source files.

Here's a step by step process of comparing images of Google's Different domain pages with google.com

Step 1: Create a **New Test Case** under a **Test Suite**

Step 2: Add a New Test Step as **Loop Start** in the Test Editor tab or Double click on **Loop Start** command under Test Commands tab.

Step 3: Enter **Google** as a Value

Step 4: Add a New Test Step as **Check** in the Test Editor tab or Double click on **Check** command under Test Commands tab.

Step 5: Enter **Image** as an Action

Step 6: Click  to enter Source 1, Source 2 and Tolerance details in Compare Image Test Command dialog box.

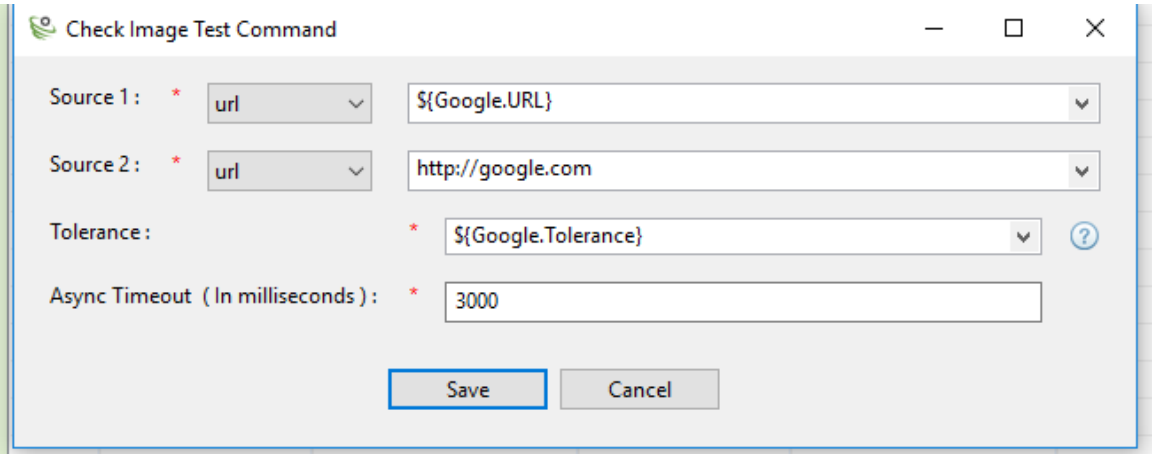
Step 7: Enter '**${Google.URL}**' in **Source 1** box.

Step 8: Enter 'http://google.com' in **Source 2** box.

Step 9: Set up **Tolerance level** which user want to check the similarities in Tolerance box.

[**Note**: *Tolerance level will be in (%) value.*]

Step 10: **Async Timeout** field allows user to compare the images of the websites which don't have synchronous loading time. Default value of this field will be 3000 ms (milliseconds).



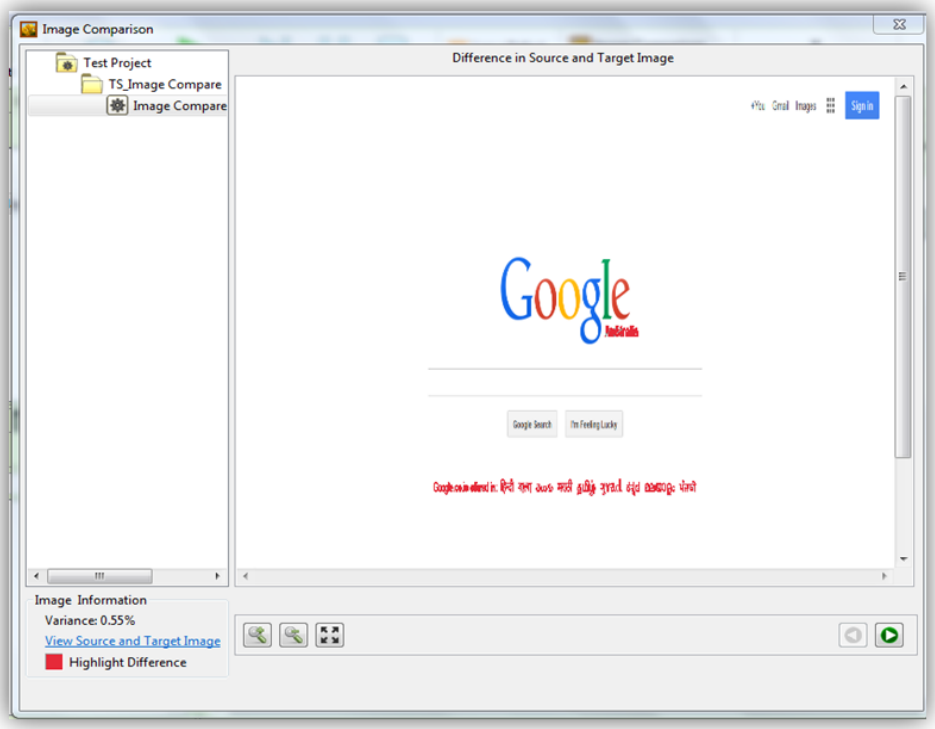
Step 10: Click on **Save** button to save the details.

Step 11: End the Test Case by adding a Test Step as **Loop End** or Double click on **Loop End** command under Test Commands tab.

Step 12: Execute Test Script in any browser.

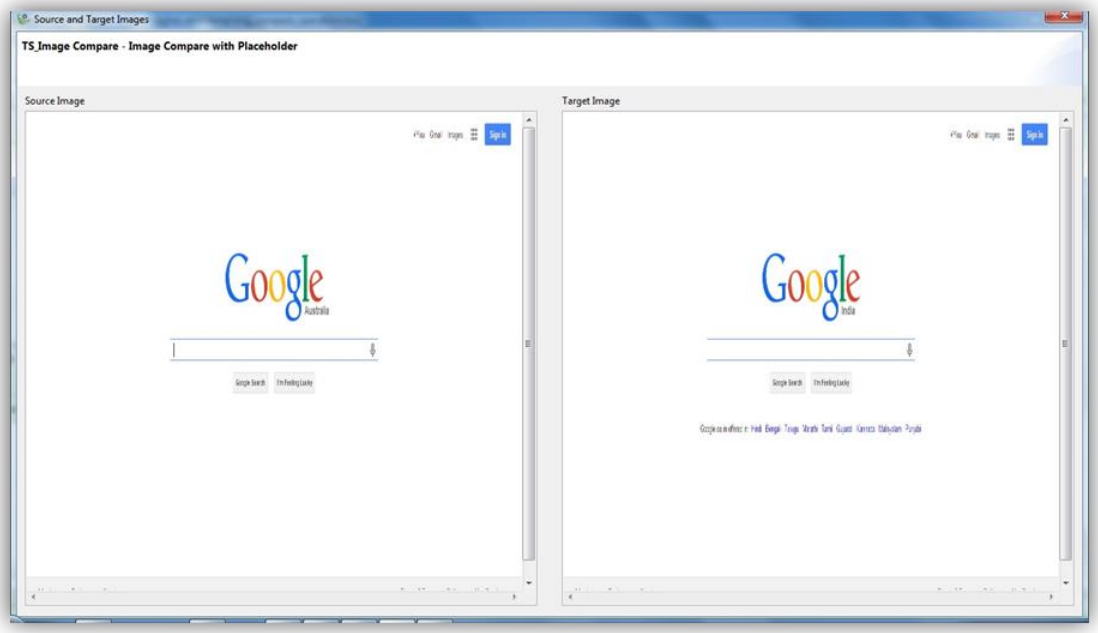
Step 13: On completion of execution user can view detailed logs in generated report.

Step 14: Click  to view the Image Comparison report.



Here a user can view **0.55%** variance between the two urls as shown in the report.

Step 15: Click "**View Source and Target Image**" to view the compared images.



Step 16: User can also view Target images in whiz folder. "**C:\Users\testingwhiz\.whiz**"

6.6 Fork

TestingWhiz offers a functionality of executing your recorded scripts in single machine or multiple machines and multiple browsers simultaneously. This feature is called Fork.

Forking can be used in two ways as follows:

A. Test Case Forking

B. Test Step Forking

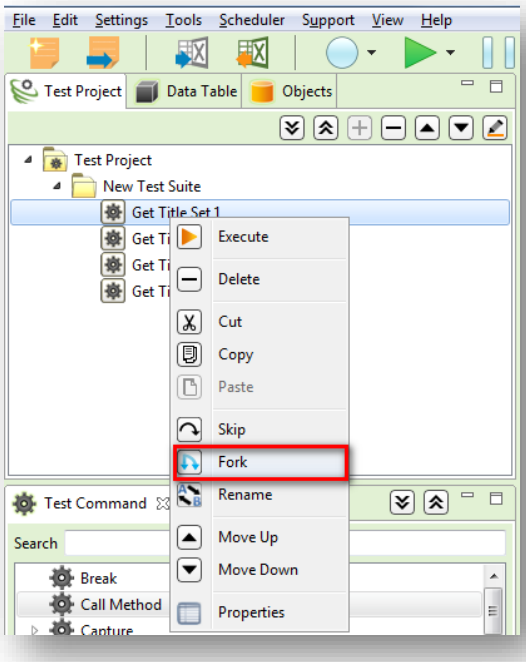
Both these ways of forking works on a single machine and also can be achieved on multiple machines if a Hub server URL is provided in the configurations Window and Nodes are connected.

6.6.1 Test Case Forking

If a user wants Test Cases to be executed in a new instance of a browser, Forking Test Cases can be used.

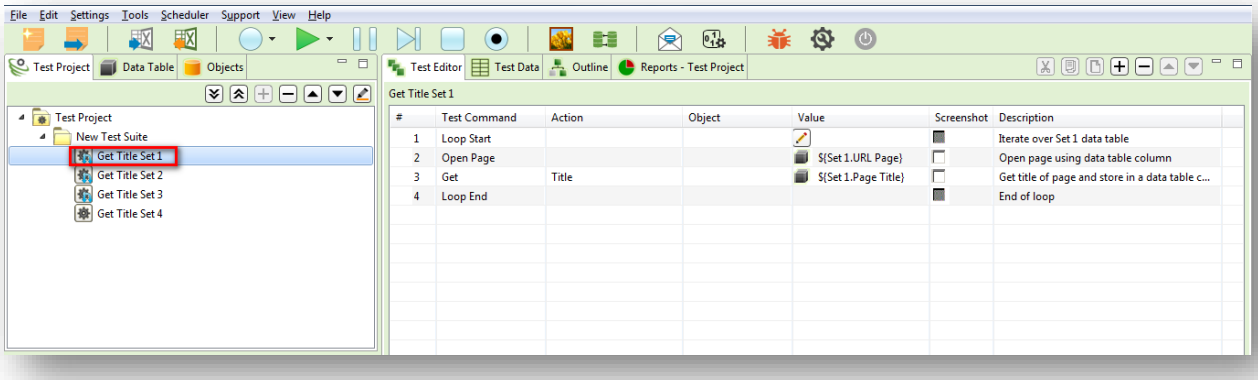
Here's a complete process of Forking a Test Case

Step 1: Select a Test Case from the existing Test Scripts and right click on it.

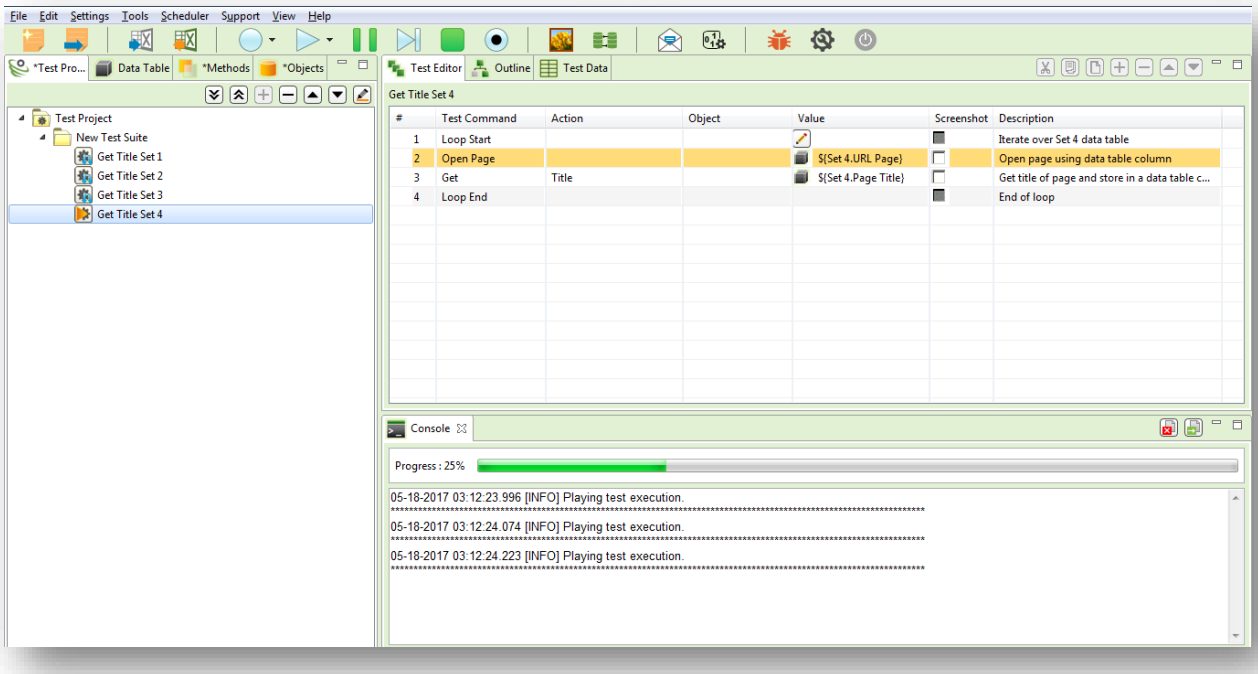

File Edit Settings Tools Scheduler Support View Help
Test Project
Data Table
Objects
Test Project
New Test Suite
Get Title Set 1
Execute
Delete
Cut
Copy
Paste
Skip
Fork
Rename
Move Up
Move Down
Properties
Test Command
Search
Break
Call Method

Step 2: Click 

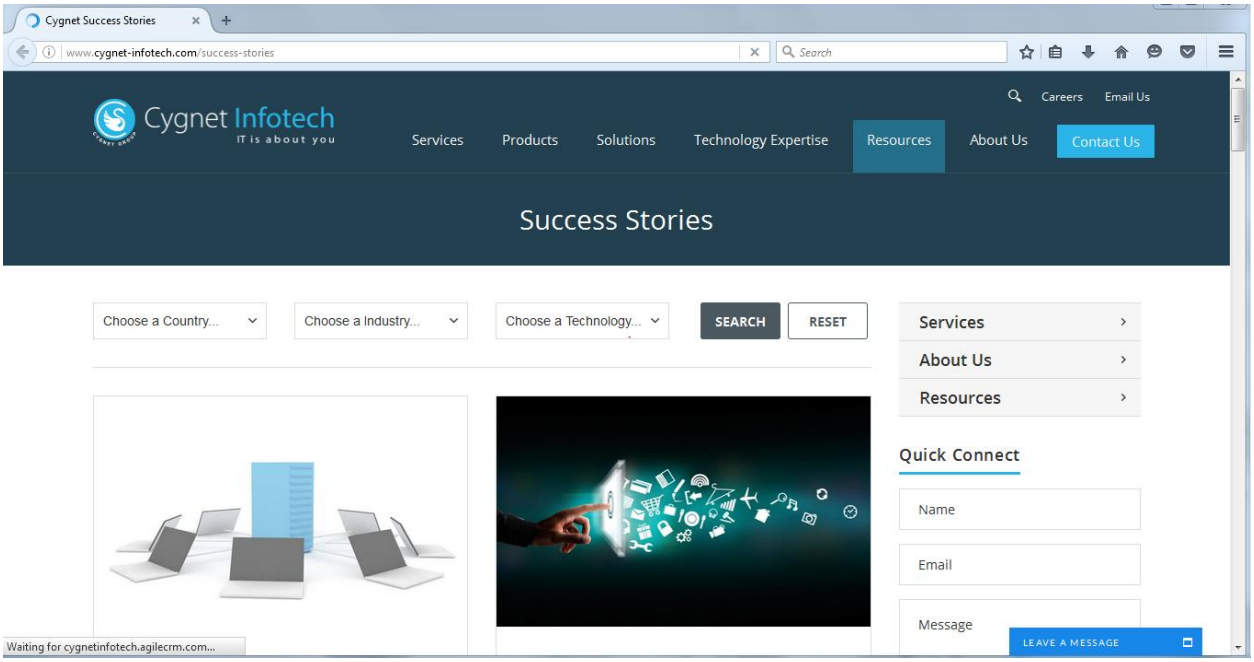
to Fork the selected Test Case.

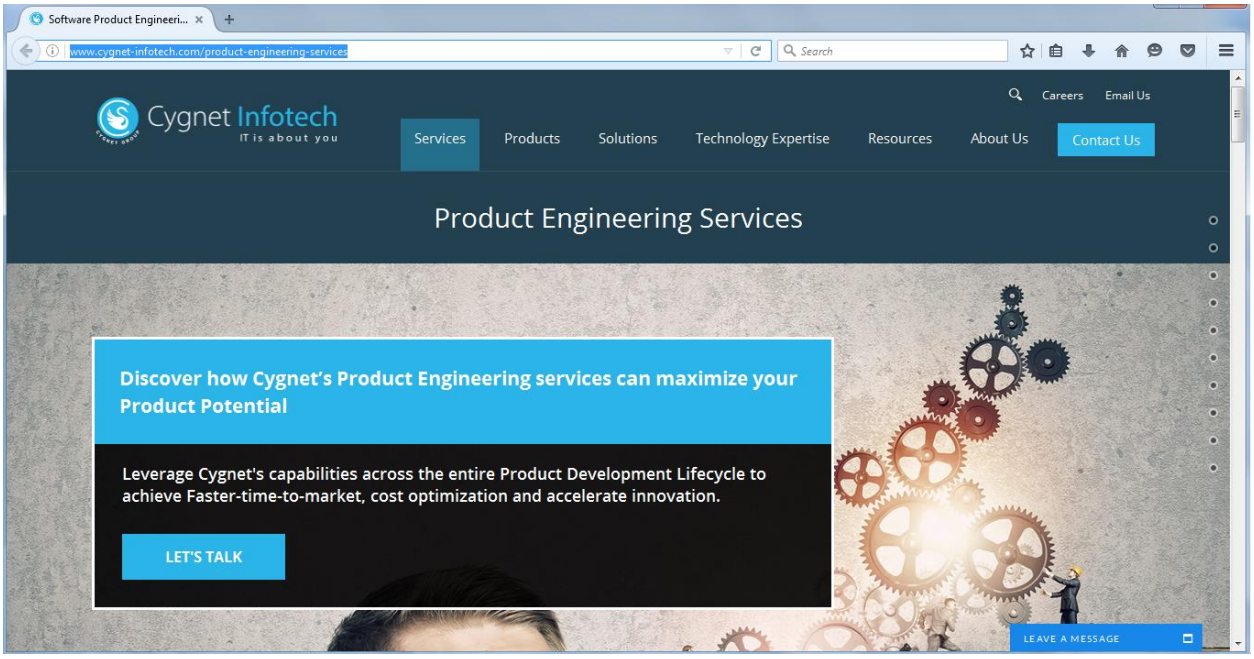


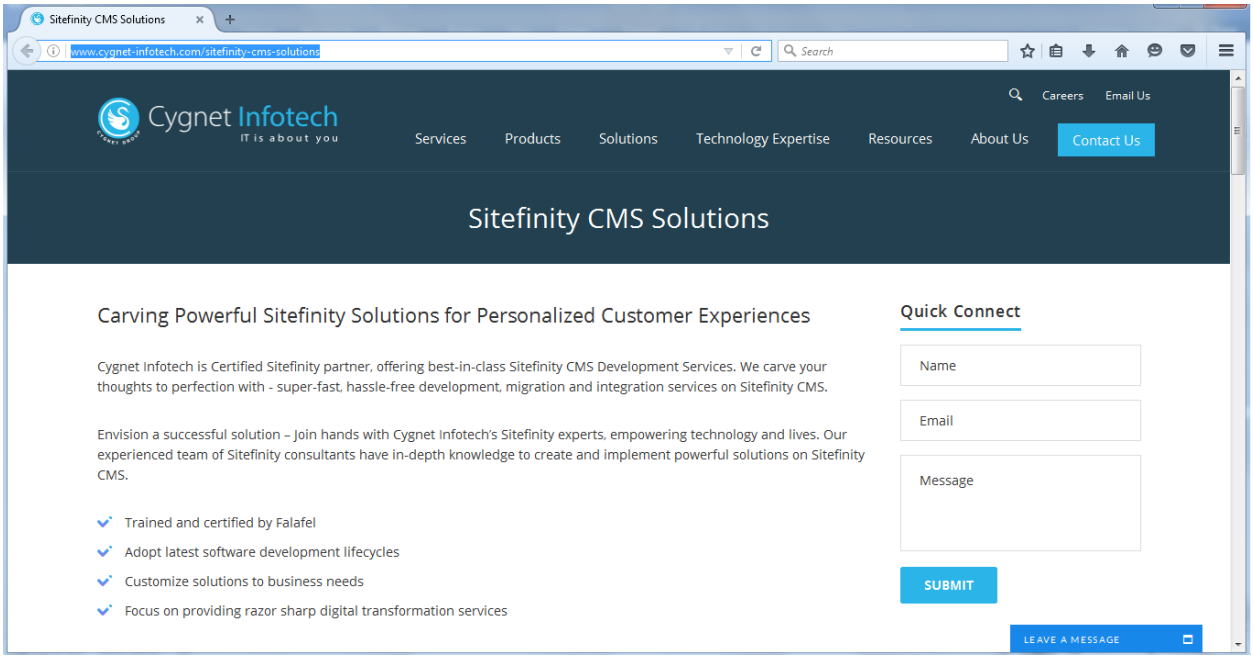
Step 3: Click 
to start execution of Test Script in any browser.

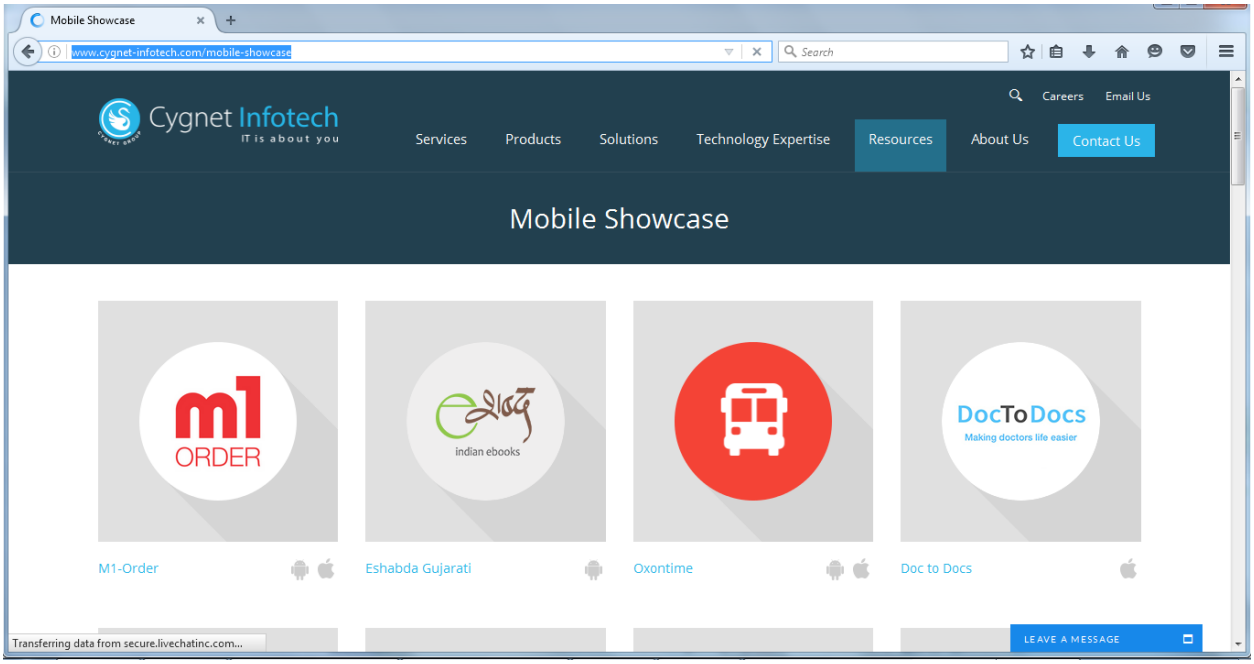


Test Script will be executed in 4 instances of the selected browser from the same machine.






Sitefinity CMS Solutions
www.cygnet-infotech.com/sitefinity-cms-solutions
Search
Cygnet Infotech
IT is about you
Careers
Email Us
Services
Products
Solutions
Technology Expertise
Resources
About Us
Contact Us
Sitefinity CMS Solutions
Carving Powerful Sitefinity Solutions for Personalized Customer Experiences
Cygnet Infotech is Certified Sitefinity partner, offering best-in-class Sitefinity CMS Development Services. We carve your thoughts to perfection with - super-fast, hassle-free development, migration and integration services on Sitefinity CMS.
Envision a successful solution – Join hands with Cygnet Infotech's Sitefinity experts, empowering technology and lives. Our experienced team of Sitefinity consultants have in-depth knowledge to create and implement powerful solutions on Sitefinity CMS.
Trained and certified by Falafel
Adopt latest software development lifecycles
Customize solutions to business needs
Focus on providing razor sharp digital transformation services
Quick Connect
Name
Email
Message
SUBMIT
LEAVE A MESSAGE

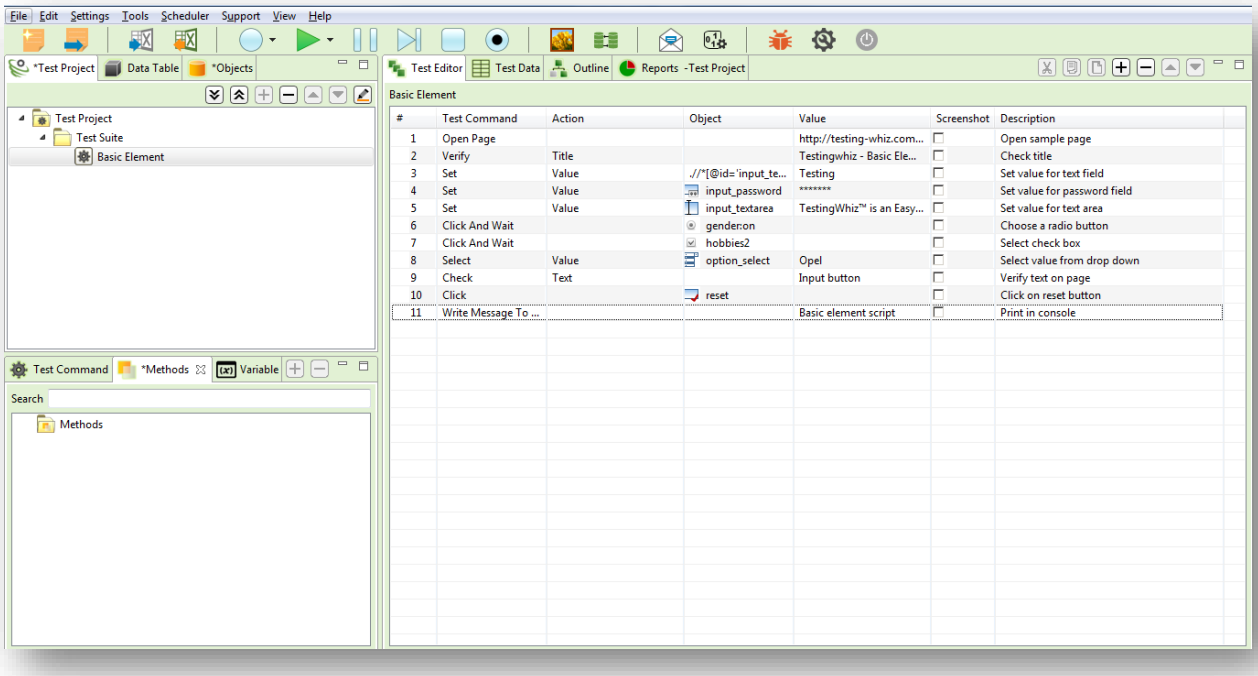

Mobile Showcase
www.cygnet-infotech.com/mobile-showcase
Search
Cygnet Infotech
IT is about you
Careers
Email Us
Services
Products
Solutions
Technology Expertise
Resources
About Us
Contact Us
Mobile Showcase
m1 ORDER
indian ebooks
DocToDocs
Making doctors life easier
M1-Order
Eshabda Gujarati
Oxontime
Doc to Docs
LEAVE A MESSAGE
Transferring data from secure.livechatinc.com...

6.6.2 Test Step Forking

User can Fork Test Steps to execute them in different instances of a browser by using **Fork Start** and **Fork End** command.

Here's the complete process of Forking Test Steps:

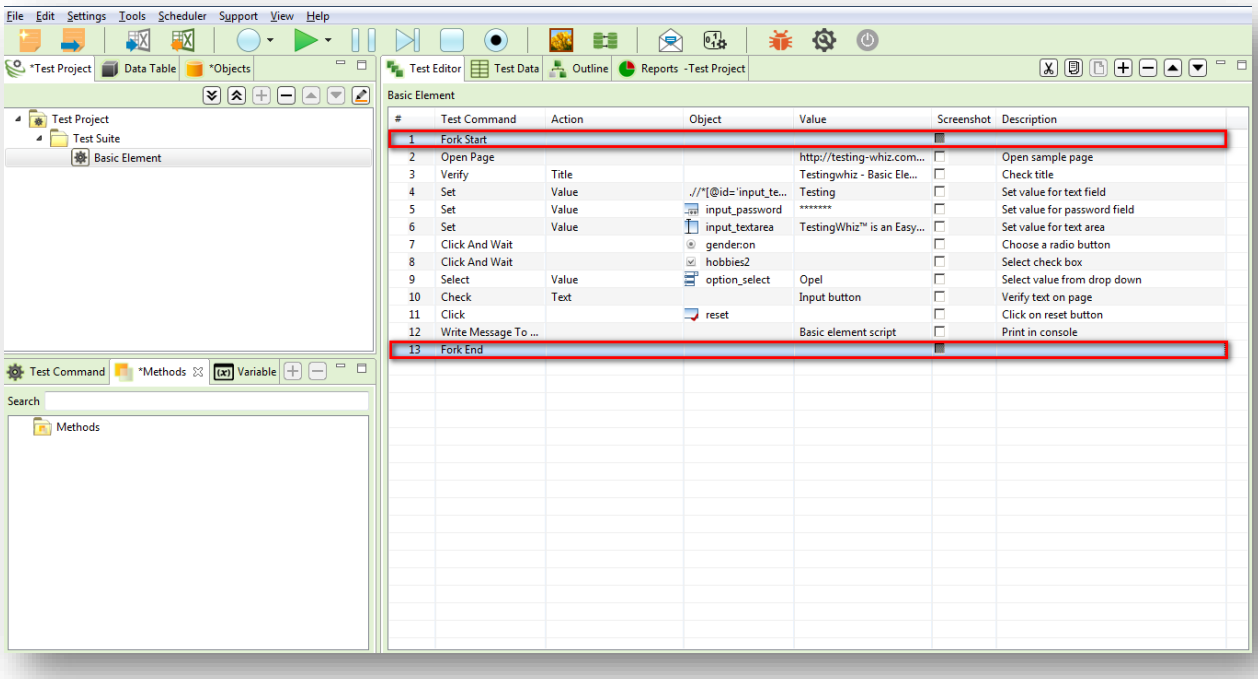
Step 1: Open any existing Test Scripts in TestingWhiz.



Step 2: Enter **Fork Start** and **Fork End** Command at the start and at the end of the Test Script.

OR

Drag & Drop **Fork Start** and **Fork End** Test Commands from the Test Command search box.



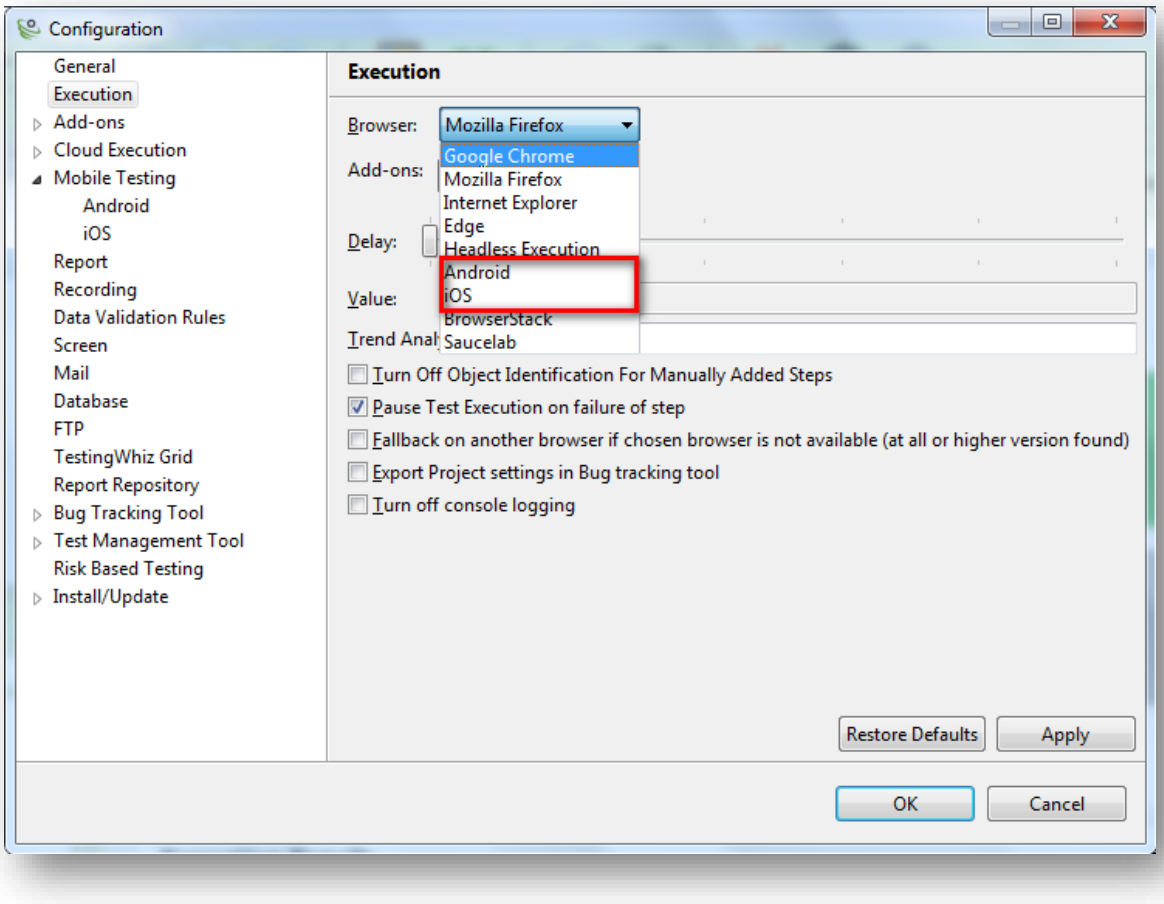
Step 3: Click  to start execution of Test Script in any browser.

6.7 Mobile Test Execution

TestingWhiz offers a functionality of executing Test Scripts on Android and iOS devices.

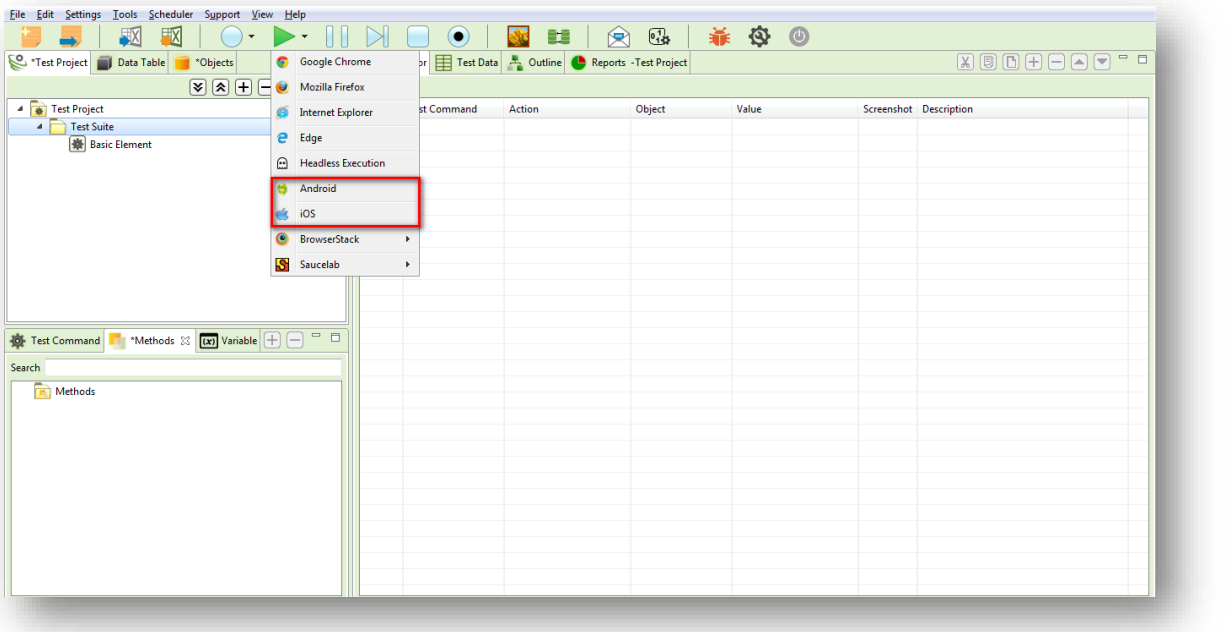
Here's the complete process of performing Mobile Test Execution

Step 1: Select a default browser as '**Android**' from the Configuration menu under Settings.



OR

Select **'Android'** or **'iOS'** from the execution drop-down, while executing the Test Script.



6.7.1 Android Environment Setup for Mobile Test Execution

I. Setup on Android Device

Step 1: Go to Settings of the Android Device

Step 2: Enable Developer Options by tapping Build Number for 7 times under About Phone menu

Step 3: Enable USB Debugging and Stay Awake option under the Developer Options menu

II. Setup on Desktop PC

Step 1: Install Android SDK

Step 2: Install Appium Server

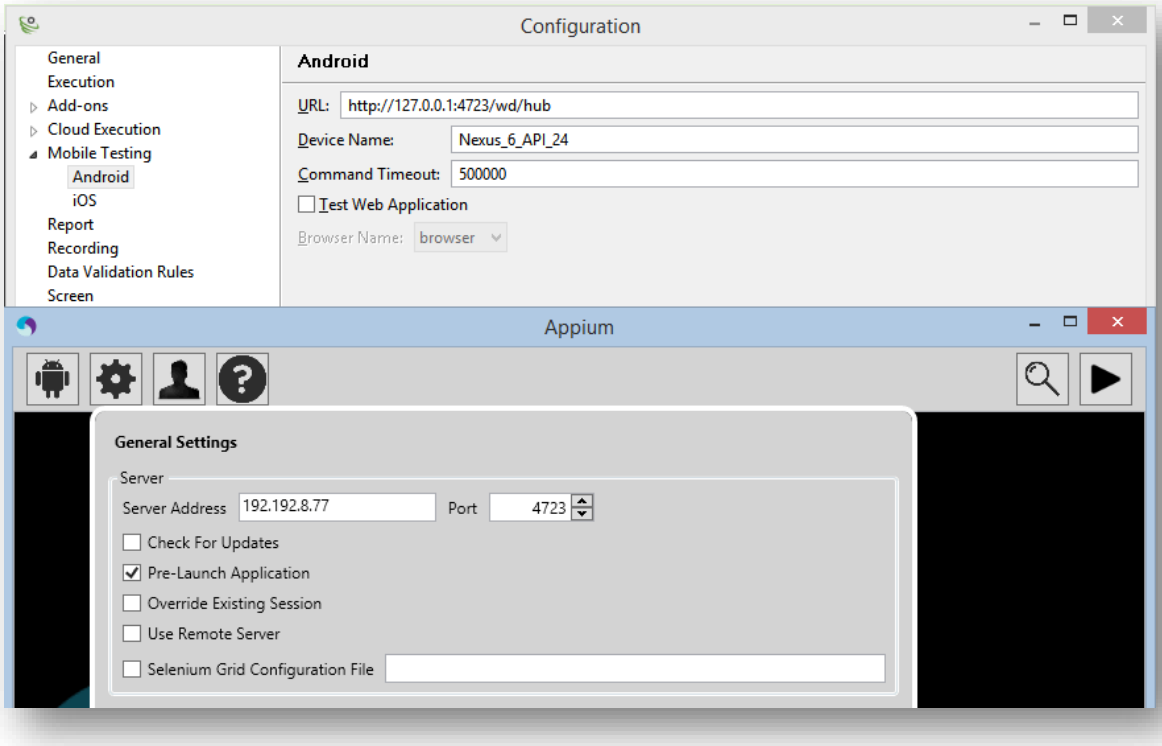
6.7.1.1 Process to Execute Test Cases on Android Simulator

Step 1: Create and start a new device by using AVD Manager.

Step 2: Start the Appium Server.

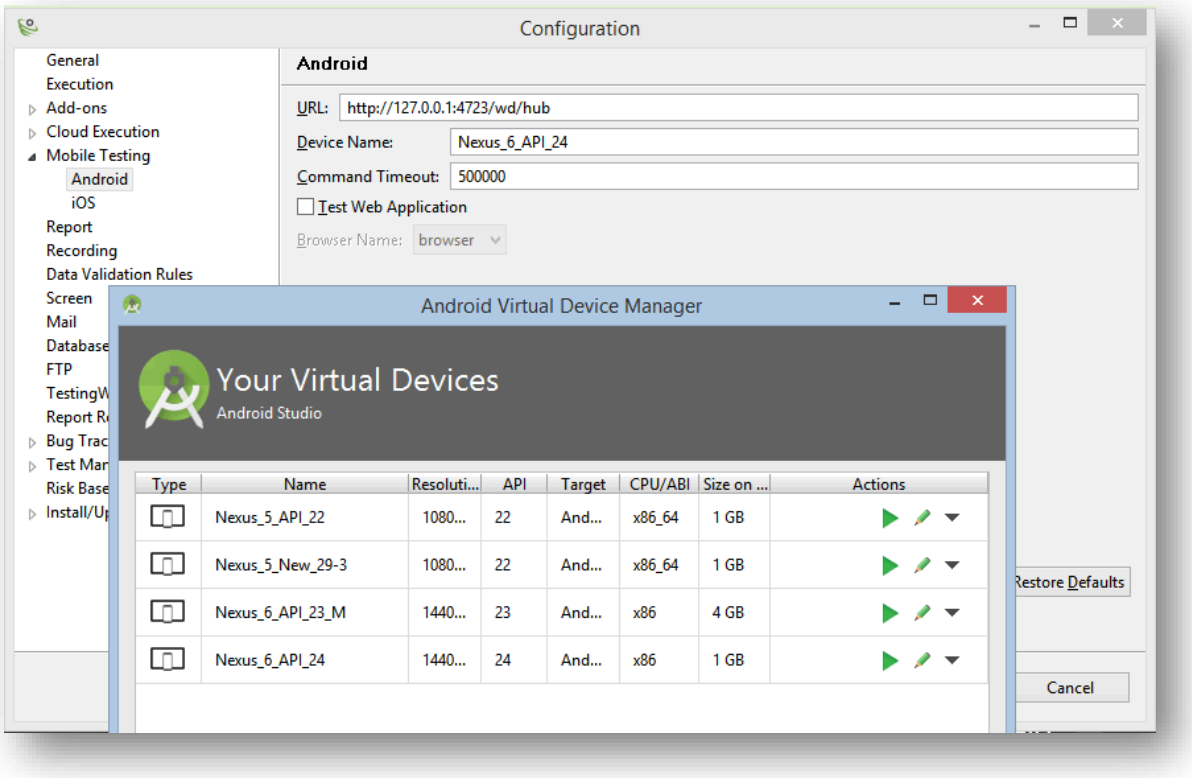
Step 3: Copy the Appium Server Address and Port Number

Step 4: Paste the copied Server Address and Port Number into Server URL field – TestingWhiz Settings>Configurations>Mobile Web Testing>Android



Step 5: Copy the Android Simulator Name.

Step 6: Paste the copied Simulator Name into Device Name field - TestingWhiz Settings>Configurations>Mobile Web Testing>Android



Step 7: Select the Browser from the Browser Name drop-down.

Step 8: Apply and Test the Connection.

Step 9: Close the Configuration Window.

Step 10: Execute the Test Case on Android Simulator.

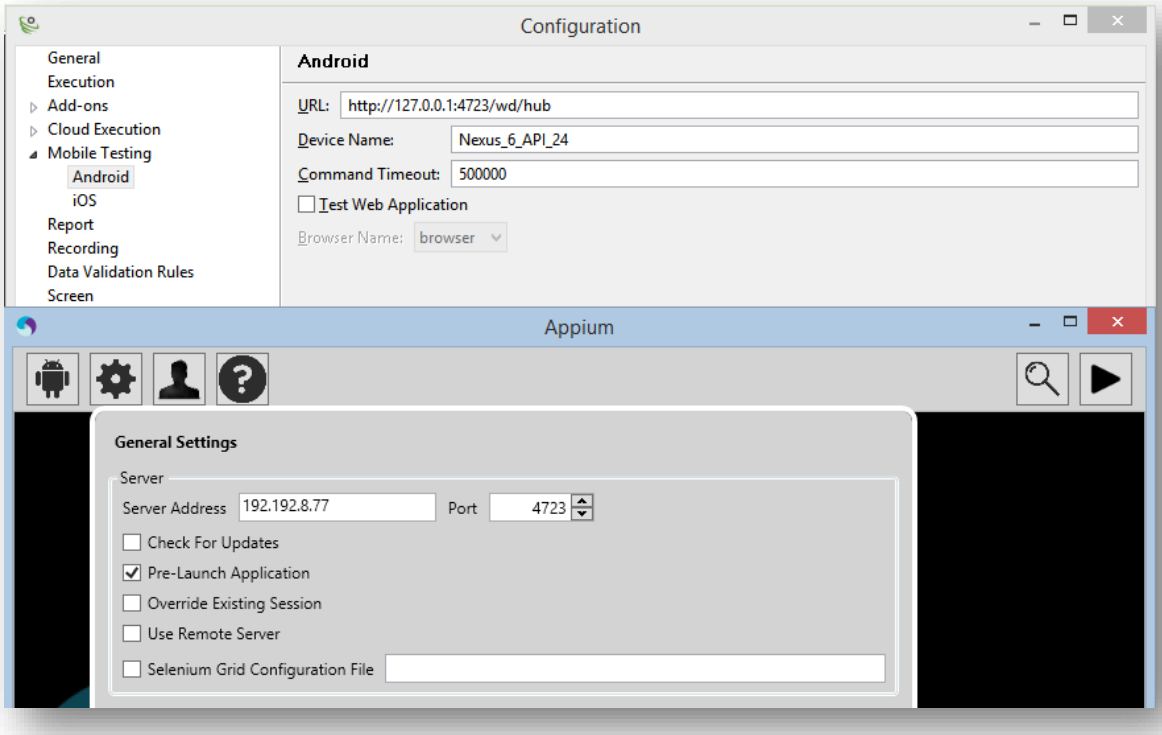
6.7.1.2 Process to Execute Test Cases on a Real Android Device

Step 1: Connect a real Android device with the Desktop PC using a cable.

Step 2: Start the Appium Server.

Step 3: Copy the Appium Server Address and Port Number.

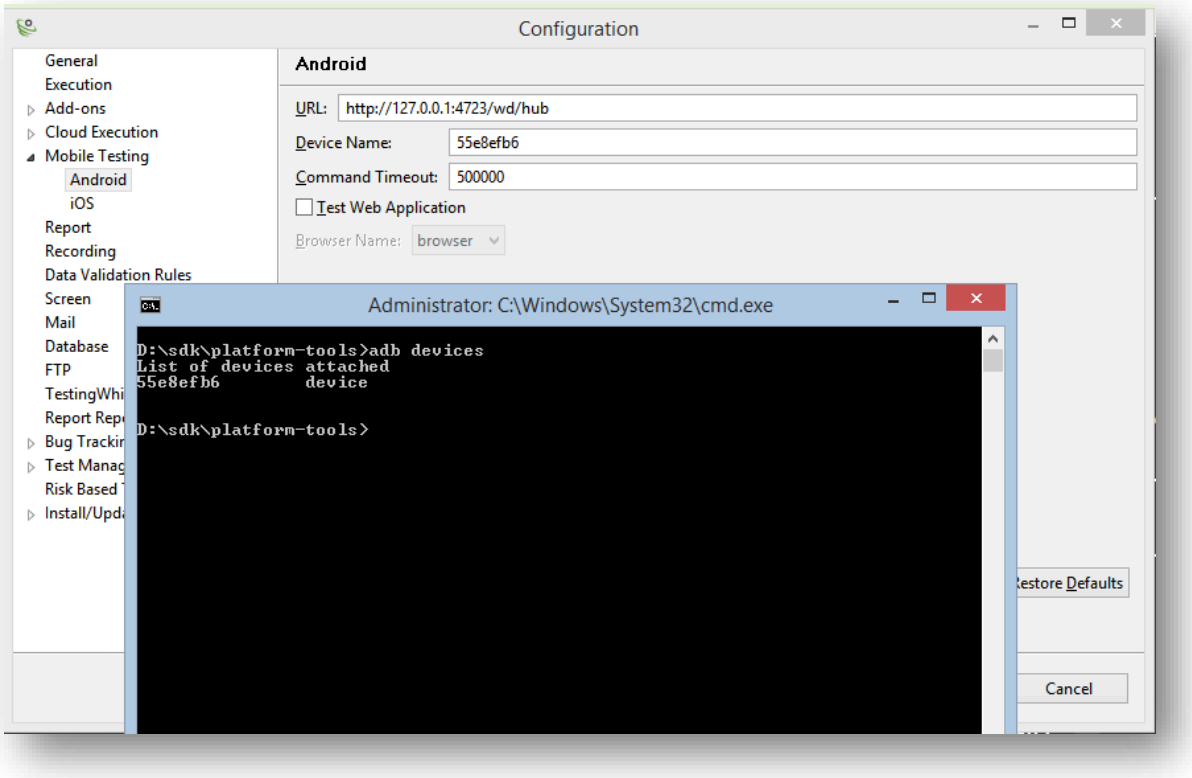
Step 4: Paste the copied Server Address and Port Number into Server URL field - TestingWhiz Settings>Configurations>Mobile Web Testing>Android.



Step 5: Press Shift + Right Click to open command prompt under Platform-tools folder of Android SDK.

Step 6: Run **'adb devices'** command and copy the **Device ID**.

Step 7: Paste the Device ID into Device Name field - **TestingWhiz Settings > Configurations > Mobile Web Testing > Android**.



Step 8: Select the Browser from the Browser Name drop-down.

Step 9: Apply and Test the Connection.

Step 10: Close the Configuration Window.

Step 11: Execute the Test Case on a real Android Device.

6.7.2 iPhone Environment Setup for Mobile Test Execution

I. Setup on iOS Device

Step 1: Go to Settings of the iOS Device

Step 2: Select Safari

Step 3: Select Web Inspector under Advanced menu

II. Setup on MAC System

Step 1: Install Apple XCode

Step 2: Install Appium (1.3.4) Server

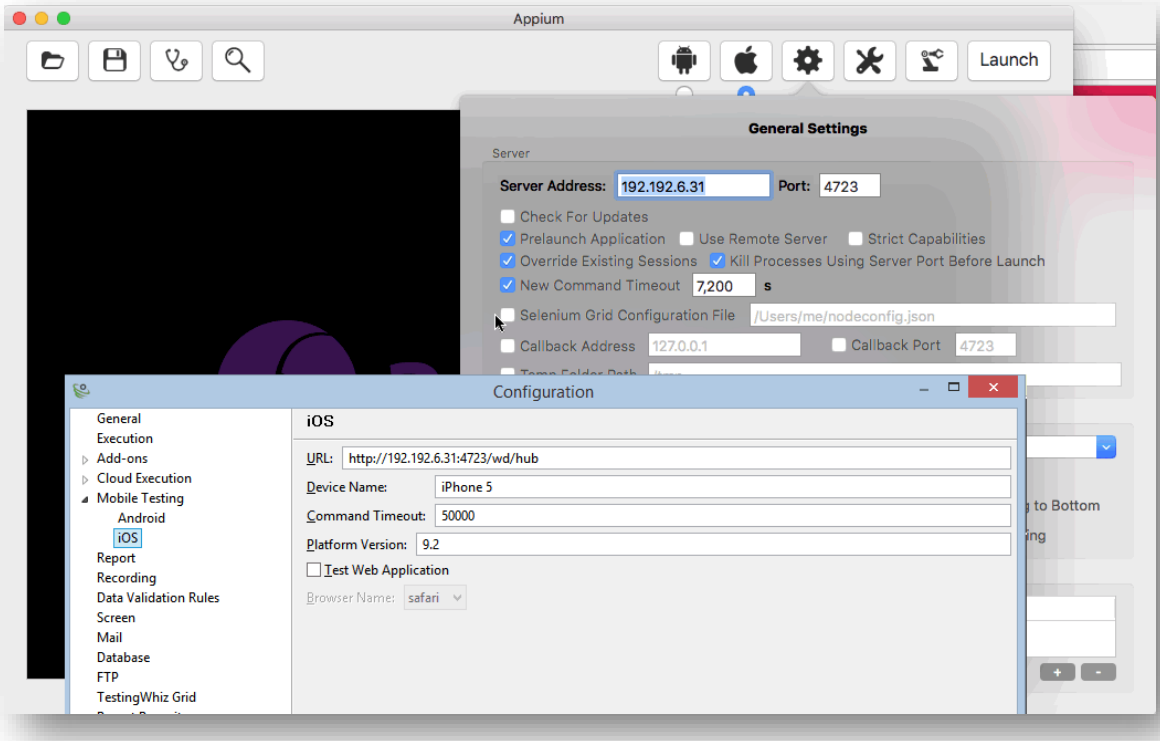
6.7.2.1 Process to Execute Test Cases on iOS Simulator

Step 1: Create and start a new iOS Simulator Device by using XCode.

Step 2: Start the Appium Server.

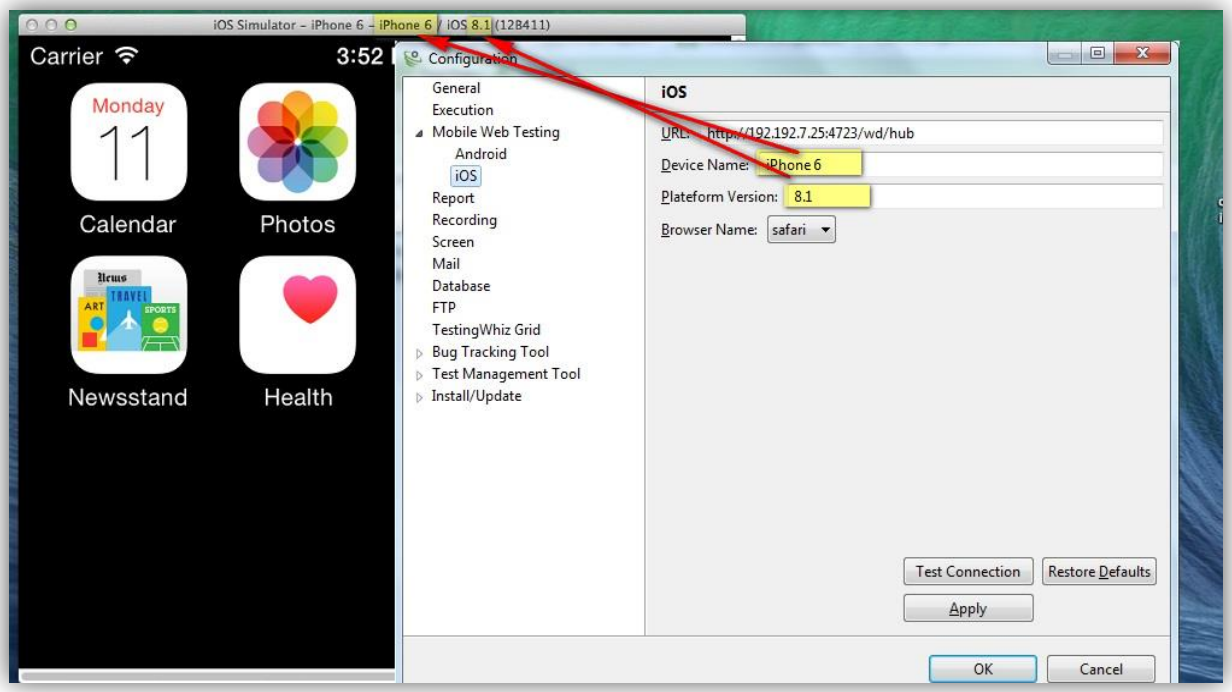
Step 3: Copy the Appium Server Address and Port Number.

Step 4: Paste the copied Server Address and Port Number into Server URL field – TestingWhiz Settings>Configurations>Mobile Web Testing>iOS.



Step 5: Copy the iOS Simulator Name.

Step 6: Paste the copied Simulator Name into Device Name field – TestingWhiz Settings>Configurations>Mobile Web Testing>iOS



Step 7: Select the Browser from the Browser Name drop-down.

Step 8: Apply and Test the Connection.

Step 9: Close the Configuration Window.

Step 10: Execute the Test Case on iOS Simulator.

6.7.2.2 Process to Execute Test Cases on Real IOS Device

Step 1: Go to Application and Select **Appium**.

Step 2: Right Click on it and Click **Show Package Contents**.

Step 3: Select **resources > > node-modules > > appium > > build**.

Step 4: Copy **SafariLauncher.zip "Link"** to the above mentioned location.

Step 5: Start the Appium server with Device UDID Capability.

Step 6: Install **ios_webkit_debug_proxy from "Link"** and follow the steps mentioned in it.

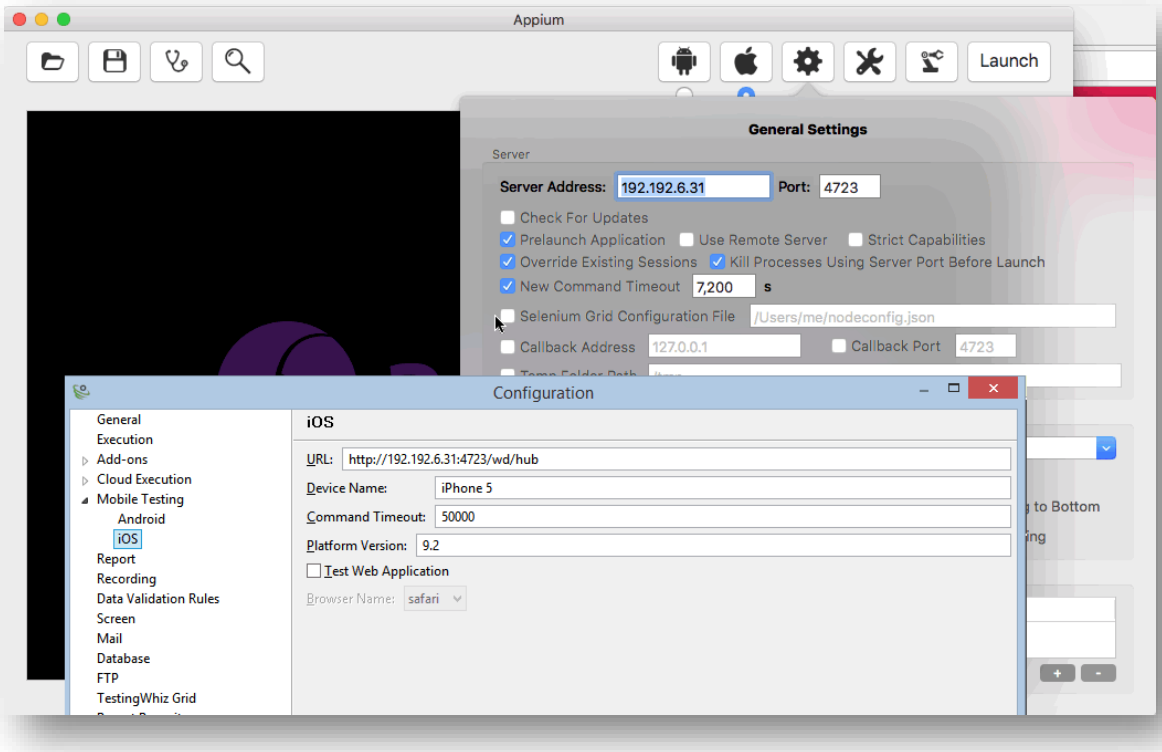
Step 7: Open Terminal and execute **"ios_webkit_debug_proxy -c "UDID of Device: 27753" -d"** command.

Step 8: Connect a real iOS device with MAC PC.

Step 9: Start the **Appium Server**.

Step 10: Copy the **Appium Server Address** and **Port Number**.

Step 11: Paste the copied **Server Address** and **Port Number** into Server URL field - TestingWhiz Settings > Configurations > Mobile Web Testing > iOS.



Step 12: Specify the the Device Name from **'iOSDeviceName'** device - TestingWhiz Settings > Configurations > Mobile Web Testing > iOS.

Step 13: Select the Browser from the browser names drop-down.

Step 14: Apply and Test the Connection.

Step 15: Close the Configuration Window.

Step 16: Execute the Test Case on a real iOS Device.

[**Note:** *User needs to have following iOS environment as a pre-requisite to perform Mobile execution on real iOS devices.*

1. Mac OS: 10.9.5

2. XCode: 6.1.1
3. iOS - 7.1 or 8.1
4. Appium 1.3.4

6.8 Data Cleansing via Data Validation

Data cleansing is the process of detecting and correcting (or removing) corrupt or inaccurate records from a set of data records or database originally caused by user entry errors, by corruption in transmission or storage or different data dictionary definitions of similar entities in different stores. Data cleansing ensures that all the data sets are consistent and can be used in a meaningful manner.

Here's the complete process of how to perform Data Cleansing in TestingWhiz:

6.8.1 How to Perform Data Cleansing

For performing Data Cleansing, a user needs to first set up Data Validation rules.

To set up data validation rules, follow the steps mentioned below:

Step 1: Click Settings > Configuration > Data Validation Rules.

Step 2: Select the rule to validate and clean the data.

[**Note:** *By default, TestingWhiz provides 9 Data Validation rules.*]

Refer Section – **Configuration** > **Data Validation Rules** to create more rules other than the default rules.

Step 3: Create a New Test Case under a Test Suite.

Step 4: Add a Test Command '**Clean' > 'Data Set'**

Step 5: Click **Value tab**.

[**Note:** *A new window will pop-up.*]

Step 6: Browse and **select the file of Database** which contains the junk data to perform the cleaning

Step 7: Specify the Delimiter to separate different columns of data set.

Step 8: Check the option **'Consider first row as the header row'** to set first row as header if column headers are not specified explicitly.

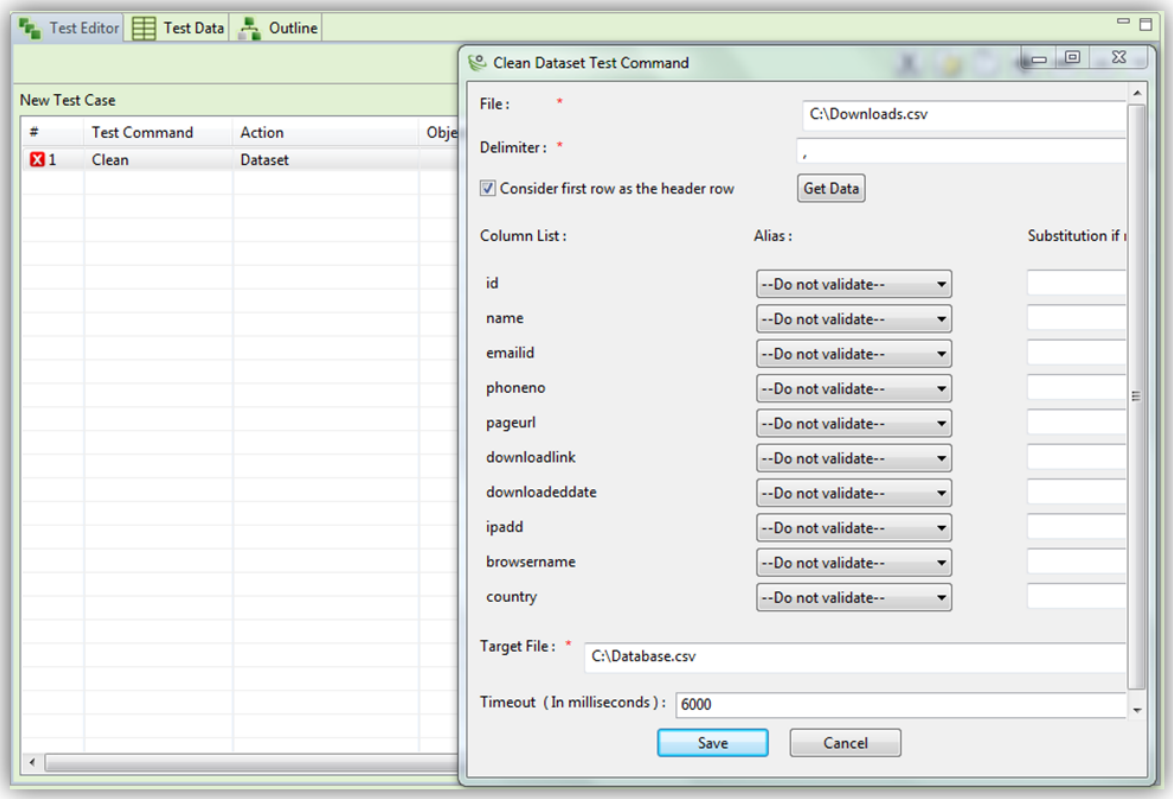
Step 9: Click **'Get Data'** to fetch all the column values of data set which populates the following

- **Column List:** This is the list of all the columns from your data set file.

- **Alias:** This dropdown populates all the rules from the Data Validation Rules setting, and each of these rules can be applied against the column they have been selected for.
- **Substitution if rule is broken:** User needs to specify a replacement string which would be replaced if any of the rule is broken against each column.

Step 10: Select the **'Target File'** location where the file after Data Cleaning needs to be saved.

Step 11: Specify **'Timeout'** according to the complexity and time taken to consume data set to replace all the fields. Number of rows is directly proportional to the time. Time is to be specified in milliseconds.



After execution TestingWhiz will create a file which has cleaned data ready for further use. All the data would be validated according to the rule applied.

6.9 Risk Based Testing

Risk Based Testing is a type of software testing in which functions and features are tested based on priority. It uses risk analysis to recognize proactive chances to take out or avoid defects through non-testing activities and to help users select which test activities to perform.

This kind of testing includes both mitigation (testing to give chances to decrease the likelihood of faults, especially high-impact faults) and contingency (testing to know a workaround to create the defects that do get past us less painful).

TestingWhiz enables a user to perform Risk Based Testing by defining the risks at the Test Case level. This gives a user the granular advantage to test even the critical & minute parts of your application.

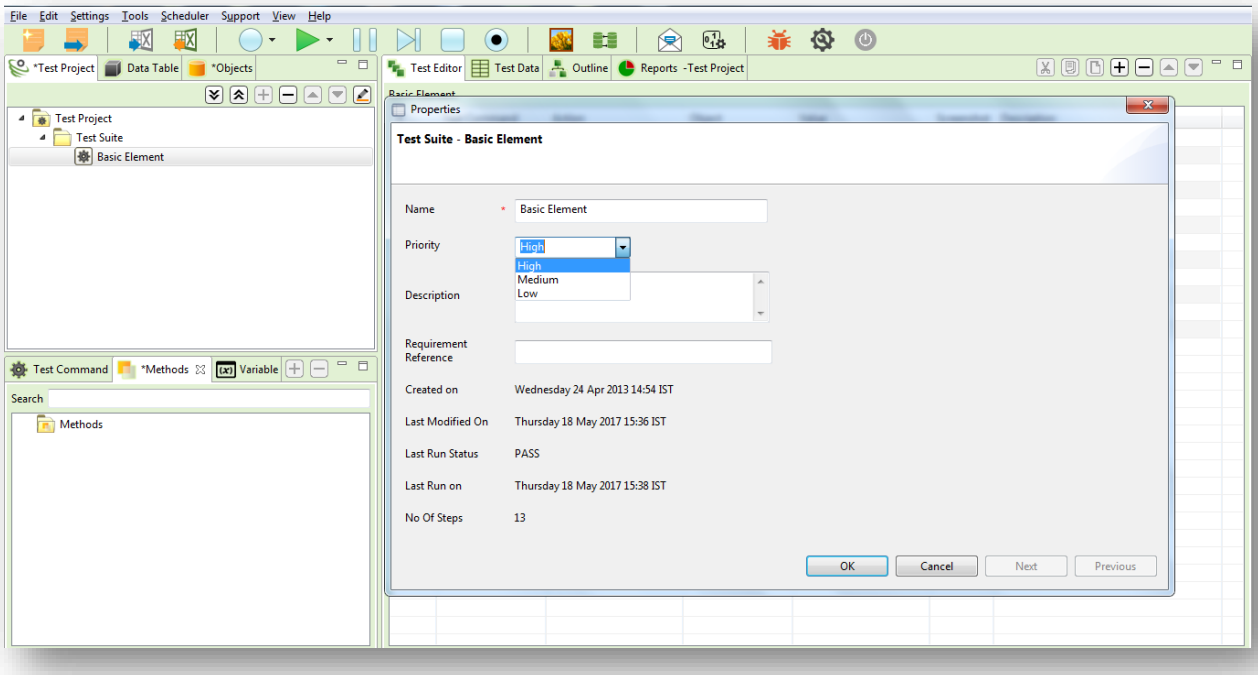
Here's the complete process of performing Risk Based Testing

6.9.1 How to perform Risk Based Testing (RBT)

Step 1: Create a new **Test Case** under a Test Suite

Step 2: Right Click on the Test Case and open **'Properties'** to define the Priority from the drop-down for performing RBT.

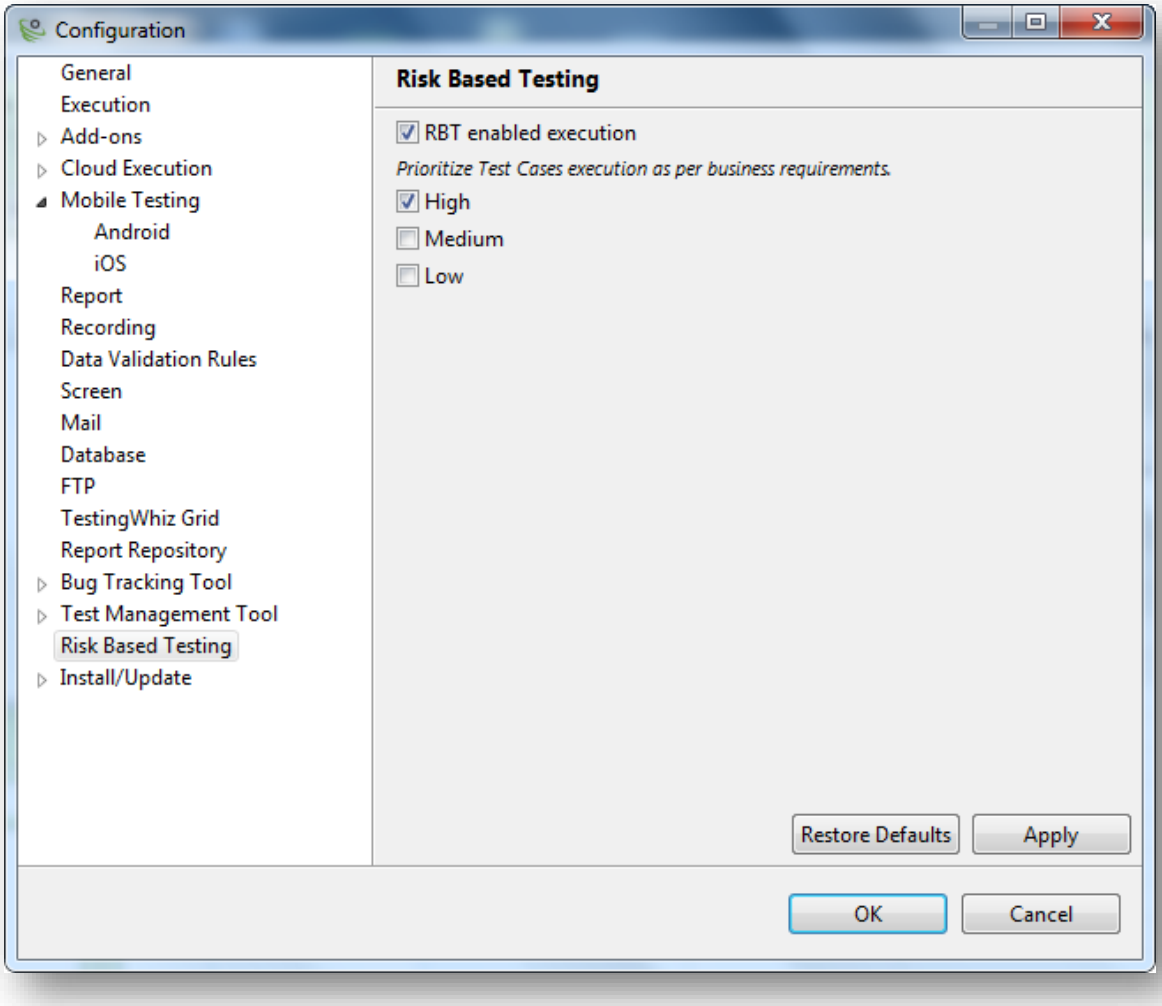
(**Note:** *By default, priority will be set as Medium.*)



Step 3: Enable Risk Based Testing by going to **Settings** >> **Configuration** >> **Risk Based Testing** and check the option **'RBT enabled execution'**.

Step 4: Choose the **Priority** of the respective Test Case as High, Medium or Low by checking on the respective options and click **Apply** to enable the execution.

Step 5: Execute the testing of the Test Cases selected under Risk Based scenario based on their priorities on the browser of choice.



6.10 Web Services Testing

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer.

TestingWhiz allows users to test **REST and SOAP** WebServices.

Here's the complete process of performing Web Services Testing with TestingWhiz:

6.10.1 REST Web Services Testing

Step 1: Create a New Test Case under a Test Suite.

Step 2: Select Test Command '**Execute > Rest Web Service'** from the available Test Commands.

Step 3: Click **Value** tab.

[**Note:** *A new window will pop-up.*]

Step 4: Enter the URL of the Web Service.

Step 5: Select the type of **Method** supported by the REST URL from the drop-down.

Step 6: Specify the **Headers (if any)** in **Request Header** fields. User can also specify multiple Headers separated in multiline.

Step 7: Specify '**Request Body'** parameters. User can also specify multiple Request Body parameters separated in multiline.

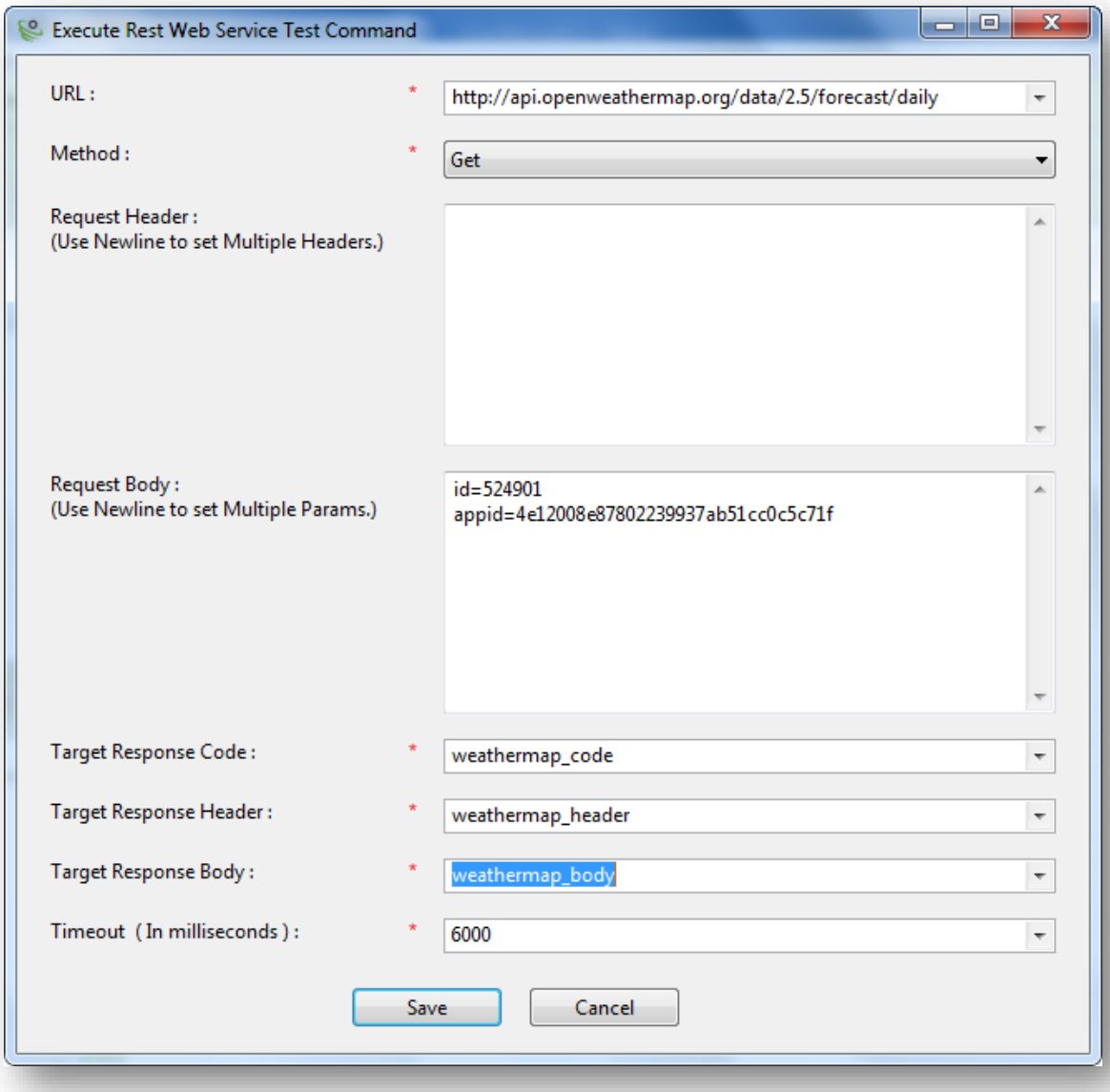
Step 8: Enter the variable name in '**Target Response Code**' field to store the Response code after execution.

Step 9: Enter the variable name in '**Target Response Header**' field to store the Header Response after execution.

Step 10: Enter the variable name in '**Target Response Body**' field to store the Response Body after execution.

Step 11: User can specify service **'Timeout'** period in milliseconds to control script behavior better. Default Timeout would be 6000 milliseconds

Step 12: Click Save.



Further, the user needs to parse message received as Target Response Body in REST Web Service.

Step 13: Select Test Command '**Parse > JSON Message**'.

Step 14: Click **Value** tab.

[**Note:** *A new window will pop-up.*]

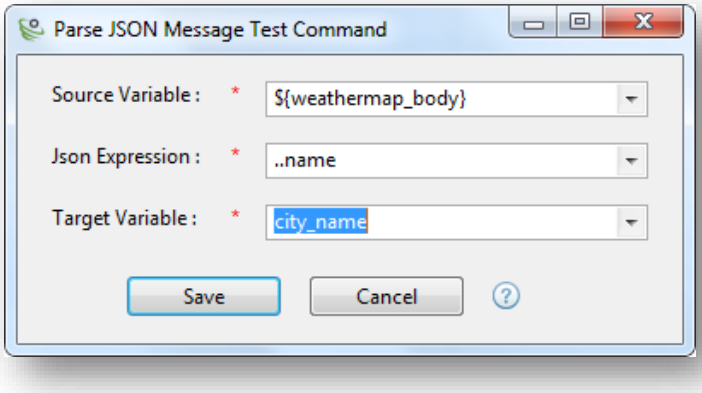
Step 15: Enter the Source Variable in which JSON Expression needs to be evaluated.

[**Note:** *User needs to specify the same variable value which was specified in the Response Body field of Execute > REST Web Service test command.*]

Step 16: Enter the JSON Expression to extract data from the JSON Response variable specified above. User can hover over the Help icon to get suggestions.

Step 17: Specify the **Target Variable** name to store the result of the JSON Expression after execution.

Step 18: Click **Save**.



6.10.2 SOAP Web Services Testing

Step 1: Create a New Test Case under a Test Suite.

Step 2: Select Test Command '**Execute > SOAP Web Service'** from the available Test Commands.

Step 3: Click **Value** tab.

[**Note:** *A new window will pop-up.*]

Step 4: Enter the WSDL of the SOAP Web Service.

Step 5: Validate the WSDL to get all the functions supported by the specified WSDL.

Step 6: Select the type of **Method** from the drop-down populated based on the specified WSDL.

Step 7: Specify the Request based on the Method selected. User can edit the parameters and XML Request text inside the Request field.

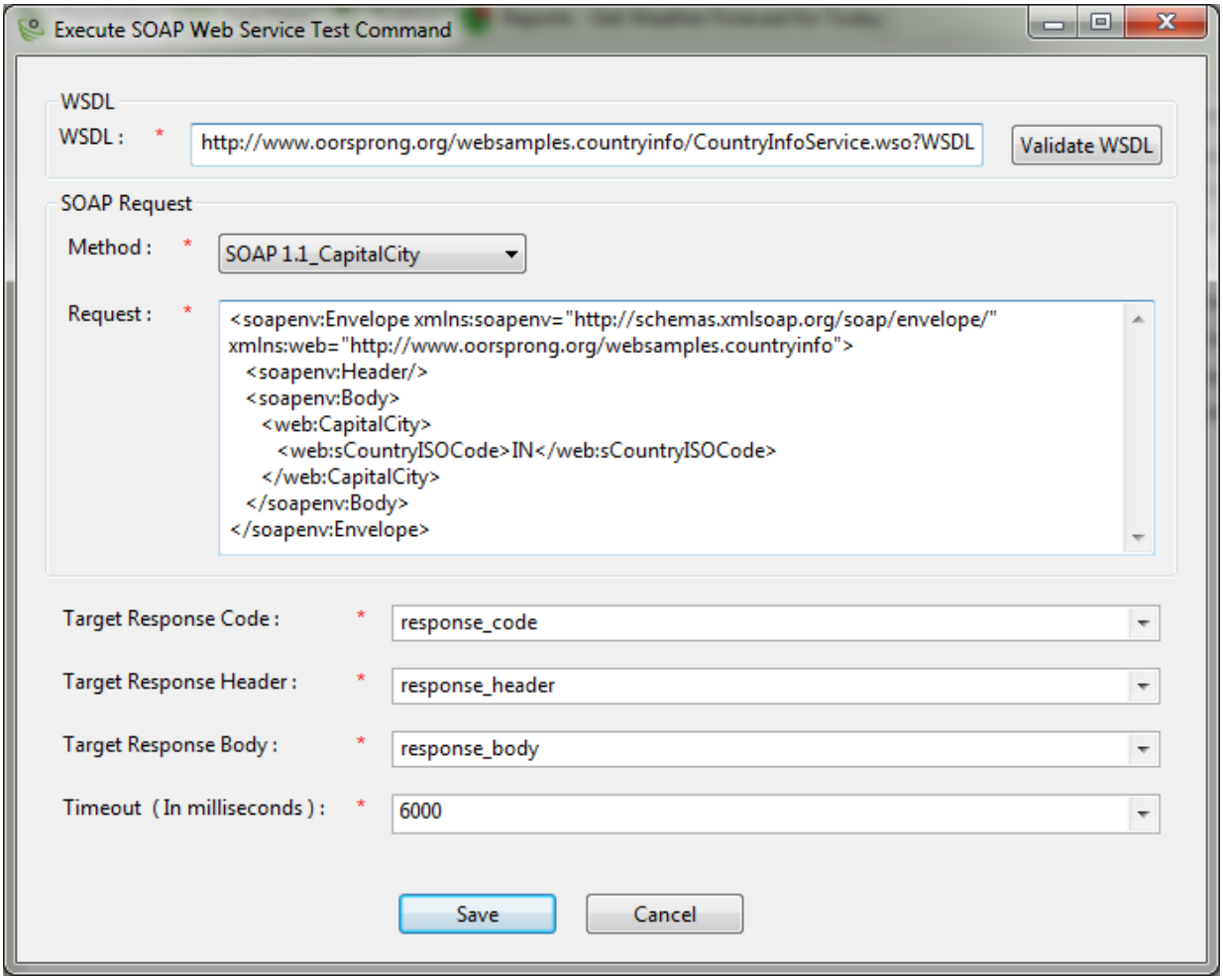
Step 8: Enter the variable name in '**Target Response Code**' field to store the Response code after execution.

Step 9: Enter the variable name in '**Target Response Header**' field to store the Header Response after execution.

Step 10: Enter the variable name in '**Target Response Body**' field to store the Response Body after execution.

Step 11: User can specify service **'Timeout'** period in milliseconds to control script behavior better. Default Timeout would be 6000 milliseconds

Step 12: Click **Save**.



Further, user needs to parse message received as Target Response Body in SOAP Web Service.

Step 13: Select Test Command '**Parse > XML Message**'.

Step 14: Click **Value** tab.

[**Note:** *A new window will pop-up.*]

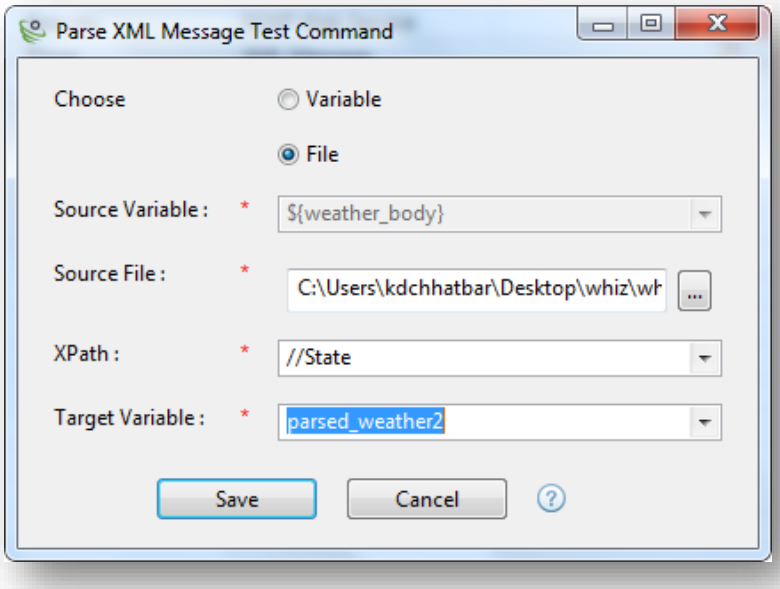
Step 15: Enter the Source Variable in which XPath needs to be evaluated

[**Note:** *User needs to specify the same variable value which was specified in the Response Body field of Execute > SOAP Web Service test command.*]

Step 16: Enter the XPath to extract data from the XML Response variable specified above. User can hover over the Help icon to get suggestions.

Step 17: Specify the **Target Variable** name to store the result of the XPath after execution.

Step 18: Click **Save**.



6.11 Execution via TestingWhiz CI Plugin

TestingWhiz allows users to execute Test Scripts on server via TestingWhiz CI plugins such as Jenkins.

Here's the process of integrating Jenkins Server with TestingWhiz:

Step 1: Download the TestingWhiz Plugin file from TestingWhiz download page.

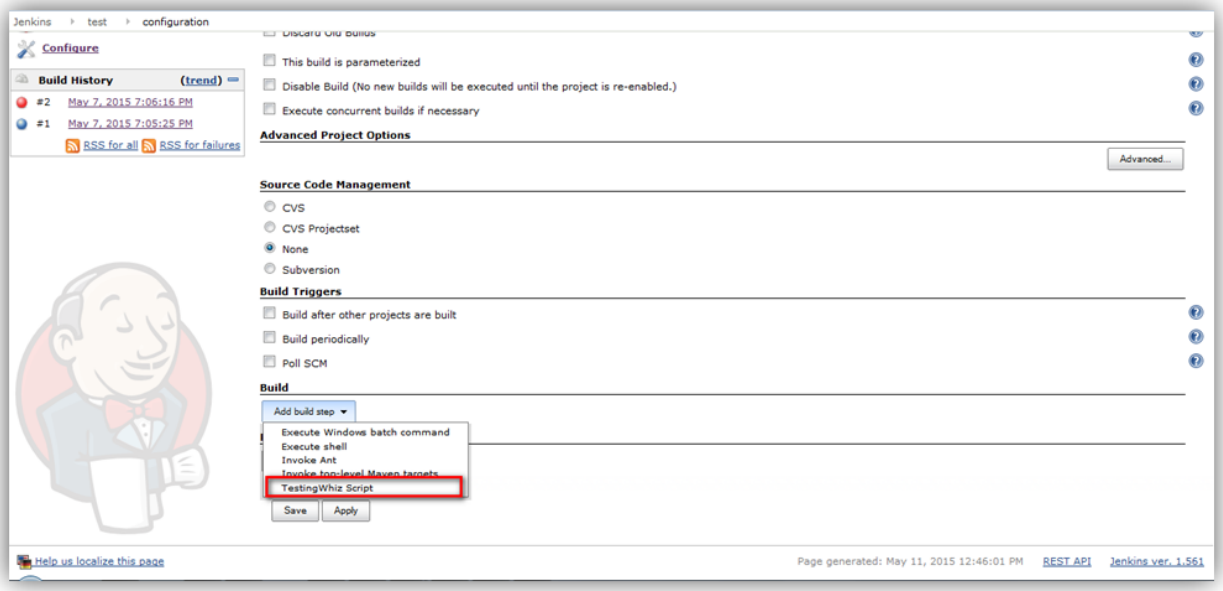
Step 2: Place the downloaded plugin file into the **.jenkins** home directory.

Step 3: Start the Jenkins server and access in the browser.

Step 4: Build a Free Style project.

Step 5: Click on **Add Build Step** and select TestingWhiz Script.

Step 6: Enter the Server address where the TestingWhiz server is running (http://ipaddress:5050).

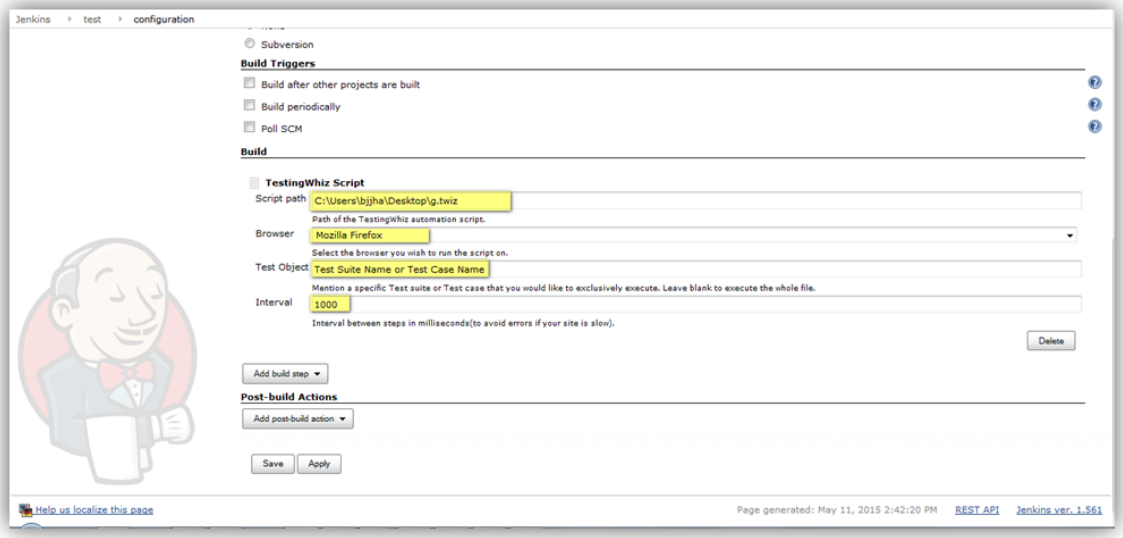


Step 7: Specify the absolute path of a test case file or users can also specify the folder path where all the test scripts are stored to execute.

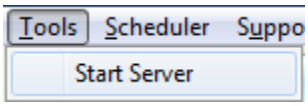
Step 8: Specify the browser for your build by selecting one from the Browser drop-down.

Step 9: Enter a specific Test suite or Test case to exclusively execute inside the Test Object column or leave it blank to execute the entire script file (optional).

Step 10: Specify the interval time between two steps that is to be performed while execution (optional).



Step 11: Click **Start Server** from Tools drop-down of TestingWhiz.



Step 12: Test Script is ready to be executed via Jenkins server.

[**Note:** *To avail Jenkins integration functionality on your TestingWhiz, email at sales@testing-whiz.com.*]

6.12 Accessing DataTable Values Without Loop

TestingWhiz allows user to access Datatable Values without loop.

Syntax to access Datatable Value without loop in value column of TestingWhiz commands is as follows:

${tablename.columnname[index#]}
Here is the example which showcases how to access the command.

e.g. If a user wants to access 2nd row of employee 'Name' column of 'Employee' table then syntax will be:

${Employee.Name[2]}

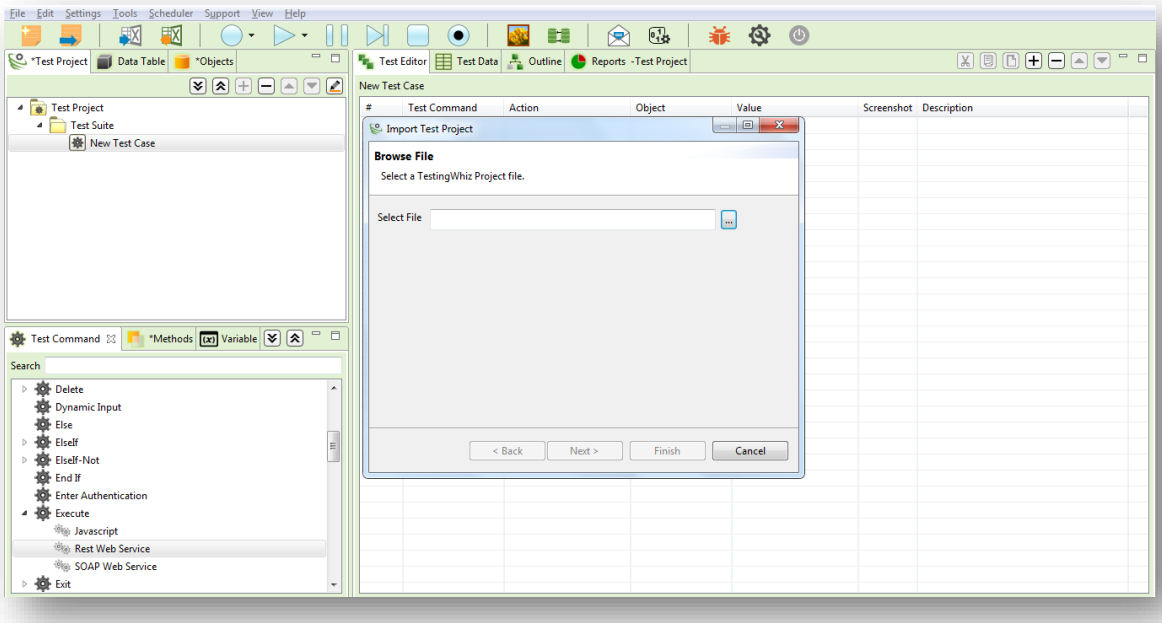
6.13 Importing Data from Other Test Projects

TestingWhiz facilitates you to import Test Cases\Suites\Data\Methods from other Test Project.

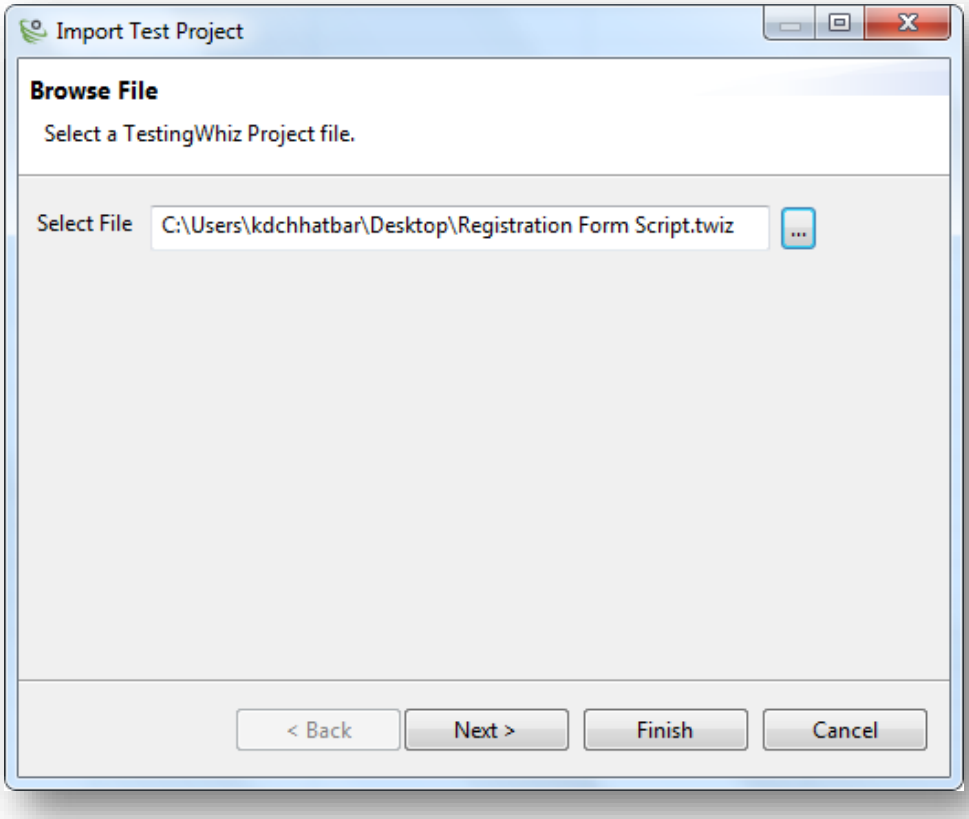
Here's the process to import Test Data from another Test Project:

Step 1: Select **File**

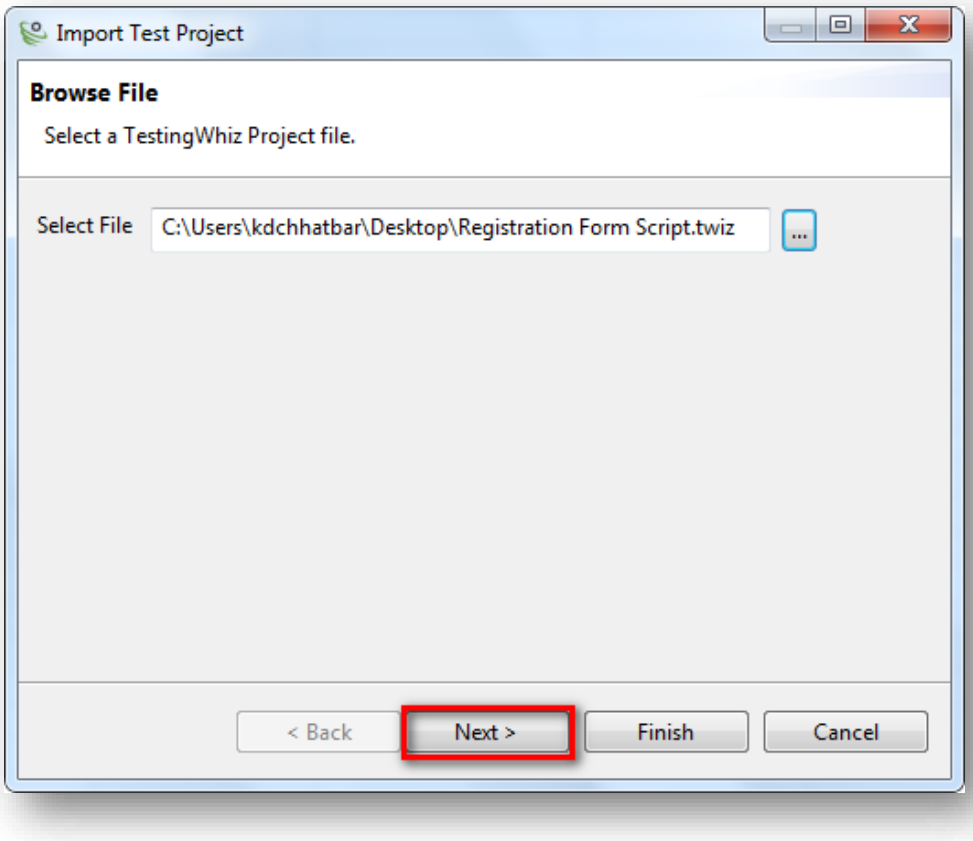
Step 2: Click on **Import Test Project.** A pop up to select Test Project will appear.



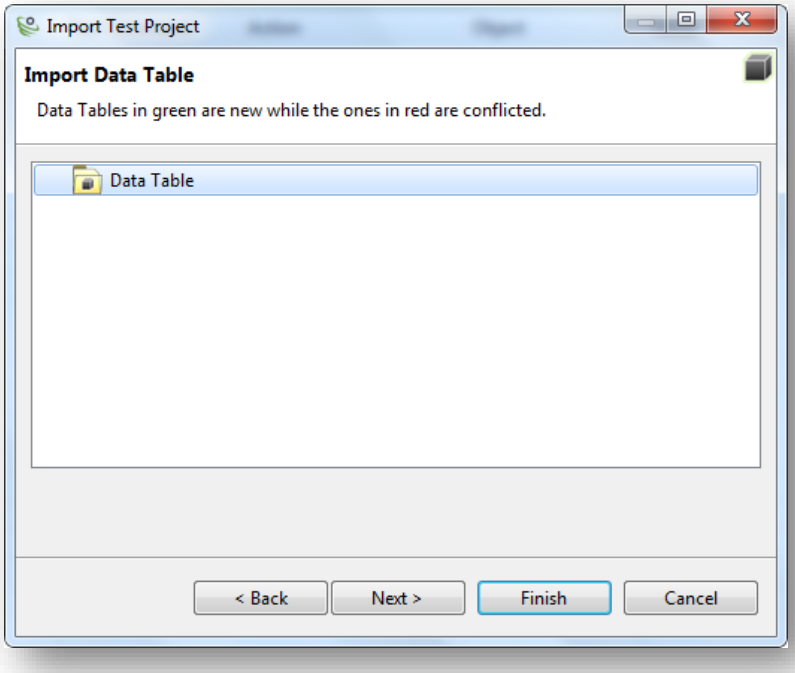
Step 3: Click  to select **.twiz** file of the Test Project.

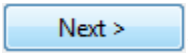


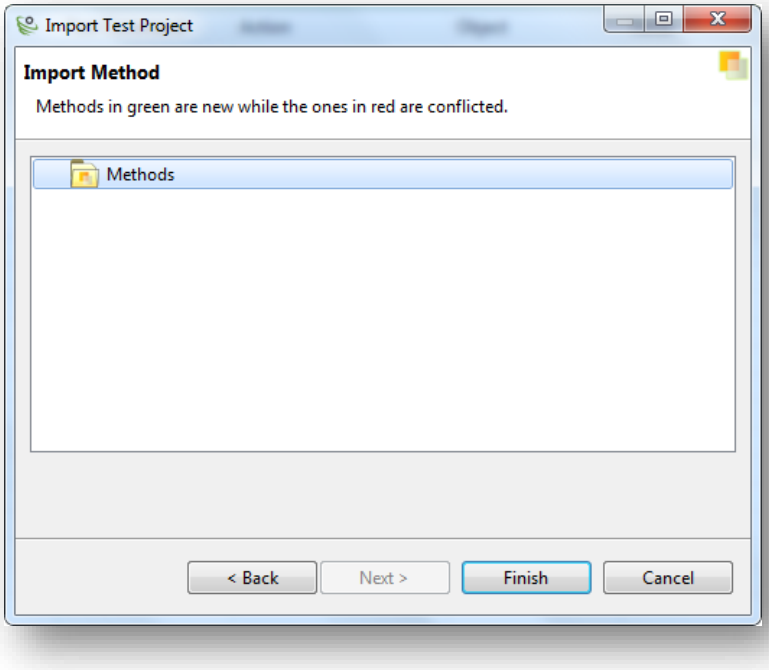
Step 4: Click Next > to select the Test Object.



Step 5: Click Next > to select the Data Table.



Step 6: Click on  to select the Method.



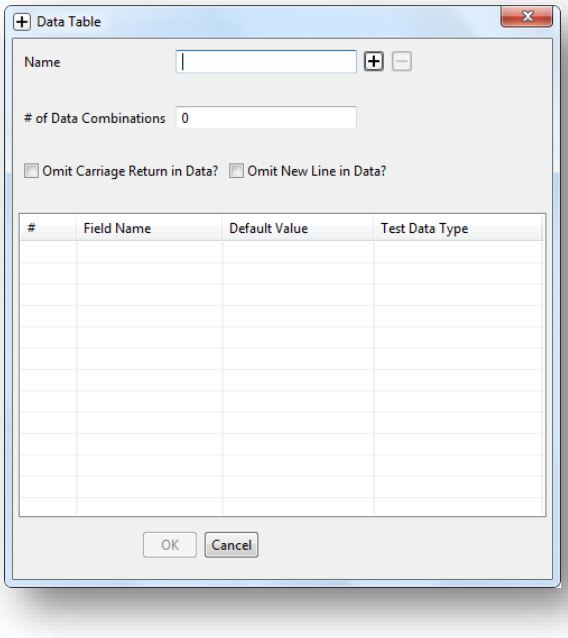
Step 7: Click  to complete the process.

6.14 Generating Test Data Table

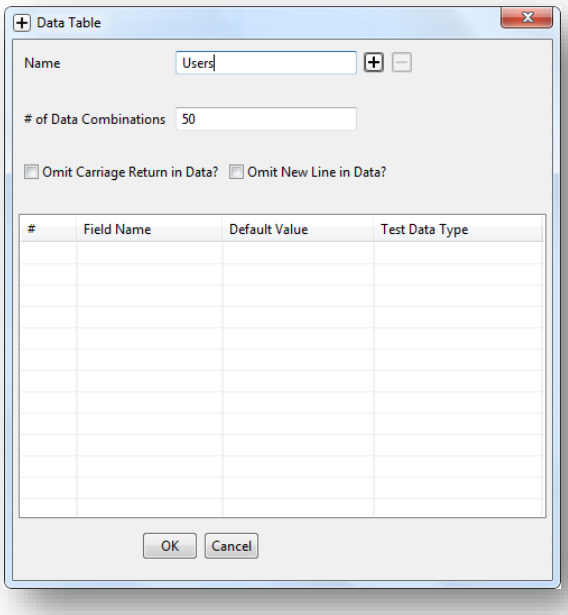
TestingWhiz allows the user to generate sample Test Data according to the Data type.

Here is the process to generate the Sample Data:

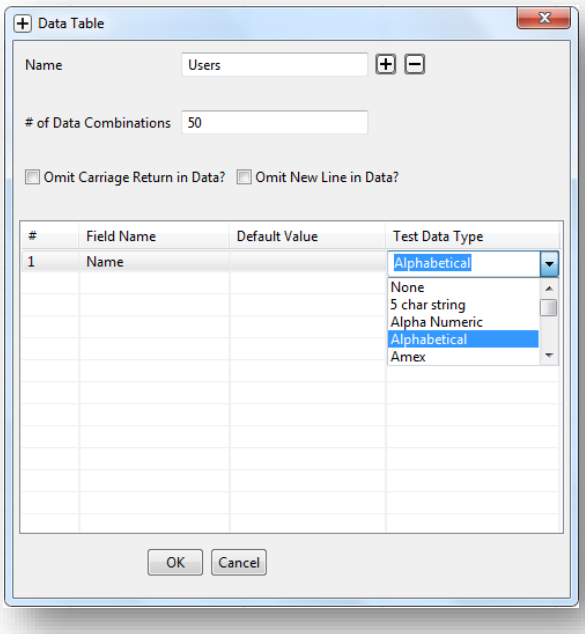
Step 1: Click on  . This will open up a Dialog box.



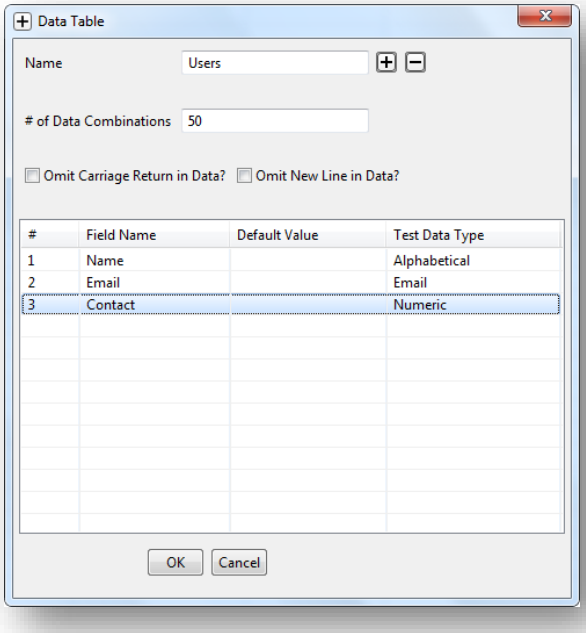
Step 2: Enter the Name for the Data Table and Number of Combination as per requirement.



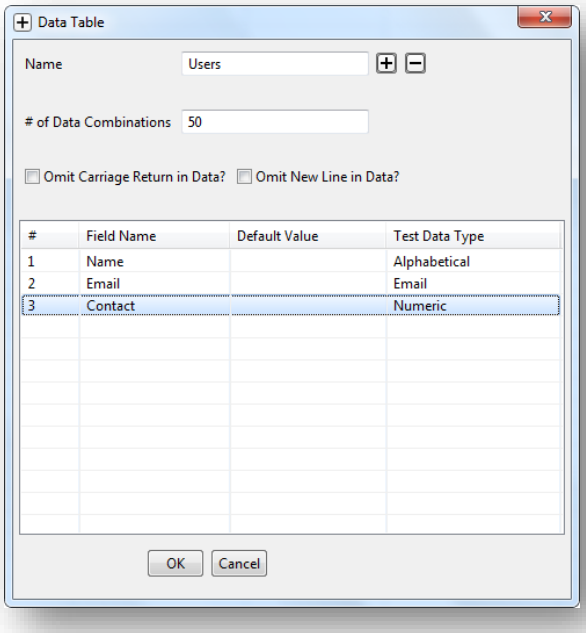
Step 4: Click on ⊞ to add the Field Name and Select the Test Data Type as shown below.



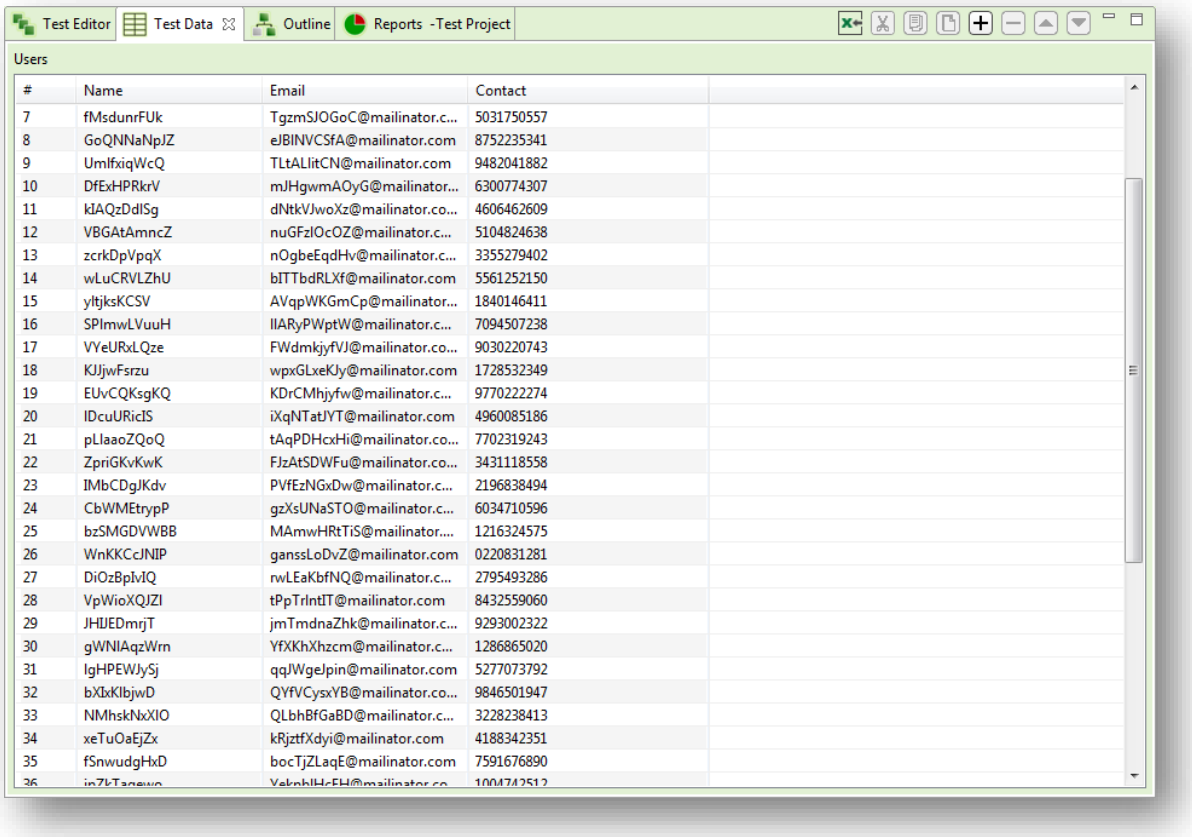
Step 5: Add Other Field Names as per requirements.



Step 6: After filling all the details, click on OK button.



Step 7: This will generate sample Test Data as shown in the below figure.



#	Name	Email	Contact
7	fMsdunrFUk	TgzmSJOGoC@mailinator.c...	5031750557
8	GoQNNaNpJZ	eJBINVCSfA@mailinator.com	8752235341
9	UmlfxiqWcQ	TLtALlitCN@mailinator.com	9482041882
10	DfExHPRkrV	mJHgwmAOyG@mailinator...	6300774307
11	kIAQzDdlSg	dNtkVJwoXz@mailinator.co...	4606462609
12	VBGAtAmncZ	nuGFzlOcOZ@mailinator.c...	5104824638
13	zcrkDpVpqX	nOgbeEqdHv@mailinator.c...	3355279402
14	wLuCRVLZhU	bITTbdRLXf@mailinator.com	5561252150
15	yltjksKCSV	AVqpWKGmCp@mailinator...	1840146411
16	SPImwLVuuH	IIARyPWptW@mailinator.c...	7094507238
17	VYeURxLQze	FWdmkjyfVJ@mailinator.co...	9030220743
18	KJJjwFsrzu	wpxGLxeKJy@mailinator.com	1728532349
19	EUvCQKsgKQ	KDrCMhjyfw@mailinator.c...	9770222274
20	IDcuURicIS	iXqNTatJYT@mailinator.com	4960085186
21	pLIaaoZQoQ	tAqPDHcxHi@mailinator.co...	7702319243
22	ZpriGKvKwK	FJzAtSDWFu@mailinator.co...	3431118558
23	IMbCDgJKdv	PVfEzNGxDw@mailinator.c...	2196838494
24	CbWMEtrypP	gzXsUNaSTO@mailinator.c...	6034710596
25	bzSMGDVWBB	MAmwHRtTiS@mailinator....	1216324575
26	WnKKCcJNIP	ganssLoDvZ@mailinator.com	0220831281
27	DiOzBpIvIQ	rwLEaKbfNQ@mailinator.c...	2795493286
28	VpWioXQJZI	tPpTrlntIT@mailinator.com	8432559060
29	JHIJEDmrjT	jmTmdnaZhk@mailinator.c...	9293002322
30	gWNIAqzWrn	YfXKhXhzcm@mailinator.c...	1286865020
31	IgHPEWJySj	qqJWgeJpin@mailinator.com	5277073792
32	bXIxKIbjwD	QYfVCysxYB@mailinator.co...	9846501947
33	NMhskNxXIO	QLbhBfGaBD@mailinator.c...	3228238413
34	xeTuOaEjZx	kRjztfXdyi@mailinator.com	4188342351
35	fSnwudgHxD	bocTjZLaqE@mailinator.com	7591676890

6.15 Integration with Test Management Tools

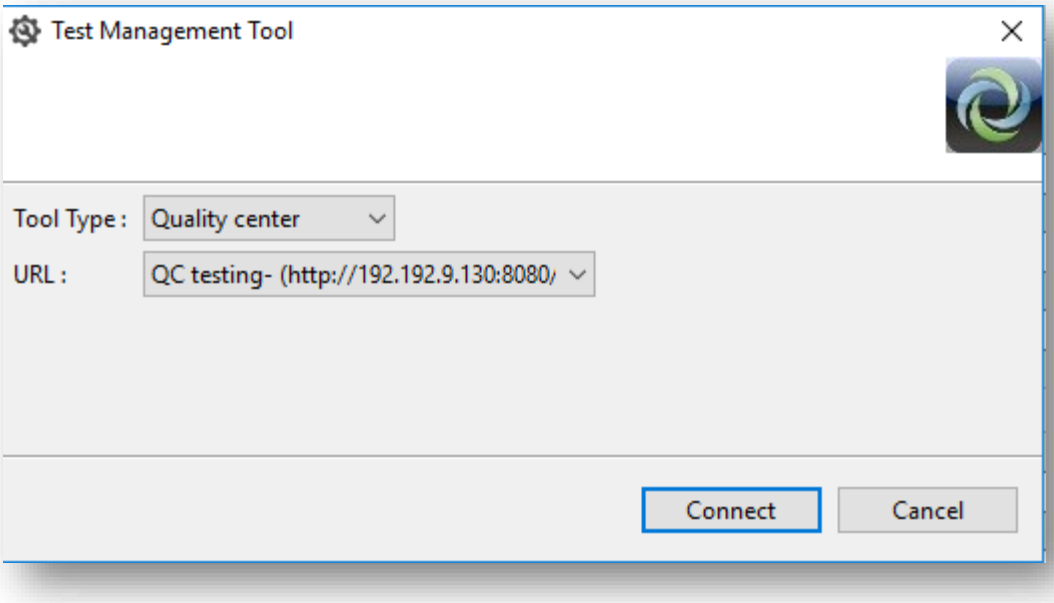
TestingWhiz offers integration with various Test Management tools to collaborate test cases, test runs, test results etc. with testing teams. TestingWhiz collaborates with some of leading third party Test Management tools like Zephyr for Jira, TestRail, and Quality Center.

6.15.1 Collaborating with Quality Center

User needs to perform following steps in order to submit their test run to **Quality Center.**

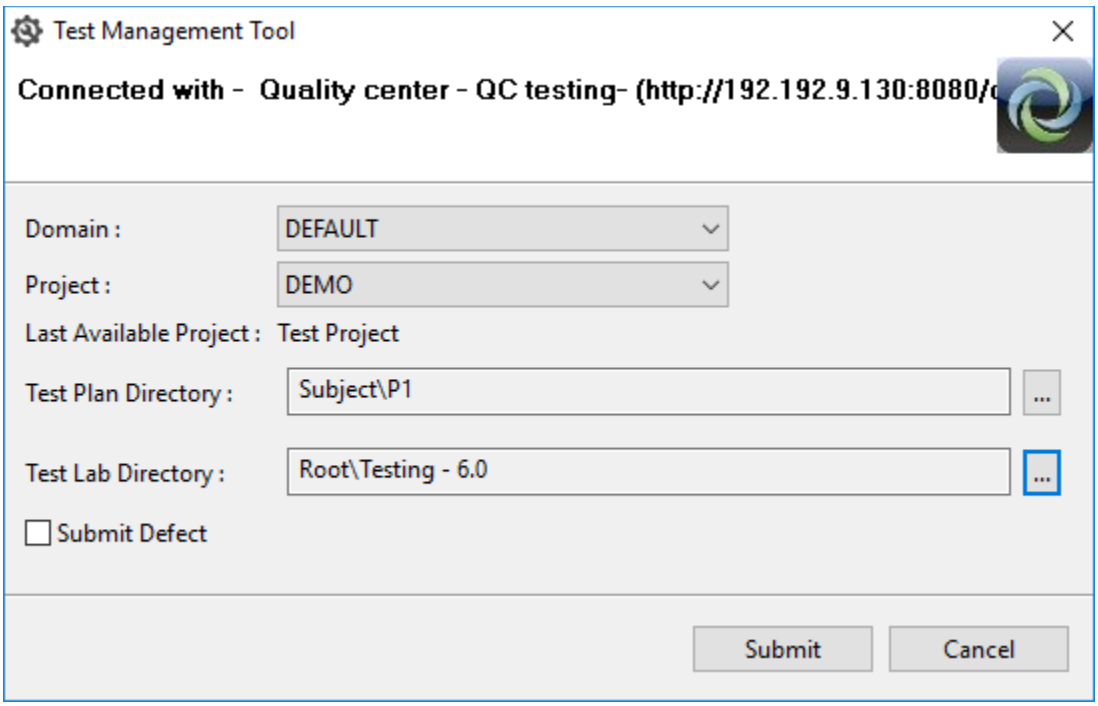
1. Click on  icon from TestingWhiz Toolbar.

2. The following dialog box will appear:



1) Select Quality Center from Tool Type drop down.
2) Select URL from URL drop down.
3) Click on "Connect" button.

3. On successful connection following dialog box will appear:



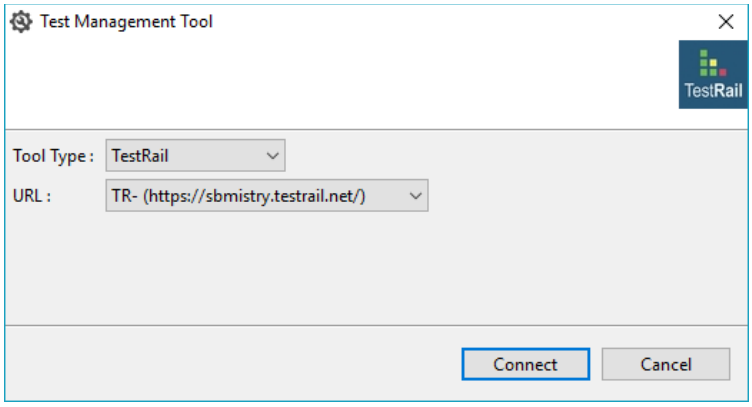
4. Select Project from Project drop down.
5. Select Test Plan Directory by clicking on ... button.
6. Select Test Lab Directory by clicking on ... button.
7. Click on "Submit" button.

6.15.2 Collaborating with Test Rail

User need to perform following steps in order to submit their test run to **TestRail.**

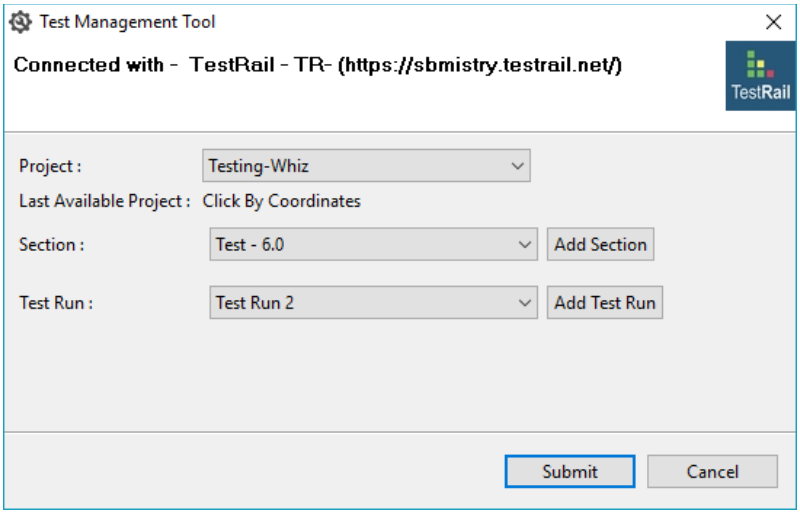
1. Click on  icon from TestingWhiz Toolbar.

2. The following dialog box will appear:



1) Select TestRail from Tool Type drop down.
2) Select URL from URL drop down.
3) Click on "Connect" button.

3. On successful connection following dialog box will appear:



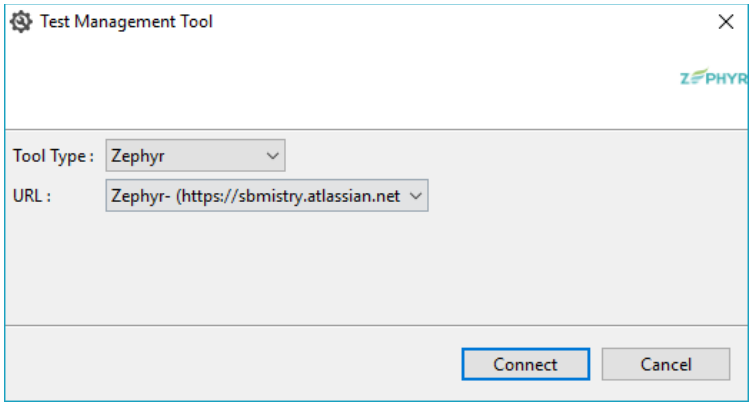
4. Select Project from Project drop down.
5. Select sections of the selected Project or Create New Section by clicking "Add Section" button.
6. Select Test Run of the selected Project or Create New Test Run by clicking "Add Test Run" button.
7. Click on "Submit" button.

6.15.3 Collaborating with Zephyr with Jira

User need to perform following steps in order to submit their test run to **Zephyr with JIRA.**

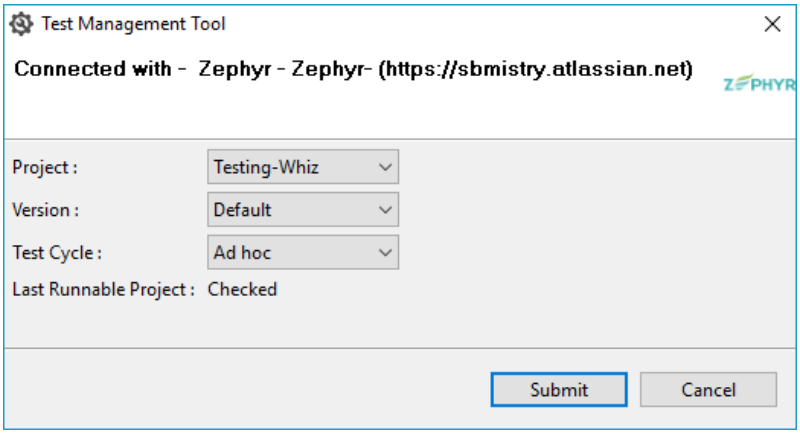
1. Click on  icon from TestingWhiz Toolbar.

2. The following dialog box will appear:



1) Select Zephyr from Tool Type drop down.
2) Select URL from URL drop down.
3) Click on "Connect" button.

3. On successful connection following dialog box will appear:



4. Select Project from Project drop down.

5. Select Version of the selected Project from Version drop down.

6. Select Test Cycle of the selected Project from Test Cycle drop down.

7. Click on "Submit" button.

7 TEST COMMANDS IN TESTINGWHIZ

TestingWhiz supports more than **290+ Test Commands**, including conditional and looping Test Commands. These Test Commands help a user build effective and reliable Automation Test Scripts with least effort.

7.1 How to Add a Test Command?

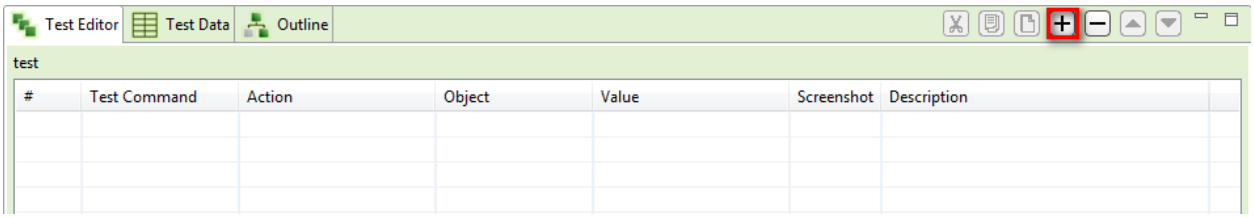
Once a Test Case in a Test Suite has been created, users can start adding Test Steps and necessary Test Commands to perform a particular function while executing that Test Case. There are 2 ways to add Test Command to a Test Case as mentioned below:

7.1.1 Drop-down List

TestingWhiz provides an easy way to add Test Command from the Drop-down list. To add Test Command via Drop-down list, follow these simple steps

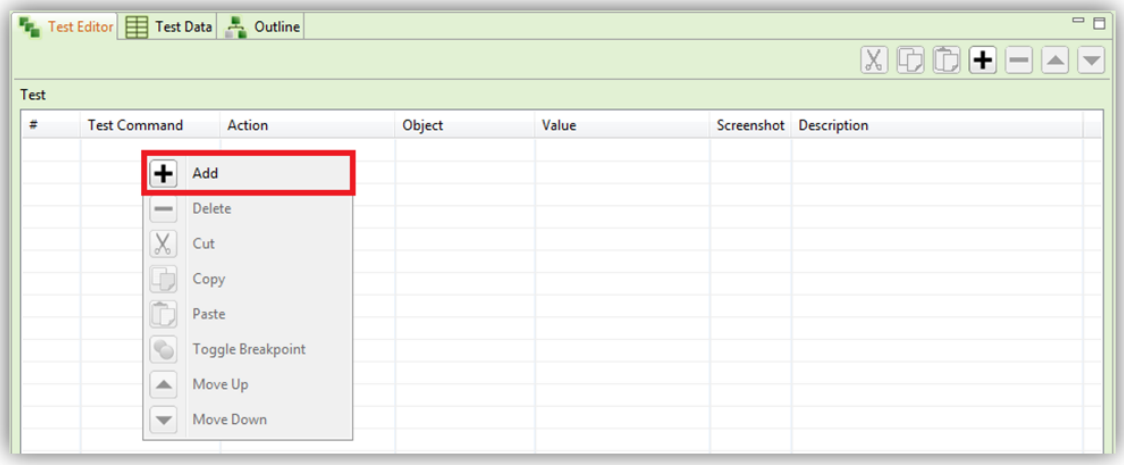
7.1.1.1 Add a Test Step

Add a Test Step in the Test Editor section by clicking on  icon above Test Editor



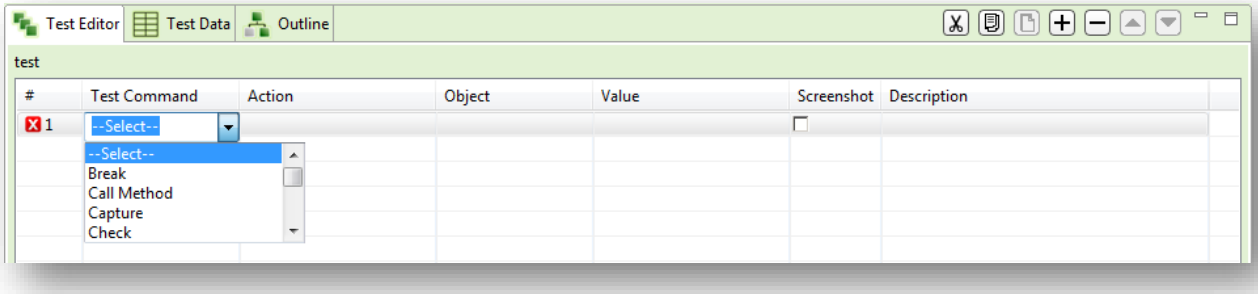
OR

Right click on the Test Editor Section and select Add



7.1.1.2 Select Test Command

After adding a Test Case, click on the corresponding Test Command cell and click on the arrow to select the Test Command from the Drop-down list.



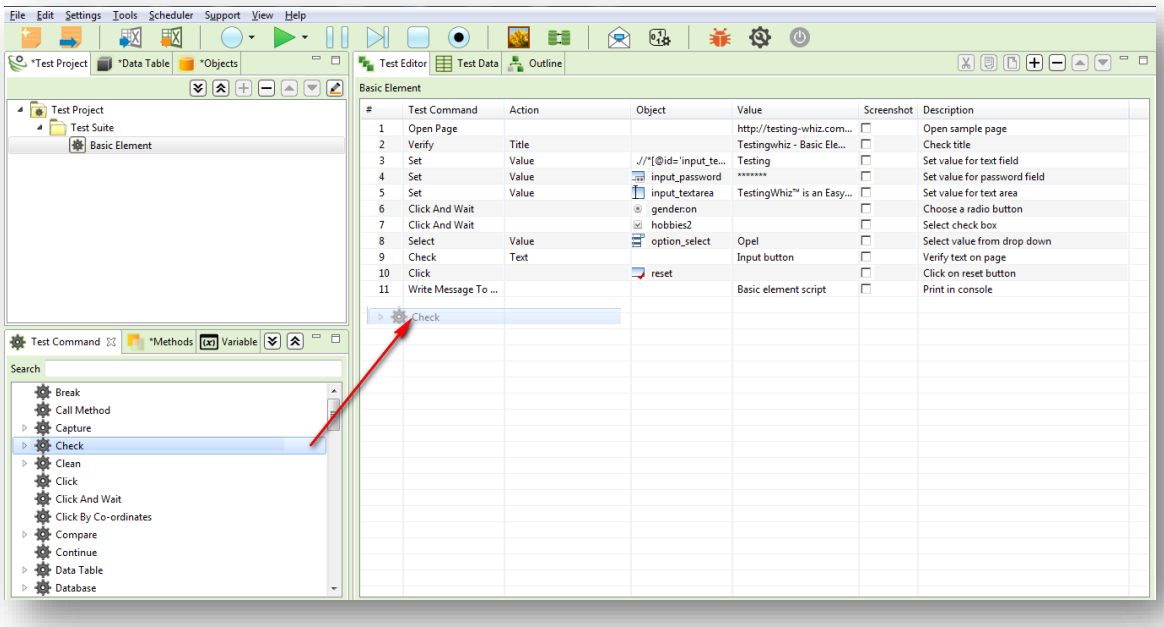
7.1.2 Drag & Drop Test Command

User can Drag & Drop a desired Test Command from the Test Commands Tab to the Test Commands column in the Test Editor section. To add Test Command using Drag & Drop, perform the following steps

Step 1: Select a Test Command from the Test Commands tab

Step 2: Drag it towards the Test Editor

Step 3: Drop in the Test Commands column



7.1.3 Double Click Test Command

User can also add Test Command to a Test Case by double clicking a particular Test Command from the Test Commands tab. To add Test Command, simply select Test Command and double click on it.

7.2 How to Add an Action Corresponding to a Particular Test Command?

Once the required Test Command to a Test Step has been added, a user needs to add the corresponding Action to that Test Command in order to execute that Test Step. Just like Test Command, a user can select Action in 2 ways:

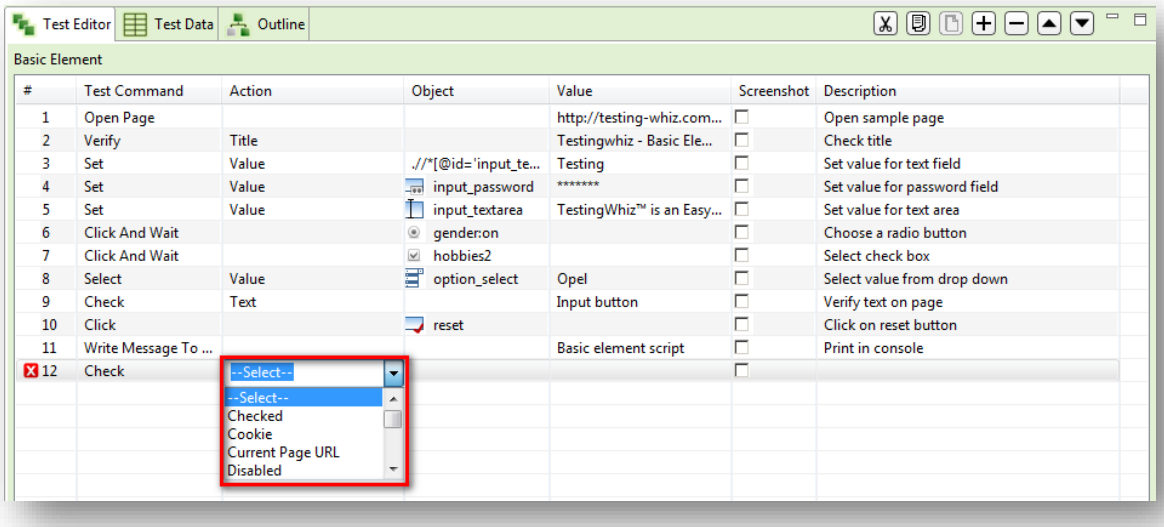
7.2.1 Drop-down List

User can select an Action corresponding to a particular Test Command from the Drop-down list. To select an Action, follow the below mentioned steps:

Step 1: Add a Test Step

Step 2: Select a Test Command

Step 3: Click on the corresponding Action cell and select the Action from the Drop-down list.



Refer Section – **Add Test Command** to learn how to add a test step and select test command before adding an Action.

7.2.2 Drag & Drop Action

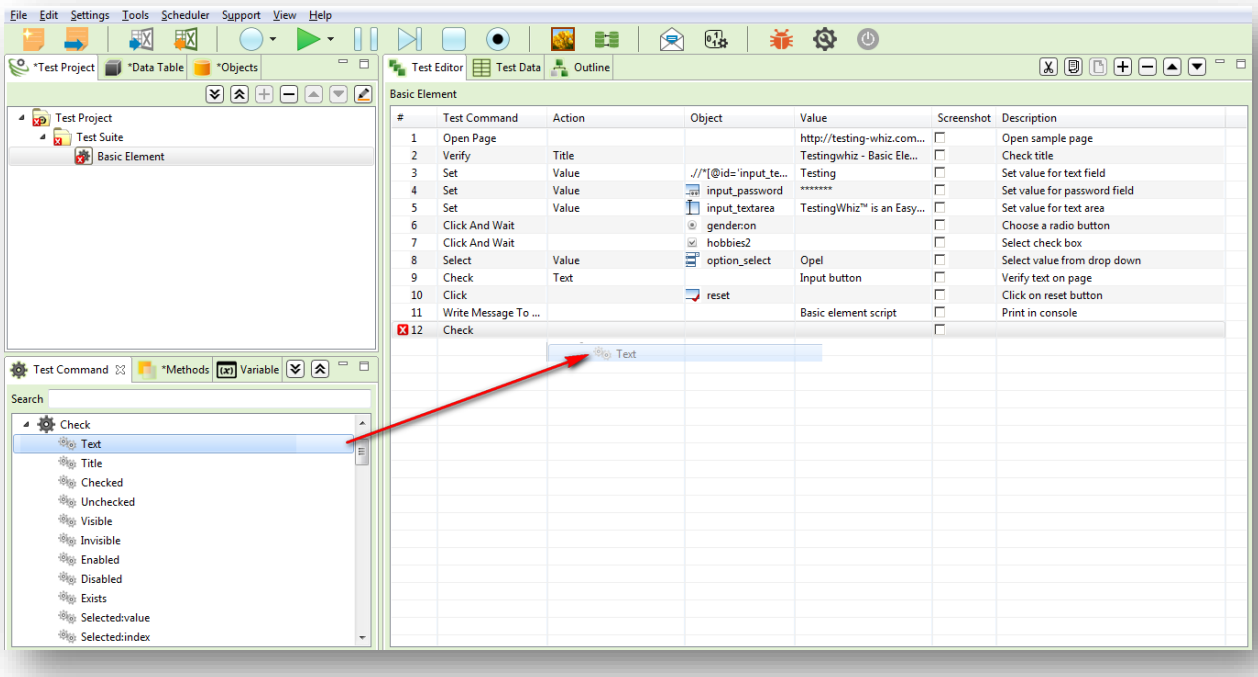
User can also select an Action for a particular Test Command by using Drag & Drop.

To add an Action using Drag and Drop, follow these steps

Step 1: Select the Action.

Step 2: Drag it towards the Test Editor.

Step 3: Drop in the Test Commands column.



7.2.3 Double Click Action

User can also add an Action corresponding to a particular Test Command by expanding a particular Test Command and double clicking on a particular Action in the Test Commands tab.

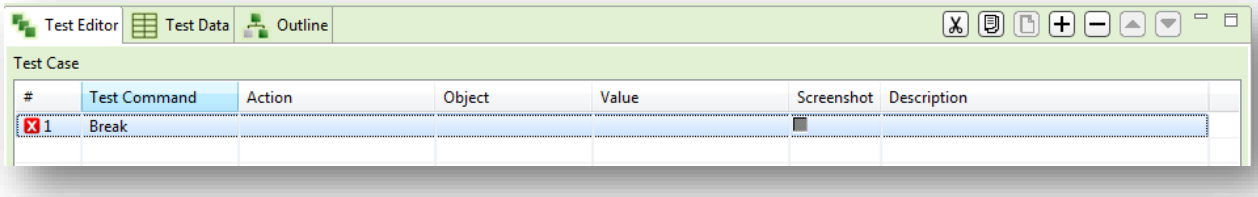
[Note: *Selecting an* Action *will auto-fill the Test Command column if a user has not selected the required Test Command before.*]

8 LIST OF TEST COMMANDS & CORRESPONDING ACTIONS

This chapter will briefly describe about all the available Test Commands in TestingWhiz and their corresponding Actions.

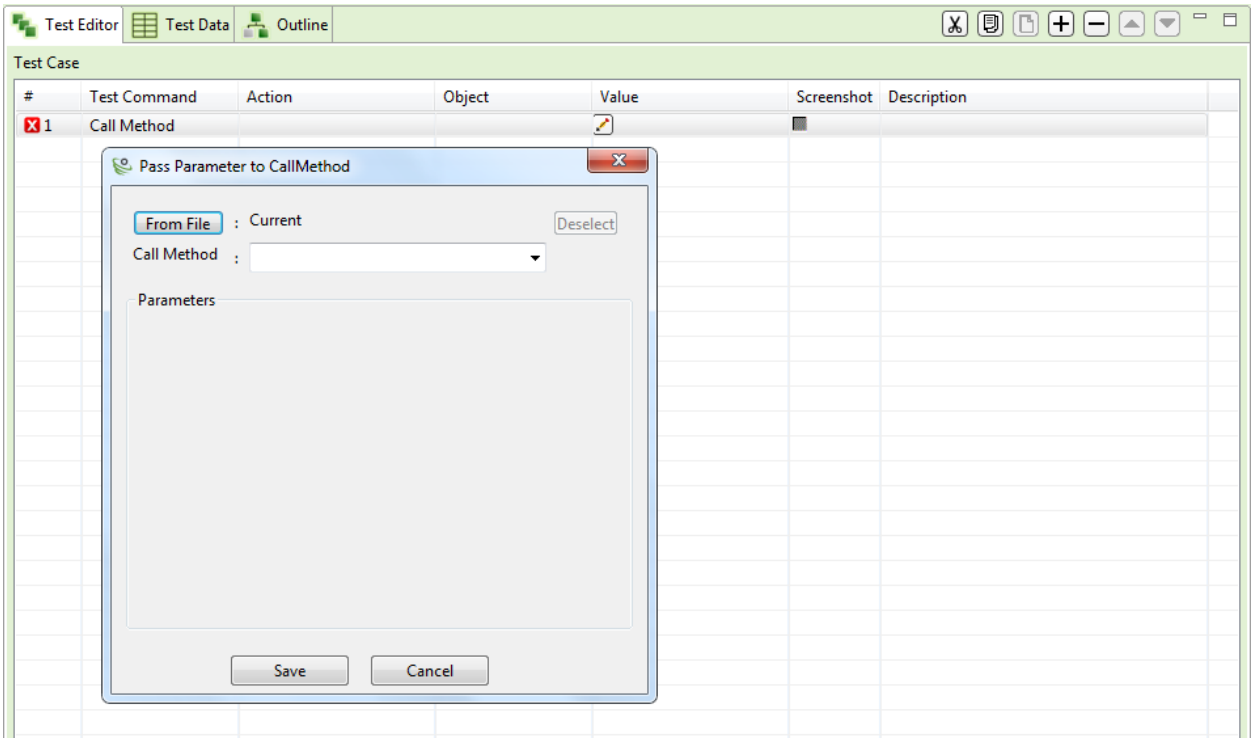
8.1 Break

Break test command allows users to break the loop of test case which is used for a particular scenario.



8.2 Call Method

Call Method test command allows users to call a user-defined Methods (Group of Test Steps clubbed for re-usage or repetitive execution). This command works in the similar manner as the method calling concept would work in other programming language. The methods can be parameterized according to the user requirements. Users can also call methods from a different file on local machine or a remote file path, and utilize them into their scripts.

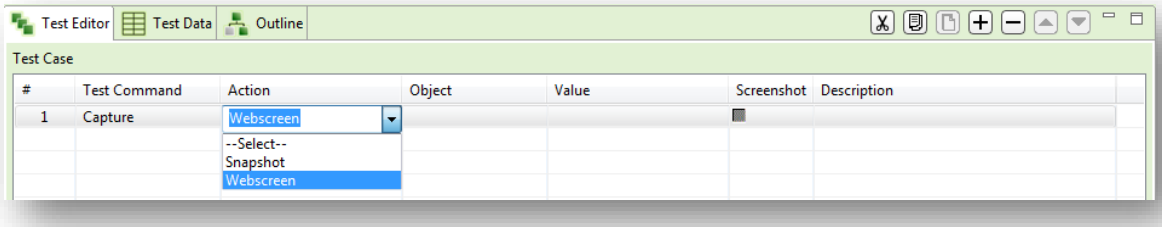


[**Note:** *Selecting a Call Method step and pressing F3 would open up the respective method.*]

8.3 Capture

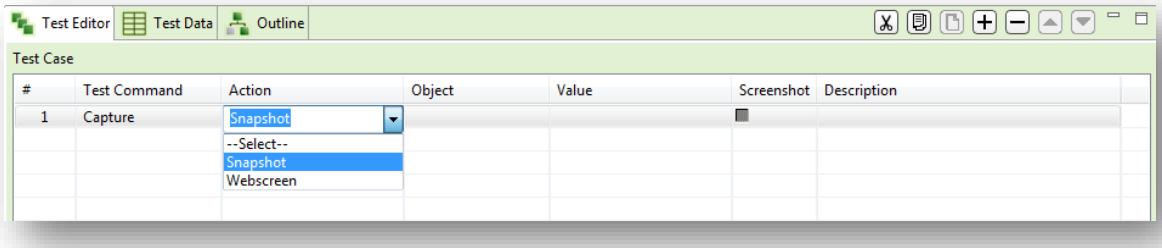
8.3.1 Webscreen

This test command allows users to capture the entire web page of a given URL, and stores it as an image at a predefined storage location.



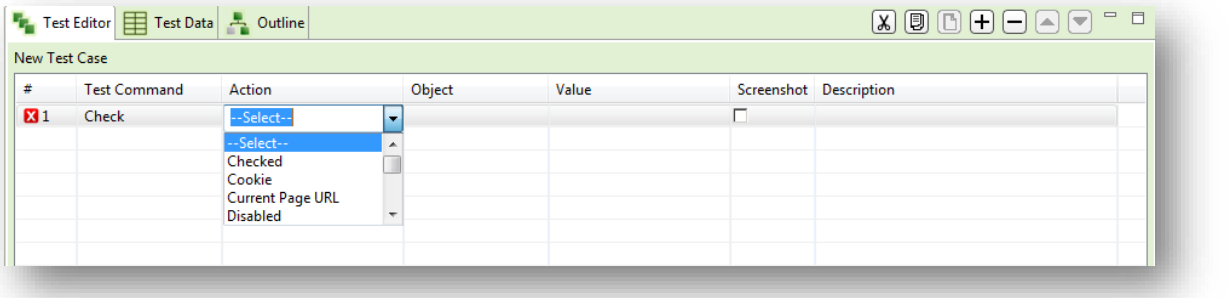
8.3.2 Snapshot

This test command allows users to capture only the visible page screen of the monitor, and stores it as an image.



8.4 Check

Check test command allows users to check a set of actions performed by another user. It will stop the execution from the point where it fails. This is applicable to all the actions that are performed using Check test command.



8.4.1 Text

This action allows users to check whether a specified text is present on a page or not. The action will be performed for all the text on the page. The check will be performed by matching the case of the text value specified. Text with special symbols will not be considered. For e.g. "hello" will be considered different from HELLO. Check test command will stop the execution from the point where it fails.

8.4.2 Title

This action allows users to check whether the title of the page has a specified value or not. Check test command will stop the execution from the point where it fails.

8.4.3 Checked

This action allows users to check whether the checkbox is checked or selected. Check test command will stop the execution from the point where it fails.

8.4.4 Unchecked

This action allows users to check whether the checkbox is unchecked or de-selected. Check test command will stop the execution from the point where it fails.

8.4.5 Visible

This action allows users to check whether a specific object is visible on the page or not. Check test command will stop the execution from the point where it fails.

8.4.6 Invisible

This action allows users to check whether a specific object is invisible/hidden on the page or not. Check test command will stop the execution from the point where it fails.

8.4.7 Enabled

This action allows users to check whether the object (links, buttons etc.) is enabled on the page. Check test command will stop the execution from the point where it fails.

8.4.8 Disabled

This action allows users to check whether the object (links, buttons etc.) is disabled on the page. Check test command will stop the execution from the point where it fails.

8.4.9 Exists

This action allows users to check whether the object exists on a specified page or not. Check test command will stop the execution from the point where it fails.

8.4.10 Selected:value

This action allows users to check whether the option of a specified value is selected in the dropdown list.

8.4.11 Selected:index

This action allows users to check whether the option of a specified index is selected in the dropdown list.

8.4.12 Text:value

This action allows users to check whether the object has a specified value or not. This test command can also be utilized by taking value from the text box.

For e.g. when the values in the textbox are automatically populated from a database, a user can check/verify these values by taking id or object of the textbox.

8.4.13 Cookie

This action allows users to check whether the page contains a specified cookie or not. The result of the cookie's presence or absence will be reflected in the log that is generated for the Report of the Test Case.

8.4.14 Single Occurrence

This action allows users to check whether the value occurs only one time on the page or not. The Single Occurrence action will occur only on page contents. It will not include page title, header etc. Check test command will stop the execution from the point where it fails.

8.4.15 Text Ignore Case

This action allows users to check whether the text is present on the page irrespective of the case of the text. The check will be performed by ignoring the case of the text value specified. Text with special symbols will not be ignored.

For e.g. "hello" will be considered same as HELLO. The check will be performed on all the contents that are present in the form of the text like labels, links etc. Check test command will stop the execution from the point where it fails.

8.4.16 URL Reachable

This action allows users to check if a supplied URL in value column is a valid URL or not.

8.4.17 Image

This action allows users to compare two images with URL to URL, File to File and URL to file comparison. It will run as per the behavior of Check functionality, which includes following scenarios:

A. The check command will fail if tolerance power given is less than actual difference in images.

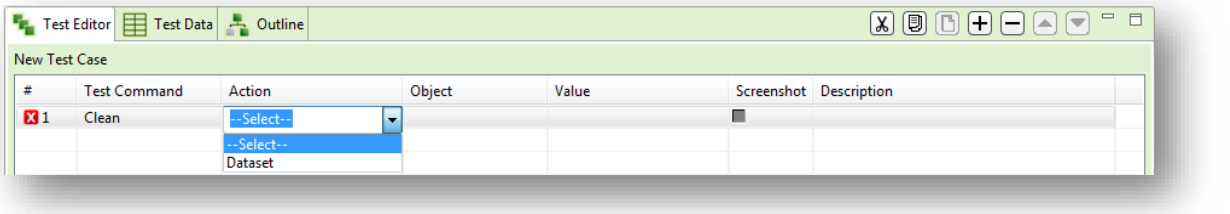
B. The check command will pass if tolerance power given is greater than actual difference in images.

8.4.18 Current Page URL

This action allows users to check the current page URL on the screen.

8.5 Clean

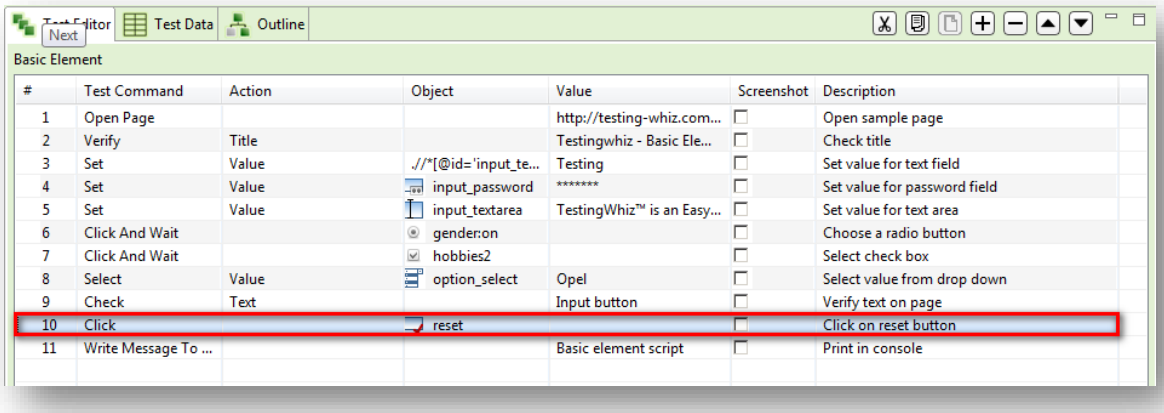
Clean test command allows users to clean junk data which are fetched from raw data sources by validating through a set of rules.



Refer Section – **Data Cleansing** to know more.

8.6 Click

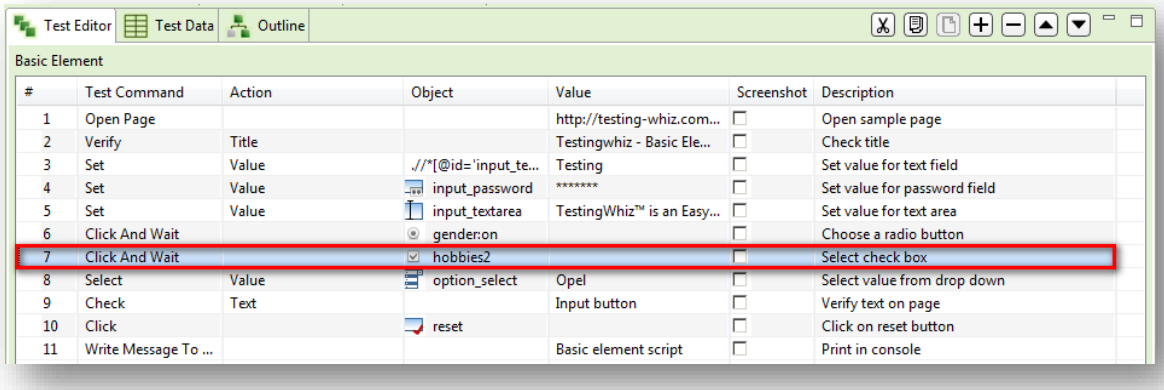
Click test command allows users to perform click on a particular object.



[**Note:** *This test command does not contain any Action.*]

8.7 Click and Wait

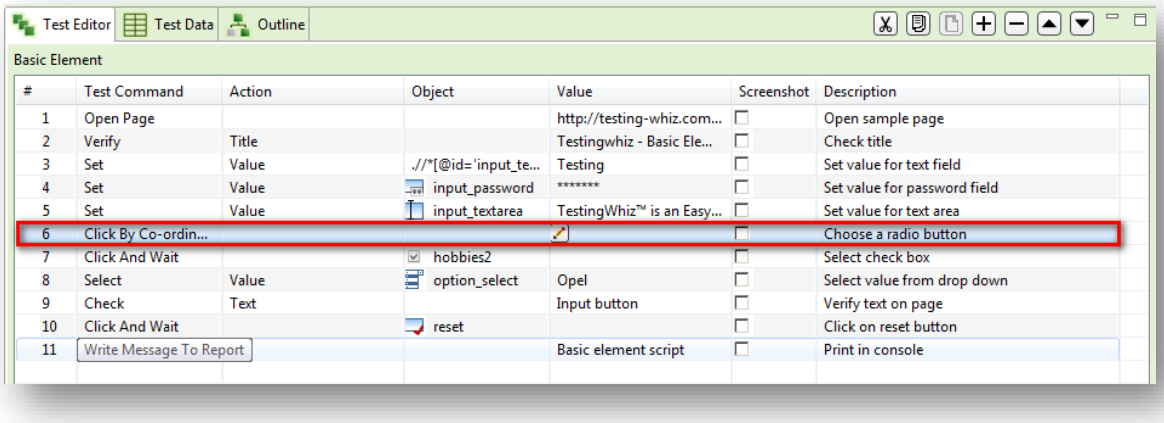
This command allows users to click an object and wait for a particular time before performing the next action.



[**Note:** *This test command does not contain any Action.*]

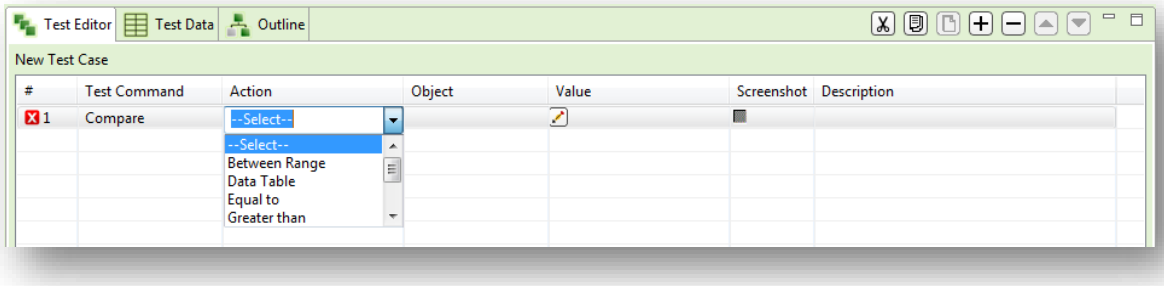
8.8 Click by Co-ordinates

This command allows users to click an object by its X and Y co-ordinates.



8.9 Compare

This test command allows users to compare numerical values and give a Boolean result.



8.9.1 Less than

This action allows users to test whether a value is less than another value.

8.9.2 Less than or equal to

This action allows users to test whether two numeric values are less than or equal to each other.

8.9.3 Greater than

This action allows users to test whether a value is greater than another value or not.

8.9.4 Greater than or equal to

This action allows users to test whether two numeric values are equal to each other.

8.9.5 Equal to

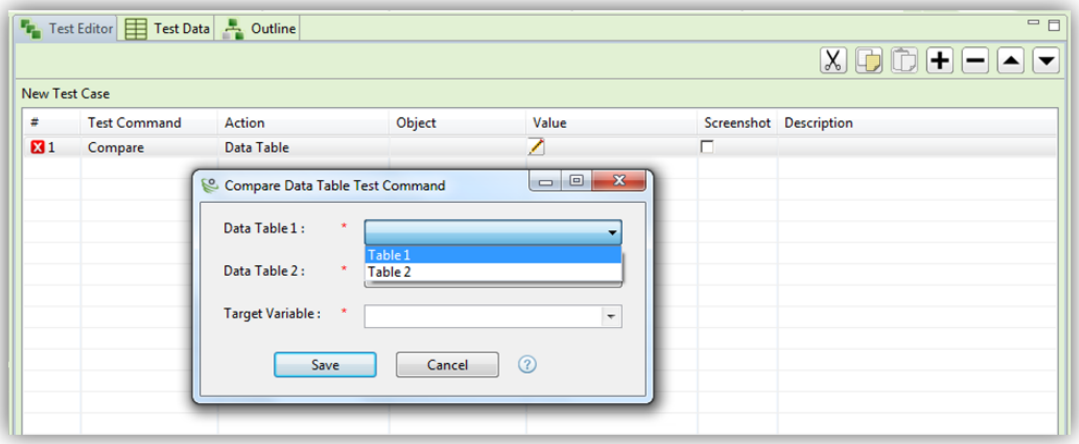
This action allows users to identify two values and return true if the values on both sides are equal to one another.

8.9.6 Not equal to

This action allows users to check if the value of two operands are equal or not.

8.9.7 Data Table

This action allows users to compare two Data Tables and return number of different rows in Data Table1 & Data Table2.



8.9.8 Between Range

This test command allows users to validate whether a number lies between the specified range.

[**Note:** *User needs to specify the Test Value, Range Start, Range End and Target Variable in the Value tab of this command.*]

8.10 Convert

8.10.1 toBinary

This action allows users to convert a decimal number to a Binary number and store it in a Target Variable name specified.

8.10.2 toHex

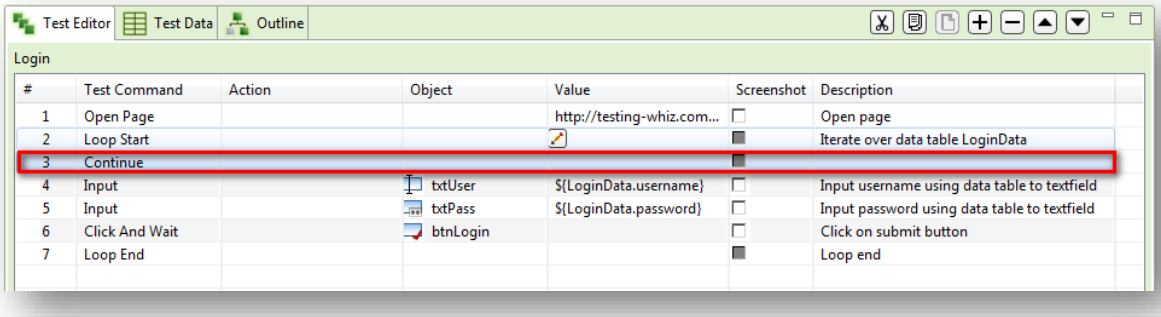
This action allows users to convert a decimal number to a Hexadecimal number and store it in a Target Variable name specified.

8.10.3 toOctal

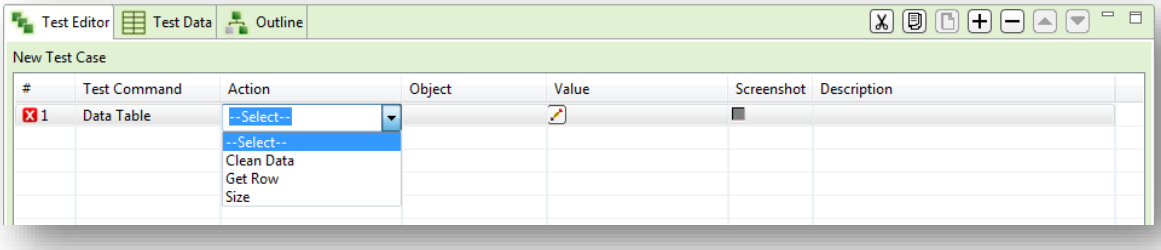
This action allows users to convert a decimal number to an Octal number system and store it in a Target Variable name specified.

8.11Continue

Continue test command helps users to continue through the loop in which it is used.



8.12 Data Table



8.12.1 Size

This action allows users to get the number of rows available in the specified datatable.

8.12.2 Row

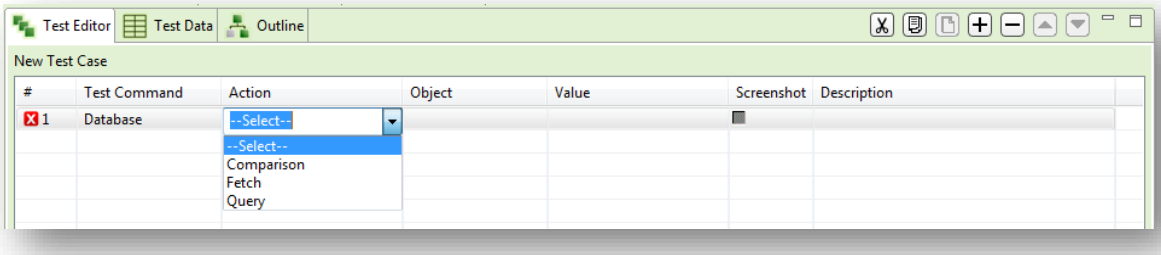
This action allows users to get the entire data of a row in a variable, in which column values are separated by comma.

8.12.3 Clean Data

This action allows users to clean the entire data of Data Table

8.13Database

This test command allows users to perform the database related queries like Select, Insert, Update and Delete.

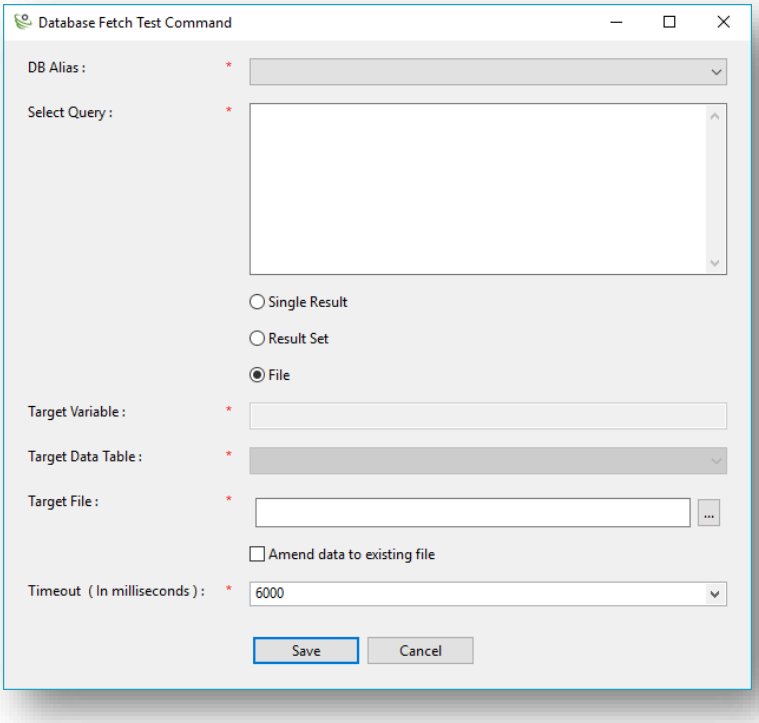


[**Note:** *User needs to configure Database Preferences in the Settings menu under Configuration section.*]

8.13.1 Fetch

This action allows users to extract the data from database by performing the Select Query. The result of the Select Query will be stored either in the Target Variable or Target Data table or Target file depending on user selection.

[**Note:** *After selecting the Fetch action, user needs to click on*  *icon in the Value column and mention* ***DB Alias***, ***Select Query & Target selection*** *details in the dialog box as shown below.*]



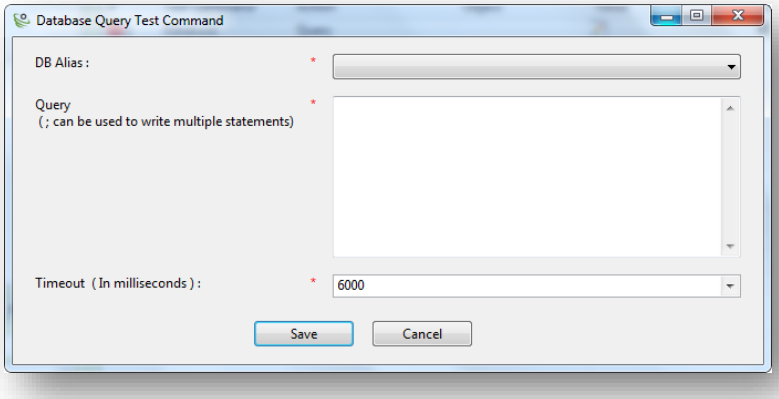
DB Alias	The Alias of the database.

Select Query	The Select Query which needs to be executed in order to fetch data.
Single Result	Select Single Result to store output generated from the Select Query to targeted variable.
Result Set	Select Result Set to store output generated from the Select Query to Targeted data table.
File	Select File to store output generated from the Fetch command to a .csv file.
Target Variable	The name of the variable which would store the output generated from the Select Query
Target Data table	The name of the Data table which would store the output generated from the Select Query.
Target File	The name of the file which would store the output generated from the Fetch command.
Amend data to existing file	User can amend the data into existing file by enabling this option.
Timeout (In milliseconds)	User can specify database timeout period in milliseconds to control script behavior better. Default Timeout would be 6000 milliseconds.

8.13.2 Query

This action allows users to manipulate the data stored in the database with the help of Insert, Update and Delete queries.

[Note: *After selecting the Query action, user needs to click on*  *icon in the Value column and mention* ***DB Alias*** *and* ***Query*** *details in the dialog box as mentioned below.*]



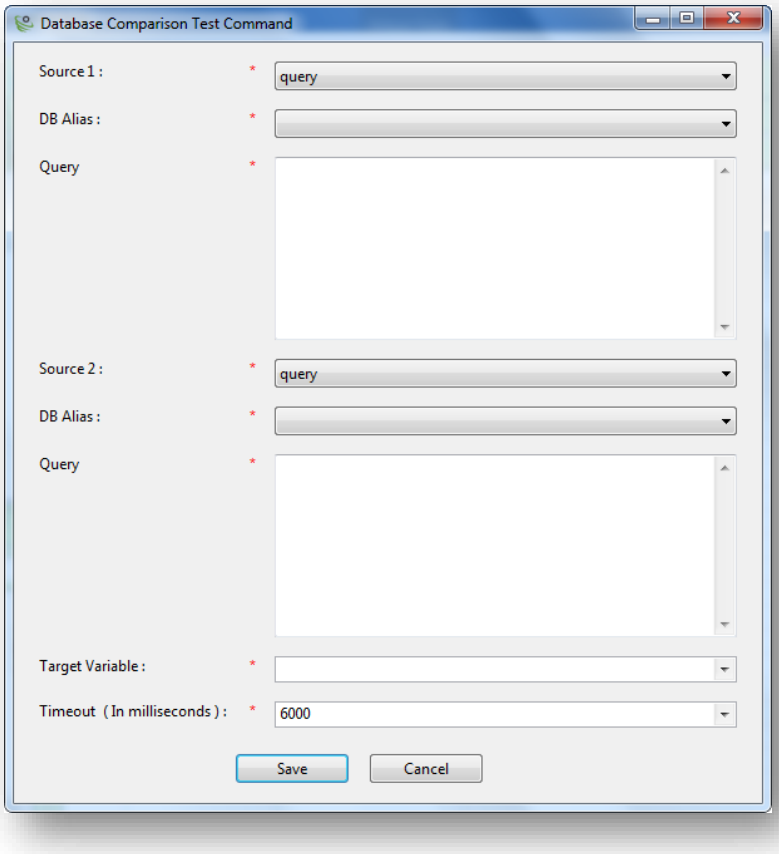
DB Alias	The Alias of the database.

Query	Enter the Query which needs to be executed.
Timeout (In milliseconds)	User can specify database timeout period in milliseconds to control script behavior better. Default Timeout would be 6000 milliseconds.

8.13.3 Comparison

This action allows users to compare query to query, file to file as well as query to file. The result of the Comparison will be stored in the targeted variable.

[**Note:** *After selecting the* Comparison *action, user needs to click on*  *icon in the Value column and mention* ***Source 1&2***, ***DB Alias***, ***Query & Target Variable*** *details in the dialog box as shown below*.]

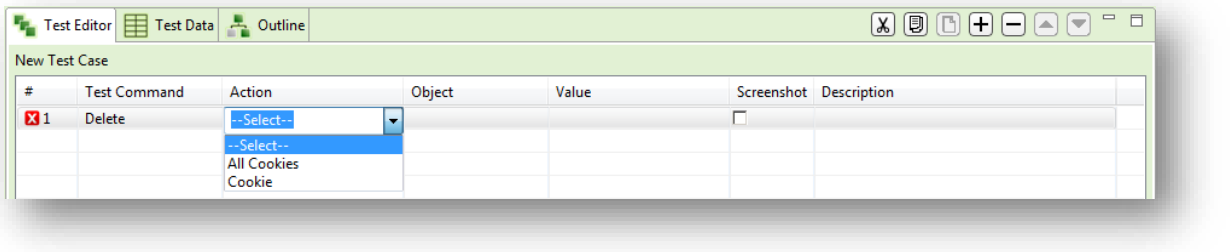


Source 1	Select "Query" or "File" to compare.
Source 2	Select "Query" or "File" to compare.

DB Alias	The Alias of the database.
Query	Enter the Query which needs to be executed.
Target Variable	The name of the variable which would store the output generated from the Select Query.
Timeout (In milliseconds)	User can specify database timeout period in milliseconds to control script behavior better. Default Timeout would be 6000 milliseconds.

8.14Delete

Delete test command allows users to delete Cookie[s] of a web page.



8.14.1 All Cookies

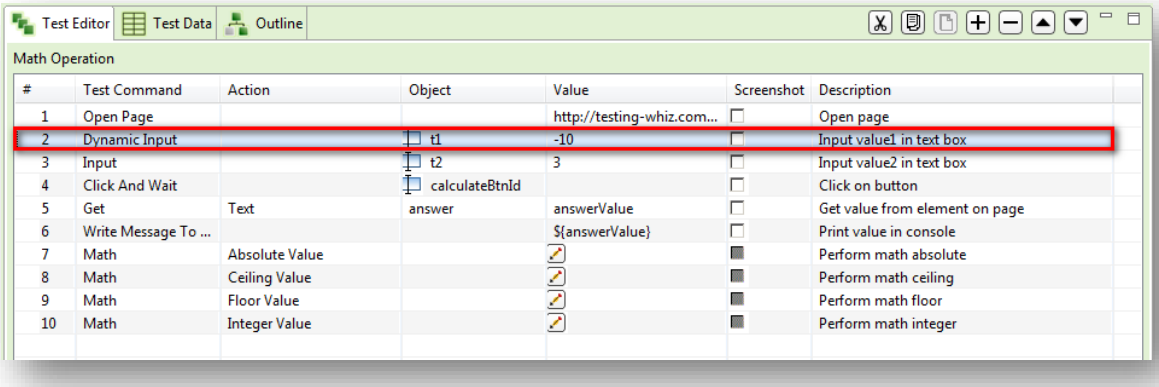
This action allows users to delete all the cookies of all the web pages used.

8.14.2 Cookie

This action allows users to delete specific cookies of a web page.

8.15 Dynamic Input

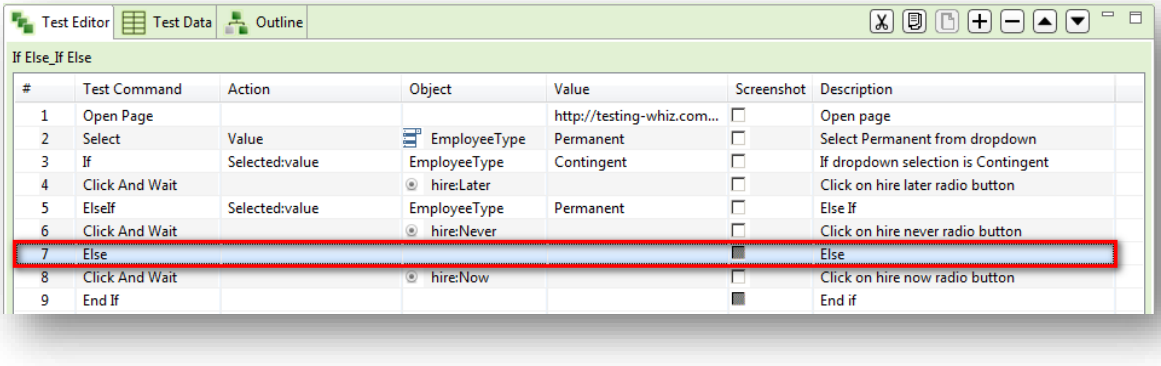
Dynamic Input test command enables users to pass a dynamic value inside the script, through an input box and make use of it further inside the script.



#	Test Command	Action	Object	Value	Screenshot	Description
1	Open Page			http://testing-whiz.com...		Open page
2	Dynamic Input		t1	-10		Input value1 in text box
3	Input		t2	3		Input value2 in text box
4	Click And Wait		calculateBtnId			Click on button
5	Get	Text	answer	answerValue		Get value from element on page
6	Write Message To ...			${answerValue}		Print value in console
7	Math	Absolute Value				Perform math absolute
8	Math	Ceiling Value				Perform math ceiling
9	Math	Floor Value				Perform math floor
10	Math	Integer Value				Perform math integer

8.16Else

Else test command allows users to execute a step for an otherwise condition.

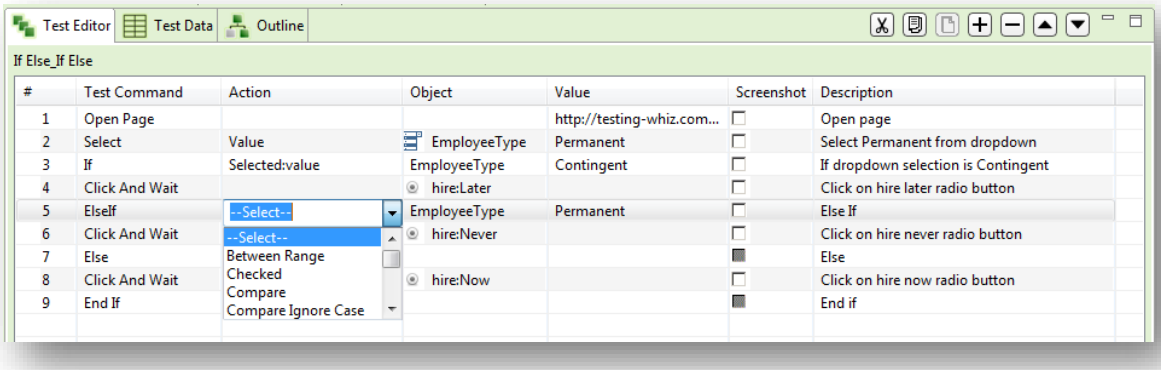


#	Test Command	Action	Object	Value	Screenshot	Description
1	Open Page			http://testing-whiz.com...		Open page
2	Select	Value	EmployeeType	Permanent		Select Permanent from dropdown
3	If	Selected:value	EmployeeType	Contingent		If dropdown selection is Contingent
4	Click And Wait		hire:Later			Click on hire later radio button
5	ElseIf	Selected:value	EmployeeType	Permanent		Else If
6	Click And Wait		hire:Never			Click on hire never radio button
7	Else					Else
8	Click And Wait		hire:Now			Click on hire now radio button
9	End If					End if

[**Note:** *This test command does not contain any Action.*]

8.17ElseIf

ElseIf test command allows users to execute another condition to be tested when all the other conditions of the loop above it are not satisfied.



8.17.1 Text

This action allows users to verify whether the specified text is present on a page or not. The action will be performed for all the text on the page. The check will be performed by matching the case of the text value specified. Text with special symbols will not be considered.

For e.g. "hello" will be considered different from HELLO.

8.17.2 Title

This action allows users to verify whether the title of a page has the specified value or not.

8.17.3 Checked

This action allows users to verify whether the checkbox is checked or selected.

8.17.4 Unchecked

This action allows users to verify whether the checkbox is unchecked or de-selected.

8.17.5 Visible

This action allows users to verify whether a specific object is visible on the page or not.

8.17.6 Invisible

This action allows users to verify whether a specific object is invisible/hidden on the page or not.

8.17.7 Enabled

This action allows users to verify whether the object (links, buttons etc.) is enabled on the page.

8.17.8 Disabled

This action allows users to verify whether the object (links, buttons etc.) is disabled on the page.

8.17.9 Selected:index

This action allows users to verify whether the option of a specified index is selected in the dropdown list.

8.17.10 Selected:value

This action allows users to verify whether the option of a specified value is selected in the dropdown list.

8.17.11 Text:value

This action allows users to verify whether an object has a specified value or not. This action can also be utilized by taking value from the text box.

For e.g. when values in the textbox are automatically populated from a database, user can check/verify these values by taking id or object of the textbox.

8.17.12 Exists

This action allows users to verify whether the object exists on the page or not.

8.17.13 Compare

This action allows users to perform comparison between two stings i.e. verify whether two strings are equal or not. The Compare action will take the case sensitivity of the Strings into consideration.

8.17.14 Compare Ignore Case

This action works in the similar manner as Compare action but with little enhancement. This action will ignore the case sensitivity of the Strings at the time of comparison.

8.17.15 isBlankOrNull

This action allows users to verify whether the value of a Variable is Null and not.

8.17.16 Contains

This action allows users to determine whether a string contains a given sub string.

8.17.17 URL Reachable

This action allows users to verify if a supplied URL in value column is a valid URL or not.

8.17.18 Image

This action allows users to compare two images with URL to URL, File to File and URL to file comparison. It will validate as per the behavior of ElseIf-not command.

8.17.19 Less than

This action allows users to test whether a value is less than another value.

8.17.20 Less than or equal to

This action allows users to test whether two numeric values are less than or equal to each other.

8.17.21 Greater than

This action allows users to test whether a value is greater than another value or not.

8.17.22 Greater than or equal to

This action allows users to test whether two numeric values are equal to each other.

8.17.23 Equal to

This action allows users to identify two values and returns true if the values on both sides are equal to one another.

8.17.24 Not equal to

This action allows users to check if the value of two operands are equal or not.

8.17.25 Between Range

This test command allows users to validate whether a number lies between the specified range.

[**Note:** *User needs to specify the Test Value, Range Start, Range End and Target Variable in the Value tab of this command.*]

8.17.26 Current Page URL

This action allows users to evaluate the current page URL on the screen.

8.18 Encrypt

8.18.1 AES String

This action of Encrypt test command allows users to encrypt their information string into AES String. This test command saves the converted AES String into a variable.

8.18.2 MD5 String

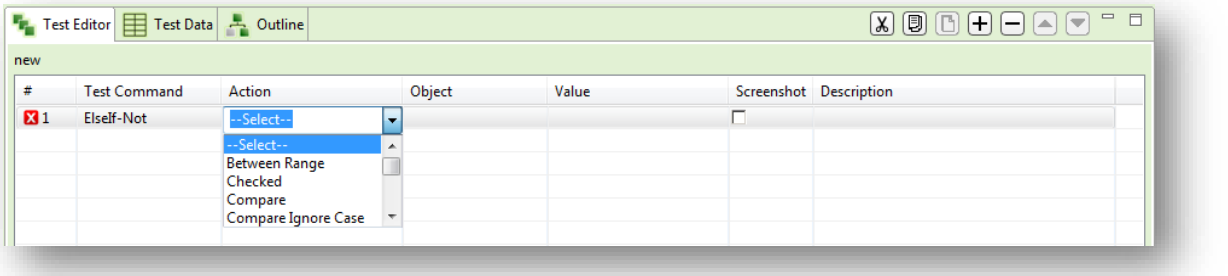
This action of Encrypt test command allows users to encrypt their information string into MD5 String. This test command saves the converted MD5 String into a variable.

8.18.3 SHA256 String

This action of Encrypt test command allows users to encrypt their information string into SHA256 String. This test command saves the converted SHA256 String into a variable.

8.19ElseIf-Not

Elseif-Not test command allows users to execute a condition if the action mentioned is not satisfied.



8.19.1 Text

This action allows users to verify whether a specified text is present on a page or not. The action will be performed for all the text on the page. The check will be performed by matching the case of the text value specified. Text with special symbols will not be considered. For e.g. "hello" will be considered different from HELLO.

8.19.2 Title

This action allows users to verify whether the title of the page has a specified value or not.

8.19.3 Checked

This action allows users to verify whether the checkbox is checked or selected.

8.19.4 Unchecked

This action allows users to verify whether the checkbox is unchecked or de-selected.

8.19.5 Visible

This action allows users to verify whether a specific object is visible on the page or not.

8.19.6 Invisible

This action allows users to verify whether a specific object is invisible/hidden on the page or not.

8.19.7 Enabled

This action allows users to verify whether the object (links, buttons etc.) is enabled on the page.

8.19.8 Disabled

This action allows users to verify whether the object (links, buttons etc.) is disabled on the page.

8.19.9 Selected:index

This action allows users to verify whether the option of a specified index is selected in the dropdown list.

8.19.10 Selected:value

This action allows users to verify whether the option of a specified value is selected in the dropdown list.

8.19.11 Text:value

This action allows users to verify whether an object has a specified value or not. This test command can also be utilized by taking value from the text box.

For e.g. when values in the textbox are automatically populated from a database, user can check/verify these values by taking id or object of the textbox.

8.19.12 Exists

This action allows users to verify whether the object exists on a page or not.

8.19.13 Compare

The Compare action allows users to perform the Comparison between two stings i.e. verify whether two strings are equal or not. The Compare action will take the case sensitivity of the Strings into consideration.

8.19.14 Compare Ignore Case

The Compare Ignore Case action will work in the similar manner as Compare action but with little enhancement. This action will ignore the case sensitivity of the Strings at the time of comparison.

8.19.15 IsBlankOrNull

The isBlankOrNull Test command enables users to verify whether the value of a Variable is Null and not.

8.19.16 Contains

The Contains action allows users to determine whether a string contains a given sub string.

8.19.17 URL Reachable

This action allows users to verify if a supplied URL in value column is a valid URL or not.

8.19.18 Image

This action allows users to compare two images with URL to URL, File to File and URL to file comparison. This command will validate as per the behavior of ElseIf-not command.

8.19.19 Less than

This action allows users to test whether a value is less than another value.

8.19.20 Less than or equal to

This action allows users to test whether two numeric values are less than or equal to each other.

8.19.21 Greater than

This action allows users to test whether a value is greater than another value or not.

8.19.22 Greater than or equal to

This action allows users to test whether two numeric values are equal to each other.

8.19.23 Equal to

This action allows users to identify two values and returns true if the values on both sides are equal to one another.

8.19.24 Not equal to

This action allows users to check if the value of two operands are equal or not.

8.19.25 Between Range

This test command allows users to validate whether a number lies between the specified ranges.

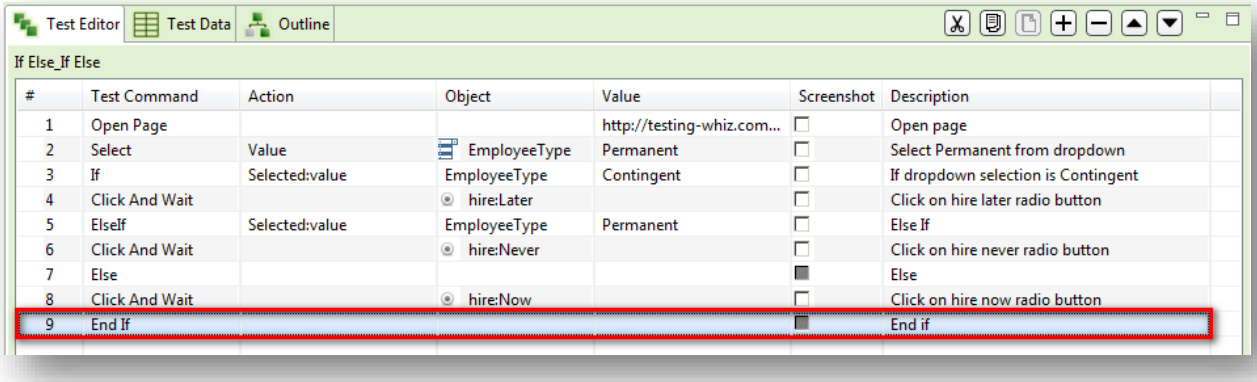
[**Note:** *User needs to specify the Test Value, Range Start, Range End and Target Variable in the Value tab of this command.*]

8.19.26 Current Page URL

This action allows users to evaluate the current page URL on the screen.

8.20 End If

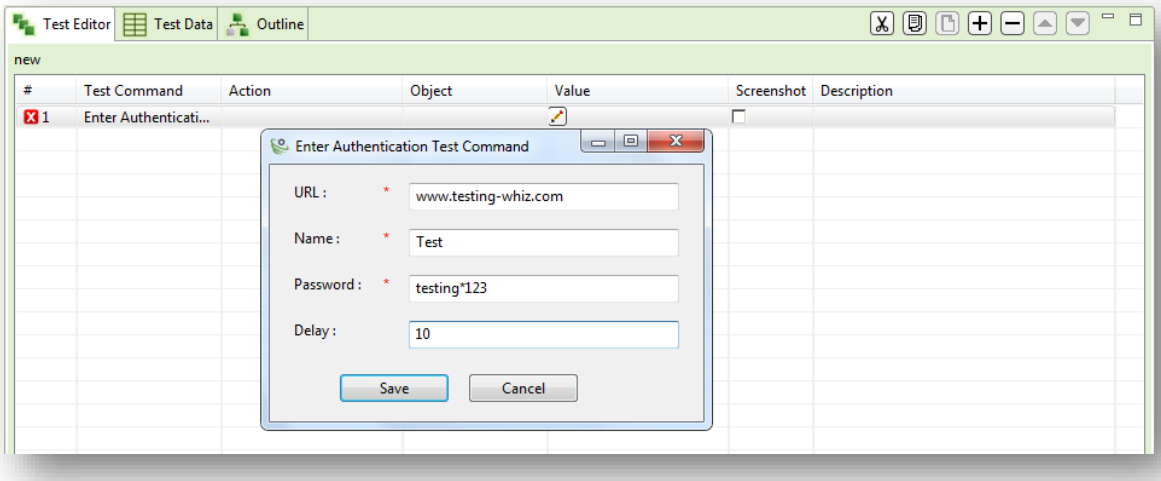
End If test command allows users to end the loop of If conditions.



[**Note:** *This test command does not contain any Action.*]

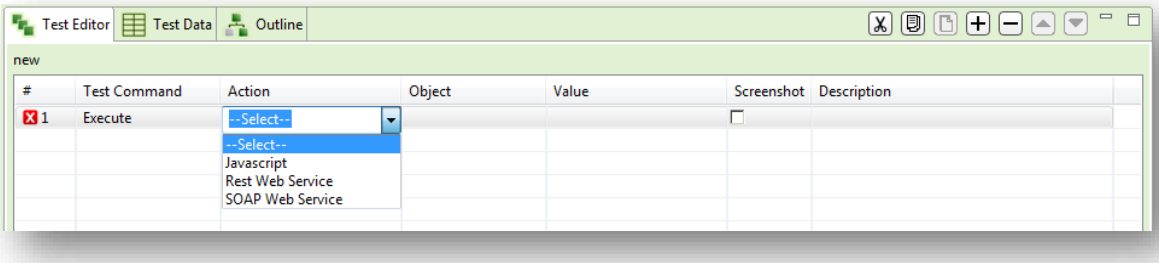
8.21 Enter Authentication

Enter Authentication test command allows users to provide authentication on a given application. This can prevent unauthorized access of applications.



8.22 Execute

Execute test command allows users to execute a particular script.



8.22.1 JavaScript

This action allows users to execute JavaScript through a simple JavaScript code. Also users can access the variable values using the format ${*variable*}.

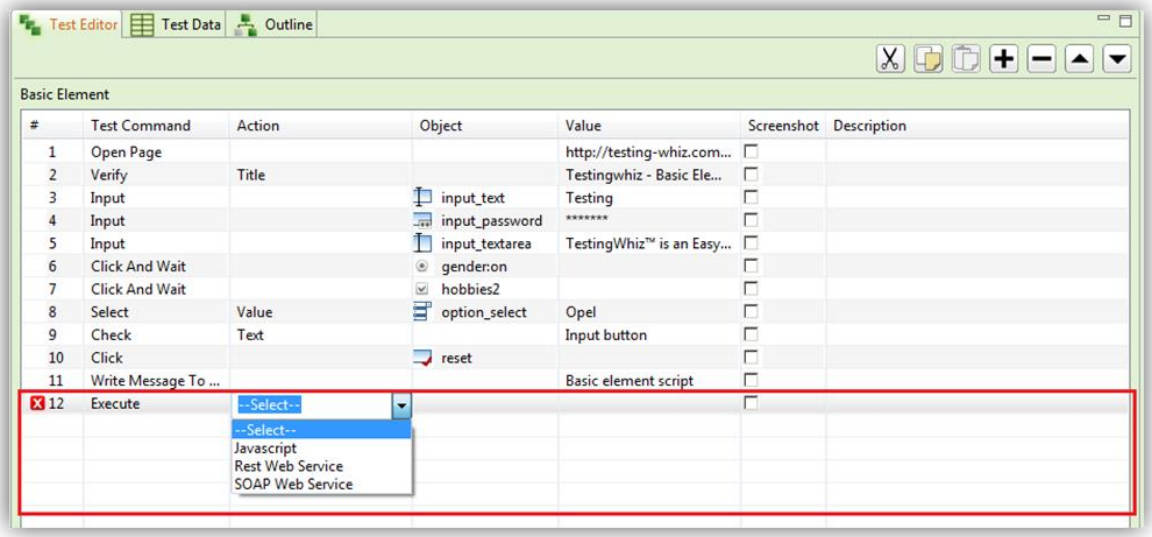
8.22.2 RESTful Web Service

This command allows users to test RESTful Web Services with a single test command, and store results in a variable.

Refer Section – **RESTful Web Services Testing** to know more

8.22.3 SOAP Web Service

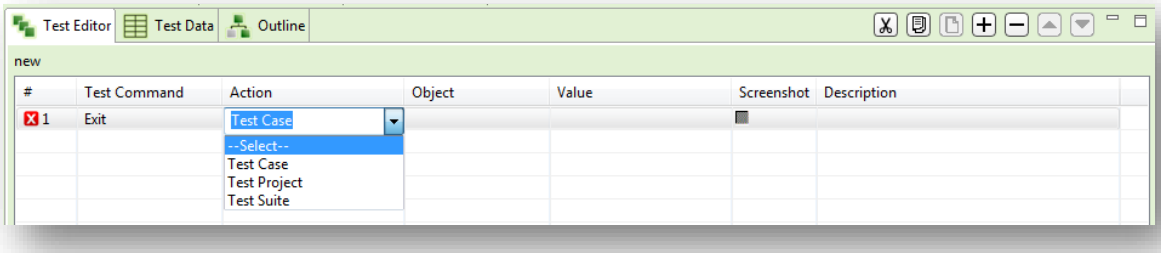
This command allows users to test SOAP Web Services with a single test command, and store results in a variable.



Refer Section – **SOAP Web Services Testing** to know more.

8.23 Exit

Exit test command allows users to exit from current Test Case/Test Suite/Test Project.



8.23.1 Test Case

This action allows users to exit from a Test Case and switch to the next consecutive Test Case.

8.23.2 Test Project

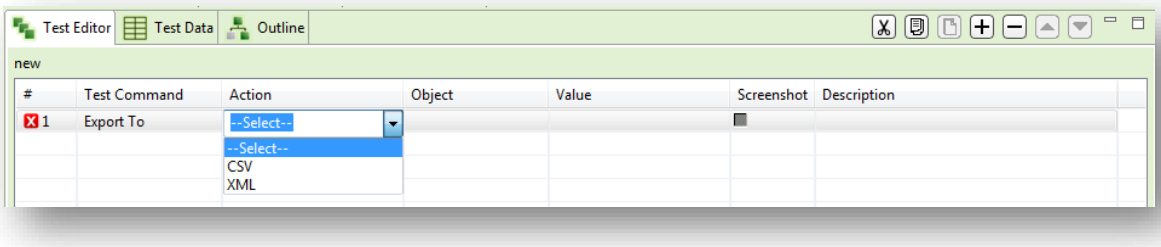
This action allows users to exit from a Test Project and switch to the next consecutive Test Project.

8.23.3 Test Suite

This action allows users to exit from a Test Suite and switch to the next consecutive Test Suite.

8.24 Export To

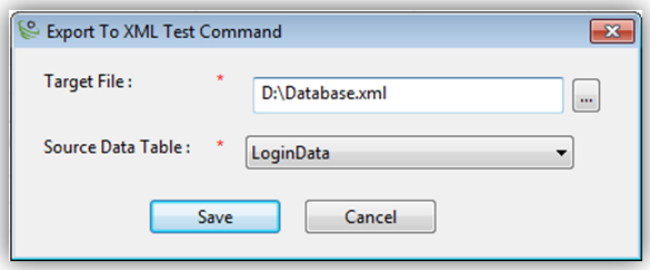
The Export to test command enables users to export the data stored in Data tables to .XML and .CSV file.



8.24.1 XML

The XML action enables users to export the data into the .XML file format.

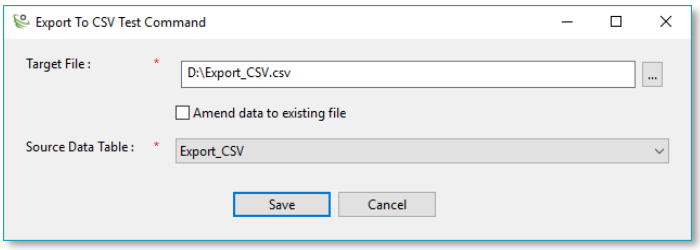
[**Note:** *After selecting the XML action, user needs to provide* ***Target File*** *in .XML format in which the exported file should be stored and also provide Source Data table details as shown below.*]



8.24.2 CSV

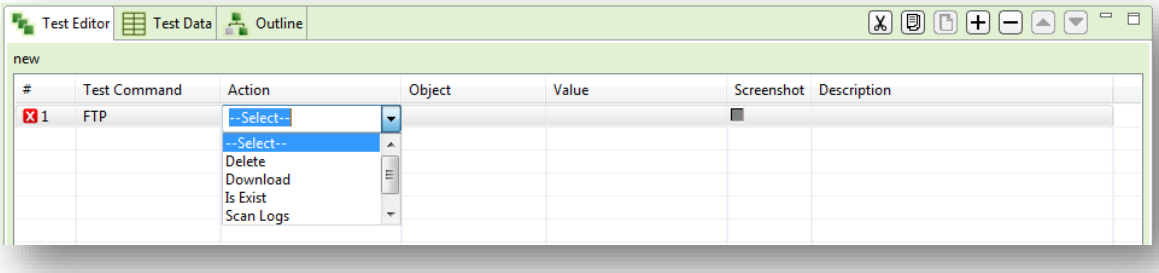
The CSV action enables users to export the data into the CSV file format.

[**Note:** *After selecting the CSV action, user needs to provide* ***Target File*** *in .CSV format in which the exported file should be stored and also provide* ***Source Data table*** *details as shown below. User can also amend the data into file by enabling 'Amend data to existing file' option.*]]



8.25 FTP

FTP command allows users to integrate and access FTP file / server for testing.



8.25.1 Upload

This action allows users to upload a local file to FTP server.

8.25.2 Is Exist

This action allows users to validate the presence of the specified file on FTP server.

8.25.3 Download

This action allows users to download a file from the FTP server to local machine.

8.25.4 Delete

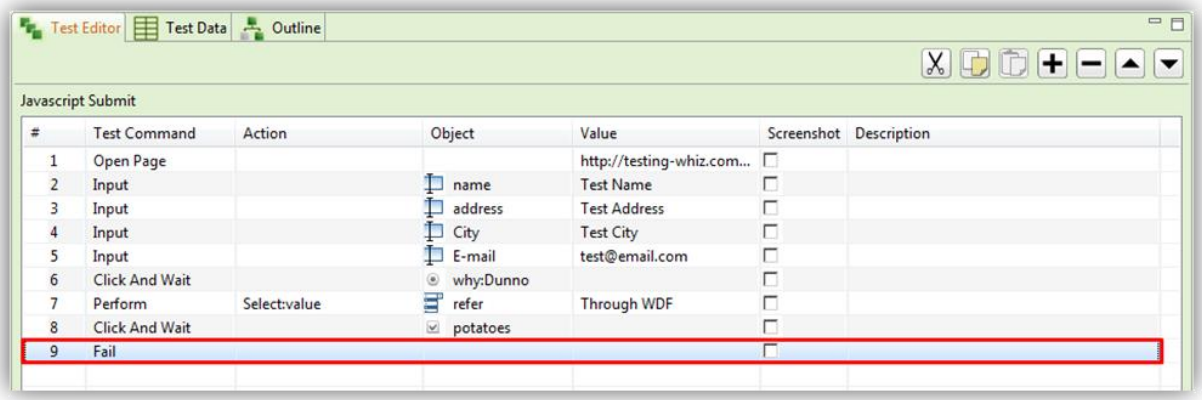
This action allows users to delete a file from the FTP server.

8.22.5 Scan Logs

This action allows user to scan logs on remote Linux server and find know the occurrences of anomalies.

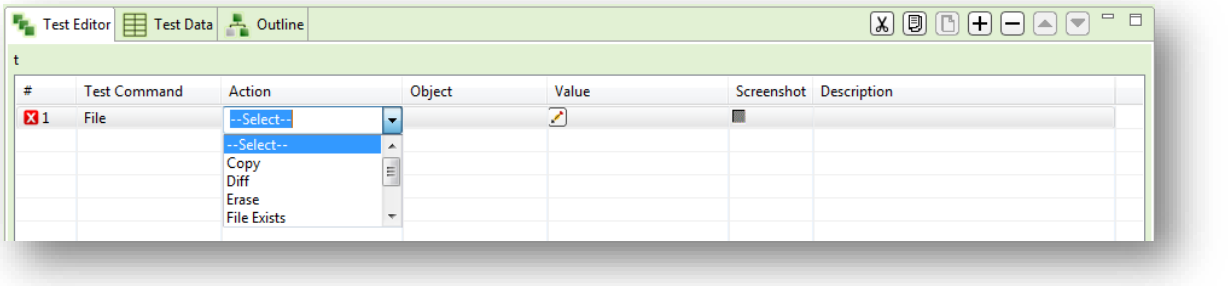
8.26 Fail

Fail test command allows users to introduce a user defined failure of a Test Step/Test Case on some condition evaluation of a test case. Users can define their own failure statement in the value column of Fail test command.



8.27 File

This test command allows the user to perform the file actions like following.



8.27.1 Copy

This action allows the user to copy the file between the directories of your local system and validate.

8.27.2 Move

This action allows the user to move the file between the directories of your local system and validate.

8.27.3 Save

This action allows the user to download any file from the web to any local system.

8.27.4 Erase

This action allows the user to delete any file from the local system.

8.27.5 Search String

This action allows the users to search a particular/given string in a particular file. This test command will return the count number of occurrences of the string in that file.

8.27.6 File Exists

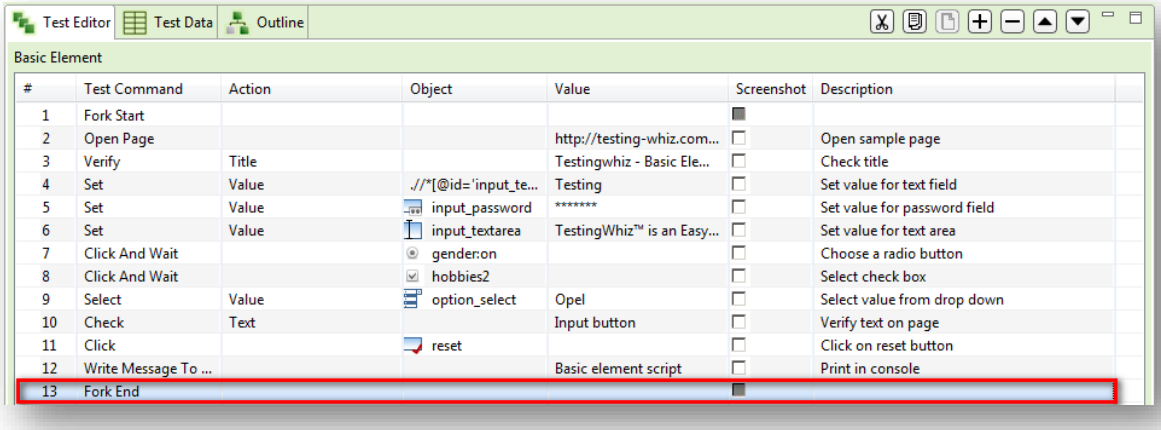
This action allows users to validate the presence of the specified file in the local system. It requires absolute file path and a variable to store the Boolean result.

8.27.7 Diff

Users can validate the difference between two CSV files. Users need to specify FTP server alias, absolute file path for file1 and file2, Target CSV file path to store the differences, and a timeout according to the complexity of files to be compared.

8.28 Fork End

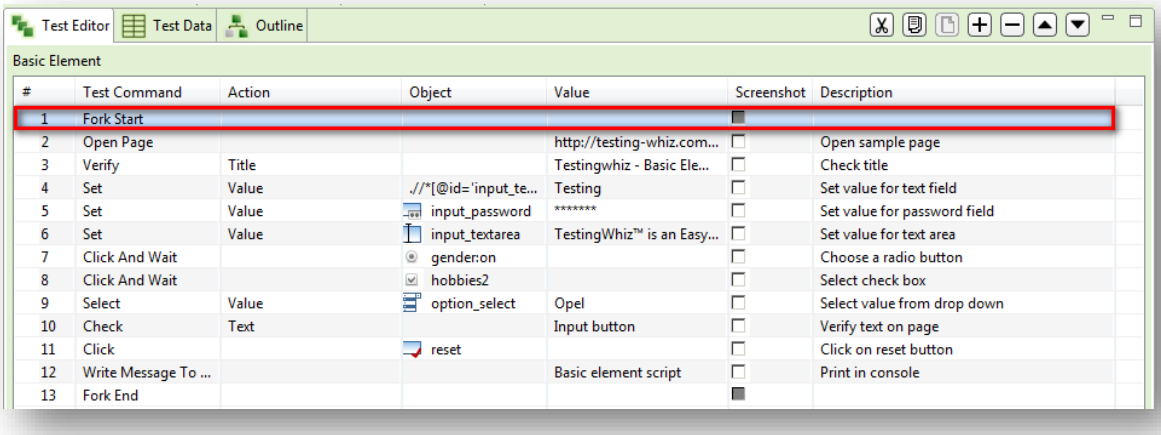
This command allows users to End a command.



[**Note:** *This test command does not contain any Action.*]

8.29 Fork Start

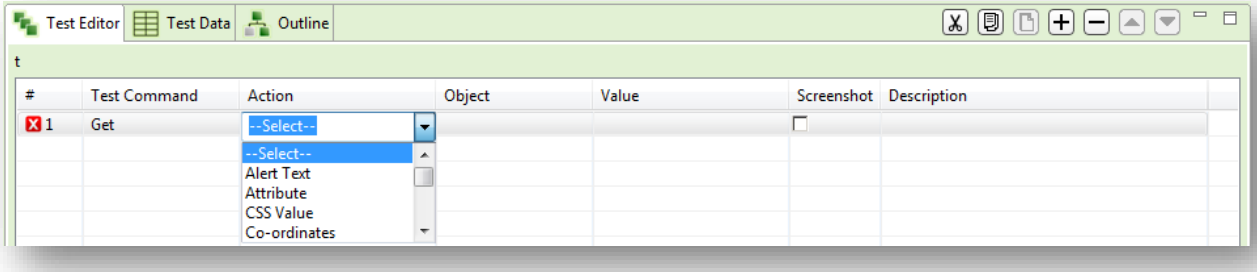
This command allows users to Start command.



[**Note:** *This test command does not contain any Action.*]

8.30 Get

Get test command allows users to fetch/get the attribute of an object, table row count, table column count, table cell data.



[**Note:** *User has to use a variable to store the fetched value and display it on the report*.]

8.30.1 Text

Text action allows users to get the text of a textbox.

8.30.2 Value

This action allows users to get the value of any object.

8.30.3 Table Row Count

This action allows users to get the total row count of a particular table.

8.30.4 Table Column Count

This action allows users to get the total column count of a particular table.

8.30.5 Attribute

This action allows users to get any attribute of an object.

8.30.6 Table Cell Data

This action allows users to get data of a particular table cell.

8.30.7 Title

Title action allows users to get the Title of a particular website or URL.

8.30.8 Current Page URL

Current Page URL action allows users to get the URL of the current website.

8.30.9 Alert Text

This action allows users to get the Text value of an Alert Pop-up.

[Note: *This operation will not work with Android and iOS mobile browsers.*]

8.30.10 Page Source

This action allows users to extract the source of the current URL which is opened in browser.

8.30.11 Table

This action allows users to extract the entire data of the specified table object into Datatable.

8.30.12 Elements

This action allows users to get elements of particular type from the specified web page as well as from specified object and allows to store to Data table.

8.30.13 ExecBrowserName

This action allows the users to get the browser name and version under execution. This will get stored it into a global variable.

8.30.14 Selected:value

This action allows users to get the selected value of drop down list.

8.30.15 Co-ordinates

This action allows users to capture X and Y co-ordinates for a particular web-element. To capture the co-ordinates, it requires an object reference and two variable names to store the X and Y co-ordinate values.

8.30.16 CSS Value:

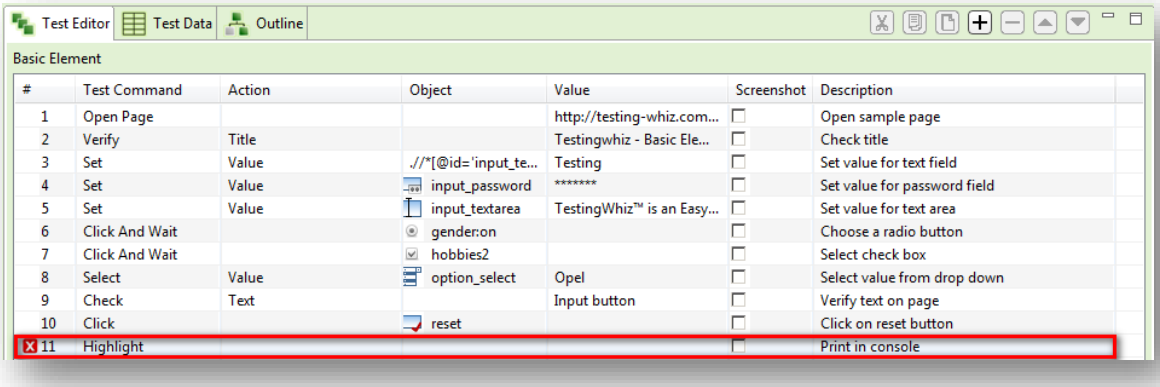
This action allows users to fetch value of a particular CSS property assigned to any object present on the website. Users need to specify CSS property and a variable name to store the fetched values from the website in the value column of the command.

8.30.17 Dropdown Values:

This action allows users to fetch all the dropdown values present inside the specified dropdown object. The dropdown object should be of Select tag. Users need to specify datatable to store the fetched values.

8.31 Highlight

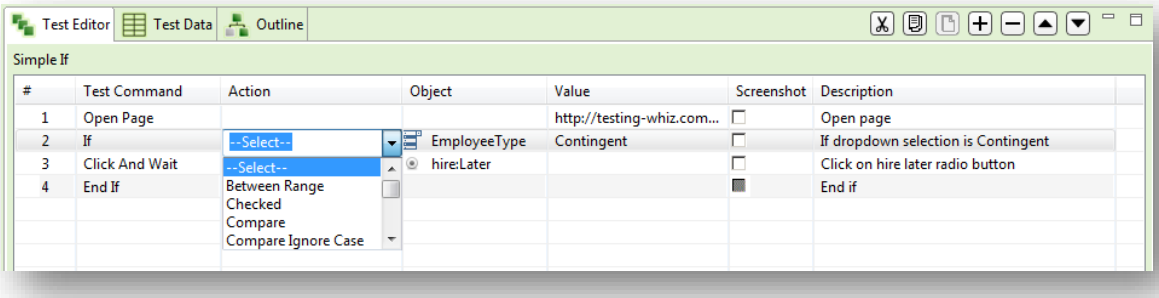
Highlight test command allows users to highlight a particular object in a page.



[**Note:** *This test command does not contain any Action.*]

8.32 If

If test command allows users to check for specific conditions before executing a test step.



8.32.1 Text

This action allows users to verify whether the specified text is present on a page or not. The action will be performed for all the text on the page. The check will be performed by matching the case of the text value specified. Text with special symbols will not be considered. For e.g. "hello" will be considered different from HELLO.

8.32.2 Title

This action allows users to verify whether the title of the page has the specified value or not.

8.32.3 Checked

This action allows users to verify whether the checkbox is checked or selected.

8.32.4 Unchecked

This action allows users to verify whether the checkbox is unchecked or de-selected.

8.32.5 Visible

This action allows users to verify whether a specific object is visible on the page or not.

8.32.6 Invisible

This action allows users to verify whether a specific object is invisible/hidden on the page or not.

8.32.7 Enabled

This action allows users to verify whether the object (links, buttons etc.) is enabled on the page.

8.32.8 Disabled

This action allows users to verify whether the object (links, buttons etc.) is disabled on the page.

8.32.9 Selected:index

This action allows users to verify whether the option of the specified index is selected in the dropdown list.

8.32.10 Selected:value

This action allows users to verify whether the option of the specified value is selected in the dropdown list.

8.32.11 Text:value

This action allows users to verify whether an object has the specified value or not. This test command can also be utilized by taking value from the text box.

For e.g. when values in the textbox are automatically populated from a database, user can check/verify these values by taking id or object of the textbox.

8.32.12 Exists

This action allows users to verify whether the object exists on the page or not.

8.32.13 Compare

The Compare action users to perform the Comparison between two stings i.e. verify whether two strings are equal or not. The Compare action will take the Case Sensitivity of the Strings into consideration.

8.32.14 Compare Ignore Case

The Compare Ignore Case action will work in the similar manner as Compare action but with little enhancement. This action would ignore the Case Sensitivity of the Strings at the time of comparison.

8.32.15 IsBlankOrNull

The isBlankOrNull Test command allows users to verify whether the value of a Variable is Null and not.

8.32.16 Contains

The Contains action allows users to remove the leading and trailing blanks in the String.

8.32.17 URL Reachable

This action allows users to verify if a supplied URL in value column is a valid URL or not.

8.32.18 Image

This action allows users to compare two images with URL to URL, File to File and URL to file comparison.

This command will validate as per the behavior of If command.

8.32.19 Less than

This action allows users to test whether a value is less than another value.

8.32.20 Less than or equal to

This action allows users to test whether two numeric values are less than or equal to each other.

8.32.21 Greater than

This action allows users to test whether a value is greater than another value or not.

8.32.22 Greater than or equal to

This action allows users to test whether two numeric values are equal to each other.

8.32.23 Equal to

This action allows users to identify two values and returns true if the values on both sides are equal to one another.

8.32.24 Not equal to

This action allows users to check if the value of two operands are equal or not.

8.32.25 Between Range

This test command allows users to validate whether a number lies between the specified range.

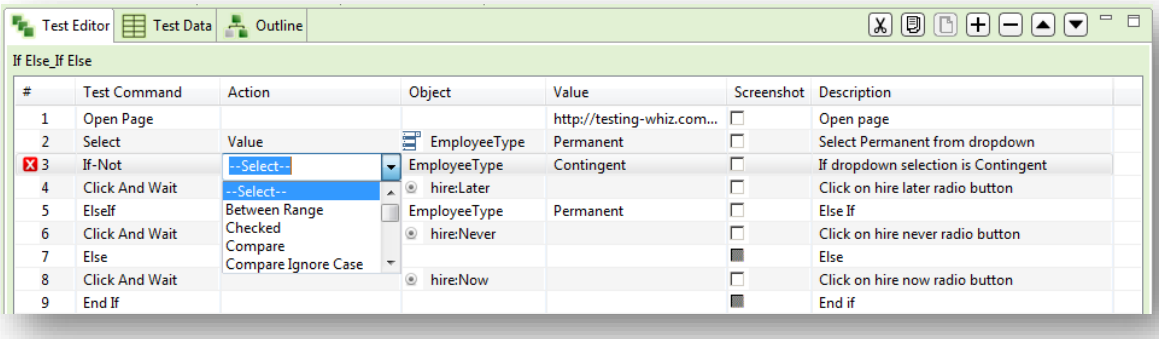
[**Note**: *User needs to specify the Test Value, Range Start, Range End and Target Variable in the Value tab of this command*.]

8.32.26 Current Page URL

This action allows users to evaluate the current page URL on the screen.

8.33 If-Not

If-Not test command allows users to check for conditions before executing a test step.



8.33.1 Text

This action allows users to verify whether the specified text is present on a page or not. The action will be performed for all the text on the page. The check will be performed by matching the case of the text value specified. Text with special symbols will not be considered. For e.g "hello" will be considered different from HELLO.

8.33.2 Title

This action allows users to verify whether the title of the page has the specified value or not.

8.33.3 Checked

This action allows users to verify whether the checkbox is checked or selected.

8.33.4 Unchecked

This action allows users to verify whether the checkbox is unchecked or de-selected.

8.33.5 Visible

This action allows users to verify whether a specific object is visible on the page or not.

8.33.6 Invisible

This action allows users to verify whether a specific object is invisible/hidden on the page or not.

8.33.7 Enabled

This action allows users to verify whether the object (links, buttons etc.) is enabled on the page.

8.33.8 Disabled

This action allows users to verify whether the object (links, buttons etc.) is disabled on the page.

8.33.9 Selected:index

This action allows users to verify whether the option of the specified index is selected in the dropdown list.

8.33.10 Selected:value

This action allows users to verify whether the option of the specified value is selected in the dropdown list.

8.33.11 Text:value

This action allows users to verify whether an object has the specified value or not. This test command can also be utilized by taking value from the text box. For e.g. when values in the textbox are automatically populated from a database, user can check/verify these values by taking id or object of the textbox.

8.33.12 Exists

This action allows users to verify whether the object exists on the page or not.

8.33.13 Compare

The Compare action allows users to perform the Comparison between two stings i.e. verify whether two strings are equal or not. The Compare action will take the Case Sensitivity of the Strings into consideration.

8.33.14 Compare Ignore Case

The Compare Ignore Case action will work in the similar manner as Compare action but with little enhancement. This action would ignore the Case Sensitivity of the Strings at the time of comparison.

8.33.15 IsBlankOrNull

The isBlankOrNull Test command allows users to verify whether the value of the Variable is Null and not.

8.33.16 Contains

The Contains action allows users to remove the leading and trailing blanks in the String.

8.33.17 URL Reachable

This action allows users to verify if a supplied URL in value column is a valid URL or not.

8.33.18 Image

This action allows users to compare two images with URL to URL, File to File and URL to file comparison. This command will validate as per the behavior of If – not command.

8.33.19 Less than

This action allows users to test whether a value is less than another value.

8.33.20 Less than or equal to

This action allows users to test whether two numeric values are less than or equal to each other.

8.33.21 Greater than

This action allows users to test whether a value is greater than another value or not.

8.33.22 Greater than or equal to

This action allows users to test whether two numeric values are equal to each other.

8.33.23 Equal to

This action allows users to identify two values and returns true if the values on both sides are equal to one another.

8.33.24 Not equal to

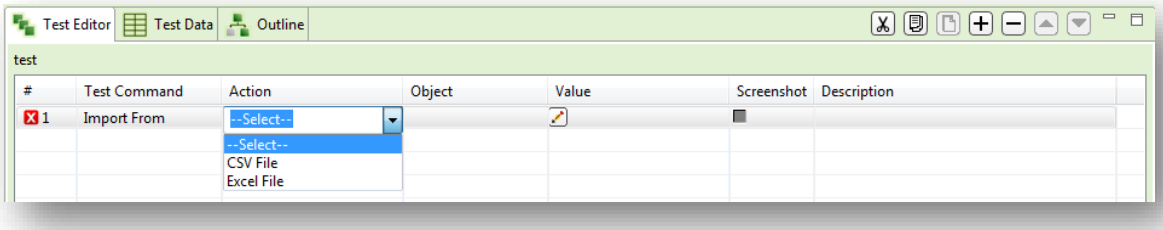
This action allows users to check if the value of two operands are equal or not.

8.33.25 Current Page URL

This action allows users to evaluate the current page URL on the screen.

8.34 Import From

This test command allows users to import data from an Excel or a CSV file dynamically during script execution and utilize them in your test cases.



8.34.1 CSV File

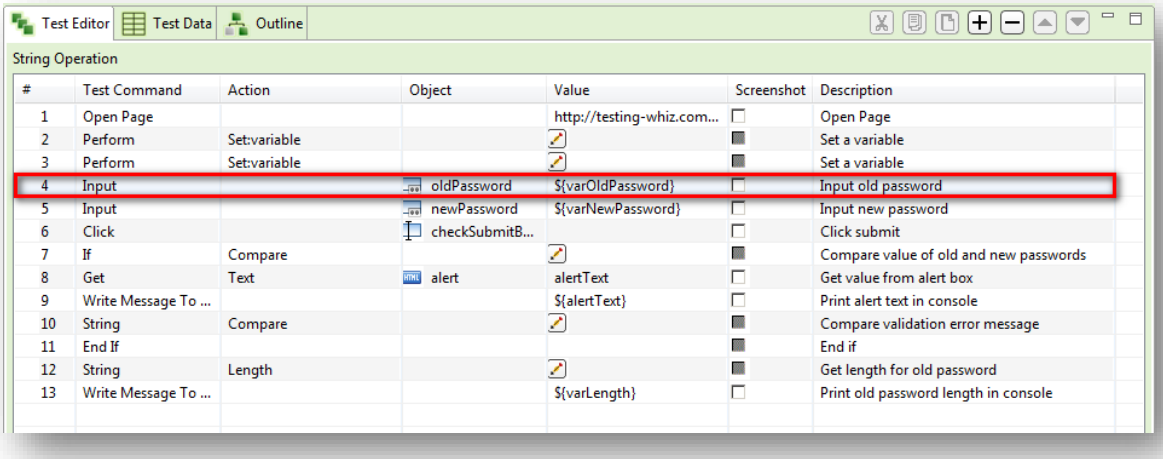
This action allows users to import data from an excel file during runtime. Absolute file path needs to be provided as an input. Delimiter needs to be specified as a separator for column recognition. Datatable to be specified to store the data from the file. Timeout can be specified according to the amount of data user is importing from the file.

8.34.2 Excel File

This action allows users to import data from an excel file during runtime. Absolute file path needs to be provided as an input. All the sheets inside the specified excel file will be populated when "Get Worksheet List" is clicked. User can choose the required sheet to be imported, and datatable to be specified to store the data from the file. Timeout can be specified according to the amount of data user is importing from the file.

8.35 Input

Input test command allows users to set a particular value in a textbox.

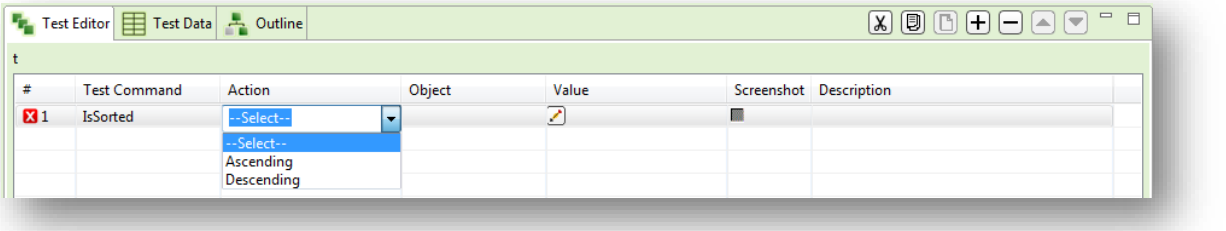


[**Note:** *This test command does not contain any Action.*]

8.36 IsSorted

This command allows users to perform following actions.

[**Note**]: It would work for strings, numeric data & alphanumeric data. For dates and other things, string based comparison will be made.



8.36.1 Ascending

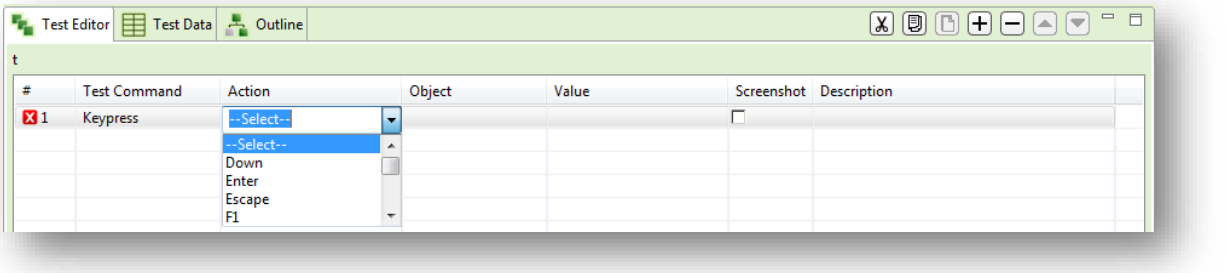
This action allows users to validate sorting in ascending manner of the specified column of a datatable.

8.36.2 Descending

This action allows users to validate sorting in Descending manner of the specified column of a datatable.

8.37 KeyPress

Keypress test command allows users to perform functionalities of various function keys on a web page.



8.37.1 Enter

This action allows users to perform the functionality of an Enter key on an object.

8.37.2 Escape

This action allows users to perform the functionality of an Escape key on an object. User can use this object when they want to escape an alert/message box or any frame/window.

8.37.3 Tab

This action allows users to use the functionality of the Tab key. It will move to the object whose id is defined in the Object column.

8.37.4 Refresh

This action allows users to Refresh and re-load the contents of the web page.

8.37.5 F1

This action allows users to view the Help contents of an application.

8.37.6 F3

This action allows users to access the Search box on the web page.

8.37.7 F6

This action allows users to Move the cursor to the URL bar of the web page.

8.37.8 F10

This action allows users to Move the cursor to the first Menu in the Menu Bar of the web page.

8.37.9 F11

This action allows users to view the Full-screen of a web page. It will hide the URL bar, menu bar, tabs of the web page.

8.37.10 Page Up

This action allows users to use the functionality of Page Up key. It scrolls the page up in the same proportion as the Page Up key functions.

8.37.11 Page Down

This action allows users to use the functionality of Page Down key. It scrolls the page down in the same proportion as the Page Down key functions.

8.37.12 Up

This action allows users to use the functionality of up key wherever scrolling is required on a page. It will move to the object whose id is defined in the Object column.

8.37.13 Down

This action allows users to use the functionality of down key wherever scrolling is required on a page. It will move to the object whose id is defined in the Object column.

8.37.14 Left

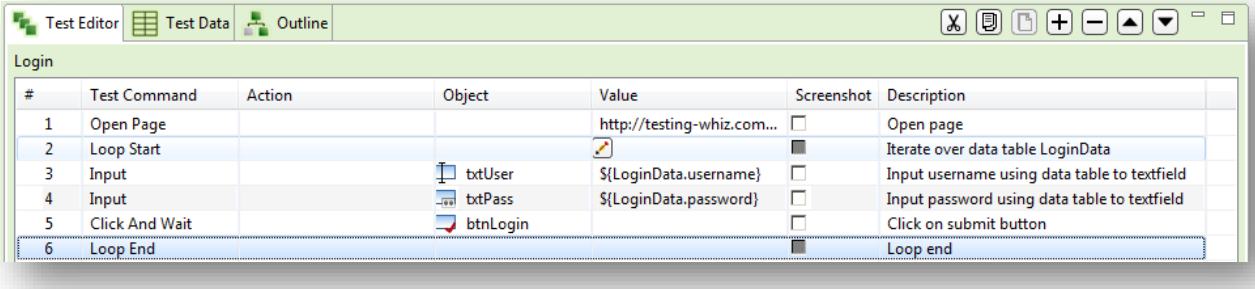
This action allows users to use the functionality of Left key. It will move to the object whose id is defined in the Object column.

8.37.15 Right

This action allows users to use the functionality of Right key. It will move to the object whose id is defined in the Object column.

8.38 Loop End

Loop End test command allows users to end a defined loop of test steps. To execute a loop, user needs to configure it using Data table.

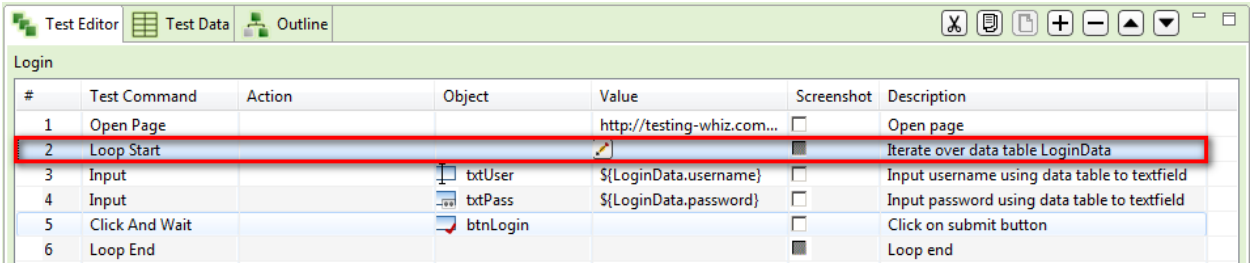


[**Note:** *This test command does not contain any Action*.]

8.39 Loop Start

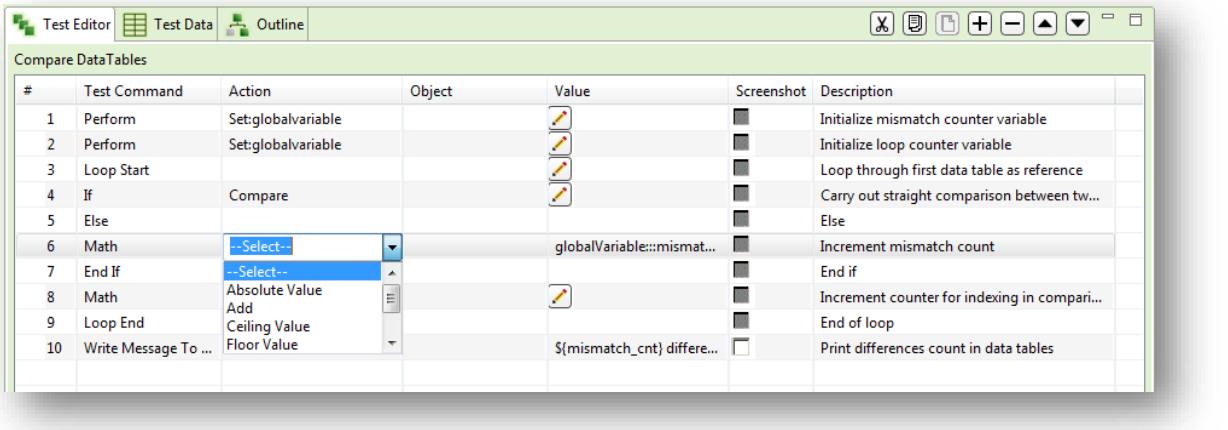
Loop Start test command allows users to start a defined loop of test steps. To execute a loop, user needs to configure it using Data table.

[**Note:** *User needs to provide details of Target Data Table, Start Index and End Index. Also, user can choose between Standard and Custom Loop based on the requirements*.]



8.40 Math

The Math Test command allows users to perform calculation on numeric data by formatting it in a desired format.



8.40.1 Absolute Value

The action allows users to return the absolute value of a double value. If the argument is not negative, the argument is returned. If the argument is negative, the negation of the argument is returned.

In Special cases: If the argument is positive zero or negative zero, the result is positive zero. If the argument is infinite, the result is positive infinity. If the argument is Nan, the result is Nan.]

8.40.2 Integer Value

The Integer action allows users to return the value of this double as an integer (by casting to type integer).

8.40.3 Floor Value

The Floor action allows users to return the largest (closest to positive infinity) double value that is less than or equal to the argument and is equal to a mathematical integer.

In Special cases: If the argument value is already equal to a mathematical integer, then the result is the same as the argument. If the argument is Nan or an infinity or positive zero or negative zero, then the result is the same as the argument.

8.40.4 Ceiling Value

The Celling action allows users to return the smallest (closest to negative infinity) double value that is greater than or equal to the argument and is equal to a mathematical integer.

In Special cases: If the argument value is already equal to a mathematical integer, then the result is the same as the argument. If the argument is Nan or an infinity or positive zero or negative zero, then the result is the same as the argument. If the argument value is less than zero but greater than -1.0, then the result is negative zero.

8.40.5 Add

The Add action returns the sum of supplied values, or variables, to the user.

8.40.6 Subtract

The Subtract action returns the difference of supplied values or variables, to the user.

8.40.7 Number Between

Test action to generate a random number between a specified range. Users need to specify Range start and Range end number and a variable to store the generated number. For e.g. Range Start = 1, Range End = 100 and Target Variable = ran_num. TestingWhiz will generate a random number between 1 and 100 i.e for e.g. 57 and store it in the variable "ran_num".

8.40.8 Multiply

The Multiply action returns the product value of supplied numbers, or variable values in a Target Variable name specified.

8.40.9 Divide

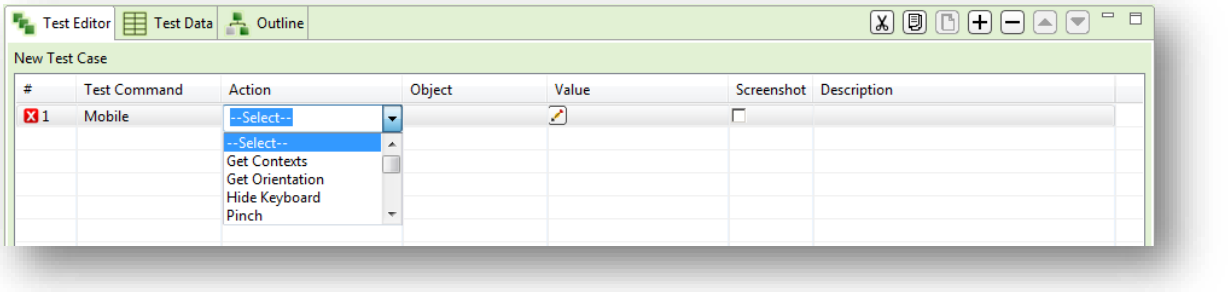
The Divide action returns the quotient value of supplied numbers, or variable values in a Target Variable name specified.

8.40.10 Remainder

The Remainder action returns the remainder value of supplied numbers, or variable values in a Target Variable name specified.

8.41 Mobile

The Mobile Test command allows users to perform actions on the Mobile or on a Simulator.



8.41.1 Tap

This action allows the user to Tap on the center of screen.

8.41.2 Tap by Co-ordinates

This action allows users to perform a tap action in mobile devices/emulators using screen X and Y co-ordinates. Users need to provide X and Y co-ordinates in the value column of this test command. Users can make use of Get >> Co-ordinates to fetch the co-ordinates of any web-element.

8.41.3 Swipe

This action allows the user to perform swipe gesture across the screen i.e. Left, Right, Up and Down.

8.41.4 Zoom on Element

This action allows the user to zoom on a particular element on the screen.

8.41.5 Zoom on Location

This action allows the user to zoom on a particular location on the screen.

8.41.6 Hide Keyboard

This action allows the user to hide/minimize the keyboard which would be visible on the screen.

8.41.7 Pinch

This action allows the user to zoom out/pinch gesture on screen.

8.41.8 Reset App

This action allows the user to reset the particular application which is running for the session and perform further actions ahead.

8.41.9 Rotate

This action allows the user to rotate the screen to portrait or landscape.

8.41.10 Scroll To

This action allows the user to scroll to the element whose "text" attribute contains the Input text.

8.41.11 Scroll to Exact

This action allows the user to scroll to exact location of the element as per the Input text.

8.41.12 Get Orientation

This action allows the user to get the orientation of screen.

8.41.13 Switch Context

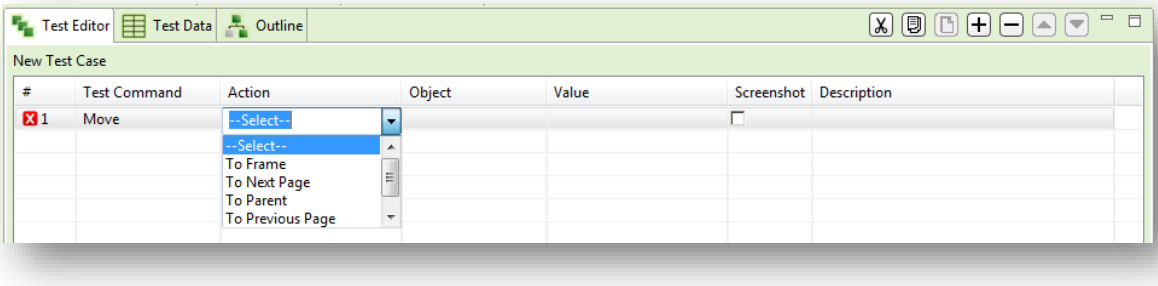
This action is basically used in a Hybrid Application. It allows the user to switch the context from Native to Webview and vice versa. Currently its scope is limited to Simulator.

8.41.14 Get Contexts

This action is basically used in a Hybrid Application. It allows user to get all available contexts of application and stores it in specified Data table. Currently its scope is limited to Simulator.

8.42 Move

Move test command allows users to move to a specific page/frame/window.



8.42.1 To Next Page

This action allows users to move to the next page after the current page.

8.42.2 To Previous Page

This action allows users to move to the previous page.

8.42.3 To Window

This action allows users to move the focus to any open window on a web page.

8.42.4 To Frame

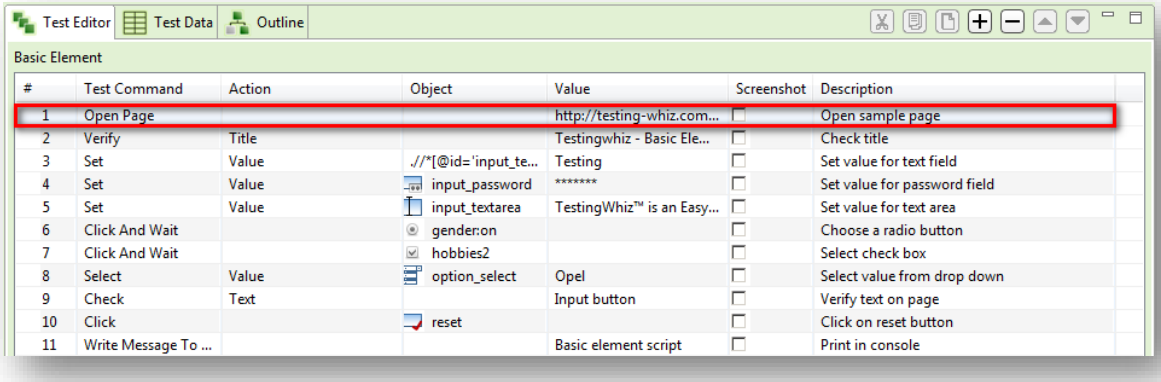
This action allows users to move to different frames of the framework-design based web page.

8.42.5 To Parent

This action allows users to move to parent window/web page from any opened window/web page.

8.43 Open Page

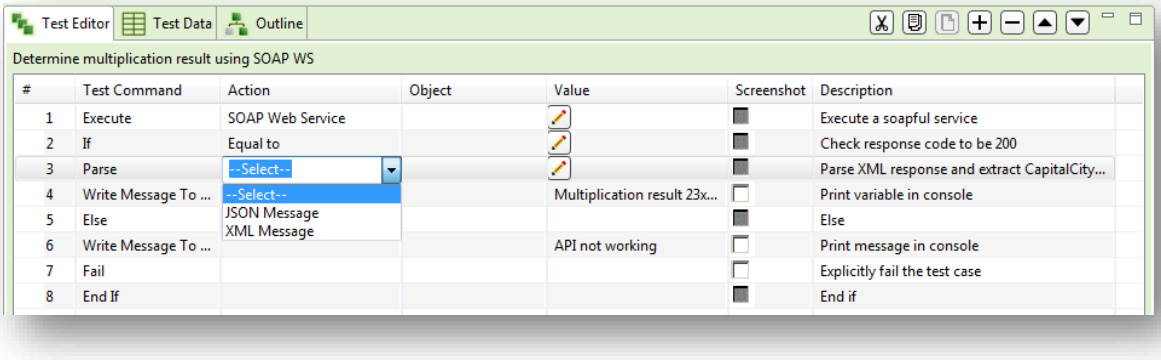
Open Page test command allows users to open a particular web page in the browser.



[**Note:** *This Test command does not contain any Action*.]

8.44 Parse

Parse test command allows users to parse meaningful information from a JSON message or an XML message which can be captured as a response of an API or from a file.



8.44.1 JSON Message

This command allows users to parse any JSON string or the JSON string which is returned as a result in REST Web Service test command.

For example:

```
"store": {
      "book": [
        {
          "category": "reference",
```

```
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95
    },
    {
        "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "price": 12.99
    },
    {
        "category": "fiction",
        "author": "Herman Melville",
        "title": "Moby Dick",
        "isbn": "0-553-21311-3",
        "price": 8.99
    },
}
```


"store.book[*].author" expression will fetch all the authors of all books.

"book[2]" expression will fetch the third book from the list.

"book[(@.length-1)]" expression will fetch the last book.

8.44.2 XML Message

This test command allows users to extract some values from an XML message or a file. XML messages can be the result of SOAP Web Services responses or users can also upload an XML file. Users can get more help on how to Parse an XML with an xpath, mouse-hover on the Help icon available.

For example:

```
<soap:Envelopexmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" [^]
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" [^]
xmlns:xsd="http://www.w3.org/2001/XMLSchema"> [^]
   <soap:Body>
      <GetCityForecastByZIPResponse xmlns="http://ws.cdyne.com/WeatherWS/">
[^]
```

```
<GetCityForecastByZIPResult>
   <Success>true</Success>
   <ResponseText>City Found</ResponseText>
   <State name="a">FL</State>
   <City>Mid Florida</City>
   <WeatherStationCity>Orlando</WeatherStationCity>
   <ForecastResult>
      <Forecast>
         <Date>2014-08-18T00:00:00</Date>
         <WeatherID>2</WeatherID>
         <Desciption>Partly Cloudy</Desciption>
         <Temperatures>
            <MorningLow/>
            <DaytimeHigh>95</DaytimeHigh>
         </Temperatures>
         <ProbabilityOfPrecipiation>
            <Nighttime/>
            <Daytime>30</Daytime>
         </ProbabilityOfPrecipiation>
      </Forecast>
   </ForecastResult>
</GetCityForecastByZIPResult>
</GetCityForecastByZIPResponse>
</soap:Body>
</soap:Envelope>
```

XPath :

Get State value : //GetCityForecastByZIPResult/State/text()

Get State node : //GetCityForecastByZIPResult/State

Get list of forecast node ://GetCityForecastByZIPResult/ForecastResult/Forecast

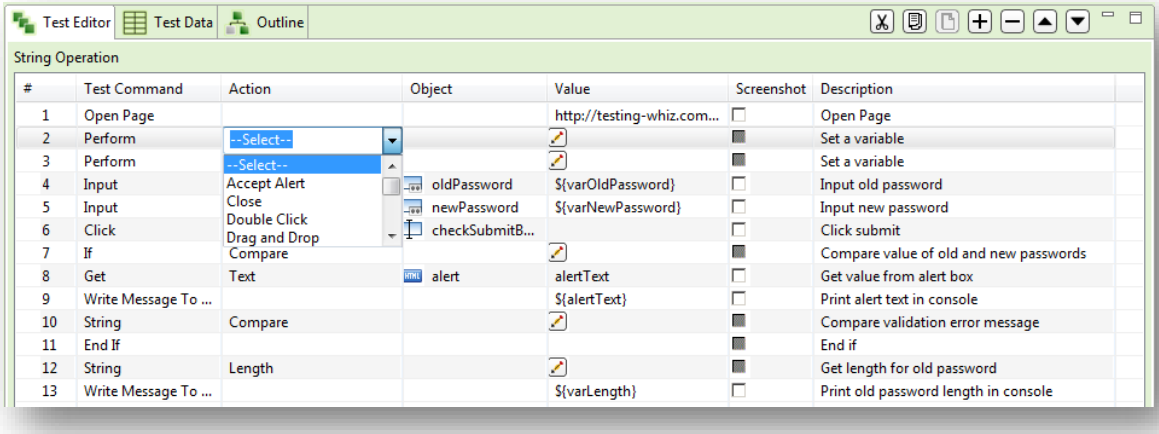
Count No. of forecast node in message :
count(//GetCityForecastByZIPResult/ForecastResult/Forecast)

Get list of nodes where DayHighTime is 95 : //Temperatures[DaytimeHigh=95]

Get name attribute value of State node : //State/@name

8.45 Perform

Perform test command allows users to perform various actions as follows.



8.45.1 Right Click

This action allows users to perform right click on a particular object.

8.45.2 Mouse Over

This action allows users to perform Mouse Over action on a particular object.

8.45.3 Scroll Up

This action allows users to perform scrolling up on a particular page.

8.45.4 Scroll Down

This action allows users to perform scrolling down on a particular page.

8.45.5 Close

This action allows users to close a particular page.

8.45.6 Set:variable

This action allows users to set a temporary variable to an object.

8.45.7 Accept Alert

This action allows users to accept the alerts/messages of the alert boxes.

[Note: *This operation will not work with Android and iOS mobile browsers.*]

8.45.8 Reject Alert

This action allows users to reject the alerts/messages of the alert boxes.

[Note: *This operation will not work with Android and iOS mobile browsers.*]

8.45.9 Set:globalvariable

This action allows users to set a permanent variable to an object. It is recommended to have a separate Test case for defining all the global variables.

8.45.10 Double Click

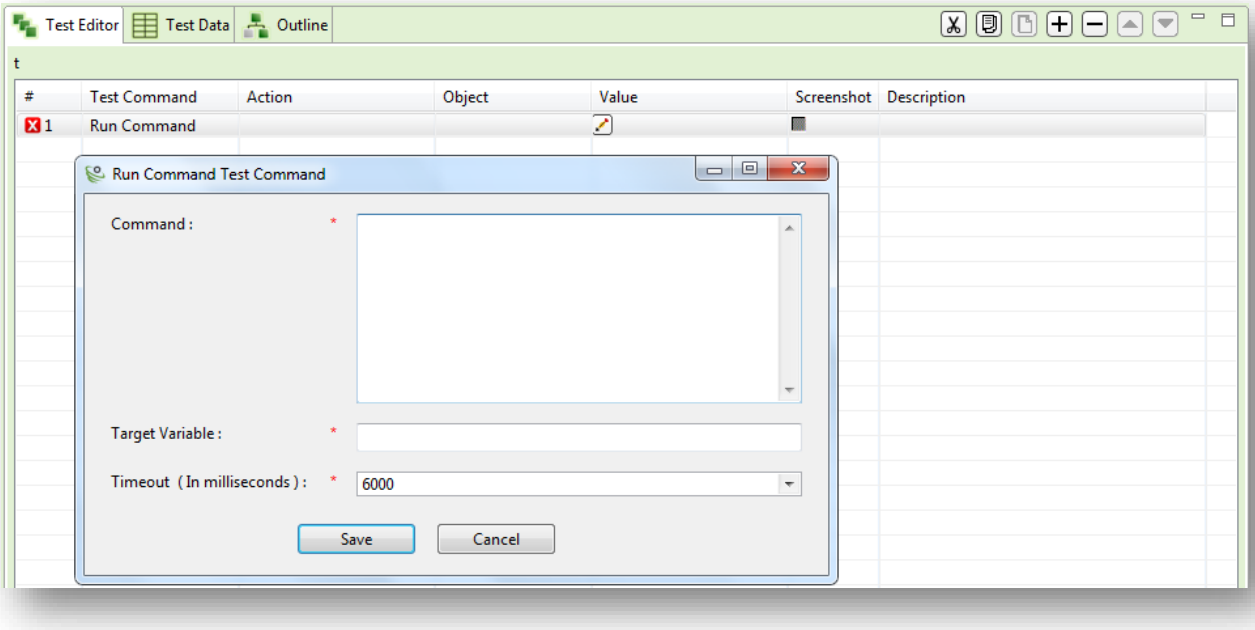
This action allows users to perform double click function on any button.

8.45.11 Drag and Drop

This action allows users to Drag and Drop facility on the web page.

8.46 Run Command

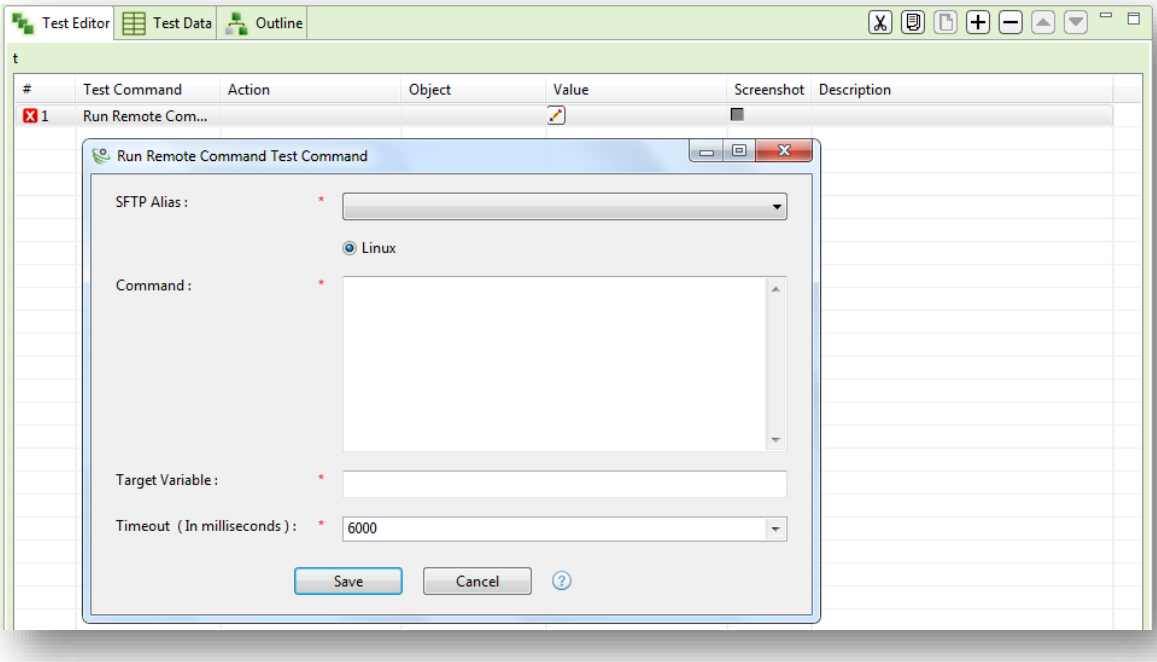
The Run Command test command allows users to execute the MS – DOS Commands.



[**Note:** *This Test command does not contain any Action*.]

8.47 Run Remote Command

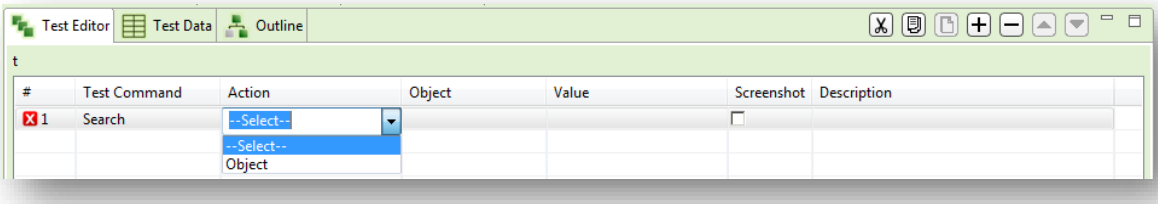
This test command allows users to execute a Linux based commands to FTP Server.



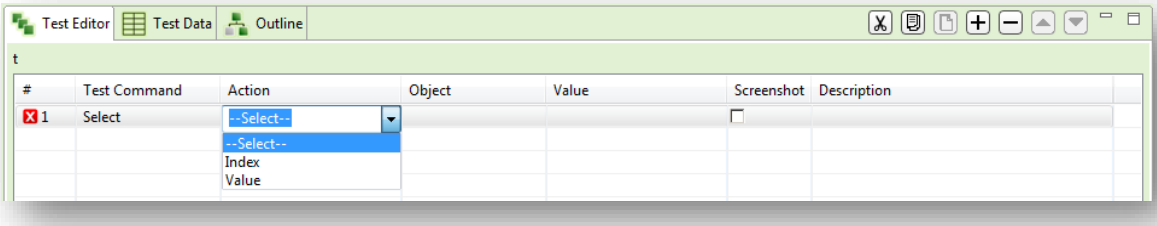
8.48 Search

8.48.1 Object

The Search > Object test command allows users to Search the Object value in a particular direction on a webpage and store it into Object repository.



8.49 Select



8.49.1 Value

This test command allows users to select a value of any object and variable. This can prevent unauthorized access of applications.

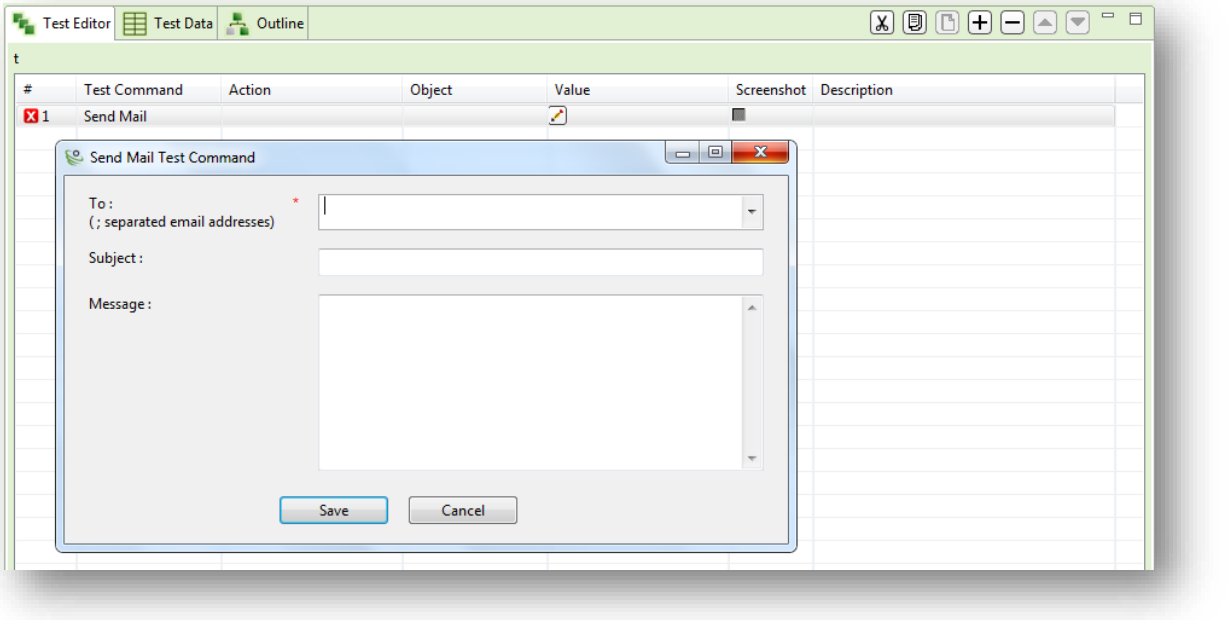
8.49.2 Index

This test command allows users to select an index of any object and variable. This can prevent unauthorized access of applications.

8.50 Send Mail

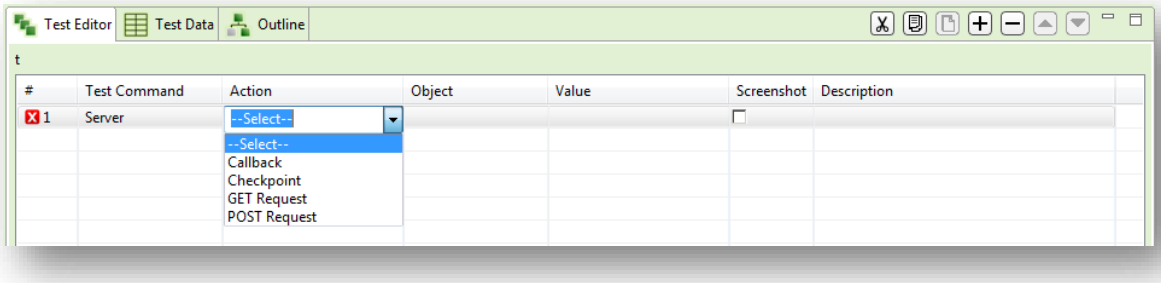
Send Mail test command allows users to send mails to desired email addresses easily during script execution. To execute this test command, users should have the required Mail configurations done.

[**Note:** *User needs to click on*  *icon and enter* ***To*** *(Recipient's ID),* ***Subject*** *and* ***Message*** *as shown below.*]



[**Note:** *This Test command does not contain any Action.*]

8.51 Server



8.51.1 GET Request

This test command allows users to request in is used by app server to signal to the test script that the HTTP request has been received.

8.51.2 POST Request

This test command allows users to request out is used by app server to signal to the test script that the HTTP response has been dispatched.

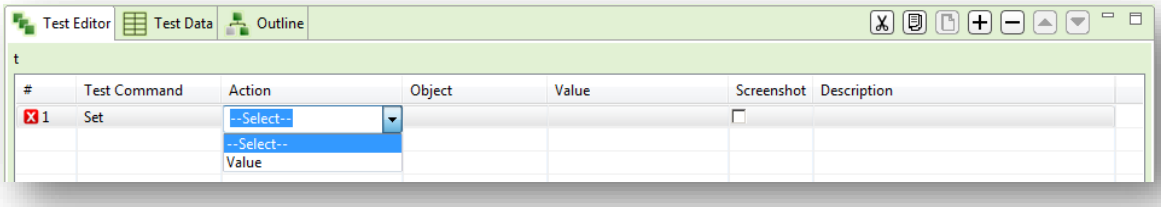
8.51.3 Checkpoint

This test command allows users to Checkpoint is used to check whether the execution logic has passed via a line of code.

8.51.4 Callback

This test command allows users to Callback is used to call a java method in the web application from the test script.

8.52 Set

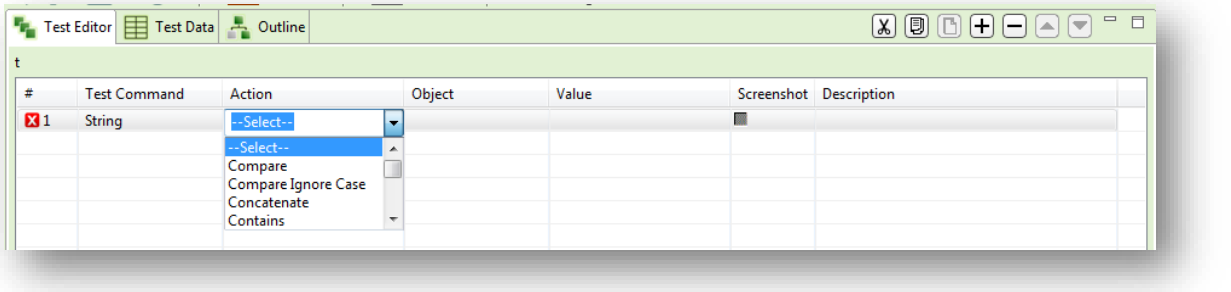


8.52.1 Value

This test command allows users to set a value of any object and variable. This can prevent unauthorized access of applications.

8.53 String

The String Test command allows users to perform various manipulations like Comparison, Finding the Length, etc. while working with the Strings.



8.53.1 Extract Substring

The Extract Substring action allows users to extract a range of characters as a Sub String from the given String.

[**Note:** *The extraction of substring would depend on the Begin and End Indexes specified by a user. If a user does not mention the End Index, then the length of the Sting would be considered as End Index.*]

8.53.2 To Lower

This action will convert the Uppercase letter to the corresponding Lowercase Letter.

8.53.3 To Upper

This action will convert the Lowercase letter to the corresponding Uppercase Letter.

8.53.4 Trim

The Trim action allows users to remove the leading and trailing blanks in the String.

8.53.5 Length

The Length action allows users to determine the length of the String.

8.53.6 Compare

The Compare action allows users to perform the Comparison between two stings i.e. verify whether two strings are equal or not. The Compare action will take the Case Sensitivity of the Strings into consideration.

8.53.7 Compare Ignore Case

The Compare Ignore Case action will work in the similar manner as Compare action, but with a little enhancement. This action will ignore the case sensitivity of the Strings at the time of comparison.

8.53.8 Concatenate

The Concatenate action will merge 2 Strings that is, it would append String2 at the end of String1.

8.53.9 IsBlankOrNull

The isBlankOrNull Test command enables a user to verify whether the value of a Variable is Null and not.

8.53.10 ToNumber

The ToNumber action will convert the numeric characters into the numbers with relevant data type.

8.53.11 Contains

The Contains action allows users to determine whether a string contains a given sub string.

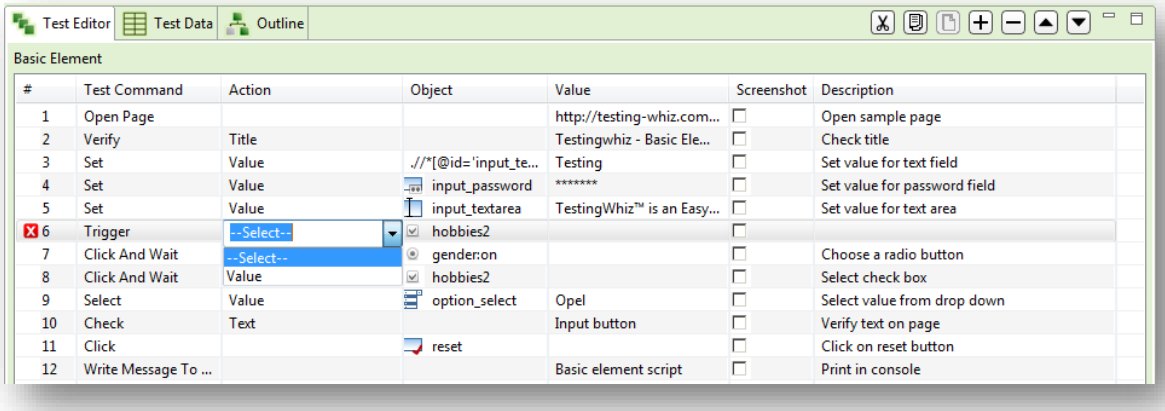
8.53.12 Split

This action allows users to split the string into multiple parts by making use of a delimiter.

8.53.13 Remove

This action allows users to remove a part by specifying it in another string.

8.54 Trigger

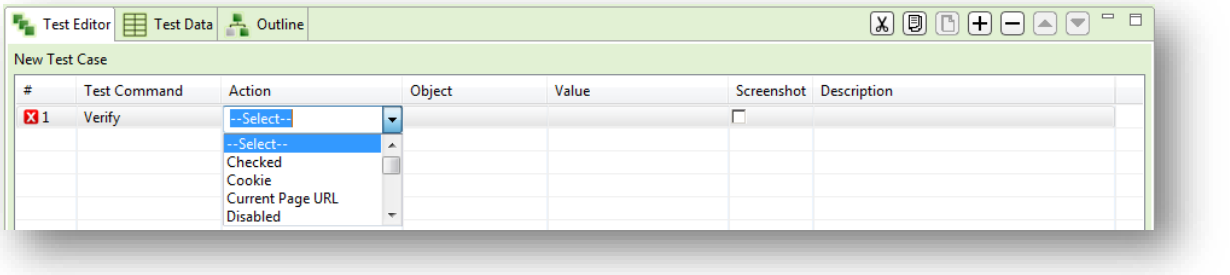


8.54.1 Value

This test command allows users to check value is used by application code to tell test script to assign a value to an object during runtime.

8.55 Verify

Verify test command allows users to verify any action. Verify test command will work same as Check test command. The only difference is that Verify test command will not stop the execution from the point where it fails.



8.55.1 Text

This action allows users whether the specified text is present on a page or not. The action will be performed for all the text on the page. The check will be performed by matching the case of the text value specified. Text with special symbols will not be considered. For e.g "hello" will be considered different from HELLO. Check test command will not stop the execution from the point where it fails.

8.55.2 Title

This action allows users to verify whether the title of a page has the specified value or not.

8.55.3 Checked

This action allows users to verify whether the checkbox is checked or selected.

8.55.4 Unchecked

This action allows users to verify whether the checkbox is unchecked or de-selected.

8.55.5 Visible

This action allows users to verify whether a specific object is visible on the page or not.

8.55.6 Invisible

This action allows users to verify whether a specific object is invisible/hidden on the page or not.

8.55.7 Enabled

This action allows users to verify whether the object (links, buttons etc.) is enabled on the page.

8.55.8 Disabled

This action allows users to verify whether the object (links, buttons etc.) is disabled on the page.

8.55.9 Selected:index

This action allows users to verify whether the option of the specified index is selected in the dropdown list.

8.55.10 Selected:value

This action allows users to verify whether the option of the specified value is selected in the dropdown list.

8.55.11 Text:value

This action allows users to verify whether an object has the specified value or not. This test command can also be utilized by taking value from the text box. For e.g. when the values in the textbox are automatically populated from a database, a user can check/verify these values by taking id or object of the textbox.

8.55.12 Exists

This action allows users to verify whether the object exists on the page or not. Verify test command will not stop the execution from the point where it fails.

8.55.13 URL Reachable

This action allows users to verify if a supplied URL in value column is a valid URL or not.

8.55.14 Image

This action allows users to compare two images with URL to URL, File to File and URL to file comparison. This command will run as per the behavior of Verify functionality, which includes following scenarios:

A. The Verify command will fail if tolerance power given is less than actual difference in images.

B. The Verify command will pass if tolerance power given is greater than actual difference in images.

8.55.15 Test Ignore Case

This action allows users to verify whether the text is present on the page irrespective of the case of the text. The check will be performed by ignoring the case of the text value specified. Text with special symbols will not be ignored.

For e.g. "hello" will be considered same as HELLO. The check will be performed on all the contents that are present in the form of the text like labels, links etc. Verify test command will not stop the execution.

8.55.16 Cookie

This action allows users to check whether the page contains a specified cookie or not. The result of the cookie's presence or absence will be reflected in the log that is generated for the Report of the Test Case.

8.55.17 Single Occurrence

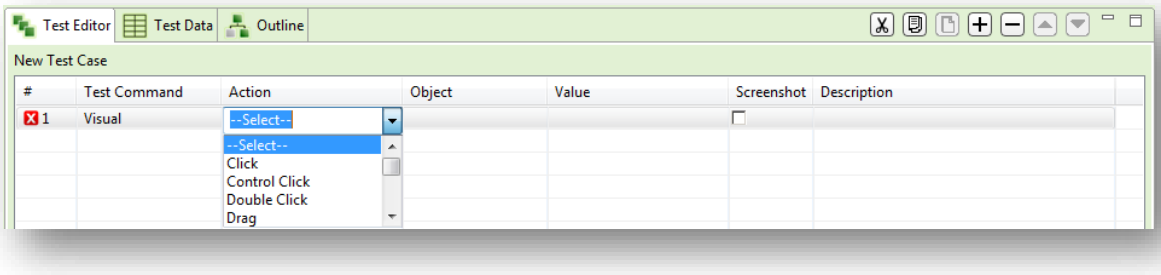
This action allows users to verify whether the value occurs only one time on the page or not. The Single Occurrence action will occur only on page contents. It will not include page title, header etc. verify test command will not stop the execution.

8.55.18 Current Page URL

This action allows users to verify the current page URL on the screen.

8.56 Visual

Visual command allows users to automate desktop popups and widgets. Its family of commands that use image matching and recognition.



8.56.1 Click

This action allows user to perform click on a particular object.

[**Note:** *This command will be performed on the objects of the recently opened page.*]

8.56.2 Input

This action allows user to input a particular value in a textbox.

[**Note:** *This command will be performed on the objects of the recently opened page.*]

8.56.3 Double Click

This action allows user to perform double click function on any button.

8.56.4 Right Click

This action allows user to perform right click on an object.

8.56.5 Middle Click

This action allows user to perform middle click in the center of the area after matching it with the stored image.

8.56.6 Drag

This action allows user to identify an area by image matching and drag it.

8.56.7 Drop

This action allows user to identify an area by image matching and drop into it.

8.56.8 Shift Click

This action allows user to click in the center of the area after matching it with the stored image while simultaneously pressing Shift key.

8.56.9 Control Click

This action allows user to click in the center of the area after matching it with the stored image while simultaneously pressing Control key.

8.56.10 Hover

This action allows user to move the mouse focus on the specified object.

8.56.11 Scroll

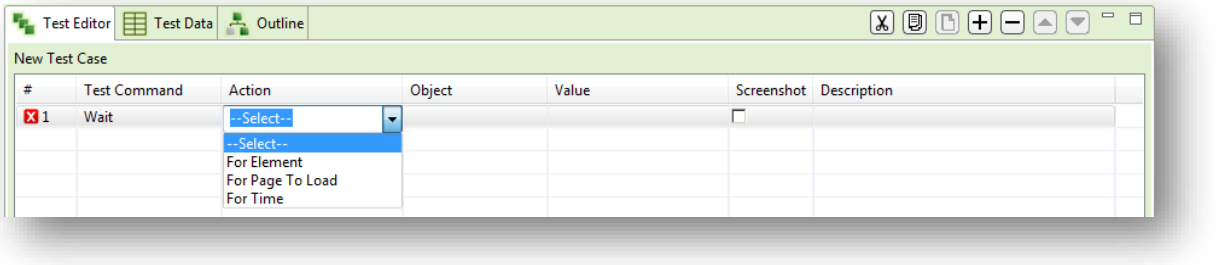
This action allows user to scroll down or up depending on value (positive or negative).

8.56.12 Read Text

This action allows user to identify an area by image matching and read the text inside using Optical character recognition(OCR), storing the result in a global variable.

8.57 Wait

Wait test command allows an element/object to wait till it is rendered on a page or for a specific time before the execution of the next action.



8.57.1 For Element

This action allows an object/element to wait for a specific time before the next action occurs.

The user can also select the Auto record feature of Wait for Element shown in the figure below. This will add the Wait for Element Test Command automatically in the Test Script for Test Command "Click" and "Select".

8.57.2 For Time

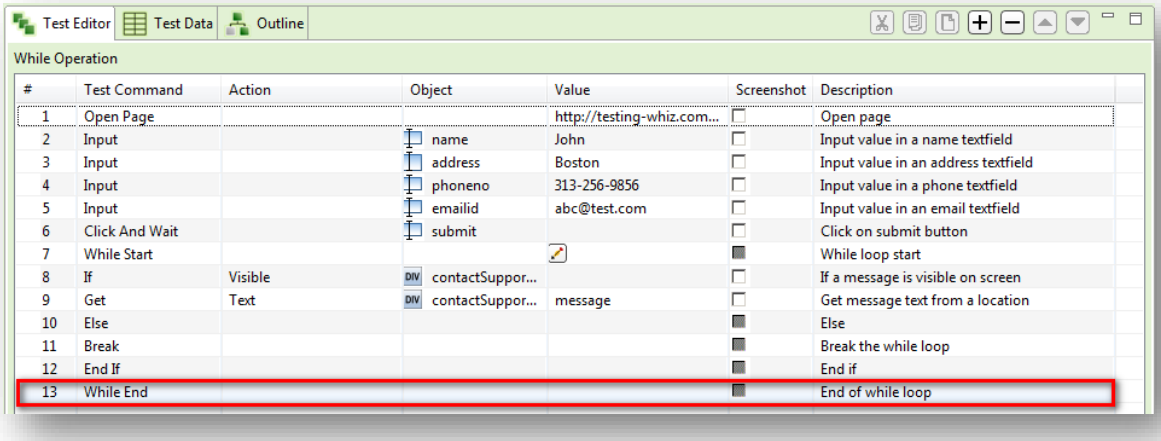
This action allows users to wait for a specific time before the next action occurs.

8.57.3 For Page to Load

This action allows users to wait till the page is loaded fully. TestingWhiz would wait for the server response for ReadyState Page and will move ahead on to next step when it receives from the browser.

8.58 While End

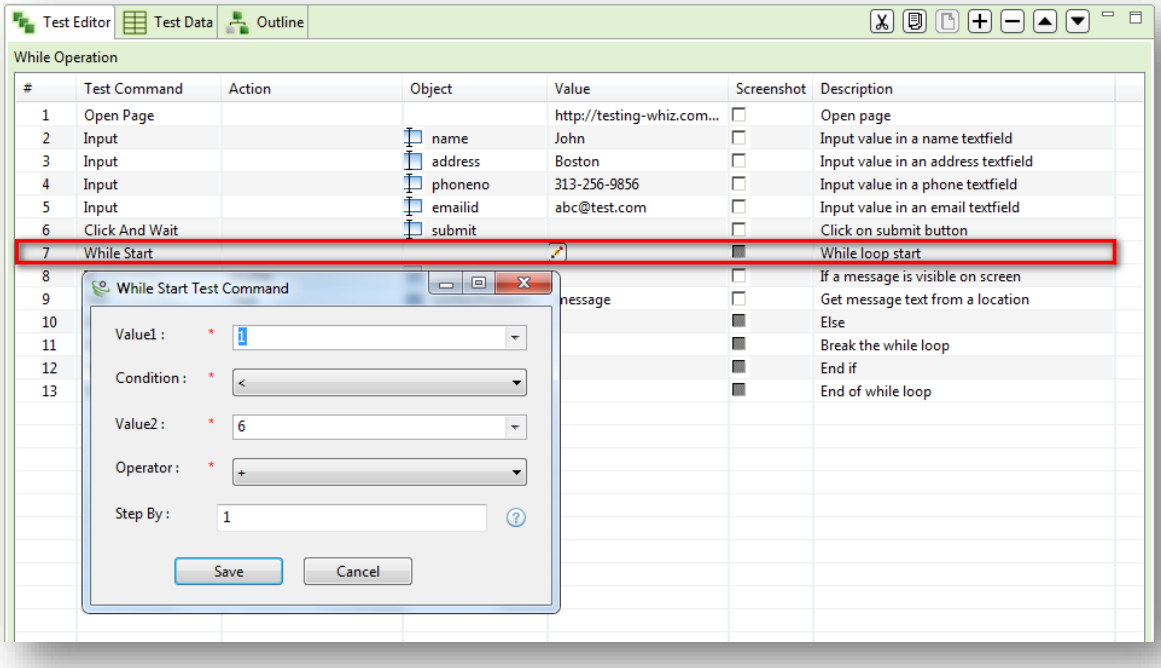
While End test command allows users to end a While loop of Test Steps for a defined condition.



[**Note:** *This Test command does not contain any Action.*]

8.59 While Start

While Start test command allows users to start a defined While Loop of Test Steps. To execute this loop, user can mention the condition for the loop.



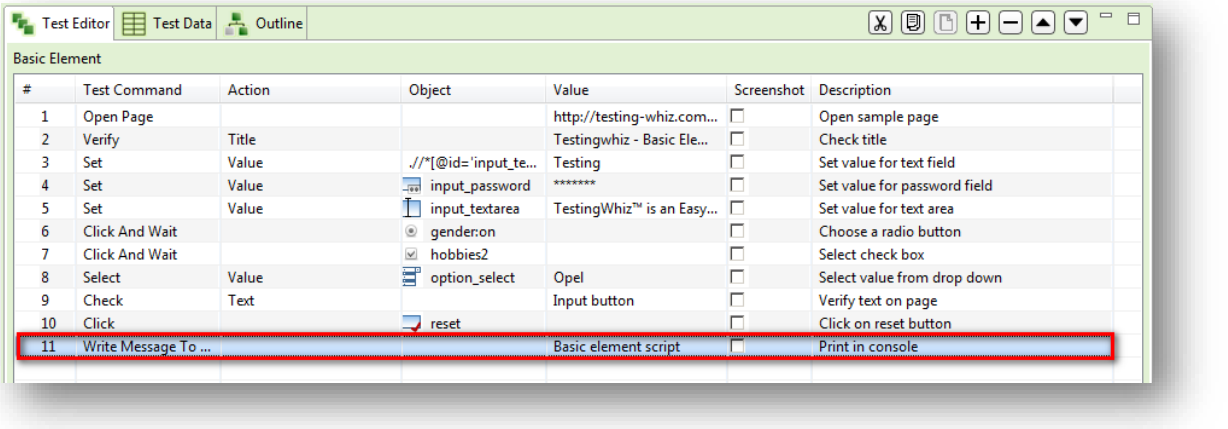
[**Note:** *User needs to input data in* ***Value1, Condition, Value 2,*** *select* ***Operator*** *information and* ***Step By*** *position.*]

[**Note:** *A script with While Start test command should also contain While End test command to end the loop.*]

[**Note:** *This Test command does not contain any Action*.]

8.60 Write Message To Report

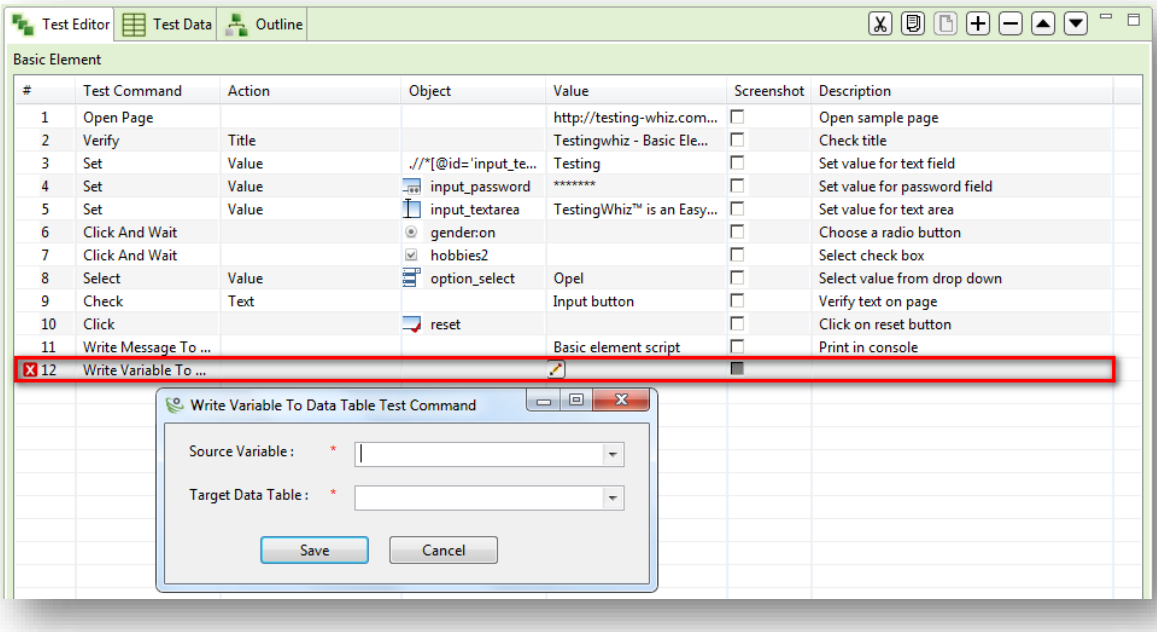
Write Message to Report test command allows users to write a particular message on the console and report window.



[**Note:** *This Test command does not contain any Action*.]

8.61 Write Variable To Data Table

This command allows a user to Write Source and Target Variable to Data Table



[**Note:** *This Test command does not contain any Action.*]