

# AdventNet **QEngine 5.2**

## Quick Start Tutorial

<http://www.adventnet.com>

[qengine-support@adventnet.com](mailto:qengine-support@adventnet.com)

AdventNet, Inc.  
5645 Gibraltar Drive  
Pleasanton, CA 94588 USA

Phone: +1-925-924-9500

Fax: +1-925-924-9600

[info@adventnet.com](mailto:info@adventnet.com)

## Table of Contents

<b>TUTORIALS</b> .....	<b>3</b>
Adding/Executing Web Performance Tests .....	4
Getting Started with Web Performance Tutorial .....	5
Illustrating Creating Business Cases .....	11
Illustrating Data Parameterization.....	12
Illustrating Load Test Configurations .....	13
Illustrating Load Test Execution.....	14
Adding/Executing Web Functional Test cases .....	15
Example 1 - Testing Form Elements .....	16
Example 2 – Adding Element Checkpoint .....	18
Example 3 - Adding Text Checkpoint .....	21
Example 4 - Adding Table Checkpoint .....	23
Example 5 - Adding Data Configuration and For Construct .....	26
Adding/Executing API Test Cases.....	29
Adding and Executing API Test Case.....	29
Test Documentation.....	30
Test Creation .....	31
Test Execution .....	35
Adding/Executing Functional Test Cases .....	36
Adding and Executing Functional Test Case.....	36
Test Documentation.....	37
Test Creation .....	38
Test Execution .....	42
Adding/Executing API Sequence Test Cases.....	44
Adding Sequence Test Cases .....	44
Test Documentation.....	45
Test Creation .....	46
Test Execution .....	48
Adding/Executing Performance Test Cases .....	49
Adding Performance Test Cases.....	49
Performance Test Plan .....	50
API Performance Test Creation.....	52
API Performance Test Execution.....	54
Resource Usage Performance Test Creation.....	57

Resource Usage Performance Test Execution .....	58
Code Instrumentation Test Creation .....	60
Code Instrumentation Test Execution.....	62
Adding/Executing Web Services Test Cases .....	63
Adding/Executing Web Services Test Cases .....	63
Test Documentation.....	64
Test Creation .....	65
Test Execution .....	67

## Tutorials

### Tutorials: Introduction

The tutorials explain the step by step procedure of creating and executing the Web Performance, Web Functional, Java API, Java Application Functionality, Java API Sequence, and Java Application Performance test cases. The topics covered in this book are as follows:

- Adding/Executing Web Functional Test Case
- Adding/Executing Web Performance Test Case
- Adding and Executing API Test Case
- Adding and Executing Functional Test Case
- Adding and Executing Sequence Test Case
- Adding and Executing Performance Test Cases
- Adding and Executing Web Services Test Cases

## **Adding/Executing Web Performance Tests**

### **Adding Web Performance Tests**

The topics covered in this book are as follows:

- Getting Started with Web Performance Tutorial
- Illustrating Creating Business case
- Illustrating Data Parameterization
- Illustrating Load Test Configurations
- Illustrating Load Test Execution

## Getting Started with Web Performance Tutorial

In this section you will learn how to develop and execute Web Performance test cases using Web Performance test Studio using the sample application Employee Scheduler. The basic steps involved in Web Performance testing are as follows:

1. Recording the list of actions that a user performs on the web-site.
2. Parameterize the user data, URL and its parameters.
3. Configure the user profiles (group of business cases), workloads and load tests.
4. Execute the load test with the specified load conditions.

### Step 1: Invoking Web Performance Studio

Invoke the Web Performance Studio from **QEngine launcher --> Web Performance Studio icon**, or through the **StartWebPerfStudio.bat/sh** file in *<QEngine Home>/bin* directory.

### Step 2: Creating New Suite

To create a new suite, click **File -> New Suite** menu. In the 'Suite Creation' dialog, specify the Suite Name, Application Name and version and click **Ok** to create the suite. The new suite would be created and added to the tree on the left side pane.

### Step 3: Configure the browser settings

The Web Browser must have the right configuration in order to enable recording and work with Web Performance tester. Following need to be set before moving on with creating/recording a business case for Web performance testing.

#### Configuring Browser Proxy Settings

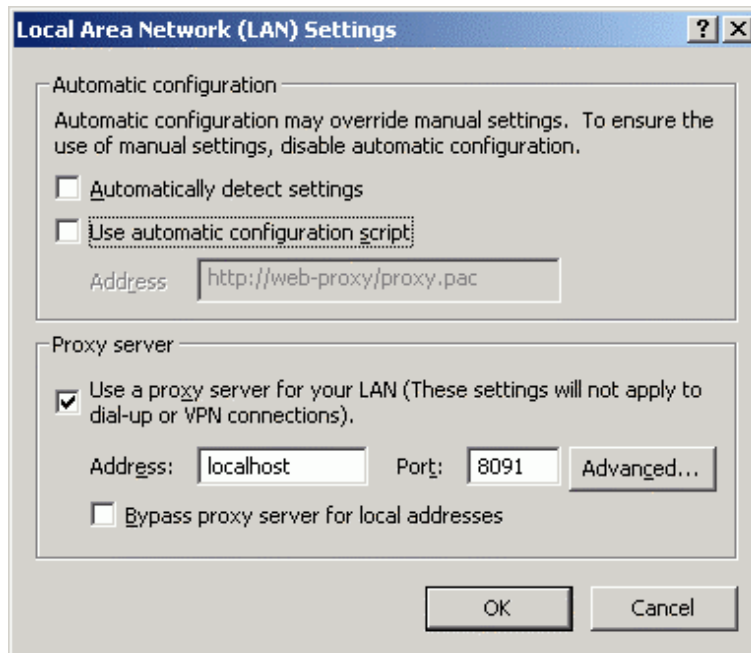
Automated Proxy Setup UI is provided to automatically configure the Web browser proxy settings for Windows platform. This option is by default enabled, if your default browser is IE/Mozilla/FireFox. Choose Control->Start Record menu item from the menu bar to view the Automated Proxy Setup UI. To know the details, please refer to the context sensitive help . For Linux platform and for other browsers not set as default browser in Windows, you need to manually set the browser proxy settings as explained below. The following table explains the same.

Browsers	Platform	
	In Windows	In Linux/Solaris
Internet Explorer (if set as default browser)	Automated Proxy Setup is enabled	NA
Mozilla (if set as default browser)	Automated Proxy Setup is enabled	Manual Proxy Setup is required
FireFox (if set as default browser)	Automated Proxy Setup is enabled	Manual Proxy Setup is required
Opera	Manual Proxy Setup is required	Manual Proxy Setup is required
Konqueror, etc.	Manual Proxy Setup is required	Manual Proxy Setup is required

## Manual Proxy Settings

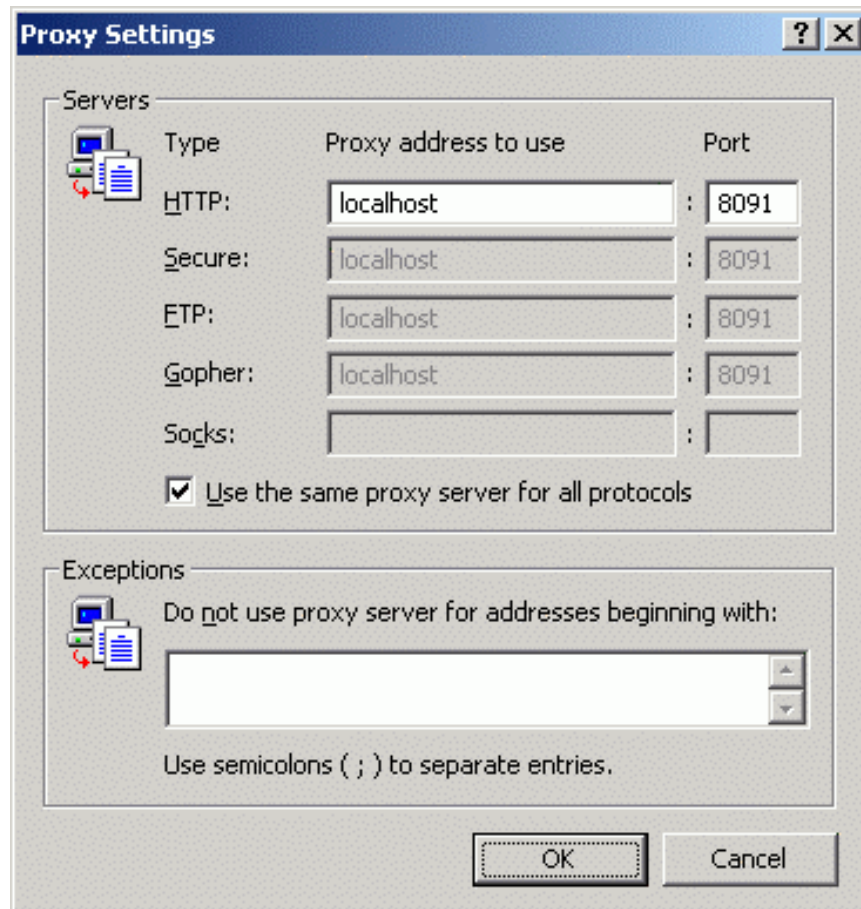
### Internet Explorer

1. Open Internet Explorer
2. Click on **Tools**
3. Click on **Internet Options**
4. Click on **Connections**
5. Click on the **LAN Settings** button
6. Make sure that the checkbox next to **Use a proxy server for your LAN (These settings will not apply for dial-up or VPN Connections)** is checked as shown in the screenshot below:



7. Enter the Address as "localhost" and Port as "8091".

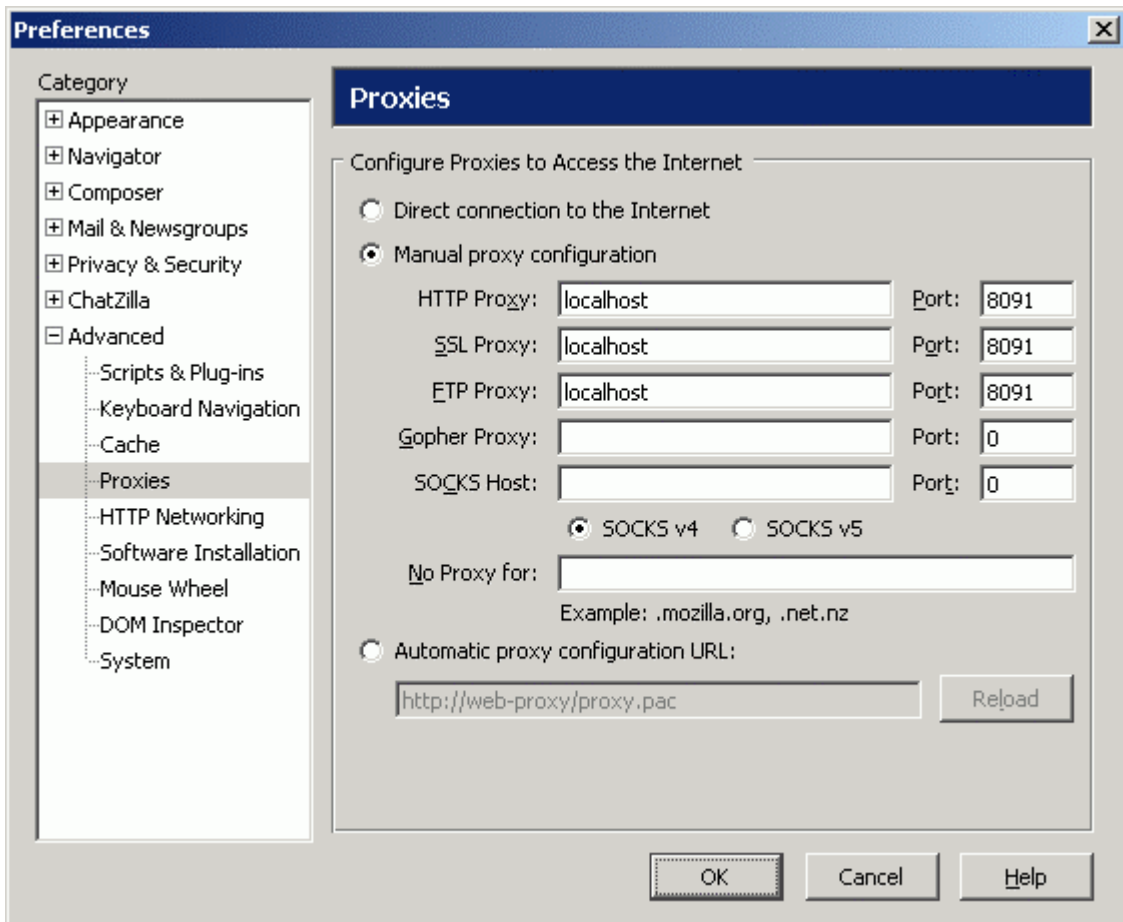
- Click the button labeled, "Advanced". For HTTP, enter the proxy host name as "localhost" and port as "8091". Check the check box "Use the same proxy for all protocols" as shown in the screenshot below:



- Click on "OK" button in the subsequent screens. This will set the proxy settings for IE.

**Mozilla**

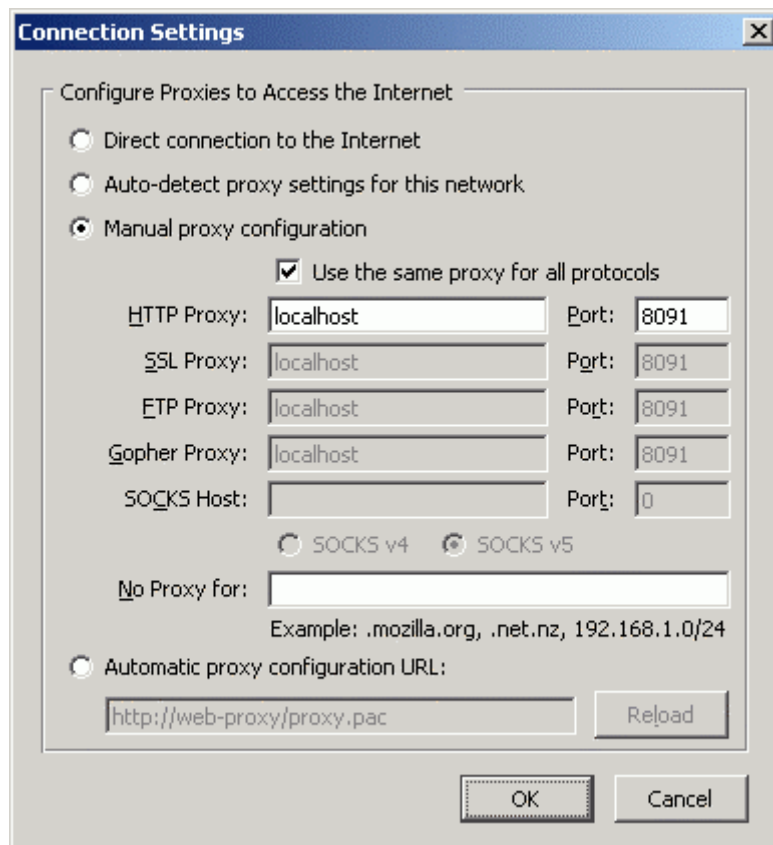
1. Open Mozilla
2. Click on **Edit->Preferences**
3. In the list on the left, click on the arrow next to **Advanced**
4. Click on **Proxies**
5. Click on the radio button next to **Manual Proxy Configuration**
6. For HTTP, SSL, and FTP, enter the proxy host name as "localhost" and port as "8091" as shown in the screenshot below:



7. Click on "OK" button. This will set the proxy settings for Mozilla.

**FireFox**

1. Open FireFox
2. Click on **Tools**
3. Click on **Options**
4. Click on **Connections Settings**
5. Click on the radio button next to **Manual Proxy Configuration**
6. For HTTP, enter the proxy host name as "localhost" and port as "8091". Check the check box "Use the same proxy for all protocols" as shown in the screenshot below:



7. Click on "OK" button in the subsequent screens. This will set the proxy settings for FireFox.

**Step 4 : Configure Proxy Settings**

When you are connected to the Internet through your Local Area Network, all your HTTP requests are routed through the proxy server. Proxy settings UI serves the purpose of holding the proxy server information.

Web Performance testing tool should know the Proxy Server Details so as to send and receive HTTP requests/response.

To configure Proxy Settings, click **Settings -> Proxy Settings**. Select your type of connection to the internet. Users who connect to the web through Dial-up modem, should select the "Direct Connection to the Internet" option. User who connect to the web through their LAN should select the "Manual Proxy Configuration" option.

**Step 5: Create a Business case**

A business case is the set of user actions performed on the web site which are recorded as URLs. To create a business case,

1. Select the suite under which you want to create the businesscase.
2. Click **File -> New Businesscase** menu. Specify the name of the business case in the save dialog and click on the **Save** button. The new business case will be created for the selected suite.

After creating a bussiness case, you need to record the user actions on the web site. To start recording, select the business case and click **Record -> Start Record** menu or use the tool bar button.

**Step 6: Parameterizing the user specific data**

Parameterization lets you use a different login name and password for each virtual user (dynamic substitution of values). There's no need for each User Scenario to contain a separate script that performs the task of logging in. Similarly, you can parameterize the cookies and other headers passed in the request header such as Scheme, Proxy-Connection, etc. You can also parameterize the parameters passed with the URL string.

**Step 7: Load Test Configuration**

A load test definition involves defining the user profiles (group of business cases), defining the workloads and associating the user profiles and workloads to the load test.

User profile defines the % of users executing a business. A profile consists of one or more business cases with a certain percent of the users executing the business cases. Workload defines the number of users along with the rate or interval at which the users execute the business case.

**Step 8: Start Play**

To run the performance test or playback, select a businesscase, click on **Play -> Start Play** menu item or use the toolbar button. The Play Status UI should show up. The chosen businesscase will be selected in the Testcase combo box. To start play, click **Start Play** button.

**Web Application under test (Sample Suite - WebPerformanceDemo)**

To illustrate how one can test the performance of a web site, a sample test suite named WebPerformanceDemo for Employee Scheduler application has been already developed and bundled with the product and placed in **<QEngine Home>/projects/WebPerformanceDemo** folder. You can open it in Web Performance Test Studio using **File --> Open Suite** menu option. Execute the load tests one by one and get a feel of them.

Employee Scheduler is an open source application available freely on the web. This application was developed to reduce the effort of employers in scheduling an appropriate work time for the employees.

In this application, the employer or the supervisor can login and create user logins for the employees. The employees can then connect to the application, and specify their work schedules. Based on the work schedules specified by the employee, the supervisor can allocate appropriate work schedules. This application mainly caters to work scheduling for part time employees of a company.

The sample suite **WebPerformanceDemo** has some default load tests. These load tests illustrate some of the important features of Web Performance Test Studio. Follow the below given links for the related documents:

- Illustrating Creating Business case
- Illustrating Data Parameterization
- Illustrating Load Test Configurations
- Illustrating Load Test Execution

## Illustrating Creating Business Cases

For the Employee Scheduler application, three business cases have been considered. They are:

1. Employee logging into the application, creating his work schedule and logging out.
2. Employee logging into the application, changing password and logging out.
3. Employee logging with a wrong password, and then logging-in with the correct password and logging out.

### Start Recording

1. Configure Browser Settings and Proxy Settings.
2. Invoke Web Performance Test Studio and click **Start record** button in the toolbar.
3. Launch the browser and key in the URL : <http://demo.qachoice.com>.

### Recording businesscase "create\_schedule"

1. Login into the application with the username and password as emp/emp. The schedule chart would be shown.
2. Click the edit button to edit the schedule.
3. Save the schedule and logout of the application

### Recording businesscase "update\_user\_details"

1. Login into the application with the username and password as emp/emp.
2. Click on My Information on the left side panel. User information form would be shown.
3. Modify the password and click the update button.
4. Logout of the application.

### Recording businesscase "validate\_user\_login"

1. Try logging into the system with incorrect username and password. An error message would be shown.
2. Login into the application with correct username and password.
3. Logout of the application

The above steps will create the business cases for the Employee Scheduler application.

## Illustrating Data Parameterization

Parameterization is the process of substituting values for dynamic parameters from a CSV file or from the Database.

For example, when testing a web application that contains a login page, "parameterization" lets you use a different login name and password for each virtual user (dynamic substitution of values). There's no need for each User Scenario to contain a separate script that performs the task of logging in. Similarly, you can parameterize the cookies and other headers passed in the request header such as Scheme, Proxy-Connection, etc. You can also parameterize the parameters passed with the URL string.

In this example suite, the user name and password are dynamically substituted from the csv file. For this purpose, a **data.csv** file is created and placed under `<QEngine_Home>/projects/WebPerformanceDemo/usersrc` folder. The csv file contains 100 usernames and password starting from emp1 to emp100. 100 logins should be populated in the application's database. This is done using the following steps:

1. Select a businesscase and click the **Record -> View or Parameterize URL** menu item. The recorded URLs will be loaded in a tree.
2. Select the URL used for logging into the application and expand the node.
3. Choose the parameters under the expanded URL node. The parameters for the URL should be shown on the right side screen.
4. Select the parameter with the name as 'userpass', set the 'Type' as 'dataset'.
5. In the parameters table, the CSV Data Configuration UI will be shown by default.
6. Enter a datasource name (say emp\_pass1), browse and select the csv file and specify the column number as '0' .
7. Similarly, select the parameter 'username' and configure the data source for the same.

The above steps will use the same script with 100 different usernames and password to login-in and simulate the load of 100 virtual users.

## Illustrating Load Test Configurations

To configure the load test, open the **Load Test Configurator UI** by clicking the **Play -> Load Test Configurator** menu item or the toolbar button. By default, the 'TestCase Configurator UI' will be shown. User Profiles, Workloads and TestCases can be configured using this UI.

### Profile Configuration

1. Choose the 'Profile Manager' tab. The Profile Manager UI will be shown on the right side panel.
2. Click the **New** Button and enter a profile name (say DemoProfile).
3. Select all the businesscases.
4. Configure the following load for the businesscases.
  1. create\_schedule - 60%
  2. update\_user\_details - 20%
  3. validate\_user\_login - 20%
5. Configure the host as `http://demo.qachoice.com/` and the corresponding User Agent.
6. Save all the businesscases and create the profile by clicking the **Update** button.

### WorkLoad Configuration

1. Choose the 'WorkLoad Configurator' tab. The WorkLoad Configurator UI will be shown on the right side panel.
2. Enter the WorkLoad Name (say DemoWorkLoad).
3. Select the type of test as 'normal' and enter the total number of users for the test, total test duration and the sample period. Click the **Update** button.

### TestCase Configuration


1. Click the 'TestCase Configurator' tab.
2. The 'TestCase Configuration UI' will be shown on the right side panel.
3. Click the **New** button. The available profiles and workloads will be displayed in the corresponding combo box.
4. Enter the TestCase name (say DemoCase).
5. Select the DemoProfile and DemoWorkLoad for Profile and WorkLoad.
6. Specify the Reporting Sample Period and click the **Update** button to save the testcase.

## Illustrating Load Test Execution

You have the following choices to run the load tests:

- Executing Load Test from Studio
- Executing Load Test from Command Line

### Executing Load Test from Studio

1. Select the business case (create\_schedule or update\_user\_details or validate\_user\_login) from the Suites Tree.
2. Choose the toolbar icon  or choose **Play->Start Play** menu item. This displays the Execution Status UI as shown below.
3. From the TestCase field, select the load test to be executed and click the **Start Test** button.
4. This executes the selected load test. You can view the execution status such as Username, URL, Status and Response Code.

### Executing Load Test from Command Line

To execute the load test from command line, you need to sequence the load tests before execution. Refer "Web Performance Test Sequencing" in the help documentation for details on how to sequence the load tests and then execute them.

## Adding/Executing Web Functional Test cases

This tutorial uses the sample Payroll application developed by AdventNet to illustrate the various features of Web Functional Testing. You can find the Web pages of this sample application bundled in the product at <QEngine Home>/examples/payrollsystem folder.

The sample Payroll system application is made up of 5 Web pages:

1. Home page
2. Add Employee details form
3. Modify Employee details form
4. View Employee details form
5. Payroll reports page.

To view this application, open **index.html** present in <QEngine Home>/examples/payrollsystem folder. Menu options are available at the top of the page. Click on each menu item to view the pages mentioned above.

The next step would be to learn how to use the **Web Functional Test Studio** for developing test cases to test these pages. Following Web Functional tests are covered in this tutorial:

- Testing Form Elements
- Adding Checkpoints
- Element Checkpoint Configuration
- Text Checkpoint Configuration
- Table Checkpoint Configuration
- Inserting Data Configuration and for construct

## Example 1 - Testing Form Elements

In this section, you will learn how to record the form elements in a Web page and playback the actions.


### Step 1: Start Web Functional Test Studio

Start the Web Functional Test Studio using the **StartWebFuncTestStudio.bat/sh** file in *<QEngine Home>/bin* directory.


### Step 2: Selecting the Suite Settings (Browser Options)

- From the Suite tree, select the Suite for which you want to add the test case.
- Choose the menu item **Browser Preferences** from the **Control** menu.
- **Select Browser** enables you to choose the appropriate browser for record and playback of browser events. Choose IE, Mozilla or Firefox. For Mozilla or Firefox, click the **Install Addin** button to install the add-ins and then choose the appropriate checkbox in the **Use** column to select the browser version.


### Step 3: Add New Script

Click **File->New Script** menu item from the menu bar or click the tool bar option . In the dialog, specify the name of the new script file as **test-form-elements** and click the **Save** button. The script file is by default saved in *<Qengine\_Home>/projects/<Suite\_Name>/webscripts/<script\_name>* directory. You can also save the script file in any other directory under *<QEngine\_Home>/projects/<Suite\_Name>/webscripts/* directory (by creating a new directory).

### Step 4: Launch Browser

Click **Control->Launch Browser** menu item from the menu bar or click the tool bar option . This will launch the browser (IE, Mozilla or Firefox) based on the option chosen in **Control->Browser Preferences** screen. Enter the URL of the page **add.html** from the sample Payroll application in *<QEngine Home>/examples/payrollsystem* directory. It is a form via which Employee details are added. This form contains most of the form elements that are commonly used.


### Step 5: Start Recording Web Page Events

To record the events, click **Control->Start Recording** menu item from the menu bar or click the tool bar option . If you have not performed Step 4, then this option will launch a new browser window. Enter the form details such as name, department, gender, LOGIN id, password, interest in communication, mode for communication, areas of interest and languages known. Finally, close the browser. The list of actions performed in the Web page should be recorded in the newly created script file. The script file can be viewed in the script editor.

### Step 6: Stop Recording Events

To stop recording events, click **Control->Stop Recording** menu item from the menu bar or click the tool bar option .

### Step 7: Play Recorded Events


To play the recorded events, click on the script name **test-form-elements** in the left tree and then either choose the toolbar icon  or choose **Play->Execute Test** menu item. You will see cursor highlight in the script editor window which indicates the line that is being executed. Also, you will see the actions performed in the launched browser. You can notice the status of the test case execution at the bottom of the tool. **Output** and **Error** messages are also displayed.

On completion of test case execution, you can click on the menu **View --> Summary Report** to view the test results.


## Example 2 – Adding Element Checkpoint

In the previous example, you would have already created the suite and set the browser preferences. Hence, we will move on to the next step of adding a new script to add an element checkpoint.


### Step 1: Add New Script

Click **File->New Script** menu item from the menu bar or click the tool bar option . In the dialog, specify the name of the new script file as **checkpoint-scr** and click the **Save** button. The script file is by default saved in `<QEngine_Home>/projects/<Suite_Name>/webscripts/<script_name>` directory. You can also save the script file in any other directory under `<QEngine_Home>/projects/<Suite_Name>/webscripts/` directory (by creating a new directory).

### Step 2: Launch Browser

Click **Control->Launch Browser** menu item from the menu bar or click the tool bar option . This will launch the browser (IE, Mozilla or Firefox) based on the option chosen in **Control->Browser Preferences** screen. Enter the URL of the page **index.html** from the sample Payroll application in `<QEngine Home>/examples/payrollsystem` directory.

### Step 3: Start Recording Web Page Events

To record the events, click **Control->Start Recording** menu item from the menu bar or click the tool bar option . If you have not performed Step 4, then this option will launch a new browser window. From the **index.html** page, select the “Employee Details” menu from the menu bar and choose the menu item “Modify”. Modify Employee Details page gets displayed. In this page, we will add two checkpoints **name-check** checkpoint to check **if the name of the employee is "Peter Patrick"** and **lang-check** checkpoint to check **if the language French has been chosen in the web page**.

### Step 4: Inserting Testcases

From Web Functional Test Studio, choose **Control->Insert Testcase** menu item.

In the Test Case Configuration screen,

1. Click the **New** button.
2. Specify the appropriate values for Test Case ID, Description and Severity.
3. Select the radio button GUI CheckPoint. Following are the sub-options displayed:
  1. **Element** - To verify the attributes of GUI objects in a Web browser such as select, check-box, etc.
  2. **Text** - To verify whether the specified text exists in the Web browser.
  3. **Table** - To verify whether the specified column count, row count or cell value in the specified column or row exists in the Web browser.
  4. For element checkpoint configuration, we need to select the Element radio option. By default, the **Element** radio button is selected. The launched browser is displayed.
  5. In the Web browser, click on the text box that displays the value “Peter Patrick” to verify the attributes of the selected object. The attributes of the selected object are displayed in a table in the **Checkpoint Configuration** dialog screen.
  6. By default, all the properties are selected. Unselect all the properties, except “defaultChecked”.
  7. Choose **Commit to List**. This adds the Checkpoint type to the Checkpoint Configuration list. Click **OK**.
  8. To add another Checkpoint “lang-check”, choose **Control->Insert Testcase** menu item and in the Test Case Configuration screen, repeat the steps 1 to 4.
  9. In the Web browser, click on the check box “French” for **Languages Known** to verify the attributes of the selected object. The attributes of the selected object are displayed in a table in the **Checkpoint Configuration** dialog screen.
  10. By default, all the properties are selected. Unselect all the properties, except “defaultChecked” and change the value of defaultChecked to “true”.
  11. Choose **Commit to List**. This adds the Checkpoint type to the Checkpoint Configuration list. Click **OK**.

The list of actions performed in the Web page should be recorded in the newly created script file. The script file can be viewed in the script editor.

## Step 5: Stop Recording Events

To stop recording events, click **Control->Stop Recording** menu item from the menu bar or click the tool bar option .

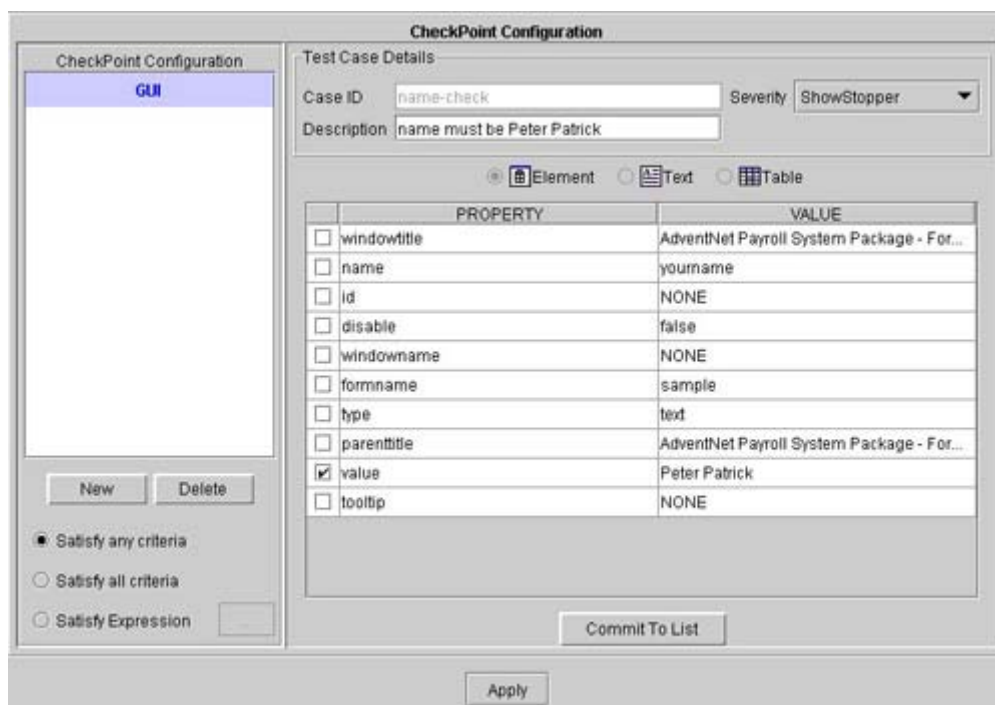
## Step 6: Play Recorded Events

To play the recorded events, click on the script name **checkpoint-scr** in the left tree and then either choose the toolbar icon or choose **Play->Execute Test** menu item. You will see cursor highlight in the script editor window which indicates the line that is being executed. Also, you will see the actions performed in the launched browser. You can notice the status of the test case execution at the bottom of the tool. **Output** and **Error** messages are also displayed.

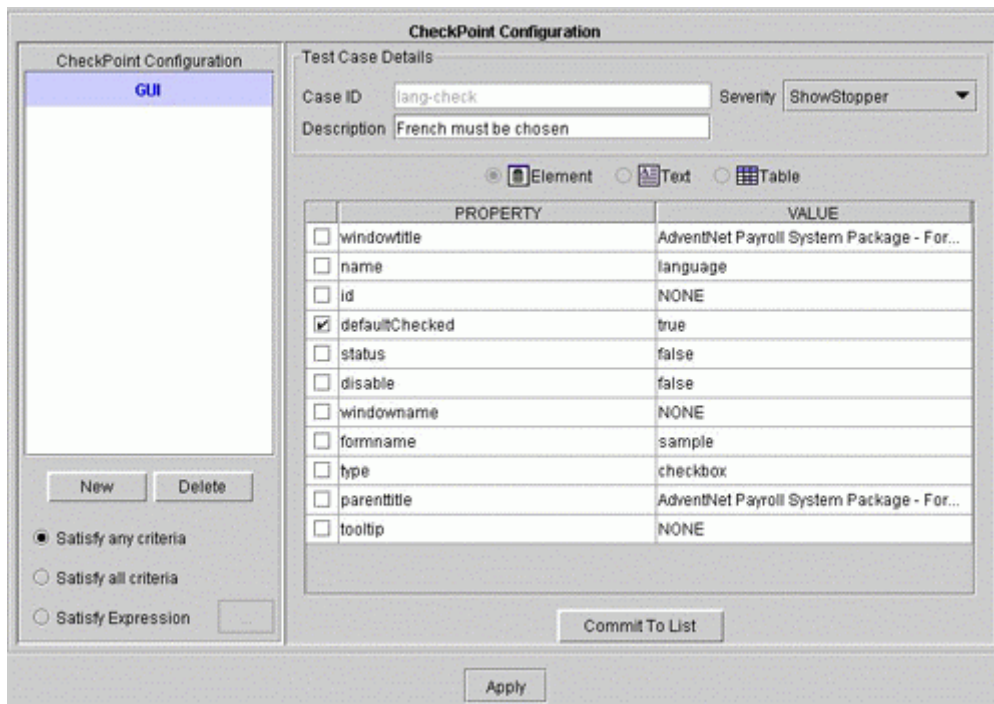
On completion of test case execution, you can click on the menu **View --> Summary Report** to view the test results.

The same has been illustrated in the script named "**test-element-checkpoints**" in the sample Payroll application. On expanding this webscript in the tree, you can see two checkpoint configurations named **lang-check** and **comm-check**.

**name-check** checkpoint has been designed to check **if the name of the employee is "Peter Patrick"**.



lang-check checkpoint has been designed to check if the language French has been chosen in the web page.




During recording, these checkpoints were inserted as explained above.

On executing the scripts, you will see that **name-check passes** but **lang-check fails** as "French" is not chosen. To view the test results, click on **View --> Summary Report**.


## Example 3 - Adding Text Checkpoint

In the previous example, you would have already created the suite and set the browser preferences. Hence, we will move on to the next step of adding a new script to add a text checkpoint.


### Step 1: Add New Script

Click **File->New Script** menu item from the menu bar or click the tool bar option . In the dialog, specify the name of the new script file as **text-scr** and click the **Save** button. The script file is by default saved in `<QEngine_Home>/projects/<Suite_Name>/webscripts/<script_name>` directory. You can also save the script file in any other directory under `<QEngine_Home>/projects/<Suite_Name>/webscripts/` directory (by creating a new directory).

### Step 2: Launch Browser

Click **Control->Launch Browser** menu item from the menu bar or click the tool bar option . This will launch the browser (IE, Mozilla or Firefox) based on the option chosen in **Control->Browser Preferences** screen. Enter the URL of the page **index.html** from the sample Payroll application in `<QEngine Home>/examples/payrollsystem` directory.

### Step 3: Start Recording Web Page Events

To record the events, click **Control->Start Recording** menu item from the menu bar or click the tool bar option . If you have not performed Step 4, then this option will launch a new browser window. From the **index.html** page, select the "Employee Details" menu from the menu bar and choose the menu item "Add". Add Employee Details page gets displayed. In this page, we will add a text checkpoint to verify the specified text given the prefix and suffix text.

### Step 4: Inserting Testcases

From Web Functional Test Studio, choose **Control->Insert Testcase** menu item.

In the Test Case Configuration screen,

1. Click the **New** button.
2. Specify the appropriate values for Test Case ID, Description and Severity.
3. Select the radio button GUI CheckPoint. Following are the sub-options displayed:
  1. **Element** - To verify the attributes of GUI objects in a Web browser such as select, check-box, etc.
  2. **Text** - To verify whether the specified text exists in the Web browser.
  3. **Table** - To verify whether the specified column count, row count or cell value in the specified column or row exists in the Web browser.
  4. For text checkpoint configuration, we need to select the Text radio option. Select the Text radio option. The launched browser is displayed.
  5. In the Web browser, click on the title text "Add Employee Details". The selected text will be displayed in the **Text Checkpoint Configuration** dialog screen.
  6. Select the Prefix text as "Add" and click the button **Set As Prefix**.
  7. Select the Search text as "Employee" and click the button **Search For**.
  8. Select the Suffix text as "Details" and click the button **Set As Suffix**.
  9. Click the **OK** button.
  10. The selected prefix, search and suffix text will be displayed in the Testcase Configuration screen.


11. Choose **Commit to List**. This adds the Checkpoint type to the Checkpoint Configuration list. Click **OK**.

The list of actions performed in the Web page should be recorded in the newly created script file. The script file can be viewed in the script editor.

### Step 5: Stop Recording Events

To stop recording events, click **Control->Stop Recording** menu item from the menu bar or click the tool bar option .

### Step 6: Play Recorded Events


To play the recorded events, click on the script name **text-scr** in the left tree and then either choose the toolbar icon  or choose **Play->Execute Test** menu item. You will see cursor highlight in the script editor window which indicates the line that is being executed. Also, you will see the actions performed in the launched browser. You can notice the status of the test case execution at the bottom of the tool. **Output** and **Error** messages are also displayed. On executing the scripts, you will see that **text-check** returns the result as Passed.

On completion of test case execution, you can click on the menu **View --> Summary Report** to view the test results.


## Example 4 - Adding Table Checkpoint

In the previous example, you would have already created the suite and set the browser preferences. Hence, we will move on to the next step of adding a new script to add a table checkpoint.


### Step 1: Add New Script

Click **File->New Script** menu item from the menu bar or click the tool bar option . In the dialog, specify the name of the new script file as **table-scr** and click the **Save** button. The script file is by default saved in `<QEngine_Home>/projects/<Suite_Name>/webscripts/<script_name>` directory. You can also save the script file in any other directory under `<QEngine_Home>/projects/<Suite_Name>/webscripts/` directory (by creating a new directory).

### Step 2: Launch Browser

Click **Control->Launch Browser** menu item from the menu bar or click the tool bar option . This will launch the browser (IE, Mozilla or Firefox) based on the option chosen in **Control->Browser Preferences** screen. Enter the URL of the page **index.html** from the sample Payroll application in `<QEngine_Home>/examples/payrollsystem` directory. From the **index.html** page, select the "Reports" menu from the menu bar and choose the menu item "Payroll". The payroll details for sales department will be shown. In this page, we will add two table checkpoints **if-peter-present** checkpoint to check **if the value at the cell 3,2 in the displayed table contains the employee name Peter Waring** and **if-mary-present** checkpoint to check **if the value at the cell 4,2 in the displayed table contains the employee name Mary**.

### Step 3: Start Recording Web Page Events

To record the events, click **Control->Start Recording** menu item from the menu bar or click the tool bar option . If you have not performed Step 4, then this option will launch a new browser window.

### Step 4: Inserting Testcases

From Web Functional Test Studio, choose **Control->Insert Testcase** menu item.

In the Test Case Configuration screen,

1. Click the **New** button.
2. Specify the appropriate values for Test Case ID, Description and Severity.
3. Select the radio button GUI CheckPoint. Following are the sub-options displayed:
  1. **Element** - To verify the attributes of GUI objects in a Web browser such as select, check-box, etc.
  2. **Text** - To verify whether the specified text exists in the Web browser.
  3. **Table** - To verify whether the specified column count, row count or cell value in the specified column or row exists in the Web browser.
4. For table checkpoint configuration, we need to select the Table radio option. The launched browser is displayed.
5. In the Web browser, click on the cell value "Peter Waring" to verify the cell value. The table attributes Count Count, Cell Value At and Row Count are displayed in a table in the **Checkpoint Configuration** dialog screen.
6. Choose **Commit to List**. This adds the Checkpoint type to the Checkpoint Configuration list. Click **OK**.
7. To add another table Checkpoint, choose **Control->Insert Testcase** menu item and in the Test Case Configuration screen, repeat the steps 1 to 4.

8. In the Web browser, click on the cell value "Betty Martha" to verify the cell value. The table attributes Count Count, Cell Value At and Row Count are displayed in a table in the **Checkpoint Configuration** dialog screen.
  1. Edit the **Cell Value At** from "Betty Martha" to "Mary".
  2. Choose **Commit to List**. This adds the Checkpoint type to the Checkpoint Configuration list. Click **OK**.

The list of actions performed in the Web page should be recorded in the newly created script file. The script file can be viewed in the script editor.

### Step 5: Stop Recording Events

To stop recording events, click **Control->Stop Recording** menu item from the menu bar or click the tool bar option .

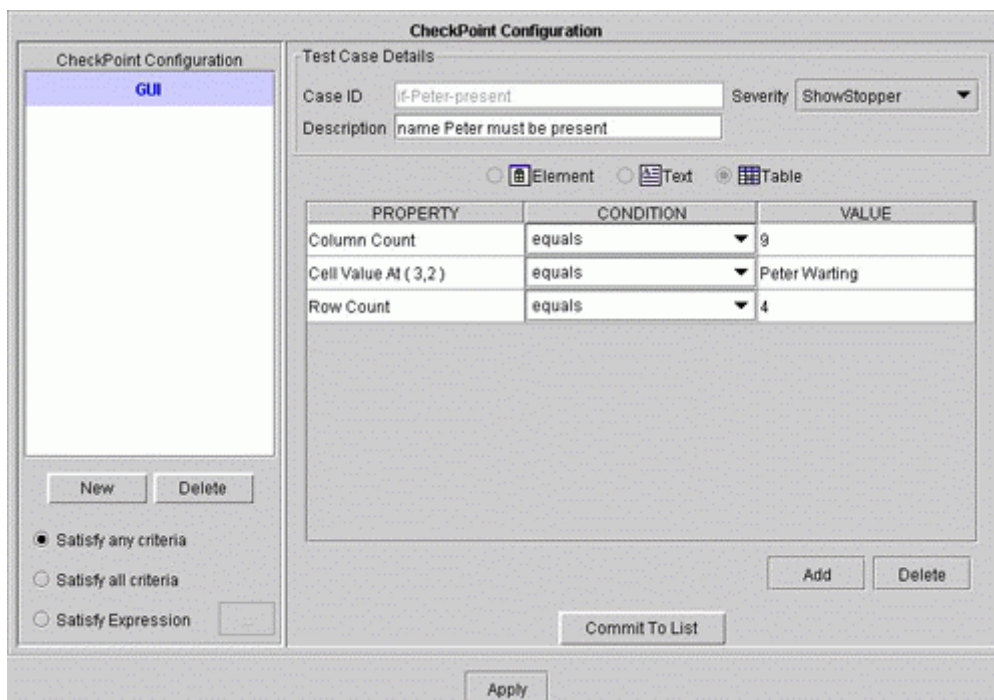
### Step 6: Play Recorded Events

To play the recorded events, click on the script name **table-scr** in the left tree and then either choose the toolbar icon or choose **Play->Execute Test** menu item. You will see cursor highlight in the script editor window which indicates the line that is being executed. Also, you will see the actions performed in the launched browser. You can notice the status of the test case execution at the bottom of the tool. **Output** and **Error** messages are also displayed.

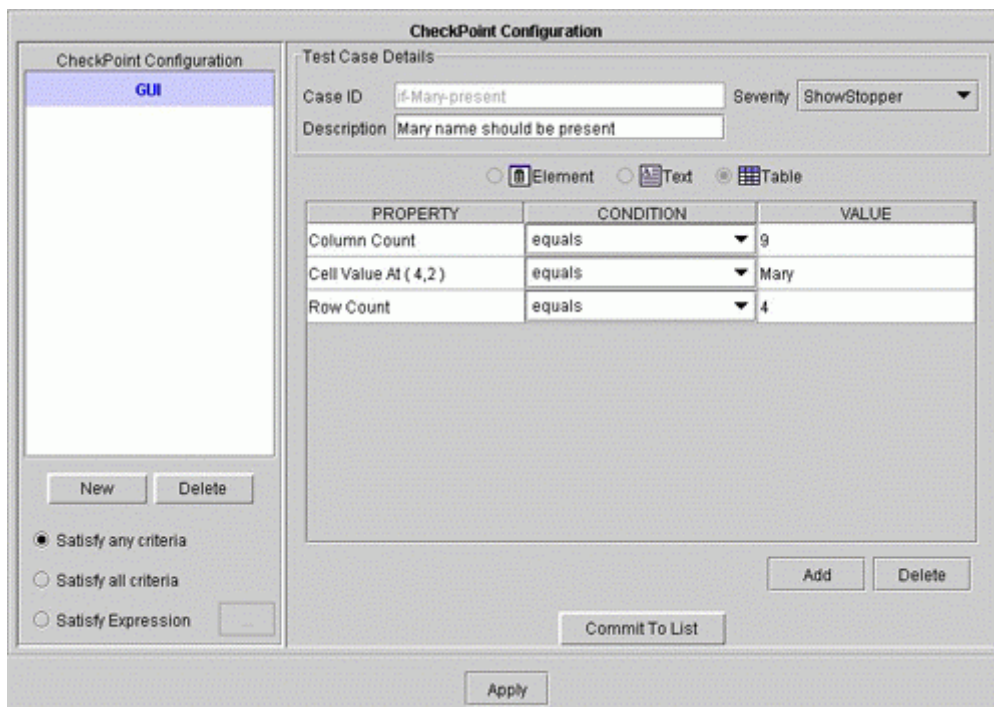
On completion of test case execution, you can click on the menu **View --> Summary Report** to view the test results.

This has been illustrated in the script named "**test-table-checkpoint**" in the sample Payroll application. On expanding this webscript in the tree, you can see two checkpoint configurations named **if-peter-present** and **if-mary-present**.

**if-peter-present** checkpoint checks **if the value at the cell 3,2 in the displayed table contains the employee name Peter Warting**.



**if-mary-present** checkpoint checks if the value at the cell 4,2 in the displayed table contains the employee name **Mary**.




During recording, these checkpoints were inserted as explained above.

On executing the scripts, you will see that **if-peter-present check passes** but **if-mary-present check fails** as "Mary" is not found in the specified cell. To view the results, click on **View --> Summary Report**.


## Example 5 - Adding Data Configuration and For Construct

In the previous example, you would have already created the suite and set the browser preferences. Hence, we will move on to the next step of adding a new script to insert data configuration and for construct.


### Step 1: Add New Script

Click **File->New Script** menu item from the menu bar or click the tool bar option . In the dialog, specify the name of the new script file as **dataconfig-scr** and click the **Save** button. The script file is by default saved in `<QEngine_Home>/projects/<Suite_Name>/webscripts/<script_name>` directory. You can also save the script file in any other directory under `<QEngine_Home>/projects/<Suite_Name>/webscripts/` directory (by creating a new directory).

### Step 2: Launch Browser

Click **Control->Launch Browser** menu item from the menu bar or click the tool bar option . This will launch the browser (IE, Mozilla or Firefox) based on the option chosen in **Control->Browser Preferences** screen. Enter the URL of the page **index.html** from the sample Payroll application in `<QEngine Home>/examples/payrollsystem` directory.

### Step 3: Start Recording Web Page Events

To record the events, click **Control->Start Recording** menu item from the menu bar or click the tool bar option . If you have not performed Step 4, then this option will launch a new browser window. From the **index.html** page, select the "Employee Details" menu from the menu bar and choose the menu item "View". The View Employee Details page will be shown. In the Employee Details page, enter the Employee Name as "Tom" and Department as "Sales". In this page, we will insert data configuration to fetch the data from the CSV file test.csv which is placed in `<QEngine Home>/examples/payrollsystem` directory.

The list of actions performed in the Web page should be recorded in the newly created script file. The script file can be viewed in the script editor.

### Step 4: Inserting Data Configuration

1. Select the text "Tom" including the quotes from the `setText()` statement.
2. Choose **Control->Data Configuration** menu item.
3. Enter the variable as **name** in the **Variable Name** field.
4. From **Choose Data Source to Import Data** combo-box, choose **CSV File** option to import data from CSV file.
5. To import data from CSV file:
  1. Enter or browse the csv file name test.csv file from `<QEngine Home>/examples/payrollsystem` directory..
  2. Enter the row index and column index value from where the value has to be fetched.
  6. Click **OK**.
  7. To fetch the department values from CSV file, select the text "Sales" including the quotes from the `setText()` statement. Repeat the steps from 2 to 6. In step 3, enter the variable as **dept** in the **Variable Name** field.

8. In the script, you will find the following lines are inserted:

```
initCSV("C:/QEngine/examples/payrollsystem/test.csv")
name=getCSVValueAt(1,1)
setText("userName",name,2)
initCSV("C:/QEngine/examples/payrollsystem/test.csv")
dept=getCSVValueAt(1,1)
setText("password",dept,2)
```

The above is the skeletal script where the web page elements are recorded. Now, remove the second initCSV statement and add the for construct to fetch multiple values from the csv file as shown below:

```
initCSV("$APPHOME\test.csv") //Here $APPHome is the environmental
//variable that will be automatically replaced with the application
//home directory by QEngine.
for i in range(0,4):
    name=getCSVValueAt(i,1)
    dept=getCSVValueAt(i,2)
    setText("yourname",name,3)
    setText("dept",dept,3)
```

Find below a walk through of the script lines after adding the above lines.

#### Launch the browser and load the Payroll system home page.

```
useLocalMapFile()

launchApplication("about:blank")
changeURL("$APPHOME/index.html",2)
```

#### Move to view.html

```
setWindow( "AdventNet Payroll System",2)
clickList("View",2)
setWindow( "AdventNet Payroll System Package - Form to View Employee Details",2)
```

#### Specify the CSV file name to be read from

```
initCSV("$APPHOME\test.csv")
```

Using the for loop, read the the lines 0 to 4 from the CSV file. Copy the values from file to local variables name and dept. Set these in the text fields of the web page.

```
for i in range(0,4):
    name=getCSVValueAt(i,1)
    dept=getCSVValueAt(i,2)
    setText("yourname",name,3)
    setText("dept",dept,3)
```

**Jython** scripting language is used in webscripts and hence you can refer its syntax manual for the syntax of various programming constructs such as if, for, while etc.

On executing the script, you will see that the values are retrieved from the CSV file one by one and displayed in the form fields. To view the results of testing , click on **View --> Summary Report**.

The sample webscript named **test-for-construct** in the suite **AdventNetPayrollSystem** is an illustration of the above steps. You can also view to the same.

Similarly, you can also fetch values from database or use variable substitution option to fetch the data at runtime.

## Adding/Executing API Test Cases

### Adding and Executing API Test Case

- Test Documentation
- Test Creation
- Test Execution

## Test Documentation

The first step to achieve test automation is Test Documentation. Let us consider a test case, which is to check the CharAt(int) method of java.lang.String. The test plan is as defined below.

### Introductions

This test plan contains the test case to test the functionality of CharAt(int) method of java.lang.String.

### Assumptions

Nil.

### Application Control Parameters

Nil

Test Case ID	Test Case Description	Test Case Procedure	Expected Behavior	Severity
api-string-char-001	Testing the CharAt(int) method of java.lang.String.	<b>API Name.</b> java.lang.String with "AdventNet"  <b>Method to be tested.</b> charAt()  <b>Argument value</b> 6  <b>Validation type</b> <i>RetVsConst</i> condition -equals value-N	Should return the 6th element of "AdventNet" that is "N".	Showstopper

## Test Creation

The Test Creation process consists of the below two steps:

- Tool Selection
- Test Case Definition

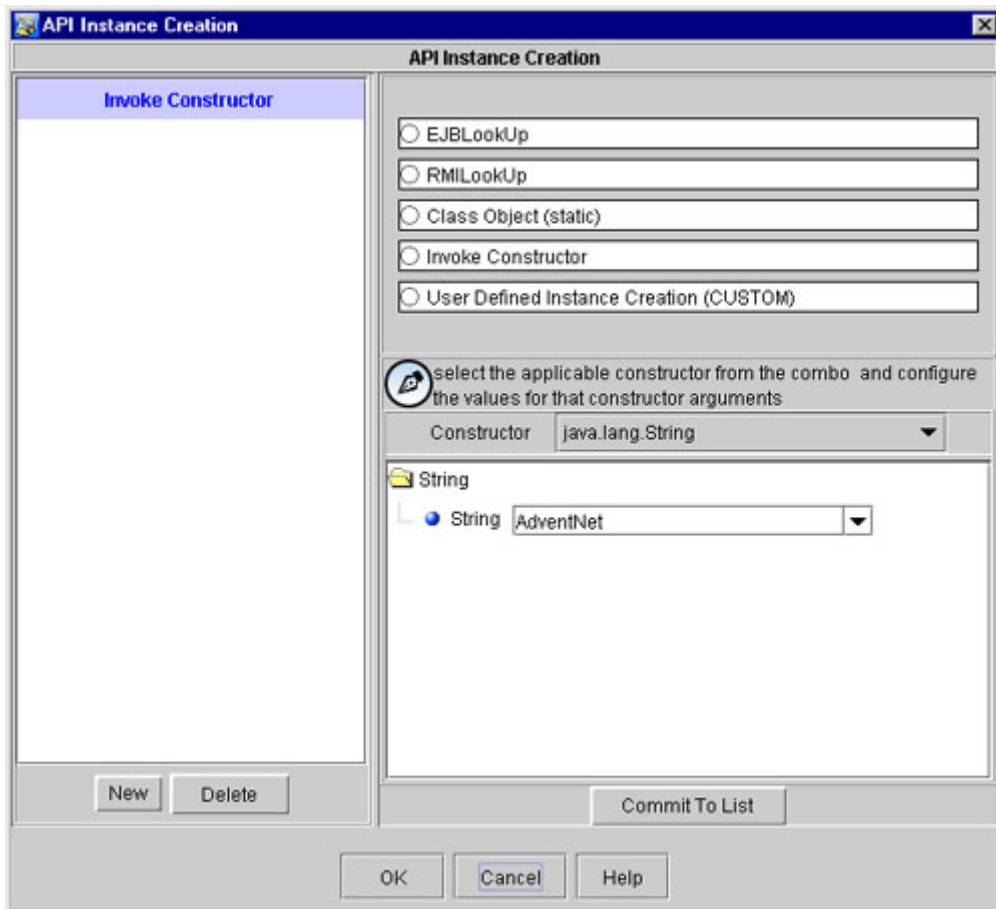
### Tool Selection

From the test plan of the test case api-string-char-001, we know that the test case's purpose is to test the function of the method CharAt(int) of java.lang.String. Hence the type of testing to be performed is Unit Testing. Hence, invoke the API Test Studio for automating this test case.

### Test Case Definition

Loading an API and configuring API Instance

1. Select any Suite on the Suites Tree.
2. Choose the menu item **New API** under the **File** menu. This displays API Details screen.
3. Enter java.lang.String in the **Name of the API** field.
4. Choose **OK**.
5. This adds the API java.lang.String onto the Suites Tree under the respective selected Suite.
6. Expand the node java.lang.String on the Suites Tree. The tree expands to display all the methods of the API.
7. Select the method CharAt(int) on the tree.
8. Choose the menu item **New Test Case** under the **File** menu. This displays a Test Case Creation screen.
9. Enter api-string-char-001 in the **Case ID** field.
10. Choose Showstopper in the **Severity** combo box.
11. Enter **Checking the behavior of CharAt** method in the **Description** field.
12. Choose the **Instance** button. This displays an API Instance Creation screen.
13. Choose Invoke Constructor option.
14. Select java.lang.String from the **Constructor** combo box.
15. Replace ---Select/Enter a value---with **AdventNet** in the **String** combo box.
16. Click the **Commit to List** button. The API Instance Creation screen after all configurations looks as shown below:



17. Click the **OK** button.
18. Replace ---Select/Enter a value---with 6 in the **int** combo box, under Method Arguments. The Test Case Creation screen after all the above configurations looks as shown below:

**New TC-API Mode**

Case Details

Case ID:  Severity:

Description:

API Details

API Name:

API Instance:  **Instance**

Method:

Method Arguments

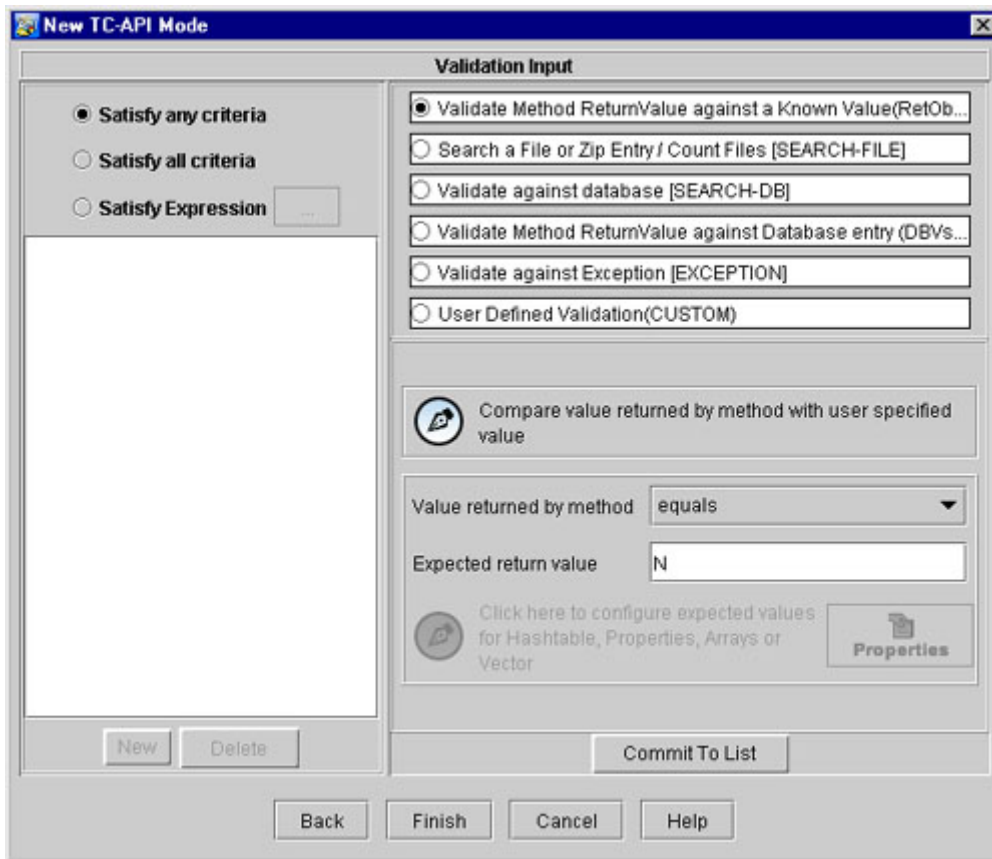
Specify values for all the arguments.

charAt

int  **Advanced Input**

----Select/Enter a value----

20. Click the **Next** button.
21. Enter the value N in the **Expected return value** field.
22. Click the **Commit to List** button. The Test Case Creation screen after all the above configurations looks as shown below:



23. Click the **Finish** button. The automated test case api-string-char-001 is added onto the Suites Tree.

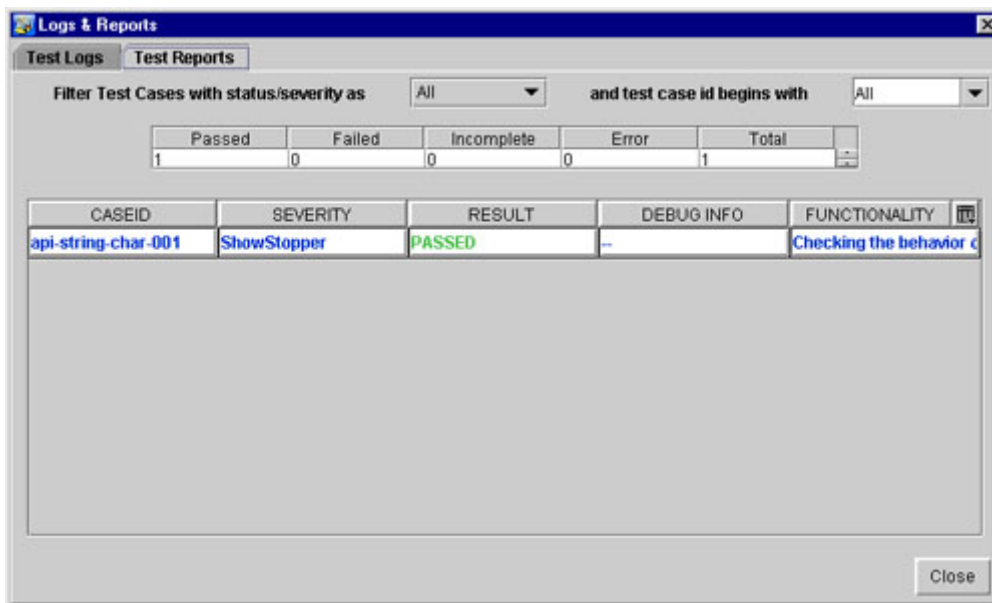
## Test Execution

### Executing Automated Test Case

1. Select the method CharAt(int) on the SuitesTree.
2. Choose the menu item **Start** under the **Test** menu.
3. Message **Test Execution Complete**, in the Progress Bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen.
2. Choose the Test Reports tab to view test results. The report will look as shown below:



### Viewing Test Logs

1. Choose the menu item Test Results under the View menu. This displays the Logs & Reports screen with the test execution logs.

## **Adding/Executing Functional Test Cases**

### **Adding and Executing Functional Test Case**

- Test Documentation
- Test Creation
- Test Execution

## Test Documentation

The first step to achieve test automation is Test Documentation. Let us consider a test case, which is to check if the LoginApplication's server starts correctly at port 2345. The test plan is as defined below.

### Introduction

This test plan contains the test case to test the functionality of Login Administrator application's server module.

### Assumptions

Nil.

### Application Control Parameters

- Run `startauthserver.bat/sh` to start application.
- Run `stopauthserver.bat/sh` to stop application.

Test Case ID	Test Case Description	Test Case Procedure	Expected Behavior	Severity
serv-conf-099	Testing to check if Login Administrator's server is started at port 2345.	<ol style="list-style-type: none"> <li>1. Start LoginApplication.</li> <li>2. Check the state of the port 2345 using CHECK-PORT validation.</li> </ol>	Server should start at port 2345.	Showstopper

## Test Creation

The Test Creation process has 3 steps:

- Tool Selection
- Suite Creation
- Test Case Definition

### Tool Selection

From the test plan of the test case serv-conf-099, we know that the test case's purpose is to test the function of LoginApplication's server. Hence the type of testing to be performed is Functional. Hence, invoke the Functional Test Studio for automating this test case.

### Suite Creation

To create a Suite for the LoginApplication follow the steps given below:

1. Choose the menu item **New Suite** under the **File** menu. This displays the Suite Creation screen as shown below :

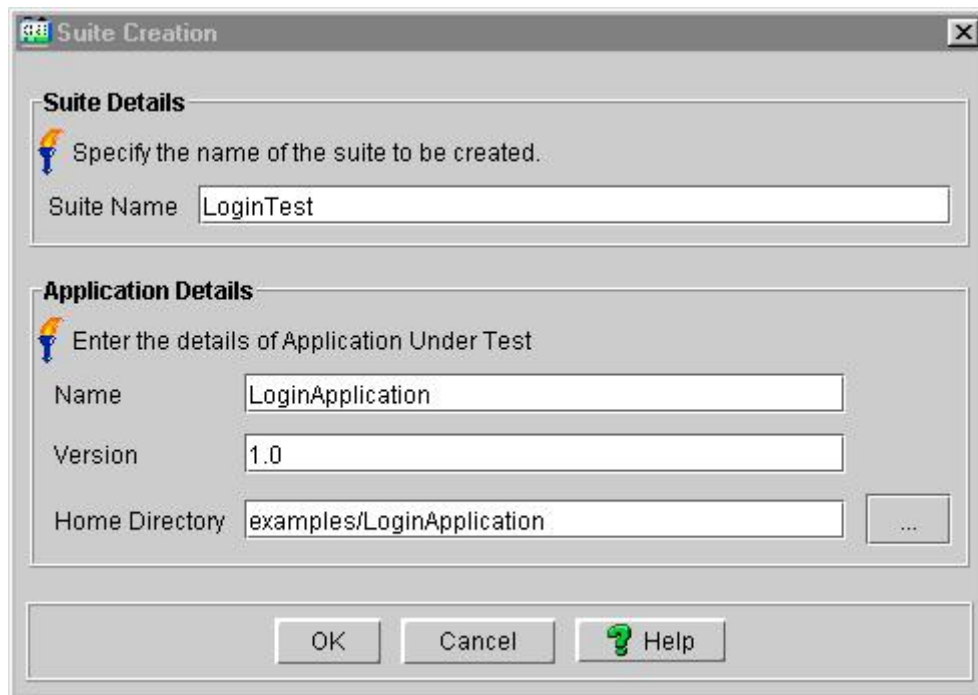
The image shows a 'Suite Creation' dialog box with the following fields:

- Suite Details:**
  - Specify the name of the suite to be created.
  - Suite Name:
- Application Details:**
  - Enter the details of Application Under Test
  - Name:
  - Version:
  - Home Directory:  ...

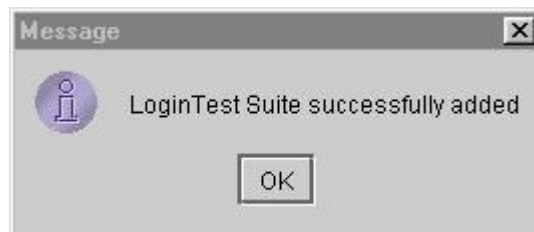
Buttons at the bottom: OK, Cancel, Help.

2. Enter the following values in the Suite Creation screen as shown in the image below table.

Field Name	Value
Suite Name	LoginTest
Name	LoginApplication
Version	1.0
Home Directory	examples/LoginApplication



3. When done, click **OK**. This displays a message as shown below:



4. Choose **OK**. The Suite LoginTest is added to the Suites Tree.

	<p><b>Note:</b> Refer "Suite Creation" in the help documentation for more details.</p>
--	--

## Test Case Definition

To define the test case,

1. Select the Suite LoginTest on the Suites Tree.
2. Choose the menu item **New Test Case** under the **File** menu. This displays the Functional Test Case Creation screen.
3. Enter serv-conf-099 in the **Test Case ID** field.
4. Enter **Testing to check if LoginApplication's server is started at port 2345** in the **Test Case Description** field.
5. Select Showstopper in the **Severity** combo box as shown below.

**Functional Test Case Creation**

**Test Case Details**

Enter testcase id and a brief description of the test case. Once the test case is created, then one must have script files to perform the test.

Case ID:

Description:

Severity:

**Application Control Parameters**

Configure startup and shutdown scripts.

**Application Control Parameters**

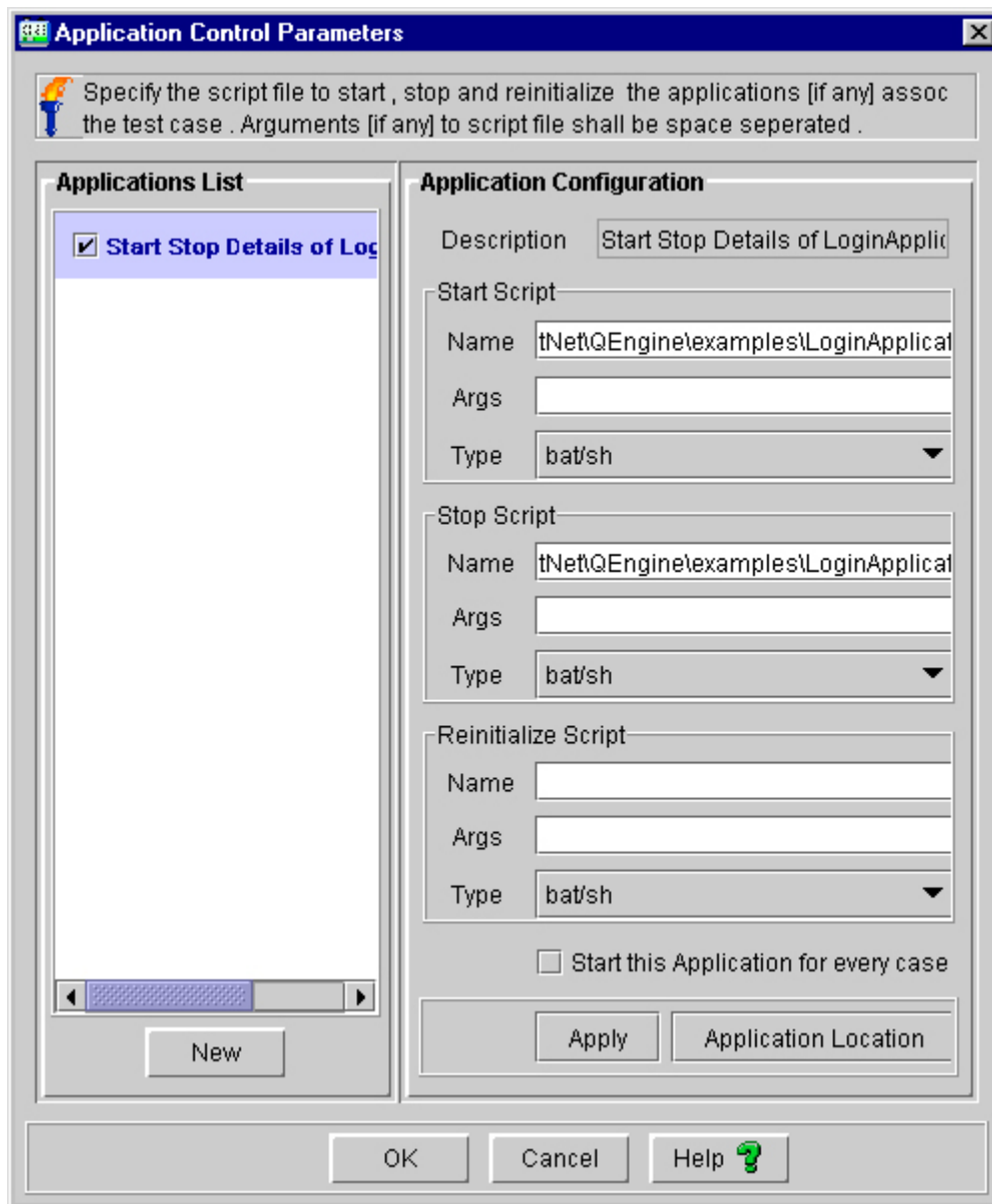
**Test Conditions**

Configure the Test Conditions associated with this test case.

**Test Conditions**

Back Next Cancel Help ?

6. Click the **Application Control Parameters** button. This displays a Application Control Parameters screen.
7. Enter **Start Stop Details of LoginApplication** in the **Description** field.
8. Give the location of LoginApplication's start and stop scripts as shown in the image below.



**Note:** Refer to Configuring Application Control Parameters for more information.

9. Click the **OK** button.
10. Click the **Next** button and proceed to configure validation parameters for the test case.
11. Select Validate against port states [ CHECK-PORT ].
12. Enter the following values as shown in the image below the table.

Field Name	Value
Port Number	2345
Protocol	TCP
Port State	Listening

13. Click **Commit to List** button.
14. Click **Finish** button. The automated test case serv-conf-099 is added onto the Suites Tree.

## Test Execution

### Executing Automated Test Case

1. Select the test case serv-conf-099 on the Suites Tree.
2. Choose the menu item **Start** under the **Test** menu.
3. Message **Test Execution Complete**, in the Progress Bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen.
2. Choose the Test Reports tab to view test results. The report will look as shown below:

The screenshot shows the 'Logs & Reports' window with the 'Test Reports' tab selected. The window contains a summary table and a detailed test case table.

Filter Test Cases with stat... All and test case i... All

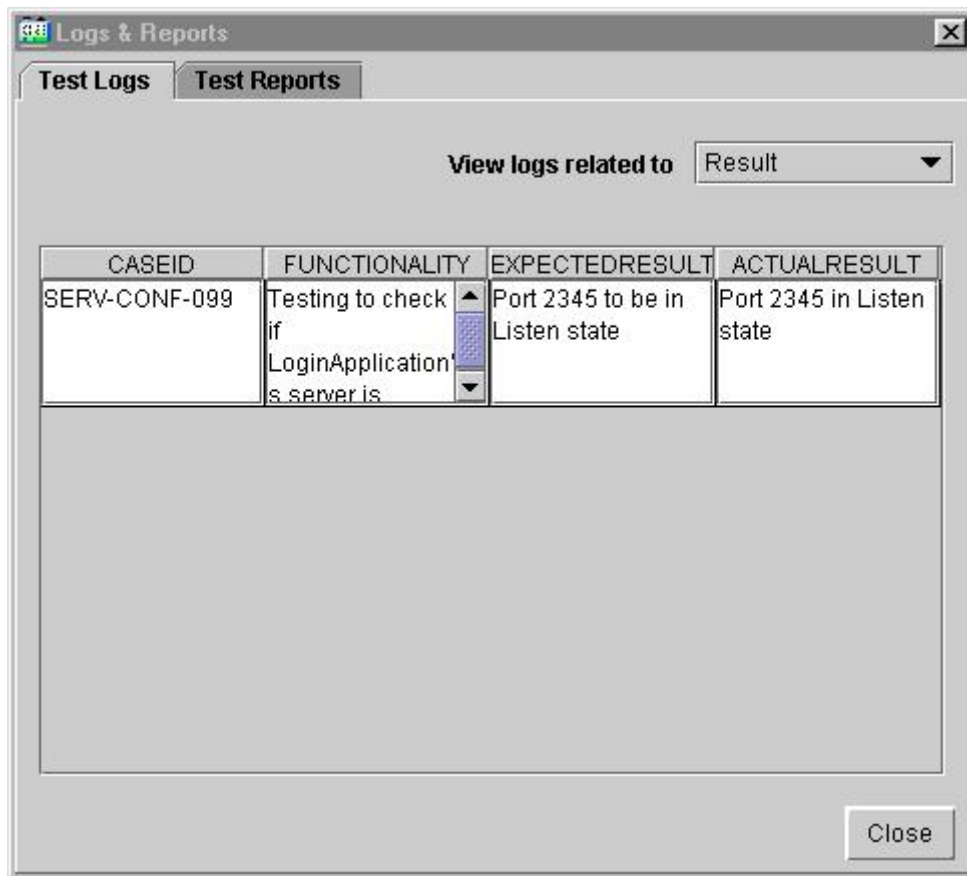
Passed	Failed	Incomp...	Error	Total
1	0	0	0	1

CASEID	SEVERITY	RESULT	FUNCTIONA...	DEBUG
SERV-CONF-099	ShowStopper	PASSED	Testing to che	--

Close

## Viewing Test Logs

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen with the test execution logs. The logs will look as shown below:



## Adding/Executing API Sequence Test Cases

### Adding Sequence Test Cases

- Test Documentation
- Test Creation
- Test Execution

## Test Documentation

The first step to achieve test automation is Test Documentation. Let us consider a test case, which is to check the sequential functioning of the methods `getProperty(String)` of the API `java.lang.System` and the method `CharAt(int)` of the API `java.lang.String`. The test plan is as defined below.

### Introduction

This test plan contains the test case to test sequential functioning of the methods `getProperty(String)` of `java.lang.System` and the method `CharAt(int)` of `java.lang.String`.

### Assumptions

Nil.

### Application Control Parameters

Nil

Test Case ID	Test Case Description	Test Case Procedure	Expected Behavior	Severity
seq-get-char-001	Testing the sequential functioning of the methods <code>getProperty(String)</code> and <code>CharAt(int)</code> .	<p><b>API Name</b></p> <p><code>java.lang.System</code> with static instance.</p> <p><b>Method to be tested</b></p> <p><code>getProperty(String)</code></p> <p><b>Argument value</b></p> <p><code>java.version</code></p> <p><b>API Name</b></p> <p><code>java.lang.String</code> with instance as return value of method <code>getProperty(String)</code> of <code>java.lang.System</code>.</p> <p><b>Method to be tested</b></p> <p><code>CharAt(int)</code></p> <p><b>Argument value</b></p> <p><code>0</code></p> <p><b>Validation type</b></p> <p>RetVsConst condition -equals value-1</p>	Should return 1.	Showstopper

## Test Creation

The Test Creation process has the following steps:

- Tool Selection
- Test Case Definition

### Tool Selection

From the test plan of the test case seq-get-char-001, we know that the test case's purpose is to test the function of a sequence of methods. Hence the type of testing to be performed is Sequential API testing. Hence, invoke the API Test Studio for automating this test case.

### Test Case Definition

1. Select the menu item **Methods in sequence** under the menu **Mode**. Now the API Test Studio is in sequence mode.
2. Select any Suite on the Suites Tree.
3. Choose the menu item **New Test Case** under the menu **File**. This displays a New Method Sequence screen.
4. Enter seq-get-char-001 in the **Case ID** field.
5. Enter **Checking sequential functioning of getProperty and CharAt methods** in the **Description** field.
6. Select the **Add** button under the heading Method Sequence. This displays Configure New Method screen
7. Click the button **Load New Class**. This displays a New API Details screen.
8. Enter java.lang.System in the **API Name** field.
9. Click **Ok**. The methods of the API java.lang.System are listed in the Configure New Method screen as shown below:
10. Select the method getProperty(String) from the **List of Methods** and click **OK**.
11. To add the method CharAt(int) in the Method Sequence List, follow the steps 5 & 6. This leaves you at the New API Details screen.
12. Enter java.lang.String in the **API Name** field.
13. Choose **Ok**. The methods of the API java.lang.String are listed in the Configure New Method screen.
14. Select the method CharAt(int) from the List of Methods and choose **OK**.

### Configuring Method Arguments and Validation

15. Select getProperty(String) in the Method Sequence List. Click Configure button. This displays a Method Arguments and Validation screen.
16. Click Instance button.
17. Select Class Object (static). Click **Commit to List** button. Click **OK**.
18. Under Method Arguments enter java.version in the **String** combo box as shown below:
19. Click **OK**.
20. Select CharAt(int) in the Method Sequence List. Click **Configure** button. This displays a Method Arguments and Validation screen.

21. Click Instance button. This displays the API Instance Creation screen.
22. Select **Another Methods Return Value**. Click **Commit To List**. Click **OK**. This leaves you at the Method Arguments and Validation screen.
23. Under Method Arguments enter 0 in the **int** combo box.
24. Click **Validation** button. This displays Method Validation screen.
25. Enter 1 in the **Expected return value** field. Click **Commit To List**. Click **OK**. This leaves you at the Method Arguments and Validation screen.
26. Click **OK**. This leaves you at the New Method Sequence screen.
27. Click **OK**. The automated test case seq-get-char-001 is added onto the Suites Tree.

## Test Execution

### Executing Automated Test Case

1. Select the test case seq-get-char-001 on the Suites Tree.
2. Choose the menu item **Start** under the **Test** menu.
3. Message **Test Execution Complete**, in Progress bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen.
2. Choose the Test Reports tab to view test results.

### Viewing Test Logs

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen with the test execution logs.

## Adding/Executing Performance Test Cases

### Adding Performance Test Cases

- Test Documentation
- URL Performance Test Creation
- API Performance Test Creation
- Resource Usage Performance Test Creation
- URL Performance Test Execution
- API Performance Test Execution
- Resource Usage Performance Test Execution

## Performance Test Plan

The first step to achieve test automation is the Test Plan or Test Documentation. Let us consider a test plan, which contains the test cases to check the performance of URL, API and resource usage. This test plan also contains the test case to measure the latency period of APIs. The test plan is defined below.

### Introduction

The test plan contains the sample performance test cases.

### Test Cases

- To test the performance of an API**  
 To test the performance of **com.adventnet.testtools.common.util.FileUtil**  
 To get the API Instance using the following properties.  
 APIName =**com.adventnet.testtools.common.util.FileUtil**  
 Instance creation=**Static**
- To test the resource usage of a process**  
 To get the CPU and memory usage of the defined process
- To measure the latency period of APIs.**  
 To measure the latency period of **addUser()** and **addAction()**.

### Application Control Parameters

Nil

Test Case ID	Test Case Description	Test Case Procedure	Expected Behavior	Severity
PERF-SampleAPI-001	Testing the FileUtil api performance.	<ol style="list-style-type: none"> <li>Specify the API <b>com.adventnet.testtools.common.util.FileUtil</b></li> <li>Specify the No. of users &amp; No. of samples.</li> <li>Select the API Instance Type as Static.</li> <li>Select any Methods and Click Configure Arguments button.</li> <li>Specify the arguments if needed.</li> <li>Select the agent.</li> <li>Click <b>Finish</b>.</li> </ol>	Test Case should execute and finally it generates the table and graph reports.	Showstopper
PERF-TRANS-CODEINST-001	To measure the latency period of AddUserPanel API and UserAdminClientAPI.	<ol style="list-style-type: none"> <li>Load the classes <b>com.adventnet.qengine.project.AddUserPanel</b> and <b>com.adventnet.testtools.sample.UserAdminClientAPI</b></li> </ol>	Test Case should execute and finally it generates the table and graph reports.	Showstopper

Test Case ID	Test Case Description	Test Case Procedure	Expected Behavior	Severity
		<ol style="list-style-type: none"> <li>2. Select the method <b>addAction()</b> from <b>com.adventnet.qengine.project.AddUserPanel.</b></li> <li>3. Select the <b>addUser()</b> from <b>com.adventnet.testtools.sample.UserAdminClientAPI.</b></li> <li>4. Specify the transaction script.</li> <li>5. Enter the SampleCount and SampleInterval.</li> <li>6. Select the agent.</li> <li>7. Click <b>Finish.</b></li> </ol>		
PERF-SampleResUsage-001	To get the resource usage details of process <b>com.adventnet.qengine.performance</b>	<ol style="list-style-type: none"> <li>1. Specify the process name <b>com.adventnet.qengine.performance</b></li> <li>2. Specify the No. of users and No. of samples.</li> <li>3. Select the Interval.</li> <li>4. Select the agent.</li> <li>5. Click <b>Finish.</b></li> </ol>	Test Case should execute and finally it generates the table and graph reports.	Showstopper

## API Performance Test Creation

The Test Creation process consists of 2 steps:

- Tool Selection
- Test Case Definition

### Tool Selection

From the test plan, we know that the purpose of the test case is to measure the performance of an API. Hence the type of testing to be performed is API Performance testing. Hence, invoke the API Test Studio and choose the API Performance Data Collector [API] option for automating the API test case.

### Test Case Definition

To define the test case follow the below given steps:

1. Select the Suite **LoginAPPJ2SESuite** from the Suites Tree.
2. Choose the menu item **New Test Case** under the menu **File**. This opens the Performance Test Case Creation screen.
3. Enter "PERF-SampleAPI-001" in the **Case ID** field.
4. Enter "Sample API testing case" in the **Description** field.
5. Select the **Severity** as "ShowStopper" from the combo box.
6. Enter the **Description** for test run as "Sample API Run".
7. Enter the **Maximum Time Limit** as "1800".
8. Ignore Application Control Parameters.
9. Click **Test Conditions** button and choose the tab **Check Condition**.
  1. Select the option **Collect Data after fixed time delay (WAIT)** and specify the **Wait Time** as 20 and select the Agent Name as "M1"
  2. Click **Commit To List** and click **OK**.
10. Click **Next**.
11. In the **Data Collection** screen, choose the **API Performance Data Collector [API]** option.
12. Click the **Load API option** to load the API `com.adventnet.testtools.common.util.FileUtil`.
13. Set the required classpath and click **OK**.
14. Select the **Instance Type** as "Static".
15. Configure the method arguments as follows:
  1. Select the method **countOccurrenceInFile(String, String)** from the method list and click **Configure Arguments** button. Specify the first string argument as "README.html" and the second string argument as "QEngine". Click **OK**.
  2. Select the method **doesFileExists(String)** from the method list and click **Configure Arguments** button. Specify the string argument as "README.html" and click **OK**.
  3. Select the method **getLinesInFile(String)** from the method list and click **Configure Arguments** button. Specify the string argument as "README.html" and click **OK**.
  4. Select the method **getTimeStamp()** from the method list.
  5. Select the method **isStringInFile(String, String)** from the method list and click **Configure Arguments** button. Specify the first string argument as "README.html" and the second string argument as "QEngine". Click **OK**.

16. Select the Agent Name as "M1".
17. Enter **Number of Users** as "1".
18. Enter **Number of Samples** as "2".
19. Enter **Sample Interval** as "10" seconds.
20. Click **Commit To List** and click **OK**.
21. Click **Finish** button. The automated test case PERF-SampleAPI-001 is added onto the Suites Tree.

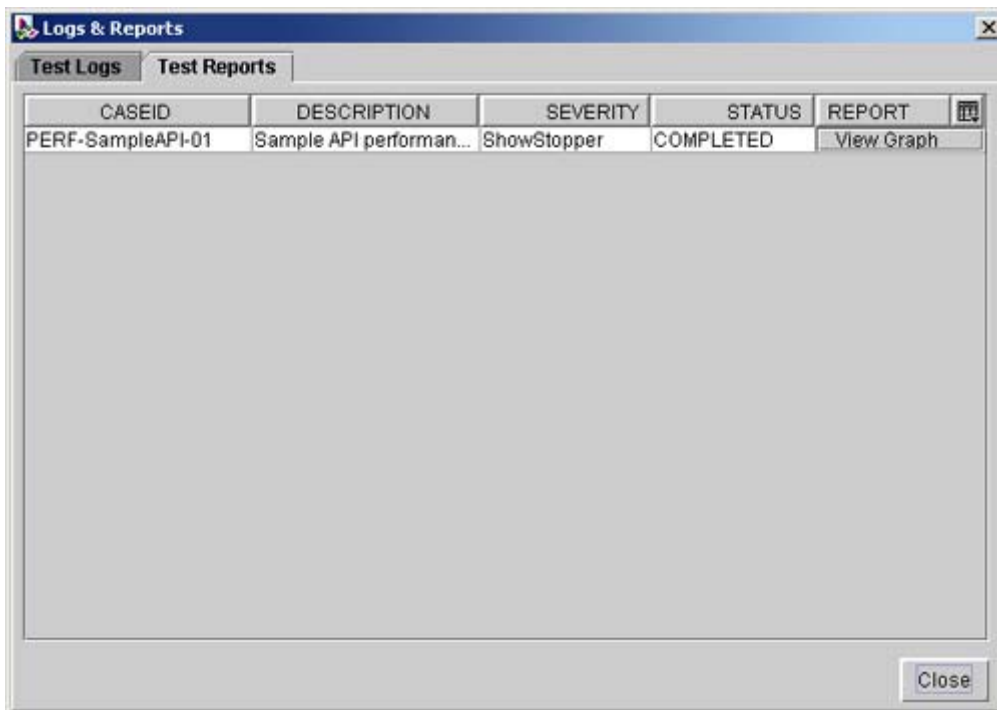
## API Performance Test Execution

### Executing API Performance Test Case

1. Select the test case PERF-API-001 from the Suites Tree.
2. Choose the menu item **Start** under the **Test** menu.
3. Message **Test Execution Complete**, in the Progress bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen.
2. Choose the Test Reports tab to view test results. The report will look as shown below:



3. Click the **View Graph** button. This displays the table view report and graph report for API performance test case as shown below:

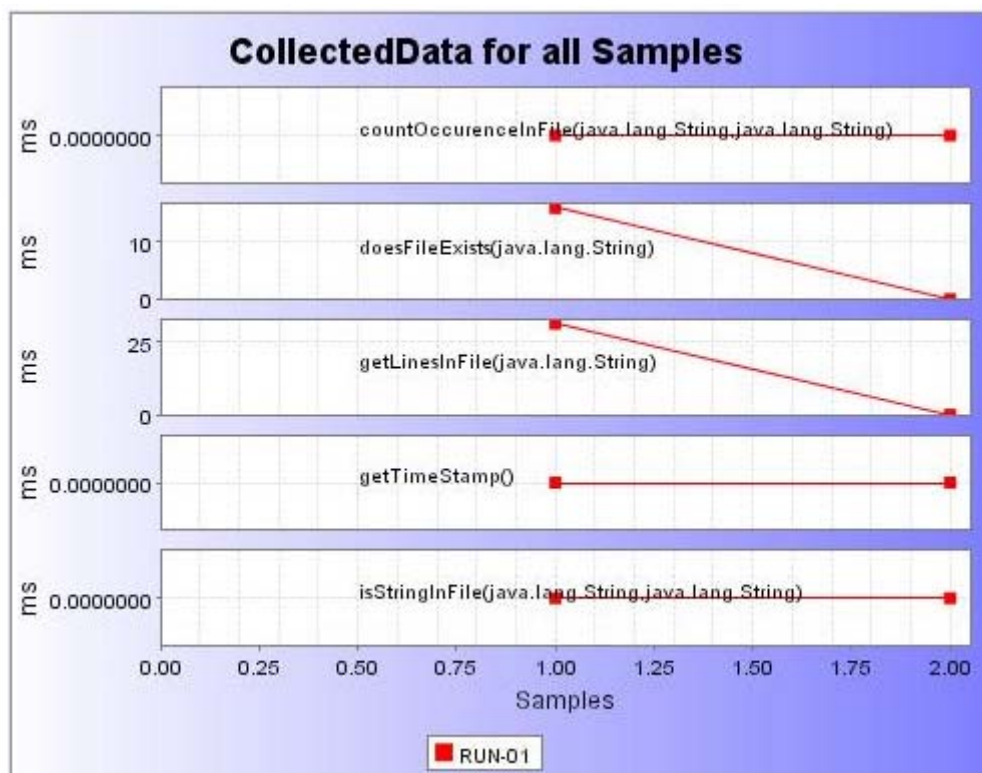
Table Report

**PERF-SampleAPI-01**

Parameter	RUN-01
Description	Sample API Run
Status	COMPLETED
Debug_Info	--
<b>countOccurenceInFile(java.lang.String,java.lang.String)</b>	0.0 ms
<b>doesFileExists(java.lang.String)</b>	8.0 ms
<b>getLinesInFile(java.lang.String)</b>	15.5 ms
<b>getTimeStamp()</b>	0.0 ms
<b>isStringInFile(java.lang.String,java.lang.String)</b>	0.0 ms

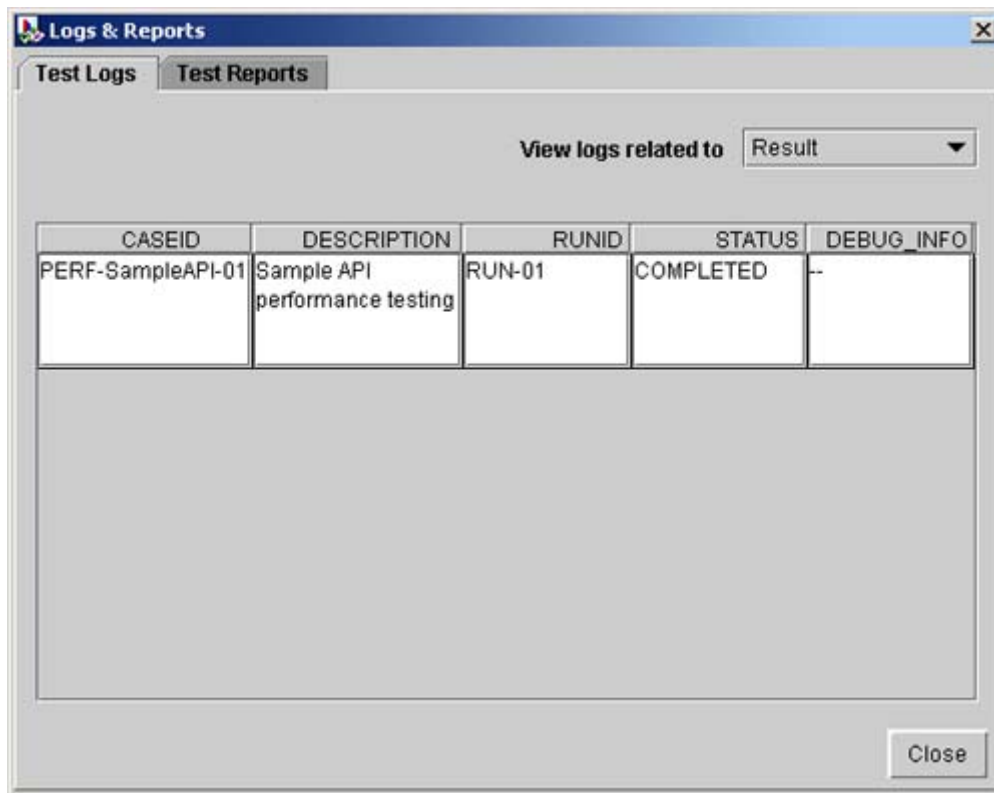
[Detailed Table](#) [Detailed Graph](#)

Graph Report



## Viewing Test Logs

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen with the test execution logs. The logs will look as shown below:



## Resource Usage Performance Test Creation

The Test Creation process consists of 2 steps:

- Tool Selection
- Test Case Definition

### Tool Selection

From the test plan, we know that the purpose of the test case is to measure the resource usage for the specified process. Hence the type of testing to be performed is Resource Usage Performance testing. Hence, invoke the Performance Test Studio and choose the Resource Utilization option for automating the resource usage performance test case.

### Test Case Definition

To define the test case follow the below given steps:

1. Select the Suite **LoginAPPJ2SESuite** from the Suites Tree.
2. Choose the menu item **New Test Case** under the menu **File**. This opens the Performance Test Case Creation screen.
3. Enter "PERF-SampleResUsage-001" in the **Case ID** field.
4. Enter "Sample Resource Usage Testing" in the **Description** field.
5. Select the **Severity** as "ShowStopper" from the combo box.
6. Ignore Application Control Parameters.
7. Click **Test Conditions** button and choose the tab **Check Condition**.
  1. Select the option **Collect Data after fixed time delay (WAIT)** and specify the **Wait Time** as 20 and select the Agent Name as "M1"
  2. Click **Commit To List** and click **OK**.
8. Click **Next**.
9. In the **Data Collection** screen, choose the **Resource Utilization (CPU/Memory) [resourceusage]** option.
10. Specify the Process Name as "com.adventnet.qengine.performance".
11. Select the Agent Name as "M1".
12. Select "Enable Parallel Data Collection".
13. Specify **Sample Count** as "4".
14. In the **Monitor** parameters, select **Every "10" Second(s)**.
15. Click **Commit To List** and click **OK**.
16. Click **Finish** button. The automated test case PERF-SampleResUsage-001 is added onto the Suites Tree.

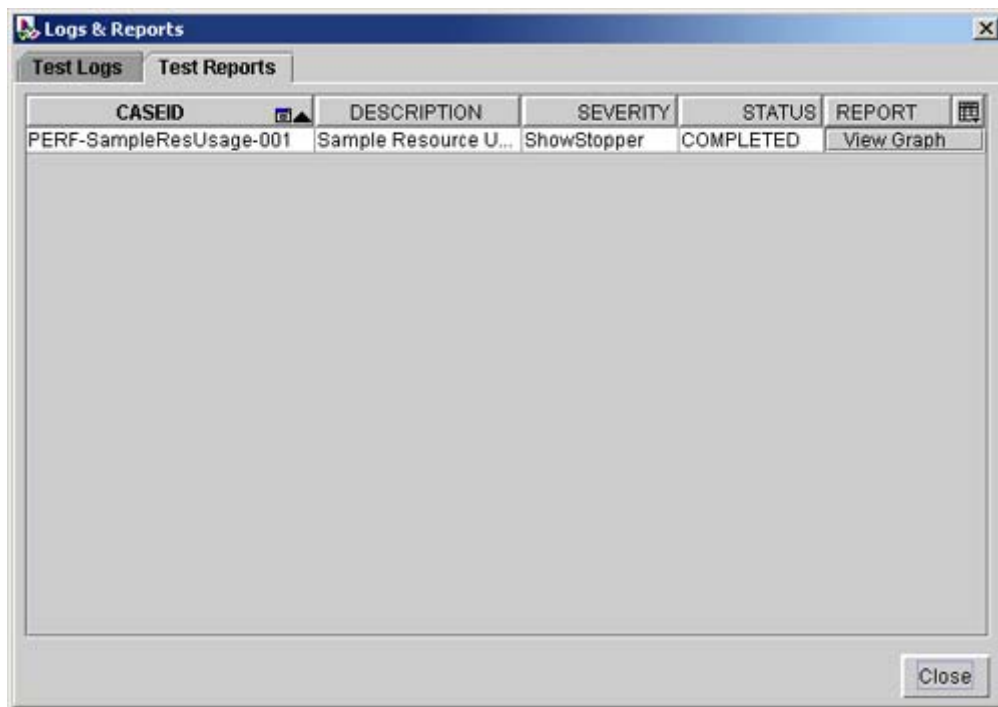
## Resource Usage Performance Test Execution

### Executing Resource Usage Performance Test Case

1. Select the test case PERF-ResUsageI-001 from the Suites Tree.
2. Choose the menu item **Start** under the **Test** menu.
3. Message **Test Execution Complete**, in the Progress bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen.
2. Choose the **Test Reports** tab to view the test report. The report will look as shown below:



3. Click the **View Graph** button. This displays the table view report and graph report for Resource Usage performance test case as shown below:

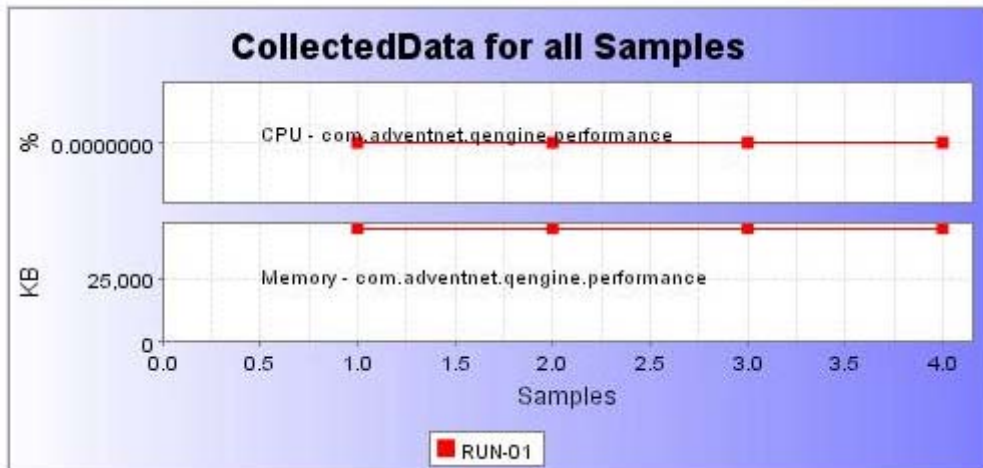
#### Table Report:

### PERF-SampleResUsage-001

Parameter	RUN-01
Description	Sample Resource Usage Testing Run
Status	COMPLETED
Debug_Info	--
<b>CPU - com.adventnet.qengine.performance</b>	0.0 %
<b>Memory - com.adventnet.qengine.performance</b>	44628.0 KB

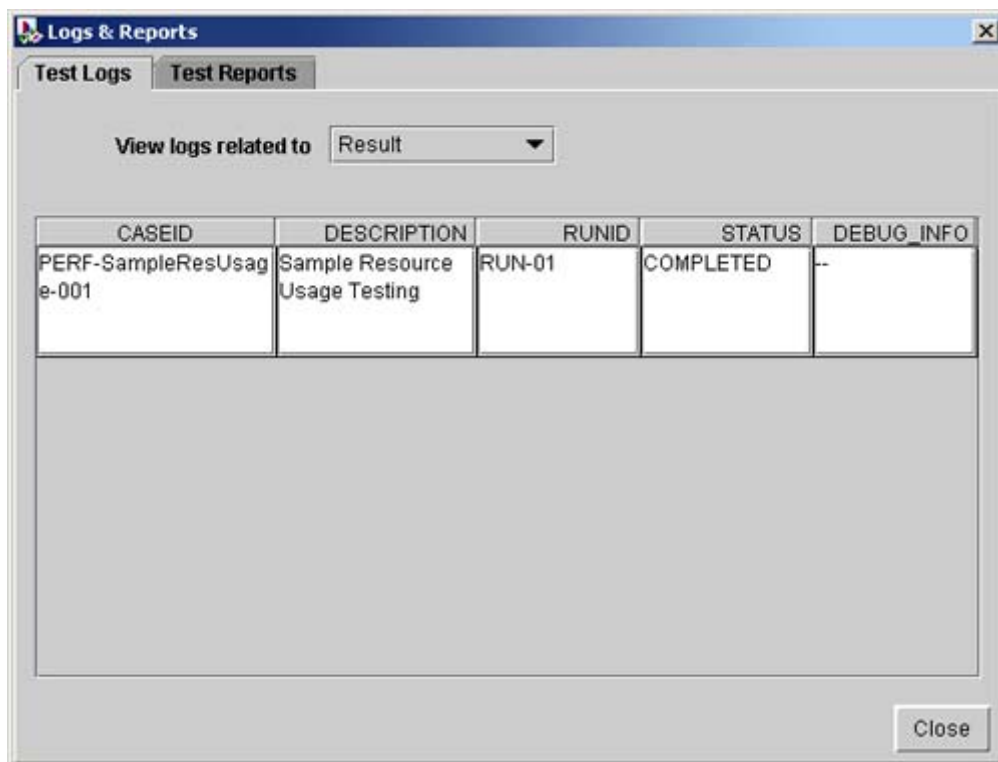
[Detailed Table](#) [Detailed Graph](#)

**Graph Report:**



**Viewing Test Logs**

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen with the test execution logs. The logs will look as shown below:



## Code Instrumentation Test Creation

The Test Creation process consists of 2 steps:

- Tool Selection
- Test Case Definition

### Tool Selection

From the test plan, we know that the purpose of the test case is to measure the latency period of an API. Hence the type of testing to be performed is Code Instrumentation Data Collection. Hence, invoke the API Test Studio and choose the Code Instrumentation Data Collection [INST] option.

### Test Case Definition

To define the test case follow the below given steps:

1. Select the Suite **LoginAPPJ2SESuite** from the Suites Tree.
2. Choose the menu item **New Test Case** under the menu **File**. This opens the Performance Test Case Creation screen.
3. Enter "PERF-TRANS-CODEINST-001" in the **Case ID** field.
4. Enter "Latency of User Addition Transaction" in the **Description** field.
5. Select the **Severity** as "ShowStopper" from the combo box.
6. Enter the **Description** for test run as "User Addition Transaction".
7. Enter the **Maximum Time Limit** as "1800".
8. Select the Application Control Parameters button and select the Start Script as **startauthserver** and Stop Script as **stopauthserver** from `<QEngine_Home>/examples/LoginApplication/bin` directory.
9. Click **Test Conditions** button and choose the tab **Check Condition**.
  1. Select the option **Collect Data after fixed time delay (WAIT)** and specify the **Wait Time** as 20 and select the Agent Name as "M1"
  2. Click **Commit To List** and click **OK**.
10. Click **Next**.
11. In the **Data Collection** screen, choose the **Code Instrumentation Data Collection [INST]** option.
12. Click the **Add** button. In the **Configure New Method** screen, click the **Classpath** button to load the classes **com.adventnet.qengine.project.AddUserPanel** and **com.adventnet.testtools.sample.UserAdminClientAPI**.
13. From the list box, select the **com.adventnet.qengine.project.AddUserPanel** and click the **Load Methods** button.
14. Select the method **addAction()** from the **All Methods** list.
15. Click **OK**.
16. From the list box, select the **com.adventnet.testtools.sample.UserAdminClientAPI** and click the **Load Methods** button.
17. Select the method **addUser(com.adventnet.testtools.sample.Message)** from the **All Methods** list.

18. Click the **Transaction Script** button and select the scriptname in the name field as AddUser from <QEngine\_Home>/projects/LoginAppJ2SESuite/testing directory.
19. Select the Agent Name as "M1".
20. Enter **Number of Samples** as "1".
21. Enter **Sample Interval** as "10" seconds.
22. Click **Commit To List** and click **OK**.
23. Click **Finish** button. The automated test case PERF-TRANS-CODEINST-001 is added onto the Suites Tree.

## Code Instrumentation Test Execution

### Executing Code Instrumentation Performance Test Case

1. Select the test case PERF-TRANS-CODEINST-001 from the Suites Tree.
2. Choose the menu item **Start** under the **Test** menu.
3. Message **Test Execution Complete**, in the Progress bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item **Test Results** under the **View** menu. This displays the Logs & Reports screen.
2. Choose the Test Reports tab to view test results. The report will look as shown below:

## **Adding/Executing Web Services Test Cases**

### **Adding/Executing Web Services Test Cases**

- Test Documentation
- Test Creation
- Test Execution

## Test Documentation

The first step to achieve test automation is Test Documentation. Let us consider a test case, which is to test the methods of UserAdminClientAPI using SOAP Lookup. The test plan is as defined below.

### Introductions

This test plan contains the test case to invoke the methods of UserAdminClientAPI using SOAP Lookup.

### Assumptions

Nil.

### Application Control Parameters

Nil

Test Case ID	Test Case Description	Test Case Procedure	Expected Behavior	Severity
SOAP-Test-007	Testing the UserAdminClientAPI using SOAP Lookup.	<b>API Name.</b> UserAdminClientAPI  <b>Method to be tested</b> addUsers  <b>Argument value</b> user7,test7  <b>Validation type</b> Validate against database [SEARCH-DB]	Should return the name as user7 and password as test7.	Showstopper

## Test Creation

The Test Creation process consists of the below three steps:

- Tool Selection
- Pre-requisites
- WebServices Setup
- Test Case Definition

### Tool Selection

From the test plan of the test case **SOAP-Test-007**, we know that the test case's purpose is to test the methods of **UserAdminClientAPI** using **SOAP Lookup**. Hence the type of testing to be performed is **Unit Testing**. Hence, invoke the **API Test Studio** for automating this test case.

### Pre-requisites

1. Start the API test studio using **startapiteststudio.bat/.sh** file in **<QEngine\_Home>/bin** directory.
2. Choose the **SOAPTestSuite** tree node from the tree-view of API Test Studio. In the right hand-side panel, browse and edit the '**Home Directory**' field to set the application home as **<QEngine\_Home>/examples/LoginApplication** directory (where **<QEngine\_Home>** is the directory where you have installed QEngine).
3. Set the **Java\_Home**: Choose the **Settings->Compilation Settings** menu item in API Test Studio to set the **JAVA\_HOME** directory.

### WebServices Setup

The default Suite SOAPTestSuite is bundled with QEngine. This Suite includes the complete setup to start the UserAdministration application on a Web Server which serves as the Webservice. This Suite also includes the client application which interacts with the Webservices via SOAP Lookup to invoke the methods. The Application Control Parameters (ACP) option in the SOAPTestSuite includes the following batch/script files to start the Webservices:

- **starttomcatserver.bat/.sh** file which starts the Webserver.
- **startuseradminserver.bat/.sh** which deploys or runs the UserAdministration application on the Webserver to serve as the WebService.
- **stopuseradminserver.bat/.sh** file to undeploy the UserAdministration application on the Webserver and stop the Webserver.

The **WSDL STUBS** for **UserAdminClientAPI** is generated using the WSDL STUB GENERATOR Tool invoked from the Tools menu of API Test Studio. The generated WSDL\_STUBS for UserAdminClientAPI is displayed in the tree-view of SOAPTestSuite. This acts as the client application. Default test cases are added to the UserAdminClientAPI to invoke the methods via SOAP Lookup.

### Test Case Definition

1. Choose File->Open Suite menu item to open the SOAPTestSuite.
2. Choose localhost.axis.services.UserAdminClientAPI.UserAdminClientAPISoapBindingStub from the tree node.
3. Choose addUsers(String[],String[]) method and choose the File->New Test Case menu item.
4. Enter the 'CASE ID' as SOAP-Test-007.

5. Enter the 'Description' as "Testing Webservices".
6. Choose the 'Instance' tab and select the SOAP LookUp radio option from the 'API Instance Creation' screen.
7. The URL is displays as http://localhost:9091/axis/services/UserAdminClientAPI.
8. Click OK.
9. Enter the 'Method Arguments' as user7 and test7.
10. Click Next.
11. In the 'Validation Input' screen, choose Validate against database [SEARCH-DB] option.
12. Select the 'Database' as MYSQL from the combo box and select the Browse button to edit the database name as UserInfo.
13. Click Edit.
14. Click Commit To List and click OK.
15. Specify the 'Query' as Select \* from LoginTable where name='user7'.
16. Choose Table Values from Verify.
17. From the 'Condition' combo-box, select the condition as Contains.
18. Click the 'Expected Table Values' button and specify the values for 'Enter Column Names' field as name,password.
19. In the 'Enter Row Count' field, specify the row count as 1.
20. Click Show Table button and enter the values for name as user7 and password as test7.
21. Click OK.
22. Click Commit To List.
23. Click Finish. The test case SOAP-Test-007 should be successfully added to the tree.

## Test Execution

### Executing Automated Test Case

1. Select the test case SOAP-Test-007 from the SuitesTree.
2. Choose the menu item Start under the Test menu.
3. Message Test Execution Complete, in the Progress Bar, confirms completion of test execution.

### Viewing Test Results

1. Choose the menu item Test Results under the View menu. This displays the Logs & Reports screen.
2. Choose the Test Reports tab to view test results.

### Viewing Test Logs

Choose the menu item Test Results under the View menu. This displays the Logs & Reports screen with the test execution logs.

