

Oracle® Application Testing Suite

Getting Started Guide

Version 9.01 for Microsoft Windows (32-Bit)

E15487-02

December 2009

Oracle Application Testing Suite Getting Started Guide, Version 9.01 for Microsoft Windows (32-Bit)

E15487-02

Copyright © 1997, 2009, Oracle and/or its affiliates. All rights reserved.

Primary Author: Rick Santos

Contributing Author: Mary Anna Brown

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface	xiii
Audience	xiii
Documentation Accessibility	xiv
Related Documents	xiv
Conventions	xv
1 Introduction	
1.1 About Oracle Application Testing Suite Administrator	1-1
1.1.1 Administrator Feature Highlights	1-2
1.2 About Oracle OpenScript	1-2
1.2.1 OpenScript Feature Highlights	1-3
1.3 About Oracle Functional Testing for Web Applications	1-4
1.3.1 Oracle Functional Testing for Web Applications Feature Highlights	1-5
1.4 About Job Scheduler	1-6
1.4.1 Job Scheduler Feature Highlights	1-7
1.5 About Oracle Load Testing for Web Applications	1-8
1.5.1 Oracle Load Testing for Web Applications Feature Highlights	1-9
1.6 About Oracle Test Manager for Web Applications	1-10
1.6.1 Oracle Test Manager for Web Applications Feature Highlights	1-10
1.7 Oracle Application Testing Suite Database Configuration	1-11
1.7.1 Database Configuration Feature Highlights	1-11
2 Oracle Application Testing Suite Basics	
2.1 Installing and Starting Oracle Application Testing Suite	2-1
2.2 Oracle Application Testing Suite Administrator Main Window Features	2-1
2.2.1 Users Tab	2-2
2.2.2 Roles Tab	2-5
2.2.3 Projects Tab	2-6
2.2.4 Fields Tab	2-8
2.3 Oracle OpenScript Main Window Features	2-10
2.3.1 Script View	2-11
2.3.1.1 Tree View	2-11
2.3.1.2 Java Code	2-12
2.3.2 Details View	2-13
2.3.3 Problems View	2-14

2.3.4	Properties View	2-14
2.3.5	Console View	2-15
2.3.6	Results View	2-15
2.3.7	Other Views	2-16
2.4	Oracle Functional Testing for Web Applications Main Window Features	2-16
2.4.1	Visual Script Pane	2-17
2.4.2	Browser Pane	2-18
2.4.3	Playback Results Pane	2-19
2.5	Job Scheduler Main Window Features	2-19
2.5.1	Visual Script Job Pane	2-20
2.5.2	Results Pane	2-20
2.5.3	Job Scheduler Wizard	2-21
2.6	Oracle Load Testing for Web Applications Main Window Features	2-22
2.6.1	Build Scenario Tab	2-22
2.6.2	Set up Autopilot Tab	2-23
2.6.3	Watch Virtual User Grid Tab	2-24
2.6.4	View Run Graphs Tab	2-24
2.6.5	Create Reports Tab	2-25
2.6.6	ServerStats	2-26
2.7	Oracle Test Manager for Web Application Main Window Features.....	2-26
2.7.1	Requirements Tab.....	2-26
2.7.2	Tests Tab.....	2-28
2.7.3	Issues Tab.....	2-30
2.7.4	Reports Tab.....	2-31
2.7.5	Dashboard Tab	2-32

3 Oracle OpenScript Tutorial

3.1	Starting the Avitek Medical Records Sample Application	3-1
3.2	Starting Oracle OpenScript.....	3-2
3.3	Example 1: Creating a Web Functional Test Script.....	3-2
3.3.1	Creating a Web Functional Test Script Project	3-2
3.3.2	Recording a Web Functional Test Script	3-2
3.4	Example 2: Working with Scripts	3-3
3.4.1	Viewing Information About Script Items.....	3-4
3.4.2	Using the Java Code View	3-6
3.5	Example 3: Playing Back a Web Functional Test Script	3-6
3.6	Example 4: Adding Tests to the Script.....	3-8
3.6.1	Inserting a Server Response Test Case	3-8
3.6.2	Inserting a Table Test	3-9
3.7	Example 5: Creating an HTTP Test Script with Databanks	3-12
3.7.1	Creating an HTTP Script Project.....	3-12
3.7.2	Recording an HTTP Script.....	3-12
3.7.3	Viewing the Parameters in the Script	3-12
3.7.4	Configuring Databanks with OpenScript Scripts	3-13
3.7.5	Mapping Script Parameters to Databank Fields	3-14
3.7.6	Inserting a Text Matching Test	3-15
3.7.7	Playing Back the Script with Iterations	3-16

3.8	Stopping the Avitek Medical Records Server	3-18
-----	--	------

4 Oracle Functional Testing for Web Applications Tutorial

4.1	Initializing the Tutorial	4-1
4.2	Example 1: Recording a New Visual Script	4-1
4.2.1	Start Oracle Functional Testing for Web Applications.....	4-2
4.2.2	Start a Recording	4-2
4.2.3	Navigate the Web Site.....	4-3
4.2.4	Stop the Recording	4-3
4.2.5	Save the Script	4-3
4.3	Example 2: Working with Visual Scripts.....	4-3
4.3.1	Viewing Information About a Visual Script Item.....	4-4
4.3.2	Turning Automatic Testing On and Off.....	4-5
4.3.3	Modifying Default Tests	4-6
4.4	Example 3: Playing Back a Visual Script	4-7
4.5	Example 4: Analyzing Test Failures	4-10
4.5.1	Ignoring Failures.....	4-11
4.5.2	Accepting Changes Shown in the Script	4-12
4.5.3	Rejecting Problems Shown in the Script.....	4-12
4.6	Example 5: Adding Test Cases to the Visual Script.....	4-13
4.6.1	Record a New Script.....	4-14
4.6.2	Stop the Recording	4-14
4.6.3	Insert a Text Matching Test Case.....	4-14
4.6.4	Insert a Server Response Test Case	4-15
4.6.5	Insert a Form Element Test Case	4-16
4.6.6	Insert Table Test Test Case	4-18
4.7	Example 6: Using the Data Bank Wizard on a Search Form.....	4-20
4.7.1	Recording a Search	4-21
4.7.2	Viewing the Parameters in the Visual Script	4-21
4.7.3	Using the Data Bank Wizard to Map Variables	4-21
4.7.4	Using the Data Bank Wizard to Bind to a Data Bank.....	4-23
4.7.5	View the Data Bank Parameters in the Visual Script	4-26
4.7.6	Insert a Text Matching Test Case.....	4-27
4.7.7	Play Back the Script with Iterations	4-27
4.7.8	Analyzing a Playback Failure	4-28
4.7.9	Save the Script and the Results Log	4-29
4.8	Example 7: Using the Data Bank Wizard on a Registration Form.....	4-29
4.8.1	Recording Information in a Form.....	4-29
4.8.2	Inserting a Text Matching Test Case.....	4-29
4.8.3	Viewing the Parameters in the Visual Script	4-29
4.8.4	Using the Data Bank Wizard to Map Variables	4-30
4.8.5	Using the Data Bank Wizard to Bind to Data Source.....	4-31
4.8.6	View the Data Bank Parameters in the Visual Script	4-34
4.8.7	Play Back the Script with Data Iterations.....	4-34
4.8.8	Analyzing a Playback Failure	4-35
4.8.9	Save the Script and the Results Log	4-36

5 Job Scheduler Tutorial

5.1	Example 1: Creating a Job and Schedule	5-1
5.1.1	Starting Job Scheduler	5-1
5.1.2	Specifying the Scripts	5-1
5.1.3	Specifying the Job Notifications	5-2
5.1.4	Specifying Email Notifications	5-3
5.1.5	Scheduling the Job	5-4
5.1.6	Saving the Schedule.....	5-5
5.1.7	Playing Back the Job	5-5
5.1.8	Activating the Schedule	5-5
5.2	Example 2: Editing a Job	5-6
5.3	Example 3: Editing a Schedule.....	5-7

6 Oracle Load Testing for Web Applications Tutorial

6.1	Example 1: Performing a Simple Load Test.....	6-2
6.1.1	Starting Oracle Load Testing for Web Applications and Specifying the Workspace	6-2
6.1.2	Specifying a Scenario Profile.....	6-3
6.1.3	Running the Scenario Profile Using Autopilot.....	6-4
6.2	Example 2: Adding Data Sources	6-5
6.3	Example 3: Editing Data Sources.....	6-12
6.4	Example 4: Creating a Scenario with Multiple Profiles.....	6-13
6.4.1	Adding a Virtual User Profile to the Scenario	6-13
6.4.2	Saving Data for Reporting.....	6-13
6.4.3	Saving the Scenario.....	6-15
6.5	Example 5: Running Multiple Profiles.....	6-15
6.5.1	Running the Scenario Profiles Using Autopilot	6-15
6.5.2	Viewing Performance Statistics	6-17
6.5.3	Viewing Graphs	6-19
6.6	Example 6: Controlling Virtual Users	6-21
6.6.1	Modifying the Run Attributes	6-21
6.6.2	Viewing Virtual User Actions.....	6-22
6.6.3	Stopping an Individual Virtual User	6-23
6.6.4	Aborting an Individual Virtual User	6-23
6.6.5	Stopping All Virtual Users	6-23
6.6.6	Aborting All Virtual Users	6-23
6.7	Example 7: Generating Reports	6-23
6.7.1	Generating Reports from Oracle Load Testing for Web Applications	6-23
6.7.2	Opening the Chart in Microsoft Excel	6-25
6.7.3	Viewing Scenario and Session Reports.....	6-27
6.8	Example 8: Creating User-Defined Profiles	6-28
6.8.1	Adding Visual Scripts to the Sections Tree.....	6-29
6.8.2	Adding Synchronization Points to the Sections Tree	6-30
6.8.3	Moving Items in the Sections Tree	6-31
6.8.4	Editing User-Defined Profiles.....	6-33

7 Oracle Test Manager for Web Applications Tutorial

7.1	Starting Oracle Test Manager for Web Applications.....	7-1
7.2	Opening the Sample Project	7-2
7.3	Example 1: Adding a Requirement	7-3
7.4	Example 2: Adding a Test.....	7-7
7.4.1	Adding a Manual Test	7-7
7.4.2	Adding an Automated Test.....	7-11
7.5	Example 3: Running a Test	7-16
7.5.1	Running a Manual Test.....	7-16
7.5.2	Running an Automated Test.....	7-18
7.6	Example 4: Adding an Issue	7-19
7.7	Example 5: Creating Reports.....	7-25

List of Figures

1-1	Administrator Main Window (Test Manager Database)	1-2
1-2	OpenScript Tree View Hierarchy	1-3
1-3	Visual Script Tree Hierachy	1-5
1-4	Job Scheduler Main Window	1-6
1-5	Job Scheduler Wizard Window	1-7
1-6	Job Scheduler Schedule Window.....	1-7
1-7	Oracle Load Testing for Web Applications Virtual Users View	1-8
1-8	Concurrent Object Requests vs. Sequential Object Requests	1-9
1-9	Oracle Test Manager for Web Applications Main Window	1-10
2-1	Oracle Application Testing Suite Administrator Main Window.....	2-2
2-2	Users Tab for Oracle Load Testing for Web Applications Users	2-3
2-3	Users Tab for Oracle Test Manager for Web Applications Users	2-4
2-4	Roles Tab for Oracle Test Manager for Web Applications Users	2-5
2-5	Projects Tab for Oracle Test Manager for Web Applications Users	2-7
2-6	Fields Tab for Oracle Test Manager for Web Applications Users	2-8
2-7	Oracle OpenScript Main Window	2-11
2-8	Script Tree View	2-12
2-9	Script Java Code View	2-12
2-10	Java Code View Intellisense Window	2-13
2-11	Details View Showing Screenshot Tab View	2-13
2-12	Problems View	2-14
2-13	Properties View	2-15
2-14	Console View	2-15
2-15	Results View	2-15
2-16	Oracle Fuctional Testing for Web Applications Main Window.....	2-17
2-17	Visual Script Pane	2-18
2-18	Browser Pane	2-19
2-19	Playback Results Pane.....	2-19
2-20	Job Scheduler Main Window	2-20
2-21	Job Scheduler Job Pane.....	2-20
2-22	Job Scheduler Results Pane.....	2-20
2-23	Job Scheduler Wizard.....	2-21
2-24	Job Scheduler Schedule Window.....	2-21
2-25	Oracle Load Testing for Web Applications Main Window	2-22
2-26	Build Scenarios Tab	2-23
2-27	Autopilot Tab.....	2-24
2-28	Watch Virtual User Grid Tab	2-24
2-29	View Run Graphs Tab.....	2-25
2-30	Create Reports Tab.....	2-25
2-31	ServerStats Metric Profiles Window	2-26
2-32	Requirements Tab	2-27
2-33	Tests Tab.....	2-28
2-34	Issues Tab	2-30
2-35	Reports Tab	2-31
2-36	Dashboard Tab	2-32
3-1	OpenScript Script Project Tree	3-2
3-2	OpenScript Script Tree After Recording.....	3-3
3-3	Script Tree with Expanded Initialize Section.....	3-4
3-4	Script Tree with Expanded Run Section.....	3-4
3-5	Script Tree with Expanded Step Group Node	3-5
3-6	Script Think Node.....	3-5
3-7	Script Think Node Properties.....	3-5
3-8	Text Matching Test Properties	3-6
3-9	Script Playback Results in the Results View	3-7

3-10	Results Report in the Details View	3-7
3-11	Details View Showing the Web Functional Test Comparison Tab.....	3-8
3-12	Server Response Test Properties Dialog Box	3-9
3-13	Server Response Test Added to the Script Tree	3-9
3-14	Table Test Properties Dialog Box.....	3-10
3-15	Table Test Properties Dialog Box with Captured Data	3-11
3-16	Table Test Added to the Script Tree.....	3-11
3-17	Script Parameter Values for HTTP Post Data	3-13
3-18	Add Databank Dialog Box with Databank Selected	3-13
3-19	Script Tree with a Databank	3-14
3-20	Substitute Variable Window	3-14
3-21	Script Parameter with a Mapped Databank Variable.....	3-15
3-22	Script Parameters with Multiple Mapped Databank Variables	3-15
3-23	Script Tree with a Text Matching Test Node	3-16
3-24	Iterations Dialog Box	3-16
3-25	Results Report for Multiple Iterations	3-17
3-26	Details View Showing the HTTP Test Comparison Tab	3-18
4-1	Oracle Functional Testing for Web Applications Main Window	4-2
4-2	Oracle Functional Testing Visual Script Tree	4-4
4-3	HTML Properties Dialog Box.....	4-5
4-4	Visual Script Frame Node.....	4-5
4-5	Script Options Dialog Box	4-6
4-6	Content Tests Manager Dialog Box.....	4-7
4-7	Images Script Node.....	4-7
4-8	Resource Validation Window	4-8
4-9	Results Report Window	4-9
4-10	Playback Results Pane.....	4-10
4-11	Visual Script with Failure Markers	4-10
4-12	Different Html Script Node	4-11
4-13	Ignore This Failure Script Node Marker.....	4-11
4-14	New Links Script Node.....	4-11
4-15	New Links Script Node with Marker.....	4-11
4-16	Different Html and Different Script Nodes.....	4-12
4-17	Master and Tested Page HTML Source Differences	4-12
4-18	Master and Tested Image Differences.....	4-13
4-19	Master and Tested Image Differences.....	4-13
4-20	Text Matching Test Properties Dialog Box.....	4-15
4-21	Text Matching Test Script Node	4-15
4-22	Server Response Test Properties Dialog Box	4-16
4-23	Server Response Test Script Node.....	4-16
4-24	Form Element Script Nodes	4-17
4-25	Form Element Test Properties Dialog Box	4-17
4-26	Form Element Test Script Node.....	4-18
4-27	Table Test Wizard Welcome Dialog Box	4-18
4-28	Select Table Element Dialog Box	4-18
4-29	Table Element Selected Example	4-19
4-30	Create Table Test Dialog Box	4-19
4-31	Create Table Test Dialog Box with Field Selected.....	4-20
4-32	Table Test Script Node	4-20
4-33	Parameters Script Node	4-21
4-34	Data Bank Wizard Variables Tab.....	4-22
4-35	Add Variable Dialog Box.....	4-22
4-36	Data Bank Wizard with a Variable Added	4-23
4-37	Data Bank Wizard Data Binding Tab.....	4-24
4-38	Select Data Bank Dialog Box	4-24

4-39	Data Bank Wizard with a Data Bank File	4-25
4-40	Data Bank Wizard with Bindings	4-26
4-41	Data Bank Wizard Fields and Records View	4-26
4-42	Data Bank Parameters Script Nodes	4-27
4-43	Iterations Dialog Box	4-28
4-44	Parameters Script Nodes in the Script	4-30
4-45	Data Bank Wizard Variables Tab.....	4-30
4-46	Data Bank Wizard with Auto-mapped Parameters.....	4-31
4-47	Data Bank Wizard Data Binding Tab.....	4-32
4-48	Select Data Bank Dialog Box	4-32
4-49	Data Bank Wizard with Bound Parameters.....	4-33
4-50	How Scripts Map and Bind to Data Bank Records	4-33
4-51	Data Bank Wizard Fields and Records View	4-34
4-52	Script Nodes with Mapped and Bound Data Bank Variables.....	4-34
4-53	Iterations Dialog Box	4-35
5-1	Job Scheduler Wizard Workspace and Scripts Dialog Box.....	5-2
5-2	Job Scheduler Wizard Log Window Dialog Box	5-2
5-3	Job Scheduler Wizard Email Dialog Box	5-3
5-4	Job Scheduler Wizard Mail Server Configuration Dialog Box.....	5-3
5-5	Job Scheduler Wizard Edit Schedule Dialog Box.....	5-4
5-6	Job Scheduler Current Schedule Window.....	5-5
5-7	Job Editor Window	5-6
5-8	Job Editor Script Selection Window	5-6
5-9	Job Editor Email Window	5-7
5-10	Job Scheduler Current Schedule Window.....	5-8
5-11	Edit Schedule Dialog Box.....	5-8
5-12	Job Scheduler Current Schedule Window.....	5-9
6-1	Oracle Load Testing for Web Applications Main Window	6-2
6-2	Configure Parameters Pane.....	6-3
6-3	Autopilot Window	6-4
6-4	Virtual User Status Grid Window	6-5
6-5	Add Configuration Dialog Box.....	6-6
6-6	Add Monitors Step 1 Dialog Box.....	6-6
6-7	Add Monitors Step 1 with Data Sources Expanded	6-7
6-8	Add Monitors Step 2 Dialog Box.....	6-7
6-9	Add Monitors Step 3 Dialog Box.....	6-8
6-10	Add Monitors Step 3 with Processors Listed.....	6-8
6-11	Add Monitors Step 3 with Processors Selected	6-9
6-12	Add Monitors Step 3 with Processors and Memory Selected	6-10
6-13	Configurations Dialog Box with Defined Monitors.....	6-11
6-14	Test Monitors Dialog Box	6-11
6-15	Edit Monitor Dialog Box	6-12
6-16	Add Monitored System Dialog Box	6-12
6-17	Session Start/Stop Options Dialog Box.....	6-14
6-18	Scenario Defaults Options Dialog Box.....	6-15
6-19	Set Up Autopilot Options	6-16
6-20	Virtual User Grid	6-17
6-21	View Run Graphs Tab.....	6-17
6-22	Performance Statistics Report	6-18
6-23	Performance Vs. Users Report	6-19
6-24	Users Vs. Time Report.....	6-20
6-25	Performance Vs. Time Report	6-20
6-26	Statistics Vs. Time Report	6-21
6-27	Virtual User Shortcut Menu	6-21
6-28	Modify Run Attributes Dialog Box	6-22

6-29	Virtual User Display Window	6-22
6-30	Create Reports Tab Filters	6-24
6-31	Sample Session Report Graph.....	6-25
6-32	Sample Excel Report Graph.....	6-26
6-33	Sample Excel Data Table Report.....	6-27
6-34	Sample Session Performance Report.....	6-28
6-35	Edit User Defined Profiles Dialog Box.....	6-29
6-36	Script Added to Run Section of User Defined Profile	6-30
6-37	Sync Point Added to Run Section of User Defined Profile.....	6-31
6-38	Edit User Defined Profile Dialog Box	6-32
6-39	Select Scripts and User-Defined Profiles Pane.....	6-32
6-40	User Defined Profile Manager Dialog Box.....	6-33
6-41	Edit User Defined Profile Dialog Box	6-34
7-1	Open Project Dialog Box	7-2
7-2	Oracle Test Manager for Web Applications Main Window	7-3
7-3	Requirements Tab	7-4
7-4	Add Requirement Window	7-5
7-5	Add Requirement Window with Sample Data	7-6
7-6	Requirements Tab with New Requirement Added	7-7
7-7	Add Test Window.....	7-8
7-8	Add Test Window with Sample Data	7-9
7-9	Manual Test Steps Window.....	7-10
7-10	Manual Test Steps Window with Sample Data	7-11
7-11	Add Test Window with Sample Automated Test.....	7-13
7-12	Tests Tab with New Test Added	7-14
7-13	Associate Requirements Window	7-15
7-14	Test Associated with Requirement.....	7-16
7-15	Run Test Info Window	7-16
7-16	Run Manual Test Window.....	7-17
7-17	Run Manual Test Summary Window	7-17
7-18	Results Report Window	7-18
7-19	Find Window	7-19
7-20	Find Window with Search Results	7-20
7-21	Add Issue Window	7-21
7-22	Add Issue Window with Sample Data	7-22
7-23	Issues Tab with New Issue Added.....	7-23
7-24	Associate Test Window with Issue Selected	7-24
7-25	Attach Files Window with File Selected	7-25
7-26	Issue with File Attachments	7-25
7-27	Reports Tab with Bar Graph Report	7-26
7-28	Add Report Window	7-27
7-29	Add Report Window with Selected Fields.....	7-28
7-30	Add Report Filters Window	7-28
7-31	Reports Tab with Custom Reports	7-29

Preface

Welcome to Getting Started with Oracle Application Testing Suite. This guide explains how to get started using the features and options of Oracle OpenScript, Job Scheduler, and Oracle Load Testing for Web Applications for testing Web pages or applications.

Audience

This guide is for Web test engineers who will be using the Oracle Application Testing Suite applications for regression testing, performance testing (load and scalability), and monitoring of a Web site or application.

Prerequisites

The tutorials in this guide assume an understanding of software or Web application testing concepts. Test engineers using the Oracle Application Testing Suite should be familiar with the concepts of regression testing, load testing, scalability testing, and operational monitoring.

Using This Guide

This guide is organized as follows:

[Chapter 1, "Introduction"](#) provides an overview of the major features of the tools included in the Oracle Application Testing Suite.

[Chapter 2, "Oracle Application Testing Suite Basics"](#) provides descriptions of the products in the Oracle Application Testing Suite and the main features of each.

[Chapter 3, "Oracle OpenScript Tutorial"](#) provides step-by-step instructions and explanations for building regression test scripts for testing Web pages or applications with Oracle OpenScript. The tutorial includes examples that highlight the script features, the Databanks, and test cases.

[Chapter 4, "Oracle Functional Testing for Web Applications Tutorial"](#) provides step-by-step instructions and explanations for building regression test scripts for testing Web pages or applications with Oracle Functional Testing for Web Applications. The tutorial includes examples that highlight the Visual Script features, the Data Bank Wizard, and test cases.

[Chapter 5, "Job Scheduler Tutorial"](#) provides step-by-step instructions for creating Job Scheduler jobs and schedules to play back multiple Oracle Oracle Functional Testing for Web Applications Visual Scripts for operational and performance monitoring of a site.

[Chapter 6, "Oracle Load Testing for Web Applications Tutorial"](#) provides step-by-step instruction for using multiple Oracle Functional Testing for Web Applications Visual

Scripts or OpenScript scripts to perform load and scalability testing of Web applications and back end systems. This chapter also explains how to configure ServerStats and generate reports from testing data.

[Chapter 7, "Oracle Test Manager for Web Applications Tutorial"](#) provides step-by-step instruction for using the main features of Oracle Test Manager for Web Applications.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at <http://www.oracle.com/accessibility/>.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

Related Documents

For more information, see the following documents in the Oracle Application Testing Suite documentation set:

- *Oracle Application Testing Suite Release Notes*
- *Oracle Application Testing Suite OpenScript User's Guide*
- *Oracle Functional Testing for Web Applications Functional Testing User's Guide*
- *Oracle Functional Testing for Web Applications Job Scheduler User's Guide*
- *Oracle Functional Testing for Web Applications Navigation Editor User's Guide*
- *Oracle Functional Testing for Web Applications Application Programming Interface Reference*
- *Oracle Functional Testing for Web Applications Result Objects Reference*
- *Oracle Functional Testing for Web Applications Settings Manager Reference*
- *Oracle Load Testing for Web Applications Load Testing User's Guide*

- *Oracle Load Testing for Web Applications Load Testing ServerStats Guide*
- *Oracle Test Manager for Web Applications Test Manager User's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
<code>monospace</code>	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Introduction

Oracle Application Testing Suite is an integrated, comprehensive Web application testing solution that provides all the tools you need to ensure the scalability and reliability of your business-critical applications.

- Oracle OpenScript is an updated scripting platform for creating automated extensible test scripts in Java.
- Oracle Functional Testing for Web Applications for automated functional and regression testing
- Job Scheduler for scheduling functional and regression testing
- Oracle Load Testing for Web Applications for load, scalability and stress testing. Oracle Load Testing for Web Applications also includes tools for server side monitoring and reporting
- Oracle Test Manager for Web Applications for organizing and managing your overall testing process.

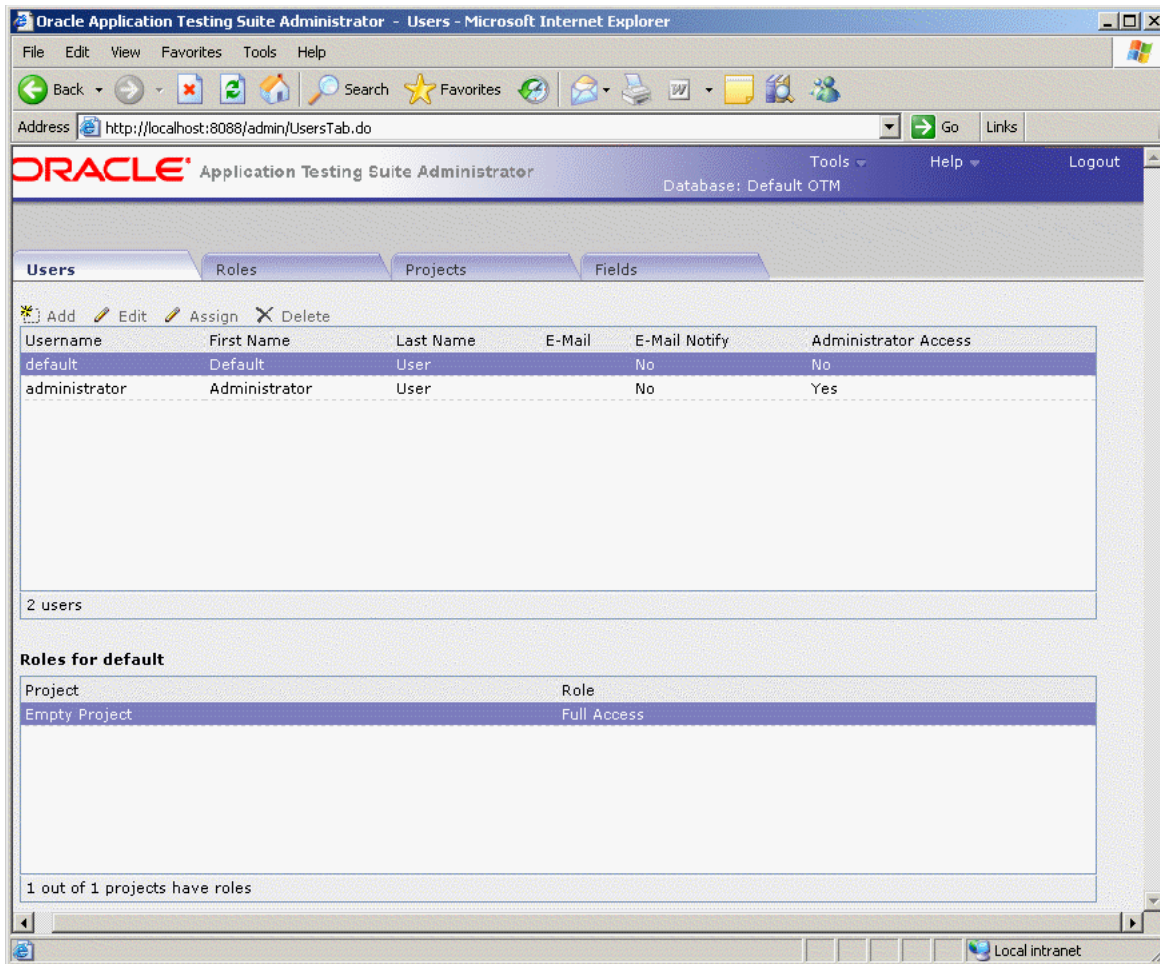
As your application changes, any differences in your tests are highlighted in the Visual Scripts, and can be automatically updated in-place. That means that your regression, and load tests will always stay synchronized with your application, and you can make automated testing a routine part of your Web development process.

This manual introduces you to the Oracle Application Testing Suite and provides step-by-step tutorials to help you get started using the tools.

1.1 About Oracle Application Testing Suite Administrator

The Administrator allows the Oracle Application Testing Suite system administrator to manage user accounts for Oracle Load Testing for Web Applications and Oracle Test Manager for Web Applications. For Oracle Load Testing for Web Applications, you define user accounts and the type of access allowed. For Oracle Test Manager for Web Applications, you define user accounts, roles, projects, and fields.

Figure 1–1 Administrator Main Window (Test Manager Database)



1.1.1 Administrator Feature Highlights

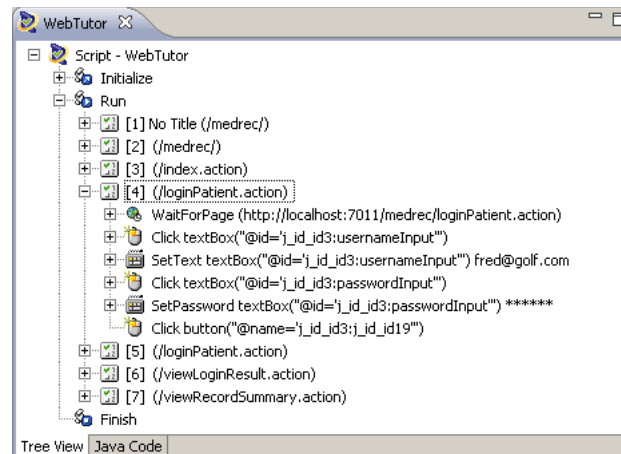
Oracle Application Testing Suite Administrator provides the following features:

- **User Accounts** - user accounts can be defined and customized for either Oracle Load Testing for Web Applications or Oracle Test Manager for Web Applications.
- **Roles** - roles with access permissions for read, write, delete, and execute can be defined and customized for Oracle Test Manager for Web Applications.
- **Projects** - projects can be defined and customized and assigned to specific user accounts for Oracle Test Manager for Web Applications.
- **Fields** - fields can be defined and customized for requirements, issues, tests, and test runs for Oracle Test Manager for Web Applications.

1.2 About Oracle OpenScript

Oracle OpenScript is built on a standards-based platform and provides the foundation for OpenScript Modules and Application Programming Interfaces (APIs). Combining an intuitive graphical interface with the robust Java language, OpenScript serves needs ranging from novice testers to advanced QA automation experts.

Figure 1–2 OpenScript Tree View Hierarchy



OpenScript APIs are used to build scripts for testing Web applications. The OpenScript API consists of a set of procedures that can be used to customize the scripts within the development environment. The API can also be used by advanced technical users to enhance scripts for unique testing needs

1.2.1 OpenScript Feature Highlights

OpenScript is the next generation environment for developing Oracle Application Testing Suite scripts for Web application testing. OpenScript offers the following advantages for Web-based application testing:

- Scripting Workbench** - OpenScript provides an Eclipse -based scripting Workbench where you can create and run your automated test scripts. Users can use the Tree View graphical scripting interface for creating and editing scripts through the UI. Users can also switch to the Java Code View programming interface and leverage the integrated Eclipse IDE for creating and editing their scripts programmatically.

Functional test scripts created in OpenScript can be played back to test and validate application functionality. Load test scripts created in OpenScript will run in Oracle Load Testing for Web Applications for application load testing, allowing users to simulate hundreds or thousands of users executing scripts at the same time.

- Test Modules** - The OpenScript Test Modules provide application-specific test automation capabilities. Each Test Module is custom built to test a specific application or protocol. OpenScript includes several functional and load testing modules for testing Web-based applications. Additional modules can be developed for the OpenScript platform.

OpenScript's Test Module interface is completely open and extendable by end-users. Users can leverage the Test Module API to build their own modules for testing specific applications or can extend an existing module to add custom functionality.

- Graphical/Tree View Scripting Interface** - The OpenScript Tree View scripting interface provides a graphical representation of the test script. Multiple script windows can actually be open at the same time. Within each script window, the Tree View is broken down into 3 main script sections:
 - Initialize**: For script commands that only execute once on the first iteration

- Run: Main body of the script for commands that will run on every iteration
- Finish: For script commands that only execute once on the last iteration

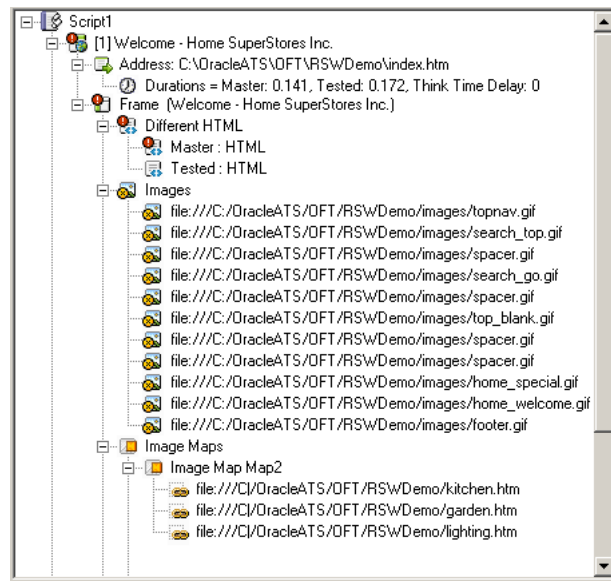
Within each section, script Steps and Navigation nodes can be created automatically during script recording or manually through the Tree View user interface. Additional script commands will also be represented as nodes in Tree View including test cases, data inputs, log messages, etc. Each Tree View node has a corresponding representation in the Java Code View.

- **Programming/Code View Scripting Interface** - The OpenScript Java Code View scripting interface provides a Java representation of the test script. This view provides full access to Eclipse IDE for creating, editing & debugging script code. Script commands in Java are mapped to a corresponding representation in the Tree View. Users can edit their script in either the code or tree view and changes will be automatically reflected in both views.
- **Properties View & Results View** - The OpenScript Properties View allows users to view detailed properties for selected script nodes in the Tree View. The Results View shows detailed step-by-step results of script playback which are linked to the OpenScript display window.
- **Data Banking** - OpenScript allows users to parameterize script data inputs to perform data driven testing using Data Banking. Users can select any data inputs for their script and then substitute a variable to drive the input from an external file during playback. Multiple Data Bank files can be attached to a single script and users can specify how OpenScript assigns data during script playback.
- **Correlation** - The OpenScript Correlation interface allows users to create correlation libraries to automatically parameterize dynamic requests during playback. Correlation libraries contain rules for automatically handling dynamic request parameters such as urls, query strings and post data for the load testing modules.
- **OpenScript Preferences** - The OpenScript Preferences interface is where users specify settings to control script recording, script playback, correlation and general preferences for the OpenScript Workbench.
- **Multi-User Execution** - launch more than one OpenScript instance under separate named Windows user accounts. Playback for multiple scripts is supported using any of the following:
 - OpenScript Playback button
 - Command-Line Interface
 - Oracle Load Testing for Web Applications
 - Oracle Test Manager for Web Applications

1.3 About Oracle Functional Testing for Web Applications

Oracle Functional Testing for Web Applications is used for functional/regression testing and serves as a script recorder for the Oracle Application Testing Suite. Oracle Functional Testing for Web Applications records all of the objects on every page that you visit and automatically inserts tests to validate the objects. The components of each page are represented graphically in the Visual Script and can be masked or augmented using simple point and click actions.

Figure 1-3 Visual Script Tree Hierachy



Oracle Functional Testing for Web Applications lets you easily create, maintain, and execute regression testing scripts for your Web applications. Oracle Functional Testing for Web Applications features a powerful, intuitive visual script, an automated test case generator, a specialized text matching component, and the ability to execute data-driven tests using the Data Bank Wizard.

1.3.1 Oracle Functional Testing for Web Applications Feature Highlights

Oracle Functional Testing for Web Applications offers the following advantages for Web-based application testing:

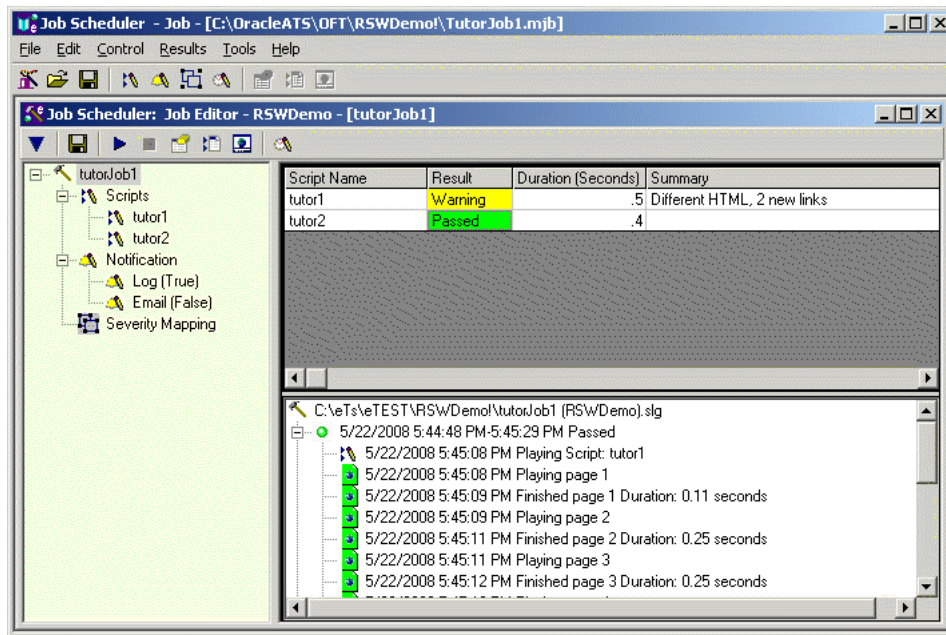
- **Visual Script Technology/Automatic Test Generation** - you can record and test your entire application in minutes with reusable, object-oriented Visual Scripts. Recorded Visual Scripts automatically capture and test Anchors, Elements, Forms, Frames, HTML, Images, Image Maps, Links, ActiveX controls, Java Applets, VBScript, and JavaScript. Visual Scripts require no programming.
- **Graphical Test Results and Simple Script Updating** - test failures and HTML differences are indicated by red flags annotated within the Visual Scripts for rapid diagnoses of application errors. Visual Scripts can be updated to reflect changes to the application with the click of a button.
- **Data Bank Wizard** - create data-driven tests without programming. A single Visual Script can be used over and over with varying input and response data using values from an external data source.
- **Visual Test Case Insertion** - additional test cases can be added to Visual Scripts to verify server response times, form elements, and the presence or absence of specific text in a page.
- **Programming Interface** - full flexibility and extensibility to match your Web testing needs. Oracle Functional Testing for Web Applications provides levels of testing extensibility from the simplicity of Visual Scripts to your own fully-customized external application that controls Oracle Functional Testing for Web Applications. Basic Visual Script capabilities can be extended using Oracle Functional Testing for Web Applications built-in test cases or your own custom Test Scriptlets.

- **HTTPs and SSL Support** - supports all popular protocols as well as certificates.
- **Built-in Application Server Support** - automatically manages session variables for Net Dynamics, Broadvision, WebObjects, ColdFusion, and Microsoft ASP platforms.
- **High Throughput Resource Validation** - automatically collects and verifies all referenced Web resources that include links and images.
- **Test Case Librarian** - allows you to create and store re-usable test cases for use across multiple test scripts.

1.4 About Job Scheduler

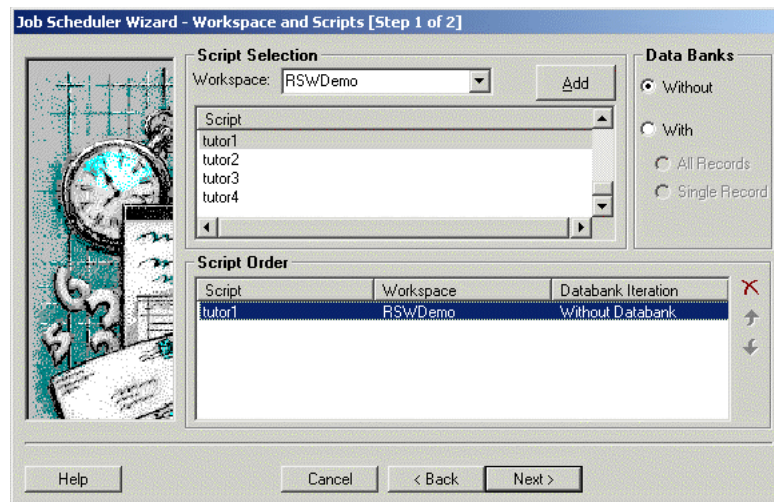
Job Scheduler is a test management tool that lets you group and run multiple Oracle Functional Testing for Web Applications Visual Scripts in sequence as a single job. Job Scheduler jobs can be scheduled to run automatically at specific times or be run manually at any time.

Figure 1–4 Job Scheduler Main Window



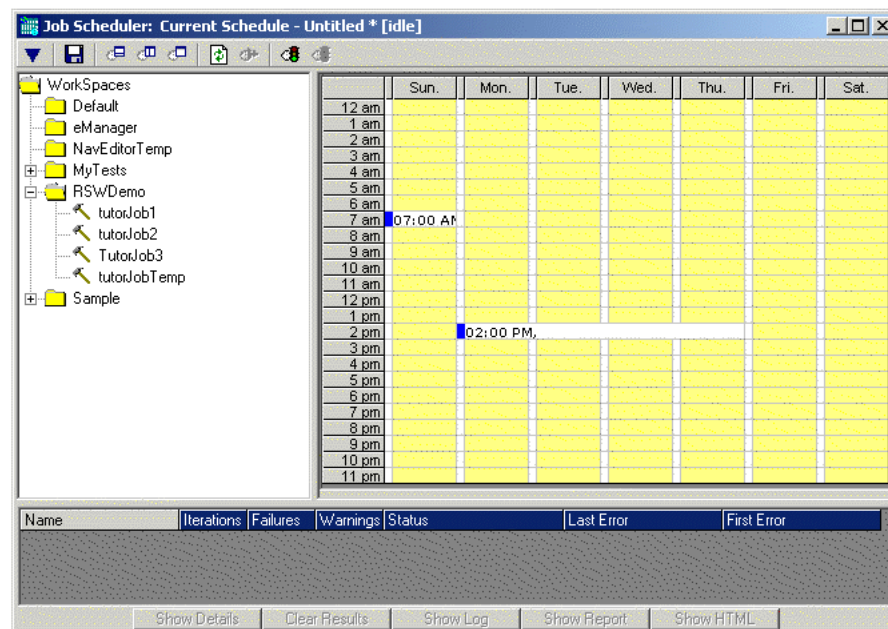
The Job Scheduler Wizard provides a convenient way to build Job Scheduler jobs, which can then be included on any schedule. The Job Scheduler Wizard includes steps for selecting Visual Scripts and setting notification options.

Figure 1–5 Job Scheduler Wizard Window



The Job Scheduler Schedule lets you specify when to start a job.

Figure 1–6 Job Scheduler Schedule Window



1.4.1 Job Scheduler Feature Highlights

Job Scheduler offers the following advantages for Web-based application testing:

Multiple Oracle OpenScript Visual Scripts - play back a series of Oracle Functional Testing for Web Applications Visual Scripts as a single job. Jobs can be run immediately or scheduled to run on a specific set of days and times.

Schedule Window - lets you schedule multiple jobs to run on specific days and times.

Job Scheduler Wizard - guides you through creating jobs with Visual Scripts created earlier with Oracle Functional Testing for Web Applications. The wizard provides options for customizing error notifications and e-mail recipients for playback results.

Integrated HTML Viewer - view pages in real time as Job Scheduler plays back Visual Scripts. The HTML viewer shows page content and provides visual indications of pages with failures.

Job Notification Messages - specify customized error notification messages using the Job Scheduler Wizard. The messages appear in the results log.

Job Notification e-Mail - send job results via e-mail to one or more recipients using MAPI or SMTP e-mail.

HTML Format Job Results Reports - playback results reports are saved to an HTML page for later review and analysis.

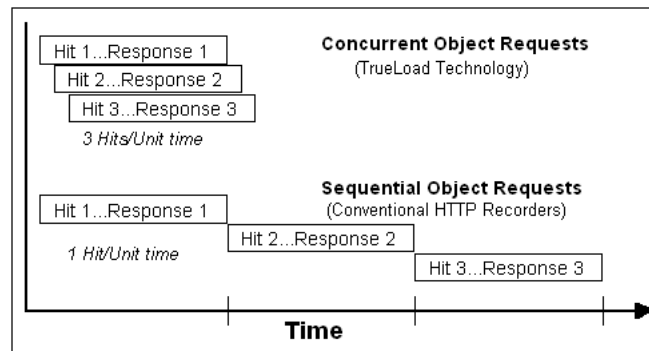
1.5 About Oracle Load Testing for Web Applications

Oracle Load Testing for Web Applications provides an easy and accurate way to test the scalability of your e-Business applications. Oracle Load Testing for Web Applications emulates thousands of virtual users accessing your site simultaneously, and measures the effect of the load on application performance.

Figure 1–7 Oracle Load Testing for Web Applications Virtual Users View

VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Page	System	Data Bank	Current Error	Previous Error
1	tutor3	Running	9		24.876	[2] Search - Home SuperStore Inc.	localhost	Record 5:Phones		
2	tutor1	Think time delay	49		9.463	[2] Kitchens - Home SuperStores Inc.	localhost			
3	tutor1	Running	48		8.892	[4] Electronics - Home SuperStores Inc.	localhost			
4	tutor3	Iteration delay	5		31.816	[3] Results - Home SuperStore Inc.	localhost	Record 4:Cabinets		
5	tutor1	Think time delay	28		9.594	[2] Kitchens - Home SuperStores Inc.	localhost			
6	tutor1	Iteration delay	29		9.053	[4] Electronics - Home SuperStores Inc.	localhost			
7	tutor3	Starting					localhost			
8	tutor1	Think time delay	2		8.513	[2] Kitchens - Home SuperStores Inc.	localhost			
9	tutor1	Think time delay	2		7.37	[2] Kitchens - Home SuperStores Inc.	localhost			

Oracle Application Testing Suite TrueLoad Technology ensures that your tests will closely correlate with real user-load so you can confidently use results from Oracle Load Testing for Web Applications to help make key decisions about your system's architecture, tuning, and hosting alternatives.

Figure 1–8 Concurrent Object Requests vs. Sequential Object Requests

1.5.1 Oracle Load Testing for Web Applications Feature Highlights

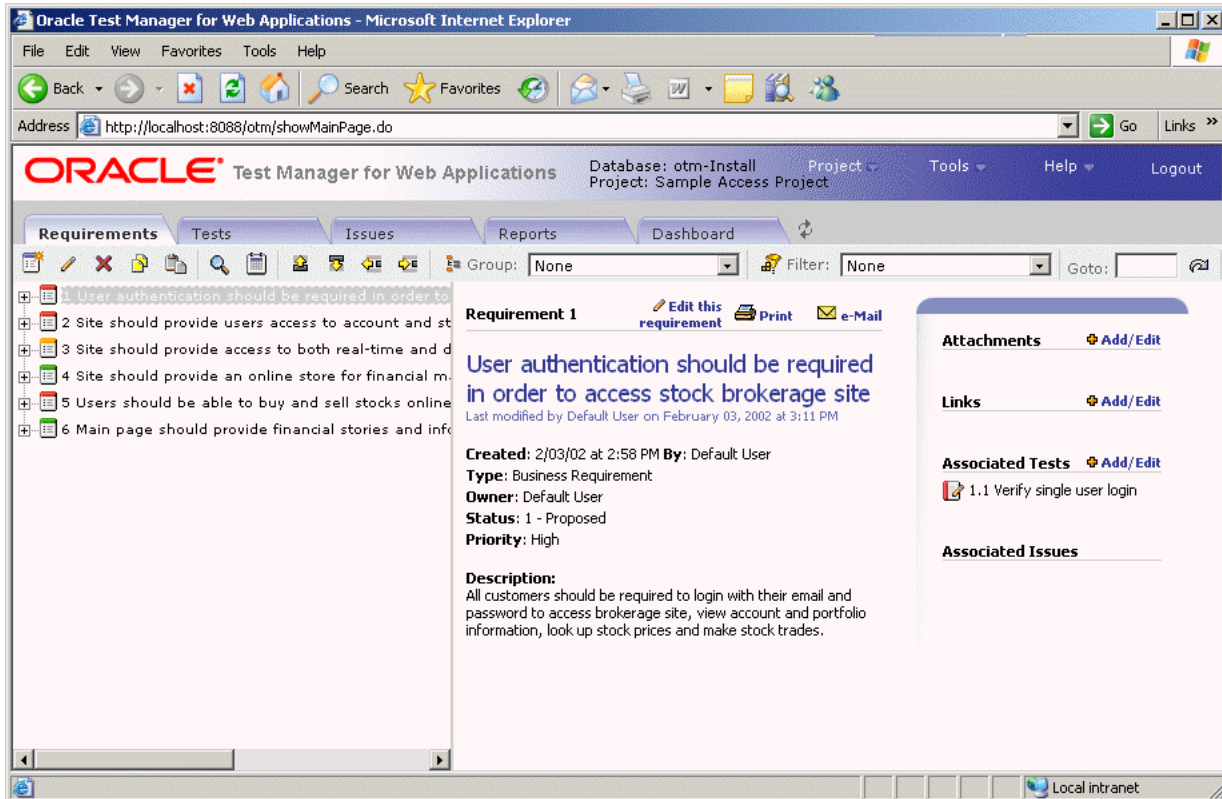
Oracle Load Testing for Web Applications offers the following advantages for Web-based application load testing:

- **Trueload Technology** - accurately emulates multi-threaded browser requests and automatically validates server responses for test results that closely correlate with real user testing.
- **Reusable Scripts** - uses the scripts created with Oracle OpenScript or Oracle Functional Testing for Web Applications to emulate hundreds or thousands of virtual users.
- **Interactive What-If Analysis and Virtual User Display** - you can change the number and type of user on-the-fly to try "what-if" scenarios as you vary the loading conditions or application settings. You can even view the actual pages seen by virtual users to aid in debugging.
- **Real-Time Graphs and Reports** - you can view real-time reports and graphs that include response time, error rates, number of users, and statistics such as hits per second, pages per second, etc.
- **Single Point of Control with Distributed Agents** - virtual users can be simulated by a single server or distributed amongst multiple servers located anywhere on a LAN or WAN.
- **Scenario Manager and Autopilot** - define any number of custom load scenarios by simply pointing and clicking on the names of the pre-recorded scripts and then specifying how many virtual users of each type you wish to run, and how you would like them to ramp up.
- **Post-run Analysis** - performance data can be accumulated at varying levels of granularity including profiles, scripts, groups of pages, individual pages, and objects on pages. Oracle Load Testing for Web Applications provides a comprehensive set of graphs and reports, and can also export data to external programs such as Microsoft Excel for further analysis.
- **Server-side monitoring with ServerStats** - server performance can be monitored for a variety of server-side application, database, system, and Web server statistics. You can configure ServerStats to display real-time performance statistics for the various hosts and services available from the server such as, percentage of CPU usage, memory usage, Web server statistics, etc.

1.6 About Oracle Test Manager for Web Applications

Oracle Test Manager for Web Applications is an easy to use tool that allows you to organize and manage your overall testing process. It provides a single unified platform for sharing information among team members.

Figure 1–9 Oracle Test Manager for Web Applications Main Window



Oracle Test Manager for Web Applications lets you create projects that group together and organize test scripts, requirements that need to be tested, and issues resulting from the tests. Once created, you can indicate the relationships among these items, allowing you to quickly and easily find all information pertaining to a particular test script, requirement, or issue.

1.6.1 Oracle Test Manager for Web Applications Feature Highlights

Oracle Test Manager for Web Applications offers the following features and advantages for integrated requirements management and defect tracking for both manual and automated tests:

Requirements Management - provides the ability to define and manage requirements for a specific project. You can specify details for each requirement, track the status of each requirement, and associate requirements with test cases to ensure testing coverage.

Test Planning and Management - provides the ability to define and manage a test plan that incorporates both manual and automated test cases. You can store Oracle Application Testing Suite scripts in the database, automatically execute scripts in Oracle OpenScript from the test plan interface, and automatically store the test results. You can also associate requirements to test cases to ensure testing coverage, and

associate test cases with issues so they can be reproduced and to keep track of how the issues were identified.

Defect Tracking - provides the ability to create and manage defects, referred to as issues, for a specific project. You can associate test cases with issues so they can be reproduced and to keep track of how the issues were identified.

Integration with Oracle Application Testing Suite - seamlessly integrates with Oracle Application Testing Suite test solutions, providing the ability to automatically launch and execute Oracle OpenScript or Oracle Functional Testing for Web Applications scripts for functional and regression testing as well as retrieve and archive the results. You can also launch third party products.

Reporting - generates reports in standard HTML format for managing the overall testing process. You can report on requirements, tests, and issues.

Administration - provides an administration tool for entering and managing user accounts, project permissions, and general tool preferences.

Custom Fields - provides the ability to add custom fields to the database for recording data specific to your projects.

Database Repository - provides the ability to store test assets including test scripts, results, attachments, requirements, test plans, and defects in a common database.

1.7 Oracle Application Testing Suite Database Configuration

The Oracle Application Testing Suite Database Configuration utility lets you add database connections for Oracle Load Testing for Web Applications and Oracle Test Manager for Web Applications. The Oracle Application Testing Suite installation includes a WebLogic web server and the Oracle 10g Express Edition Database by default. The Oracle Application Testing Suite Database Configuration utility can be used set the current database or to connect to databases other than the default.

1.7.1 Database Configuration Feature Highlights

The Database configuration utility provide the following features:

- **Add, Update, Delete Database Connections** - provides a convenient way to manage database connections for Oracle Load Testing for Web Applications and Oracle Test Manager for Web Applications.
- **Create Schemas and Tables** - automatically uses existing schemas or creates schemas and tables for additional databases.

Oracle Application Testing Suite Basics

This chapter explains how to get started using Oracle Application Testing Suite. It explains how to install and start the applications, and the features of the main windows.

2.1 Installing and Starting Oracle Application Testing Suite

To install Oracle Application Testing Suite:

1. Go to:
<http://www.oracle.com/technology/software/products/app-testing/index.html>.
2. Download the Oracle Application Testing Suite product from the Oracle Web site and save it to a temporary directory on your hard disk.
3. Unzip the download file and then run `oats###.exe` to install Oracle Application Testing Suite.
4. Run `openScript###.exe` to install Oracle OpenScript.
5. Follow the setup instructions to install the Oracle Application Testing Suite and OpenScript. The Oracle Application Testing Suite install program will install the WebLogic web server and the Oracle 10g Express Edition Database by default.

During the Oracle Application Testing Suite installation, you will be required to enter a default password to be used with Oracle Application Testing Suite products. *Remember this password.* It will be required to log in to the Administrator, Oracle Load Testing for Web Applications, and Oracle Test Manager for Web Applications.

6. Select applications from the **Oracle Application Testing Suite** start menu to start the user interface for specific products.

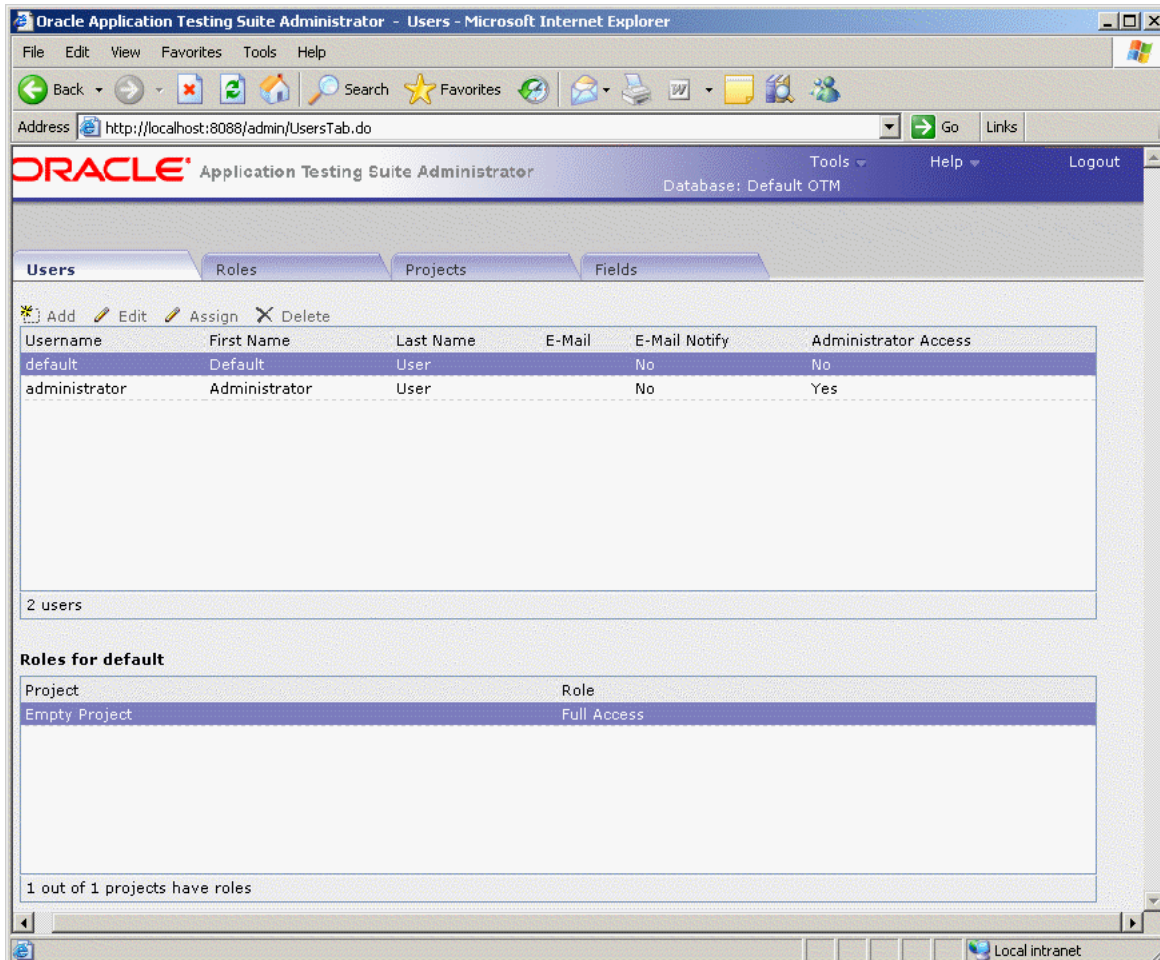
The installation creates a default Administrator user name in the Oracle Application Testing Suite database. The first time you log into the Administrator, Oracle Load Testing for Web Applications, or Oracle Test Manager for Web Applications, enter the username **Administrator** and the password you defined during the installation. You can use the Oracle Application Testing Suite Administrator to change the default users and customize the usernames and passwords for Oracle Application Testing Suite users.

2.2 Oracle Application Testing Suite Administrator Main Window Features

The Oracle Application Testing Suite Administrator is where you customize user access for Oracle Load Testing for Web Applications and Oracle Test Manager for Web

Applications. For Oracle Test Manager for Web Applications, you can also customize roles, projects, and fields.

Figure 2-1 Oracle Application Testing Suite Administrator Main Window

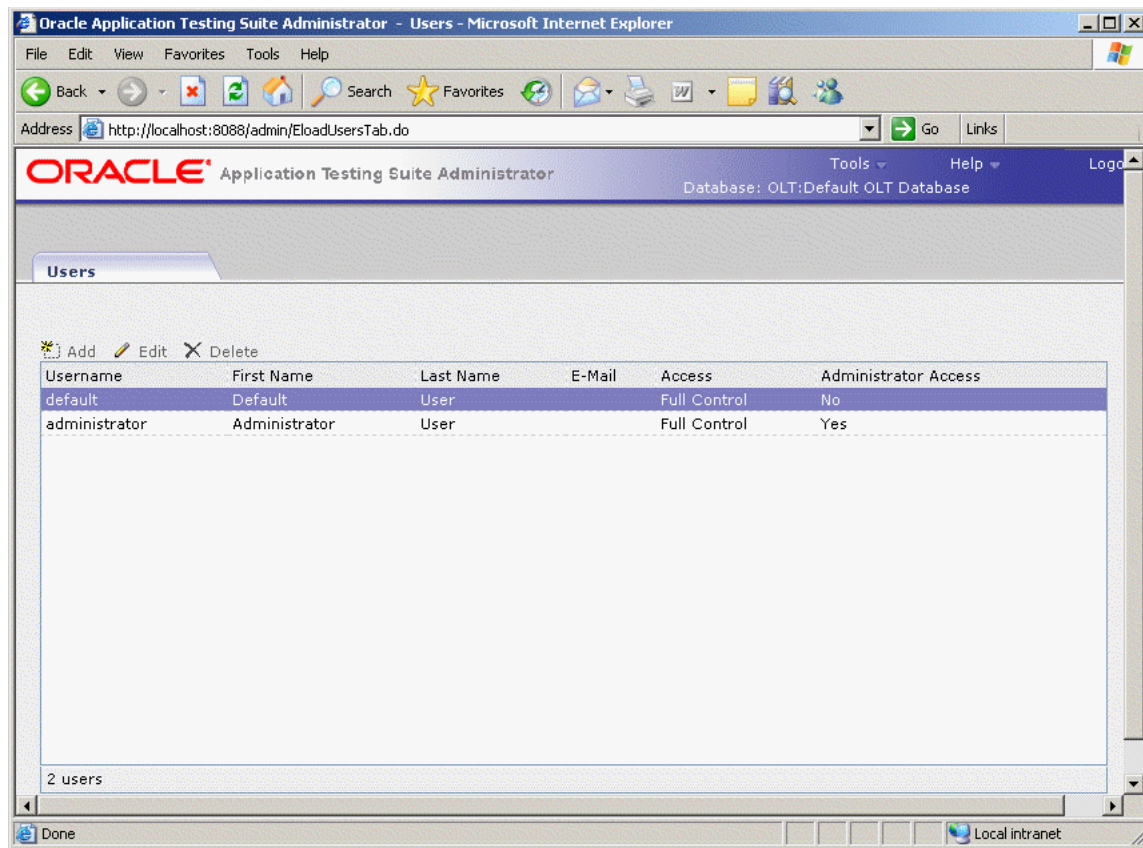


When you log into the Administrator and select an Oracle Load Testing for Web Applications database, only the **Users** tab appears. When you log into the Administrator and select an Oracle Test Manager for Web Applications database, the **Users**, **Roles**, **Projects**, and **Fields** tabs appear.

2.2.1 Users Tab

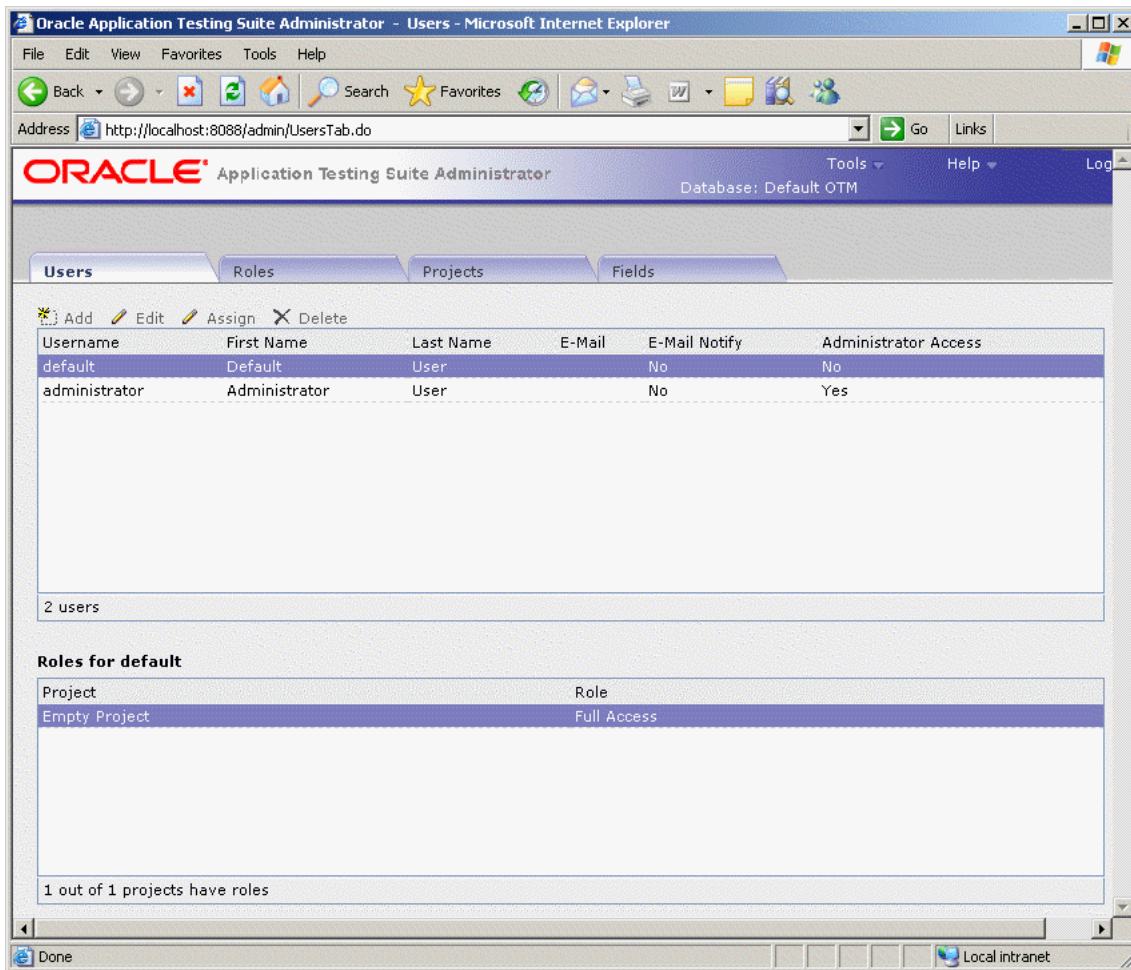
The Users tab is where you add, delete, and configure users for Oracle Load Testing for Web Applications and Oracle Test Manager for Applications.

For Oracle Load Testing for Web Applications databases, the Users tab lets you add, edit, and delete users.

Figure 2-2 Users Tab for Oracle Load Testing for Web Applications Users

For Oracle Test Manager for Web Applications databases, the Users tab lets you add, edit, and delete users, and assign specific projects to users.

Figure 2-3 Users Tab for Oracle Test Manager for Web Applications Users



The Users tabs can have the following options:

Add - displays the Add User dialog box for adding a new user.

Edit - displays the Edit User dialog box for the selected user. You can change the user's name, username, or password.

Assign - displays the Edit Role dialog box for assigning roles to the selected user for the selected projects.

Delete - deletes the selected user.

Username - displays the name the user will use to log on. Click the column header to sort users in ascending order by this field.

First Name - displays the user's first name. Click the column header to sort users in ascending order by this field.

Last Name - displays the user's last name. Click the column header to sort users in ascending order by this field.

E-mail - displays the user's email address. When **Enable E-mail notification** is selected for the user, the user will receive email notifications when items are created, or when the owner or assigned to field is changed for issues. Click the column header to sort users in ascending order by this field.

E-mail Notify - indicates whether the user will receive email notifications. Click the column header to sort users in ascending order by this field.

Administrator Access - indicates whether the user can access the Oracle Test Manager for Web Applications Administrator.

Active - this column is only displayed when named user licenses are being used. Indicates whether the user is active, that is, allowed to log in using a named user license.

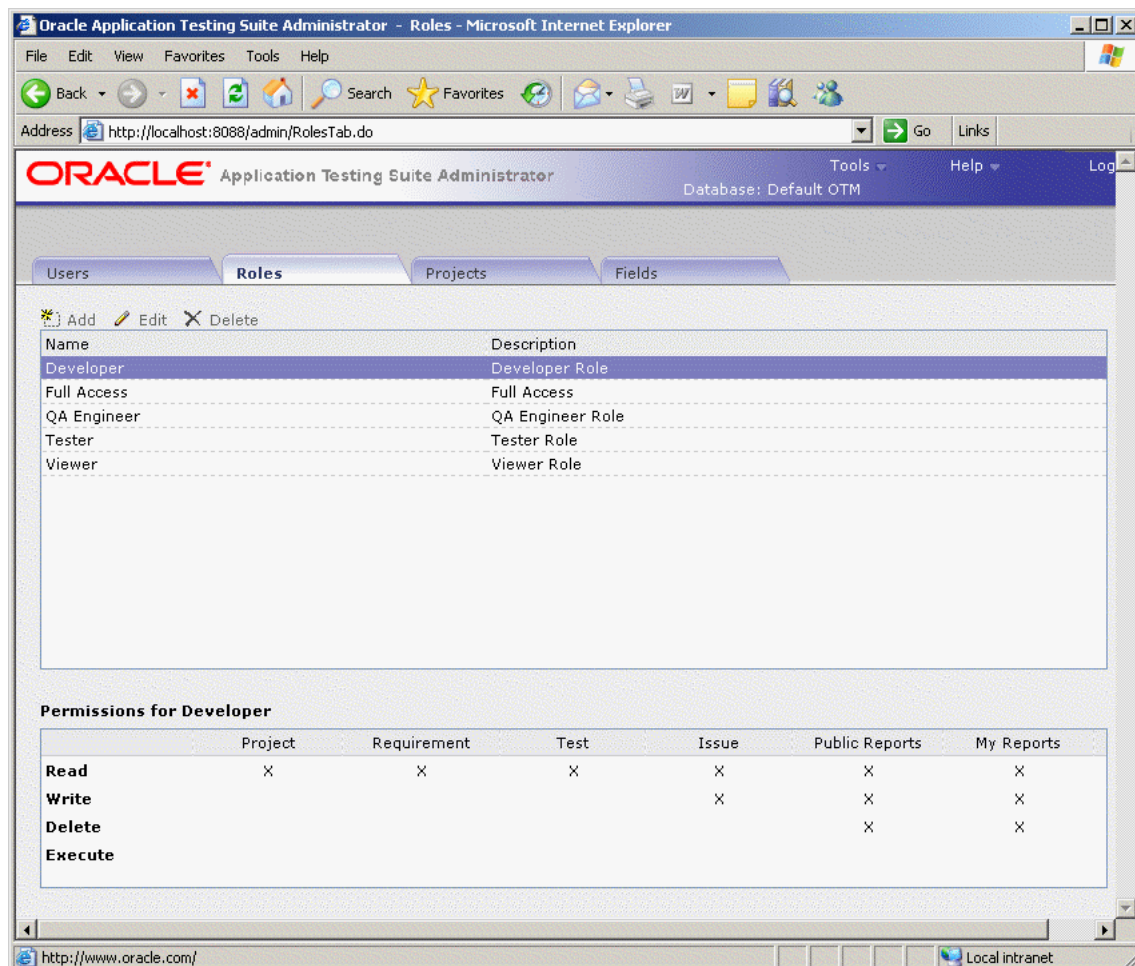
Roles for <user> - displays the roles assigned to the selected user for each project in the database.

- **Project** - displays a list of the projects in the database.
- **Role** - displays the role of the selected user for the project.

2.2.2 Roles Tab

The Roles tab is where you configure roles. Roles determine the read, write, delete, and execute permissions for users in projects. Once roles are created, you assign them to users for each project that you want them to have access to. A user's role can differ from project to project. Click **Assign** on either the Projects tab or Users tab to assign roles.

Figure 2-4 Roles Tab for Oracle Test Manager for Web Applications Users



The Roles tab has the following options:

Add - displays the Add Role dialog box for adding a role.

Edit - displays the Edit Role dialog box for editing the selected role.

Delete - deletes the selected role. If a role is in use, you will be asked to assign another role to users assigned to this role.

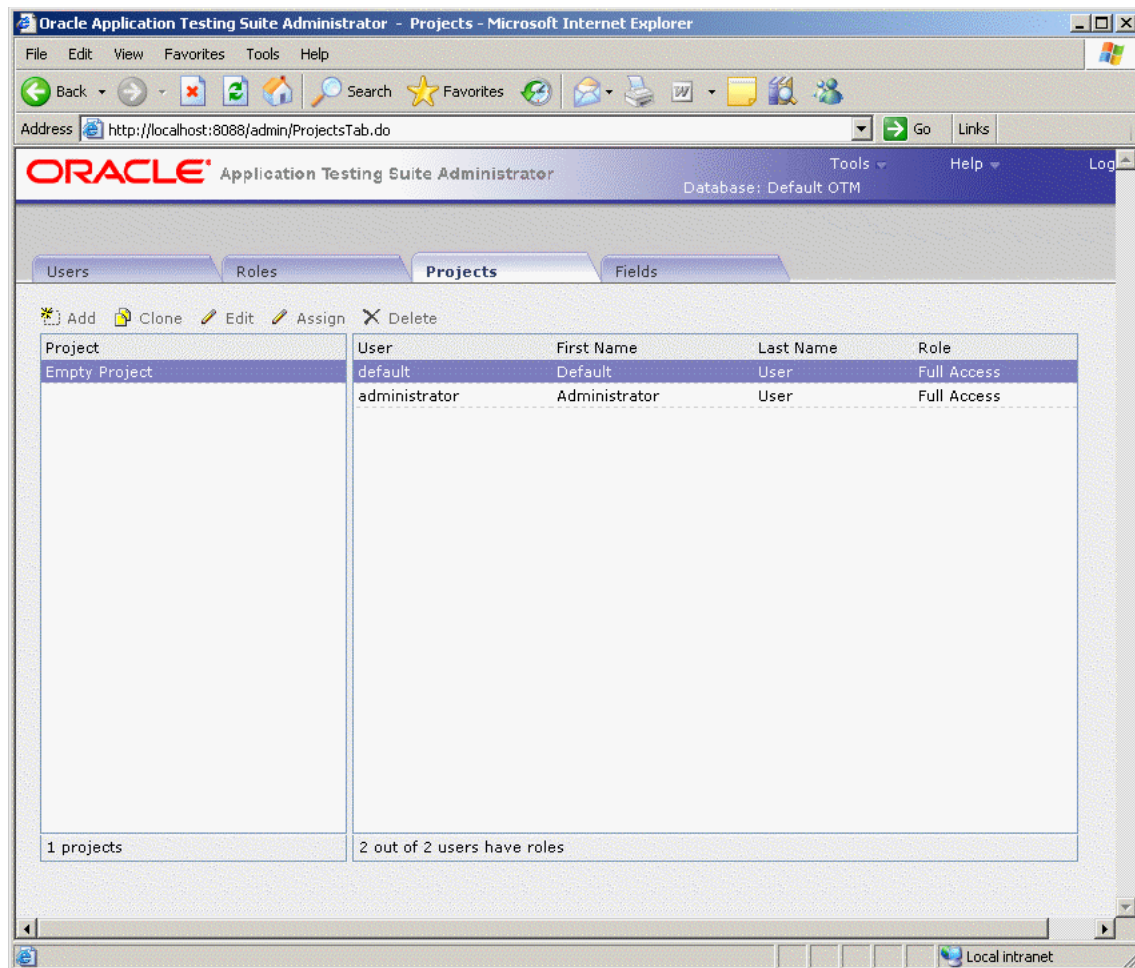
Permissions for <role> - displays the read, write, delete, and execute permissions for this role for projects, requirements, tests, and issues.

- **Project** - displays the read, write, and delete permissions for projects for users assigned to this role.
- **Requirement** - displays the read, write, and delete permissions for requirements for users assigned to this role.
- **Test** - displays the read, write, delete, and execute permissions for tests for users assigned to this role.
- **Issue** - displays the read, write, and delete permissions for issues for users assigned to this role.
- **Public Reports** - displays the read, write, and delete permissions for public reports assigned to this role.
- **My Reports** - displays the read, write, and delete permissions for reports assigned to this role.

2.2.3 Projects Tab

The Projects tab is where you maintain projects.

Figure 2-5 Projects Tab for Oracle Test Manager for Web Applications Users



The Projects tab has the following options:

Add- displays the Add Project dialog box for adding a new project.

Clone - displays the Clone Project dialog box for duplicating the selected project. When you clone a project, user roles are the same as the original project.

Edit - displays the Edit Project dialog box for changing the name of the selected project.

Assign - displays the Edit Role dialog box for assigning roles to users for the selected project(s).

Delete - deletes the selected project.

Project - displays a list of projects in the database.

User - displays the users in the database.

First Name - displays the user's first name.

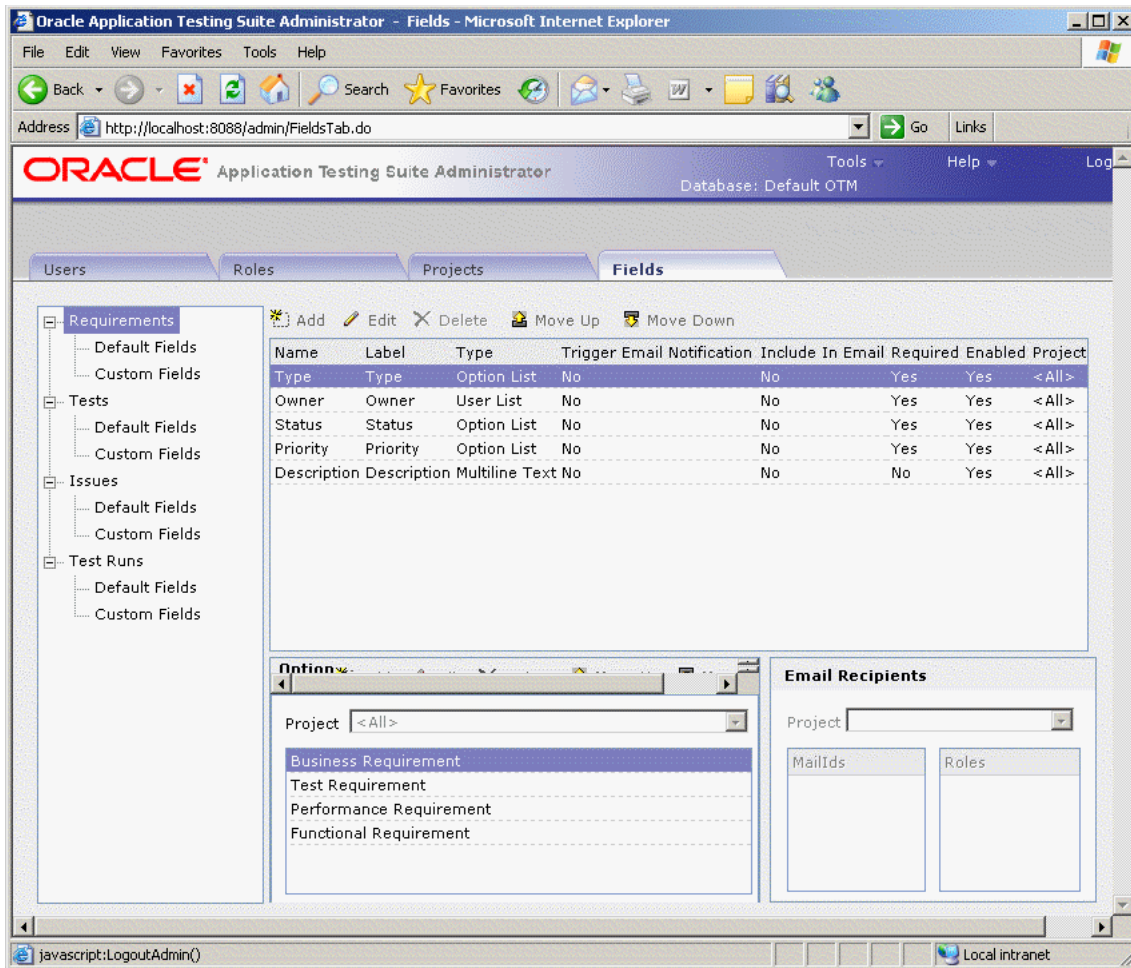
Last Name - displays the user's last name.

Role - displays the users' roles in the selected database.

2.2.4 Fields Tab

The Fields tab is where you customize both default and custom fields. These fields are used in Oracle Test Manager for Web Applications for maintaining details about requirements, tests, issues, and test runs.

Figure 2–6 Fields Tab for Oracle Test Manager for Web Applications Users



The Fields tab has the following options:

<Field List> - lists the categories and types of fields that you can customize. The categories are:

- **Requirements** - fields that pertain to requirements that appear in the Add/Edit Requirements dialog boxes and are displayed in the right pane.
- **Tests** - fields that pertain to tests that appear in the Add/Edit Tests dialog boxes and are displayed in the right pane.
- **Issues** - fields that pertain to issues that appear in the Add/Edit Issues dialog boxes and are displayed in the right pane.
- **Test Runs** - fields that pertain to test runs that appear in the Run Test dialog box and are displayed in the Result Parameters section of the right pane when you click the run date of a test in the Run History section.

Each category has two types of fields:

- **Default Fields** - these are the fields that are shipped with the product. You can add and delete options and change the labels.
- **Custom Fields** - these are user-created fields. These fields are used for entering information in Oracle Test Manager for Web Applications. They are added to the input and edit dialog boxes for requirements, tests, and issues. They are displayed in the right pane with the default fields, and they can be used for grouping and reporting.

Name - displays the field name.

Label - displays the label displayed in Oracle Test Manager for Web Applications.

Type - displays the type of field. The options are:

- **Option List** - lets you select an option from a list.
- **Option List/Text** - lets you select an option from a list or enter text.
- **User List** - creates a list of the users in the database and lets you select one.
- **Text** - lets you enter one line of text.
- **Multiline Text** - lets you enter multiple lines of text.
- **Multiline/Append** - creates a multiline text field and when editing, lets you choose to append new text to existing text. If you choose to append, the date and user name are automatically added.
- **Heading** - lets you create a heading for grouping custom fields. The heading is for display purposes only in the right-hand pane of Oracle Test Manager for Web Applications.

Trigger Email Notification - indicates whether an email will be sent to the configured recipients when this field changes.

Include In Email - indicates whether the field should be included in email.

Required - indicates whether the field is required, that is, data must be entered when the requirement, test, or issue is created.

Enabled - indicates whether the field is being used in Oracle Test Manager for Web Applications.

Project - indicates the project to which this field applies.

Add - displays the Add Field dialog box for adding a custom field.

Edit - displays the Edit Field dialog box for the selected custom field.

Delete - deletes the selected custom field.

Move Up - moves the selected field up one place.

Move Down - moves the selected field down one place.

Option Lists - lets you maintain the options for the selected field if the field is an option list type of field.

- **Add** - displays the Add Option dialog box for adding an option to the selected field.
- **Edit** - displays the Edit Option dialog box for renaming the selected option.
- **Delete** - deletes the selected option.
- **Move Up** - moves the selected option up one place.
- **Move Down** - moves the selected option down one place.

- **Project** - lets you select the project to which the options apply. This field is only available when you select the **Project Specific Options** check box in the Add Custom field dialog box for this field. When this check box is selected you can add options specific to each project; otherwise, all projects will have the same options.

Email Recipients - lists the roles and/or email addresses of the people that will receive an email when the selected field changes.

- **Project** - select the project for which you want to display the email recipients.
- **MailIds** - lists the email addresses to which email will be sent when the field changes.
- **Roles** - lists the roles of the email recipients to which email will be sent when the field changes.

When you select the category in the left pane, the associated fields are displayed in the top right pane. When you select the field in the top right pane, its associated options are displayed in the Option List.

2.3 Oracle OpenScript Main Window Features

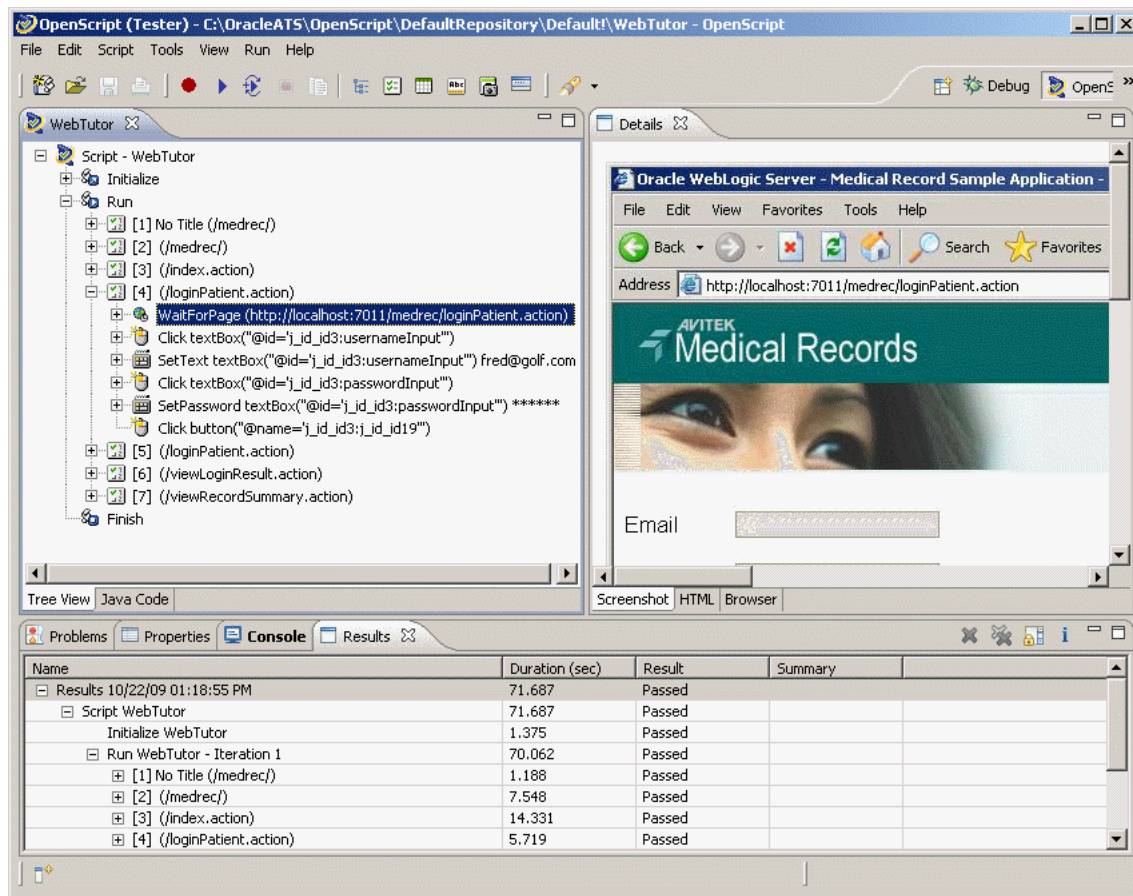
The Oracle OpenScript main window is where you develop the scripts used for functional/regression testing and load/performance testing of your Web site or application. Specific types of scripts you develop using Oracle OpenScript can also be used by Oracle Load Testing for Web Applications or Oracle Test Manager for Web Applications.

Scripts represent a sequence of actions and tests performed on a Web site or application. Scripts are used by Oracle OpenScript for regression testing and Oracle Load Testing for Web Applications for performance (load and scalability) testing.

The Oracle OpenScript main window consists of the menu bar, toolbar, and the scripting workbench in the Eclipse Integrated Development Environment (IDE). The Workbench provides an Eclipse-based scripting platform where you can create and run your automated test scripts. Users can use the Tree View graphical scripting interface for creating and editing scripts through the UI. Users can also switch to the Java Code view programming interface and leverage the integrated Eclipse IDE for creating and editing their scripts programmatically.

The workbench includes a Tester Perspective and a Developer Perspective. The Tester Perspective provides a convenient way to record and edit scripts and view the playback results. The Developer Perspective provides advanced options for developers when creating and editing scripts using the advanced features of OpenScript and the Eclipse development platform.

Figure 2-7 Oracle OpenScript Main Window



The OpenScript Test Modules provide application-specific test automation capabilities. Each Test Module is custom built to test a specific application or protocol. OpenScript includes several functional and load testing modules for testing Web-based applications.

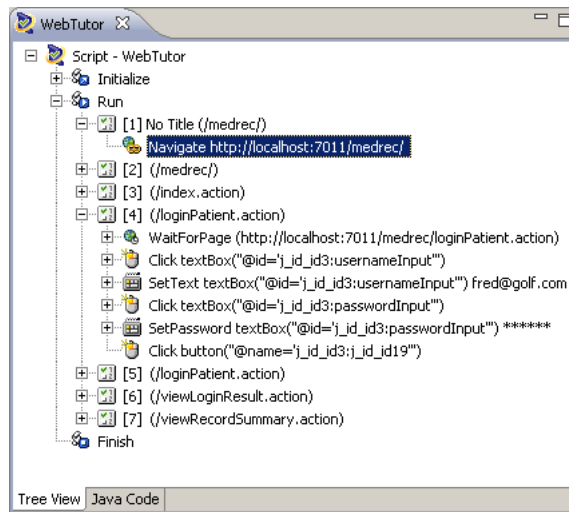
2.3.1 Script View

Shows the recorded script in two tabs: Tree View and Java Code. The Tree View tab shows the steps and pages and the Initialize, Run, and Finish nodes of each step using a graphical tree view. The Java Code tab shows the underlying Java code used for the script.

The script view is where you perform the majority of script editing actions. The Script view has the following tab views:

2.3.1.1 Tree View

The Tree View shows the script navigations and data as nodes in a collapsible tree view. The Tree View corresponds to the Java Code view. Any changes in the Tree View will be automatically updated in the Java Code view.

Figure 2–8 Script Tree View

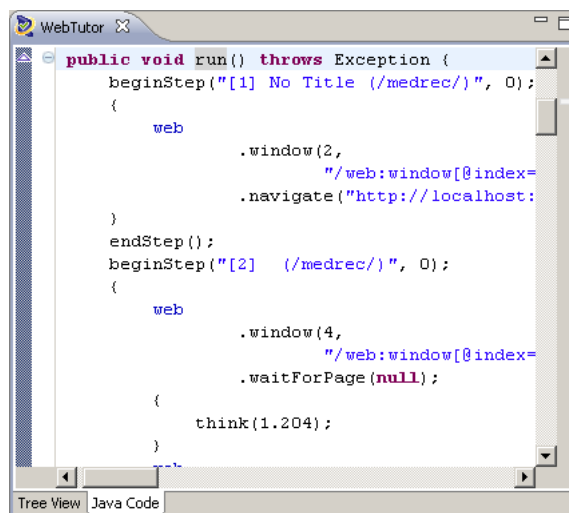
The Tree View has the following standard nodes:

- Initialize - specifies script actions to perform once at the beginning of script playback.
- Run - specifies script actions to perform one or more times during script playback depending upon databanks or other custom programming.
- Finish - specifies script actions to perform once at the end of script playback.

Use the Record options and right-click shortcut menu to add options to script nodes or modify the properties of script nodes in the Tree View.

2.3.1.2 Java Code

The Java Code view shows the script navigations and data as Java programming code. The Java Code view corresponds to the Tree View. Any changes in the Code View will be automatically updated in the Tree View.

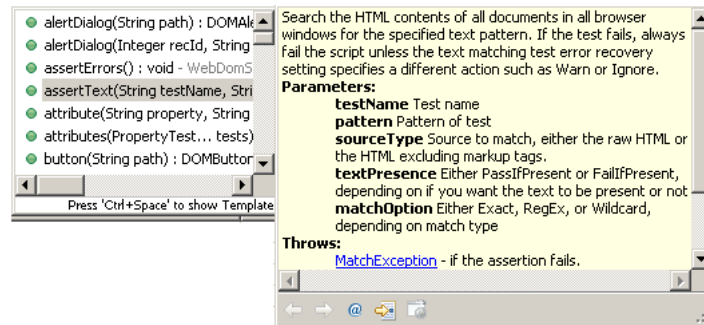
Figure 2–9 Script Java Code View

The Java Code view has the following standard procedures:

- `initialize()` - corresponds to the Initialize node of the Tree View and executes any custom code added once at the beginning of script playback.
- `run()` - corresponds to the Run node of the Tree View and executes recorded and custom code one or more times during script playback depending upon databanks or other custom programming.
- `finish()` - corresponds to the Finish node of the Tree View and executes any custom code added once at the end of script playback.

Use Ctrl-space to open an Intellisense window listing available procedures.

Figure 2–10 Java Code View Intellisense Window

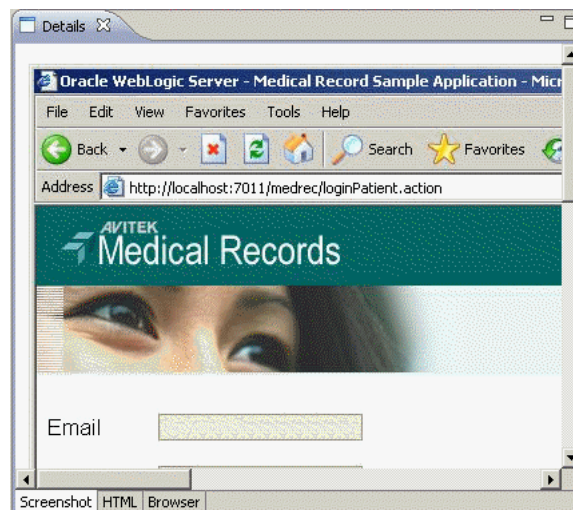


See the API Reference in the OpenScript Platform Reference help for additional programming information.

2.3.2 Details View

The Details view shows the content details for URL navigations added to the script.

Figure 2–11 Details View Showing Screenshot Tab View



The Details view may have the following tab views depending upon the selected script node and type of script:

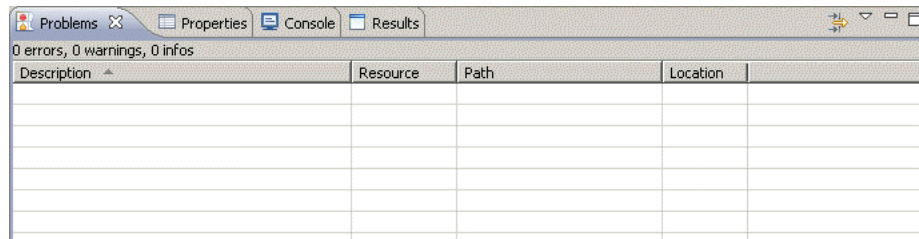
- ScreenShot - shows a screen capture of the web page.
- Browser - shows the Browser rendered page for the script navigation selected in the tree view.

- HTML - shows the HTML source for the script navigation selected in the tree view.
- Headers - shows the Request Header and Response Header source for the script navigation selected in the tree view.
- Comparison - shows the recorded and playback text for the Content, Request Header, or Response Header selected in the Compare list. The Comparison tab appears only after a script is played back and a navigation is selected in the Results View.
- Results Report - shows the results report for the script playback. The Results Report tab appears only after a script is played back and a navigation is selected in the Results View.

2.3.3 Problems View

The Problems view shows any problems in the script code that may produce errors or prevent compiling the script.

Figure 2–12 Problems View



The Problems view shows the following information:

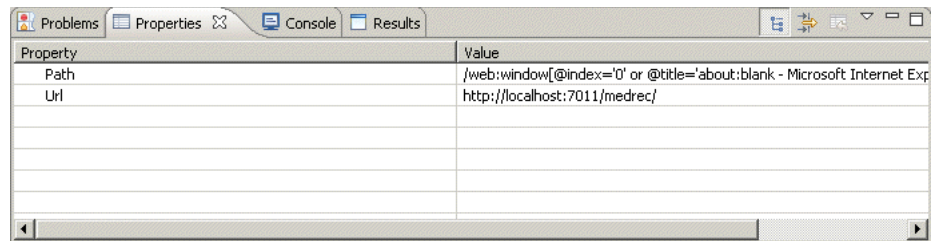
- # error, # warnings, # infos - shows the number of errors, warning messages, and information messages in the problems view.
- Description - shows a description of the errors, warning messages, and information messages.
- Resource - shows the name of the resource file where the error, warning, or information message was generated.
- Path - shows the script name, workspace, and repository path where the resource file is located.
- Location - shows the location/line number where the error, warning, or information message was generated.

The following toolbar button is available in the Problems View:

- Configure the filters to be applied to this view - opens a dialog box for configuring the filters to apply to the Problems View.

2.3.4 Properties View

The Properties view shows the properties for the selected node in the script.

Figure 2–13 Properties View

The Properties view shows the following information:

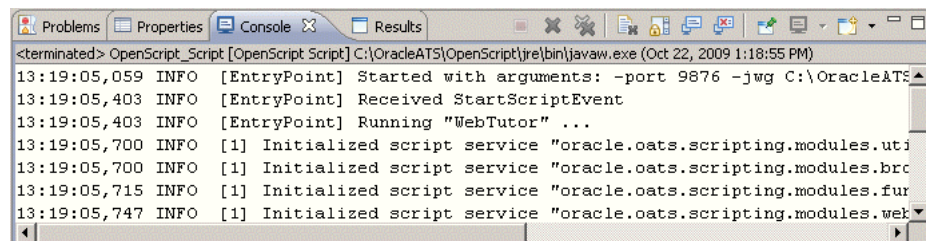
- Property - shows the names of the properties for the script node. The properties vary depending upon which type of script node is selected.
- Value - shows the value of the script node properties. Property values can be edited in the properties view.

The following toolbar buttons are available in the Properties View:

- Show Categories - toggles the property categories.
- Show Advanced Properties - toggles the advanced properties.
- Restore Default Value - restores any changed property values to the default values.

2.3.5 Console View

The Console view shows the playback command output and status information for the script. Script log message also appear in the Console.

Figure 2–14 Console View

See the Process Console View topics in the reference section of the Java development user guide online help for additional information about console toolbar options.

2.3.6 Results View

The Results view shows the playback results for the script.

Figure 2–15 Results View

Name	Duration (sec)	Result	Summary
Results 10/22/09 01:18:55 PM	71.687	Passed	
Script WebTutor	71.687	Passed	
Initialize WebTutor	1.375	Passed	
Run WebTutor - Iteration 1	70.062	Passed	
[1] No Title (/medrec/)	1.188	Passed	
[2] (/medrec/)	7.548	Passed	
[3] (/index.action)	14.331	Passed	
[4] (/loginPatient.action)	5.719	Passed	

The Results view shows the following information:

- Name - shows the test date or navigation name.
- Duration - shows the playback time for the page navigations.
- Result - shows the playback result: Passed or Failed.
- Summary - shows the data values from the Data Bank that are passed to parameters or it shows failure descriptions.

The following toolbar buttons are available in the Result View:

- Delete Result - deletes the selected result row.
- Delete All Results - deletes all rows from the Results View.
- Scroll Lock - toggles scroll lock on and off for the Result View.
- Properties - opens the Properties for the selected result.

2.3.7 Other Views

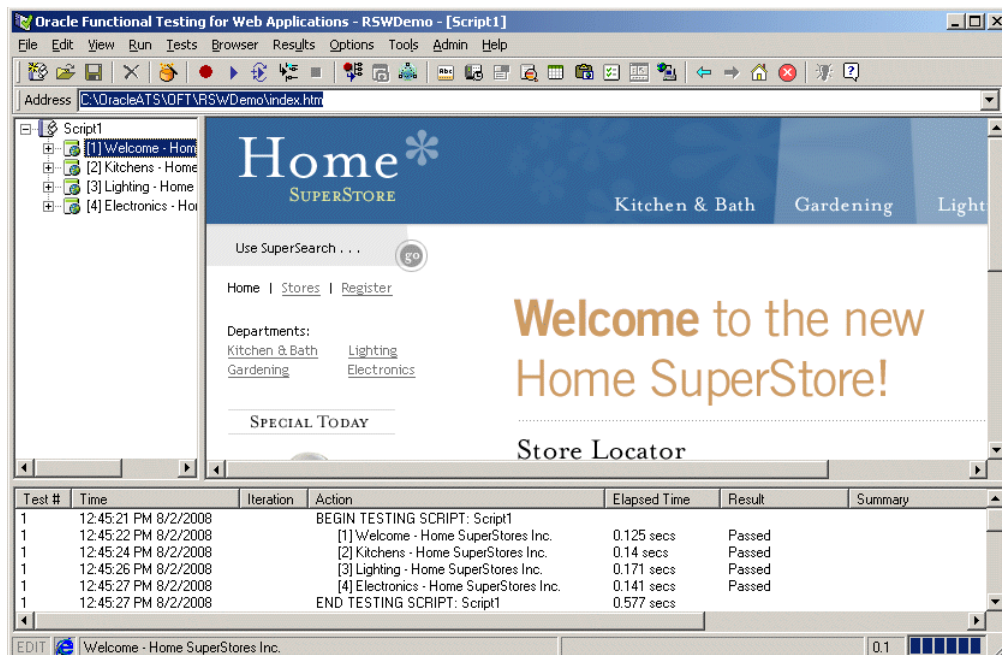
The OpenScript Developer Perspective includes several other views, such as Debug, Breakpoints, Navigator, and Package, that can be used by advanced users for specific testing and debugging purposes. See the *OpenScript User's Guide* for additional information.

2.4 Oracle Functional Testing for Web Applications Main Window Features

The Oracle Functional Testing for Web Applications main window is where you develop the Visual Scripts used for functional/regression testing, performance testing, and operational monitoring of your Web site or application. The Visual Scripts you develop using Oracle Functional Testing for Web Applications are also used by Job Scheduler, Oracle Load Testing for Web Applications, and Oracle Test Manager for Web Applications.

Visual Scripts represent a sequence of actions and tests performed on a Web site or application. Visual Scripts are used by Oracle Functional Testing for Web Applications and Job Scheduler for regression testing, Oracle Load Testing for Web Applications for performance (load and scalability) testing.

The Oracle Functional Testing for Web Applications main window consists of the menu bar, toolbar, and three panes: the Visual Script pane, Browser pane, and Playback Results Log pane.

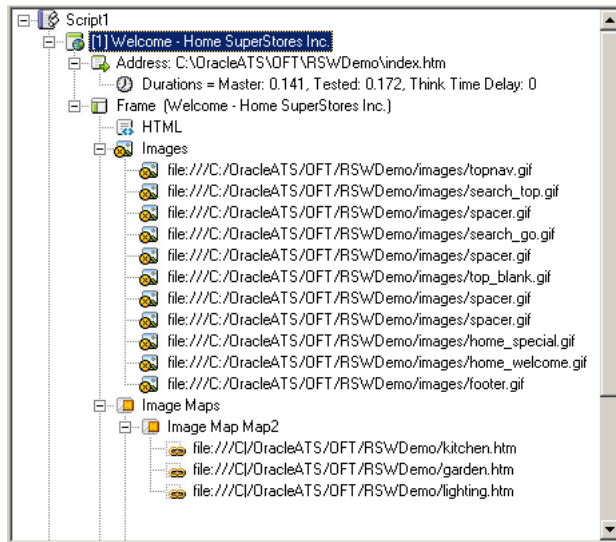
Figure 2–16 Oracle Fuctional Testing for Web Applications Main Window

The Title bar of the window shows the program name followed by the current Workspace and Visual Script name.

The Address box directly above the Web browser pane is where you enter the URL or file location of the Web page(s) to test. The bottom of the main window includes a status line.

2.4.1 Visual Script Pane

The Visual Script pane shows the tree hierarchy of recorded Web sites and pages. When you first start Oracle Functional Testing for Web Applications, the Visual Script pane is empty. When you record Web pages, Oracle Functional Testing for Web Applications creates the Visual Script tree for you.

Figure 2–17 Visual Script Pane

Click the Plus icon to expand a branch or the Minus icon to collapse a branch.

The Visual Script tree will include any test cases you insert into the Visual Script. Each item in the tree is identified by an icon and a text description.

You can toggle the Visual Script pane width by selecting **Resize Visual Script View** from the **View** menu or by dragging the border between the Browser pane and the Visual Script pane.

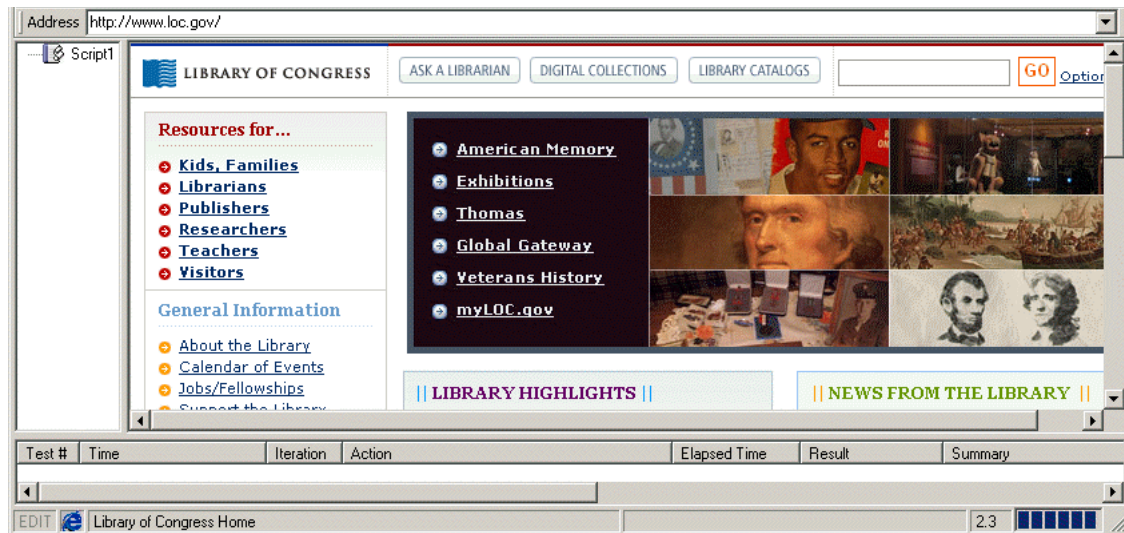
The Visual Script uses additional icons in the tree to represent the following:

- Yellow flag: skip test.
- Red flag: test case failure.
- Red and yellow flags: ignore test failure.

2.4.2 Browser Pane

The browser pane contains a seamlessly integrated Web browser that you use to select the Web pages to test. It provides full navigation and Web access.

Figure 2–18 Browser Pane



Enter the full path and file name of the URL or local file, or drop down the list to select from recently accessed Web pages.

2.4.3 Playback Results Pane

The Playback Results pane shows a summary of the Visual Script test playback.

Figure 2–19 Playback Results Pane

Test #	Time	Iteration	Action	Elapsed Time	Result	Summary
1	10:49:39 AM 4/6/2004		[1] Welcome - Home SuperStores Inc.	0.29 secs	Passed	
1	10:49:40 AM 4/6/2004		[2] Registration - Home SuperStores Inc.	0.15 secs	Passed	
1	10:49:43 AM 4/6/2004		[3] Registered - Home SuperStores Inc.	0.17 secs	Passed	
1	10:49:47 AM 4/6/2004		Resource Validation		Passed	
1	10:49:47 AM 4/6/2004		END TESTING SCRIPT: tutor2	0.61 secs		

You can adjust the widths of the individual columns by dragging the dividers.

Icons in the Visual Script show the location of any specific failures of default tests or test cases. Resource Validation test results are listed in a separate window after playback of the script.

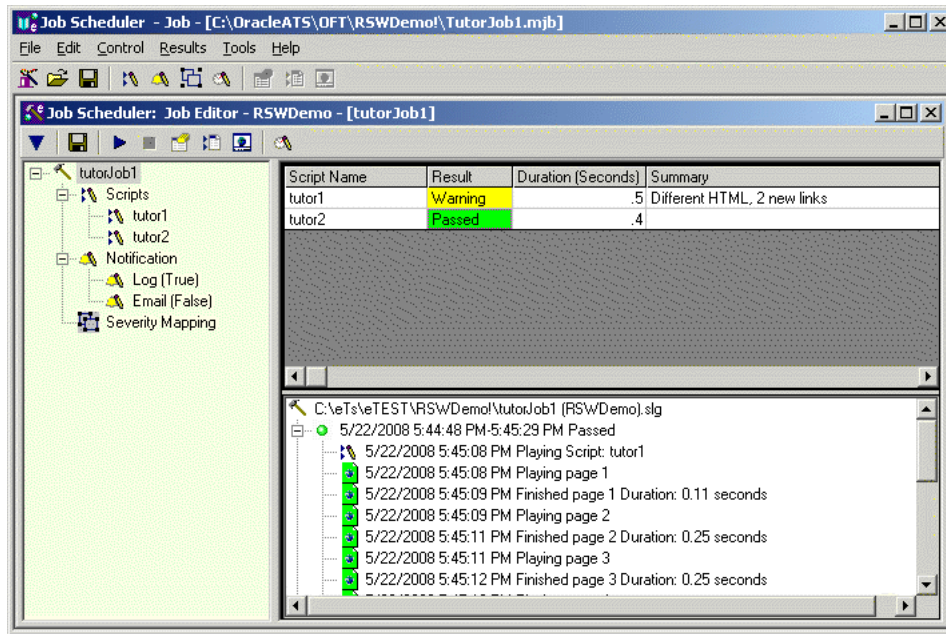
You can turn on and off the display of the Playback Results pane by selecting **Resize Results Window** from the **View** menu.

2.5 Job Scheduler Main Window Features

Job Scheduler is a regression testing tool used for running multiple Oracle Functional Testing for Web Applications Visual Scripts as a single job. The Job Scheduler main window is where you perform immediate or scheduled playback of a set of Oracle Functional Testing for Web Applications Visual Scripts.

The main window consists of the menu bar and toolbar. Job Scheduler has three windows that run within the main window: Current Schedule window, Current Job window, and Job Editor window.

Figure 2–20 Job Scheduler Main Window



You can open the Job Scheduler application from the Start menu or by selecting **Job Scheduler** from the **Tools** menu in Oracle Functional Testing for Web Applications.

2.5.1 Visual Script Job Pane

The Visual Script job pane lists the Visual Scripts in an Job Scheduler job and the real-time playback results. You create Job Scheduler jobs and schedules using the Job Scheduler Wizard.

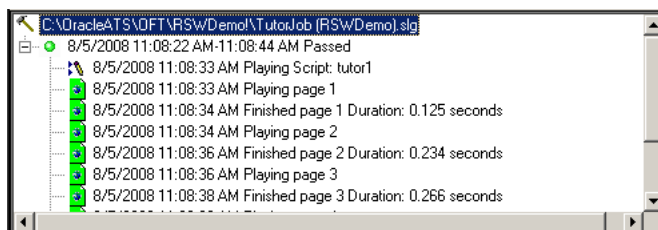
Figure 2–21 Job Scheduler Job Pane

Script Name	Result	Duration (Seconds)	Summary
tutor1	Failed	1.3	Different HTML, 2 missing links
tutor2	Passed	2.7	

2.5.2 Results Pane

The Results pane shows any log messages generated during playback of the job.

Figure 2–22 Job Scheduler Results Pane

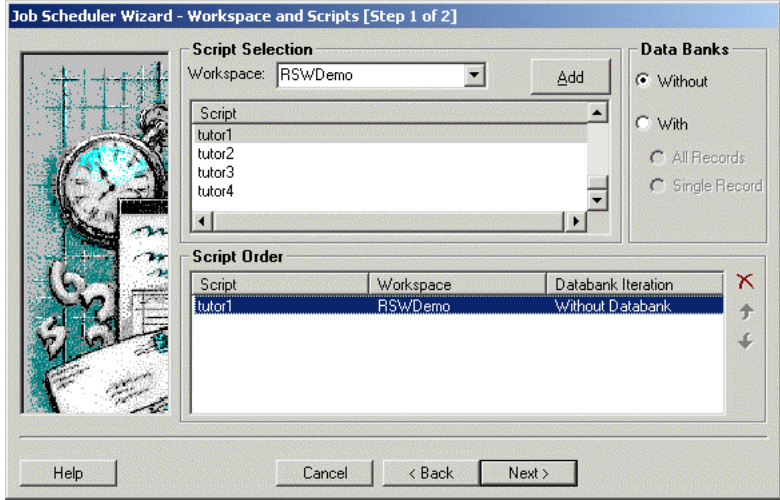


You can customize log messages as required using the Job Scheduler Wizard.

2.5.3 Job Scheduler Wizard

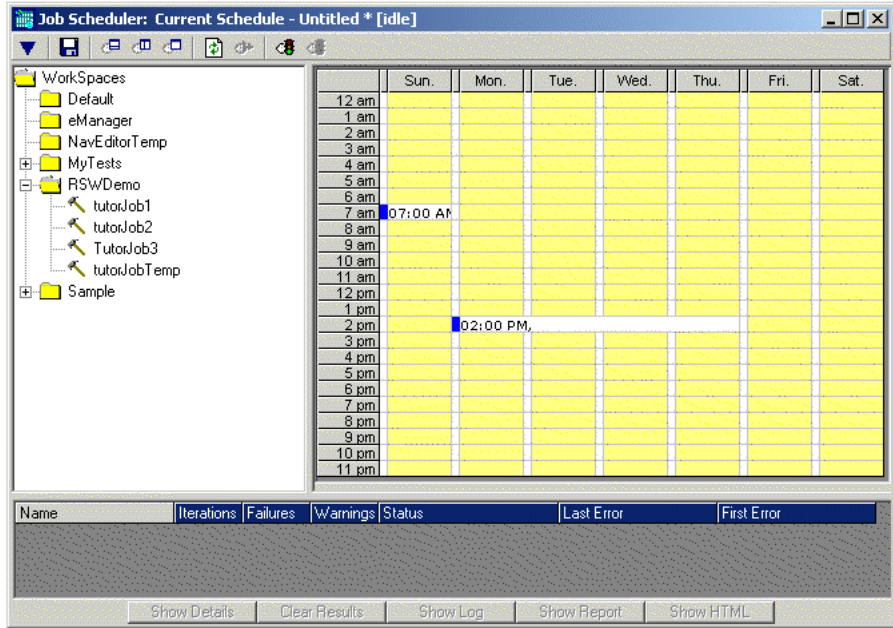
The Job Scheduler Wizard provides a convenient way to build and schedule Job Scheduler jobs. The Wizard includes steps for selecting Visual Scripts, setting notification options, and scheduling playback times.

Figure 2-23 Job Scheduler Wizard



The successive steps of the Wizard provide options for setting results notifications. When the Wizard finishes, you can add the job to any schedule.

Figure 2-24 Job Scheduler Schedule Window



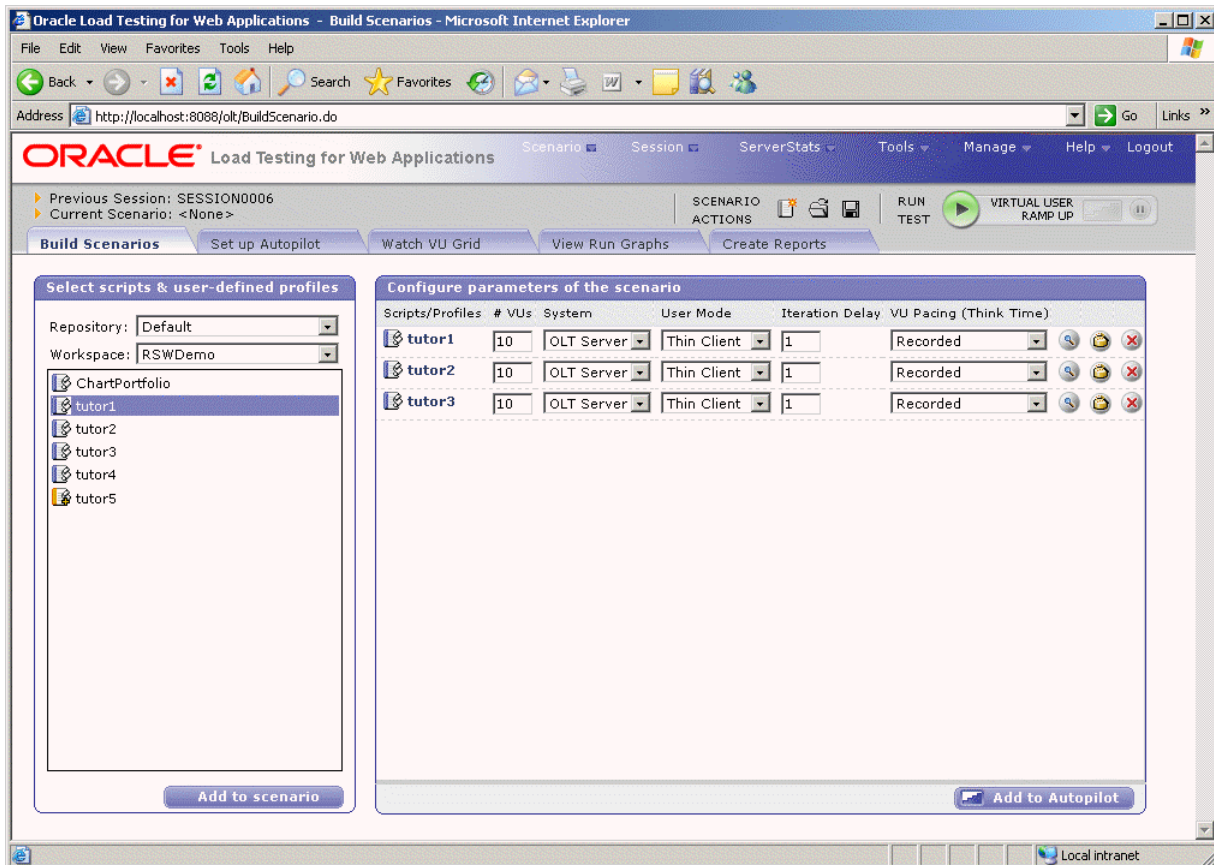
Schedules can be used with the current job or other saved jobs.

2.6 Oracle Load Testing for Web Applications Main Window Features

The Oracle Load Testing for Web Applications main window is where you perform the majority of your load/performance testing activities. Oracle Load Testing for Web Applications uses Visual Scripts that you develop using Oracle Functional Testing for Web Applications or Java-based scripts you develop using OpenScript.

The main window consists of the menu bar, toolbar, and the controller tab dialogs.

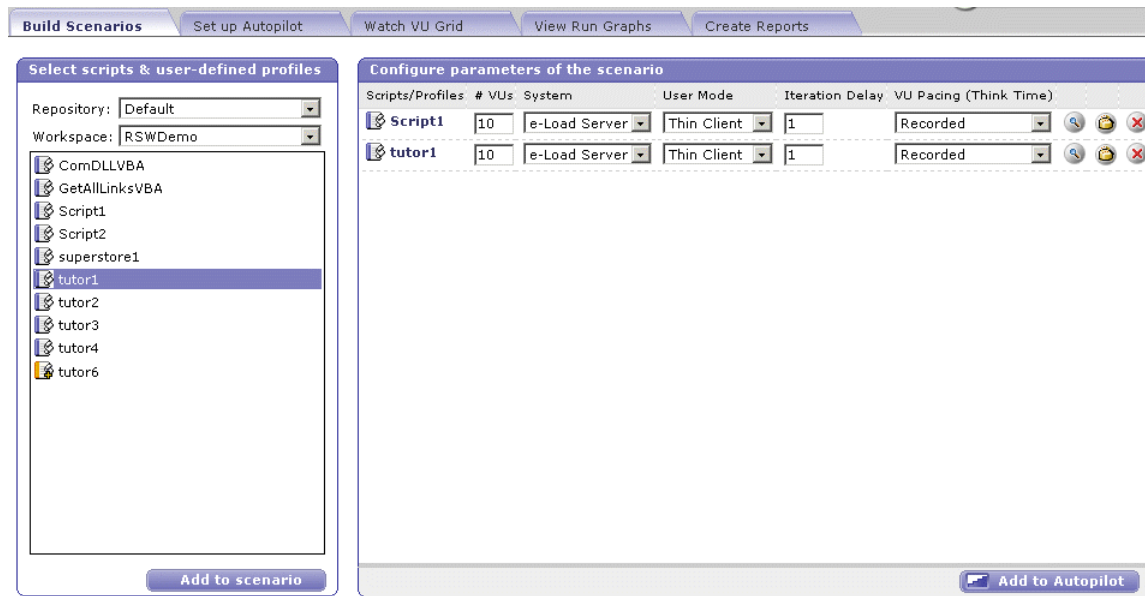
Figure 2–25 Oracle Load Testing for Web Applications Main Window



You can open the Oracle Load Testing for Web Applications application from the **Start** menu or by selecting **Oracle Load Testing for Web Applications** from the **Tools** menu in Oracle Functional Testing for Web Applications.

2.6.1 Build Scenario Tab

The Build Scenario tab is where you specify information about the virtual users to include in the load test and the attributes for each set of virtual users.

Figure 2–26 Build Scenarios Tab

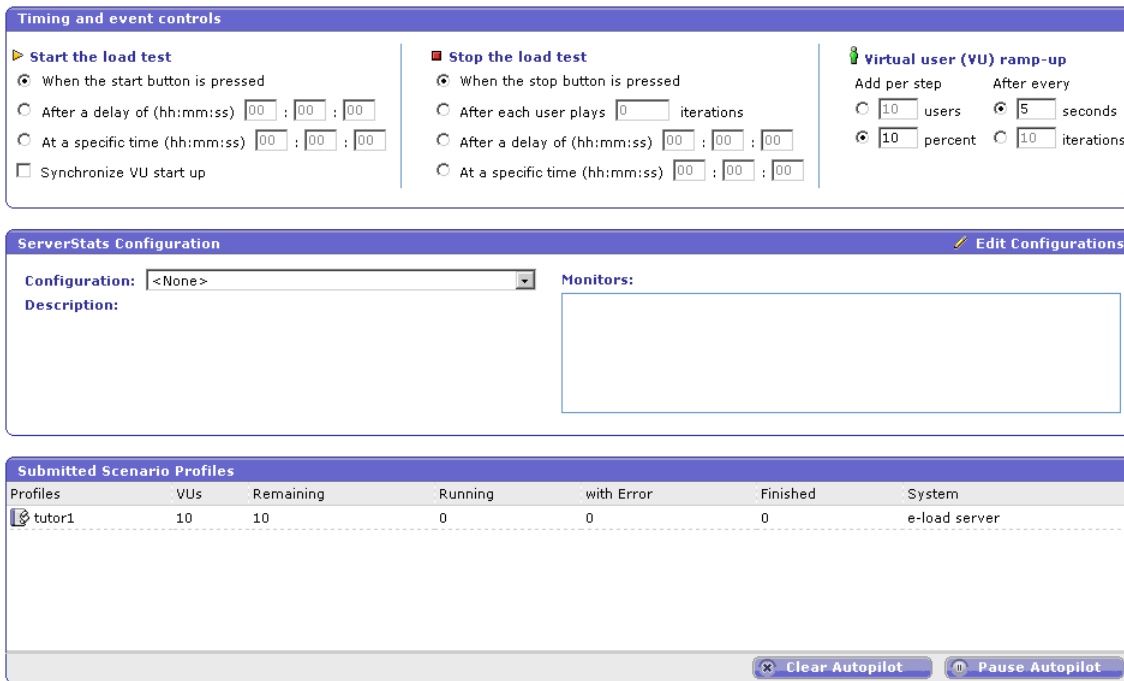
You can define user profiles that specify which visual scripts the users playback to emulate real users and how many virtual users to emulate.

2.6.2 Set up Autopilot Tab

The Autopilot tab is where you specify the information needed to control how the scenario starts and runs. The Autopilot controls the starting and stopping of the scenario, the rate at which new virtual users are started, and shows the total number of virtual users and the number of running virtual users.

You specify the session, start and stop times, and the virtual user rampup specifications for the Submitted Scenario Profile. It also shows the list of virtual user profiles submitted in the Oracle Load Testing for Web Applications scenario.

Figure 2–27 Autopilot Tab



2.6.3 Watch Virtual User Grid Tab

The Watch Virtual User Grid tab lists the currently running virtual users and the profile and playback details associated with each.

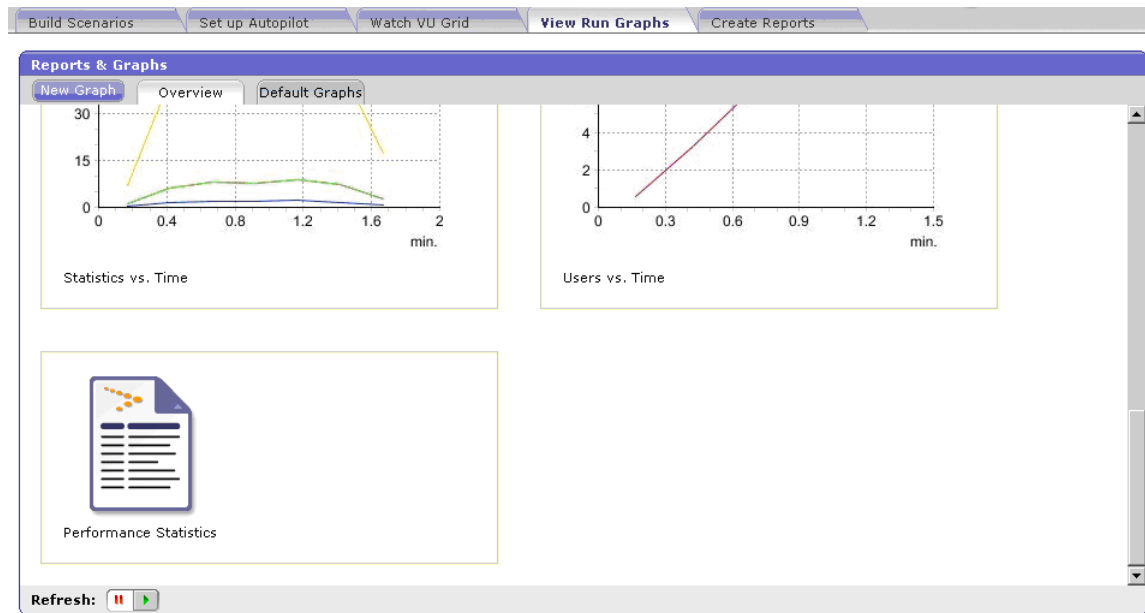
Figure 2–28 Watch Virtual User Grid Tab

VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Page	System	Data Bank	Current Error	Previous Error
1	tutor1	Finished	5		7.09		localhost			
2	tutor1	Finished	5		6.649		localhost			
3	tutor1	Finished	5		6.179		localhost			
4	tutor1	Finished	5		6.149		localhost			
5	tutor1	Finished	5		6.109		localhost			
6	tutor1	Finished	5		6.249		localhost			
7	tutor1	Finished	5		6.079		localhost			
8	tutor1	Finished	5		6.569		localhost			
9	tutor1	Finished	5		6.149		localhost			
10	tutor1	Finished	5		6.089		localhost			

2.6.4 View Run Graphs Tab

The View Run Graphs tab lets you define graphs and view the graphs at run-time.

Figure 2–29 View Run Graphs Tab

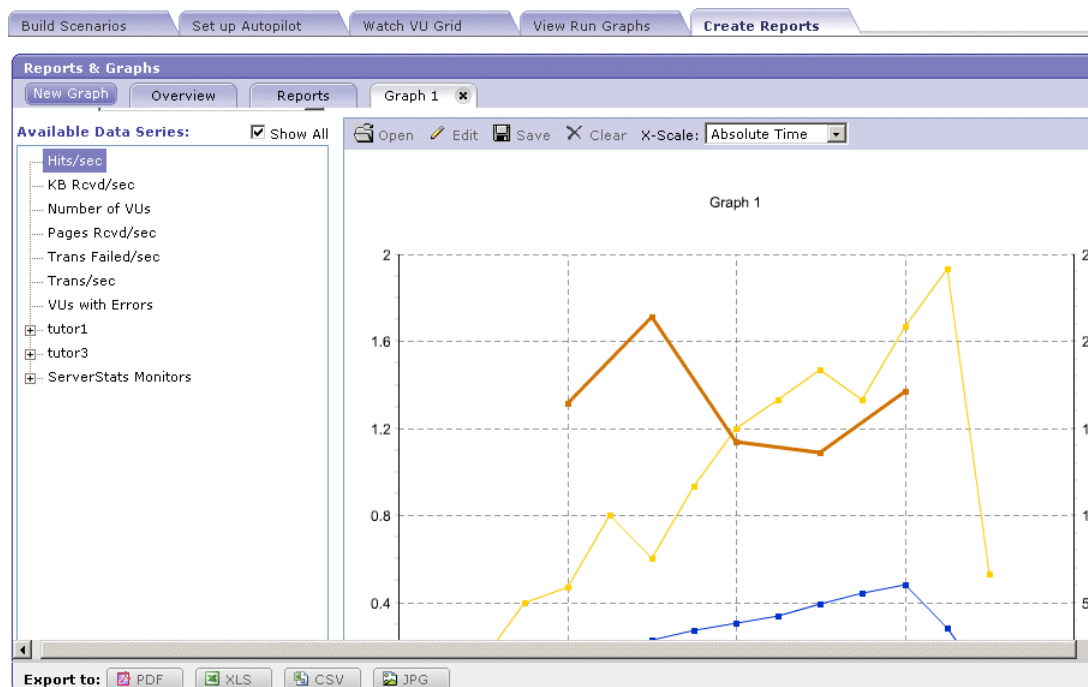


You can also view the Performance Statistics report from the View Run Graphs tab. The Performance Statistics window shows a summary of the performance data for the running virtual users.

2.6.5 Create Reports Tab

The Create Reports tab lets you create post session reports and graphs.

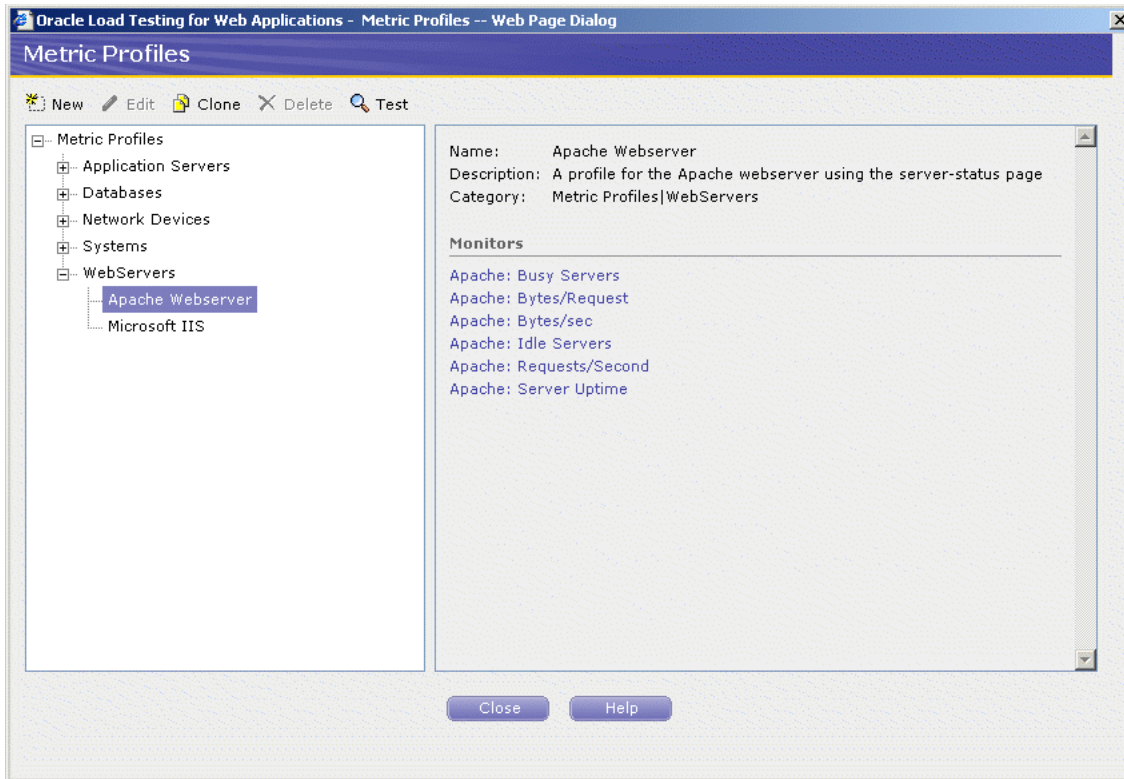
Figure 2–30 Create Reports Tab



2.6.6 ServerStats

The ServerStats component of Oracle Load Testing for Web Applications lets you monitor a variety of server-side application, database, system, and Web server statistics. You can configure ServerStats to display real-time performance statistics for the various hosts and services available from the server such as, percentage of CPU usage, memory usage, Web server statistics, etc.

Figure 2–31 *ServerStats Metric Profiles Window*



You can monitor specific counters in real time using the visual indicator gauges or using graphs. In addition to performance monitoring, ServerStats let you define scripts that can log warnings or alarms if a server's counter performance goes outside a defined range.

2.7 Oracle Test Manager for Web Application Main Window Features

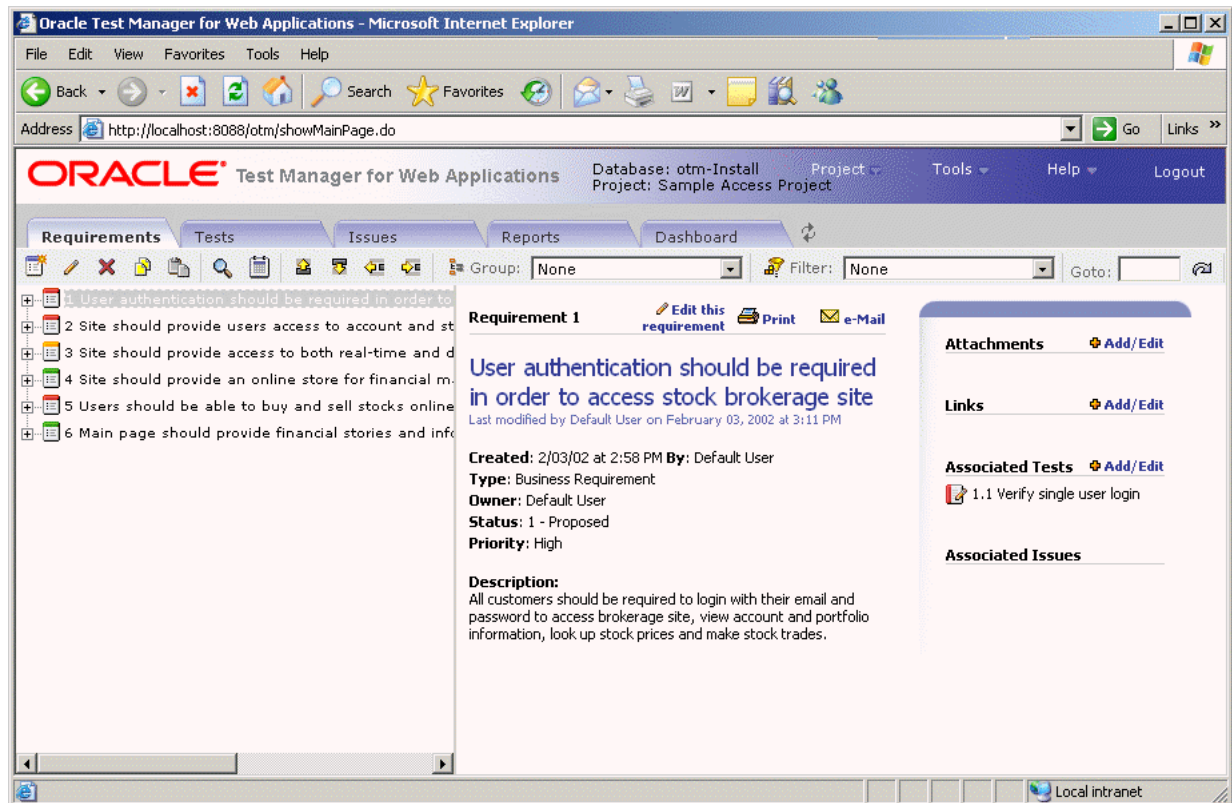
Oracle Test Manager for Web Applications lets you create projects that group together and organize test scripts, requirements that need to be tested, and issues resulting from the tests. Once created, you can indicate the relationships among these items, allowing you to quickly and easily find all information pertaining to a particular test script, requirement, or issue.

The main window consists of the menu bar, toolbar, and the tab views.

2.7.1 Requirements Tab

The requirements tab lets you work with requirements.

Figure 2–32 Requirements Tab



The number of requirements displayed is determined by the number you enter in the Maximum Tree Nodes field in options. You can:

- Expand and collapse the tree view by clicking on the plus and minus signs.
- View the next or previous group of requirements by clicking the **Next** or **Previous** button.
- Hold the mouse over a node to view a count of its child nodes.
- Move requirements by selecting a requirement and using the move buttons.
- Search requirements by clicking the **Find** button.
- Group requirements by clicking **Group**.
- Filter requirements by clicking **Filter**.
- Toggle between the tree view and grid view by clicking the **Tree View** and **Grid View** buttons.

The color of the icon in front of the requirement indicates its priority. The default colors are as follows and can be changed by changing the order of the Requirement Priorities in the Administrator.

- Red, high priority
- Yellow, medium priority
- Green, low priority

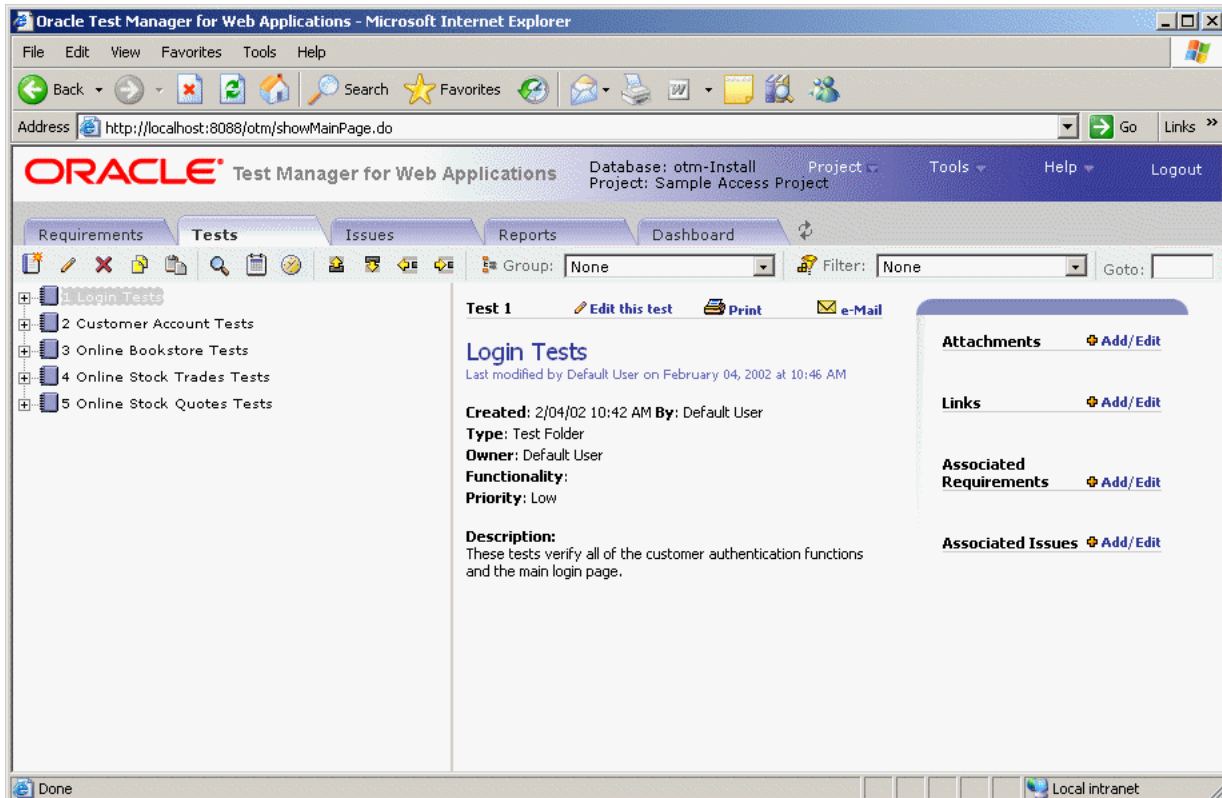
The right pane lists the selected requirement's details. In the upper right corner, associated tests and issues are listed as well as attachments and links. You can:

- Click **Edit this requirement** to open the Edit Requirement dialog box.
- Click **Print** to print the right pane.
- Click **e-Mail** to e-mail the requirement. The title and description are automatically copied to the e-mail.
- Click on an associated test or issue to view its details.
- Click on an attachment to open it in the appropriate application.
- Click on a link to view the URL in a separate browser window.
- Select **Add/Edit** to add or edit attachments, links, or associated items.

2.7.2 Tests Tab

The tests tab lets you work with tests.

Figure 2–33 Tests Tab



The number of tests displayed is determined by the Maximum Tree Nodes setting in options. You can:

- Expand and collapse the tree view using the plus and minus signs.
- View the next or previous group of tests by clicking the **Next** or **Previous** button.
- Hold the mouse over a node to view a count of its child nodes.
- Move tests by selecting the test and using the **Move** buttons.
- Search tests by clicking the **Find** button.
- Group tests by clicking the **Group** button.

- Filter tests by clicking the **Filter** button.
- Schedule when to run tests by clicking the **Schedule** button.
- Toggle between the tree view and grid view by clicking the **Tree View** and **Grid View** buttons.

The icon in front of the test indicates the type of test as follows:

- Manual test - blue with a pencil.
- Test folder - blue with a spiral.
- Oracle Functional test script - blue with a scroll.
- Oracle OpenScript - blue with a pencil.
- Test group - blue with a plus sign.
- 3rd Party test - blue with two stars

The color of the icon in front of the test indicates the last result from running the test, as follows:

- Green, passed
- Red, failed
- Yellow, warning
- Blue, not run
- Silver, currently running

The right pane lists the selected test's details. Test steps and run history are displayed. You can:

- Click **Edit this test** to open the Edit Test dialog box.
- Click **Print** to print the right pane.
- Click **e-Mail** to e-mail the test. The title and description are automatically copied to the e-mail.
- Click **Run this test** to start the Run Manual Test wizard or run an Oracle Functional Testing for Web Applications test.
- Click **Delete Results** to display the Delete Results dialog box for deleting results from particular test runs.
- For Oracle Functional Testing for Web Applications Tests, click **Download Test Script** to download the test script to your machine.
- Click the date in the **Run History** section to display result details for a particular run.

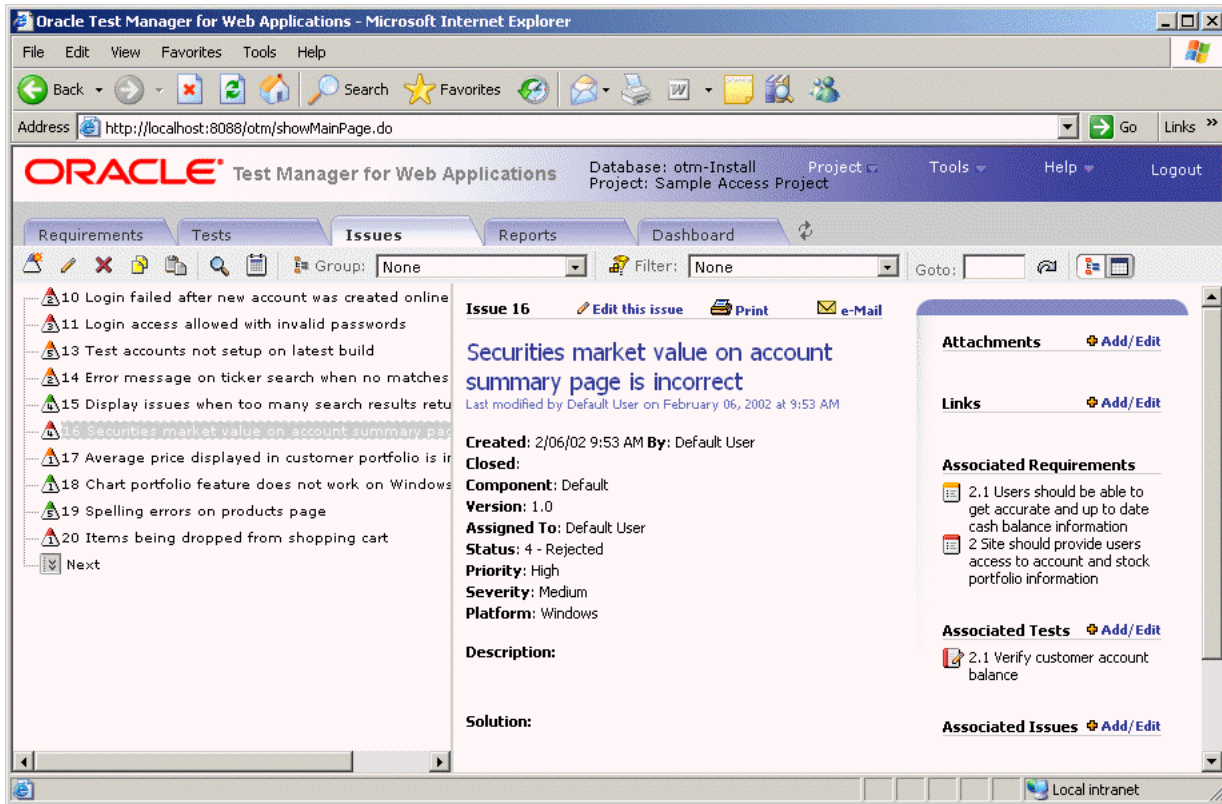
In the upper right corner of the right pane, associated requirements and issues are listed as well as attachments and links. You can:

- Click on an associated requirement or issue to view its details.
- Click on an attachment to open it in the appropriate application.
- Click on a link to view the URL in a separate browser window.
- Select **Add/Edit** to add or edit attachments, links, or associated items.

2.7.3 Issues Tab

The issues tab lets you work with issues.

Figure 2–34 Issues Tab



The number of issues displayed is determined by the Maximum Tree Nodes setting in options. You can:

- View the next or previous group of issues by clicking the **Next** or **Previous** button.
- Group issues by clicking the **Group** button.
- Filter issues by clicking the **Filter** button.
- Hold the mouse over a node to view a count of its child nodes.
- Search issues by clicking the **Find** button.
- Toggle between the tree view and grid view by clicking the **Tree View** and **Grid View** buttons.
- Display a particular issue by entering the issue number in the **Goto** field and clicking the **Goto** button.

The color of the icon in front of the issue indicates its priority. The default colors are as follows and can be changed by changing the order of the Issue Priorities in Oracle Test Manager for Web Applications Administrator:

- Red, high priority
- Yellow, medium priority
- Green, low priority

The number inside the icon corresponds to the status number.

The right pane lists information about the selected issue including the issue's details, solution, priority, and status. You can:

- Click **Edit this issue** to open the Edit Issue dialog box.
- Click **Print** to print the right pane.
- Click **e-Mail** to e-mail the issue. The title and description are automatically copied to the e-mail.

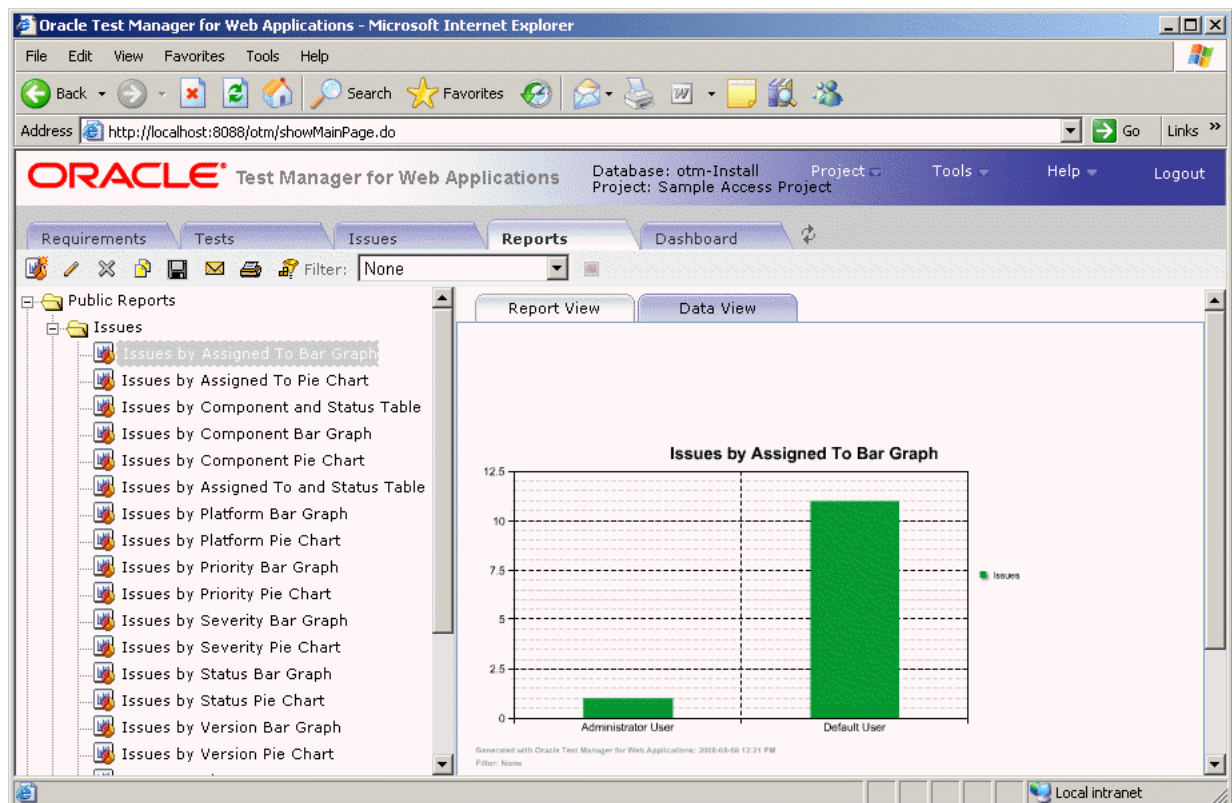
In the upper right corner of the right pane, associated requirements, tests, and issues are listed as well as attachments and links. You can:

- Click on an associated test, requirement, or issue to view its details.
- Click on an attachment to open it in the appropriate application.
- Click on a link to view the URL in a separate browser window.
- Select **Add/Edit** to add or edit attachments or links.

2.7.4 Reports Tab

The Reports tab lets you work with both standard and custom reports.

Figure 2–35 Reports Tab



Oracle Test Manager for Web Applications comes with a standard set of reports that can be viewed as either a graphic or as data. In addition, you can create custom reports to display only the data that you are interested in. You can:

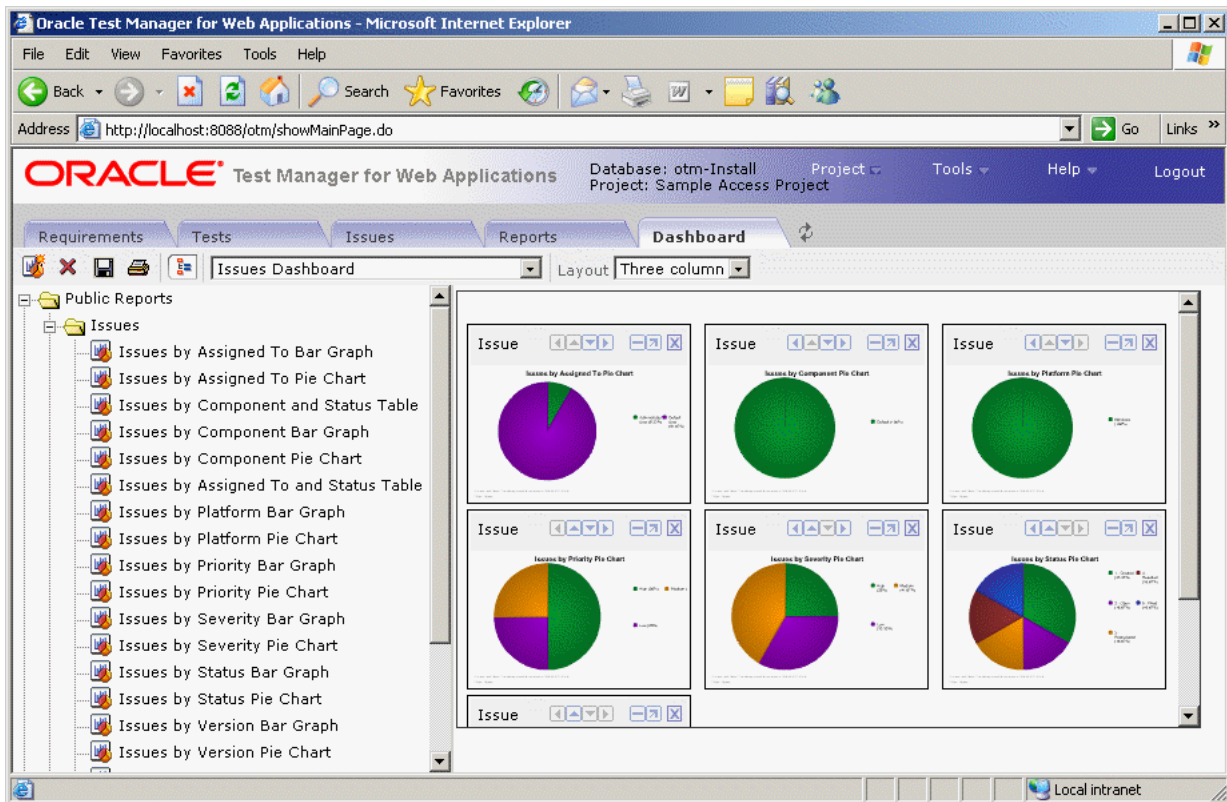
- View a standard or custom report by selecting it from the left tree.
- Add a custom report by clicking **Add**.

- Edit a custom report by clicking **Edit**.
- Delete a custom report by clicking **Delete**.
- Clone an existing report to create a copy of it that you can then edit by clicking **Clone**.
- Save custom reports by clicking **Save**.
- Email reports by clicking **Email**.
- Print reports by clicking **Print**.
- **Filter** the fields in the report to display only the data in which you are interested.
- Export reports to jpg, and xls formats.

2.7.5 Dashboard Tab

The Dashboard tab lets you view an overview of reports.

Figure 2–36 Dashboard Tab



One Dashboard report is available for requirements, tests, and issues. You can customize which reports are displayed for each and then save the view. In addition, you can select the number of columns to use for the display.

You can:

- Add reports to the view by clicking **Add**.
- Remove the selected dashboard report by clicking **Delete**.
- Save the customized view by clicking **Save**.

- Print dashboard reports by clicking **Print**.
- Toggle the display of the report tree by clicking **Toggle**.

Each report has the following toolbar with the following options, described from left to right.

Move Left - moves the report one space to the left.

Move Up - moves the report up one space.

Move Down - moves the report down one space.

Move Right - moves the report one space to the right.

Minimize - minimizes the report.

Maximize - displays the report in a separate window. From there you can toggle between report and data views, and export the report.

Delete - removes the report from the display.

Oracle OpenScript Tutorial

This tutorial walks you through the main features of Oracle OpenScript. The tutorial consists of the following examples:

- **Starting the Avitek Medical Records Sample Application** - explains how to start the sample Avitek Medical Records Server for use with this tutorial.
- **Starting Oracle OpenScript** - describes how to start OpenScript.
- **Creating a Web Functional Test Script** - describes how to create a Web Functional test script project and record a script.
- **Working with Scripts** - explains the features of the OpenScript script tree and how to examine the structure and content of a recorded script.
- **Playing Back a Visual Script** - explains the procedure for playing back Web Functional Test scripts that you have recorded and viewing the results.
- **Adding Tests to the Script** - explains how to add tests to your OpenScript scripts.
- **Creating a HTTP Test Script with Databanks** - introduces Databanks and explains how to use Databanks to run iterative tests on a login form using data from an external file.
- **Stopping the Avitek Medical Records Server** - explains how to stop the sample Avitek Medical Records Server used with this tutorial.

The tutorial is designed to be followed sequentially from beginning to end. Many of the examples are interrelated and build upon the steps in previous examples.

3.1 Starting the Avitek Medical Records Sample Application

The tutorial uses the Avitek Medical Records Sample Application to record and playback OpenScript scripts. The WebLogic server and Avitek Medical Records Sample Application need to be started on the system before creating OpenScript script projects and recording scripts.

To start the Weblogic Server and Avitek Medical Records Sample Application:

1. Select **Programs** from the **Start** menu and then select **Oracle WebLogic** from the **Oracle Application Testing Suite** menu.
2. Select **Examples** from the **Weblogic Server 11gR1** submenu, then select **Start Medical Records Server**.
3. Wait until Avitek Medical Records Sample Application starts in the Weblogic server.

4. Close the browser window after the server and sample application are started. When recording OpenScript scripts, the browser window will automatically open again to specify the Web address to record.

3.2 Starting Oracle OpenScript

To start OpenScript:

- Select **Programs** from the **Start** menu, then select **Oracle OpenScript** from the **Oracle Application Testing Suite** menu to start OpenScript.

3.3 Example 1: Creating a Web Functional Test Script

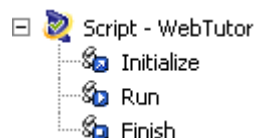
This example illustrates the creation of an OpenScript Web Functional test script project and recording a script. A script project creates the basic structure of an OpenScript script. Initially, the script project includes only the Initialize, Run, and Finish script nodes with the underlying Java code. You use the recording capabilities of OpenScript to record page navigations and generate the Java code for the script.

3.3.1 Creating a Web Functional Test Script Project

To create a Web Functional Test Script Project:

1. Select **New** from the **File** menu.
2. Select **Web** under the **Functional Testing (Browser/GUI Automation)** folder.
3. Click **Next**.
4. Make sure the Repository is set to Default.
5. Set the Workspace to RSWDemo.
6. Enter WebTutor as the script name.
7. Click **Finish**. OpenScript creates the script project and shows the Initialize, Run, and Finish nodes in the Script View as follows:

Figure 3–1 OpenScript Script Project Tree



3.3.2 Recording a Web Functional Test Script

Recording scripts captures the page navigations and generates the Java code that will be used to drive script playback for testing purposes. Web Functional test scripts record and playback actions performed on browser objects.

In this example, we will open the Avitek Medical Records sample application and log in as a patient. During recording, we will add a Text Matching test to verify that the log in was successful.

To record a Web Functional Test Script:

1. Select **Record** from the **Script** menu or click the Record toolbar button. OpenScript opens a browser window and the OpenScript toolbar window.

2. Enter `http://localhost:7011/medrec/` into the browser Address line and press **Enter**.
3. When the application loads, click **Start using MedRec!**. A second browser window opens with Avitek Medical Records Sample Application.
4. Click Login under the Patient section.
5. Enter `fred@golf.com` as the Email address.
6. Enter `weblogic` as the password.
7. Click **Submit**.
8. Click the Text Matching Test button on the OpenScript floating toolbar.
9. Enter `WebText1` as the Test name.
10. Enter `Successfully logged in! Click here to continue` as the Text to Match.
11. Make sure Source is set to **HTML Display Contents**.
12. Make sure Pass when is set to **Pass if present**.
13. Make sure Match is set to **Exact**.
14. Make sure **Verify only, never fail** is selected.
15. Click **OK**.
16. Click Successfully logged in! Click here to continue.
17. Click Logout in the Avitek Medical Records application.
18. Close the Avitek Medical Records application browser window.
19. Close the Weblogic Server Avitek Medical Records Sample Application browser window.

Recording automatically stops when you close the second browser window. The script tree view includes plus icons to indicate additional nodes have been added to the script as follows:

Figure 3–2 OpenScript Script Tree After Recording



20. Save the script.

3.4 Example 2: Working with Scripts

This example explains the features of the OpenScript Script tree and how to examine the structure and content of a recorded script.

After recording a script, you can expand the Tree View to view the page navigations, actions, and parameters recorded to the script. Before starting this example, make sure the script that you recorded in Example 1 is still displayed.

3.4.1 Viewing Information About Script Items

To view information about script items:

1. Expand the Initialize section to expand the tree as follows:

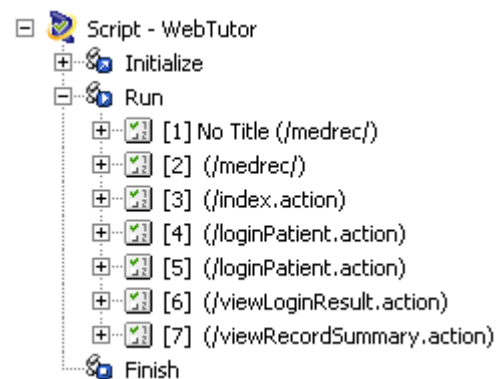
Figure 3–3 Script Tree with Expanded Initialize Section



The Initialize section executes commands that will be run only once at the beginning of script playback. In our recorded script, Launch Browser is the only action in the Initialize section.

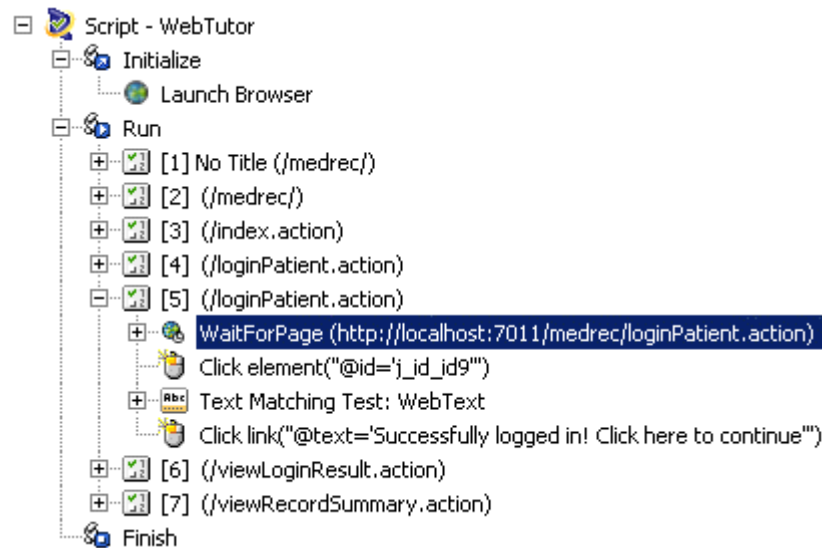
2. Expand the Run section to expand the tree as follows:

Figure 3–4 Script Tree with Expanded Run Section



The Run node contains the navigations recorded to the script as Step Groups. By default, OpenScript Web Functional test scripts create Step Group nodes based on web page navigations. Step Group preferences can be changed in the OpenScript Preferences.

3. Expand the [5] (/loginPatient.action) Step Group and select the WaitForPage (<http://localhost:7011/medrec/loginPatient.action>) node as follows:

Figure 3–5 Script Tree with Expanded Step Group Node

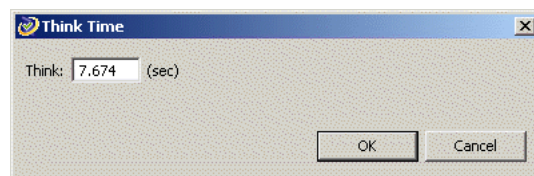
The Step Group node shows the actions and parameters recorded for specific navigations and actions on a Web page.

The Details View tabs show the details for a specific navigation including a screen shot of the recorded page, the HTML source, and the browser rendering of the page.

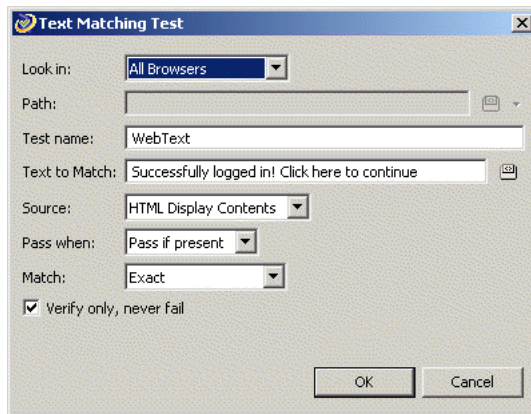
4. Click the Screenshot, HTML, and Browser tabs to view the script details.
5. Expand the WaitForPage (http://localhost:7011/medrec/loginPatient.action) node. The script records Think time nodes to include the amount of user delay that occurred during recording as follows:

Figure 3–6 Script Think Node

6. Select the Think node, click the right mouse button and select **Properties**. The Think time properties lets you edit the delay recorded to a script as follows:

Figure 3–7 Script Think Node Properties

7. Change the Think time value and click **OK**. This is useful if you want to reduce long a delay time that may have occurred during recording of script.
8. Right-click the Text Matching Test: Web Text node and select **Properties**. You can view or change the Text Matching test properties as follows:

Figure 3–8 Text Matching Test Properties

9. Click **Cancel** to leave the Text Matching test properties unchanged.
10. Open and view the properties of other script nodes.

3.4.2 Using the Java Code View

When you create a script project and record a script, the Java code is automatically generated. The Java Code view shows the script navigations and data as Java programming code. The Java Code view corresponds to the Tree View. Any changes in the Code View will be automatically updated in the Tree View. The Java Code view has the following standard procedures:

- `initialize()` - corresponds to the Initialize node of the Tree View and executes any custom code added once at the beginning of script playback.
- `run()` - corresponds to the Run node of the Tree View and executes recorded and custom code one or more times during script playback depending upon databanks or other custom programming.
- `finish()` - corresponds to the Finish node of the Tree View and executes any custom code added once at the end of script playback.

Use Ctrl-space to open an Intellisense window listing available procedures. See the API Reference in the OpenScript Platform Reference help for additional programming information.

3.5 Example 3: Playing Back a Web Functional Test Script

This example explains the procedure for playing back Web Functional Test scripts that you have recorded. It also shows the results log. Web Functional Test scripts perform actions on browser objects. During playback, the browser will open and you will be able to watch as script actions occur.

To play back an OpenScript script:

1. Select **Playback** from the **Script** menu or click the Playback toolbar button to play back the recorded script. The navigations and actions in the script Step Groups will be played back in the order recorded. The Browser navigates to each page, executes the any tests for each page, and shows the results in the Result view.
2. Expand the Results tree to view the results for each Step Group as follows:

Figure 3–9 Script Playback Results in the Results View

Name	Duration (sec)	Result	Summary
Results 10/22/09 01:18:55 PM	71.687	Passed	
Script WebTutor	71.687	Passed	
Initialize WebTutor	1.375	Passed	
Run WebTutor - Iteration 1	70.062	Passed	
[1] (/medrec/)	1.188	Passed	
[2] (/medrec/)	7.548	Passed	
[3] (/index.action)	14.331	Passed	
[4] (/loginPatient.action)	5.719	Passed	

The "passed" results indicate that all referenced resources are available. Oracle OpenScript automatically generates a results report and opens the report in the Details view.

3. Select the top-level Result node or the script name in the Results view to view the results report in the Details view as follows:

Figure 3–10 Results Report in the Details View

Report Type: Functional Test Report

Script Name: WebTutor

Report Generated By: oracle.oats.scripting.modules.functionalTest.api
 Script Name: WebTutor
 Workspace: Default
 Date Time: 10/22/2009 13:18:55 PM

Iterations: 1
 Total Pages: 7
 Total Tests: 1
 Total Failures: 0 (0.00%)
 Total Warnings: 0 (0.00%)
 Overall Result: **Passed**

Script Summary

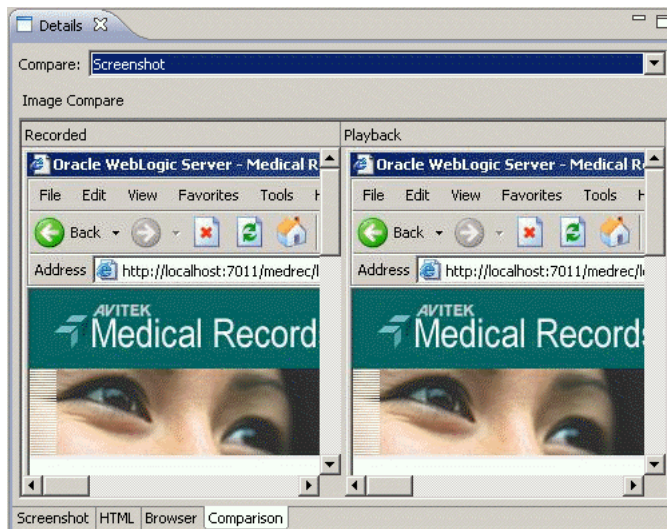
Section	Name	Duration (sec)	Result	Summary

Results Report

Notice that all tests passed. This is because you played back the script using the same version of the Web pages that was used to record the script. This establishes a baseline of tests for the Web application or Web site's content and structure.

4. Expand the Run node in the Results view then expand the results for Step Group [5] (/loginPatient.action).
5. Select the WaitForPage node under [5] (/loginPatient.action). The Details view shows the results for the specific Step Group.
6. Click the Comparison tab in the Details view. The Comparison tab lets you compare the recorded HTML content, screenshots, and browser renderings of page navigations to the play back values as follows:

Figure 3–11 Details View Showing the Web Functional Test Comparison Tab



7. Select the Content and Browser options in the **Compare** list to compare the recorded and playback HTML source and browser renderings of the page.

3.6 Example 4: Adding Tests to the Script

This example explains how to add tests to your OpenScript scripts. Make sure the script you recorded in the previous example is open in OpenScript.

OpenScript provides the ability to add the following test types to the pages in your Web Functional test script:

- Text Matching Test
- Server Response Test
- Object Test
- Table Test

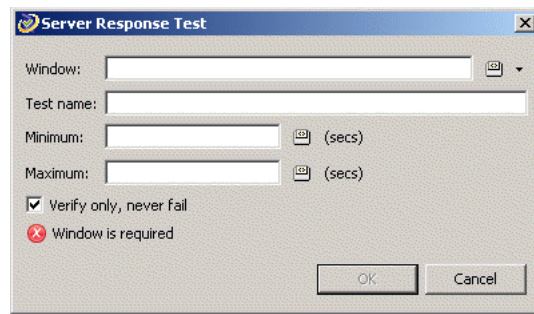
Other OpenScript test modules provide test types specific to the type of module.

3.6.1 Inserting a Server Response Test Case

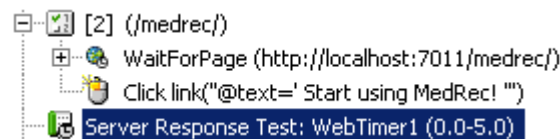
Server Response test cases measure the response time of a server access for a page in the script.

To add a Server Response test:

1. Select the [2] (/medrec/) Step Group in the script tree.
2. Select **Add** from the **Script** menu, then select **Other**.
3. Expand the Web Tests folder, select **Server Response Test**, and click **OK**.
OpenScript opens the Server Response Test properties dialog box as follows:

Figure 3–12 Server Response Test Properties Dialog Box

4. Click the down-arrow next to the **Window** field and select **Capture Object** from the menu. OpenScript starts the capture mode and opens a new browser window.
5. Load the Medical Records Sample Application (<http://localhost:7011/medrec/>) into the browser.
6. Move the mouse over the page until the page is highlighted.
7. Press F10 to capture the object path in the Select Object dialog box.
8. Click **OK**. The object path for the Medical Records Sample Application page is added to the Server Response Test properties.
9. Type WebTimer1 as the test case name.
10. Set the **Minimum** time to 0 seconds.
11. Set the **Maximum** time to 5 seconds.
12. Click **OK** and view the test in the script. OpenScript adds the test to the script [2] (/medrec/) Step Group as follows:

Figure 3–13 Server Response Test Added to the Script Tree

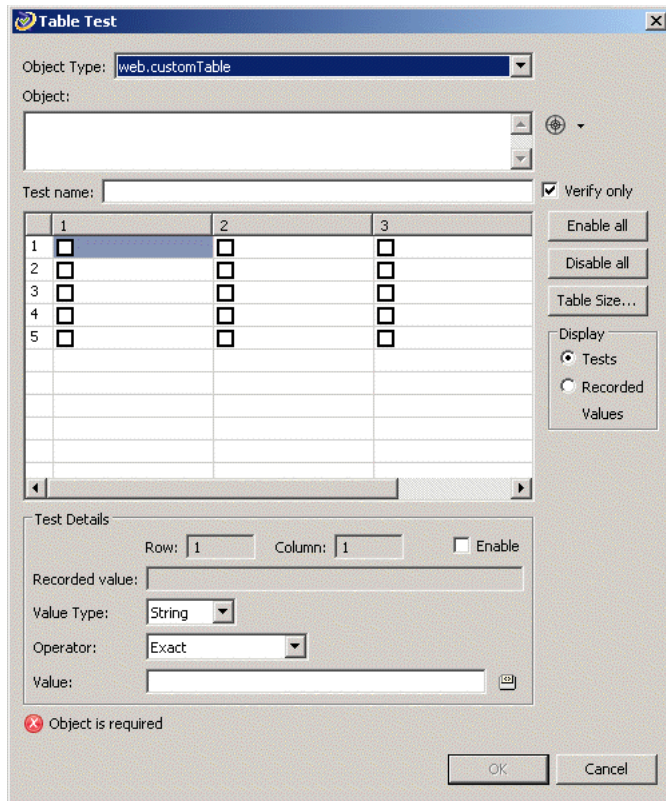
3.6.2 Inserting a Table Test

Table test cases let you define a custom test on a Web page table object. The Table Test properties lets you select the table object directly from the Web page by highlighting it with the mouse. The properties also let you specify the table object property to test and the type of test to perform.

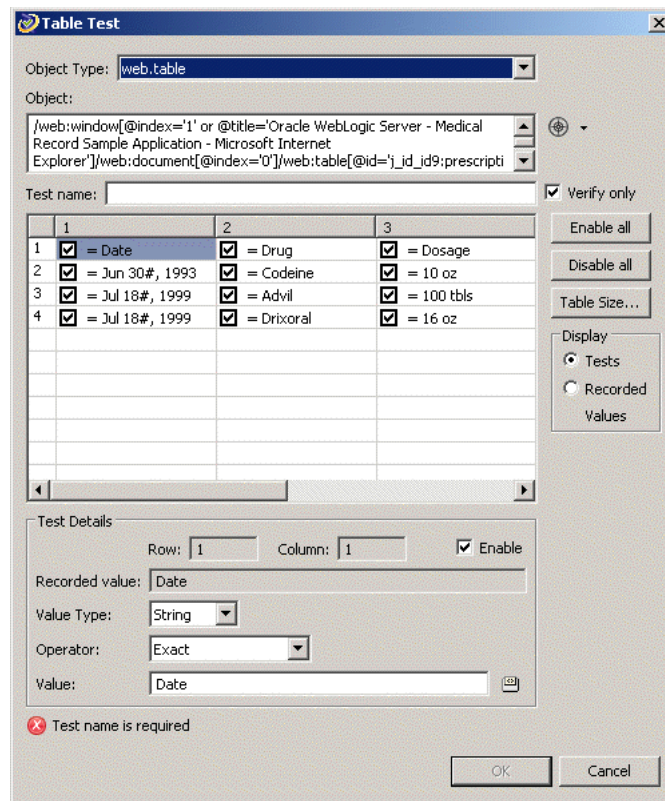
To add a Table test:

1. Select the [6] (/viewLoginResult.action) Step Group in the script tree.
2. Select **Add** from the **Script** menu, then select **Other**.
3. Expand the Web Tests folder, select **Table Test**, and click **OK**. OpenScript opens the Table Test properties dialog box as follows:

Figure 3–14 Table Test Properties Dialog Box

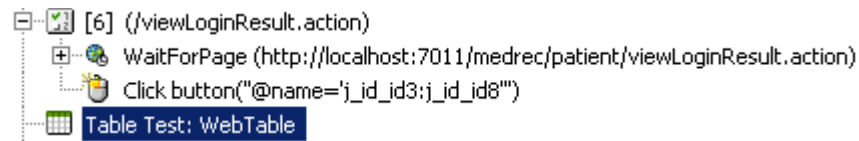


4. Click the down-arrow next to the **Window** field and select **Capture Object** from the menu. The capture mode starts. The
5. Switch to the Medical Records Sample Application (<http://localhost:7011/medrec/>) in the browser window (which should still be open from the previous example).
6. Click **Start using MedRec!** to navigate to the page with the table data.
7. Click Login under the Patient section.
8. Enter `fred@golf.com` as the Email address.
9. Enter `weblogic` as the password.
10. Click **Submit**.
11. Click [Successfully logged in! Click here to continue.](#)
12. Move the mouse over the Prescriptions table so that the table is highlighted.
13. Press F10 to capture the object path in the Select Object dialog box.
14. Click **OK**. The object path for the Medical Records Sample Application web page table is added to the Table Test properties as follows:

Figure 3–15 Table Test Properties Dialog Box with Captured Data

The Table Test properties lets you enable or disable testing for each table cell individually. The Test Details section shows the details for the currently selected table cell.

15. Clear the check boxes for all of the data items in rows 2, 3, and 4 (scroll the table as necessary). This will perform the testing on heading rows but not the data rows.
16. Enter WebTable as the Test name.
17. Click **OK** and view the test in the script. OpenScript adds the test to the script [6] (/viewLoginResult.action) Step Group as follows:

Figure 3–16 Table Test Added to the Script Tree

18. Save the script.
19. Close the Medical Records Sample Application browser windows.
20. Playback the script and verify that all tests pass.
21. Close the WebTutor script when finished.

3.7 Example 5: Creating an HTTP Test Script with Databanks

This example explains how to create an HTTP test script and use a databank to drive testing. HTTP test scripts record and playback navigations performed using the HTTP protocol. HTTP script are typically used for performing load tests on an application.

This example also explains one way to use the Databanks with the Text Matching test case to verify Login results pages. The Databanks provides the capability to run iterative tests using data from a Databank file.

3.7.1 Creating an HTTP Script Project

To create an HTTP script project:

1. Select **New** from the **File** menu.
2. Select **Web/HTTP** under the **Load Testing (Protocol Automation)** folder.
3. Click **Next**.
4. Make sure the Repository is set to Default.
5. Set the Workspace to RSWDemo.
6. Enter HTTPtutor as the script name.
7. Click **Finish**. OpenScript creates the script project and shows the Initialize, Run, and Finish nodes in the Script view.

3.7.2 Recording an HTTP Script

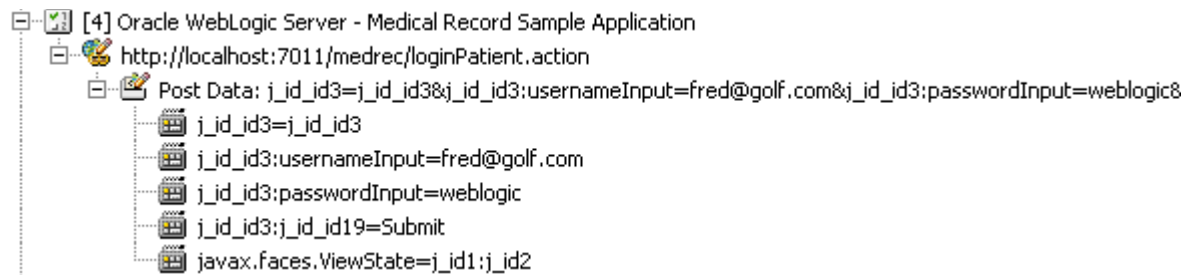
To record an HTTP script:

1. Select **Record** from the **Script** menu or click the Record toolbar button. OpenScript opens a browser window and the OpenScript toolbar window.
2. Reload the Medical Records Sample Application (<http://localhost:7011/medrec/>) in the browser window.
3. When the application loads, click **Start using MedRec!**. A second browser window opens with Avitek Medical Records Sample Application.
4. Click Login under the Patient section.
5. Enter `fred@golf.com` as the Email address.
6. Enter `weblogic` as the password.
7. Click **Submit**.
8. Click [Successfully logged in! Click here to continue.](#)
9. Click Logout in the Avitek Medical Records application.
10. Close the Avitek Medical Records application browser window.
11. Close the Weblogic Server Avitek Medical Records Sample Application browser window. Recording automatically stops.

3.7.3 Viewing the Parameters in the Script

To view the script parameters:

1. Expand the [4] Oracle WebLogic Server - Medical Record Sample Application Step Group in the script tree. Notice the parameters under the Post Data node of the tree as follows:

Figure 3–17 Script Parameter Values for HTTP Post Data

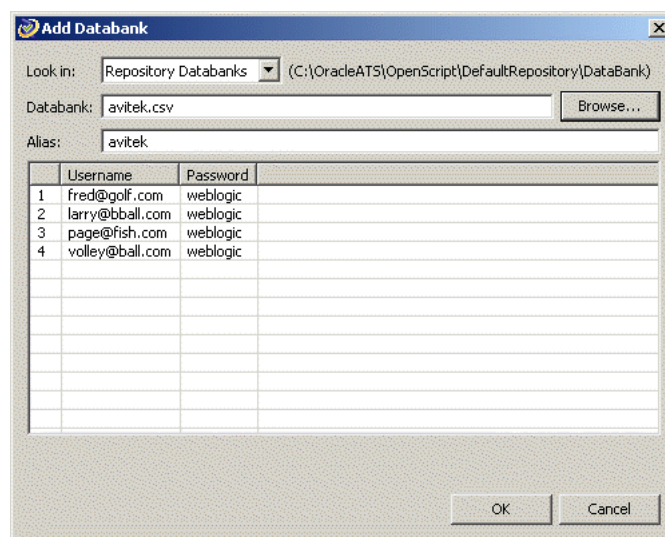
The usernameInput and passwordInput parameters can be mapped to a databank file to pass data from an external file.

3.7.4 Configuring Databanks with OpenScript Scripts

Before you can map parameters to databank, you must configure the script with the databank file.

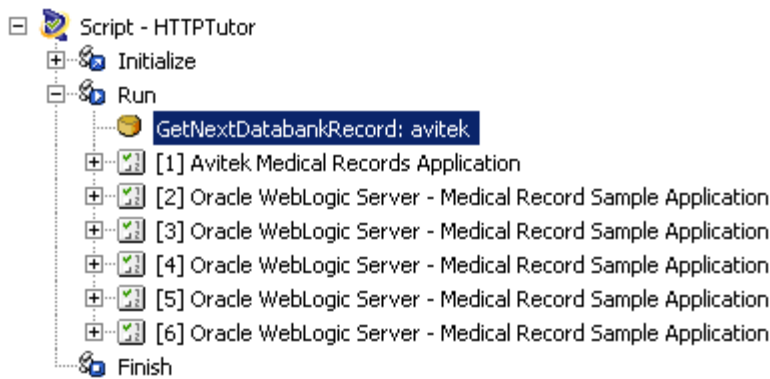
To configure a databank in OpenScript scripts:

1. Select **Configure Databanks** from the **Script** menu. The Databank properties window opens.
2. Click **Add**. The Add Databank dialog box opens for specifying a databank file.
3. Make sure **Repository Databanks** is selected as the **Look in** option.
4. Click **Browse**.
5. Select the avitek.csv file and click **Open**. The data from the databank file appears as follows:

Figure 3–18 Add Databank Dialog Box with Databank Selected

6. Click **OK** to close the Add Databank dialog box.
7. Click **OK** to close the databanks properties. The databank node appears at the top of the run section of the script tree as follows:

Figure 3–19 Script Tree with a Databank



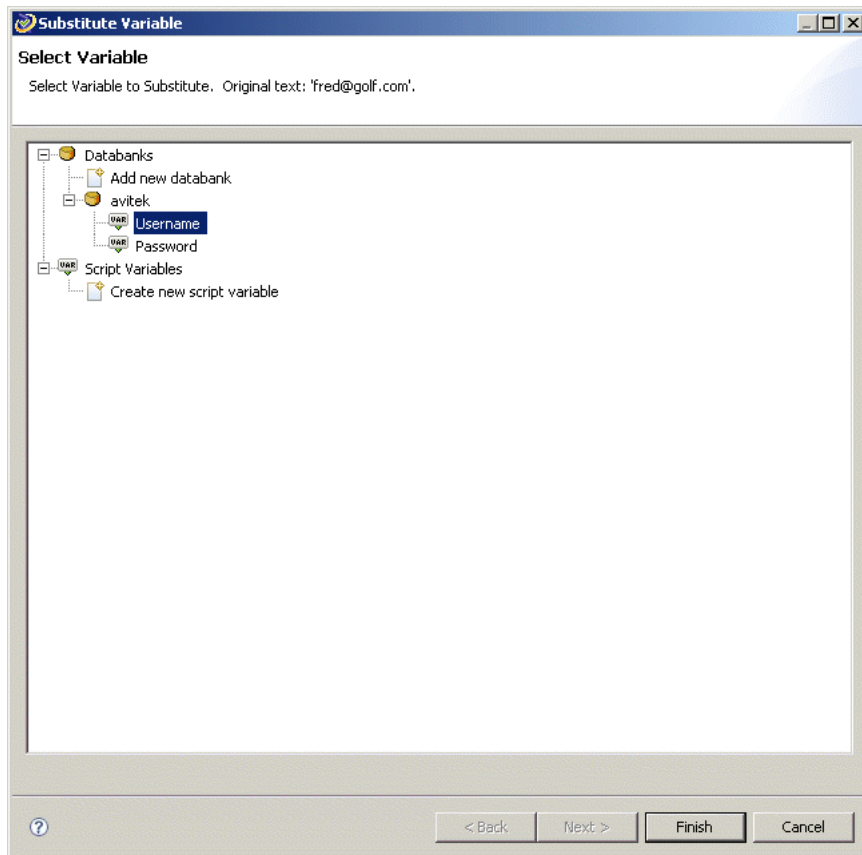
3.7.5 Mapping Script Parameters to Databank Fields

After configuring a databank in a script, you can map the databank fields to specific script parameters.

To map databank fields to script parameters:

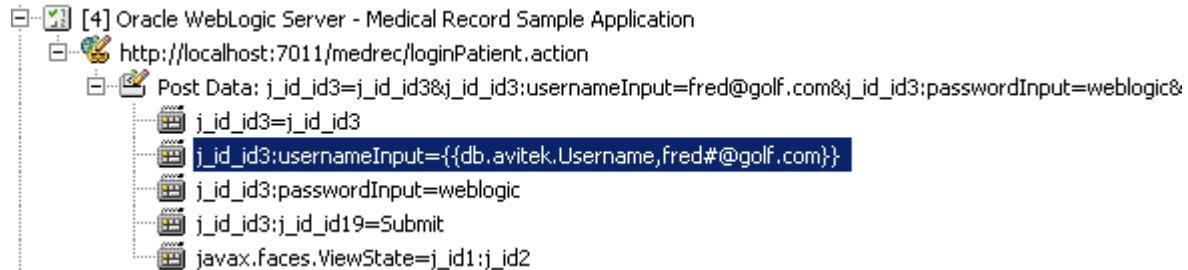
1. Expand the [4] Oracle WebLogic Server - Medical Record Sample Application script tree node.
2. Right-click the usernameInput parameter and select **Substitute Variable**. The Substitute Variable window opens with the databank field names listed as follows:

Figure 3–20 Substitute Variable Window



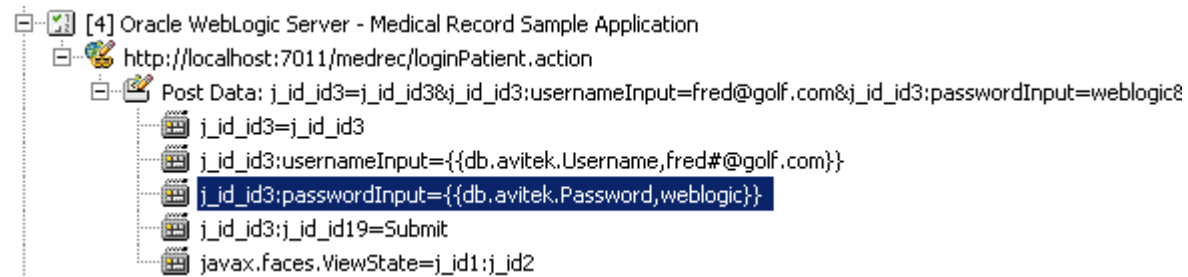
3. Select the Username field and click **Finish**. The parameter value changes to a databank variable in double braces `{{db.avitek.Username,fred#@golf.com}}` as follows:

Figure 3–21 Script Parameter with a Mapped Databank Variable



4. Right-click the passwordInput parameter and select **Substitute Variable**. The Substitute Variable window opens with the databank field names listed.
5. Select the Password field and click **Finish**. The parameter value changes to a databank variable in double braces `{{db.avitek.Password,weblogic}}` as follows:

Figure 3–22 Script Parameters with Multiple Mapped Databank Variables



6. Save the script.

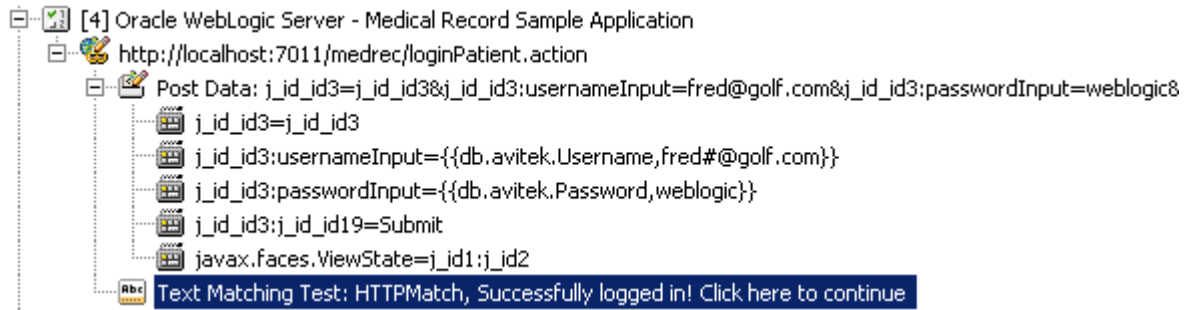
3.7.6 Inserting a Text Matching Test

Now we want to insert a Text Matching text case that verifies that the login results were successful.

1. Select the `http://localhost:7011/medrec/loginPatient.action` node under the [4] Oracle WebLogic Server - Medical Record Sample Application script tree node. The page appears in the Details view.
2. Select **Add** from the **Script** menu then select **Other**.
3. Select **Text Matching Test** in the HTTP Tests folder and click **OK**.
4. Enter HTTPMatch as the test name.
5. Enter Successfully logged in! Click here to continue as the Text to Match.
6. Make sure the **Source** options is **HTML Display Contents**.
7. Make sure the **Pass When** option is **Pass if Present**.
8. Make sure the **Match** option is **Exact**.

- Click **OK**. The Text Matching test node appears in the script tree at the end of the Step Group as follows:

Figure 3–23 Script Tree with a Text Matching Test Node



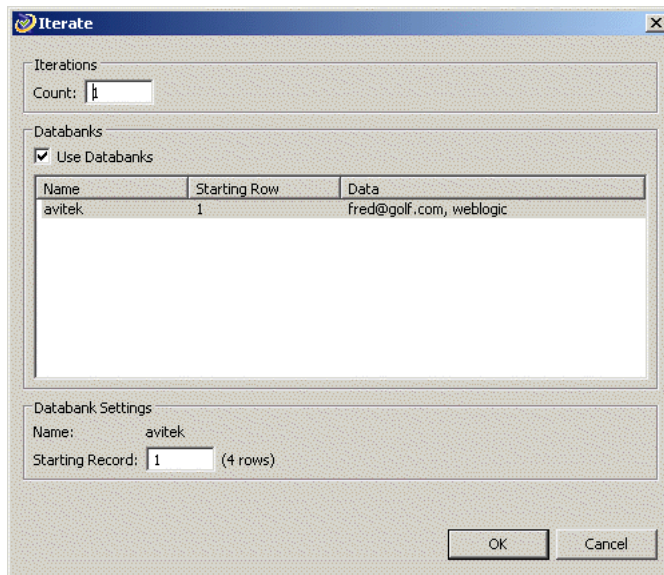
- Save the script.

3.7.7 Playing Back the Script with Iterations

Now that the script has a databank mapped to the Post Data parameters, we can play back the script with multiple iterations to use all of the values in the databank file.

- Select **Iterate** from the **Script** menu. The Iterations dialog box opens as follows:

Figure 3–24 Iterations Dialog Box



- Make sure the **Count** is set to 4 as there are four records in the sample databank file.
- Make sure **Use Databanks** is selected.
- Click the **OK** button to playback the script with multiple iterations.
- Watch the Details view as the script plays back the script several times using a different data value for the login each time.
- View the playback results in the Results view. At the end of playback the Details view shows the results report as follows:

Figure 3–25 Results Report for Multiple Iterations

Report Type: Http Results Report

Script Name: HTTPTutor

Report Generated By: oracle.oats.scripting.modules.http.api
 Script Name: HTTPTutor
 Workspace: Default
 Date Time: 10/28/2009 15:04:45 PM

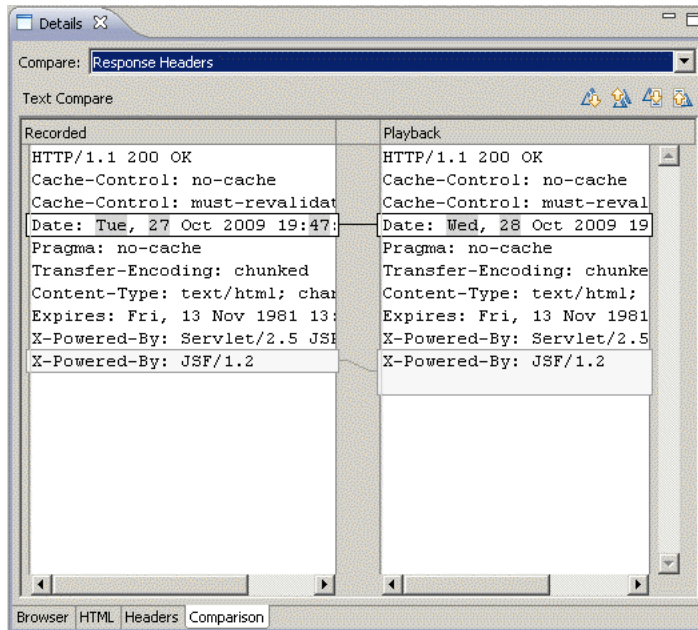
Iterations: 4
 Total Pages: 24
 Total Tests: 4
 Total Failures: 0 (0.00%)
 Total Warnings: 0 (0.00%)
 Overall Result: **Passed**

Script Summary

Section	Page	Recorded Time (sec)	Playback Time (sec)	Result	Summary
+	Initialize	Initialize Total (sec)	0.000	0.032	Passed
+	Iteration1	Iteration Total (sec)	4.665	36.787	Passed
+	Iteration2	Iteration Total (sec)	4.665	36.073	Passed
+	Iteration3	Iteration Total (sec)	4.665	36.671	Passed
+	Iteration4	Iteration Total (sec)	4.665	36.278	Passed
+	Finish	Finish Total (sec)	0.000	0.000	Passed
		Script Total (sec)	9.330	149.021	Passed

Results Report

7. Scroll the Results view and select one of the HTTP navigation nodes such as `http://localhost:7011/medrec/index.action`.
8. Click the Comparison tab in the Details view. The Comparison tab for HTTP scripts lets you compare the recorded HTML content, Request Headers, Response Headers, and Cookies to the playback values as follows:

Figure 3–26 Details View Showing the HTTP Test Comparison Tab

9. This completes the OpenScript tutorial. Save the script and close the OpenScript application.

3.8 Stopping the Avitek Medical Records Server

When finished with the tutorial be sure to stop the sample Medical Records Server. If you want to try using scripts recorded against the sample Medical Records Server application in Oracle Load Testing for Web Applications or Oracle Test Manager for Web Applications tutorials, leave the Avitek Medical Records Server running until after completing those trials.

To stop the Weblogic Server and Avitek Medical Records Server:

1. Select **Programs** from the **Start** menu and then select **Oracle WebLogic** from the **Oracle Application Testing Suite** menu.
2. Select **Examples** from **Weblogic Server 11gR1** submenu, then select **Stop Medical Records Server**.
3. Wait until the Weblogic server stops.
4. Close the command windows after the server stops.

Oracle Functional Testing for Web Applications Tutorial

This tutorial walks you through the main features of Oracle Functional Testing for Web Applications. The tutorial consists of the following examples:

- **Recording a New Visual Script** - describes basic recording of Visual Scripts.
- **Working with Visual Scripts** - describes the features and components of Visual Scripts and how to modify the default tests.
- **Playing Back a Visual Script** - explains the procedure for playing back Visual Scripts and the option settings for playback and the results log.
- **Analyzing Test Failures** - explains how to analyze the differences found between the baseline set of Web pages and a new version.
- **Adding Test Cases to the Visual Script** - explains how to add test cases to your Visual Scripts.
- **Using the Data Bank Wizard on a Search Form** - introduces the Data Bank Wizard and explains how to use the Data Bank Wizard to run iterative tests on a search form using data from an external file.
- **Using the Data Bank Wizard on a Registration Form** - explains how to use the Data Bank Wizard to create automated data-driven tests.

The tutorial is designed to be followed sequentially from beginning to end. Many of the examples are interrelated and build upon the steps in previous examples.

4.1 Initializing the Tutorial

The tutorial uses two versions of web pages to demonstrate the capabilities of Oracle Functional Testing for Web Applications. To make sure the initial version of the tutorial web pages is the current version, do the following:

1. Select **Programs** from the **Start** menu and then select **Build A - Home Superstores** from the **Oracle Application Testing Suite** submenu. A command window will appear briefly as the batch file copies the Build A files.
2. Close the command window, if necessary.

4.2 Example 1: Recording a New Visual Script

This example illustrates the creation and recording of a Visual Script.

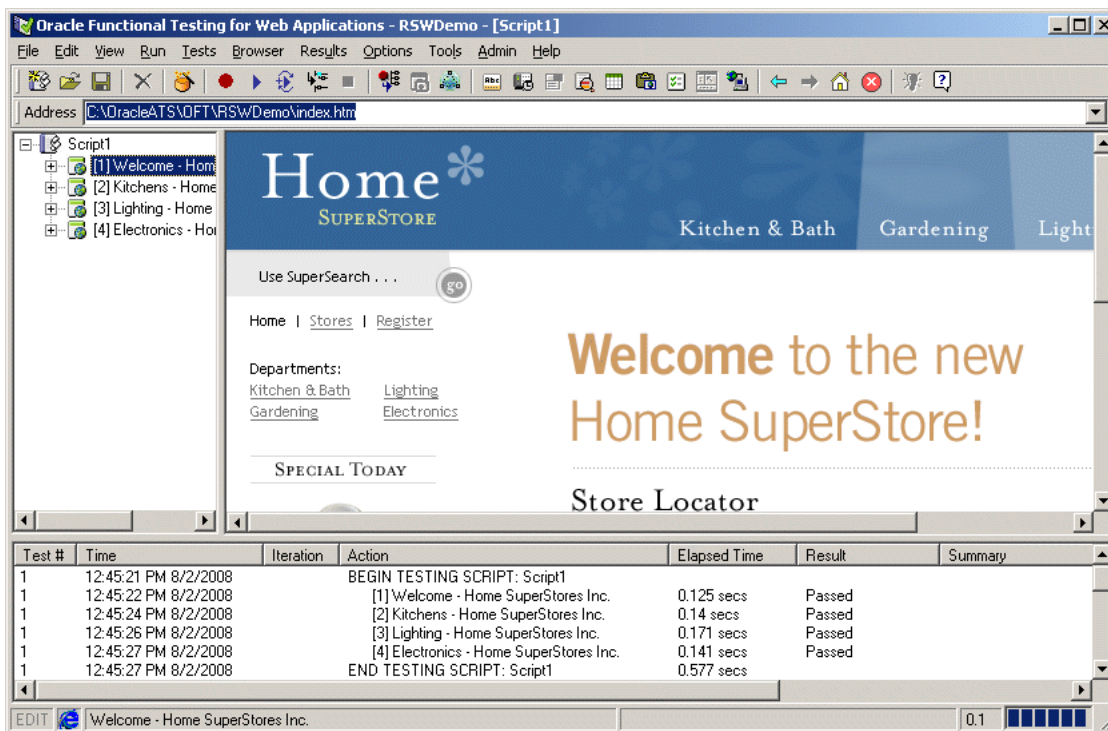
4.2.1 Start Oracle Functional Testing for Web Applications

1. Select **Programs** from the **Start** menu and then select **Oracle Functional Testing for Web Applications** from the **Oracle Application Testing Suite** submenu to start Oracle Functional Testing for Web Applications.
2. Select **Open Workspace** from the **File** menu, select RSWDemo as the Workspace, and click **OK** to get to the main window.

Note: If you installed this version of the Oracle Application Testing Suite over a previous version, your default installation directory will still be the old directory. If this version is the first time you installed the Oracle Application Testing Suite, the default installation directory is c:\OracleATS\OFT.

3. Type c:\OracleATS\OFT\rswdemo\index.htm in the URL drop down list and then press ENTER. (The tutorial assumes that you installed Oracle Functional Testing for Web Applications in the default c:\OracleATS\OFT directory. If you installed to another directory, enter the appropriate path.)

Figure 4–1 Oracle Functional Testing for Web Applications Main Window



Oracle Functional Testing for Web Applications opens the Home Superstores tutorial Web page into the Browser pane.

4.2.2 Start a Recording

4. Select **New Script** from the **File** menu and select **No** if asked to save changes to Script1.
5. Click the Record button on the toolbar. Oracle Functional Testing for Web Applications is now recording your actions as indicated by the REC in the status

bar. The [1] Welcome - Home Superstores Inc. title is recorded into the Visual Script pane.

4.2.3 Navigate the Web Site

6. Click on the Kitchen and Bath link on this page. The Kitchen and Bath page appears in the Browser pane and the address should show c:\OracleATS\OFT\rswdemo\kitchen.htm. You should now see [2] Kitchens - Home Superstores Inc. in the Visual Script.
7. Click on the Lighting link on this page. The Lightings page appears in the Browser pane and the address should show c:\OracleATS\OFT\rswdemo\lighting.htm. You should now see [3] Lightings - Home Superstores Inc. in the Visual Script.
8. Click on the Electronics link on this page. The Home Electronics page appears in the Browser pane and the address should show c:\OracleATS\OFT\rswdemo\etronics.htm. You should now see [4] Electronics - Home Superstores Inc. in the Visual Script.

4.2.4 Stop the Recording

9. Click the Stop button on the toolbar to stop the recording. The Visual Script pane should list four pages in the script.

4.2.5 Save the Script

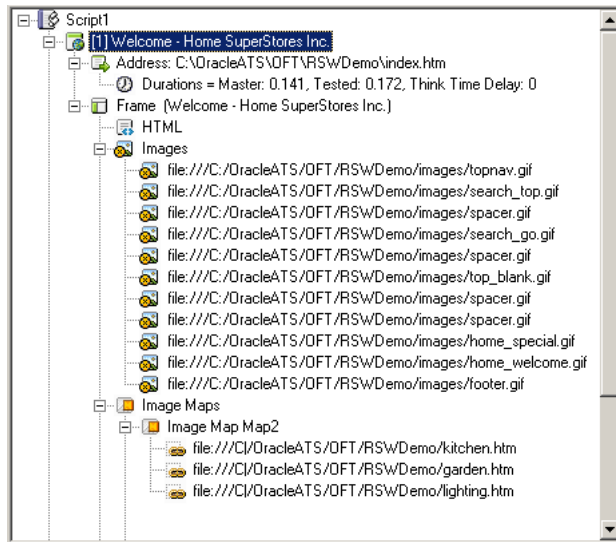
10. Select **Save Script** from the **File** menu to save the script. The autonaming feature initially defaults the name of a new script to Script1.
11. Type in tutor1 for the name of the script in the Save As dialog box and click **Save**.

4.3 Example 2: Working with Visual Scripts

This example explains the features of the Visual Script tree and how to examine the structure and content of a Web page. It also explains how to modify the built-in Oracle Functional Testing for Web Applications tests.

Before starting this example, make sure the Visual Script that you recorded in Example 1 is still displayed.

1. Select **Resize Visual Script View** from the **View** menu to expand the Visual Script pane.
2. Click the [1] Welcome - Home Superstores Inc. node in the Visual Script and then click the Plus next to the node. The script shows the Address and Frame nodes to the page tree.
3. Click the right mouse button and select **Expand Page** to show the entire page tree, which should look similar to the following:

Figure 4–2 Oracle Functional Testing Visual Script Tree

The [1] Welcome - Home Superstores Inc. page contains the following nodes in the tree:

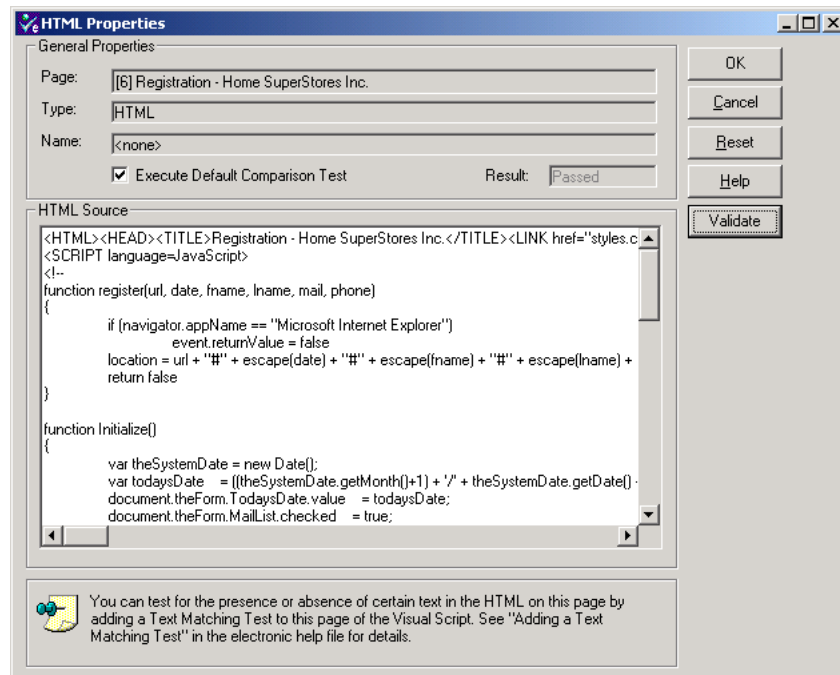
- An Address node that has the recorded URL for the page.
- A Duration node under the Address node that displays how long it took to download the page and the think time delay associated with the page. The think time delay is the actual amount of time the user spent on the page before going to another page. For additional information about think time delay, see the online help.
- A Frame node that is the main frame for the entire page. Below this node are the other constituents of the page.
- An HTML node that has the HTML source for the page.
- An Images node that has all the images in the page under it.
- A Image Maps node that has all image maps in the page under it.
- A Scripts node that has all the VBScripts and JavaScripts under it.
- A Links node that has all the links in the page under it.

Note: Web pages that include Frame Sets, Anchors, Forms, Elements, Active X objects, Java Applets, etc. will have additional tree nodes displayed in a similar fashion.

4.3.1 Viewing Information About a Visual Script Item

4. Select the HTML node in the Visual Script.
5. Click the right mouse button and select **HTML Properties**. The following dialog box is displayed:

Figure 4–3 HTML Properties Dialog Box



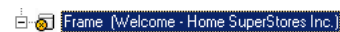
You can get more information about any item in the Visual Script using the Properties option. The properties for the different items vary.

6. Close the Properties dialog box.
7. Repeat steps 5 and 6 with any other items in the tree and view the properties.

4.3.2 Turning Automatic Testing On and Off

8. Select the Frame node in the Visual Script.
9. Click the right mouse button and select **Don't Test Frame**. Notice a small yellow circle appears next to the Frame node to indicate that the automatic existence test for the frame is turned off, as shown below:

Figure 4–4 Visual Script Frame Node



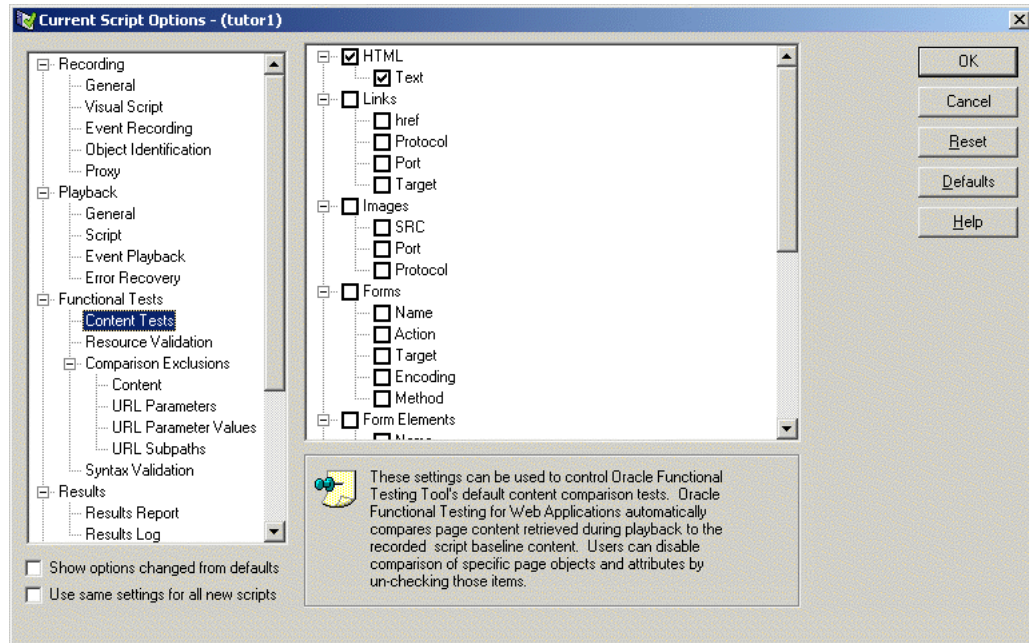
10. Click the right mouse button and select **Test Frame**. The yellow circle disappears to indicate that the automatic existence test is activated.

Note: Oracle Functional Testing for Web Applications maximizes your productivity by virtually eliminating the need to program test scripts. When a Visual Script is recorded, it captures your interaction with the Web application under test. A series of default test cases are automatically generated and added to the Visual Script. These tests are designed for Images, Links, Frames, Forms, Elements, HTML, Java Applets, ImageMaps, and Active-X controls and can be customized to suit your requirements.

4.3.3 Modifying Default Tests

11. Select **Current Script (tutor1)** from the **Options** menu and then select **Content Tests** in the **Functional Tests** section.

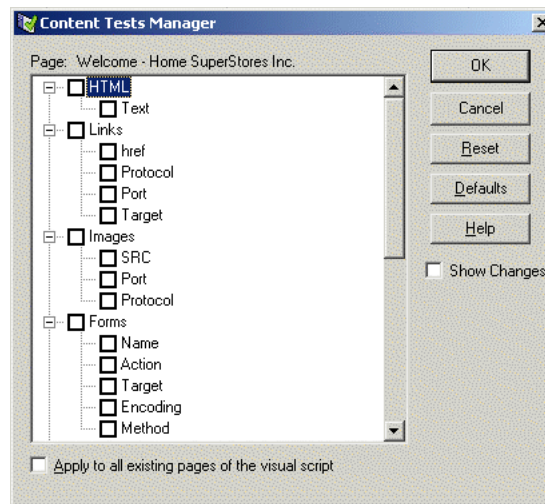
Figure 4–5 Script Options Dialog Box



This dialog box indicates which comparison tests will be performed on which page attributes as default tests for the current script. You can enable or disable testing of specific attributes by selecting or clearing the appropriate check box(es). You can set global defaults for all Visual Scripts by selecting **New Scripts (Global)** from the **Options** menu and then selecting **Content Tests** in the **Functional Tests** section.

12. Make sure the following check boxes are selected:
 - HTML.
 - Images.
 - Scripts.
 - Links.
 - Frames.
13. Click **OK**.
14. Select the [1] Welcome - Home Superstores Inc. node in the Visual Script.
15. Click the right mouse button and select **Page Content Tests Manager**. The Content Tests Manager dialog box opens.

Figure 4–6 Content Tests Manager Dialog Box



This dialog box indicates which comparison tests will be performed on which page attributes as default test cases for a specific page. You can enable or disable testing of specific page attributes by selecting or clearing the appropriate check box(es), and clicking the **OK** button.

16. Clear the Images check box, and press the **OK** button. A yellow circle appears next to the Images node in the Visual Script for the image collection, as shown below:

Figure 4–7 Images Script Node



When the Visual Script is played back, all Image tests will be ignored.

4.4 Example 3: Playing Back a Visual Script

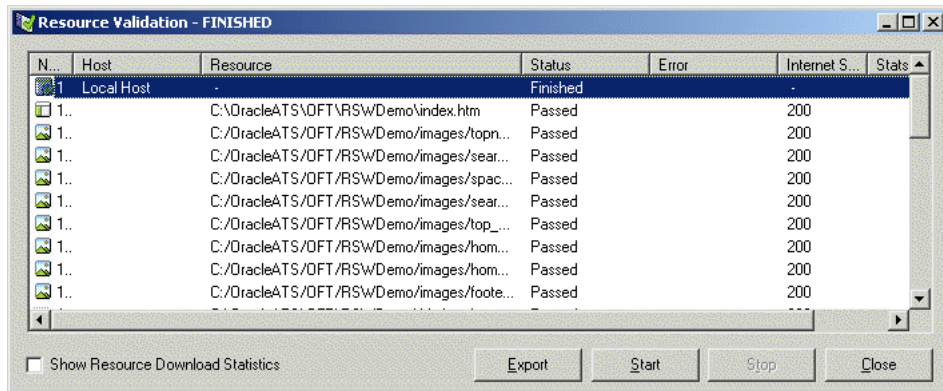
This example explains the procedure for playing back Visual Scripts that you have recorded. It also shows the option settings for playback and the results log.

1. Select **Current Script** from the **Options** menu and then select **Results Log** in the **Results** section. Make sure the **Append to log** and **All details** radio buttons are selected.
2. Select **Results Report** in the **Results** section and make sure the **Automatically create report after script playback** option is selected.
3. Select **General** in the **Playback** section, make sure the **Automatically Run Resource Validation After Playback** check box is selected, and then click **OK**.
4. Select **Resize Visual Script View** from the **View** menu to expand the Browser pane.

- Click the Playback Script button to play back the recorded script. The pages in the script will be played back in the order recorded. The Browser navigates to each page, executes the default tests for each page, and shows the results visually in the script. At the end of the play back, Oracle Functional Testing for Web Applications runs a Resource Validation test and shows the results.

The Resource Validation test checks the integrity of the referenced resources (i.e. links, images, etc.) in your pages.

Figure 4–8 Resource Validation Window



The "passed" results indicate that all referenced resources are available.

- Click the **Close** button to close the Resource Validation window.

Oracle Functional Testing for Web Applications automatically generates a results report and opens the report in a new browser window:

Figure 4–9 Results Report Window

Script Name: tutor1

Report Generated By: Oracle Functional Testing for Web Applications 8.40.90
Script Name: tutor1
Workspace: RSWDemo
Date & Time: 8/2/2008 1:25:53 PM

Iterations: 1
Total Pages: 4
Total Tests: 129
Total Failures: 0 (0.00%)
Total Warnings: 0 (0.00%)
Overall Result: ● Passed

Script Summary

Iteration	Page	Recorded Time (sec)	Playback Time (sec)	Result	Summary
1	Iteration Total (sec)	0.562	0.594	● Passed	
	[1] Welcome - Home SuperStores Inc.	0.172	0.250	● Passed	
	[2] Kitchens - Home SuperStores Inc.	0.187	0.141	● Passed	
	[3] Lighting - Home SuperStores Inc.	0.125	0.078	● Passed	
	[4] Electronics - Home SuperStores Inc.	0.078	0.125	● Passed	
	Script Total (sec)	0.562	0.594		

Script Details

Note: The Results Report uses active content. If the links in the report are not working, check if the browser shows the restricted active content security warning at the top of the browser. If so, click on the warning and select **Allow Blocked Content**. Use the browser Internet Options to set the default settings for allowing active content. Select **Browser** from the **Options** menu and click the **Advanced** tab. Select the **Allow active content to run in files...** setting under the Security section if you do not want the restricted active content security warning to appear each time the Results Report is generated.

7. Click on the page names in the Script Summary section to view the information for each page.

Notice that all tests passed. This is because you played back the script using the same version of the Web pages that was used to record the script. This establishes a baseline of tests for the Web application or Web site's content and structure.

8. Close the browser window when finished with the report.

The Results pane also shows a summary of the playback actions.

Figure 4–10 Playback Results Pane

Test #	Time	Iteration	Action	Elapsed Time	Result	Summary
1	11:52:37 AM 4/5/2004		[1] Welcome - Home SuperStores Inc.	0.28 secs	Passed	
1	11:52:39 AM 4/5/2004		[2] Kitchens - Home SuperStores Inc.	0.2 secs	Passed	
1	11:52:40 AM 4/5/2004		[3] Lighting - Home SuperStores Inc.	0.2 secs	Passed	
1	11:52:41 AM 4/5/2004		[4] Electronics - Home SuperStores Inc.	0.17 secs	Passed	
1	11:54:15 AM 4/5/2004		Resource Validation		Passed	
1	11:54:15 AM 4/5/2004		END TESTING SCRIPT: tutor1	0.85 secs		

In the next example, you'll see how playback and the results are affected by Web page changes.

4.5 Example 4: Analyzing Test Failures

This example explains how to analyze the differences found between the baseline Web pages and a new version with changes. The Oracle Functional Testing for Web Applications tutorial includes a batch file that copies a new version of three of the pages that you recorded in Example 1.

1. Select **Programs** from the **Start** menu and then select **Build B - Home Superstores** from the **Oracle Application Testing Suite** submenu. This runs a batch file that updates the Home Superstores page to a new version.
2. If necessary, close the command window after the batch file finishes copying the files.
3. Click the Playback Script button to play back the recorded script again. The pages are played back in the order recorded.

The resource validation test found one resource that failed. There is one failed image called moviei.gif. The results of the resource validation test are displayed in the output log window and the failure will be displayed under the appropriate pages with red flags.

4. Click the **Close** button to close the Resource Validation window and close the Results Report browser window.
5. Notice the red circles next to the pages in the Visual Script, as shown below:

Figure 4–11 Visual Script with Failure Markers



Note: Oracle Functional Testing for Web Applications displays errors and problems encountered during playback using simple color-coded circles in the script itself. Test results are displayed dynamically in the Visual Script tree as the script is being played back. You can double-click on any error event in the result log pane to advance to the corresponding page in the script. Errors encountered upon Visual Script playback can be rejected, ignored, or can be accepted to create a modified baseline script.

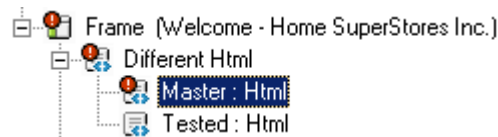
6. Select **Find Next Failure** from the **Results** menu, or press the F3 key to locate the next failure in the Visual Script. The first page [1] Welcome - Home Superstores Inc. has two places where differences are indicated: Different HTML and New Links with two new links under the Tested node. The new links are admlist.htm and adminfo.htm.

4.5.1 Ignoring Failures

Occasionally, you may want to ignore a known problem or discrepancy that does not affect the overall test being performed.

7. Locate the Master: Html node under the Different Html node.

Figure 4–12 Different Html Script Node



8. Click on the Different Html node, and then click the right mouse button and select **Ignore This Failure**. This adds a yellow circle to the Different Html node to indicate that the failure caused by the HTML change should be ignored, as follows:

Figure 4–13 Ignore This Failure Script Node Marker



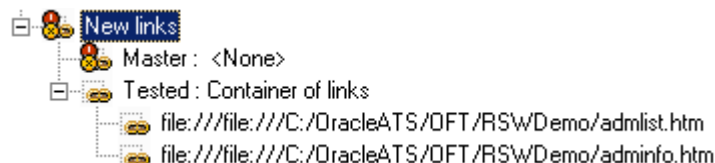
9. Press the F3 key to get to the New Links item and expand the Tested node to view the new links.

Figure 4–14 New Links Script Node



10. Click on the New Links item, and then click the right mouse button and select **Ignore This Failure**. This adds a yellow circle to the New Links node to indicate that the failure caused by the presence of new links should be ignored.

Figure 4–15 New Links Script Node with Marker



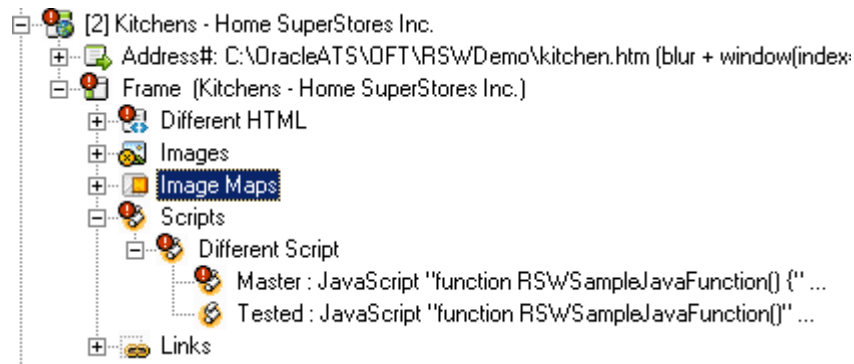
4.5.2 Accepting Changes Shown in the Script

Often, you will get new versions of Web pages that you want to use as the new baseline for testing.

11. Follow the red circles in the [2] Kitchens - Home Superstores Inc. page.

The problems on this page are indicated by the presence of two nodes called Different Html and Different Script. Below the Different Html node are the recorded and tested versions of the HTML for the current page. Below the Different Script node are the recorded and tested versions of the JavaScript function that has changed for the current page.

Figure 4–16 Different Html and Different Script Nodes



12. Double-click on the Different HTML node and a dialog box with the differences for the HTML opens.
13. Click **Next Difference** as many times as necessary to locate the following differences between Master and Tested page HTML source.

Figure 4–17 Master and Tested Page HTML Source Differences

```
<TD vAlign=top>Departments:<BR>Kitchen &amp; Bath<BR><A href="garden.htm">Gardening</A></TD>
<TD vAlign=top>Other Departments:<BR>Kitchen &amp; Bath<BR><A href="garden.htm">Gardening</A></TD>
```

Notice that the changes are textual changes to the Web page content. The Master text (baseline for testing) is shown in blue. The tested text (new version) is shown in red.

14. Click the Cancel button to close the window.
15. Repeat the same process for the Different Script node.
16. Select the [2] Kitchens - Home Superstores Inc. page node and select **Accept Tested Page** from the right-click shortcut menu. The red circles in the Visual Script for that page disappear. The accepted change becomes the new baseline for future testing of this page.

4.5.3 Rejecting Problems Shown in the Script

There may be times when you do not want a change to a Web page to be accepted as the new baseline for testing.

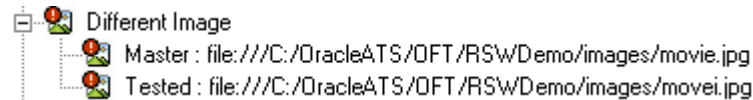
17. Follow the red circles in the [4] Electronics - Home Superstores Inc. page.

The problems on this page are indicated by the presence of two nodes called Different Html and Different Image. Below the Different Html node are the

recorded and tested versions of the HTML for the current page. Below the Different Image node are the recorded and tested versions of the image found on this page.

18. If necessary, expand the Different Image node to display the Master and Tested Image nodes. You will notice that the file name of the image file is spelled differently [movie vs. movei]

Figure 4–18 Master and Tested Image Differences



The Master node has a red circle because playback revealed that the link to the image is no longer in the HTML. The Tested node has a red flag because the image failed the Resource Validation test.

19. Double click on the Different HTML node and you will see the same spelling change in the HTML source for the page.

Figure 4–19 Master and Tested Image Differences

```
<TD><IMG height=140 src="images/movie.jpg" width=164></TD>
<TD><IMG height=140 src="images/movei.jpg" width=164></TD>
```

We wish to discard these differences and continue to use the original spelling as the baseline for testing in the Visual Script.

20. Click the **Cancel** button.
21. Select the [4] Electronics - Home Superstores Inc. page node and select **Discard Tested Page** from the right-click shortcut menu. This causes the originally recorded baseline to be left intact. The baseline differences will be discarded and all red circles will be removed.
22. Click the Playback toolbar button to play back the script again. Oracle Functional Testing for Web Applications still flags the Different HTML and the image name spelling problem in the [4] Electronics - Home Superstores Inc. page of the Visual Script.
23. Close the Resource Validation window and the Results Report.
24. Select **Save Output Log As** from the **File** menu.
25. Enter the name tutor1.log and click Save.
26. Select **Save Script** from the **File** menu to save the changed Visual Script.
27. Select **Clear Results Window** from the **View** menu to clear the results log pane.

4.6 Example 5: Adding Test Cases to the Visual Script

This example explains how to add four types of test cases to your Visual Scripts. In addition to the automatic existence and resource validation tests, Oracle Functional Testing for Web Applications provides the ability to add the following test cases to the pages in your Visual Script:

- Text Matching
- Server Response

- Form Element
- Table Test
- WinForms Test
- Siebel Test

4.6.1 Record a New Script

1. Select **Programs** from the **Start** menu and then select **Build A - Home Superstores** from the **Oracle Application Testing Suite** submenu. This batch file restores the original Web pages for the Home Superstores site.
2. If necessary, close the command window.
3. Select **New Script** from the **File** menu to create a new Visual Script.
4. Reload the c:\OracleATS\OFT\rswdemo\index.htm page in the Browser pane by selecting it from the Browser drop-down list.
5. Click the Record button on the toolbar.
6. Click on the Register link in the Browser pane. The Registration page appears in the Browser pane and the address should show c:\OracleATS\OFT\rswdemo\register.htm.
7. Type Admin as the first name, enter any email address, and phone in the text area, and click the **submit my entry** button. The Browser returns the Database Authorization and Administration options of the registration page (regres.htm).

4.6.2 Stop the Recording

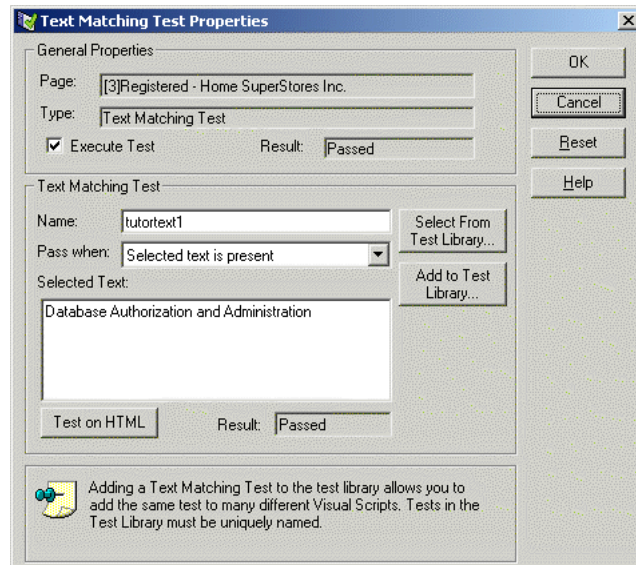
8. Click the Stop button on the toolbar to stop the recording. The Visual Script pane should list three pages in the script.

4.6.3 Insert a Text Matching Test Case

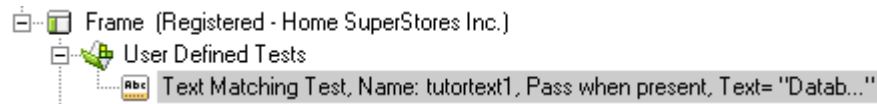
Text Matching test cases compare selected text from a Web page to the text you specify in the test case.

1. Select [3] Registered - Home Superstores, Inc. in the Visual Script.
2. Click the right mouse button and select **Goto Page** to open the Registered - Home Superstores, Inc page in the Browser.
3. Scroll the Browser pane so that the text "Database Authorization and Administration" is visible.
4. Highlight the "Database Authorization and Administration" text with the mouse.
5. Click the Insert Text Matching Test Case button on the toolbar.

Oracle Functional Testing for Web Applications captures the highlighted text and opens the Insert Text Matching Test Case dialog box.

Figure 4–20 Text Matching Test Properties Dialog Box

6. Type tutortext1 as the test case name.
7. Make sure the **Pass when:** option is set to **Selected text is present**.
8. Click **OK** and view the test case in the Visual Script. Oracle Functional Testing for Web Applications adds the test case to the Visual Script under the Frame node.

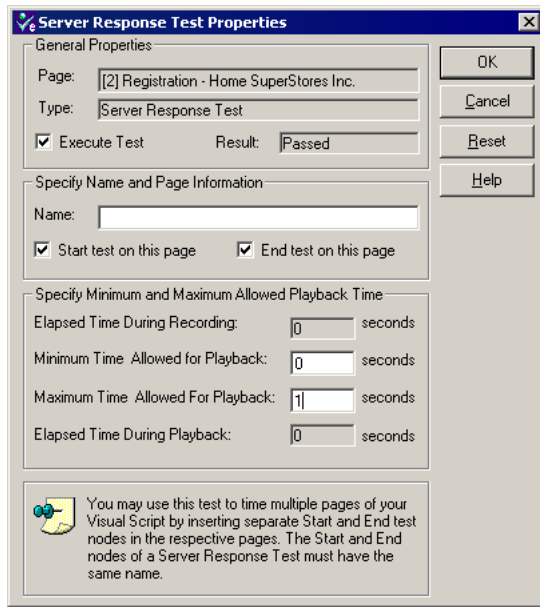
Figure 4–21 Text Matching Test Script Node

4.6.4 Insert a Server Response Test Case

Server Response test cases measure the response time of a server access for a page in the Visual Script.

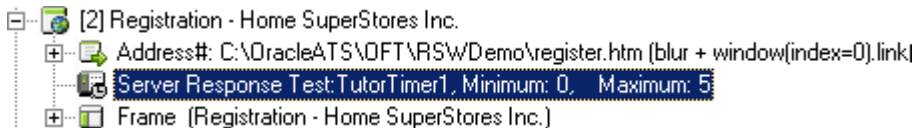
9. Select the [2] Registration - Home Superstores, Inc. item in the Visual Script.
10. Click the Insert Server Response test case button on the toolbar. Oracle Functional Testing for Web Applications opens the Insert Server Response Test Case dialog box.

Figure 4–22 Server Response Test Properties Dialog Box



11. Type TutorTimer1 as the test case name.
12. Set the **Maximum Time Allowed for Playback** option to 5 seconds. Leave the **Minimum Time** at 0 seconds.
13. Click **OK** and view the test case in the Visual Script. Oracle Functional Testing for Web Applications adds the test case to the Visual Script between the Address and Frame nodes.

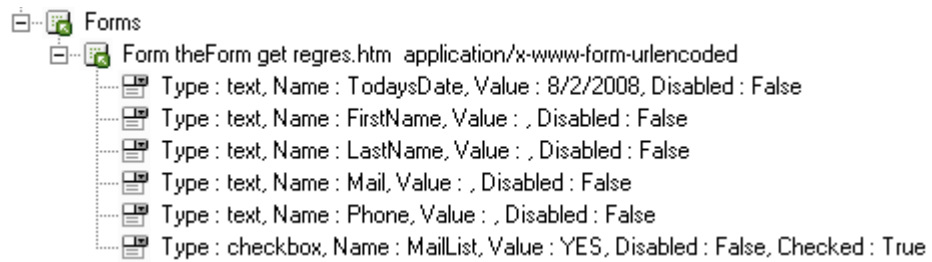
Figure 4–23 Server Response Test Script Node



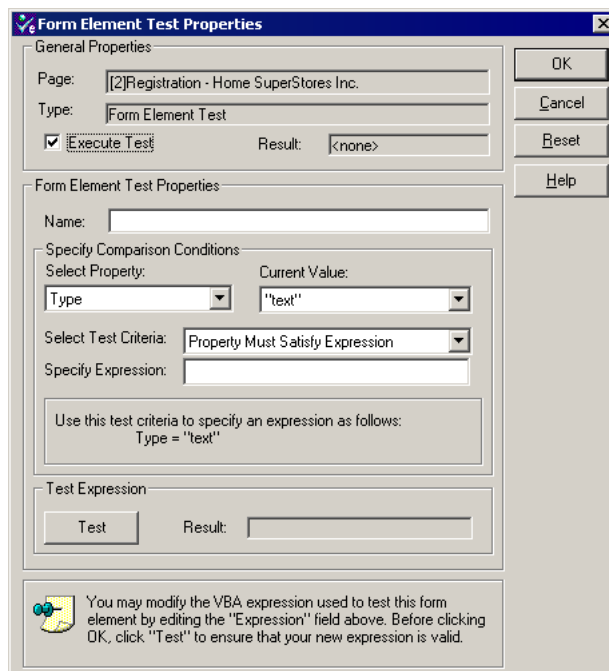
4.6.5 Insert a Form Element Test Case

Form Element test cases compare attributes and values of the elements in an HTML form.

14. Select **Current Script** from the **Options** menu and then select **Content Tests** in the **Functional Test** section. Make sure the **Forms** and **Form Elements** check box are selected and click **OK**.
15. Select the [2] Registration - Home Superstores, Inc. item in the Visual Script.
16. Expand the page and select the Today'sDate element of the regres.htm form.

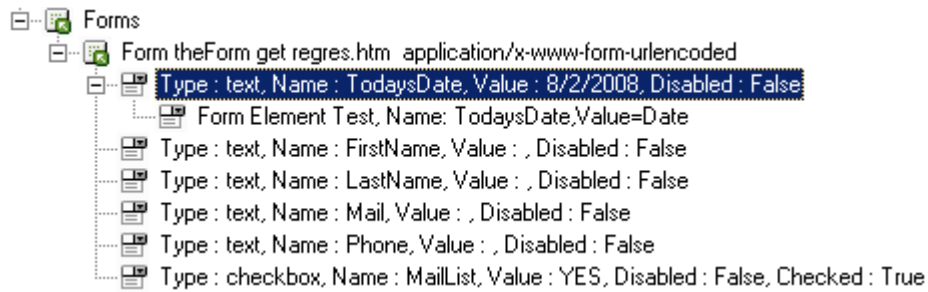
Figure 4–24 Form Element Script Nodes

17. Select **Insert Form Element Test** from the **Tests** menu. Oracle Functional Testing for Web Applications opens the Form Element Test dialog box.

Figure 4–25 Form Element Test Properties Dialog Box

18. Type **TodaysDate** as the test case name.
19. Set the **Select Property** option to **Value**.
20. Set the **Select Test Criteria** to **Property Must Satisfy Expression**.
21. Change the **Specify Expression** field to **Value = Date**.
22. Click the **Test** button. Oracle Functional Testing for Web Applications should return **True** in the **Result** field.
23. Click **OK** and view the test case in the Visual Script. Oracle Functional Testing for Web Applications adds the test case to the Visual Script under the **Form Element** node.

Figure 4–26 Form Element Test Script Node



4.6.6 Insert Table Test Test Case

Table test test cases let you define a custom test on a Web page table object. The Table Test Wizard lets you select the table object directly from the Web page by highlighting it with the mouse. The wizard also lets you specify the table object property to test and the type of test to perform.

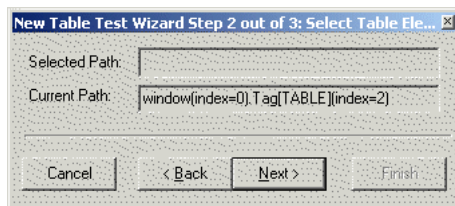
24. Select [1] Welcome - Home Superstores, Inc. in the Visual Script.
25. Click the right mouse button and select **Goto Page** to open the Welcome - Home Superstores, Inc page in the Browser.
26. Select **Insert Table Test** from the **Tests** menu. Oracle Functional Testing for Web Applications opens the Table Test Wizard.

Figure 4–27 Table Test Wizard Welcome Dialog Box



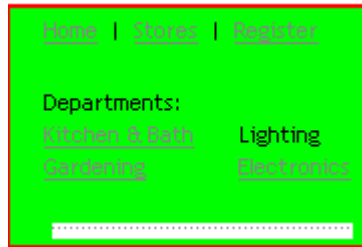
27. Type Depts as the test case name.
28. Click **Next**. The Select Table Element dialog box is displayed.

Figure 4–28 Select Table Element Dialog Box



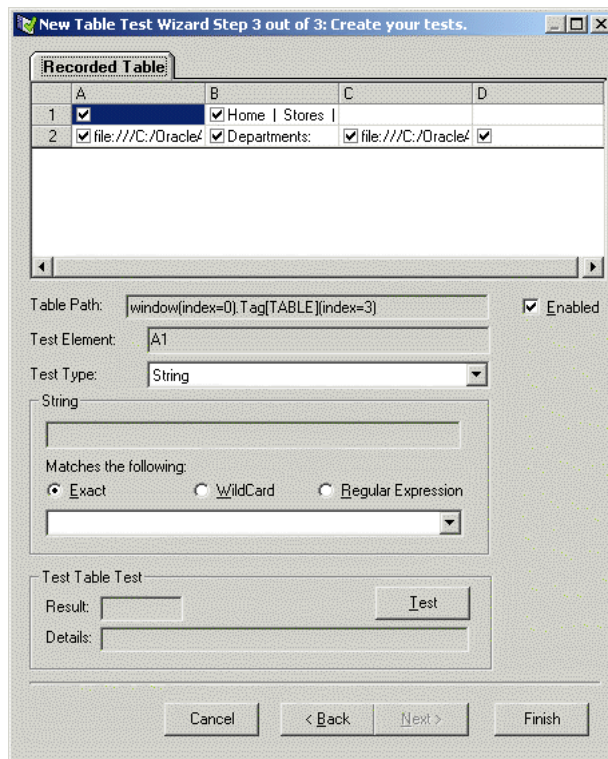
- Moving the cursor over the web page highlights the tables on the page. Highlight the Departments area in the upper left-hand corner of the page.

Figure 4–29 Table Element Selected Example



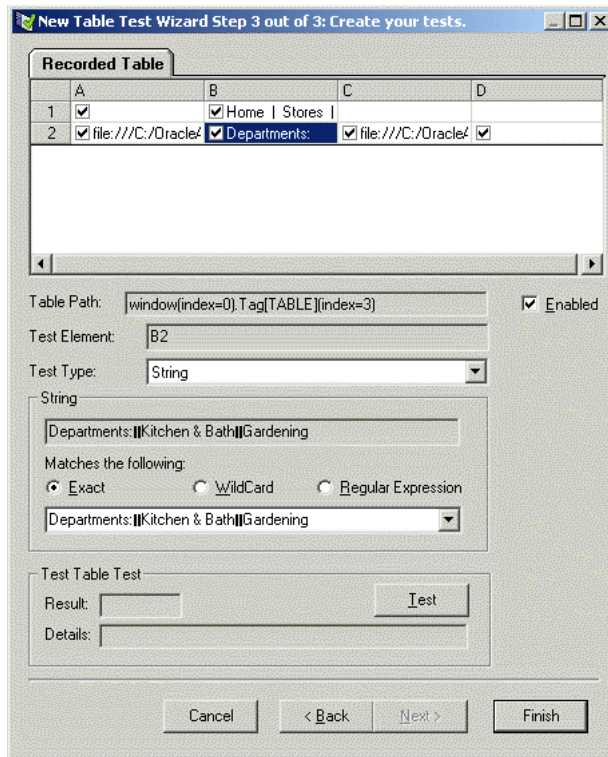
- Select the table. The path is displayed in the Select Table Element dialog box. Click Next to display the Create your tests dialog box.

Figure 4–30 Create Table Test Dialog Box



- This dialog box displays the table in a spreadsheet format, showing the data that is in each table cell. The dialog box options change depending on the type of data in the cell. Cells that are checked will be tested. Select cell B2.

Figure 4–31 Create Table Test Dialog Box with Field Selected



32. Make sure that Exact is selected in the Matches the following field.
33. Click **Test** to display the Last Played Table Tab. The result is displayed in the Result field.
34. Click **Finish** and view the test case in the Visual Script. Oracle Functional Testing for Web Applications adds the test case to the Visual Script under the User Defined Tests node.

Figure 4–32 Table Test Script Node



35. Save the script as tutor2.

4.7 Example 6: Using the Data Bank Wizard on a Search Form

This example introduces the Data Bank Wizard and explains one way to use the Data Bank Wizard with the Text Matching test case to verify Search results pages. The Data Bank Wizard provides the capability to run iterative tests using data from a Data Bank file.

1. Select **New Script** from the **File** menu to create a new Visual Script (save the previous script if prompted).
2. Reload the c:\OracleATS\OFT\rswdemo\index.htm page in the Browser pane by selecting it from the Browser drop down list.

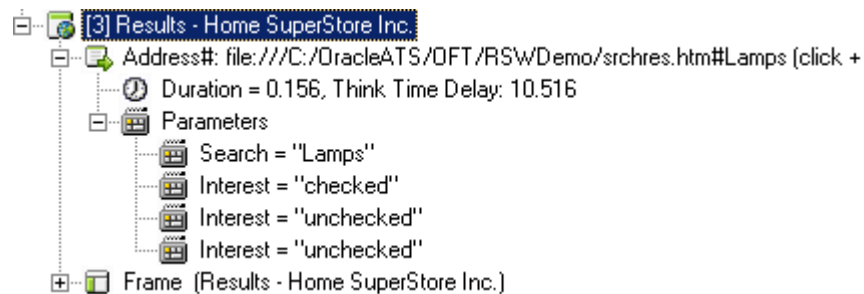
4.7.1 Recording a Search

3. Click the Record button on the toolbar.
4. Scroll the Browser pane and click the Go graphic next to Use SuperSearch.
5. Enter Lamps in the **Product Name** field and click the **Search** button. Oracle Functional Testing for Web Applications records the search including the text you typed into the field.
6. Click the Stop Record button on the toolbar.

4.7.2 Viewing the Parameters in the Visual Script

7. Expand the [3] Results - Home Superstore, Inc page in the Visual Script. Notice the Search Parameter under the Address node of the tree.

Figure 4–33 Parameters Script Node

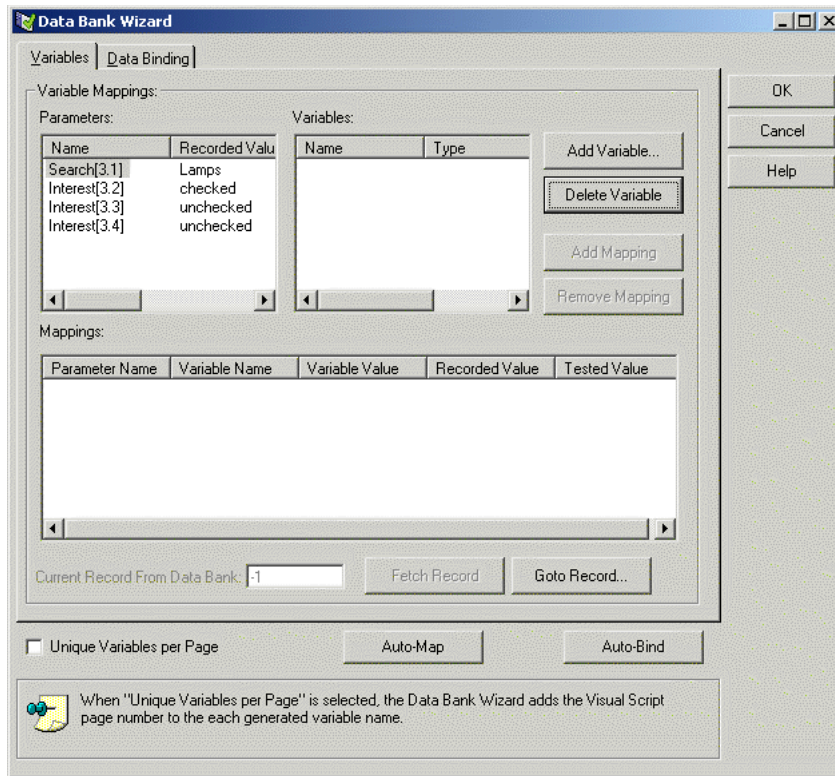


4.7.3 Using the Data Bank Wizard to Map Variables

8. Select **Data Bank Wizard** from the **Edit** menu.

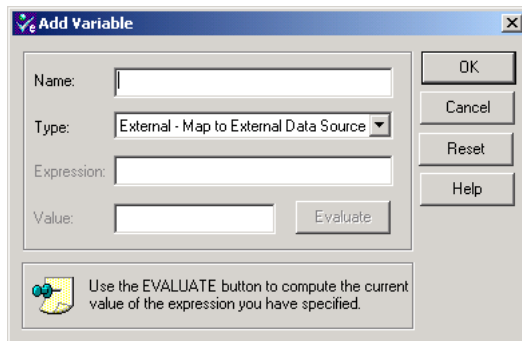
Oracle Functional Testing for Web Applications opens the Data Bank Wizard window with the parameters from the Visual Script in the **Parameters** list.

Figure 4–34 Data Bank Wizard Variables Tab



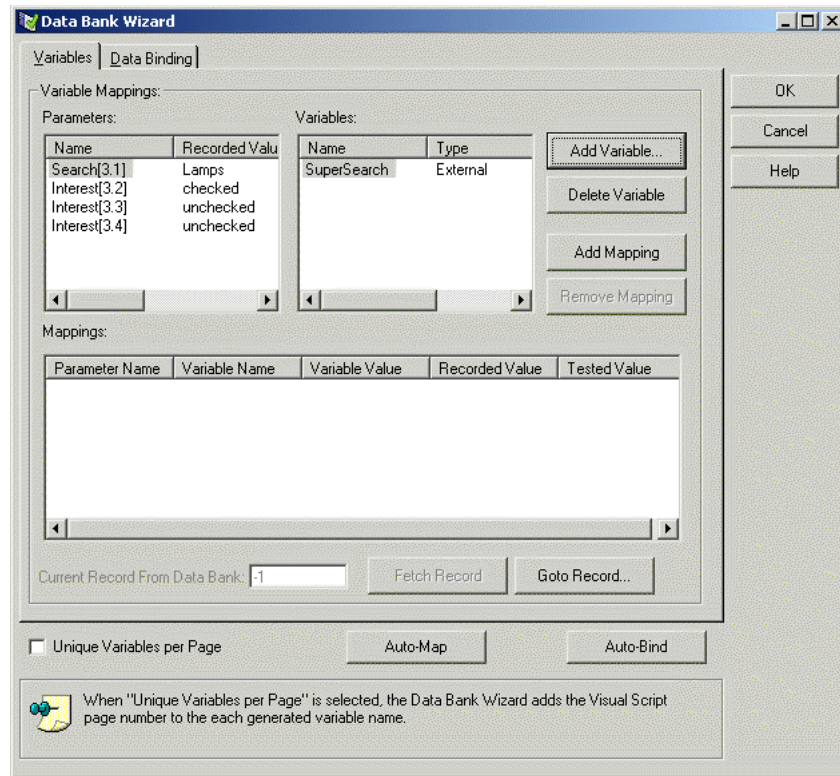
- Click the **Add Variable** button. The Data Bank Wizard opens a dialog box for specifying a variable name.

Figure 4–35 Add Variable Dialog Box



- Type SuperSearch as the variable name, and then click **OK**. The Data Bank Wizard adds the name to the **Variables** list.

Figure 4–36 Data Bank Wizard with a Variable Added

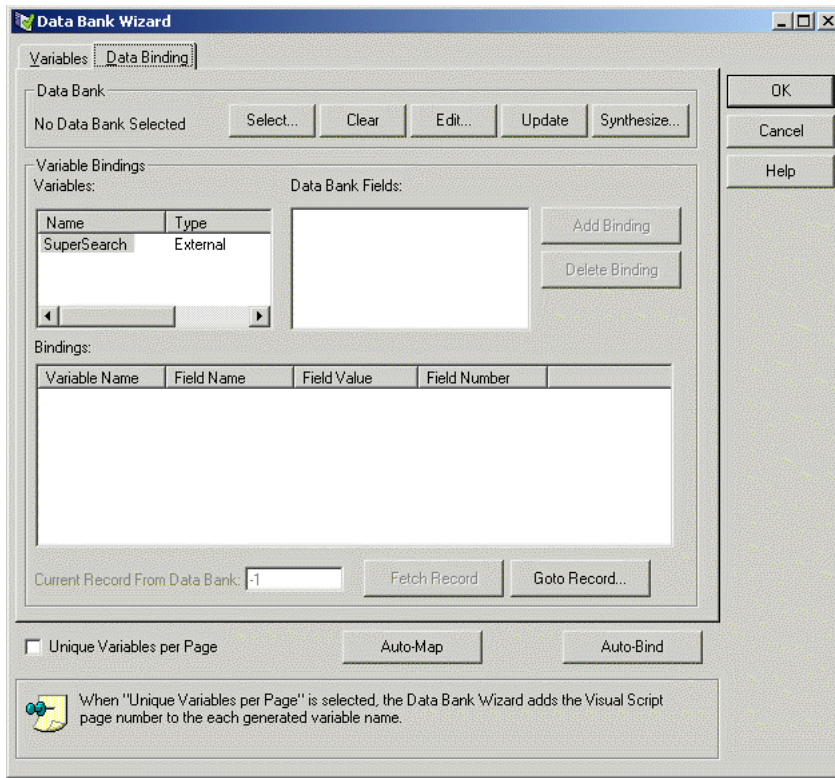


11. Select the Search[3.1] item in the **Parameters** list, and then click **Add Mapping**. The Data Bank Wizard creates a mapping between the Search[3.1] parameter and the SuperSearch variable. You now need to bind the variable name to a field in a Data Bank file.

4.7.4 Using the Data Bank Wizard to Bind to a Data Bank

12. Click the **Data Binding** tab. The Data Bank Wizard opens the Data Binding options with the variable name in the **Variables** list.

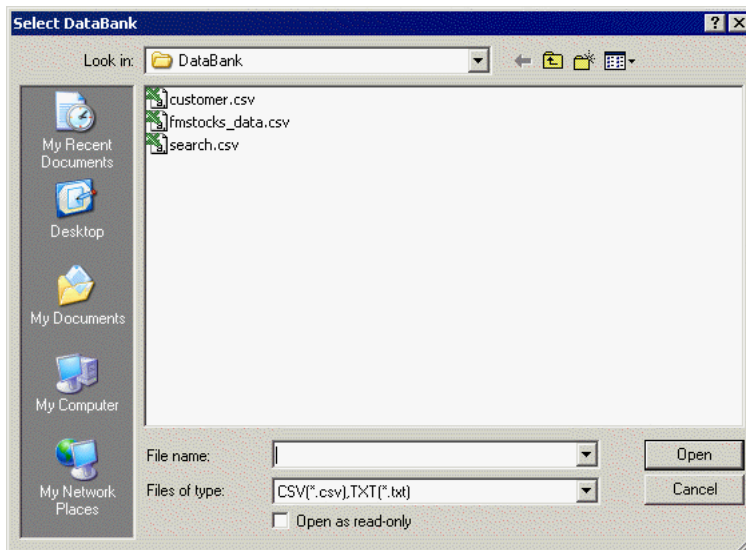
Figure 4–37 Data Bank Wizard Data Binding Tab



Now we want to select the Data Bank file that contains the values we want to use for iterative testing.

13. Click the **Select** button. The Data Bank Wizard opens a dialog box for selecting the Data Bank file.

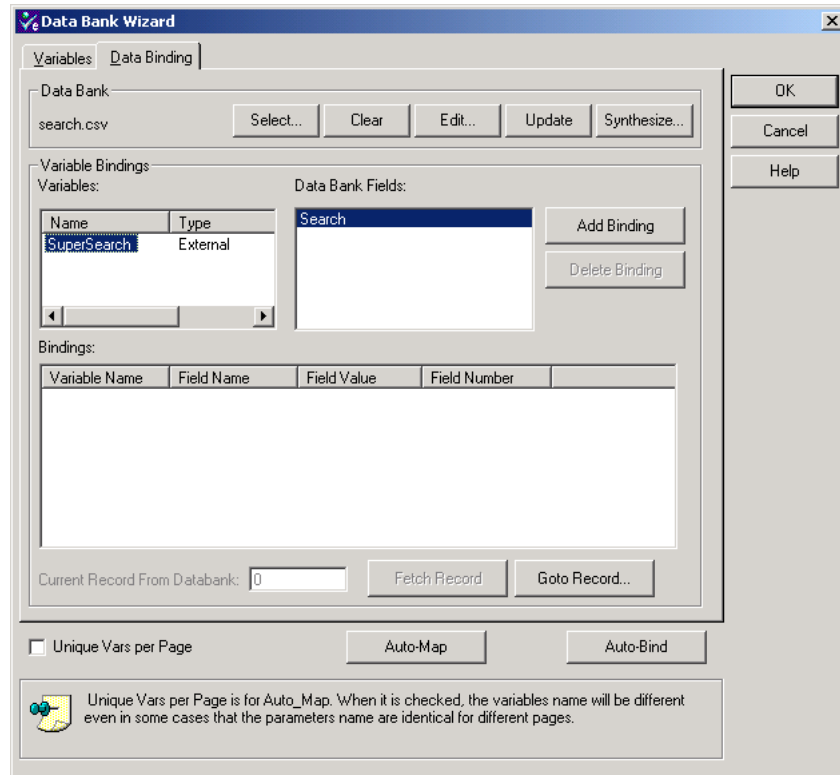
Figure 4–38 Select Data Bank Dialog Box



Note: The .csv file name extension may or may not be visible depending upon your system settings.

14. Select search.csv, and then click **Open**. The Data Bank Wizard adds the Field name from the Data Bank file.

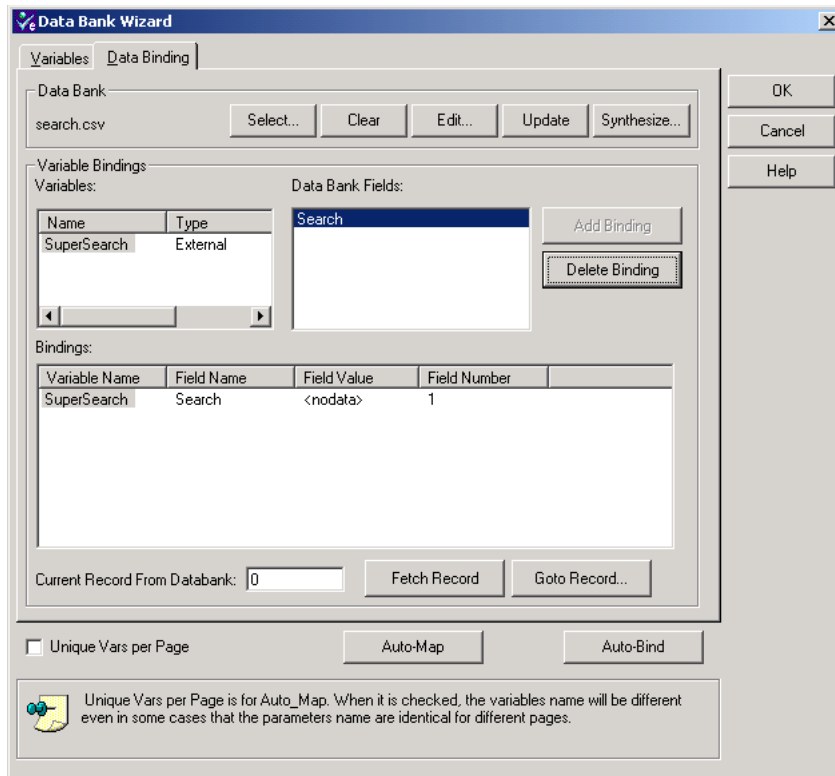
Figure 4–39 Data Bank Wizard with a Data Bank File



Note: The Data Bank file is a comma-delimited ASCII file with the field names as column headers on the first line of the file. Subsequent lines of the file contain data. You can view the contents of any of the sample files in the c:\OracleATS\OFT\DataBank directory using Notepad or any other ASCII editor.

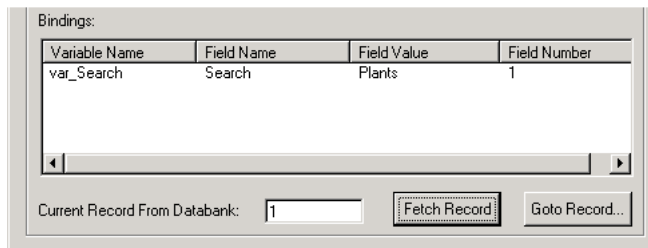
15. Select the Search field in the **Data Bank Fields** list and then click the **Add Binding** button. The Data Bank Wizard adds the variable name to the Bindings list.

Figure 4–40 Data Bank Wizard with Bindings



16. Click **Fetch Record** to cycle through the records in the external data file.

Figure 4–41 Data Bank Wizard Fields and Records View.



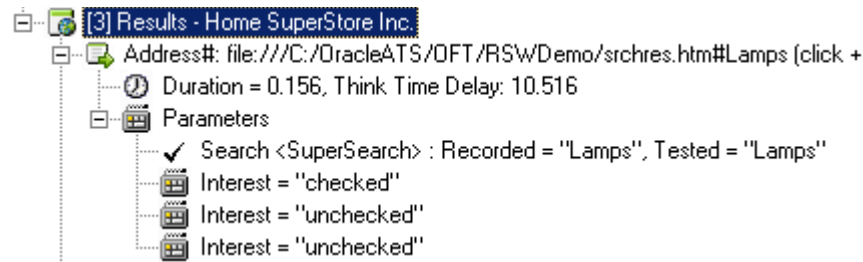
17. Continue clicking the **Fetch Record** button to cycle through all of the records in the file. There are five records in the sample Data Bank file.

18. Click the **OK** button to close the Data Bank Wizard.

4.7.5 View the Data Bank Parameters in the Visual Script

19. Examine the Parameters node under the Address node of the Visual Script tree.

Figure 4–42 Data Bank Parameters Script Nodes



The Visual Script now includes the variable names as part of the Parameters. The check mark indicates that the parameter is mapped to a variable and bound to a field in a Data Bank file.

4.7.6 Insert a Text Matching Test Case

Now we want to insert a Text Matching text case that verifies that the search results were successful.

20. Highlight the text "successfully found" in the search results page shown in the Browser pane.
21. Click the Insert Text Matching test case button on the tool bar. Notice the text you highlighted is automatically captured by Oracle Functional Testing for Web Applications.
22. Type VerifySearch as the case name, make sure **Pass when:** is set to **Selected text is present**, and then click **OK**.
23. Select the HTML node in the Visual Script. Notice the yellow flag next to the HTML node. Oracle Functional Testing for Web Applications automatically turns off the HTML comparison test when you insert a Text Matching test.

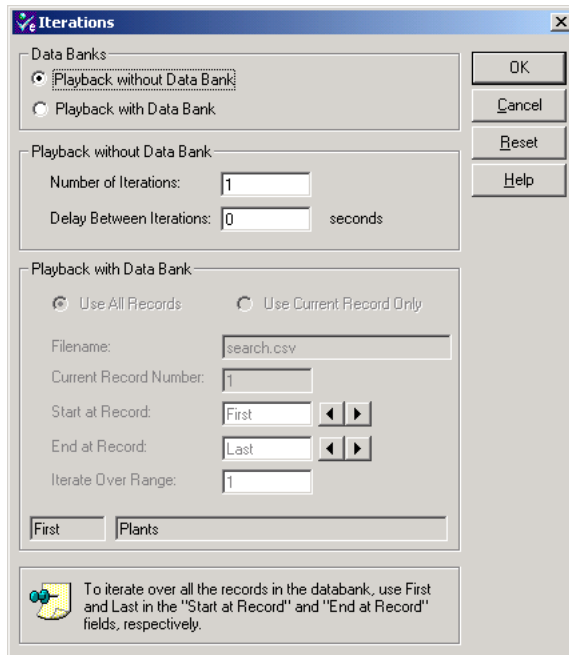
Since the search criteria will be different for each search during iterative play back, we know the HTML and product images on the page will be different from the recorded master each time. Instead of testing the HTML and images, the Text Matching test case will be used to verify a successful search.

24. Save the Visual Script as tutor3.

4.7.7 Play Back the Script with Iterations

25. Select **Playback** from the **Run** menu and then select **Iterate**. Oracle Functional Testing for Web Applications opens the Iterations dialog box.

Figure 4–43 Iterations Dialog Box



26. Select **Playback with Data Bank** and **Use All Records**.
27. Click the **OK** button to playback the Visual Script.
28. Watch as Oracle Functional Testing for Web Applications plays back the script several times using a different data value for the search each time.

4.7.8 Analyzing a Playback Failure

29. Close the Results Report.
 - Notice the Results log indicates that the search using the Furniture value (record 3) fails. We now need to analyze this failure.
30. Select **Data Bank Wizard** from the **Edit** menu.
31. Click the **Goto Record** button. The Data Bank Wizard opens a dialog box for entering the record number.
32. Enter 3, and then click **OK**.
33. Click **OK** to close the Data Bank Wizard.
34. Click the Playback button on the toolbar to play back this one record.
35. Scroll the Browser pane to view the Search Results page. Notice that the page indicates there is no product information for Furniture. Oracle Functional Testing for Web Applications was able to find this problem because the Text Matching test case you added to test for a successful search produced a failure.
36. Make sure the [3] Results - Home Superstore, Inc page is still selected in the Visual Script.
37. Click the right mouse button and select **Discard Tested Page** from the shortcut menu. The red circles are removed from the Visual Script. The Master version of the Visual Script is still the baseline to use for testing of the Web page.

4.7.9 Save the Script and the Results Log

38. Save the Visual Script.
39. Save the Output Log as tutor3.log.
40. Select **Clear Results Window** from the **View** menu to clear the results log for the next test.

4.8 Example 7: Using the Data Bank Wizard on a Registration Form

This example explains how the Data Bank Wizard makes it easy to create automated data-driven tests. Data Banks are used to hold unlimited amounts of input data that can be fed automatically into your Web application.

1. Select **New Script** from the **File** menu to create a new Visual Script.
2. Reload the c:\OracleATS\OFT\rswdemo\index.htm page in the Browser pane by selecting it from the Browser drop-down list.

4.8.1 Recording Information in a Form

3. Click the Record button on the toolbar.
4. Scroll the Browser pane and click the [Register](#) link. The Registration page contains a form for entering Name, Email Address, and Phone number information.
5. Enter your own information into the form and click the **submit my entry** button. The Results page returns showing the information you entered with a "successful registration" message.

4.8.2 Inserting a Text Matching Test Case

Now we want to insert a Text Matching text case that verifies that the Registration results were successful.

6. Highlight the text "Your registration has been added" in the search results page.
7. Click the Insert Text Matching test case button on the tool bar. Notice the text you highlighted is automatically captured by Oracle Functional Testing for Web Applications.
8. Type VerifyRegistration as the test case name, make sure **Pass when:** is set to **Selected text is present**, and then click **OK**.
9. Click the Stop Record button on the toolbar.
10. Select **Save Script As** from the **File** menu and save the file as tutor4.

4.8.3 Viewing the Parameters in the Visual Script

11. Expand the [3] Registered - Home Superstore, Inc page in the Visual Script. Notice the Parameters under the Address node of the tree.

Figure 4–44 Parameters Script Nodes in the Script



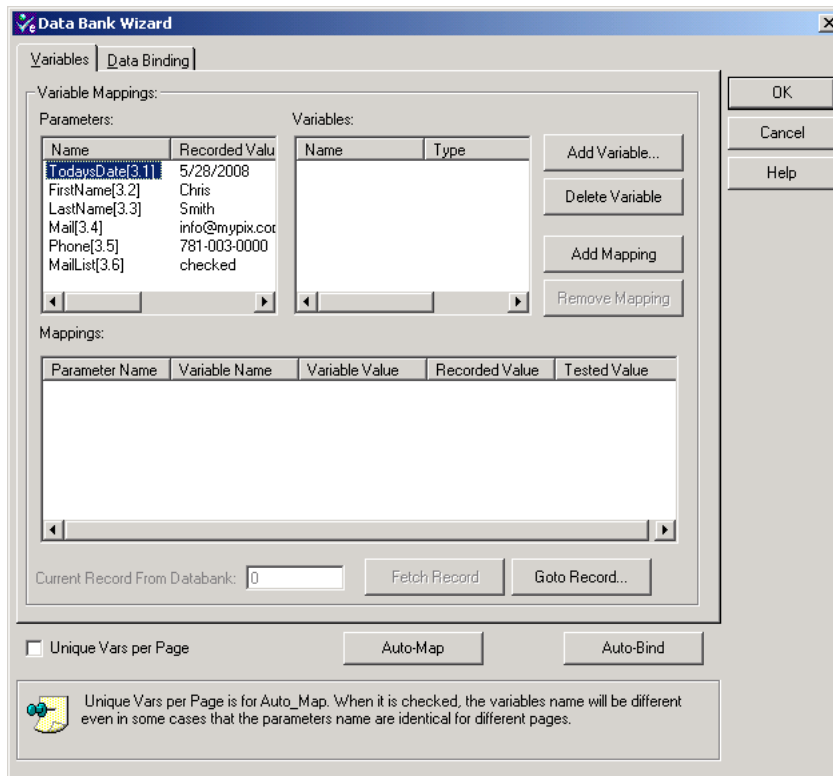
12. Make sure the [3] Registered - Home Superstore, Inc page is selected in the Visual Script.

4.8.4 Using the Data Bank Wizard to Map Variables

13. Select **Data Bank Wizard** from the **Edit** menu.

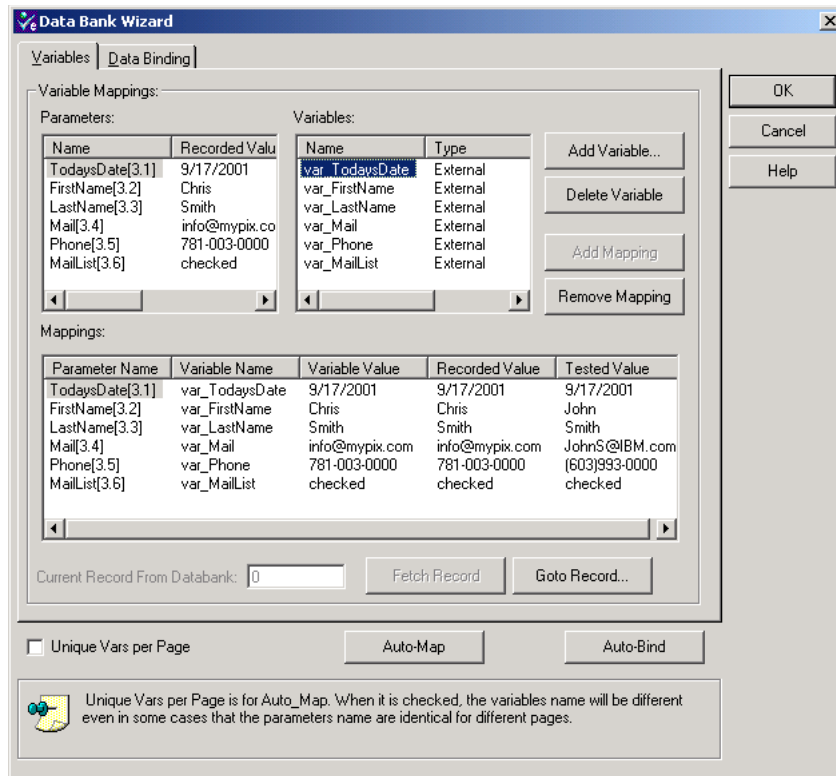
Oracle Functional Testing for Web Applications opens the Data Bank Wizard window with all the parameters from the Visual Script in the **Parameters** list.

Figure 4–45 Data Bank Wizard Variables Tab



14. Click the **Auto-Map** button. The Data Bank Wizard automatically creates variable names and maps the variable names to the Parameter names.

Figure 4–46 Data Bank Wizard with Auto-mapped Parameters



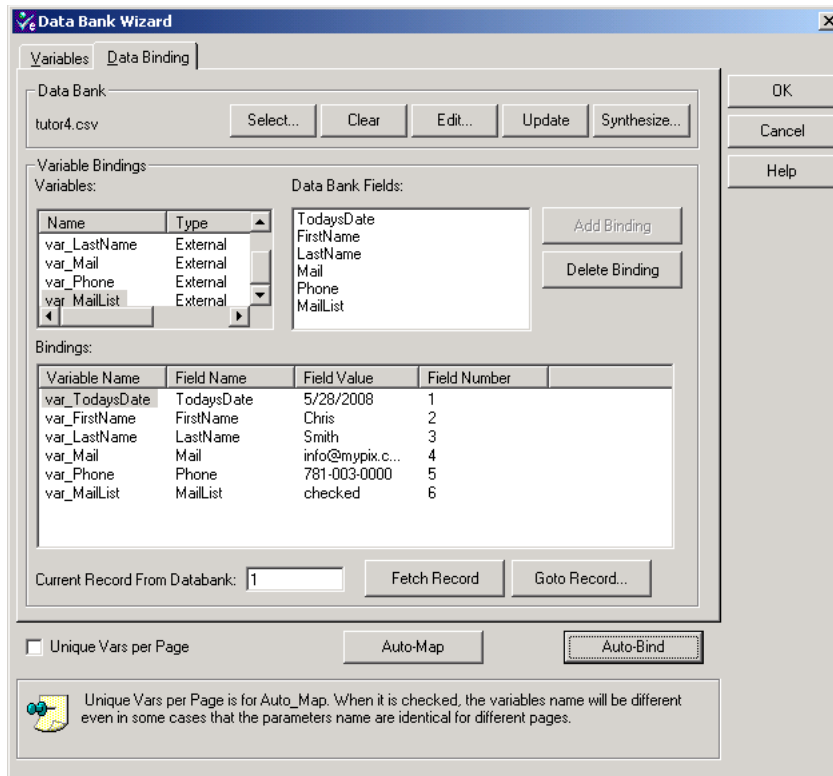
You now need to bind the variable names to fields in a Data Bank file.

4.8.5 Using the Data Bank Wizard to Bind to Data Source

15. Click the **Auto Bind** button. The Data Bank Wizard opens the Data Binding options with the variable names in the **Variables** list.

Note: Auto Bind automatically creates a Data Bank file with field definitions and one record of data. It also automatically binds the fields to the variables created.

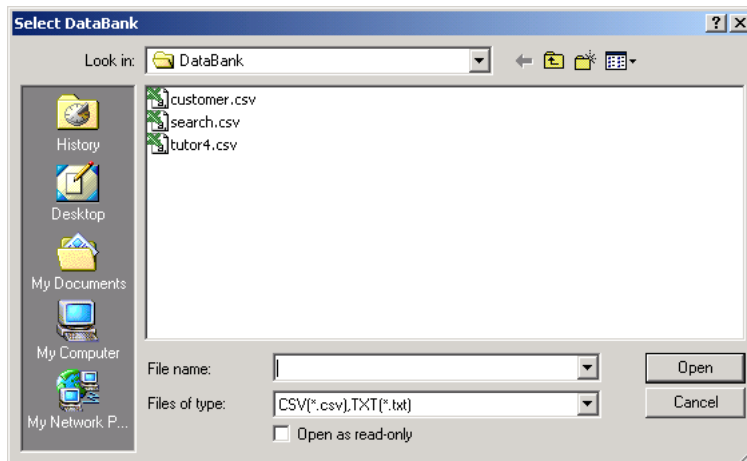
Figure 4–47 Data Bank Wizard Data Binding Tab



Now we want to select the data source that contains the values we want to use for iterative testing.

- Click the **Select** button. The Data Bank Wizard opens a dialog box for selecting the Data Bank file.

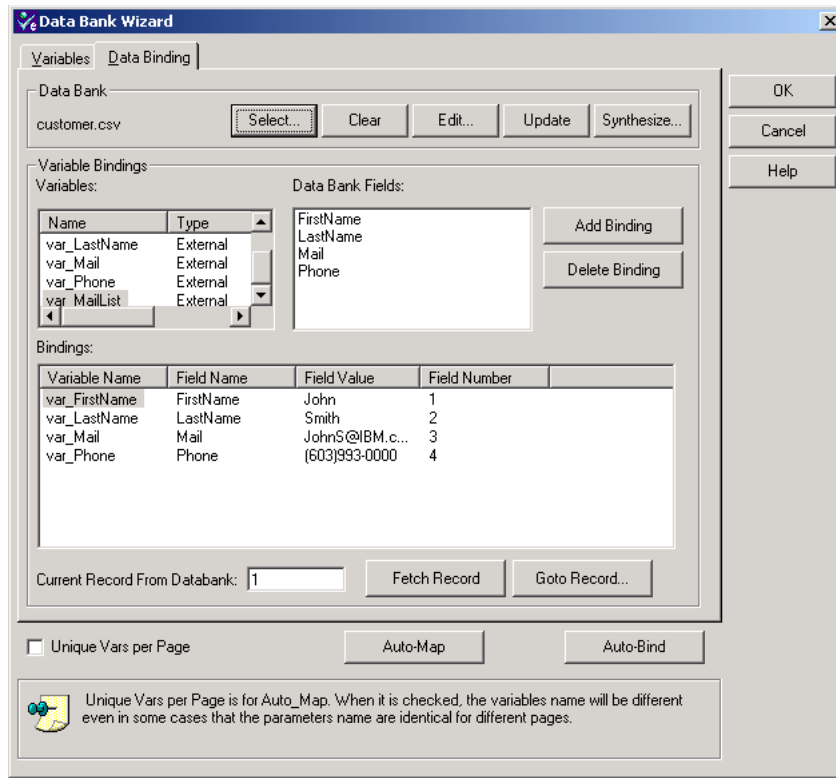
Figure 4–48 Select Data Bank Dialog Box



- Select **customer.csv**, and then click **Open**.

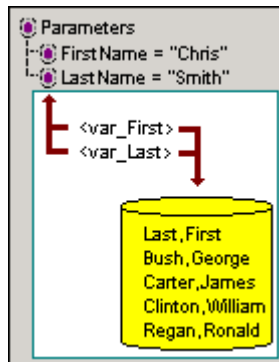
The Data Bank Wizard automatically re-binds the appropriate data field names from the Data Bank file to the variable names.

Figure 4–49 Data Bank Wizard with Bound Parameters



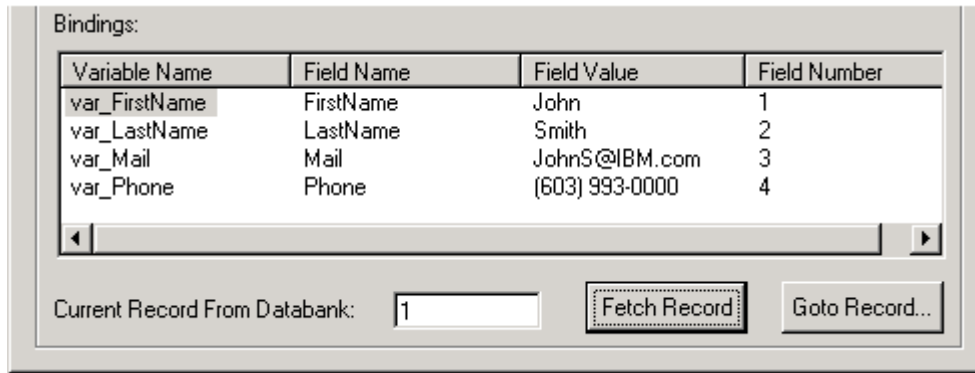
Note: The Data Bank file is a comma-delimited ASCII file with the field names as column headers on the first line of the file. Subsequent lines of the file contain data. You can view the contents of any of the sample files in the c:\OracleATS\OFT\DataBank directory using Notepad or any other ASCII editor. The following illustration shows how Data Banks map to variables and Visual Scripts.

Figure 4–50 How Scripts Map and Bind to Data Bank Records



18. Click the **Fetch Record** button to cycle through the records in the Data Bank file.

Figure 4–51 Data Bank Wizard Fields and Records View



19. Continue clicking the **Fetch Record** button to cycle through all of the records in the Data Bank file.
20. Click the **OK** button to close the Data Bank Wizard.

4.8.6 View the Data Bank Parameters in the Visual Script

21. Examine the Parameters node under the Address node of the Visual Script tree.

Figure 4–52 Script Nodes with Mapped and Bound Data Bank Variables



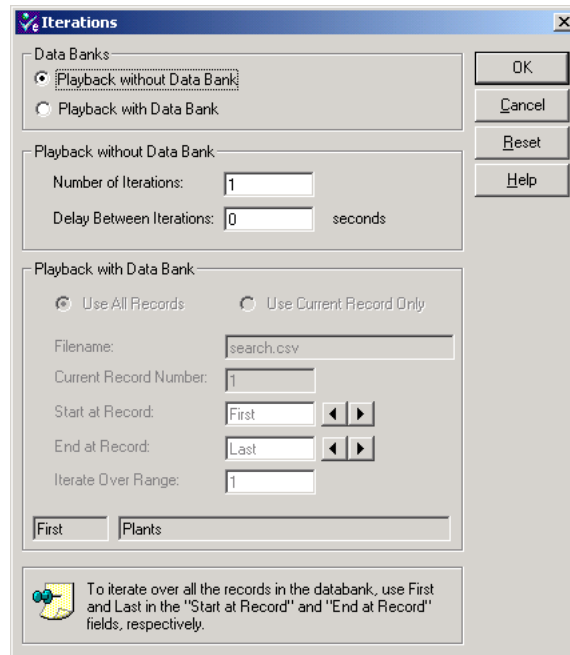
The Visual Script now includes the variable names as part of the Parameters. The check marks indicate the parameters that are mapped to variables and bound to fields in a Data Bank file. The triangles indicate variables that are mapped, but not bound to fields in a Data Bank file.

22. Save the Visual Script.

4.8.7 Play Back the Script with Data Iterations

23. Select **Current Script** from the **Options** menu and then select **Results Log**, make sure the **Failures Only** radio button in the **Results** section is selected, and then click **OK**.
24. Select **Playback** from the **Run** menu and then select **Iterate**. Oracle Functional Testing for Web Applications opens the Iterations dialog box.

Figure 4–53 Iterations Dialog Box



25. Select **Playback with Data Bank** and **Use All Records**.
26. Click the **OK** button to play back the Visual Script.
27. Watch as Oracle Functional Testing for Web Applications plays back the script several times using different data values for the registration.

4.8.8 Analyzing a Playback Failure

Notice that the Results log indicates a failure for Record 4 of the playback iteration. We now need to analyze this failure.

28. Select **Data Bank Wizard** from the **Edit** menu.
29. Click the **Goto Record** button. The Data Bank Wizard opens a dialog box for entering the record number.
30. Enter 4, and then click **OK**.
31. Click the **OK** button to close the Data Bank Wizard.
32. Click the **Playback** button on the toolbar to play back this one record.
33. Close Results Report.

You can play back the current record repeatedly using **Current Script** from the **Options** menu, select the **Playback** section, and then select the **Use Current Record** option.
34. Scroll the Browser pane to view the Registration Results page. Notice that the page indicates a server error. Oracle Functional Testing for Web Applications was able to find this error because the Text Matching test case you added to test for successful registration produced a failure.
35. Make sure the [3] Results - Home Superstore, Inc page is still selected in the Visual Script.

36. Click the right mouse button and select **Discard Tested Page** from the shortcut menu. The red circles are removed from the Visual Script. The Master version of the Visual Script is still the baseline to use for testing of the Web page.

4.8.9 Save the Script and the Results Log

37. Save the Visual Script.
38. Save the Output Log as `tutor4.log`.
39. Select **Clear Results Window** from the **View** menu to clear the results log.
This completes the Oracle Functional Testing for Web Applications tutorial.

Job Scheduler Tutorial

This tutorial walks you through the main features of Job Scheduler. You can follow the examples in this chapter to become familiar with the features and use of Job Scheduler.

This tutorial consists of the following examples:

- **Creating a job and schedule** - describes how to create a job using the Job Scheduler Wizard and then add it to a schedule.
- **Editing a Job** - explains how to edit a job after you have created it.
- **Editing a Schedule** - explains how to edit a schedule after you have created it.

The tutorial is designed to be followed sequentially from beginning to end and assumes you have completed the Oracle Functional Testing for Web Applications tutorial in Chapter 3. The examples in this tutorial refer to Visual Scripts recorded in the Oracle Functional Testing for Web Applications tutorial.

5.1 Example 1: Creating a Job and Schedule

This example illustrates how to create a job using the Job Scheduler Wizard and then add it to the Current Schedule.

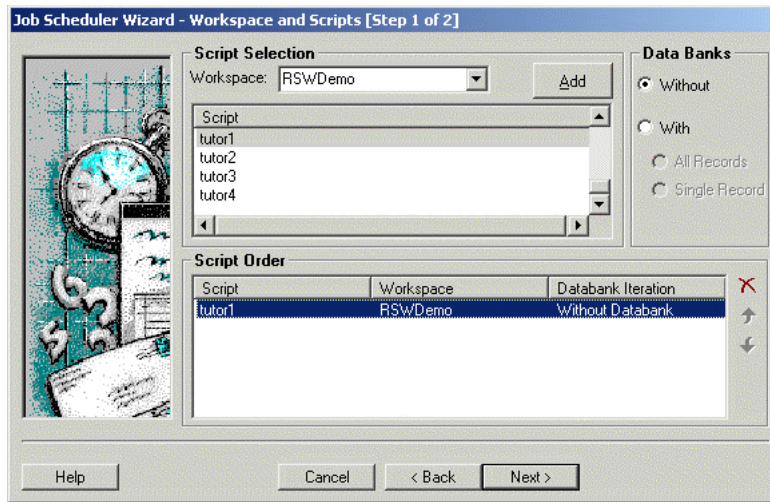
5.1.1 Starting Job Scheduler

1. Select **Programs** from the **Start** and then select **Job Scheduler** from the **Oracle Application Testing Suite** submenu to start Job Scheduler.
2. Select **New Job** from the **File** menu or click the Job Scheduler Wizard toolbar button. Job Scheduler opens the Wizard Welcome screen. If you do not want the Welcome screen to appear each time you run Job Scheduler, select the **kip this screen in the future** check box.
3. Click the **Next** button to continue to the Workspace and Scripts [Step 1 of 2] screen.

5.1.2 Specifying the Scripts

The Workspace and Scripts [Step 1 of 2] screen of the Job Scheduler Wizard is where you specify the scripts to include in the job.

Figure 5–1 Job Scheduler Wizard Workspace and Scripts Dialog Box

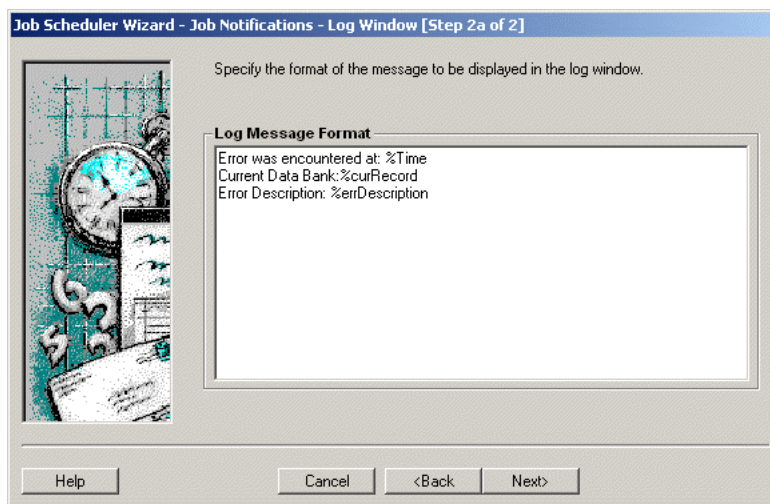


4. Select the RSWDemo workspace. Note that you can include scripts from more than one workspace in a job.
5. Select tutor3 and click on the **Add** button to add the script to the **Script Order** list.
6. Select tutor4 and click on the **Add** button.
7. Click the **Next** button to continue to the Job Notifications Log Window [Step 2a of 2] screen of the Job Scheduler Wizard.

5.1.3 Specifying the Job Notifications

The Job Notifications Log Window [Step 2a of 2] screen of the Job Scheduler Wizard is where you specify the message that will appear in the Results Pane and the Results Log if an error occurs during playback of a Visual Script.

Figure 5–2 Job Scheduler Wizard Log Window Dialog Box



The default log message writes the following information to the results log if an error occurs during playback of a Visual Script:

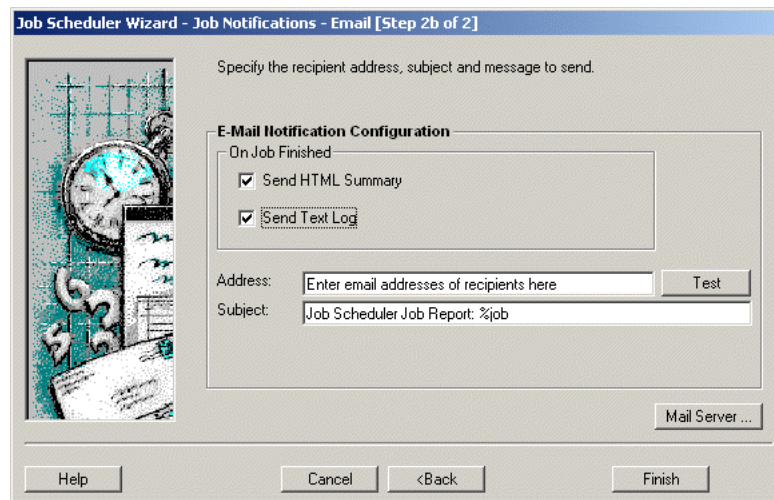
- Time of the error

- Databank record
 - An error number
 - A description of the error
8. Click the **Next** button to continue to the Job Notifications Email [Step 2b of 2] screen of the Job Scheduler Wizard.

5.1.4 Specifying Email Notifications

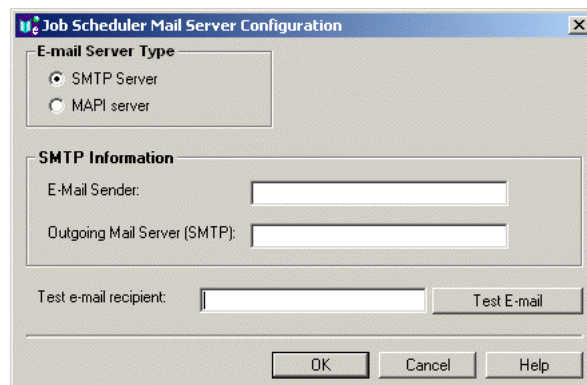
The Job Notifications Email [Step 2b of 2] screen of the Job Scheduler Wizard is where you specify who to notify when the job is finished and what information to send.

Figure 5–3 Job Scheduler Wizard Email Dialog Box



9. Select the **Send HTML Summary** check box to send the Job Report to the recipients.
10. Select the **Send Text Log** to send a text version of the log.
11. Enter your email address in the **Address** field. Separate additional email addresses by a comma or semicolon.
12. Click **Mail Server**. Job Scheduler opens the server configuration dialog box.

Figure 5–4 Job Scheduler Wizard Mail Server Configuration Dialog Box

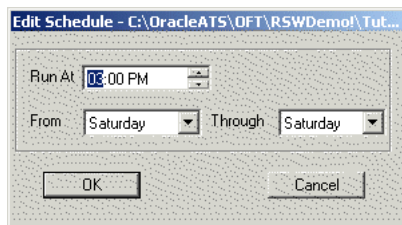


13. Select your email server type and enter the email information for your email system. Check with your system administrator if you are not sure of the information to enter for your email configuration. You can use the **Test email recipient** address and **Test Email** button to verify email capabilities from within Job Scheduler.
14. Click **OK** to return to the Job Scheduler Wizard.
15. Click the **Test** button next to the **Address** field if you want to verify the email notification from the wizard.
16. Click the **Finish** button. The Save Job As dialog box is displayed.
17. Enter tutorJob1 for the job name and click **Save**.
18. You are then asked if you want to schedule the job. Click **Yes**.

5.1.5 Scheduling the Job

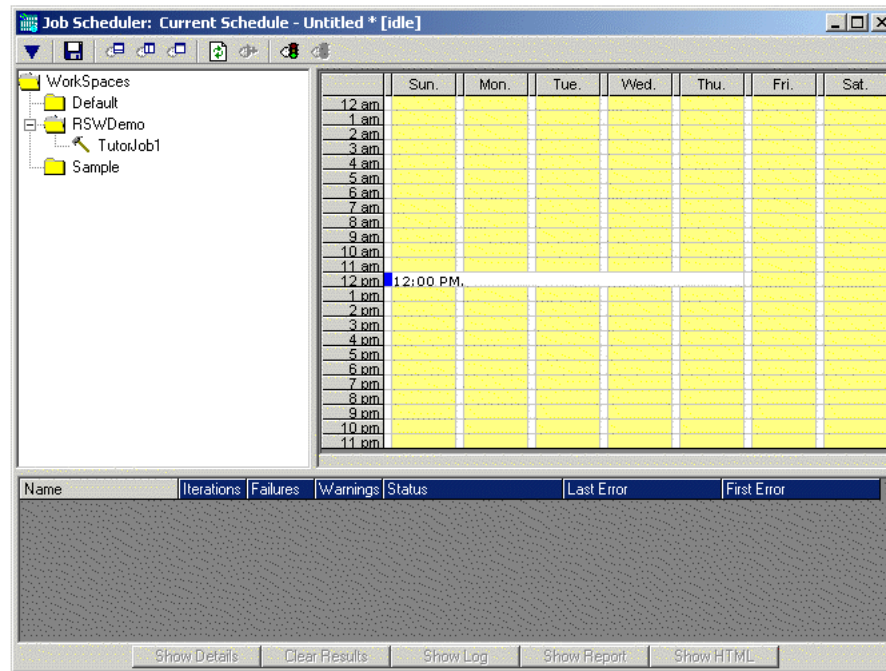
The Current Schedule window is displayed as well as the Edit dialog box for scheduling the job you just created.

Figure 5–5 Job Scheduler Wizard Edit Schedule Dialog Box



19. Change the **Run At** time to 12:00 pm.
20. Change the starting day in the From field to **Sunday** and the ending day in the Through field to **Thursday**
21. Click **OK** to schedule the job.

Figure 5–6 Job Scheduler Current Schedule Window



5.1.6 Saving the Schedule

The job file is separate from the schedule file. The job is the list of scripts to run and the notification information. The schedule file contains the information about when the scripts will be played back and which jobs will be played back.

1. Select **Save Schedule As** from the **File** menu.
2. Enter tutorSch1 for the schedule name and click **Save**.

The file name is displayed in the title bar of the Current Schedule window.

5.1.7 Playing Back the Job

1. In the Job Editor window, click the **Start** button to play back the job.
2. Watch as Job Scheduler processes the Visual Scripts. A yellow bar in the row of the script indicates that the script is being processed. A red bar indicates that the script failed; a green bar indicates that the script passed.

Upon completion an email will be sent to you with the job report and text log if you used your email address when setting up the job notifications.

5.1.8 Activating the Schedule

When the schedule is activated, the scheduled jobs automatically run at the scheduled times.

1. Make the Current Schedule window active by clicking anywhere in it.
2. Select **Activate Schedule** from the **Control** menu or click on the Activate Schedule toolbar button. The results are displayed in the Job Summary pane of the Current Schedule window.

Only one schedule can be opened or activated at any one time.

Double-click on the job name in the Job Summary pane to view Job Details as the scripts are played back.

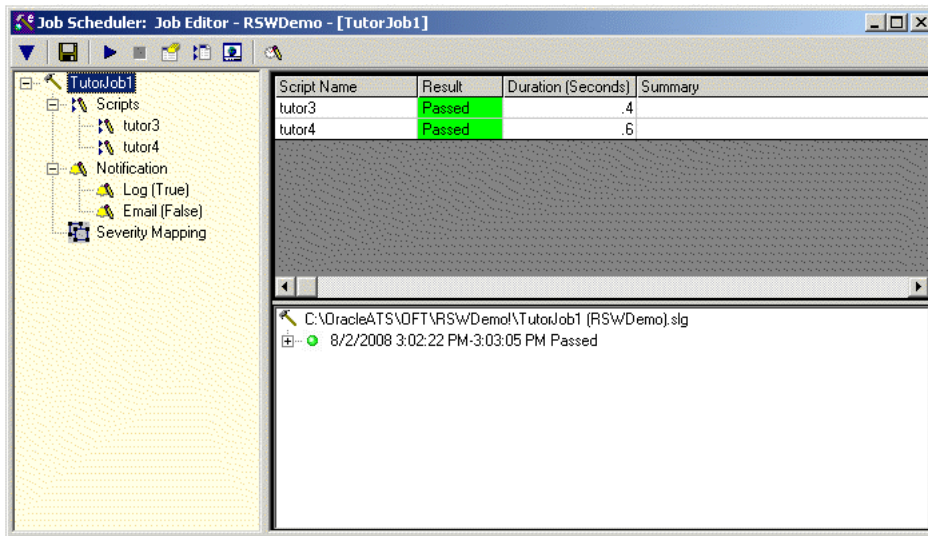
3. Select **Deactivate Schedule** from the **Control** menu or click on the Deactivate Schedule toolbar button to deactivate the schedule.

5.2 Example 2: Editing a Job

This example illustrates how to edit a job using the Job Editor window after the job has been created with the Job Scheduler Wizard. Note that you can have more than one job open at any one time.

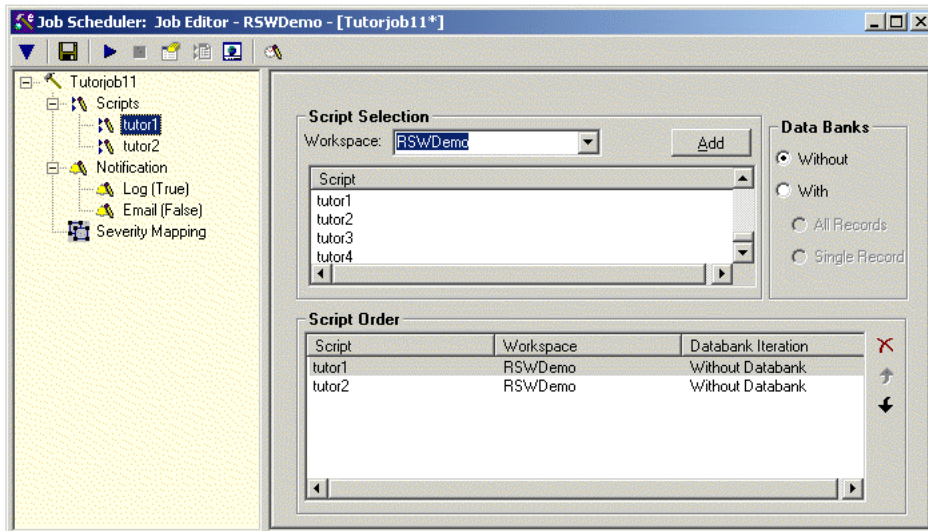
1. Select **Open Job** from the **File** menu to open a job or make the Job Editor window active by clicking on it to edit the currently open job.

Figure 5–7 Job Editor Window



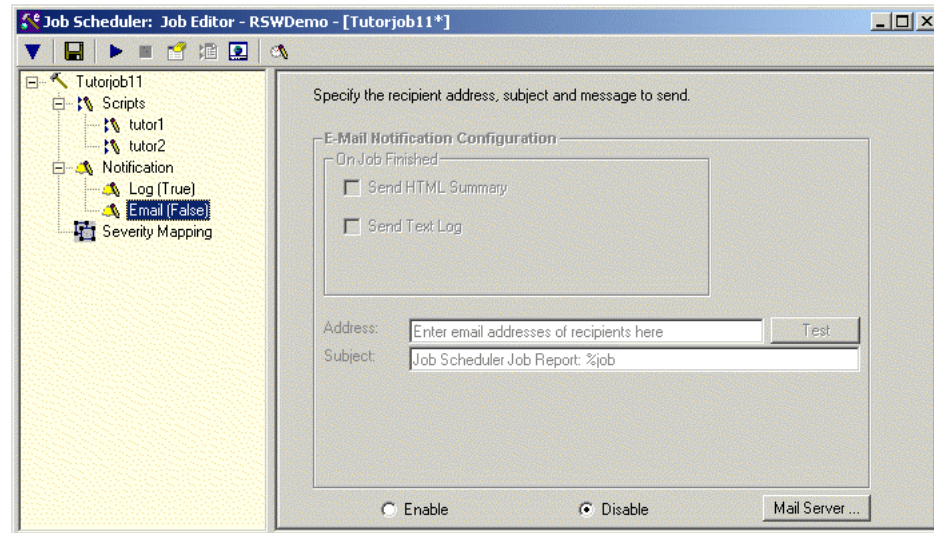
2. Click **Scripts** in the job tree to display the script selection options.

Figure 5–8 Job Editor Script Selection Window



3. Select the RSWDemo workspace and add the tutor2 script to the **Script Order** list.
4. Click Email in the job tree to display the email options.

Figure 5–9 Job Editor Email Window



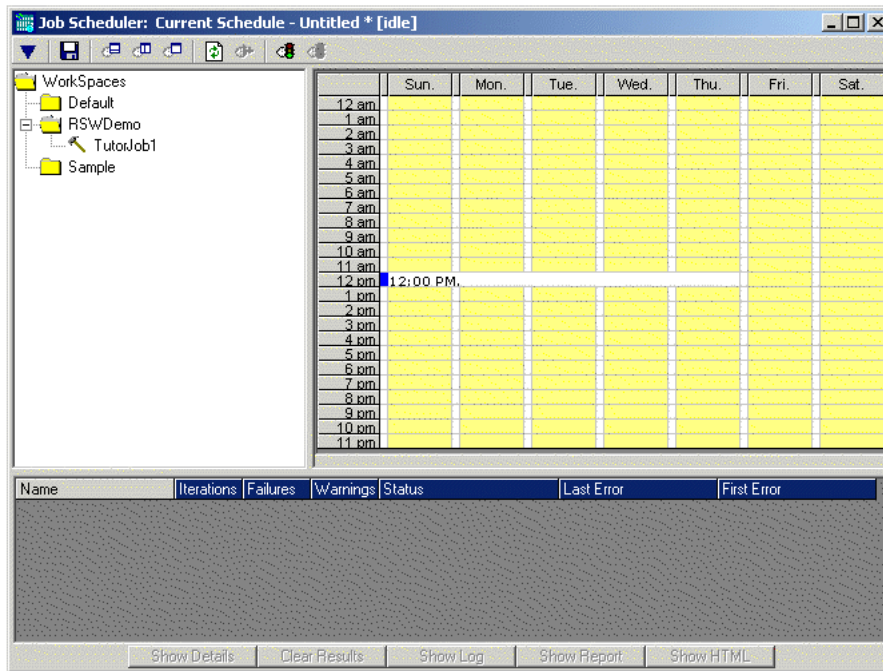
5. Select the **Disable** option button to disable sending emails on error.
6. Save the job by selecting **Save tutorJob1** from the **File** menu.

5.3 Example 3: Editing a Schedule

This example illustrates how to edit a schedule by adding and changing playback times of jobs. This example uses only one job but you can schedule more than one job.

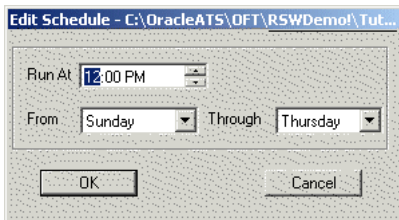
1. Select **Open Schedule** from the **File** menu to open tutorSch1.mjs or make the Current Schedule window active by clicking on it to edit the currently open job.

Figure 5–10 Job Scheduler Current Schedule Window



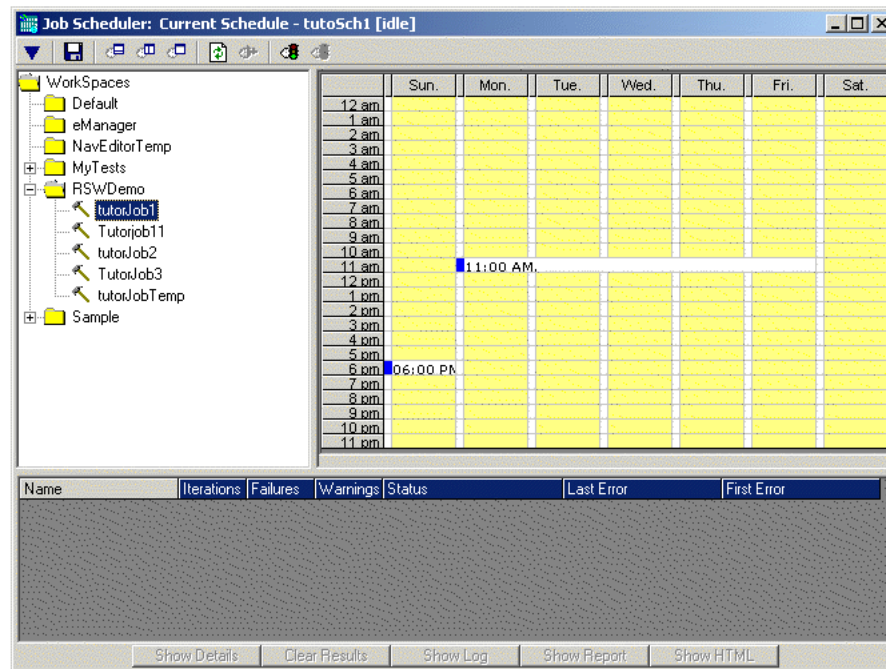
2. Double-click on the job in the calendar or right-click on the job and select **Edit Schedule**.

Figure 5–11 Edit Schedule Dialog Box



3. Change the time to 11:00 am, the starting day to **Monday** and the ending day to **Friday**.
4. Click on **tutorJob1** in the job tree and drag it to Sunday at 6:00 pm.

Figure 5–12 Job Scheduler Current Schedule Window



5. Save the schedule by selecting **Save tutorSch1** from the **File** menu.

Oracle Load Testing for Web Applications Tutorial

This tutorial walks you through the main features of Oracle Load Testing for Web Applications. Oracle Load Testing for Web Applications is a separate product in the Oracle Application Testing Suite, which you may or may not have purchased. If you have the Oracle Load Testing for Web Applications version of the Oracle Application Testing Suite, you can follow the examples in this chapter to become familiar with the features and use of Oracle Load Testing for Web Applications.

The tutorial consists of the following examples:

- **Performing a Simple Load Test** - shows how to use Oracle Load Testing for Web Applications to run virtual users to simulate load on a Web application.
- **Adding Data Sources** - shows how to add data sources to the Oracle Load Testing for Web Applications ServerStats configuration to monitor server-side statistics, such as CPU usage, and available memory.
- **Editing Data Sources** - shows how to edit existing Oracle Load Testing for Web Applications ServerStats configurations to modify specific counters.
- **Creating a Scenario with Multiple Profiles** - shows how to add a new Visual Script to the Oracle Load Testing for Web Applications Scenario. This example also shows how to set the Reporting options and Session Start/Stop options to save data for use in post-run analysis.
- **Running Multiple Profiles** - shows how to use Oracle Load Testing for Web Applications to run multiple Scenario profiles with different amounts of virtual users and how to view statistical and performance information.
- **Controlling Virtual Users** - shows how to modify individual virtual user attributes, view actions, and stop and abort virtual users.
- **Generating Reports** - explains how to view the default reports and generate reports for post-run analysis.
- **Creating User-Defined Profiles** - explains how to create user-defined virtual user profiles.

The tutorial is designed to be followed sequentially from beginning to end and assumes you have completed the Oracle Open Script tutorial in Chapter 3 and the Oracle Functional Testing for Web Applications tutorial in Chapter 4. The examples in this tutorial refer to scripts recorded in the previous tutorials.

6.1 Example 1: Performing a Simple Load Test

This example shows how to use Oracle Load Testing for Web Applications to run virtual users to simulate load on a Web application. The example illustrates how to run a previously recorded Visual Script to simulate multiple users accessing a Web application.

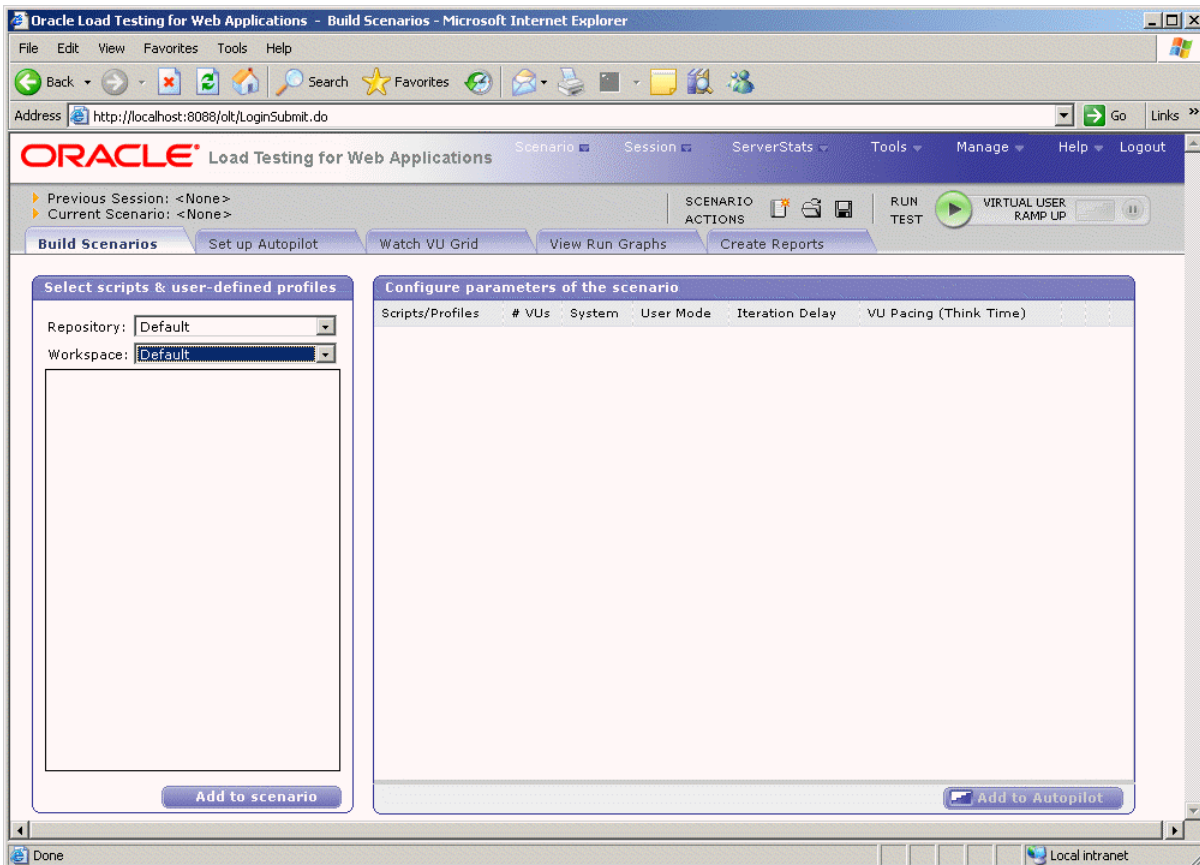
6.1.1 Starting Oracle Load Testing for Web Applications and Specifying the Workspace

Note: This section uses the default login credentials from the Oracle Application Testing Suite installation.

To start Oracle Load Testing for Web Applications:

1. Select **Programs** from the **Start** menu and then select **Oracle Load Testing for Web Applications** from the **Oracle Application Testing Suite** menu.
2. Enter **Administrator** as the user name.
3. Enter the password specified during the Oracle Application Testing Suite installation process.
4. Click **Login**. The main window appears, as follows:

Figure 6–1 Oracle Load Testing for Web Applications Main Window

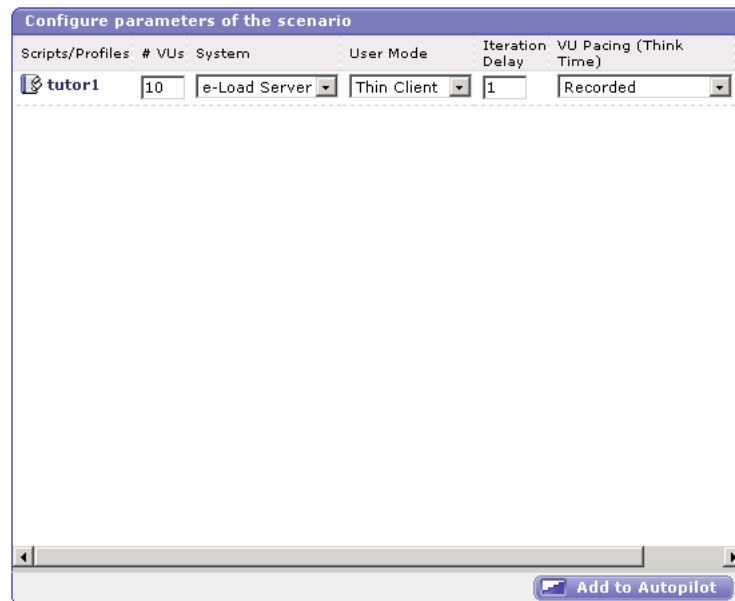


5. Select RSWDemo in the Workspace list.

6.1.2 Specifying a Scenario Profile

6. Make sure the **Build Scenario** tab is displayed in Oracle Load Testing for Web Applications.
7. Select tutor1 in the **Select scripts** list. These are the scripts that you record using Oracle OpenScript or Oracle Functional Testing for Web Applications. For Oracle OpenScript scripts, only load testing and general scripts appear in the list. Functional test scripts do not.
8. Click the **Add to scenario** button to add tutor1 to the **Configure Parameters** list. You can also double-click the script name to add it to the **Configure Parameters** list.

Figure 6–2 Configure Parameters Pane

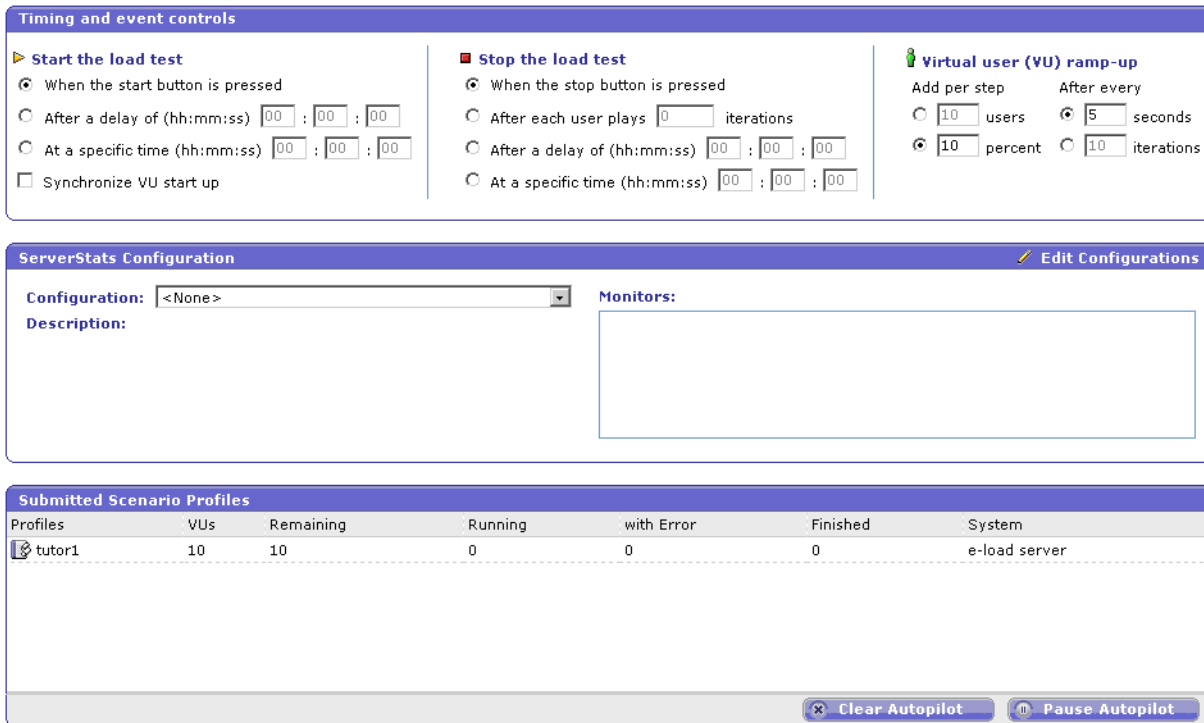


Oracle Load Testing for Web Applications automatically specifies a set of default virtual user attributes for the Scenario Profile in the Scenario tab. For this example, we'll use the default attributes.

9. Click the **Add to Autopilot** button on the Build Scenario tab.

Oracle Load Testing for Web Applications automatically opens the Set up Autopilot tab with the tutor1 Scenario Profile listed in **Submitted Scenario Profiles** list.

Figure 6-3 Autopilot Window



6.1.3 Running the Scenario Profile Using Autopilot

10. Select **After each user plays [#] iteration** option from the **Stop the load test** group of the Autopilot tab.
11. Enter 5 in the **After each user plays** edit box.
12. Select [#] users below **Add per step** in the Virtual User ramp up group.
13. Enter 1 in the **Add per step [#] users** edit box.
14. Select [#] seconds below **After every** in the Virtual User ramp up group.
15. Enter 10 in the [#] **seconds** edit box.
16. Click the **Run Test** button on the Autopilot tab or the toolbar.
17. Select **Yes** when asked to record session data and click **OK**.

Oracle Load Testing for Web Applications starts running the virtual users in the Virtual User Grid. Watch as the Autopilot starts running the tutor1 Visual Script as ten virtual users.

Figure 6–4 Virtual User Status Grid Window

VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Page	System	Data Bank	Current Error	Previous Error
1	tutor1	Finished	5		7.09		localhost			
2	tutor1	Finished	5		6.649		localhost			
3	tutor1	Finished	5		6.179		localhost			
4	tutor1	Finished	5		6.149		localhost			
5	tutor1	Finished	5		6.109		localhost			
6	tutor1	Finished	5		6.249		localhost			
7	tutor1	Finished	5		6.079		localhost			
8	tutor1	Finished	5		6.569		localhost			
9	tutor1	Finished	5		6.149		localhost			
10	tutor1	Finished	5		6.089		localhost			

18. Allow the virtual users to continue running until all of them indicate Finished in the **Status** column of the virtual user grid.

Congratulations. You have just performed a simple load test on the Demo Web application. Oracle Load Testing for Web Applications performs the virtual user Web interaction in the background. You can monitor the virtual users in the grid as they are running. In the later examples of this tutorial, you'll see how to use Oracle Load Testing for Web Applications to view statistical and performance information, and how to view virtual user actions.

6.2 Example 2: Adding Data Sources

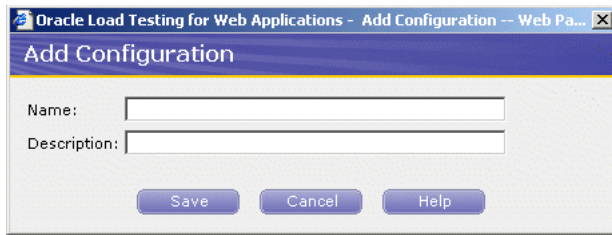
This example shows how to add data sources to the Oracle Load Testing for Web Applications ServerStats configuration to monitor server-side statistics, such as CPU usage, and available memory.

Oracle Load Testing for Web Applications ServerStats can monitor statistics from a variety of systems and server types. This tutorial adds counters from your local Windows 200x/XP system to demonstrate the features of Oracle Load Testing for Web Applications ServerStats. If you are not running the Oracle Application Testing Suite on a Windows 200x/XP machine, you should skip examples two and three and continue the tutorial with example 4.

To configure counters from a Windows 200x/XP data source, do the following:

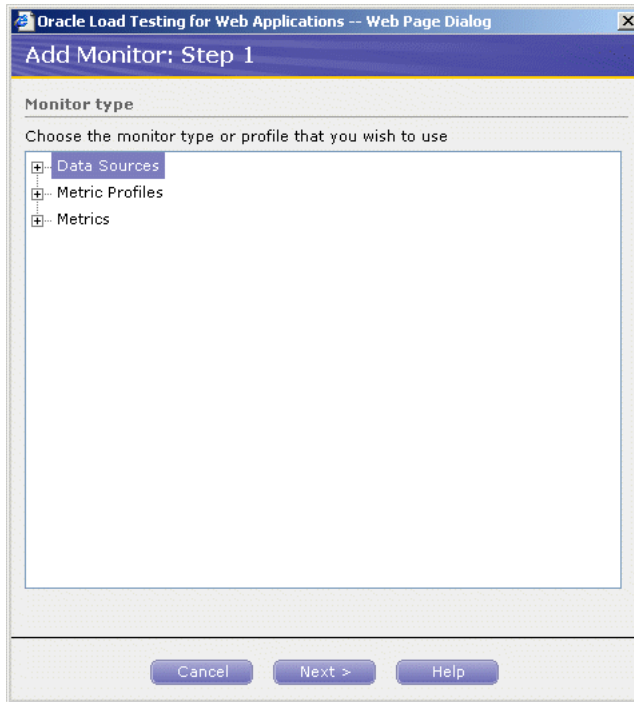
1. Select **Configurations** from the **ServerStats** menu. Oracle Load Testing for Web Applications opens the ServerStats Configurations window.
2. Click **New** to add a new configuration.

Figure 6–5 Add Configuration Dialog Box

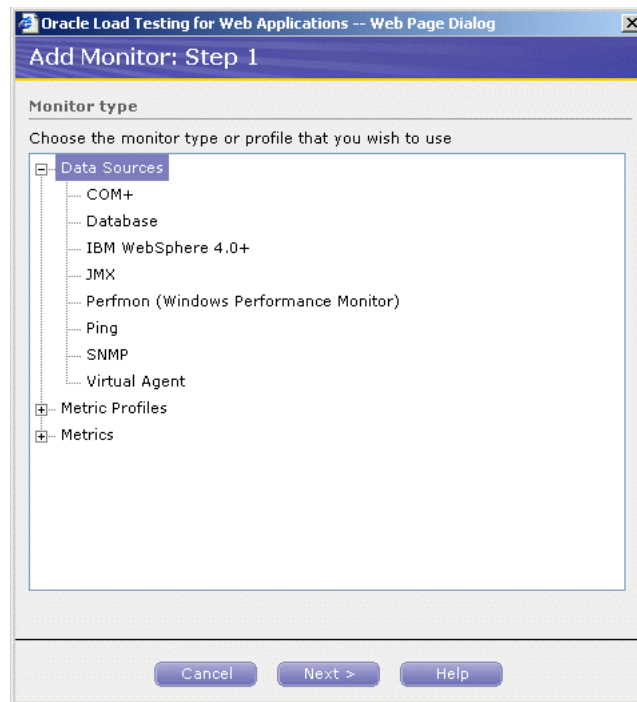


3. Type Tutorial for the Name and the Description.
4. Click **Save**. The Configuration window changes to include the Monitors configuration options.
5. Click New in the Monitor pane to open the Add Monitor Step 1 window.

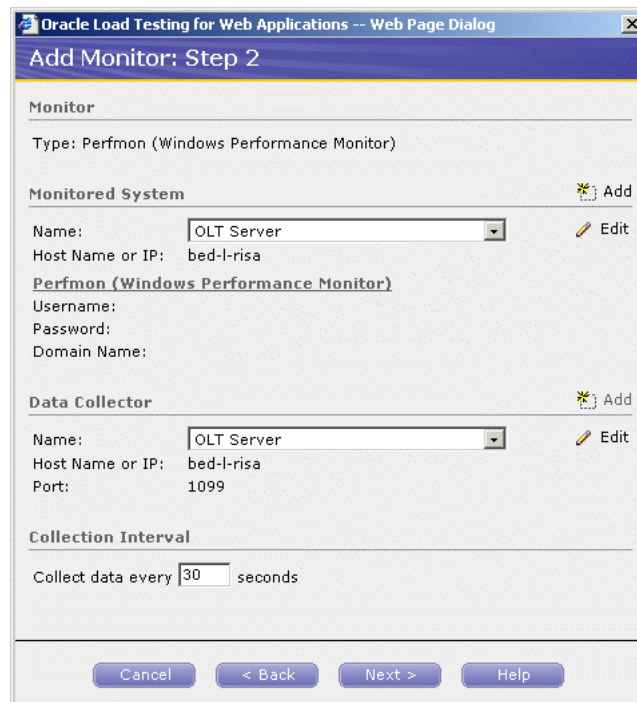
Figure 6–6 Add Monitors Step 1 Dialog Box



6. Click the Plus icon next to Data Sources to expand the list of data sources.

Figure 6–7 Add Monitors Step 1 with Data Sources Expanded

7. Select Perfmon (Windows Performance Monitor) and click **Next**.

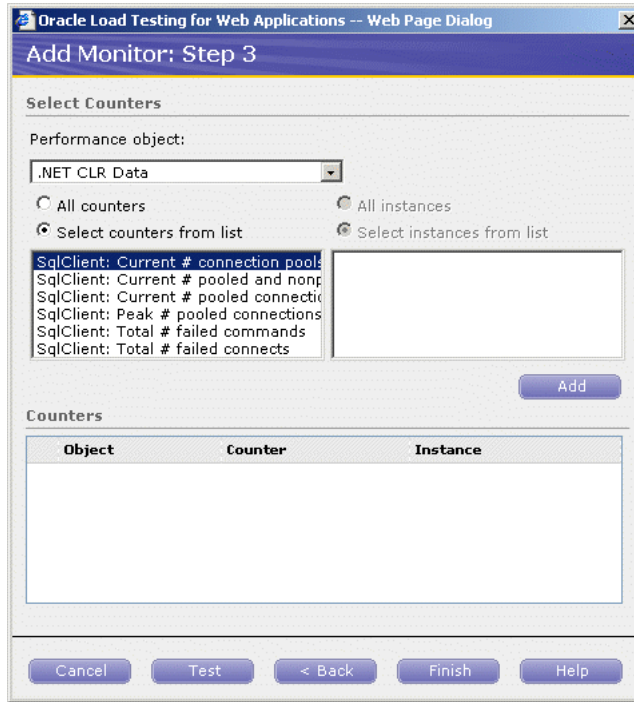
Figure 6–8 Add Monitors Step 2 Dialog Box

This step lets you specify which to monitor and which system to use for the data collector.

8. Leave the default settings for **Monitored System** and **Data Collector**.

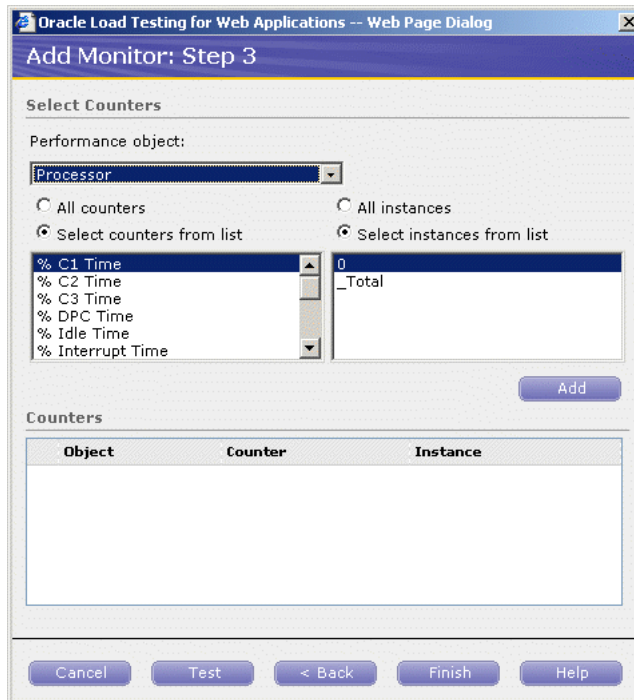
- Click **Next** to select the specific counters to monitor.

Figure 6–9 Add Monitors Step 3 Dialog Box



- Select **Processor** in the **Performance object** list.

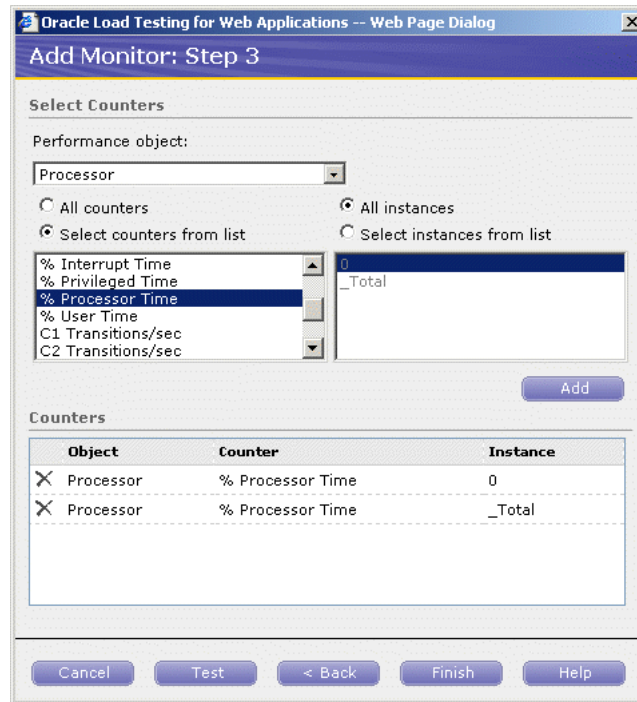
Figure 6–10 Add Monitors Step 3 with Processors Listed



- Select **% Processor Time** in the **Select counters from list** group.

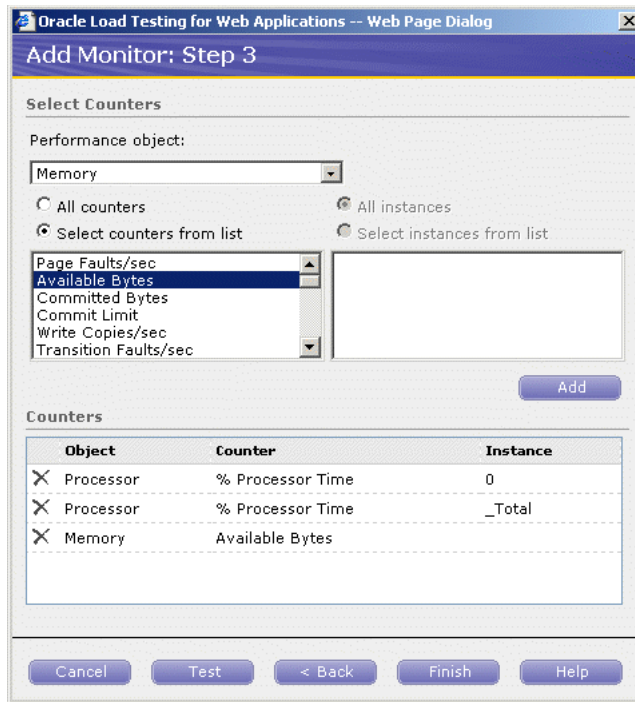
12. Select the **All instances** option.
13. Click **Add**. The counters are added to the **Counters** list.

Figure 6–11 Add Monitors Step 3 with Processors Selected



14. Select **Memory** in the **Performance object** list.
15. Select **Available Kbytes** and click **Add**.

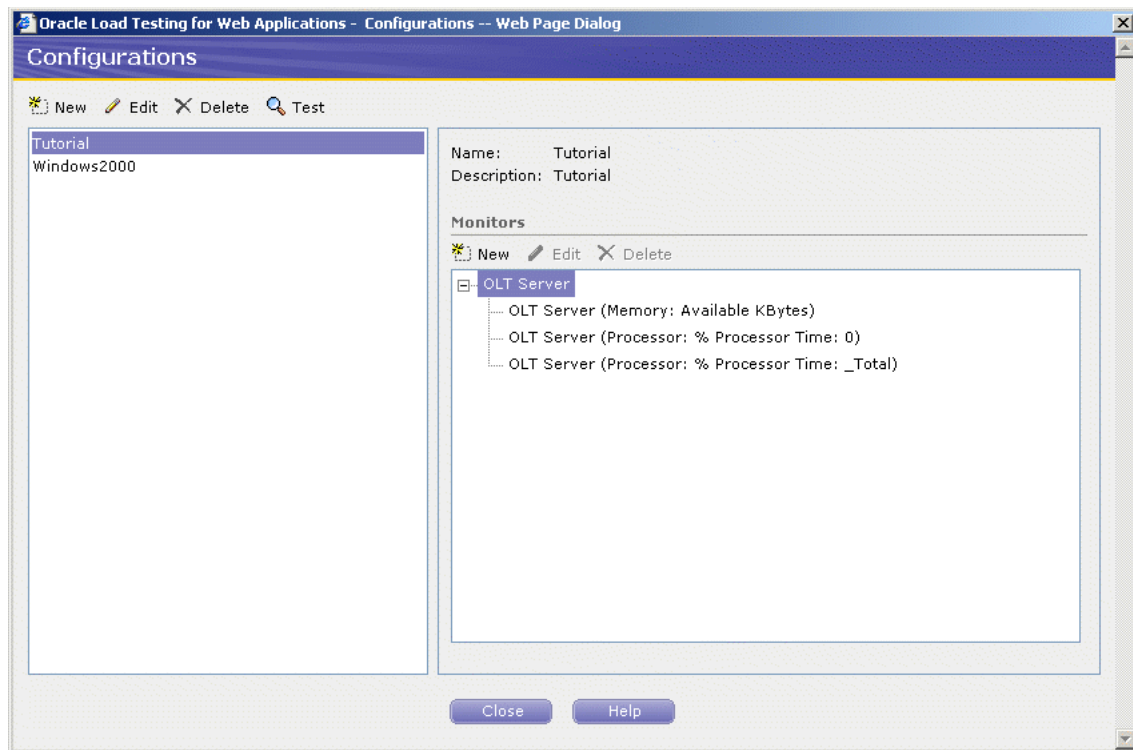
Figure 6–12 Add Monitors Step 3 with Processors and Memory Selected



16. Click **Finish** to complete adding monitors. ServerStats display a status while it verifies the counters (this may take a few moments).

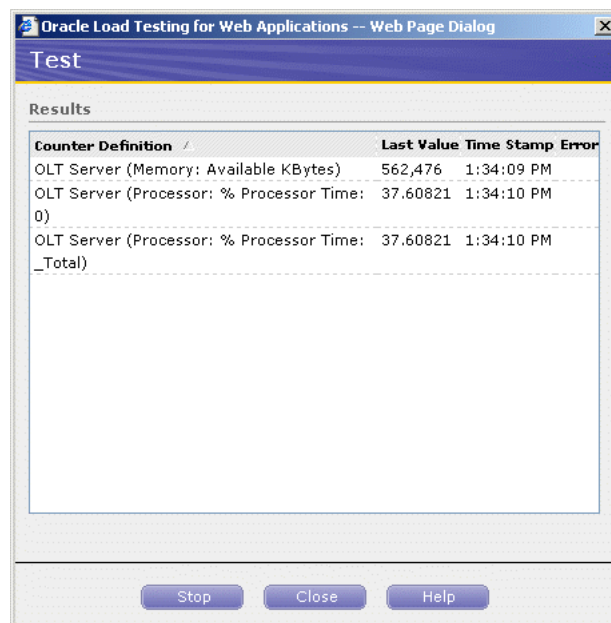
When the verification is complete, the Configuration window is updated with the list of monitors.

Figure 6–13 Configurations Dialog Box with Defined Monitors



17. Click **Test** to test the counters.

Figure 6–14 Test Monitors Dialog Box



18. Review the results to verify the counters are working properly.

19. Click **Close** to exit the test results.

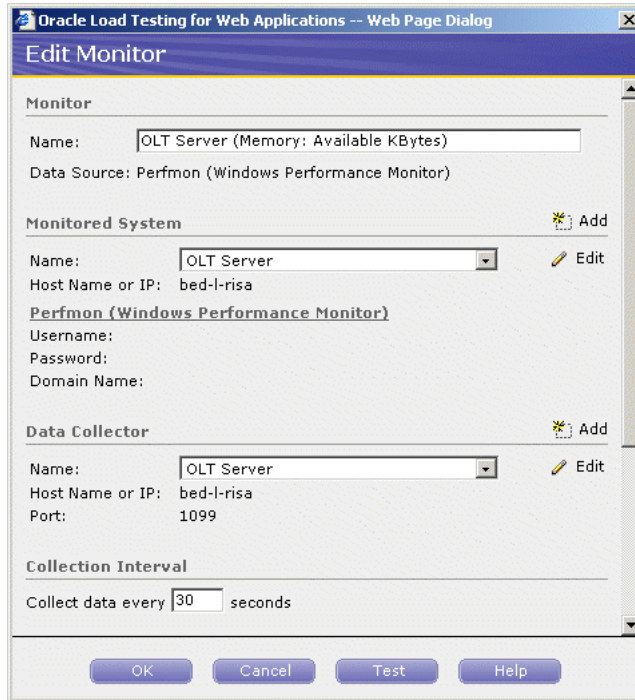
The next example explains the procedures for editing existing ServerStats configurations.

6.3 Example 3: Editing Data Sources

This example shows how to edit existing Oracle Load Testing for Web Applications ServerStats configurations to modify specific counters. The steps in this example are based upon steps completed in the previous example.

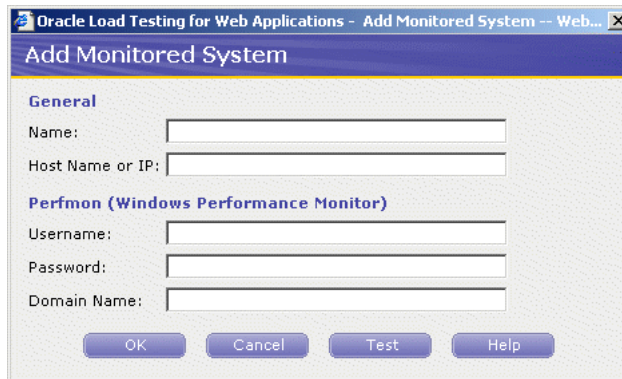
1. If not already open, select **Configurations** from the **ServerStats** menu to open the ServerStats Configurations window.
2. Select the Tutorial configuration.
3. Click the Memory (Available Kbytes) monitor and click **Edit**.

Figure 6–15 Edit Monitor Dialog Box



4. Click **Add** next to **Monitored System**.

Figure 6–16 Add Monitored System Dialog Box



If you have additional systems to monitor you can specify the information here to add the system to the ServerStats Configuration.

5. Click **Cancel**.
6. Change the **Collection Interval** to 45 seconds.
7. Click **OK**.
8. Click **Close** to close the ServerStats Configurations window.

See the *Oracle Load Testing for Web Applications User's Guide* for additional information about using the features and options of Oracle Load Testing for Web Applications ServerStats.

6.4 Example 4: Creating a Scenario with Multiple Profiles

This example shows how to create scenarios with multiple virtual user profiles and how to set the attributes for each scenario. It also shows how to specify the reporting options.

6.4.1 Adding a Virtual User Profile to the Scenario

1. Click the **Build Scenarios** tab.
2. Double-click tutor3 in the **Default Profiles** list to add it to the **Configure parameters of the scenario** list.
3. Click the **Configure all parameters** button on the tutor3 line to display the Edit Scenario Details dialog box.
4. Change the **#VUs** value to 3.
5. Make sure the **Virtual User Pacing** is set to Recorded and the **Maximum** value is set to 1 second.
6. Change the **Caching Emulation** to Repeat User.
7. Change the **User Mode** to Thick Client.
8. Make sure the **Use Databanks** field is True.
9. Leave the default settings for remainder of the attributes and click **OK**.
10. Click the **Configure all parameters** button on the tutor1 line to display the Edit Scenario Details dialog box.
11. Change the **# VUs** value to 6.
12. Make sure the **Virtual User Pacing** is set to Recorded and the **Maximum** value is set to 1 second and click **OK**.

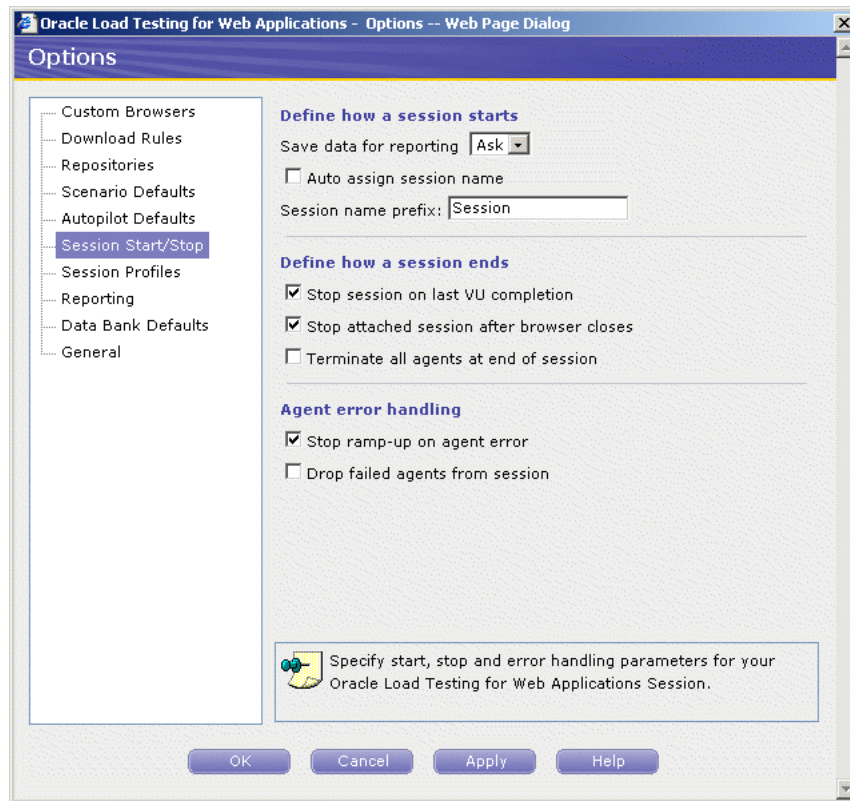
Notice that each profile in the **Scenario Profiles** list can have a different set of attributes.

6.4.2 Saving Data for Reporting

The data generated by a Oracle Load Testing for Web Applications Autopilot session can be saved to the Oracle Load Testing for Web Applications database for post-session analysis. The Session Start/Stop options let you specify if Oracle Load Testing for Web Applications should save the data.

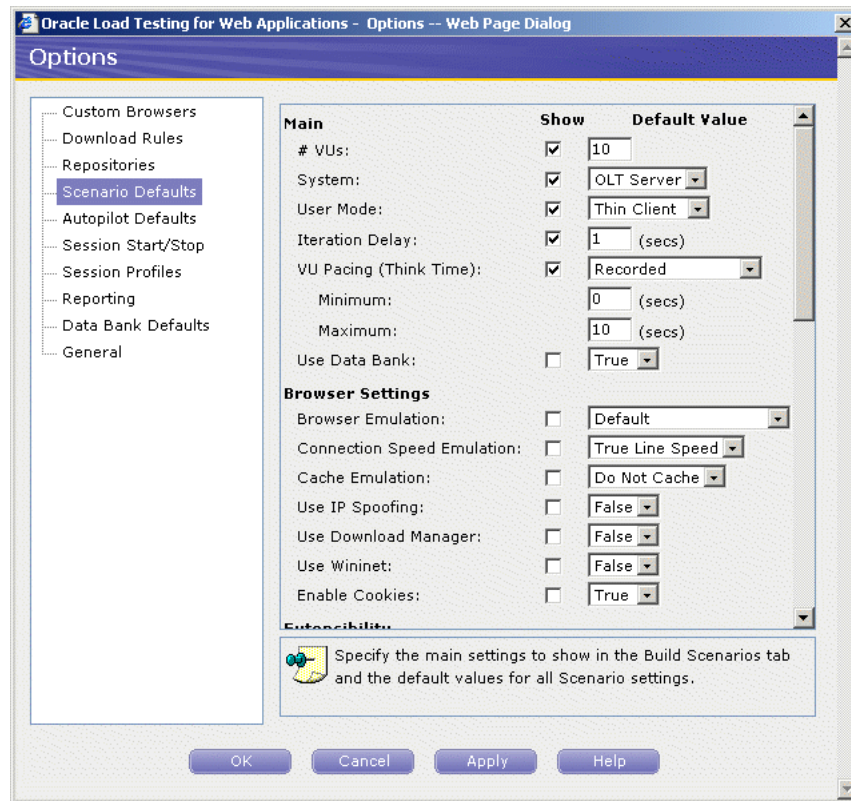
13. Select **Options** from the **Tools** menu and then select **Session Start/Stop**.

Figure 6–17 Session Start/Stop Options Dialog Box



14. Set the **Save data for reporting** option to Yes.
15. Select the **Terminate all agents at end of session** checkbox.
16. Select **Scenario Defaults**.

Figure 6–18 Scenario Defaults Options Dialog Box



17. Set **View All Responses** in the VU Display section to Always.
18. Scroll down the screen and set **Auto generate timers for all resources** in the Reporting section to True.
19. Click **OK**.

6.4.3 Saving the Scenario

20. Select **Save As** from the **Scenario** menu.
21. Leave the filename as LoadTest1 and click **OK**.

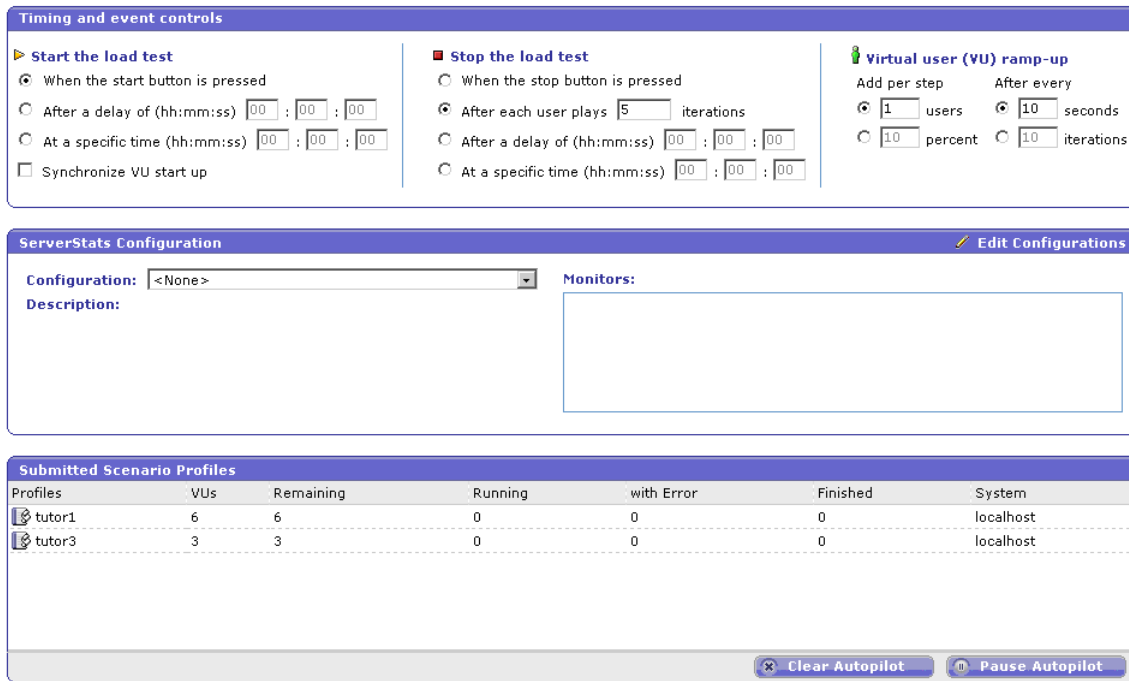
6.5 Example 5: Running Multiple Profiles

This example shows how to use Oracle Load Testing for Web Applications to run multiple Scenario profiles with different amounts of virtual users and how to view statistical and performance information.

6.5.1 Running the Scenario Profiles Using Autopilot

1. Make sure the Scenario from the previous example is still shown in the Build Scenarios tab.
2. Click the **Add to Autopilot** button on the Scenario tab or the toolbar.
3. Oracle Load Testing for Web Applications automatically opens the Set Up Autopilot tab with the tutor1 and tutor3 Scenario Profiles listed in **Submitted Scenario Profiles** list.

Figure 6–19 Set Up Autopilot Options



4. Select When the stop button is pressed in the **Stop the load test** group of the Set Up Autopilot tab.
5. Enter 3 in the edit box next to # **users** under **Add per step** in the **Virtual User (VU) Ramp-up** section.
6. Enter 5 in the edit box next to # **iterations** under **After every** in the **Virtual User (VU) Ramp-up** section.
7. In **ServerStats Configuration** section, select Tutorial from the Configuration drop down list to add the configuration to the load test.
8. Click **Save** to save the Rampup Specification in the Scenario file.
9. Click the **Run test** button on the Oracle Load Testing for Web Applications toolbar.
10. Oracle Load Testing for Web Applications opens the Save Session data dialog box.
11. Click **Ok**.

The Save Session data dialog box appears because we used the Ask setting in the **Session Start/Stop** options (select **Options** from the **Tools** menu). You can bypass this dialog box and use automatic or default values when running virtual users under routine testing conditions by changing the **Session Start/Stop** options.

12. Watch as the Autopilot starts running the tutor1 and tutor3 Visual Scripts as virtual users. Notice also that tutor3 is playing back records from the Data Bank.

Figure 6–20 Virtual User Grid

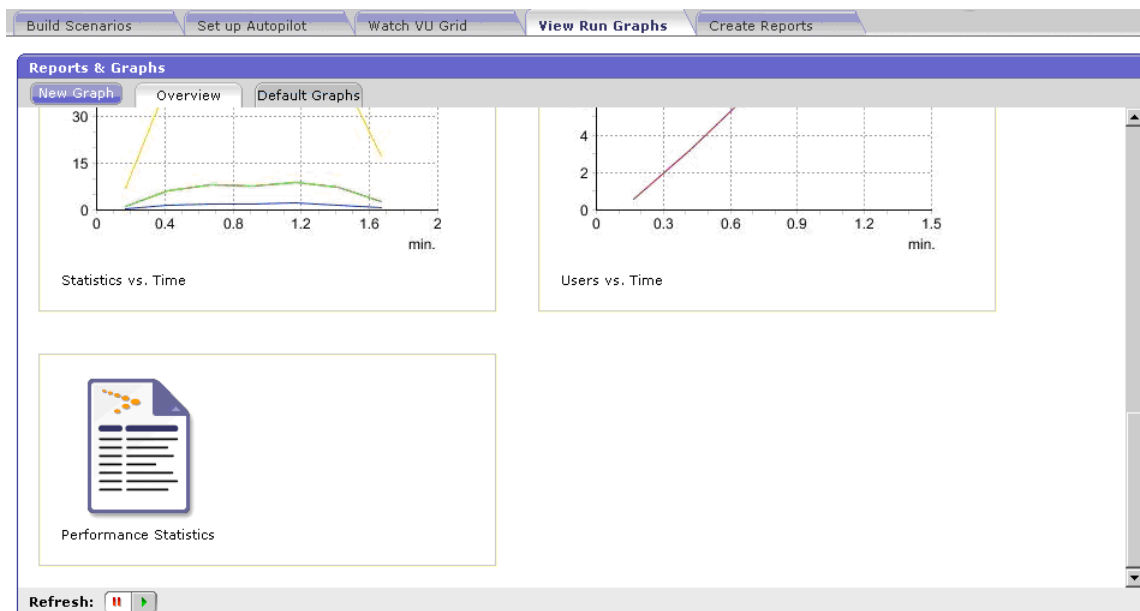
VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Page	System	Data Bank	Current Error	Previous Error
1	tutor3	Running	9		24.876	[2] Search - Home SuperStore Inc.	localhost	Record 5:Phones		
2	tutor1	Think time delay	49		9.463	[2] Kitchens - Home SuperStores Inc.	localhost			
3	tutor1	Running	48		8.892	[4] Electronics - Home SuperStores Inc.	localhost			
4	tutor3	Iteration delay	5		31.816	[3] Results - Home SuperStore Inc.	localhost	Record 4:Cabinets		
5	tutor1	Think time delay	28		9.594	[2] Kitchens - Home SuperStores Inc.	localhost			
6	tutor1	Iteration delay	29		9.053	[4] Electronics - Home SuperStores Inc.	localhost			
7	tutor3	Starting					localhost			
8	tutor1	Think time delay	2		8.513	[2] Kitchens - Home SuperStores Inc.	localhost			
9	tutor1	Think time delay	2		7.37	[2] Kitchens - Home SuperStores Inc.	localhost			

Initially, the Autopilot starts only three virtual users. After the first three have completed five iterations, the Autopilot starts another three virtual users. Once the second three virtual users have completed five iterations, the remaining three virtual users start. The **Virtual User (VU) Ramp-up** options of the Autopilot let you control the rate at which virtual users start running.

6.5.2 Viewing Performance Statistics

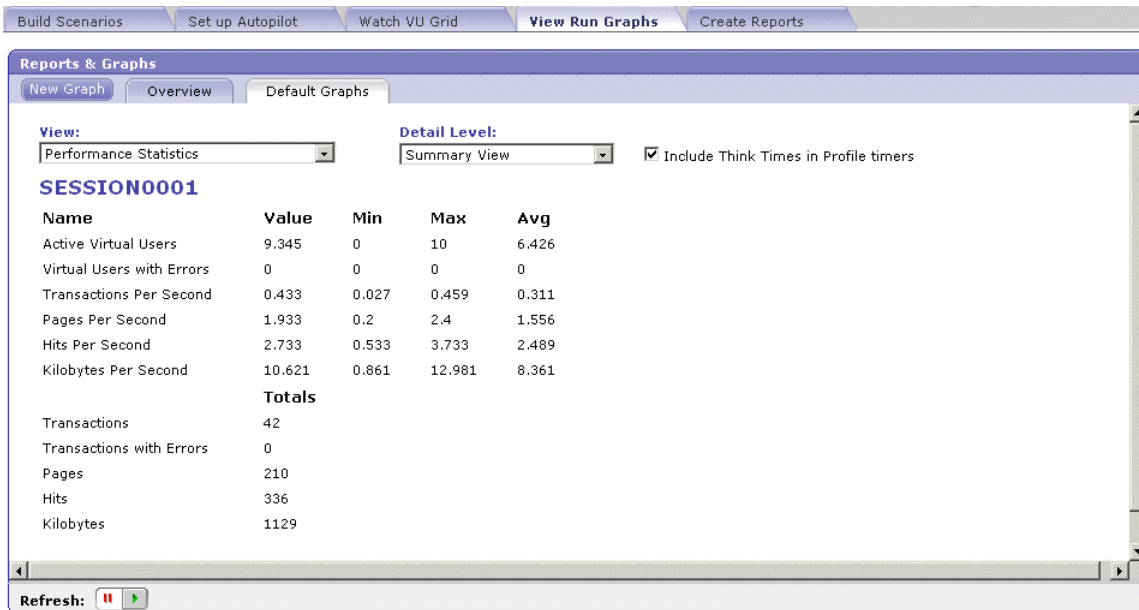
Oracle Load Testing for Web Applications automatically displays run time graphs in the View Run Graphs tab. Scroll to the bottom of the window until you see the Performance Statistics icon.

Figure 6–21 View Run Graphs Tab



Click on the icon to view the Performance Statistics report. The Performance Statistics window shows a summary of the performance data for the running virtual users.

Figure 6–22 Performance Statistics Report



The statistics show the values for the following performance categories:

<Session Name> Current

- **Active Virtual Users** - the number of virtual users currently running in the Autopilot.
- **Virtual Users with Errors** - the number of virtual users with errors.
- **Transactions Per Second** - the number of times the virtual user played back the Visual Script per second.
- **Pages Per Second** - the number of pages returned by the server per second. A "page" consists of all of the resources (i.e. page HTML, all images, and all frames) that make up a Web page.
- **Hits Per Second** - the number of resource requests to the server per second. Each request for a page, individual images, and individual frames is counted as a "hit" by Oracle Load Testing for Web Applications. If Oracle Load Testing for Web Applications does not request images from the server (as specified in the Download Manager), images are not included in the hit count. The **Hits Per Second** and **Pages Per Second** counts will be the same if images are not requested and there are no frames in the page.
- **Kilobytes Per Second** - the number of kilobytes transferred between the server and browser client per second.

<Session Name> Totals

- **Transactions** - the total number of times the virtual user played back the virtual user profile.
- **Transactions with Errors** - the total number of virtual user profile iterations that had errors.
- **Pages** - the total number of number of pages returned by the server.

- **Hits** - the total number of resource requests to the server.
- **Kilobytes** - the total number of kilobytes transferred between the server and browser client.

Performance by Profile and Timer

- **<Profile Name>** - the latest, minimum, maximum, and average performance for the virtual user profile in seconds.
- **<Timer Name>** - the latest, minimum, maximum, and average performance for the server response timers in seconds. Server Response timers are added to Visual Scripts using Oracle Functional Testing for Web Applications.

Performance by Profile and VUs

- **<Profile Name> # VUs** - shows the time it took to run the virtual user profile with the indicated number of virtual users running. When ramping up virtual users, Performance by Profile and VUs values are added when additional virtual users start running. Once additional Performance by Profile and VUs values are added, the previous Performance by Profile and VUs values are no longer updated. For example, the statistics show elapsed time values for each profile for three, six, and nine virtual users. The <profile name> 3 VUs values are updated only while three virtual users are running. Once the Autopilot ramps up to run six virtual users, the <profile name> 3 VUs values stop updating and the <profile name> 6 VUs values are added and are updated while six virtual users are running. Once the Autopilot ramps up to run nine virtual users, the <profile name> 6 VUs values stop updating and the <profile name> 9 VUs values are added and are updated while nine virtual users are running.

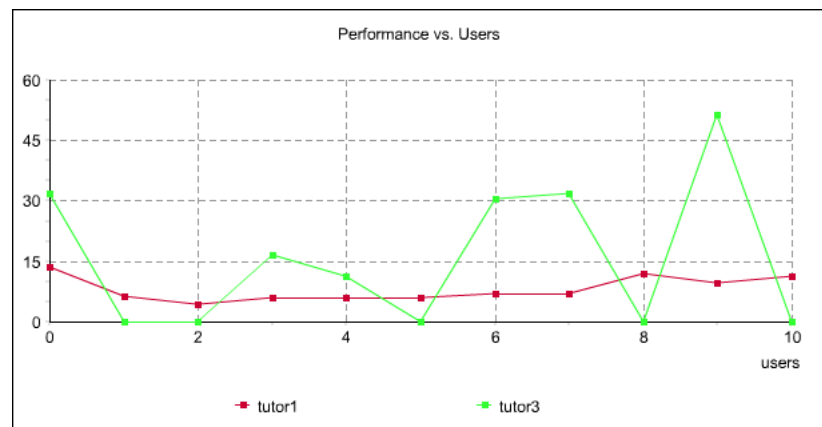
6.5.3 Viewing Graphs

13. Click the Overview tab in the View Run Graphs tab.

Oracle Load Testing for Web Applications provides several types of graphs that show performance, error, and statistical information for the running virtual users. Click on the graph in the Overview tab to view a larger image in the Default Graphs tab.

14. Select the Performance Vs. Users in the **View** dropdown of the Default Graphs tab.

Figure 6–23 Performance Vs. Users Report



This graph shows the average run time for the number of running virtual users in each profile. The plot points represent the Autopilot rampup of virtual users. In

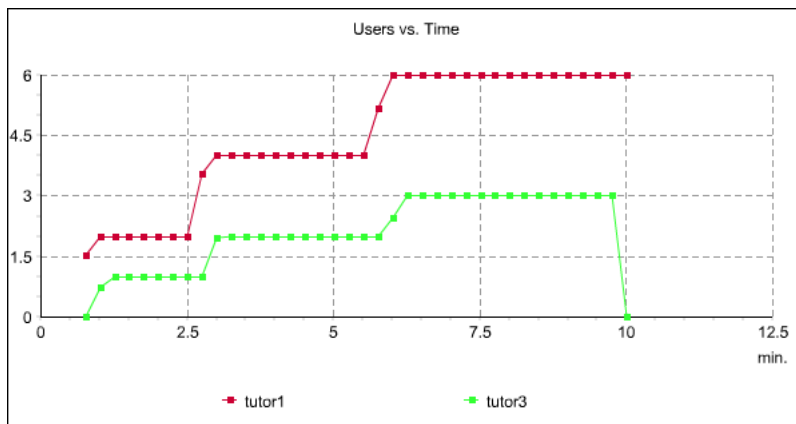
the above graph, the first plot points for each profile shows the average run time while three virtual users were running. Once the Autopilot ramps up to run six virtual users, the plot points for three virtual users are no longer updated.

The second plot points show the average run time while six virtual users were running. Once the Autopilot ramps up to run nine virtual users, the plot points six virtual users are no longer updated.

The third plot points show the average run time while nine virtual users are running. In this example, nine virtual users is the total number of virtual users the Autopilot ramps up to run. The third plot points will be updated continuously while the nine virtual users are running.

15. Select the Users Vs. Time in the **View** dropdown of the Default Graphs tab.

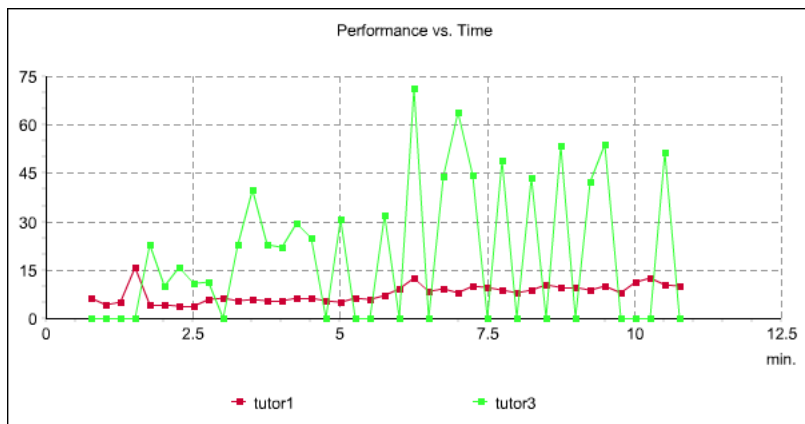
Figure 6–24 Users Vs. Time Report



This graph shows the relative time when the virtual users for each profile started running. The graph represents the Autopilot ramp up times and the number of virtual users ramped up for each profile.

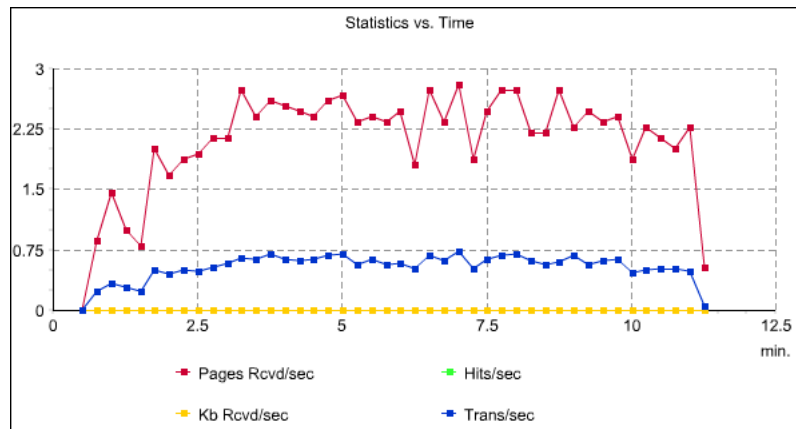
16. Select the Performance Vs. Time in the **View** dropdown of the Default Graphs tab.

Figure 6–25 Performance Vs. Time Report



This graph shows the average run time for the active virtual users running each profile over time.

17. Select the Statistics Vs. Time in the **View** dropdown of the Default Graphs tab.

Figure 6–26 Statistics Vs. Time Report

This graph shows averages for virtual user hits, pages, transactions, and Kilobytes per second over time.

The error graphs show percentages of errors vs. virtual users over time.

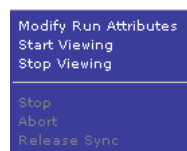
6.6 Example 6: Controlling Virtual Users

This example shows how to modify individual virtual user attributes, view actions, and stop and abort virtual users in Oracle Load Testing for Web Applications.

1. Make sure the virtual users from Example 3 are still running.

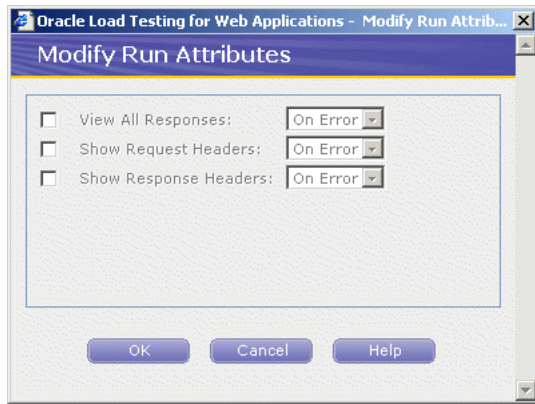
6.6.1 Modifying the Run Attributes

2. Click on any virtual user in the virtual user grid.
3. Click the right mouse-button to open the popup menu.

Figure 6–27 Virtual User Shortcut Menu

4. Select **Modify Run Attributes**. Oracle Load Testing for Web Applications opens a dialog box for changing the run attributes for the selected virtual user.

Figure 6–28 Modify Run Attributes Dialog Box



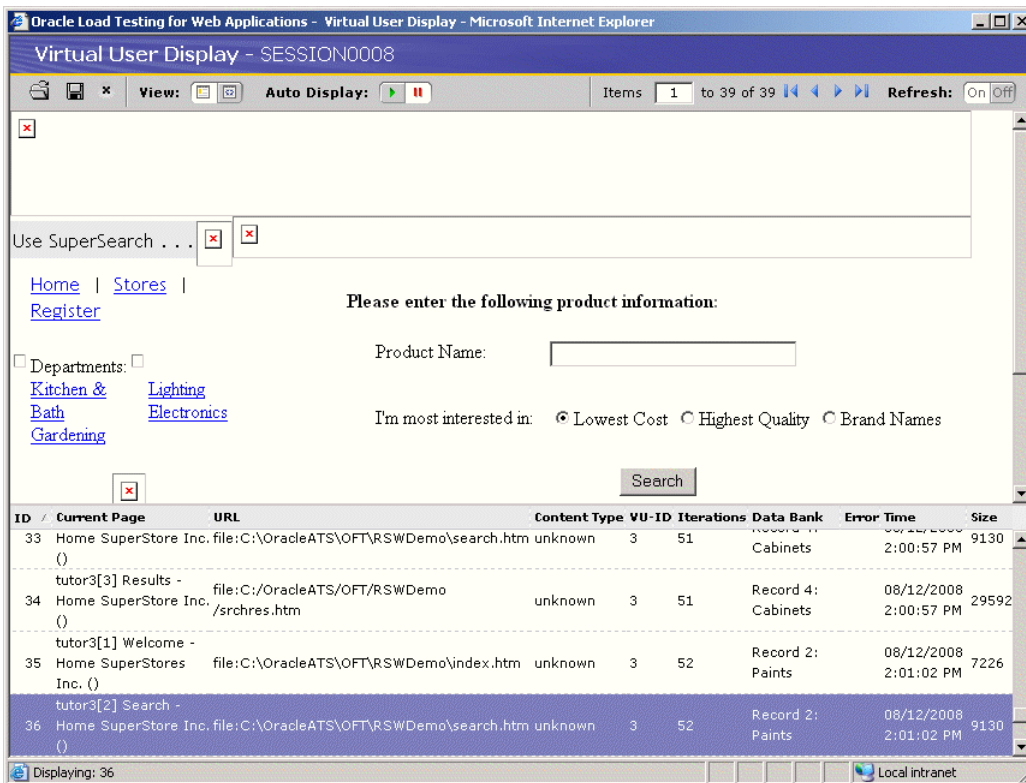
You can change the attributes of each virtual user individually.

5. Click **Cancel** to close the dialog box.

6.6.2 Viewing Virtual User Actions

6. Select **VU Display** from the **Tools** menu or you can also use the right-click popup menu from the virtual user grid. Oracle Load Testing for Web Applications opens a browser window in which you can view the actions of the virtual user.

Figure 6–29 Virtual User Display Window



7. Click the **Navigate to Previous Page** toolbar button. The viewer shows only the previous page.

8. Click the Navigate to Next Page toolbar button. The viewer shows only the next page.
9. Click the Auto Mode toolbar button. The view shows new pages accessed by the virtual user as they arrive to the viewer.
10. Click the Stop Accepting New Pages toolbar button. The viewer stops accepting pages from the virtual user.
Note: Because of the speed at which new pages arrive in the viewer, it may take a few moments for cached pages to stop appearing.
11. Close the window to exit the viewer.

6.6.3 Stopping an Individual Virtual User

12. Click on any virtual user in the virtual user grid.
13. Click the right mouse-button to open the popup menu.
14. Select **Stop**. Oracle Load Testing for Web Applications stops running the selected virtual user. The virtual user will complete the current Visual Script iteration and then stop.

6.6.4 Aborting an Individual Virtual User

15. Click on any virtual user in the virtual user grid.
16. Click the right mouse-button to open the popup menu.
17. Select **Abort**. Oracle Load Testing for Web Applications aborts running the selected virtual user without completing the current visual script iteration.

6.6.5 Stopping All Virtual Users

18. Click the Stop toolbar button to stop all virtual users. The virtual users will complete the current visual script iteration and then stop.

6.6.6 Aborting All Virtual Users

19. Click the Abort toolbar button to abort all virtual users. The virtual users will abort the virtual user without completing the current visual script iteration.

6.7 Example 7: Generating Reports

This example explains the automatic report generation features of Oracle Load Testing for Web Applications. The data collected by Oracle Load Testing for Web Applications and Oracle Load Testing for Web Applications ServerStats while the Autopilot is running virtual users is saved to a database when the **Save Data for Reporting** option in the Oracle Load Testing for Web Applications Session Start/Stop options is set to Yes or Ask. You can use Oracle Load Testing for Web Applications to analyze the data and generate a variety of graphs and reports.

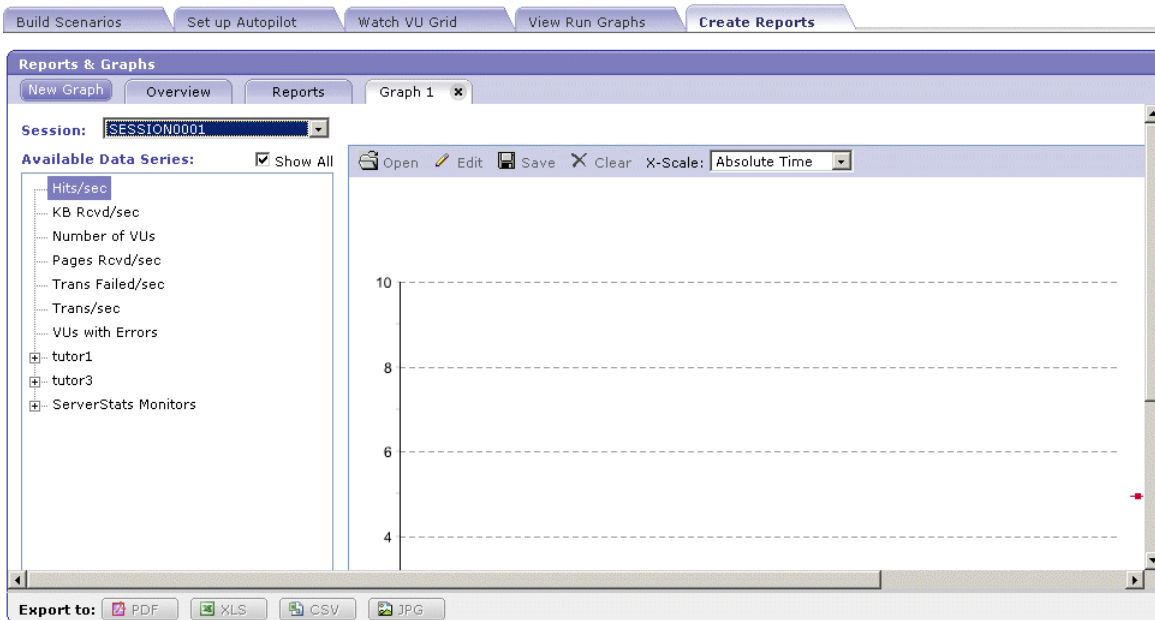
6.7.1 Generating Reports from Oracle Load Testing for Web Applications

This example shows how to use previously saved data to create custom reports and to view Session and Scenario reports.

1. Select the **Create Reports** tab.

2. The Graph 1 tab is displayed with a blank graph. Select Session0001 from the **Session** list. The data categories appear in the Available Data Series list.
3. Click **Show All**.

Figure 6–30 Create Reports Tab Filters



4. Double-click Hits/Sec, kb Rcvd/Sec, Pages Rcvd/Sec, and Trans/sec at the top of the **Available Data Series** field to add the counters to the graph. These are the overall counters. You can also select them and click the **Add Data Series** button.
5. Expand the ServerStats Monitors node then double-click the Oracle Load Testing for Web Applications Server (Processor, % Processor Time, 0) node to add it to the graph.

Figure 6–31 Sample Session Report Graph



The legends show which color line represents which virtual user profile, Visual Script page, and Oracle Load Testing for Web Applications ServerStats counter. The legends for Oracle Load Testing for Web Applications data show the session, the virtual user profile, and the Visual Scripts page in the form session.profile.page[#]. The legends for ServerStats data show the session, counter object, counter instance and counter in the form session.object.instance.counter.

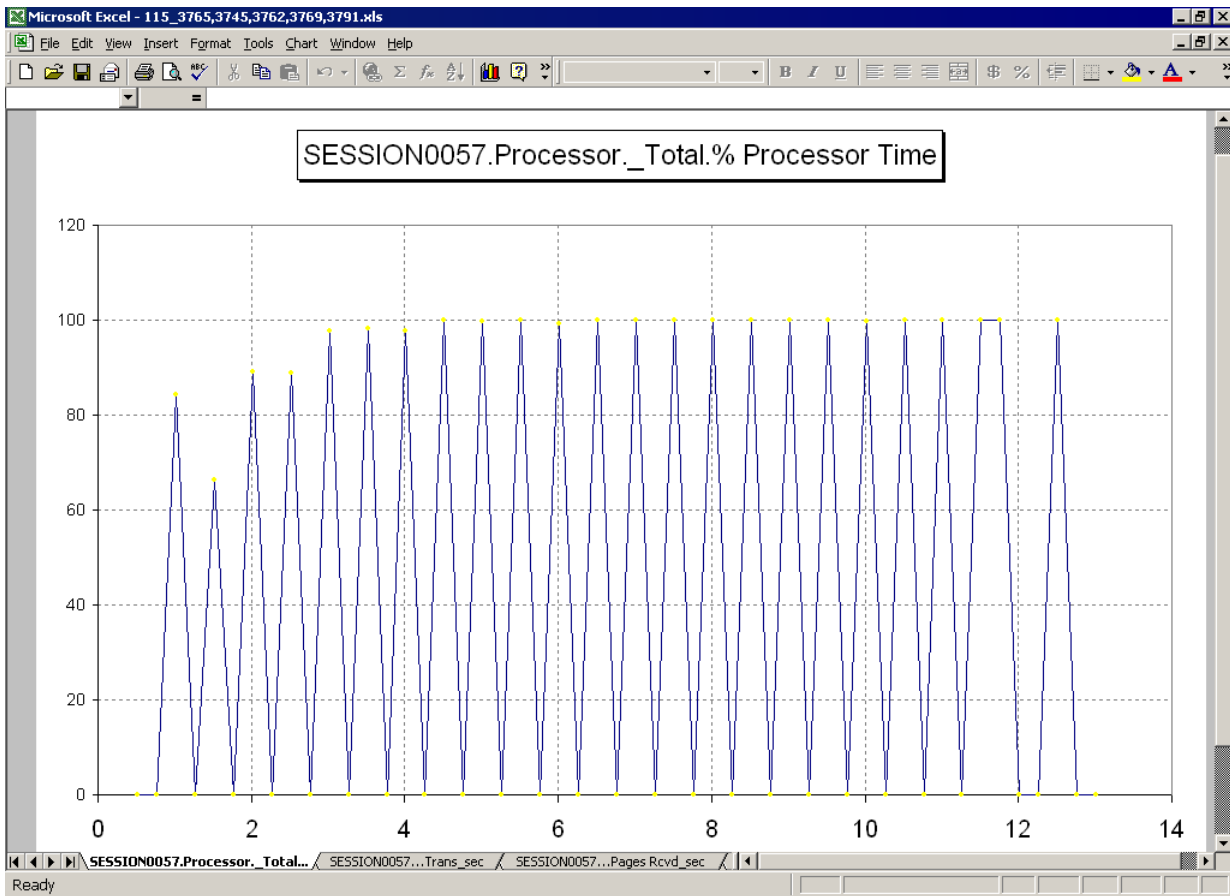
You can export the data to an HTML file, a comma separated value file, or a Microsoft Excel Workbook file.

6.7.2 Opening the Chart in Microsoft Excel

Note: Skip this section if you do not have Microsoft Excel installed on your system.

6. Click the **Export to Excel** button.
7. The File Download dialog box is displayed. Click **Save**.
8. The Save As dialog box is displayed. Select a location to save the report and click **Save**.
9. The Download Complete dialog box is displayed. Click **Open**.

Figure 6–32 Sample Excel Report Graph



10. The workbook contains a chart sheet and a worksheet. You can use the features and capabilities of Microsoft Excel to change the chart format.
11. Click the Data Table tab to view the actual data values.

Figure 6–33 Sample Excel Data Table Report

1	A	B	C	D	E	F	G
	Time	SESSION0057...Hits/sec	SESSION0057...Kb Rcvd	SESSION0057...Pages Rcvd/sec	SESSION0057...Trans/sec	SESSION0057.Processor	Total.% Processor
2	0.516667	0	0	0	0	0	0
3	0.766667	0	0	0.866666675	0.239476457	0	0
4	1.016667	0	0	1.466666698	0.335020155	84.19095612	0
5	1.266667	0	0	1	0.281544954	0	0
6	1.516667	0	0	0.800000012	0.232386635	66.43212891	0
7	1.766667	0	0	2	0.489549309	0	0
8	2.016667	0	0	1.666666627	0.439875633	89.0617218	0
9	2.266667	0	0	1.866666675	0.48924464	0	0
10	2.516667	0	0	1.933333278	0.483719409	88.9006424	0
11	2.766667	0	0	2.133333445	0.531117678	0	0
12	3.016667	0	0	2.133333445	0.588385761	97.65746307	0
13	3.266667	0	0	2.733333349	0.64770627	0	0
14	3.516667	0	0	2.400000095	0.637961388	98.34191132	0
15	3.766667	0	0	2.599999905	0.692626715	0	0
16	4.016667	0	0	2.533333302	0.634567797	97.81159973	0
17	4.266667	0	0	2.466666698	0.605970085	0	0
18	4.516667	0	0	2.400000095	0.625144482	99.90200043	0
19	4.766667	0	0	2.599999905	0.672647893	0	0
20	5.016667	0	0	2.666666746	0.688855529	99.66309357	0
21	5.266667	0	0	2.333333254	0.566414356	0	0
22	5.516667	0	0	2.400000095	0.630684972	100	0
23	5.766667	0	0	2.333333254	0.568952858	0	0
24	6.016667	0	0	2.466666698	0.575444639	99.23466492	0
25	6.266667	0	0	1.799999952	0.516341984	0	0
26	6.516667	0	0	2.733333349	0.673134208	100	0
27	6.766667	0	0	2.333333254	0.608788192	0	0
28	7.016667	0	0	2.799999952	0.72968665	100	0
29	7.266667	0	0	1.866666675	0.511885405	0	0
30	7.516667	0	0	2.466666698	0.623806238	100	0
31	7.766667	0	0	2.733333349	0.672902584	0	0
32	8.016667	0	0	2.733333349	0.696171105	100	0
33	8.266667	0	0	2.200000048	0.608623266	0	0
34	8.516667	0	0	2.200000048	0.56425792	100	0
35	8.766667	0	0	2.733333349	0.590327859	0	0

12. Row one contains the counter names. Subsequent rows contain the actual data values for the chart.

13. Select **Exit** from the **File** menu to close Microsoft Excel.

6.7.3 Viewing Scenario and Session Reports

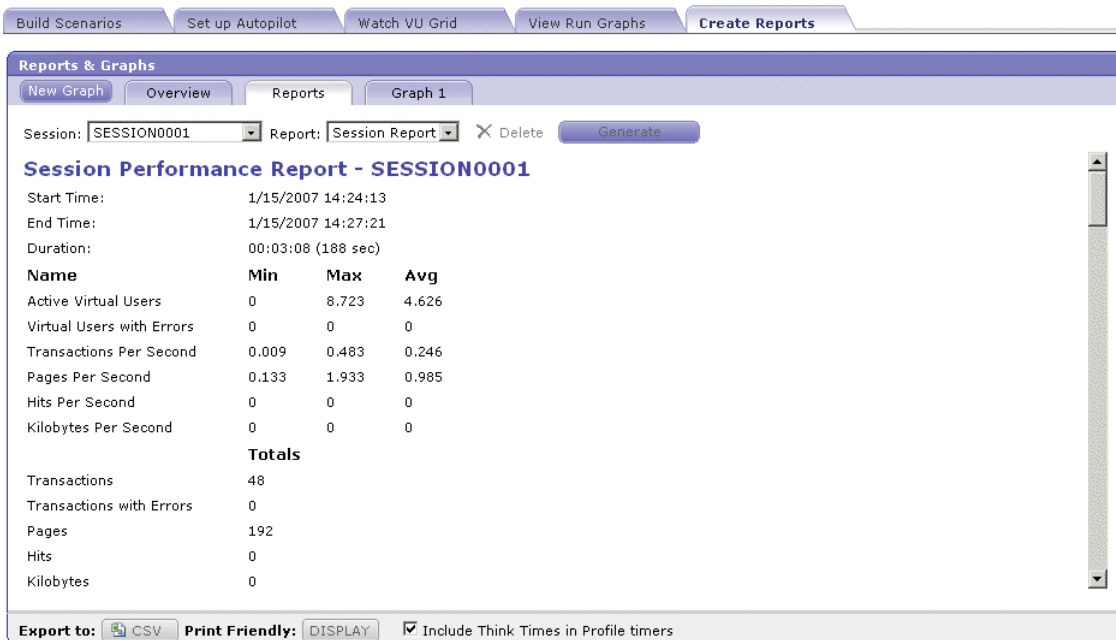
The report shows the current and total performance over time for the Oracle Load Testing for Web Applications scenario. The report also shows the Oracle Load Testing for Web Applications scenario settings used for the session.

14. Oracle Load Testing for Web Applications also generates textual reports for Oracle Load Testing for Web Applications Scenario settings and Oracle Load Testing for Web Applications and Oracle Load Testing for Web Applications ServerStats session data.

15. Select the **Reports** tab.

16. Select the session for which you want to view the report from the **Session** dropdown list and click **Generate**. Oracle Load Testing for Web Applications displays the report.

Figure 6–34 Sample Session Performance Report



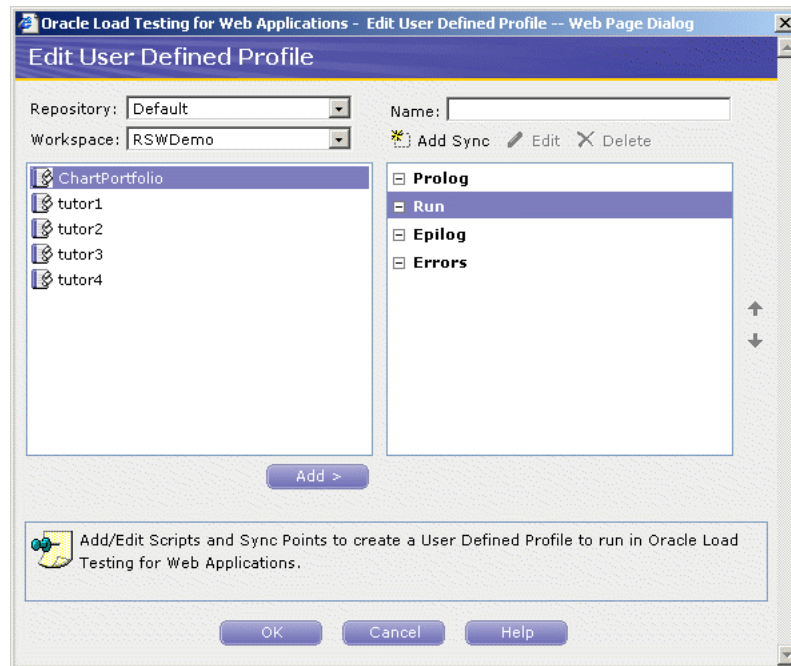
17. Scroll to the end of the Session Performance Report to view the Oracle Load Testing for Web Applications Scenario Report.
18. You can print the report by clicking the **Print Friendly** button and selecting **Print** from the **File** menu in the browser window.

6.8 Example 8: Creating User-Defined Profiles

This example explains how to create user-defined virtual user profiles in Oracle Load Testing for Web Applications.

1. Select **User Defined Profiles** from the **Manage** menu. Oracle Load Testing for Web Applications opens a dialog for managing profiles.
2. Click **New**.

Figure 6–35 Edit User Defined Profiles Dialog Box



3. The dialog box shows the section tree, the available Visual Scripts, and the default synchronization points.
4. Select the workspace where you want to save the profile and enter a name for the profile in the **Name** editbox.
5. The profile sections tree allows you to specify which Visual Scripts and synchronization points to include in the Sections tree of the profile.

Prolog - the Visual Scripts in this section play back only once at the beginning of the Scenario run. An example of what you may add to this section is a login script.

Run - the Visual Scripts in this section iterate over as many times as is specified in the Autopilot. An example of what you may add in this section is the business transaction that you wish to load test.

Epilog - the Visual Scripts in this section play back only once at the end of the Scenario run. An example of what you add to this section is a logoff script.

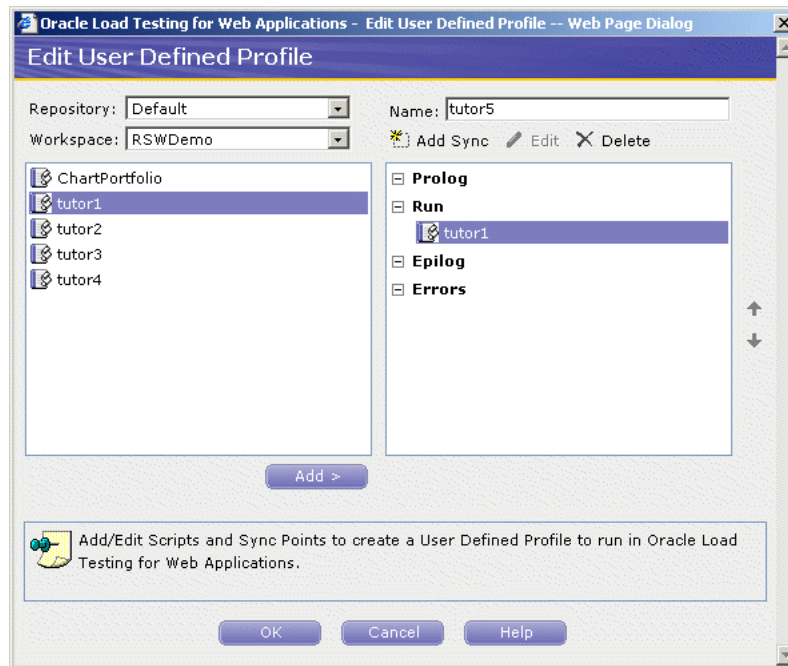
Errors - the Visual Scripts in this section play back only if an error occurs during the Scenario run. An example of what you may add to this section is a visual script that resets your application on an error.

6.8.1 Adding Visual Scripts to the Sections Tree

6. Select the section in the tree where you want to add a Visual Script.
7. Double-click the Visual Script to add to the section or select the script and click the **Add** button.

The Visual Script appears as a node of the tree.

Figure 6–36 Script Added to Run Section of User Defined Profile



8. Repeat steps 4 and 5 to add additional Visual Scripts to the Sections tree.

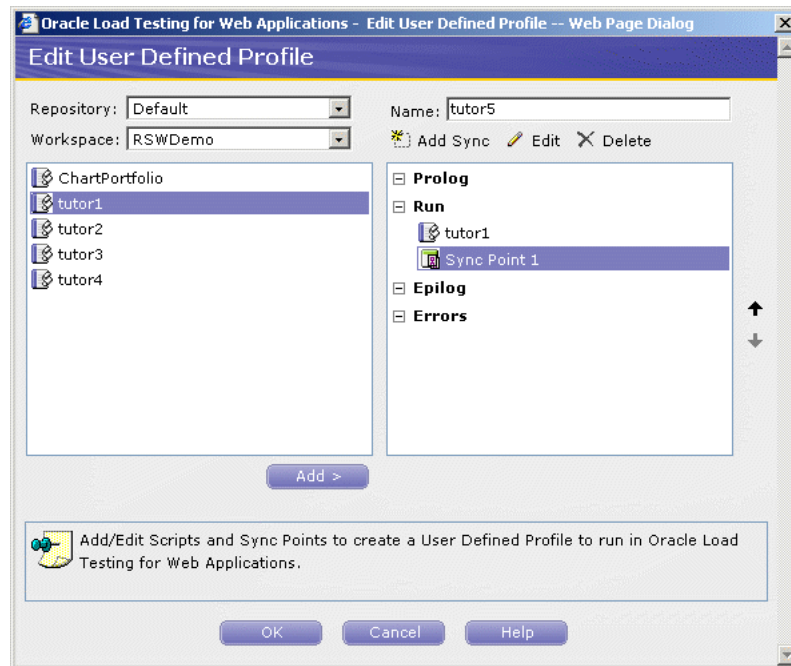
6.8.2 Adding Synchronization Points to the Sections Tree

A sync point allows multiple virtual users to synchronize their actions and interactions with the application under test. Sync points provide the ability to create realistic multi-user situations that may expose resource conflicts such as deadlocks. When you specify a sync point, multiple virtual users executing the script will reach this sync point at various times depending on a number of factors (for example, the speed of the machine).

Sync points cause each virtual user to wait until all virtual users have reached that sync point. Each of the virtual users notifies the master upon reaching the sync point. The master waits for all of the virtual users to notify it and then issues the go-ahead for all the virtual users to continue past that sync point.

9. Select the section in the tree where you want to add a Sync point and click **Add Sync**.

The Sync point appears as a node of the tree.

Figure 6–37 Sync Point Added to Run Section of User Defined Profile

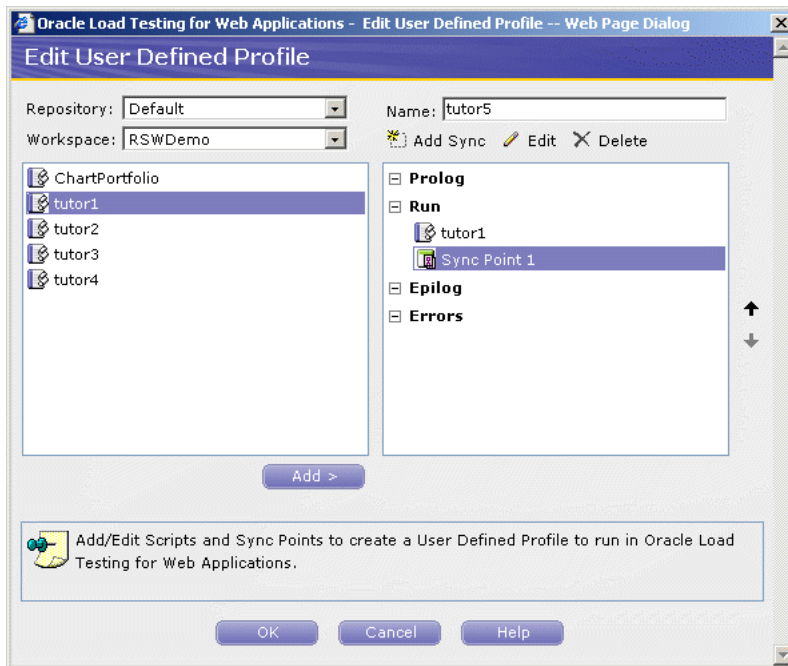
- Repeat steps 7 and 8 to add additional Sync points to the Sections tree.

6.8.3 Moving Items in the Sections Tree

When you have multiple items under any section of the tree, you can move the items up and down within that section.

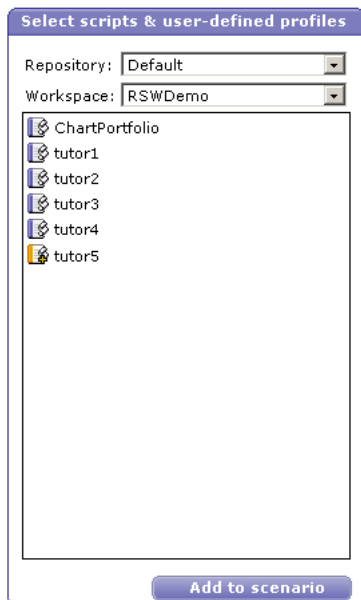
- Select the item to move in a section.
- Click the up or down button as appropriate.

Figure 6–38 Edit User Defined Profile Dialog Box



- 13. Click the **OK** button when you finish defining the profile.
- 14. The new profile appears in the **Select scripts and user-defined profiles** list of the **Build Scenarios** tab.

Figure 6–39 Select Scripts and User-Defined Profiles Pane



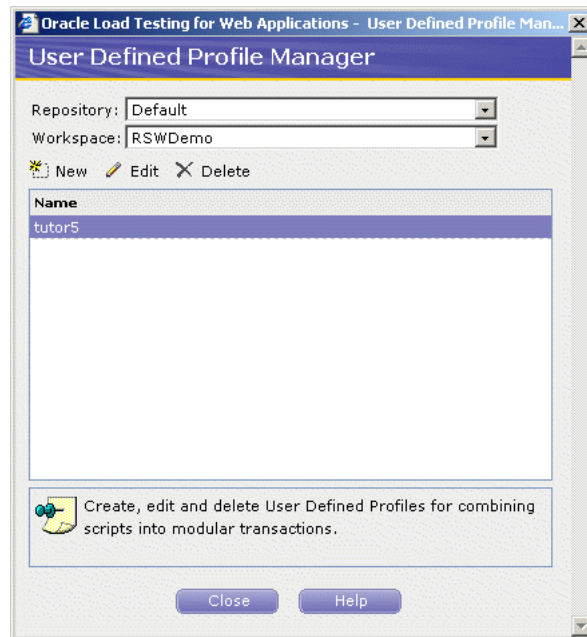
You can include user-defined profiles as part of the Scenario Profiles the same way you use the default profiles.

6.8.4 Editing User-Defined Profiles

After you have created a user-defined virtual user profile, you can make changes to the profile at any time.

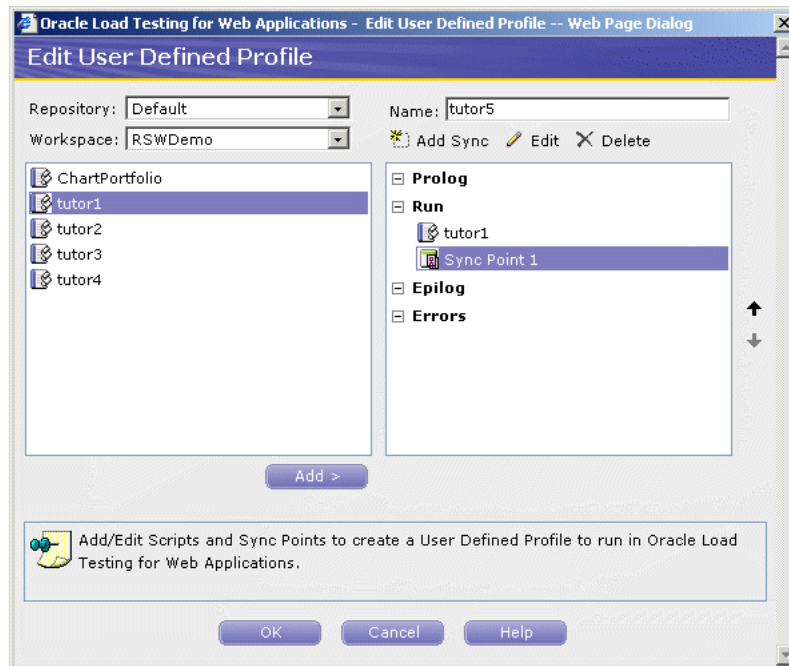
15. Select **User Defined Profiles** from the **Manage** menu.

Figure 6–40 *User Defined Profile Manager Dialog Box*



16. Select the profile you want to edit and click **Edit**. Oracle Load Testing for Web Applications opens a dialog for editing the sections tree of the profile. The dialog box shows the current sections tree, the available Visual Scripts, and the default synchronization points.

Figure 6–41 Edit User Defined Profile Dialog Box



17. Click the Plus icon to expand the nodes of the tree.
18. Use the arrow buttons as necessary to add items to the sections of the tree. Select an item and click **Delete** to remove it from the sections tree.
19. Click the **OK** button when you finish editing the profile.
20. Select **Exit** to close Oracle Load Testing for Web Applications.
21. Click **No** if asked to save the scenario.

This completes the Oracle Load Testing for Web Applications tutorial. See the *Oracle Load Testing for Web Applications User's Guide* for additional information about load testing and using Oracle Load Testing for Web Applications.

Oracle Test Manager for Web Applications Tutorial

This tutorial walks you through the main features of Oracle Test Manager for Web Applications. It consists of the following examples.

- **Adding a Requirement** - describes how to add a requirement.
- **Adding a Test** - describes how to add a both a manual and automated test and how to associate a requirement with a test.
- **Running a Test** - explains how to run both a manual test and an automated test and how to view the results of the automated test.
- **Adding an Issue** - describes how to find and issue, add an issue and associate it with a test.
- **Creating Reports** - explains how to view reports and charts.

The tutorial is designed to be followed sequentially from beginning to end. Many of the examples are interrelated and build upon the steps in previous examples.

7.1 Starting Oracle Test Manager for Web Applications

Note: This section uses the default login credentials from the Oracle Application Testing Suite installation.

To start Oracle Test Manager for Web Applications:

1. Select **Programs** from the **Start** menu and then select **Oracle Test Manager for Web Applications - Web** from the **Oracle Application Testing Suite** menu.
2. Enter **Administrator** as the user name.
3. Enter the password specified during the Oracle Application Testing Suite installation process.
4. Make sure the **Database** is set to Default OTM.
5. Make sure the **View Type** is set to All modules.
6. Click **Login**.

7.2 Opening the Sample Project

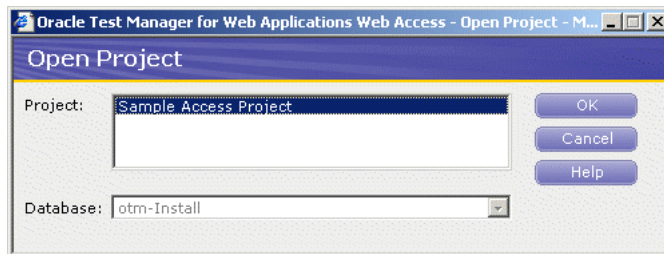
A demonstration project is installed with Oracle Test Manager for Web Applications for testing the sample stock brokerage application Fitch & Mather. This application can be viewed at <http://demo.fmstocks.com>. To open the sample project:

1. Start Oracle Test Manager for Web Applications and log in.

The default installation user names are **Administrator** and **Default** unless changed by an administrator. The password is the password specified during the installation procedure unless changed by an administrator.

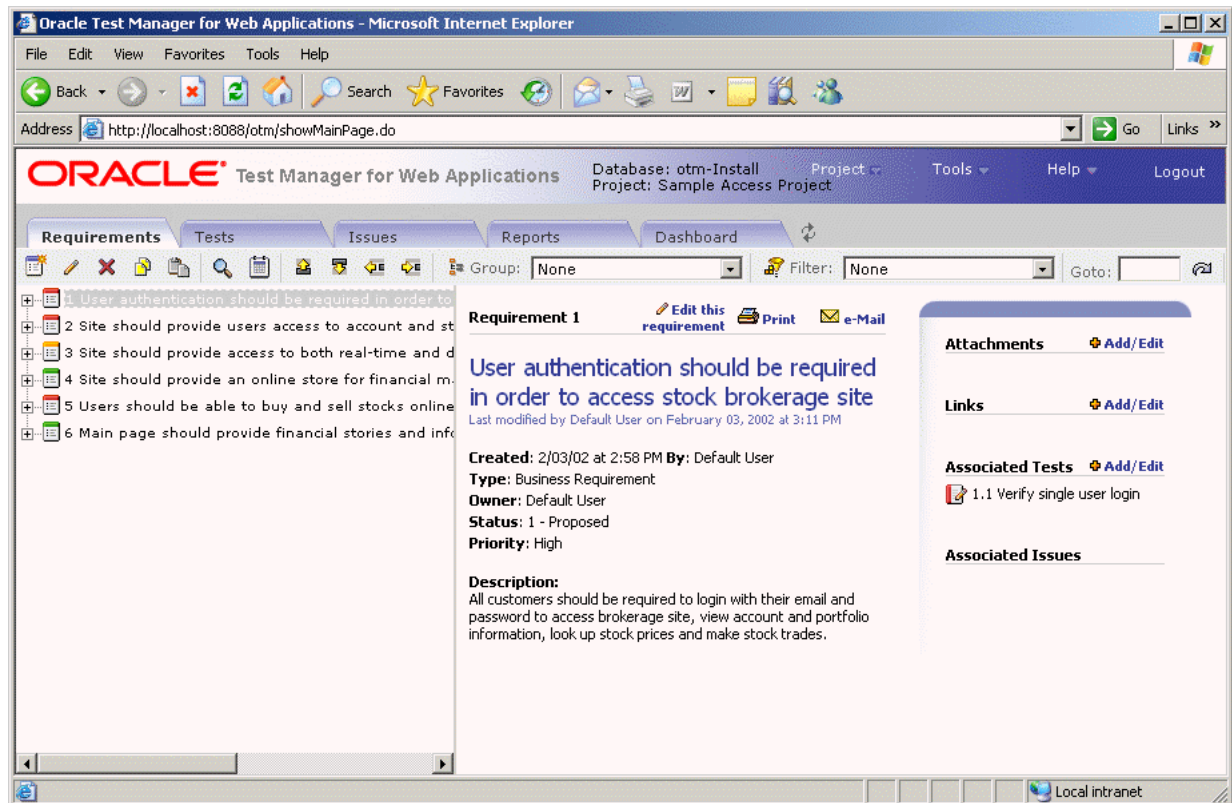
2. Click **Open** from the **Project** menu to display the Open Project dialog box.

Figure 7–1 Open Project Dialog Box



3. Make sure the **otm-install** database is selected. This is the database that was created when you installed Oracle Test Manager for Web Applications.
4. Select the sample database.
5. Click **OK**. The main window appears as follows:

Figure 7-2 Oracle Test Manager for Web Applications Main Window



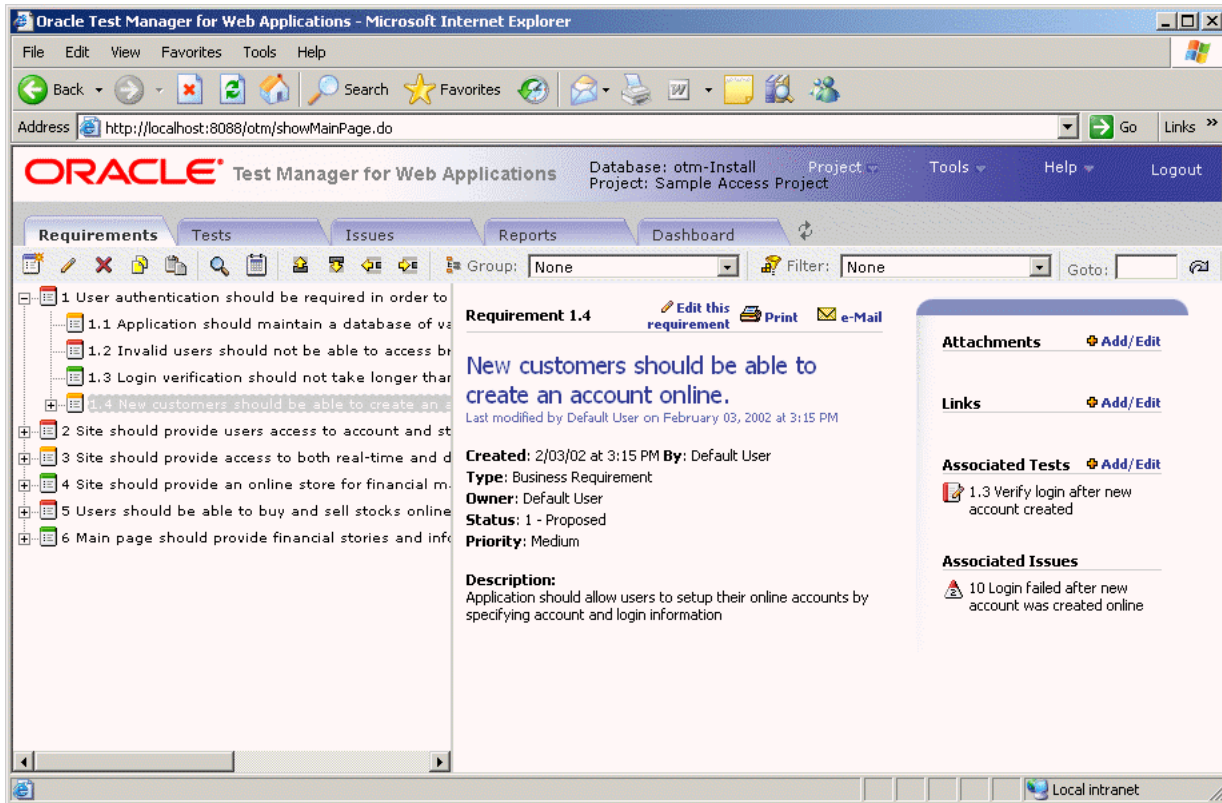
The left pane contains tabs for the three modules that comprise the application. They are requirements, tests, and issues. The detailed information for the selected item is displayed in the right pane. The information displayed may be different from the examples shown in this tutorial if your system administrator has customized the database to add new fields or disable default fields.

7.3 Example 1: Adding a Requirement

This example explains how to add a requirement.

1. Click the **Requirements** tab.
2. Expand item 1 and click item 1.4. The new requirement will be added as item 1.5.

Figure 7-3 Requirements Tab



3. Click **Add** to display the Add Requirement dialog box.

Figure 7-4 Add Requirement Window

Oracle Test Manager for Web Applications Web Access - Add - Microsoft Internet Explorer

Add Requirement

*Name: Save

*Type: -- Select -- Reset

*Owner: -- Select -- Cancel

*Status: -- Select -- Help

*Priority: -- Select --

Description:

* Denotes required fields

Attachment :

File : Browse... Capture >>

Link :

Title :

Link :

ex. http://www.yoursite.com

4. Enter "Users should have access to the chart view of their portfolio" in the Name field.
5. Select a type, owner, and status.
6. Select Medium from the Priority list.
7. Enter "All customers should be able to view their portfolio in the chart view" in the Description field.

Figure 7-5 Add Requirement Window with Sample Data

Oracle Test Manager for Web Applications Web Access - Add - Microsoft Internet Explorer

Add Requirement

*Name: Save

*Type: Reset

*Owner: Cancel

*Status: Help

*Priority:

Description:

** Denotes required fields*

Attachment :

File : Browse... Capture >>

Link :

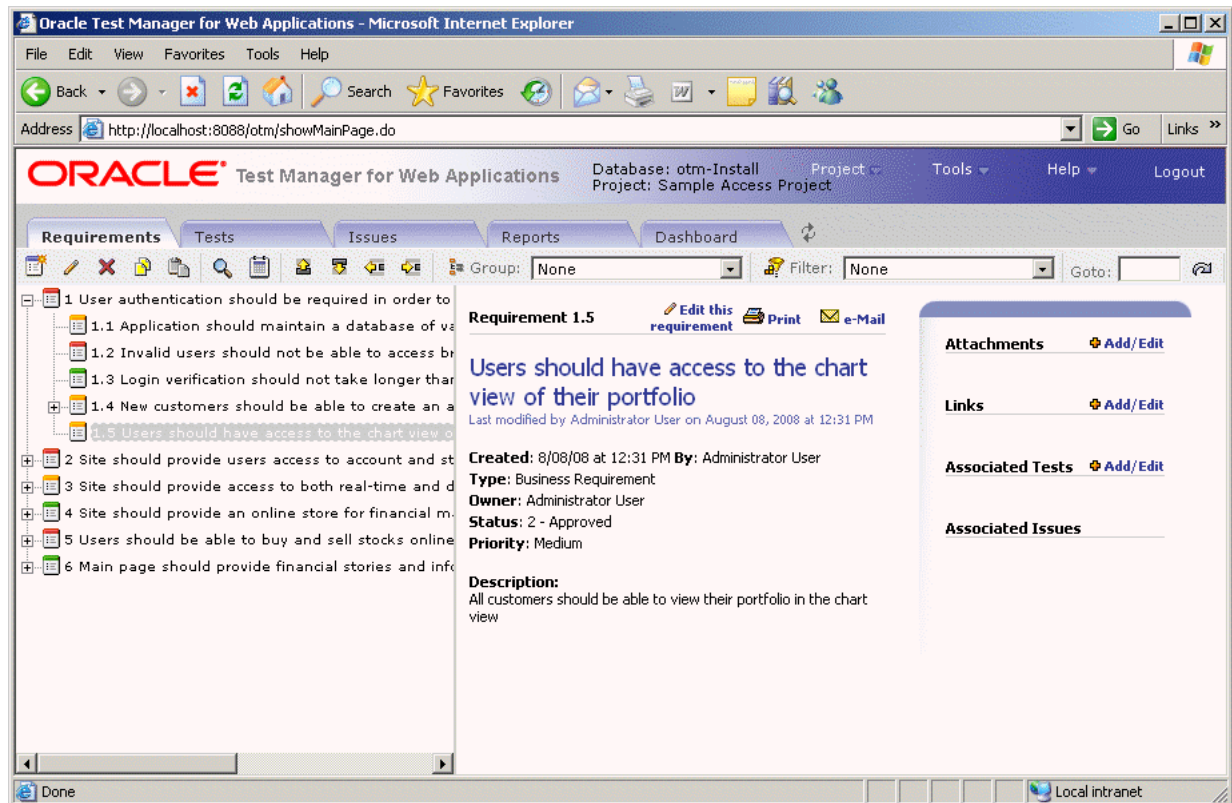
Title :

Link :

ex. http://www.yoursite.com

8. Click **Save**. The new requirement appears in the tree.

Figure 7-6 Requirements Tab with New Requirement Added



7.4 Example 2: Adding a Test

You can add two types of test cases to Oracle Test Manager for Web Applications:

Manual tests - these are tests that allow you to organize your test cases. For each step in the test, you enter the action, expected result, and pertinent comments. When you run the test, the Oracle Test Manager for Web Applications Manual Test Wizard takes you step by step through the test, allowing you to manually execute each test and enter the result.

Automated tests - these are Oracle OpenScript or Oracle Functional Testing for Web Applications tests that can be run automatically without manual intervention. You can run one test, an entire branch of tests, or all automatic tests in the project.

This example has two parts. The first part explains how to add a manual test. The second part explains how to create the same test using Oracle Functional Testing for Web Applications and add it as an automated test.

7.4.1 Adding a Manual Test

This example explains how to add a manual test.

1. Click the **Tests** tab.
2. Click item 1, Login Tests.
3. Click **Add** to display the Add Test dialog box.

Figure 7-7 Add Test Window

*Name:

Type:

Test File :

*Owner:

Functionality:

*Priority:

Description:

** Denotes required fields*

Attachment :

File :

Link:

Title:

Link:

ex. http://www.yoursite.com

4. Enter "Verify customer chart view of portfolio" in the **Name** field.
5. Select **Manual Test** in the **Type** field.
6. Enter "This test verifies that the chart view of the portfolio is accessible" in the **Description** field.
7. Select an owner and priority.

Figure 7-8 Add Test Window with Sample Data

Oracle Test Manager for Web Applications Web Access - Add - Microsoft Internet Explorer

Add Test

*Name: Save

Type: Reset

Test File : Browse... Cancel

*Owner: Help

Functionality:

*Priority:

Description:

* Denotes required fields

Attachment :

File : Browse... Capture >>

Link:

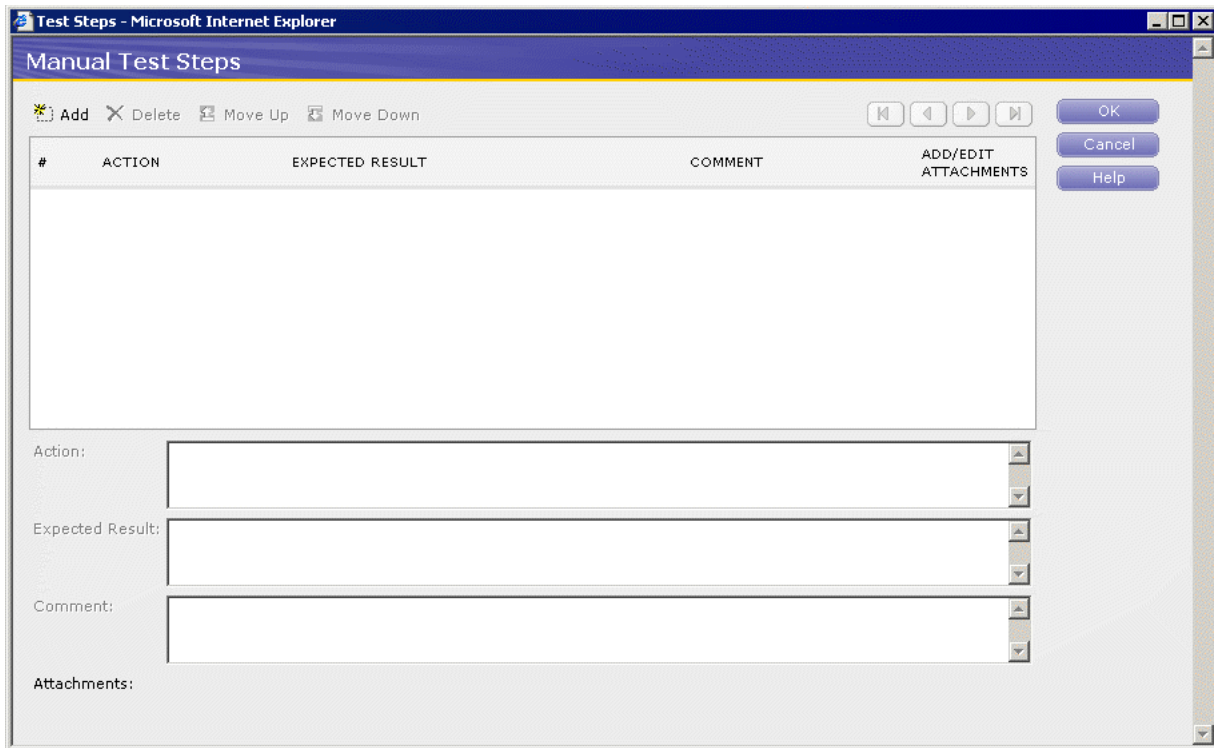
Title:

Link:

ex. <http://www.yoursite.com>

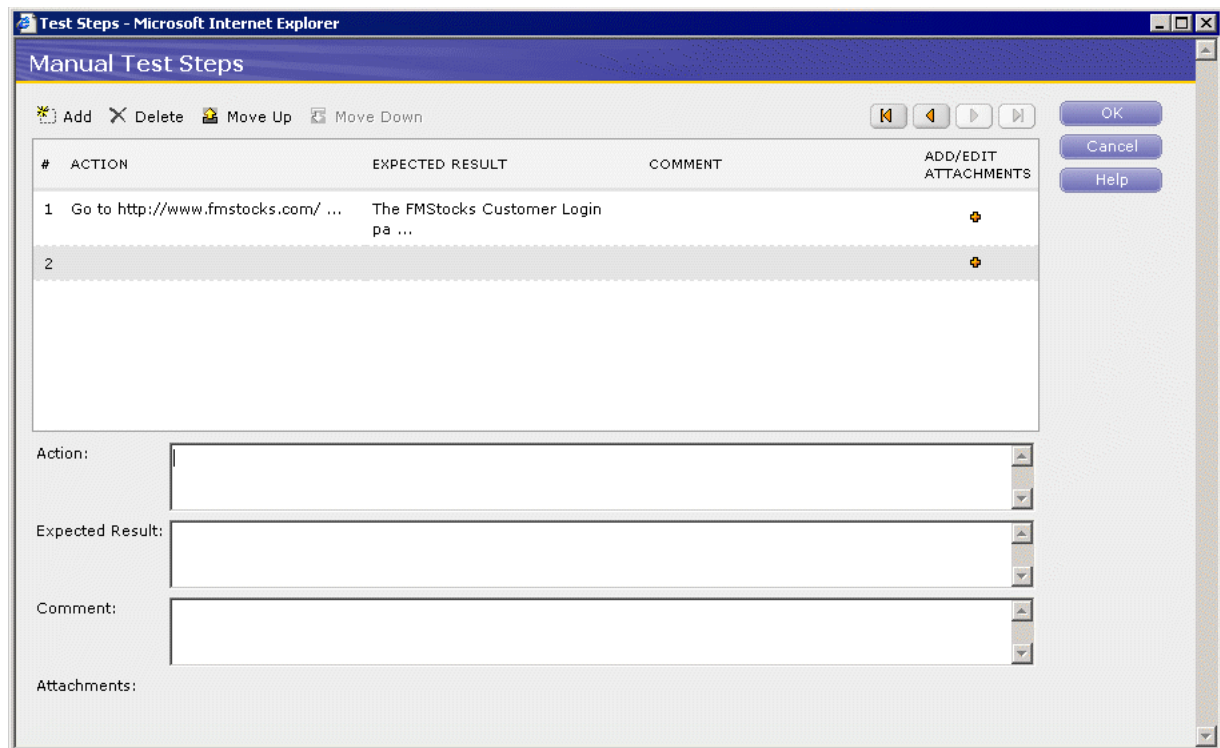
8. Click **Save**. The test is added as item number 2. You are now ready to add test steps.
9. Click **Add/Edit** in the Test Steps section of the right-hand pane to display the Test Steps dialog box.

Figure 7-9 Manual Test Steps Window



10. Click **Add**.
11. Enter, "Go to <http://www.fmstocks.com/fmstocks>" in the **Action** field.
12. Enter, "The FMStocks Customer Login page should be displayed" in the **Expected Results** field.
13. Click **Add** to update the top of the dialog box and to go to the next step.

Figure 7-10 Manual Test Steps Window with Sample Data



14. Enter the following steps, clicking **Add** to start each step:

Step	Action	Expected Result
2	Click Login.	The Welcome to FMStocks page should be displayed.
3	Click the Chart Your Portfolio link	The Your Portfolio page should be displayed with a spreadsheet and graphs.

The Manual Test Steps dialog box shows the entries as entered.

15. Click **OK**.

7.4.2 Adding an Automated Test

Oracle Test Manager for Web Applications can run Oracle OpenScript scripts or Oracle Functional Testing for Web Applications tests that have been saved in the Oracle Test Manager for Web Applications Package format. Package format files have a `.otmPKG` extension. This example explains how to create a test in Oracle Functional Testing for Web Applications, add it to the Tests in Oracle Test Manager for Web Applications, and associate it with the requirement just added.

1. Select **Programs** from the **Start** menu and then select **Oracle Functional Testing for Web Applications** from the **Oracle Application Testing Suite** menu.
2. Go to `http://www.fmstocks.com/fmstocks`.
3. Select **New Script** from the **File** menu.
4. Select **Record** from the **Run** menu and then select **Start** to begin recording the script.

5. Click **Login**.
6. Click the **Chart Your Portfolio** link.
7. Select **Record** from the **Run** menu and then select **Stop** to stop recording the script.
8. Select **Save Script** from the **File** menu.
9. Navigate to the OracleATS/OFT/Default! workspace.
10. Select Oracle Test Manager for Web Applications Package (*.otmPKG) as the format to save the file.
11. Enter `ChartPortfolio` as the name of the script.
12. Click **Save**.
13. Select **Exit** from the **File** menu to exit Oracle Functional Testing for Web Applications and return to Oracle Test Manager for Web Applications.
14. Click the **Tests** tab.
15. Expand Customer Account Tests and click item 3.3.
16. Click **Add** to display the Add Test dialog box.
17. Enter "Verify customer chart view of portfolio" in the Name field.
18. Select **Oracle Functional Testing Script** in the Type field.
19. Click **Browse** in the Test File field.
20. Select `ChartPortfolio.otmPKG`.
21. Select an owner and priority.
22. Enter "This test verifies that the chart view of the portfolio is accessible" in the Description field.

Figure 7-11 Add Test Window with Sample Automated Test

Oracle Test Manager for Web Applications Web Access - Add - Microsoft Internet Explorer

Add Test

*Name:

Type:

Test File :

*Owner:

Functionality:

*Priority:

Description:

** Denotes required fields*

Attachment :

File :

Link:

Title:

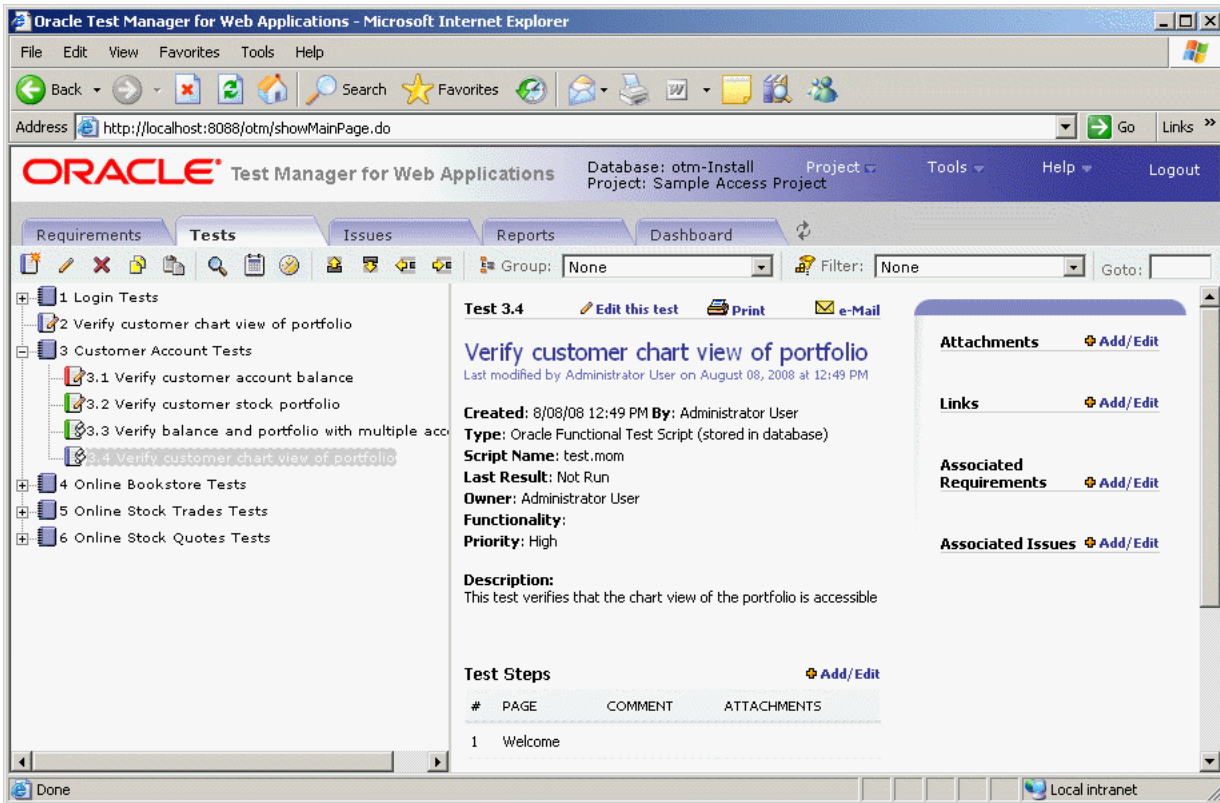
Link:

ex. http://www.yoursite.com

23. Click **Save**.

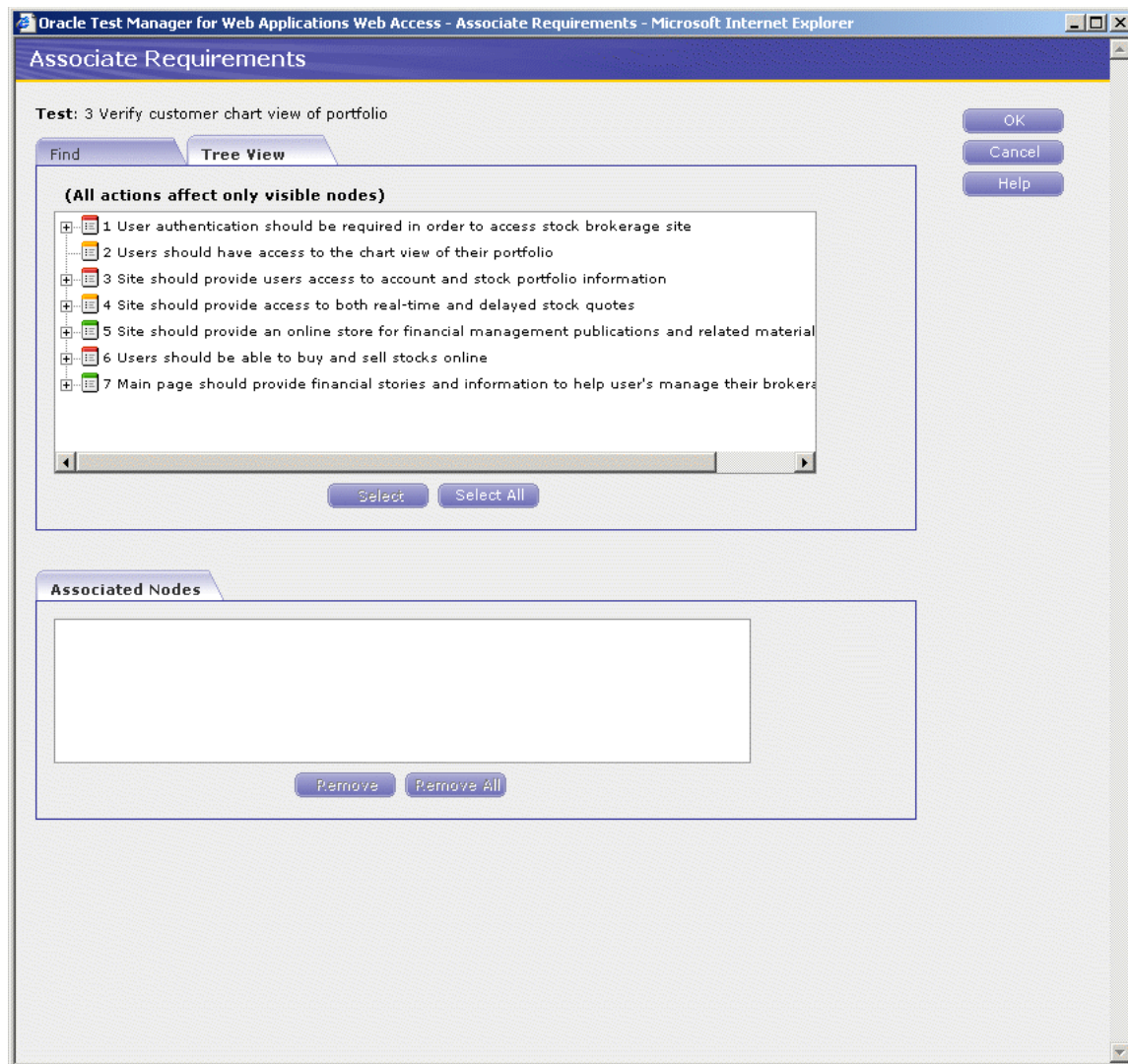
24. The test is added as item 3.4.

Figure 7-12 Tests Tab with New Test Added



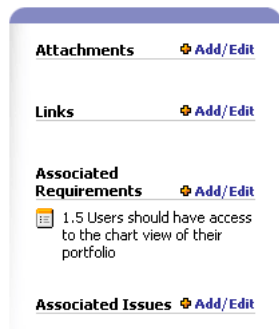
25. Click **Add/Edit** next to Associated Requirements in the right pane, to display the Associate Requirement dialog box.
26. Click the **Tree View** tab.

Figure 7-13 Associate Requirements Window



27. Expand item 1 and click item 1.5.
28. Click **Select**.
29. Click **OK**. The test is now linked to the requirement and is listed on the right-hand side of the screen. The test is also listed in the Associated Tests section of the requirement.

Figure 7–14 Test Associated with Requirement



7.5 Example 3: Running a Test

This example has two parts. The first part explains how to run the manual test created in the previous example. The second part explains how to run the automated test created in the previous example, then how to view the detailed Results Report.

7.5.1 Running a Manual Test

This example explains how to run the manual test entered in the previous example.

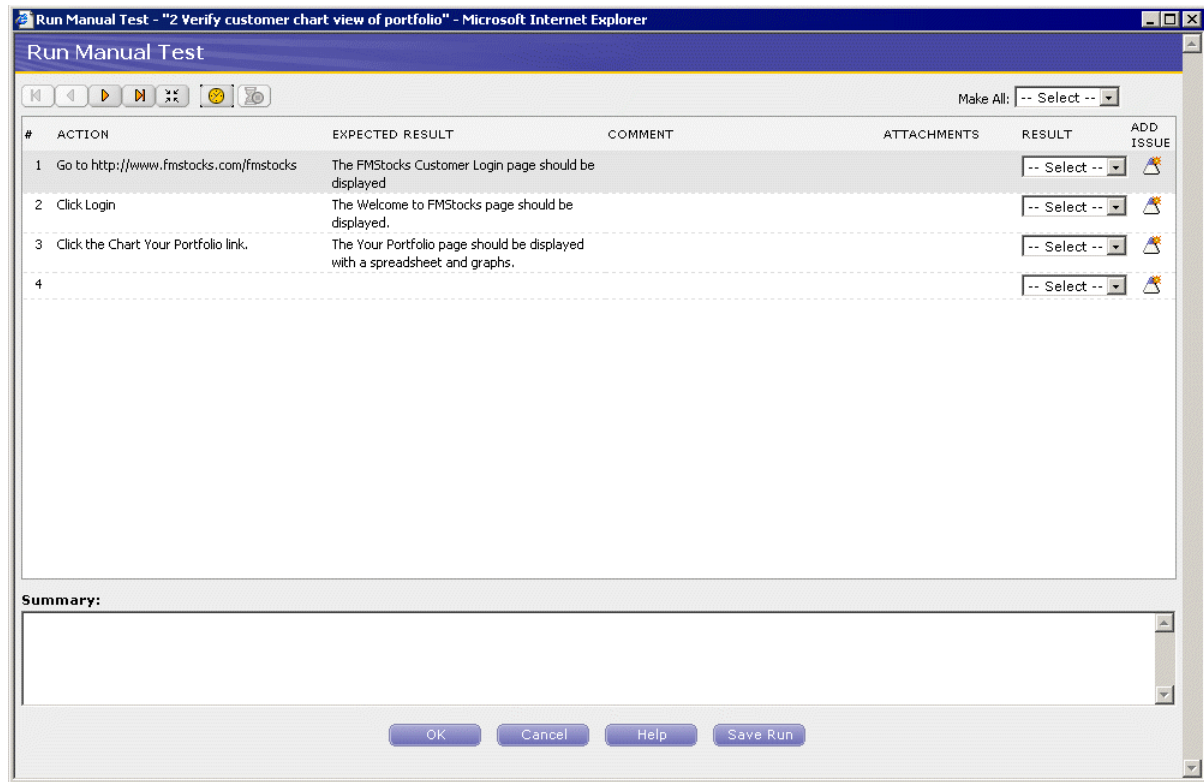
1. Click the Tests tab and select number 2, Verify customer chart view of portfolio.
2. Click **Run this test** in the Run History section of the right-hand pane.

Figure 7–15 Run Test Info Window



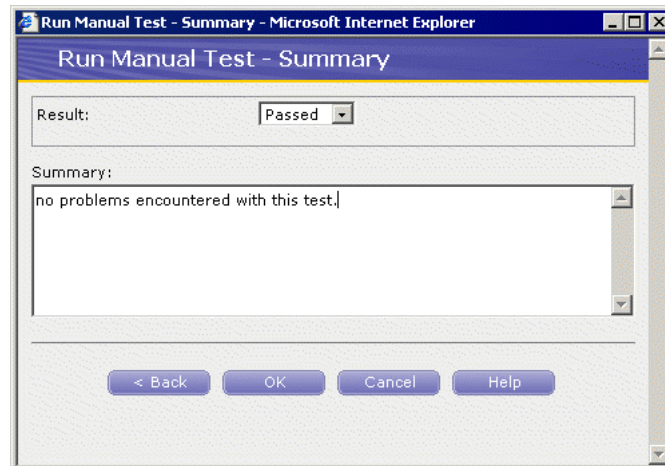
3. Select 1.0 in the **Build** field. This is the version number of the application you are testing.
4. Click **Save**.

Figure 7-16 Run Manual Test Window



5. This dialog box tells you the action to take and the expected result. Open your browser and perform the first action. Select the result **Passed**. Enter comments, when needed, in the Summary field.
6. Repeat for steps 2 and 3.
7. Click **OK** to display the Run Manual Test - Summary window.

Figure 7-17 Run Manual Test Summary Window



8. Select **Passed**.
9. Enter, "No problems encountered with this test." in the Summary field.

10. Click **OK**.

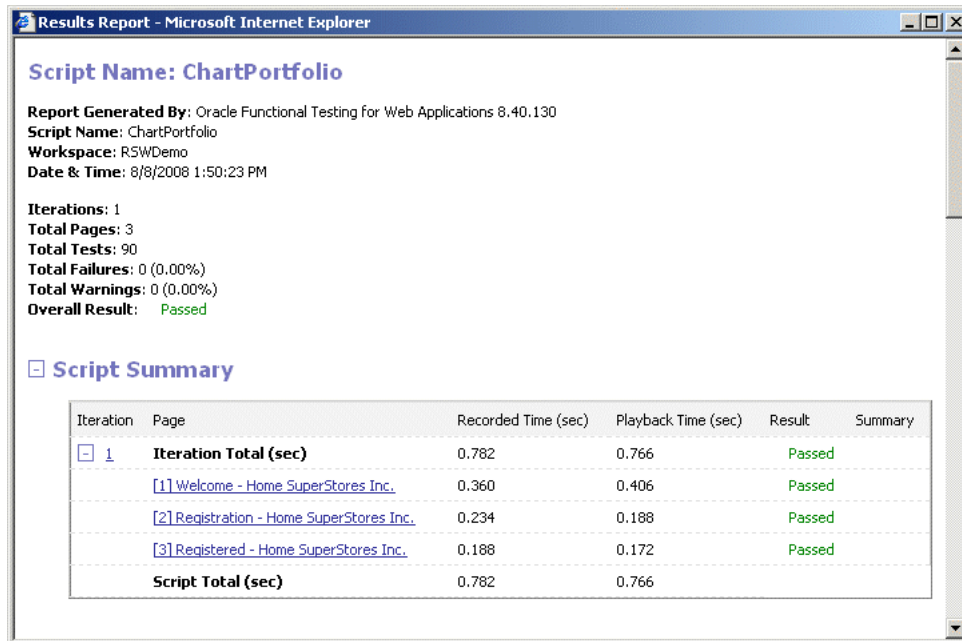
The Run History and Result Detail are displayed.

7.5.2 Running an Automated Test

This example explains how to run an the automated test created in the previous example and how to view the results detail and the detailed Oracle Functional Testing for Web Applications Results Report.

1. Click the **Tests** tab.
2. Expand item number 3 and click item 3.4.
3. Click **Run this Test** in the Run History section of the right-hand pane to display the Test run Info dialog box.
4. Select **OTM Server** in the **System** field. This is the machine on which the test will be run.
5. Select 1.0 in the **Version** field. This is the version number of the application you are testing.
6. Click **Save**. Oracle Functional Testing for Web Applications executes the script. Oracle Test Manager for Web Applications displays the Result Detail Summary when the test is finished.
7. Click the date of the test in the Run History section of the right pane.
8. Click **View Report** in the Result Summary to display the detailed Oracle Functional Testing for Web Applications Results Report in a separate browser window.

Figure 7–18 Results Report Window



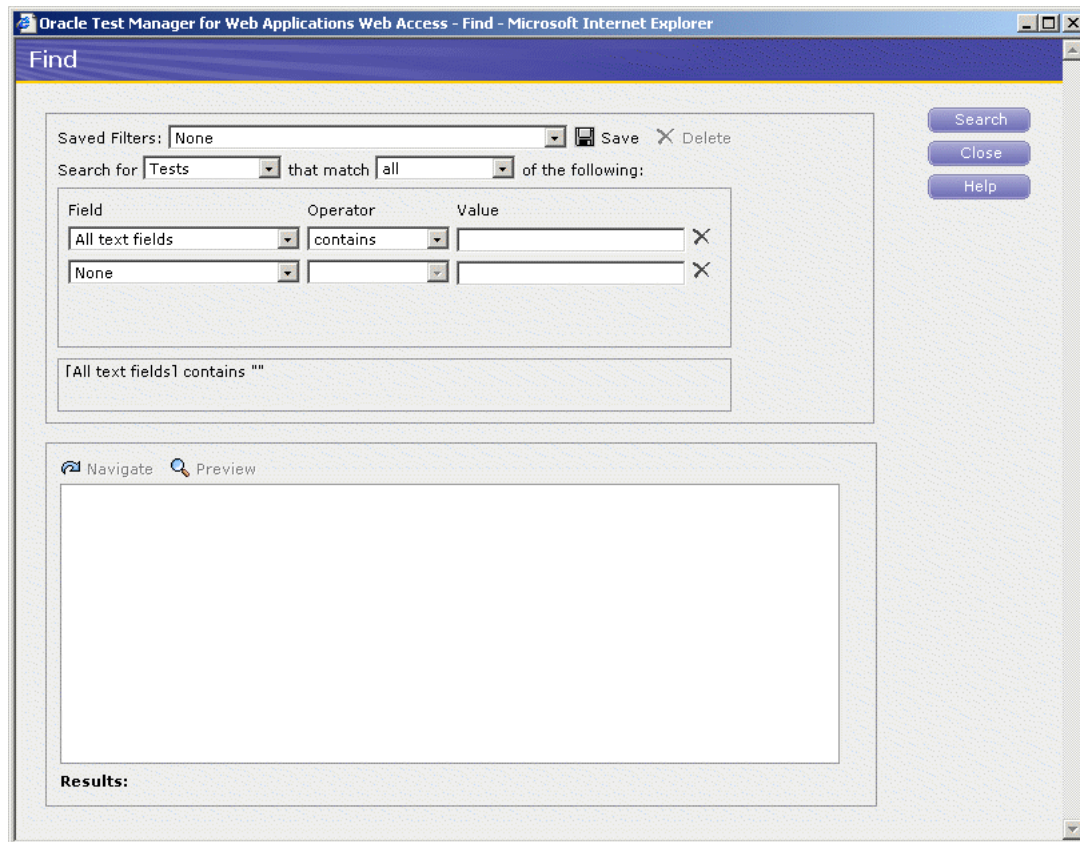
7.6 Example 4: Adding an Issue

This example explains how to search through issues, add an issue, associate it with a test, and add an attachment with additional information. For the purposes of this example, we will assume that the script run in example 4 failed.

To search through issues to see if an issue already exists for this topic, or to find related issues:

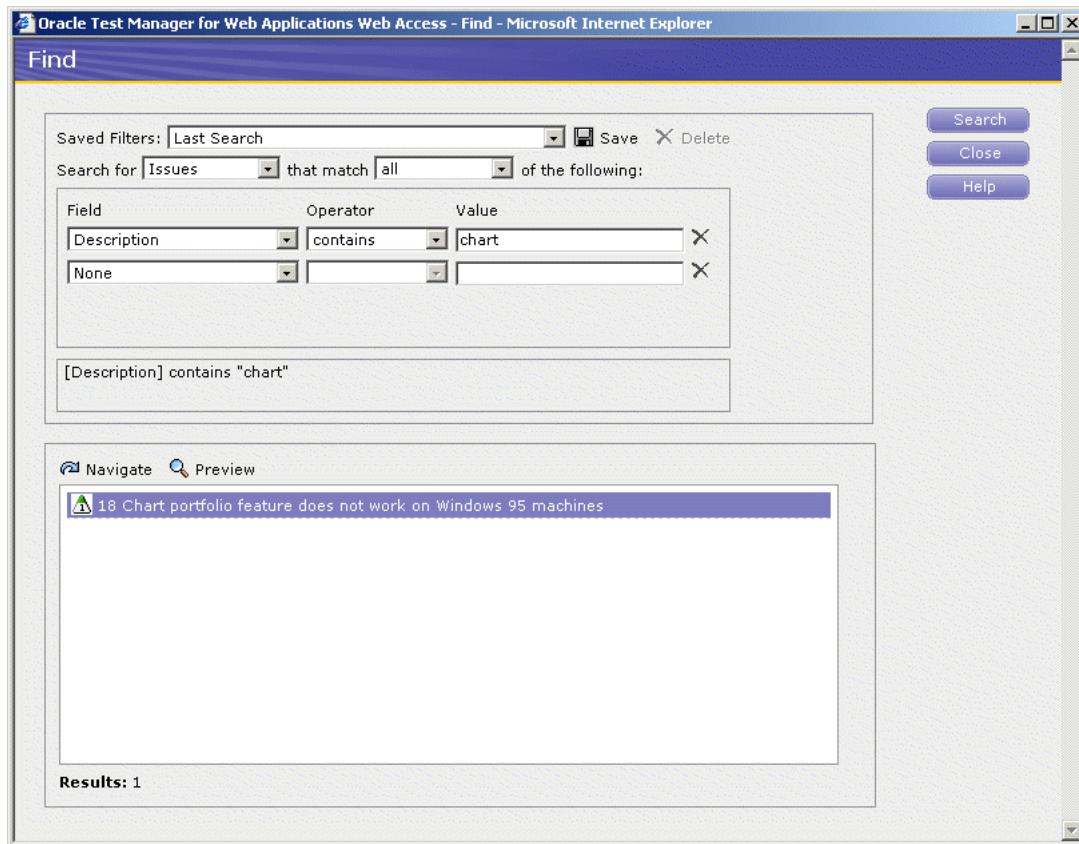
1. Click the **Issues** tab.
2. Click **Find** to display the Find dialog box.

Figure 7–19 Find Window



3. Select **Description** for the field to search.
4. Enter the value, "chart."
5. Make sure that **Issues** is selected in the **Search for** field and that **all** is selected in the **match** field. This will search all of the issue descriptions for the word, "chart."
6. Click **Search**. If there are any matches they are displayed in the Results portion of the window.

Figure 7-20 Find Window with Search Results

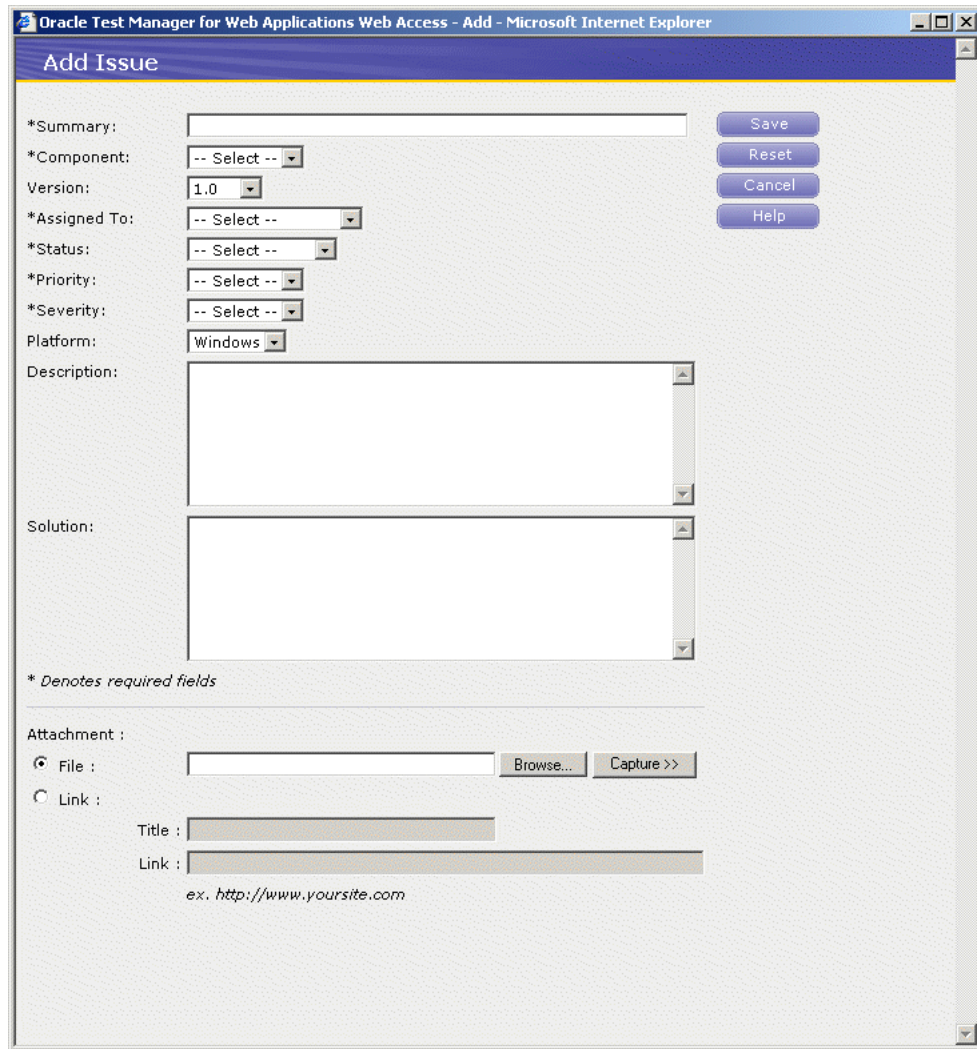


One match was found. You can click **Navigate** the Results list to display it in Oracle Test Manager for Web Applications or click **Preview** to view the issue in a separate preview window. We will assume that our failure is different enough to warrant creating a new issue.

To create an issue.

1. Click the **Issues** tab.
2. Click **Add** to display the Add Issue dialog box.

Figure 7-21 Add Issue Window



Oracle Test Manager for Web Applications Web Access - Add - Microsoft Internet Explorer

Add Issue

*Summary :

*Component : -- Select --

Version : 1.0

*Assigned To : -- Select --

*Status : -- Select --

*Priority : -- Select --

*Severity : -- Select --

Platform : Windows

Description :

Solution :

* Denotes required fields

Attachment :

File :

Link :

Title :

Link :

ex. <http://www.yoursite.com>

Buttons: Save, Reset, Cancel, Help

3. Enter "Chart portfolio failed" in the **Summary** field.
4. Select the default component.
5. Select 1.0 for the version.
6. Assign the issue to Default User.
7. Set the status to Created.
8. Change the Priority to High.
9. Change the Severity to Medium.
10. Select the Windows platform.

Figure 7-22 Add Issue Window with Sample Data

The screenshot shows a web browser window titled "Oracle Test Manager for Web Applications Web Access - Add - Microsoft Internet Explorer". The main content area is titled "Add Issue".

Fields and values shown:

- *Summary: Chart portfolio failed
- *Component: -- Select --
- Version: 1.0
- *Assigned To: Default User
- *Status: 1 - Created
- *Priority: High
- *Severity: Medium
- Platform: Windows
- Description: (empty text area)
- Solution: (empty text area)

Buttons on the right side: Save, Reset, Cancel, Help.

* Denotes required fields

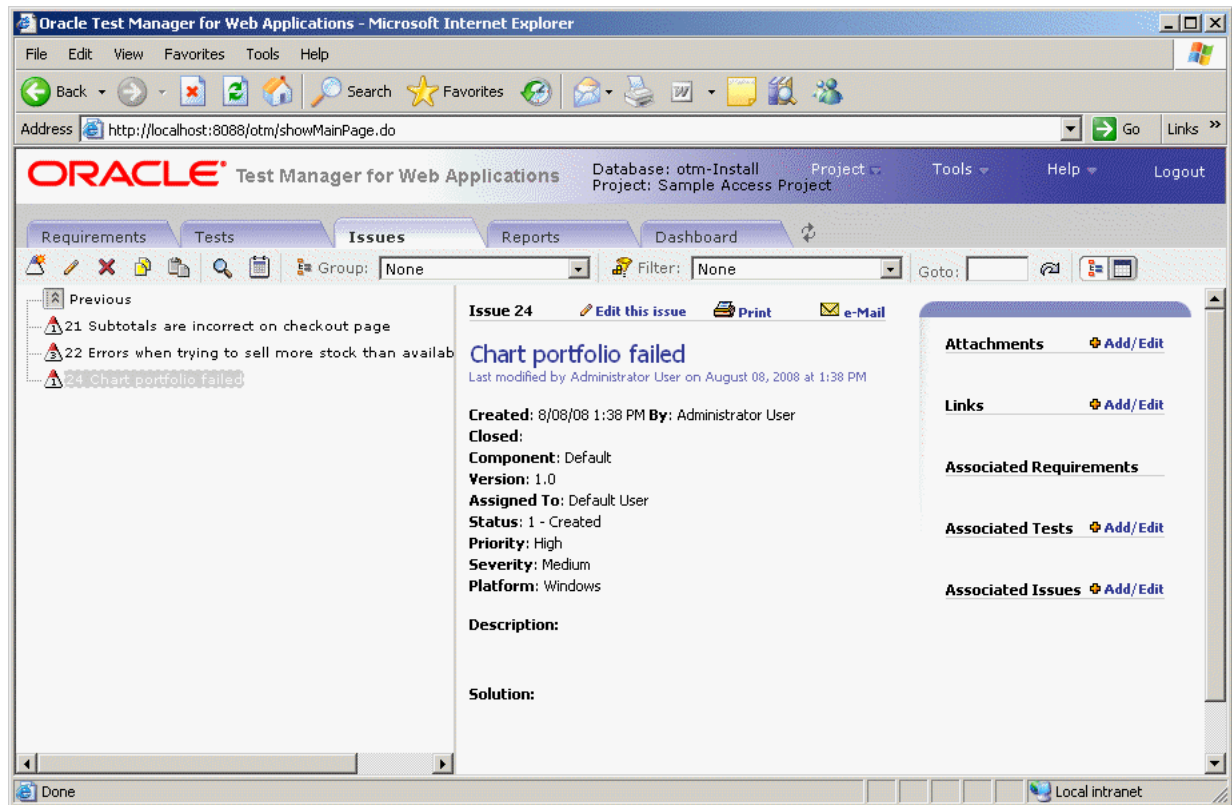
Attachment section:

- File : (input field) Browse... Capture >>
- Link : Title : (input field) Link : (input field)

Example link: *ex. http://www.yoursite.com*

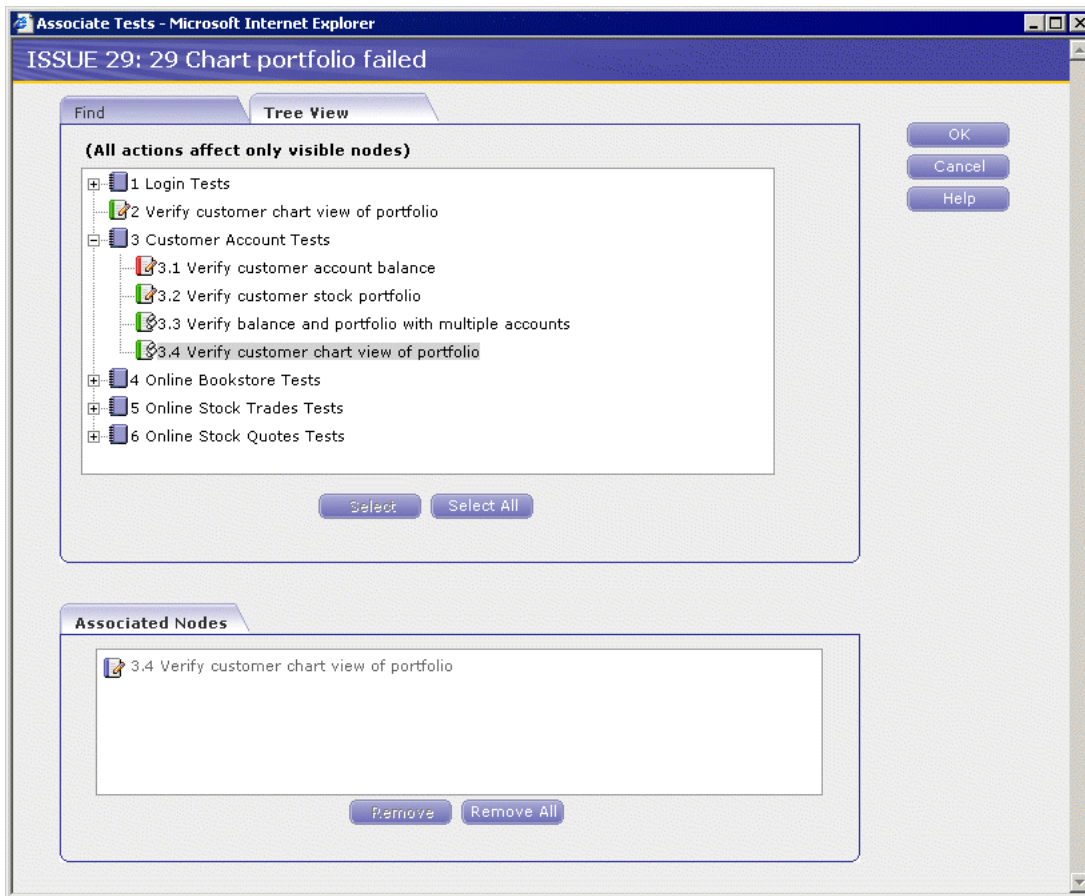
11. Click **Save**. The issue is assigned the next available number and added to the bottom of the list.

Figure 7-23 Issues Tab with New Issue Added



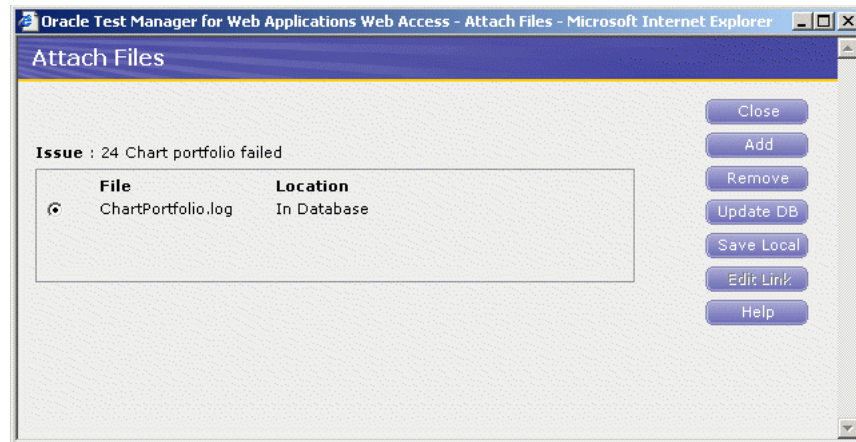
12. Click **Add/Edit** next to Associated Tests in the right pane to display the Associate Test dialog box.
13. Click the **Tree View** tab.
14. Expand item 3 and select item 3.4.

Figure 7-24 Associate Test Window with Issue Selected



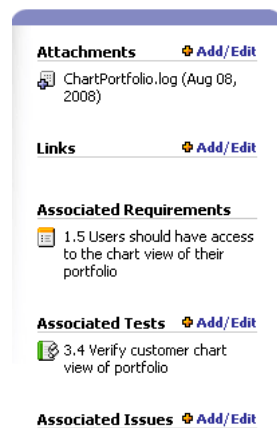
15. Click **Select**.
16. Click **OK**. The test is listed in the Associated Tests list.
17. Select **Add/Edit** in the Attachments section to display the Attach Files dialog box.
18. Click **Add**.
19. Click **Browse**.
20. Select a file to attach and click **Open**.
21. Click **Upload**.

Figure 7–25 Attach Files Window with File Selected



22. Click **Close**. The attachment is listed in the Attachments list in the right pane. Click on the attachment to open it in the appropriate application.

Figure 7–26 Issue with File Attachments

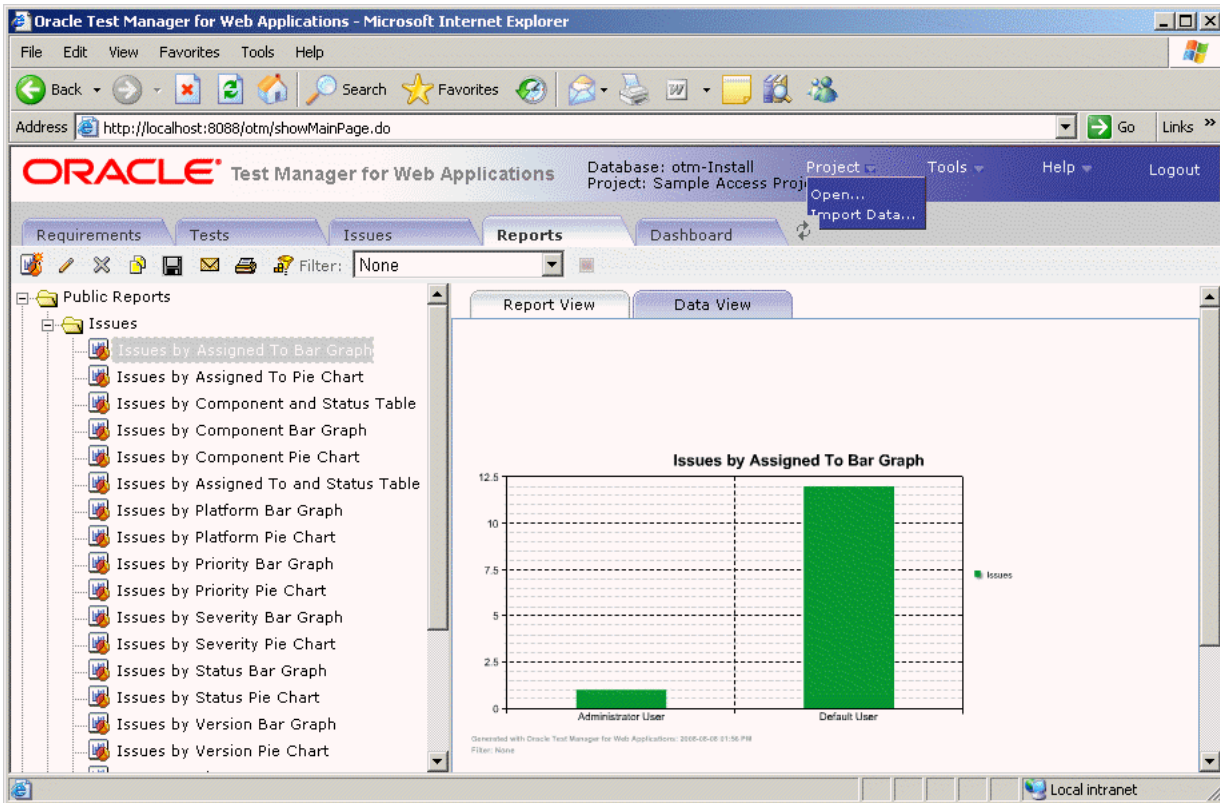


7.7 Example 5: Creating Reports

Oracle Test Manager for Web Applications comes with a standard set of reports. In addition, you can create and optionally save custom reports. Reports can be either public, that is, available to all users, or private, available only to you. This example explains how to view standard reports and how to create a custom report and save it.

1. Click the **Reports** tab.
2. Expand the Public Reports node, then the Issues node.
3. Select **Issues by Assigned to Bar Graph** to display the graph in the right pane.

Figure 7-27 Reports Tab with Bar Graph Report



4. Click once on the graph area then mouse over the bars to view the actual values. Click **Data View** to view just the data.
5. To create a custom report click **Add**.

Figure 7-28 Add Report Window

Add/Edit Report - Microsoft Internet Explorer

Add Report

Define report

Define filters

Report title:

Report on: Requirements

Report type: Vertical bar chart ?

Report template: Plain

Report data:

Available fields

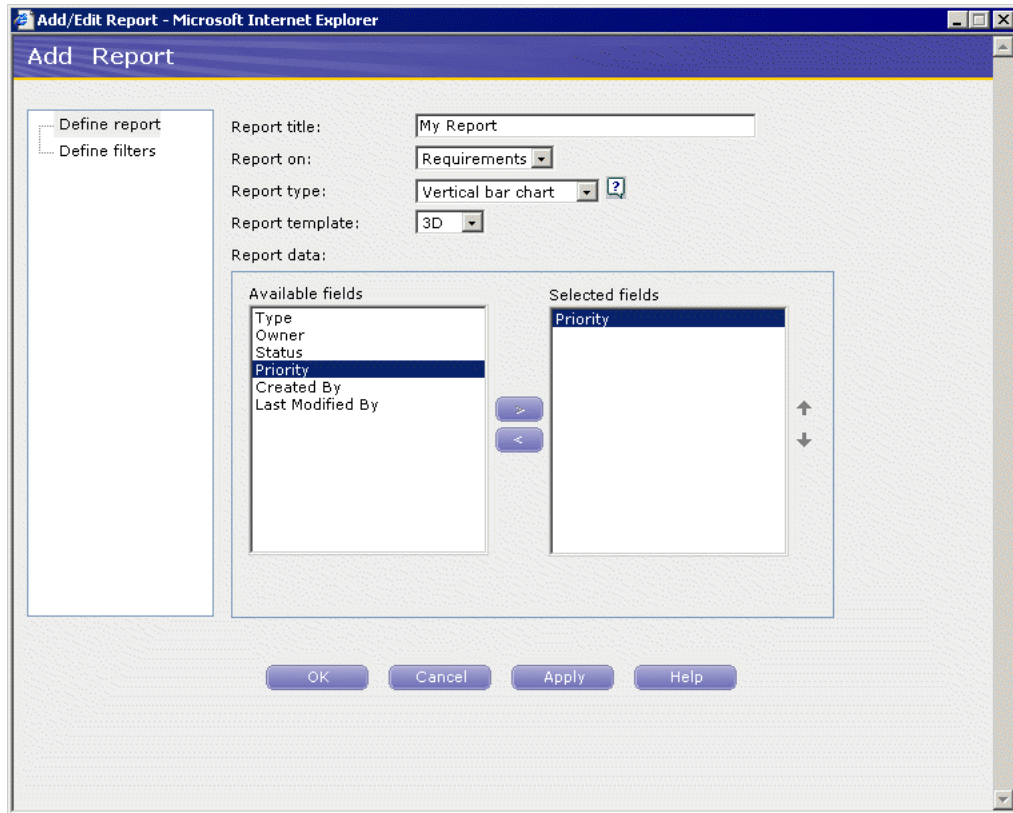
Type
Owner
Status
Priority
Created By
Last Modified By

Selected fields

OK Cancel Apply Help

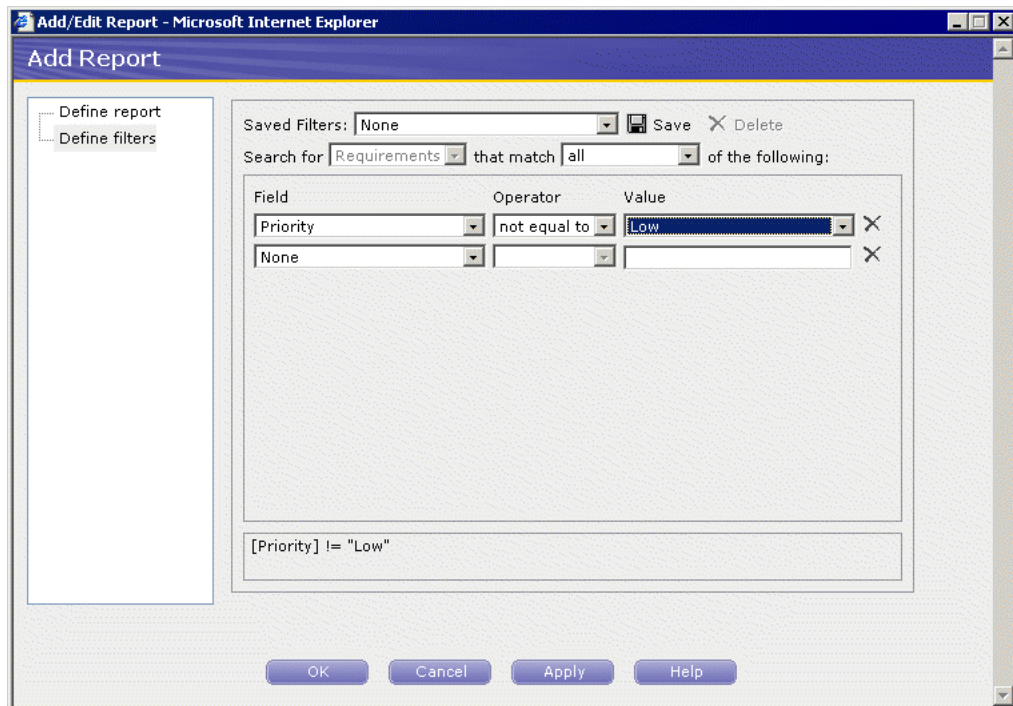
6. Enter a name for the report.
7. Select **Vertical bar chart** for the report type and **3D** for the report template.
8. Select **Priority** from the **Available fields** and click the right arrow to add it to the **Selected fields** list.

Figure 7-29 Add Report Window with Selected Fields



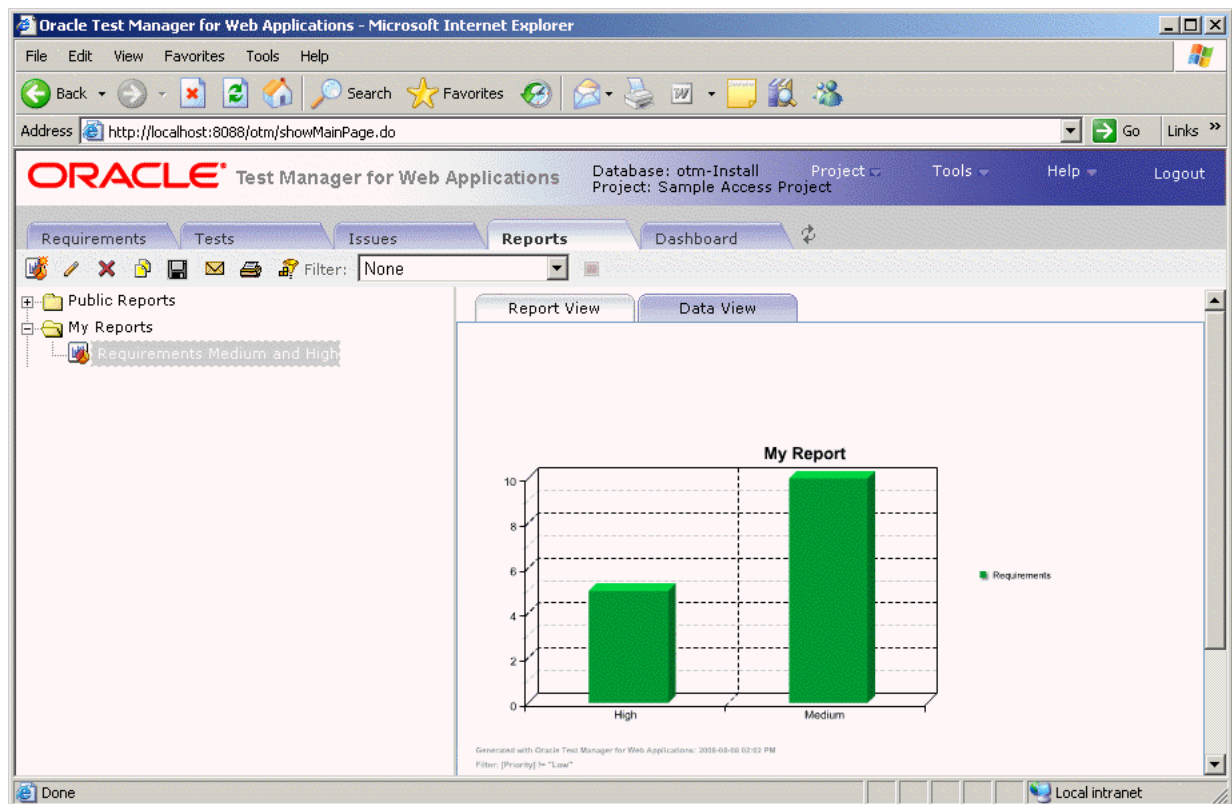
9. Click **Define filters**.

Figure 7-30 Add Report Filters Window



10. Select **Priority** as the field, **not equal to** in the **Operator** field, and **Low** for the Value. This means that the report will only display information for Medium and High priority requirements.
11. Click **OK** to display the report.
12. Click **Save** to save this report.
13. Enter **Requirements Medium and High** for the report name. The name will be displayed in the report tree.
14. Select **My Reports** as the Report Category.
15. Click **OK**. The new report is added under the My Reports folder.

Figure 7–31 Reports Tab with Custom Reports



This completes the Oracle Test Manager for Web Applications tutorial. Click **Logout** to exit the application.

A

accept a new baseline script, 4-12
Accept Tested Page, 4-12
Active Virtual Users performance statistics, 6-18
Add Binding option, 4-25
Add Mapping option, 4-23
Add per step option, 6-16
Add to Autopilot option, 6-3
Add to Scenario option, 6-3
Add Variable option, 4-22
address line, 2-17
Address node, 4-4
Administration, 1-11
Allow Blocked Content option, 4-9
Auto generate timers for all resources option, 6-15
automated tests
 adding, 7-11
 running, 7-18
automatic tests, 4-5
autopilot
 adding scenarios, 6-15
 running scenario profiles, 6-15
 setting up, 2-23, 6-4
 starting, 6-4
 stopping, 6-4
Available Data Series settings, 6-24

B

browser pane, 2-17, 2-18
Build Scenarios
 overview of, 2-22

C

Caching Emulation setting, 6-13
Clear Results Window option, 4-13, 4-29, 4-36
Collection Interval, 6-13
comma-delimited files, 4-25
configurations
 adding, 6-5
 testing, 6-11
Configure Parameters list, 6-3, 6-13
Console View, 2-15
Content Tests

 enabling, 4-6
CSV files, 4-25
custom fields, 2-8

D

Dashboard tab
 Web interface, 2-32
Data Bank Fields list, 4-25
Data Bank file
 opening, 4-25, 4-32
 selecting, 4-24, 4-32
 using in load test, 6-13
Data Bank Wizard
 auto-binding variables, 4-31
 auto-mapping variables, 4-30
 binding variables, 4-23, 4-31
 Fetch Record option, 4-26
 going to a specific record, 4-28
 mapping variables, 4-21, 4-30
 using with forms, 3-12, 4-20, 4-29
Data Binding tab, 4-23
Data Collector, 6-7
Data Mapping tab, 4-22
Data Sources
 adding, 6-5
 editing, 6-12
databanks
 configuring in OpenScript, 3-13
 definition, 1-4
database configuration, 1-11
Database Repository, 1-11
default fields, 2-9
Default Profiles list, 6-13
default tests
 modifying, 4-6
Defect Tracking, 1-11
Details View, 2-13
Developer Perspective
 Console View, 2-15
 Details View, 2-13
 Problems View, 2-14
 Properties View, 2-14
 Results View, 2-15
 Script View, 2-11
Different HTML Script node, 4-11

Discard Tested Page, 4-13
Duration node, 4-4

E

email notifications, 5-3
Epilog section in user-defined profiles, 6-29
error notifications, 5-2
Errors section in user-defined profiles, 6-29
expanding script pages, 4-3
exporting reports, 6-25

F

failure markers, 4-10
Failures Only option, 4-34
features, 1-10

- Administrator, 2-1
- functional testing, 1-5, 2-16
- Job Scheduler, 1-7, 2-19
- load testing, 1-9, 2-22
- OpenScript, 2-10

Fetch Record option, 4-26, 4-33, 4-34
Fields tab, 2-8
Find Next Failure option, 4-11
Finish section

- definition, 1-4

Form Element script nodes, 4-17
Form Element tests

- inserting, 4-16
- properties, 4-17
- script node, 4-18

Frame node, 4-4

G

Goto Page option, 4-14, 4-18
graphs

- opening in Microsoft Excel, 6-25
- Performance vs. Time report, 6-20
- Performance vs. Users report, 6-19
- sample session report, 6-25
- Statistics vs. Time report, 6-21
- Users vs. Time report, 6-20
- viewing, 6-17, 6-19

H

Hits Per Second performance statistics, 6-18
Hits performance statistics, 6-19
HTML node, 4-4

I

icons

- Issues tab, 2-30
- Requirements tab, 2-27
- Test tab, 2-29

Ignore This Failure option, 4-11
Images node, 4-4
Initialize section

definition, 1-3
Insert Table Test option, 4-18
installation, 2-1
introduction, 1-1
issues

- adding, 7-19

Issues tab

- Web interface, 2-30

Issues tab icons, 2-30
iterations

- playing back with, 3-16, 4-27

J

Java Code Editor, 2-12

- finish(), 2-13, 3-6
- initialize(), 2-13, 3-6
- run(), 2-13, 3-6

Job pane, 2-20
Job results pane, 2-20
Job Scheduler

- activating schedules, 5-5
- deactivating schedules, 5-6
- editing jobs, 5-6
- editing schedules, 5-7
- overview of, 2-19
- playing back jobs, 5-5
- saving schedules, 5-5
- selecting scripts, 5-1
- specifying notifications, 5-2
- specifying schedule time, 5-4
- starting, 5-1
- tutorial, 5-1
- using schedules, 2-21, 5-4
- viewing results, 2-20, 5-5

Job Scheduler Wizard, 2-21, 5-2
starting, 5-1

K

Kilobytes Per Second performance statistics, 6-18
Kilobytes performance statistics, 6-19

L

Links node, 4-4
load scenarios

- building, 2-22, 6-3
- setting up autopilot, 2-23, 6-4

Log Message Format notification, 5-2

M

Mail Server

- configuring, 5-3

manual tests

- adding, 7-7
- running, 7-16

Maps node, 4-4
Master node, 4-11
Maximum Time Allowed for Playback setting, 3-9,

- 4-16
- Minimum Time setting, 3-9, 4-16
- modifying default tests, 4-6
- Monitored System, 6-7
- monitors
 - adding, 6-6
 - editing, 6-12

N

- New Links script node, 4-11
- New Script option, 4-14
- Next Difference option, 4-12
- notifications
 - specifying for email, 5-3
 - specifying for results log, 5-2

O

- OpenScript
 - Console View, 2-15
 - Correlation interface, 1-4
 - Data Banking, 1-4
 - Details View, 2-13
 - Java Code View, 1-4
 - preferences, 1-4
 - Problems View, 2-14
 - Properties View, 1-4, 2-14
 - Results View, 2-15
 - tree view, 1-3
- OpenScript script Parameters
 - viewing, 3-12
- OpenScript scripts
 - adding test cases, 3-8
 - displaying properties, 3-4
 - playing back, 3-6
 - recording, 3-12
- Oracle Functional Testing for Web Applications
 - tutorial, 4-1
- Oracle Load Testing for Web Applications
 - tutorial, 6-1
- Oracle OenScript tutorial, 3-1

P

- Page Content Tests Manager, 4-6
- Pages Per Second performance statistics, 6-18
- Pages performance statistics, 6-18
- Parameters, 4-21, 4-26, 4-29
 - Data Bank script node, 4-27
 - script node, 4-21
 - viewing in scripts, 3-12
- Pass when property, 4-15, 4-27, 4-29
- Perfmon (Windows Performance Monitor), 6-7
 - selecting counters, 6-8
 - specifying instances, 6-9
- Performance object list, 6-8
- performance statistics
 - categories of, 6-18
 - viewing, 6-17
- playback

- analyzing failures, 4-10, 4-28, 4-35
- playing with iterations, 3-16, 4-27, 4-34
- starting, 3-6, 4-8
- starting jobs, 5-5
- using all records, 4-28
- using Data Bank, 3-16, 4-28, 4-35
- viewing results, 2-19, 4-10
- playing back OpenScript scripts, 3-6
- playing back Visual Scripts, 4-7
- Problems View, 2-14
- Profile performance statistics, 6-19
- profiles
 - adding scripts, 6-29
 - adding synchronization points, 6-30
 - creating user-defined, 6-28
 - editing, 6-33
 - moving scripts, 6-31
- Projects tab, 2-6
- Prolog section in user-defined profiles, 6-29
- Properties View, 2-14

R

- recording OpenScript Web Functional scripts, 3-2
- recording Visual Scripts, 3-2, 4-1
 - starting, 4-2, 4-21, 4-29
 - stopping, 4-3, 4-21, 4-29
- reporting
 - saving data, 6-13, 6-23
 - setting options, 6-14
- reports
 - creating, 2-25, 7-25
 - exporting, 6-25
 - generating, 6-23
 - overview of, 1-11
 - Performance vs. Time report, 6-20
 - Performance vs. Users report, 6-19
 - sample session report, 6-25
 - Statistics vs. Time report, 6-21
 - test results, 4-9
 - Users vs. Time report, 6-20
 - viewing scenario reports, 6-27
 - viewing session reports, 6-27
- Reports tab
 - Web interface, 2-31
- requirements
 - adding, 7-3
- Requirements Management, 1-10
- Requirements tab
 - Web interface, 2-26
- Requirements tab icons, 2-27
- resizing Visual Scripts, 4-3
- Resource Validation
 - overview of, 4-8
 - running automatically, 4-7
- Result View
 - toolbar buttons, 2-16
- Results Log
 - appending to, 4-7
 - disabling, 4-7

- enabling, 4-7
- saving, 4-29, 4-36
- Results Report
 - overview of, 4-9
 - running automatically, 4-7
- Results View, 2-15
- Roles tab, 2-5
- run graphs
 - viewing, 2-24, 6-17
- Run section
 - definition, 1-4
- Run section in user-defined profiles, 6-29
- Run Test button, 6-4, 6-16

S

- sample application
 - starting, 3-1
- sample application server
 - stopping, 3-18
- sample project
 - opening, 7-2
- Save data for reporting option, 6-14
- Save Output Log As option, 4-13
- Save Schedule As option, 5-5
- saving Results logs, 4-29
- scenarios
 - adding Virtual User profiles, 6-13
 - running multiple profiles, 6-15
 - running tests, 6-4, 6-16
 - saving, 6-15
 - setting up autopilot, 6-4
 - specifying profiles, 6-3
 - viewing reports, 6-27
- schedule window, 2-21, 5-5
- schedules
 - activating, 5-5
 - creating, 5-1
 - deactivating, 5-6
 - editing, 5-7
 - saving, 5-5
 - specifying, 5-4
- script
 - commands, 1-4
 - steps, 1-4
- Script Content Tests options, 4-6
- Script View
 - Java Code, 2-12
 - Tree View, 2-11
- Scripting Workbench, 1-3
- scripts
 - creating an OpenScript script project, 3-2
- Scripts node, 4-4
- Select Property setting, 4-17
- Select Table Element, 4-18
- Select Test Criteria setting, 4-17
- Selected text is present setting, 4-15, 4-27, 4-29
- Send HTML Summary option, 5-3
- Send Text Log option, 5-3
- Server Response tests

- inserting, 3-8, 4-15
- properties, 3-9, 4-16
- script node, 4-16
- ServerStats
 - adding monitors, 6-6
 - configuring, 6-5
 - editing configurations, 6-12
 - overview of, 2-26
 - testing configurations, 6-11
 - using configurations, 6-16
- sessions
 - saving data, 6-16
 - starting and stopping options, 6-14, 6-16
 - viewing reports, 6-27
- setting script options, 4-6
- Specify Expression setting, 4-17
- status line, 2-17
- submit my entry button, 4-14
- Submitted Scenario Profiles list, 6-3
- synchronization points
 - adding to user-defined profiles, 6-30
 - moving in user-defined profiles, 6-31

T

- Table Tests
 - creating, 4-19
 - inserting, 3-9, 4-18
 - script node, 4-20
 - selecting elements, 4-19
 - selecting fields, 4-20
- Terminate all agents at end of session option, 6-14
- test cases
 - adding to Visual Scripts, 3-8, 4-13
 - inserting Form Element tests, 4-16
 - inserting Server Response tests, 3-8, 4-15
 - inserting Table Tests, 3-9, 4-18
 - inserting Text Matching tests, 3-15, 4-14, 4-27
- test failures, 4-10
 - analyzing, 4-10
 - finding different images, 4-13
 - ignoring, 4-11
 - rejecting, 4-12
- Test Modules, 1-3
- Test Planning and Management, 1-10
- Test tab icons, 2-29
- Tested node, 4-11
- Tester Perspective
 - Console View, 2-15
 - Details View, 2-13
 - Problems View, 2-14
 - Properties View, 2-14
 - Results View, 2-15
 - Script View, 2-11
- tests
 - adding, 7-7
 - emailing, 2-29
 - running, 7-16
- Tests tab
 - Web interface, 2-28

- Text Matching tests
 - inserting, 3-15, 4-14, 4-27, 4-29
 - properties, 4-15
 - script node, 4-15
- Timer performance statistics, 6-19
- Transactions Per Second performance statistics, 6-18
- Transactions performance statistics, 6-18
- Transactions with Errors performance statistics, 6-18
- Tree View
 - Finish section, 2-12
 - Initialize section, 2-12
 - Run section, 2-12
- tutorial, 7-1
 - initializing, 4-1
 - Job Scheduler, 5-1
 - Oracle Functional Testing for Web Applications, 3-1, 4-1
 - Oracle Load Testing for Web Applications, 6-1

U

- URL dropdown, 2-17
- Use Databanks setting, 6-13
- User Mode setting, 6-13
- user-defined profiles
 - adding scripts, 6-29
 - adding synchronization points, 6-30
 - creating, 6-28
 - editing, 6-33
 - moving scripts, 6-31
- Users tab, 2-2
- using Data Banks, 3-12, 4-20, 4-29

V

- value, 6-13
- variables
 - auto-binding in Data Bank Wizard, 4-31
 - auto-mapping in Data Bank Wizard, 4-30
 - binding in Data Bank Wizard, 4-23, 4-31
 - mapping in Data Bank Wizard, 3-13, 4-21, 4-30
- View All Responses option, 6-15
- viewing failure markers, 4-10
- viewing in scripts, 4-21, 4-26, 4-29
- Virtual User (VU) Ramp-up options, 6-16, 6-17
- Virtual User Display, 6-22
- Virtual User Grid, 2-24
- Virtual User Pacing setting, 6-13
- virtual users
 - aborting all, 6-23
 - aborting individual, 6-23
 - adding per step, 6-16
 - controlling, 6-21
 - creating user-defined profiles, 6-28
 - editing user-defined profiles, 6-33
 - modifying run attributes, 6-21
 - navigating, 6-22
 - ramping up, 6-4
 - setting number of, 6-13
 - stopping all, 6-23

- stopping individual, 6-23
- viewing, 2-24
- viewing status, 6-5
- viewing user actions, 6-22
- watching, 6-22
- watching currently running, 2-24
- Virtual Users with Errors performance statistics, 6-18
- Visual Script Parameters
 - viewing, 4-21, 4-26, 4-29
- Visual Scripts
 - adding test cases, 4-13
 - analyzing playback failures, 4-35
 - collapsing, 2-18
 - displaying properties, 4-4
 - expanding, 2-18
 - expanding pages, 4-3
 - playing back, 4-7
 - recording, 4-1
 - rejecting problems, 4-12
 - resizing, 2-18, 4-3
 - saving, 4-3, 4-29
 - test flags, 2-18
 - viewing, 2-17
- VUs performance statistics, 6-19

W

- Watch Virtual User Grid tab, 2-24
- Web interface
 - Dashboard tab, 2-32
 - Issues tab, 2-30
 - Reports tab, 2-31
 - Requirements tab, 2-26
 - Tests tab, 2-28

