Oracle® Application Testing Suite

Getting Started Guide

Version 8.40

September 2008

Oracle Application Testing Suite Getting Started Guide Version 8.40

Contents

Chapter 3

Oracle Functional Testing for Web Applications Tutorial 29

Chapter 5

Oracle Load Testing for Web Applications Tutorial 99

[This page intentionally left blank]

Preface

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

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.

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.

About This Guide

This guide contains the following chapters:

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 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, test cases, and custom tests using Visual Basic for Applications (VBA).

Chapter 4 – Job Scheduler Tutorial: provides step-by-step instructions for creating Job Scheduler jobs and schedules to play back multiple Oracle

Functional Testing for Web Applications Visual Scripts for operational and performance monitoring of a site. The tutorial includes an example for adding custom tests using Visual Basic for Applications (VBA).

Chapter 5 – Oracle Load Testing for Web Applications Tutorial: provides step-by-step instruction for using multiple Oracle Functional Testing for Web Applications Visual 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.

Related Documentation

The Oracle Application Testing Suite includes a complete set of electronic manuals and online help.

Oracle Application Testing Suite Manuals

The Oracle Application Testing Suite includes the following manuals:

Oracle Application Testing Suite Getting Started Guide – introduces the software tools in the Oracle Application Testing Suite and provides step-by-step tutorials for Oracle Functional Testing for Web Applications, Job Scheduler, and Oracle Load Testing for Web Applications (which includes ServerStats).

Oracle Functional Testing for Web Applications Functional Testing User's Guide – explains how to use the features and options of Oracle Functional Testing for Web Applications to create Visual Scripts for regression/performance testing and monitoring of Web sites or applications. This guide also includes reference information for using the Oracle Functional Testing for Web Applications Programming Interface for expanding Visual Script capabilities using Visual Basic for Applications (VBA).

Oracle Functional Testing for Web Applications Job Scheduler User's Guide – explains how to use the features and options of Job Scheduler to schedule and play back multiple Visual Scripts for regression testing of Web sites or applications. Job Scheduler uses the Visual Scripts developed by Oracle Functional Testing for Web Applications.

Oracle Functional Testing for Web Applications Navigation Editor User's Guide – explains how to use the features and options of the Navigation Editor to you modify the navigation paths used by of Oracle Load Testing for Web Applications Thin Client to play back the sequence of pages in the Visual Script.

Oracle Functional Testing for Web Applications Application Programming Interface Reference – provides reference information for using the Oracle Functional Testing for Web Applications Programming Interface for expanding Visual Script capabilities using Visual Basic for Applications (VBA).

Oracle Functional Testing for Web Applications Result Objects Reference – explains how to use the Application Programming Interface to access Oracle Functional Testing for Web Applications Result Objects generated after playback of Visual Scripts.

Oracle Load Testing for Web Applications Load Testing User's Guide – explains how to use the features and options of Oracle Load Testing for Web Applications to simulate multiple users accessing a Web site or application for performance and scalability testing. Oracle Load Testing for Web Applications uses the Visual Scripts developed by Oracle Functional Testing for Web Applications.

Oracle Test Manager for Web Applications Test Manager User's Guide – explains how to use the features and options of Oracle Test Manager for Web Applications to organize and manage your overall testing process. It provides a single unified platform for sharing information among team members.

Electronic Documentation

The Oracle Application Testing Suite download from our Web site automatically installs an Adobe Portable Document Format (PDF) version of the *Oracle Application Testing Suite Getting Started Guide*. This document requires the Adobe® Acrobat® Reader™, version 4.x (or higher), to open and view the .PDF file. You can download the free Acrobat Reader from the Adobe Web site: http://www.adobe.com/products/acrobat/readstep2.html.

The *Oracle Application Testing Suite Getting Started Guide* PDF file is an option on the Oracle Application Testing Suite **Start** menu.

The installation also includes a readme.htm file that contains release notes and the latest updates to the product documentation. The readme.htm file opens in Microsoft® Internet Explorer when you select the **Release Notes** option on the Oracle Application Testing Suite **Start** menu.

The full Oracle Application Testing Suite documentation set is provided on the product CD-ROM in Adobe Portable Document Format (PDF). You can use the Acrobat Reader, version 4.x (or higher), to view, search, and print the documentation set. The Oracle Application Testing Suite CD-ROM includes the Acrobat Reader application. You can install Acrobat Reader using the Oracle Application Testing Suite CD-ROM setup program or by selecting the Acrobat Reader executable file in the Supporting Products\Acrobat Reader directory. See the Adobe Web site, http://www.adobe.com, for additional information about Acrobat Reader.

All of the .PDF files are in the Documentation directory of the product CD-ROM. *The Oracle Application Testing Suite Getting Started Guide* PDF and readme.htm files are installed during the Oracle Application Testing Suite setup procedure. You can copy the user guide .PDF files to a local drive or open them in Acrobat Reader from the CD-ROM.

If you downloaded the product from our Web site and wish to receive electronic versions of the User Guides in .PDF format, go to http://oracle.com/support/index.html. You'll need the Acrobat Reader v4 (or higher) to open and view the documents.

The Oracle Application Testing Suite CD-ROM also includes the Microsoft® Visual Basic® Scripting Edition (VBScript) Language Reference documentation. You can install the files using the Oracle Application Testing Suite CD-ROM setup program or by selecting vbsdoc.exe in the Supporting Products\MS VB Script Documentation directory.

Using Help

Oracle Functional Testing for Web Applications provides a comprehensive on-line help system. The help topics include step-by-step "how to" instructions for common tasks and a complete reference.

You can press the F1 key at any open dialog box for an explanation of the options. You can select **Help → Contents** to open the on-line help contents and index.

Conventions

This guide uses the following typographical conventions to identify specific items:

Convention	Description
Bold Sans Serif	Menu options and dialog box selections.
Sans Serif Underlined	Web page hyperlinks.
Sans Serif	File names and Visual Script nodes.
`Fixed-Pitch type`	Program code.
→ Arrow between menu options.	Select each option in sequence.

Obtaining Technical Support

If you have questions about Oracle Application Testing Suite, first look in this guide, the online Help, and the release notes delivered with the software.

You can also check our Web site for the latest information about upgrades and other issues at http://www.oracle.com/support/index.html.

[This page intentionally left blank]

Chapter 1

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 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

The applications are powered by a common set of Visual Scripts and require no programming. There are no proprietary languages to learn, no special proxies to set up, no training classes required. 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.

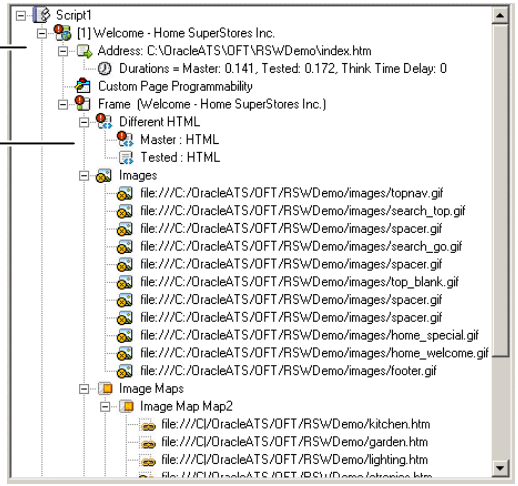
This manual introduces you to the Oracle Application Testing Suite and provides step-by-step tutorials to help you get started using the tools.

About Oracle Functional Testing for Web Applications

Oracle Functional Testing for Web Applications is used for functional/regression testing and serves as the script recorder for the entire 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.

VisualScripts Show Page Objects in a Tree Hierarchy

Scripts Use Flags to Show Differences Between Baseline and Current Version



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.

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.
- **e-Spider** – automatically maps your entire site and creates Visual Scripts for all or part of your application.
- **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 six 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's built-in test cases or your own custom Visual Basic code in the fully-integrated Visual Basic for Applications (VBA) development environment.
- **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.

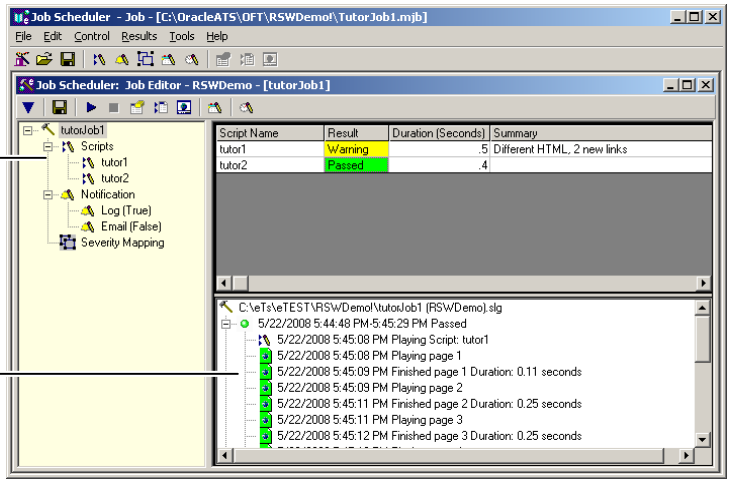
- **Broad Extensibility with Standard Languages** – Visual Scripts can be extended through external callouts with your custom tests written in Visual Basic, C++, or Java.
- **High Throughput Resource Validation** – automatically collects and verifies all referenced Web resources that include links and images.
- **Test Case Librarian** – allows you to create and store re-useable test cases for use across multiple test scripts.

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.

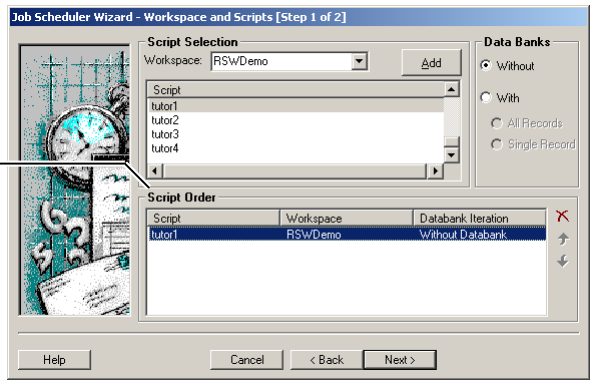
Multiple Visual Scripts Can be Played Back Immediately or at a Scheduled Time

Playback Results are Logged at Runtime and Can Be Customized Using the Wizard



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.

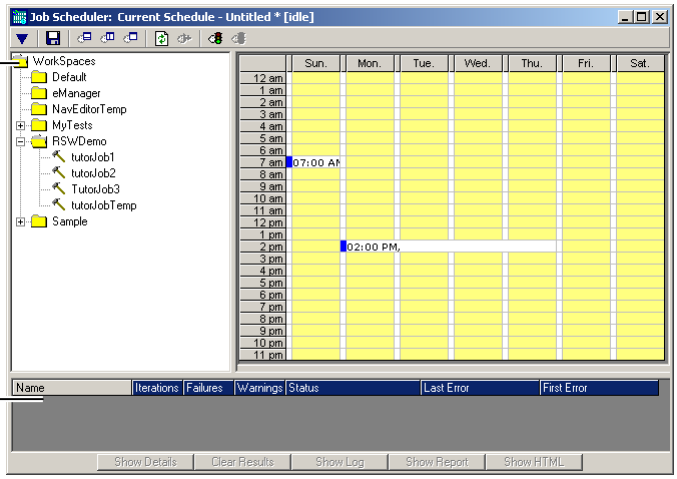
The Wizard Provides a Convenient Way to Create Jobs



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

The Job Tree lists Available Jobs and Workspaces

The Job Summary shows Playback Results



Feature Highlights

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

Multiple Oracle Functional Testing for Web Applications 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.

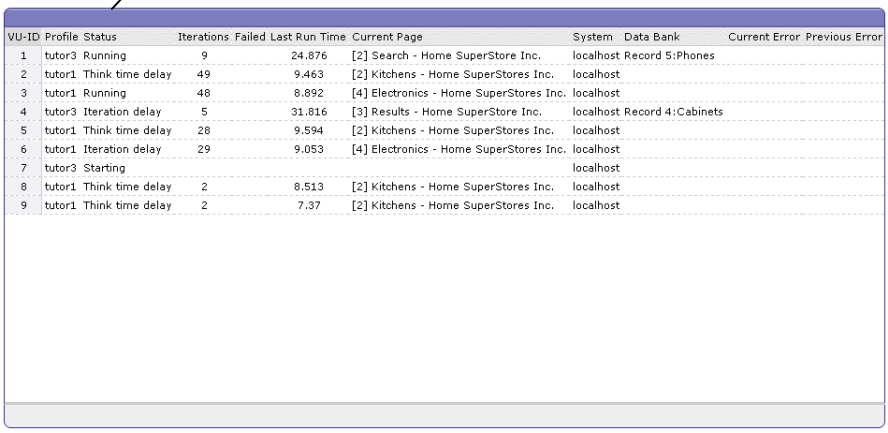
HTML Format Job Results Reports – playback results reports are saved to an HTML page for later review and analysis.

Visual Basic for Applications (VBA) Integration – lets you add custom code to playback jobs for customized testing requirements.

About Oracle Load Testing for Web Applications

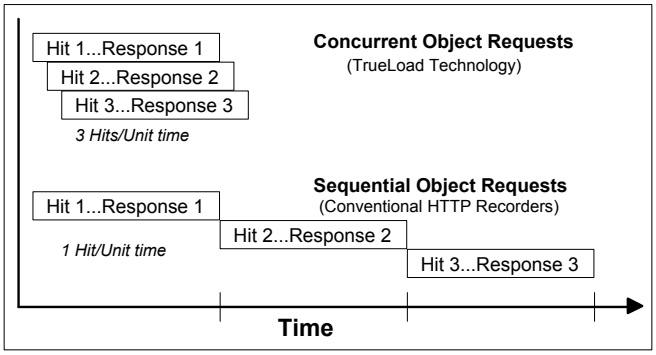
Oracle Load Testing for Web Applications provides an easy and accurate way to test the scalability of your e-Business applications. Oracle Load Testing for Web Applications emulates thousands of virtual users accessing your site simultaneously, and measures the effect of the load on application performance.

Oracle Load Testing for Web Applications
Emulates Multiple Virtual Users and User Profiles



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 Oracle Load Testing for Web Applications results to help make key decisions about your system's architecture, tuning, and hosting alternatives.



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 same Visual Scripts created for functional testing with 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 & Autopilot** – define any number of custom load scenarios by simply pointing and clicking on the names of the pre-recorded Visual 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.

System Requirements

Oracle Functional Testing for Web Applications has the following system requirements:

- Operating System: Windows XP, Windows Vista, Windows 2003, Windows 2000 (SP1 or higher)
- Memory: Minimum 256 MB; 512 MB or greater recommended
- System: IBM-compatible PC with Pentium processor, 1.5 GHz or faster
- Disk Space: 1 GB minimum
- Browser: Internet Explorer 6.0 or higher

Oracle Load Testing for Web Applications has the following system requirements:

- Operating System: Windows XP, Windows Vista, Windows 2003, Windows 2000 (SP1 or higher)
- Memory: Minimum 1 GB
- System: IBM-compatible PC with Pentium processor, 1.5 GHz or faster
- Disk Space: 1 GB minimum
- Browser: Internet Explorer 6.0 or higher, Firefox 2.0 or higher

- Database: Microsoft SQL Server or Microsoft Access (Access not recommended for production usage)
- Application Server: JBoss 4.0.2 (downloadable from http://www.jboss.org/jbossas/downloads)

> **Note**: These are minimum requirements only and actual requirements for the Oracle Load Testing for Web Applications Server will vary depending on the size and configuration of your load test. If you are running larger load tests, of greater than 1000 Virtual Users, you should consider deploying Oracle Load Testing for Web Applications on a faster server class machine with a minimum of 1GB RAM recommended.
>
> **Note**: The amount of memory required on the Oracle Load Testing for Web Applications Agent systems may increase based on the number of Virtual Users that will be assigned to run on each Agent. For more information, visit the QA/Testing Technology Center on Oracle Technology Network or contact your Oracle representative.

Oracle Test Manager for Web Applications has the following system requirements:

- Operating System: Windows XP, Windows Vista, Windows 2003, Windows 2000 (SP1 or higher)
- Memory: Minimum 1 GB RAM
- System: IBM-compatible PC with Pentium processor, 1.5GHz or faster
- Disk Space: 1 GB minimum
- Browser (for Web client): Microsoft Internet Explorer 6.0 or higher, Firefox 2.0 or higher
- Database: Microsoft SQL Server or Microsoft Access (Access not recommended for production usage)
- Application Server: JBoss 4.0.2 (downloadable from http://www.jboss.org/jbossas/downloads)

> **Note**: These are minimum requirements only and actual requirements for Oracle Test Manager for Web Applications will vary depending on the quantity of test assets stored and number of concurrent users

accessing the application. If you have a large quantity of test cases, requirements or issues stored and/or are deploying Oracle Test Manager for Web Applications for a team greater than 10 users, you should consider deploying Oracle Test Manager for Web Applications on a faster server class machine with a minimum of 1GB RAM recommended.

[This page intentionally left blank.]

Chapter 2

Oracle Application Testing Suite Basics

This chapter explains how to get started using Oracle Functional Testing for Web Applications. It explains how to install and start the program, and the features of the main window.

Installing and Starting Oracle Application Testing Suite

There are two ways that you can install Oracle Application Testing Suite:

1. From the web site:

   http://www.oracle.com/

   - Download the Oracle Application Testing Suite product from the Web site and save it to a temporary directory on your hard disk.
   - Unpack oats##-IE#.exe and then run setup.exe to install Oracle Application Testing Suite.

2. From the CD:

   - Insert the Oracle Application Testing Suite CD into your computer's CD-ROM drive.
   - In Windows, click **Start → Run** and browse to the drive letter that corresponds to the CD-ROM drive.
   - Run setup.exe located in the root directory of the CD-ROM.

3. Follow the setup instructions to install the Oracle Application Testing Suite.

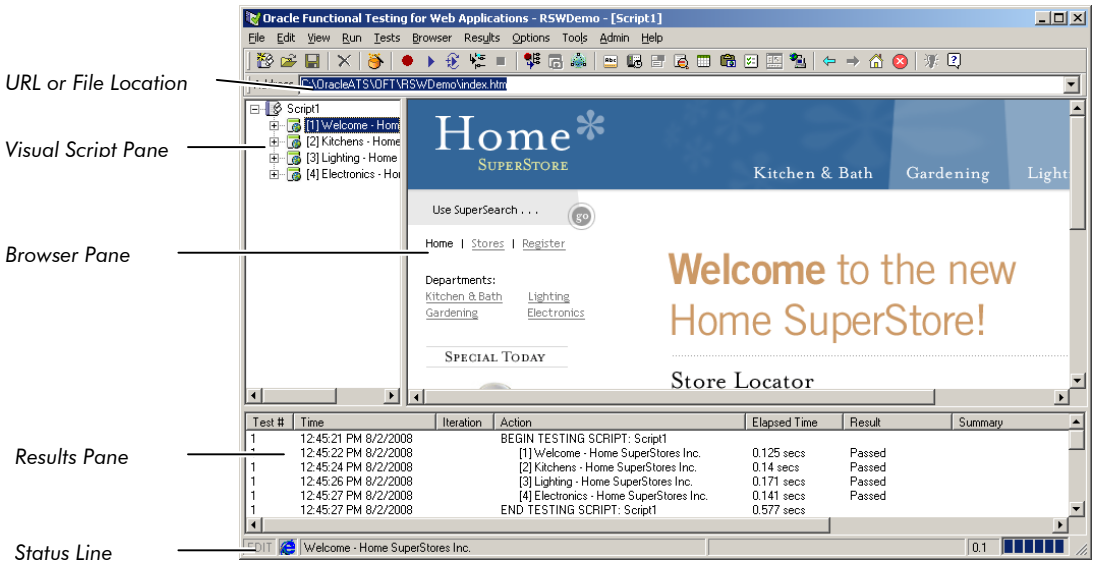
4. Click **Start → Programs → Oracle Application Testing Suite → Oracle Functional Testing for Web Applications** to start Oracle Functional Testing for Web Applications.

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, and Oracle Load Testing 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.

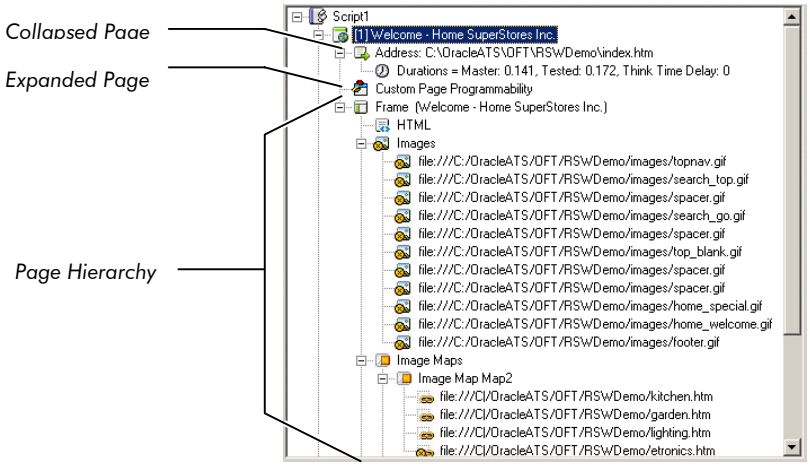


The Title bar of the window shows the program name followed by the current Workspace and Visual Script name.

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.

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 (either manually or using the e-Spider), Oracle Functional Testing for Web Applications creates the Visual Script tree for you.



Click ⊞ to expand a branch or ⊟ 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.

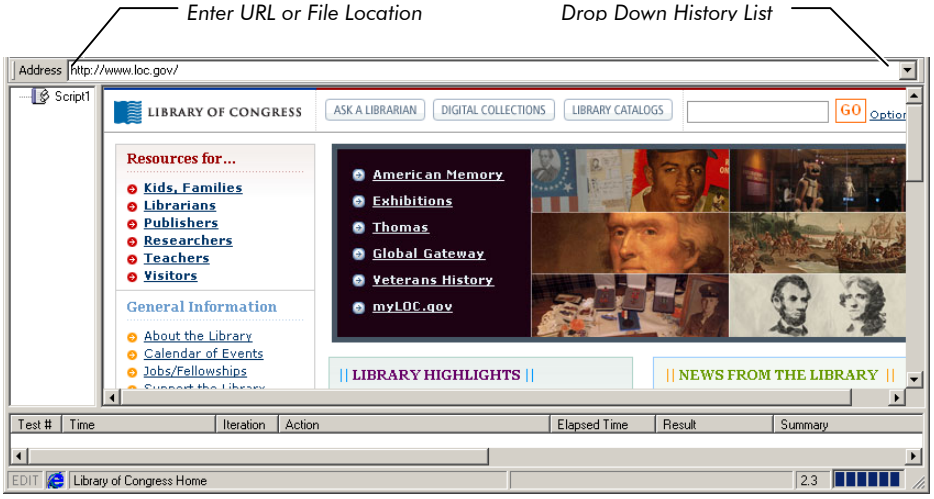
You can toggle the Visual Script pane width using the **View → Resize Visual Script View** option or by dragging the border between the Browser pane and the Visual Script pane.

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.

Browser Pane

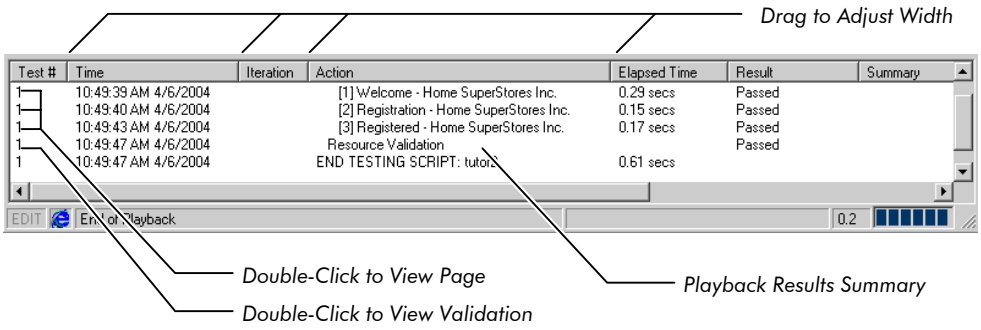
The browser pane contains a seamlessly integrated Web browser that you use to select the Web pages to test. It provides full navigation and Web access.



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.

Playback Results Pane

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



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

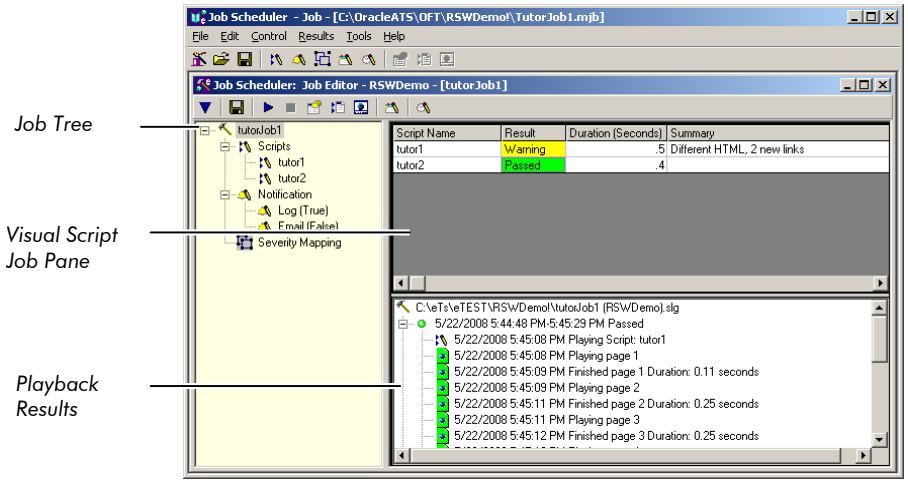
Icons in the Visual Script show the location of any specific failures of default tests or test cases. Resource Validation test results are listed in a separate window after playback of the script.

You can turn on and off the display of the Playback Results pane using the **View → Resize Results Window** option.

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.

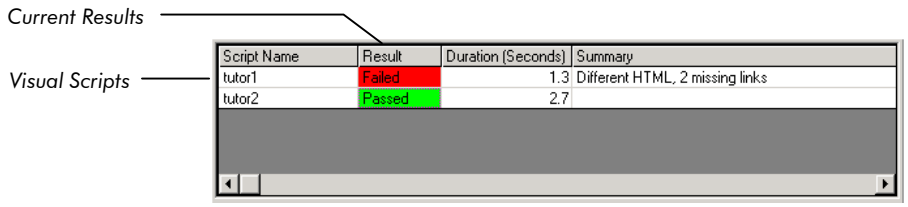
The main window consists of the menu bar and toolbar. Job Scheduler has three windows that run within the main window: Current Schedule window, Current Job window, and Job Editor window.



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

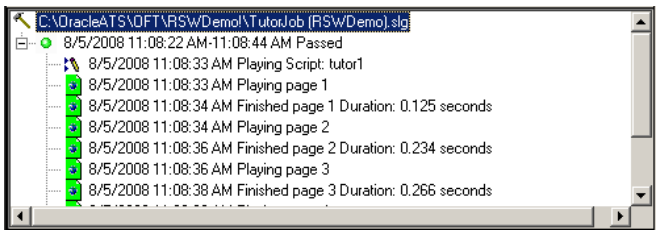
Visual Script Job Pane

The Visual Script job pane list 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.



Results Pane

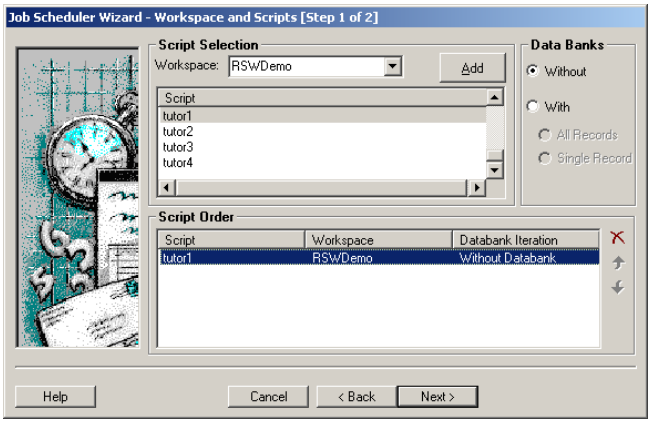
The Results pane shows any log messages generated during playback of the job.



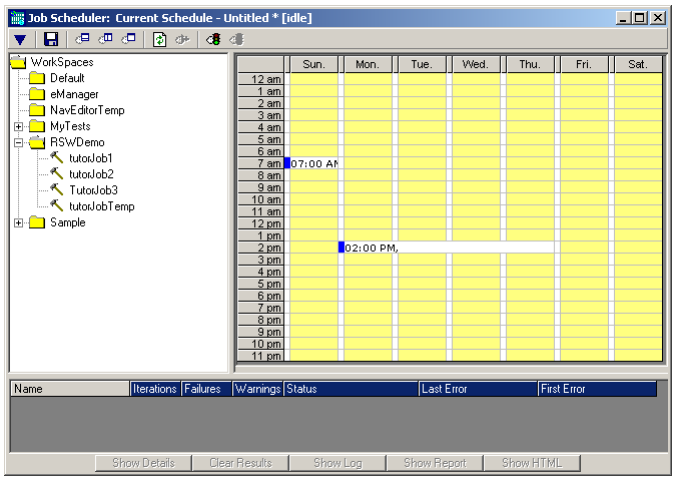
You can customize log messages as required using the Job Scheduler Wizard or using your own VBA code.

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.



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

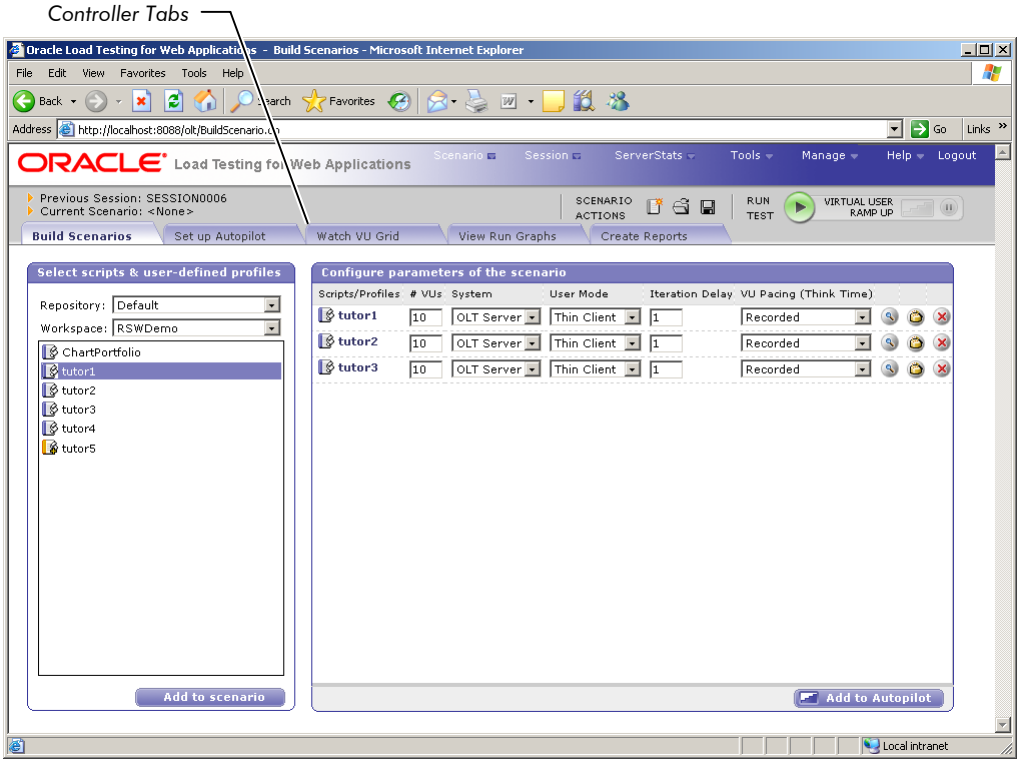


Schedules can be used with the current job or other saved jobs.

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 the same Visual Scripts that you develop using Oracle Functional Testing for Web Applications.

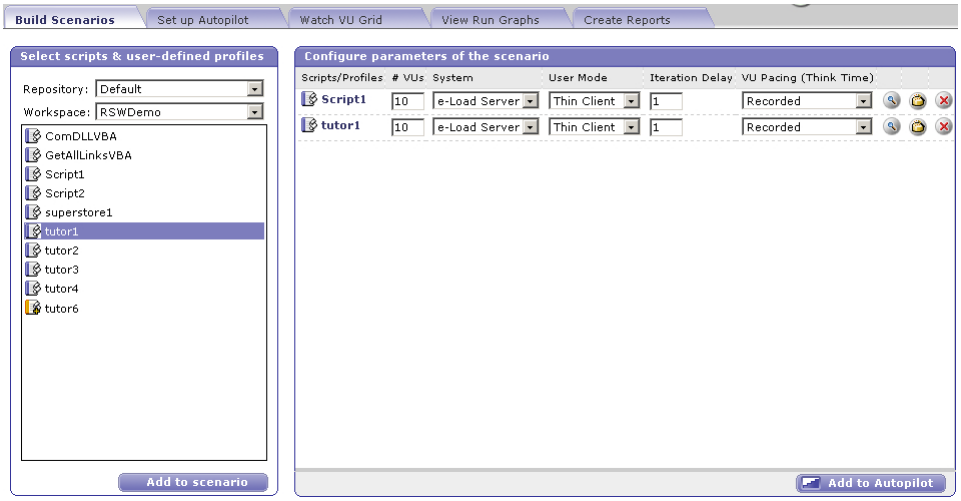
The main window consists of the menu bar, toolbar, and the controller tab dialogs.



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

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.

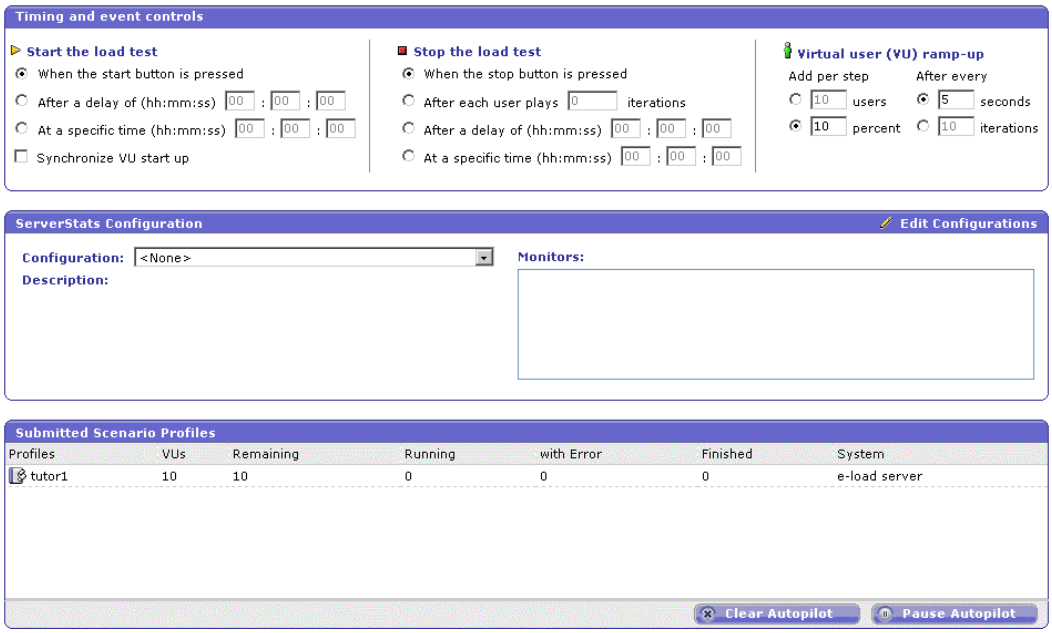


You can define user profiles that specify which visual scripts the users playback to emulate real users and how many virtual users to emulate.

Set up Autopilot Tab

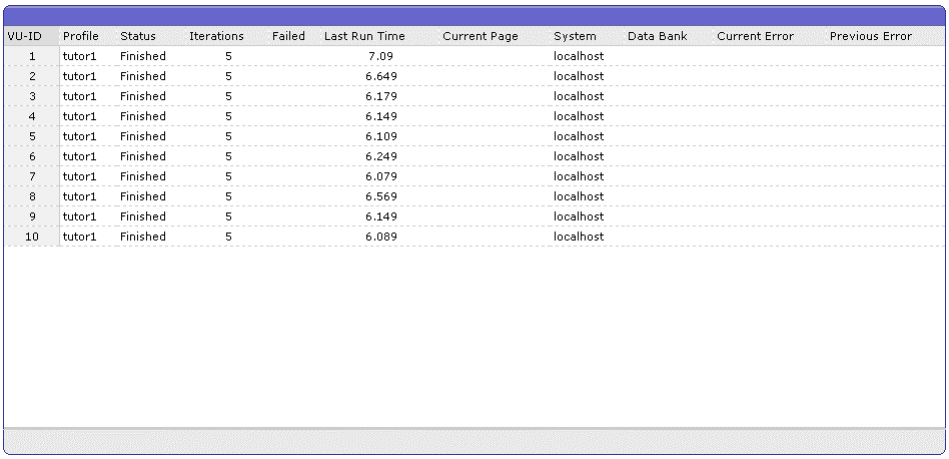
The Autopilot tab is where you specify the information needed to control how the scenario starts and runs. The Autopilot controls the starting and stopping of the scenario, the rate at which new virtual users are started, and shows the total number of virtual users and the number of running virtual users.

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.



Watch Virtual User Grid Tab

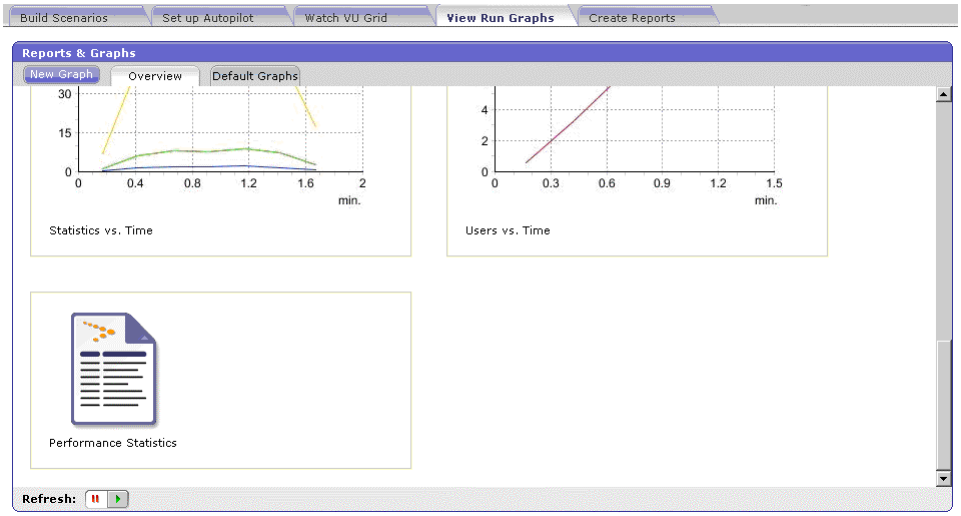
The Watch Virtual User Grid tab lists the currently running virtual users and the profile and playback details associated with each.



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			

View Run Graphs Tab

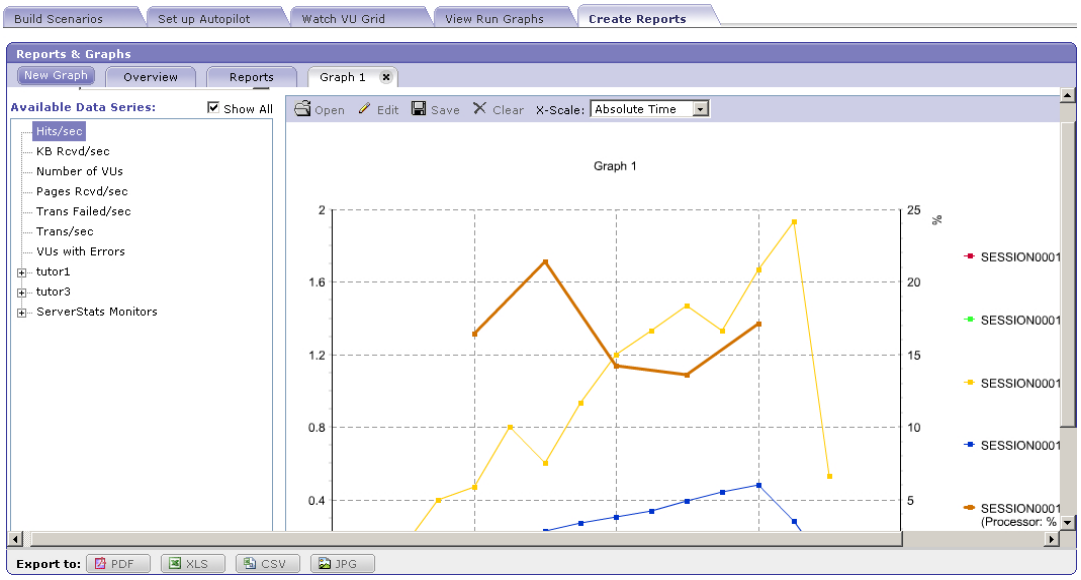
The View Run Graphs tab lets you define graphs and view the graphs at run-time.



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.

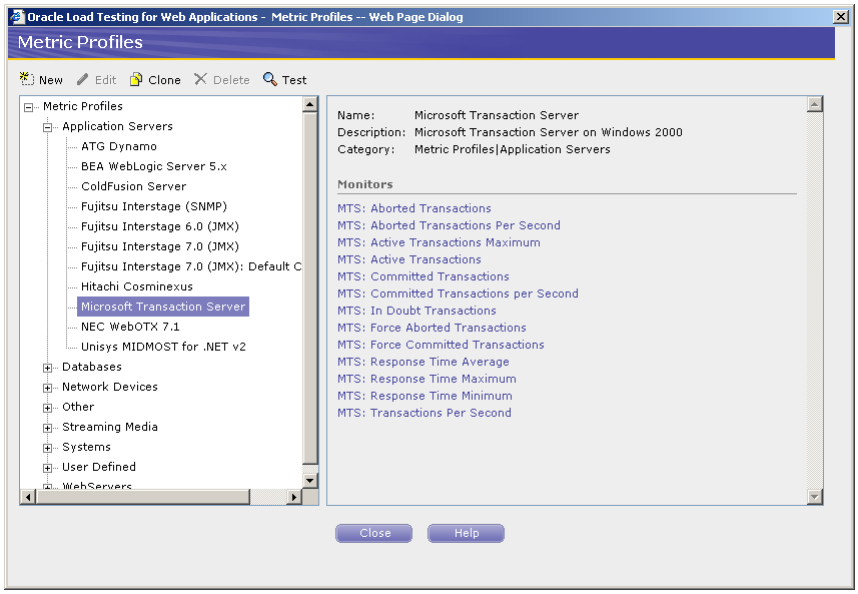
Create Reports Tab

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



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.



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.

[This page intentionally left blank]

Chapter 3

Oracle Functional Testing for Web Applications Tutorial

This tutorial walks you through the main features of the 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 Oracle Functional Testing for Web Applications's 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.
- **Using Custom Tests** – explains how to use the Custom Test Wizard to add custom Visual Basic code to your Visual Scripts to extend your testing capabilities.

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.

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 **Start → Programs → Oracle Application Testing Suite → Build A - Home Superstores**. A DOS window will appear briefly as the batch file copies the Build A files.
2. Close the DOS window, if necessary.

Example 1

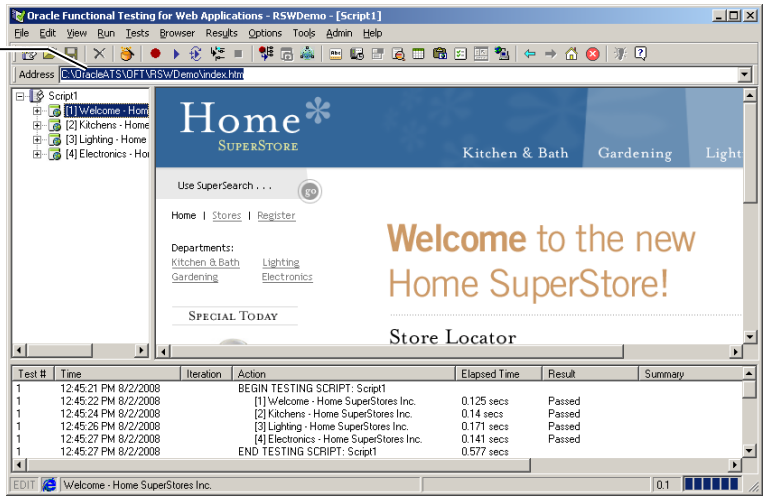
Recording a New Visual Script

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

Start Oracle Functional Testing for Web Applications

1. Select **Start → Programs → Oracle Application Testing Suite → Oracle Functional Testing for Web Applications** to start Oracle Functional Testing for Web Applications.
2. Select **File → Open Workspace**, 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.)



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

Start a Recording

4. Select **File → New Script** 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.

Navigate the Web Site

6. Click on the Kitchen & Bath link on this page. The Kitchen & 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.

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.

Save the Script

10. Select **File → Save Script** 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**.

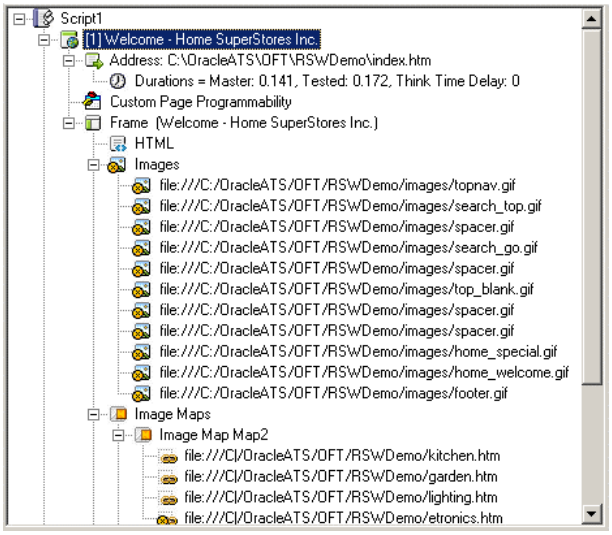
Example 2

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 **View → Resize Visual Script View** to expand the Visual Script pane.
2. Click the [1] Welcome - Home Superstores Inc. node in the Visual Script and then click the ⊞ icon 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:



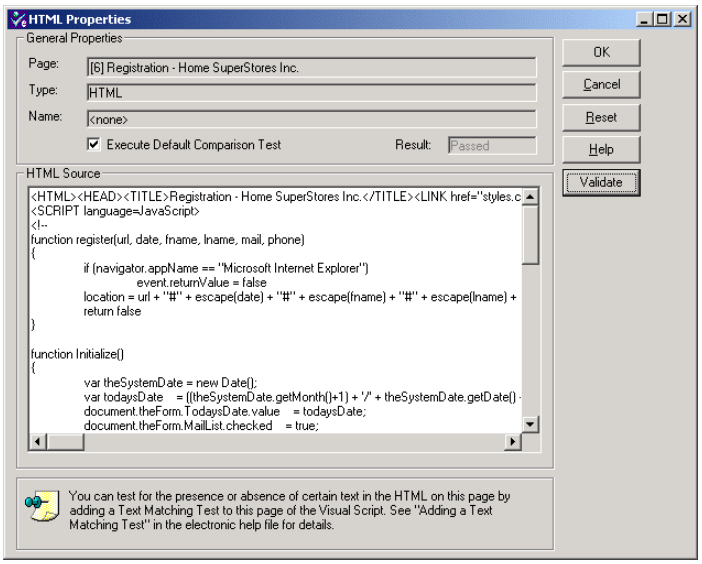
The [1] Welcome - Home Superstores Inc. page contains the following nodes in the tree:

- 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 Custom Page Programmability node that provides custom programming options using the Visual Basic for Applications (VBA) Integrated Development Environment. You can use all of the features and capabilities of the VBA IDE and the Oracle Functional Testing for Web Applications e-Programming Interface to include advanced testing capabilities to your Visual Scripts.
- 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.

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:



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.

Turning Automatic Testing On and Off

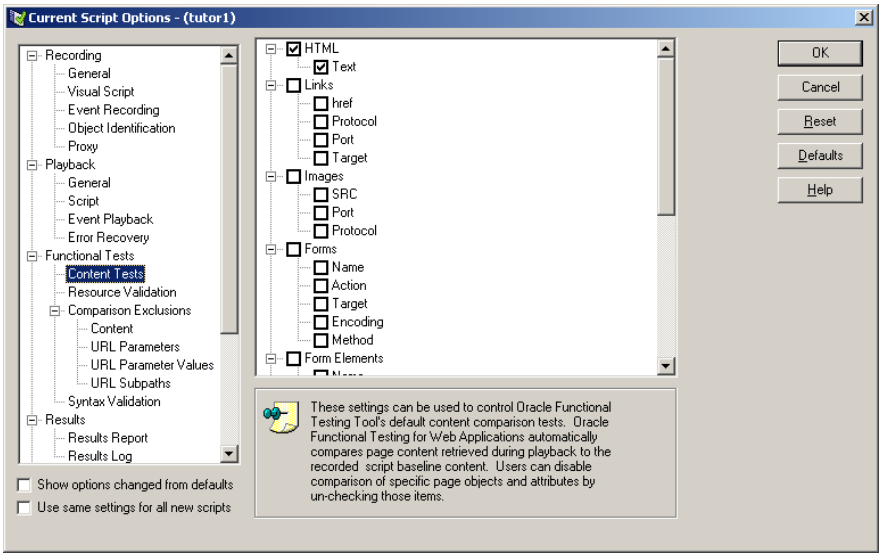
8. Select the Frame node in the Visual Script.
9. Click the right mouse button and select **Don't Test Frame**. Notice a small yellow circle appears next to the Frame node to indicate that the automatic existence test for the frame is turned off, as shown below:

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.

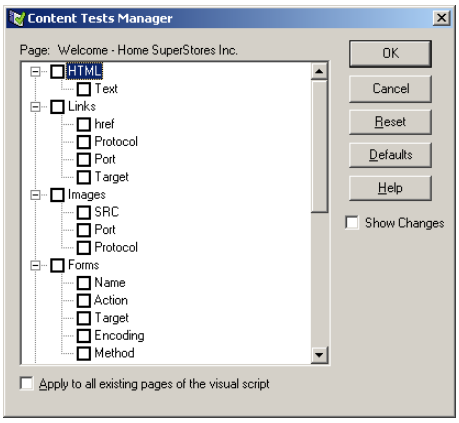
Modifying Default Tests

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



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 using **Options → New Scripts (Global) → Functional Tests → Content Tests**.

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



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.

17. 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:

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

Example 3

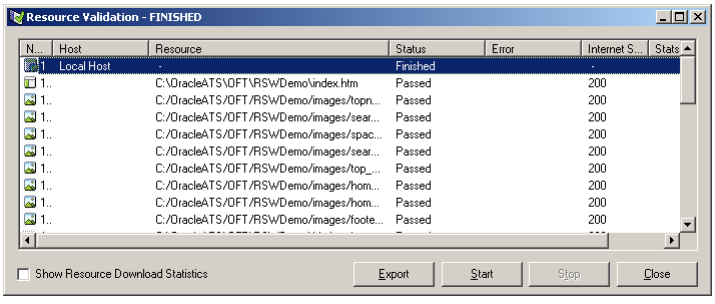
Playing Back a Visual Script

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 **Options → Current Script** and 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 **View → Resize Visual Script View** to expand the Browser pane.

5. 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.

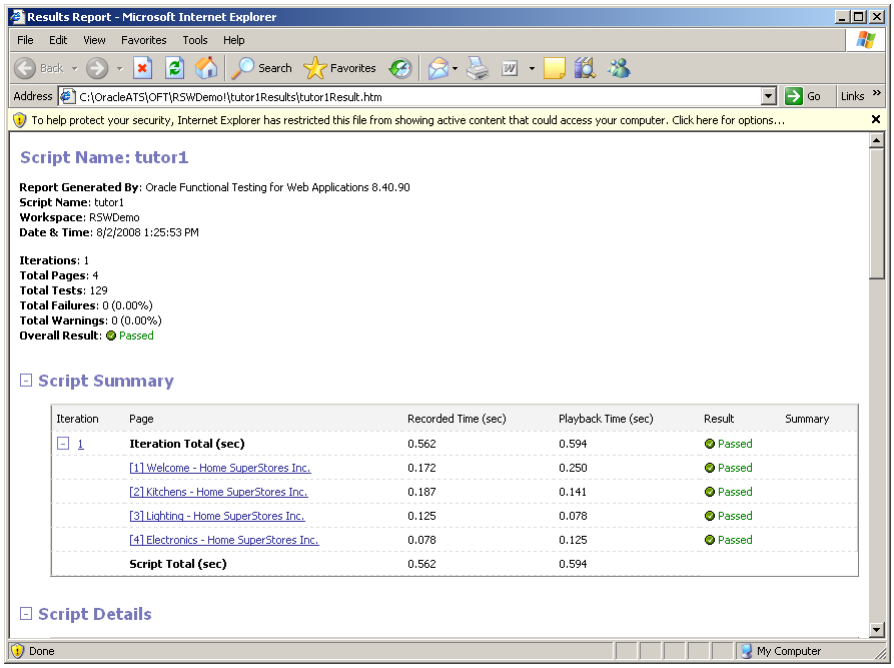
   The Resource Validation test checks the integrity of the referenced resources (i.e. links, images, etc) in your pages.



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

6. 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:



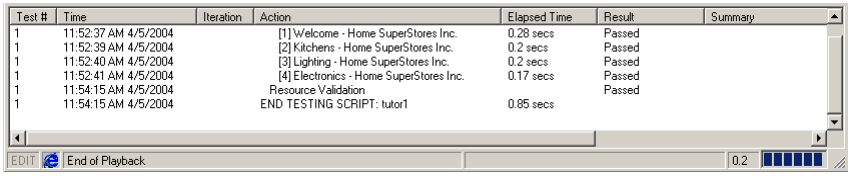
Note: The Results Report uses active content. If the links in the report are not working, check if the browser shows the restricted active content security warning at the top of the browser. If so, click on the warning and select **Allow Blocked Content**. Use the browser Internet Options to set the default settings for allowing active content. Select **Options → Browser → 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.

6. 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.

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

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



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		

EDIT End of Playback 0.2

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

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 **Start → Programs → Oracle Application Testing Suite → Build B - Home Superstores**. This runs a batch file that updates the Home Superstores page to a new version.
2. If necessary, close the DOS 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:

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

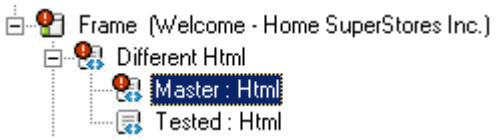
encountered upon Visual Script playback can be rejected, ignored, or can be accepted to create a modified baseline script.

6. Select **Results → Find Next Failure**, 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.

Ignoring Failures

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

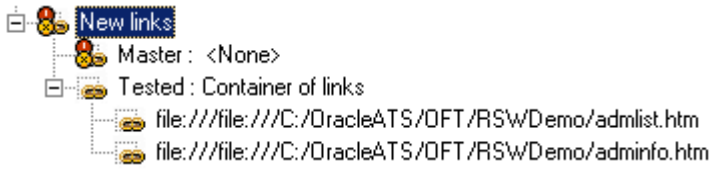
7. Locate the Master: Html node under the Different Html node.



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:

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

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.

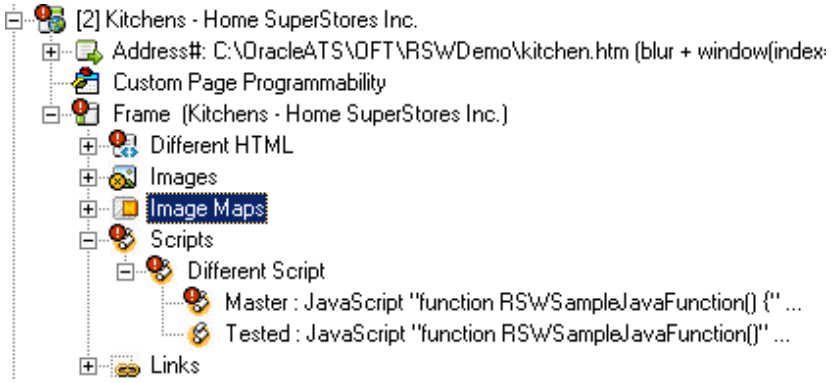


Accepting Changes Shown in the Script

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.



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.

```
<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>
```

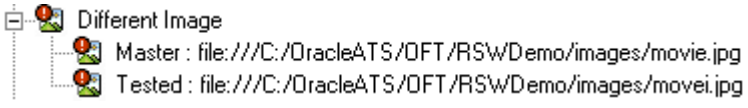
    Notice that the changes are textual changes to the Web page content. The Master text (baseline for testing) is shown in blue. The tested text (new version) is shown in red.
14. Click the **Cancel** button to close the window.
15. Repeat the same process for the Different Script node.
16. Select the [2] 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.

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].



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.

```
<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 **File → Save Output Log As**.

25. Enter the name tutor1.log and click **Save**.

26. Select **File → Save Script** to save the changed Visual Script.

27. Select **View → Clear Results Window** to clear the results log pane.

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
- Custom Object
- Custom Variable
- Table Test
- WinForms Test
- Siebel Test

Record a New Script

1. Select **Start → Programs → Oracle Application Testing Suite → Build A - Home Superstores**. This batch file restores the original Web pages for the Home Superstores site.
2. If necessary, close the DOS window.

3. Select **File → New Script** 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 he registration page (regres.htm).

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.

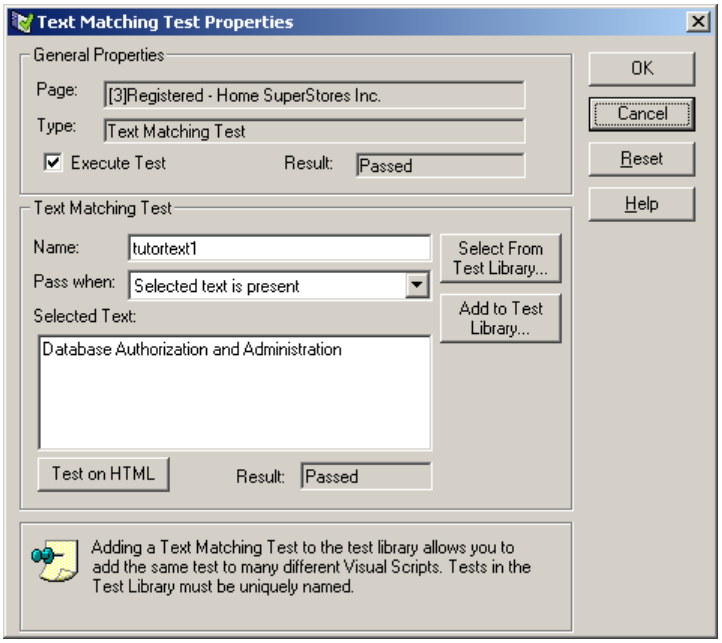
Insert a Text Matching Test Case

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.



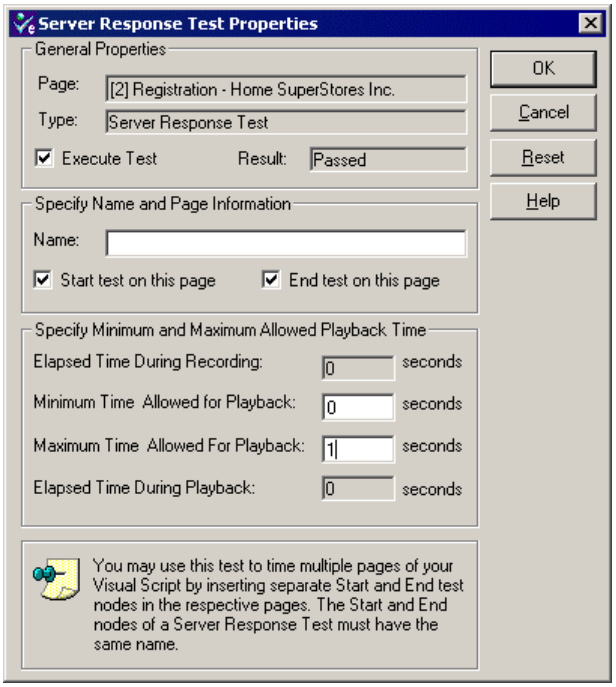
6. Type tutortext1 as the test case name.
7. Make sure the **Pass when:** option is set to **Selected text is present**.
8. Click **OK** and view the test case in the Visual Script. Oracle Functional Testing for Web Applications adds the test case to the Visual Script under the Frame node.

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.



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.

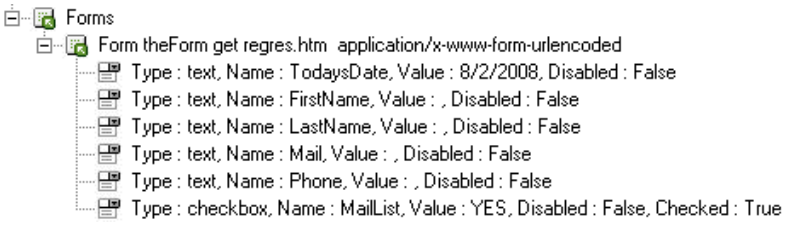
Insert a Form Element Test Case

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

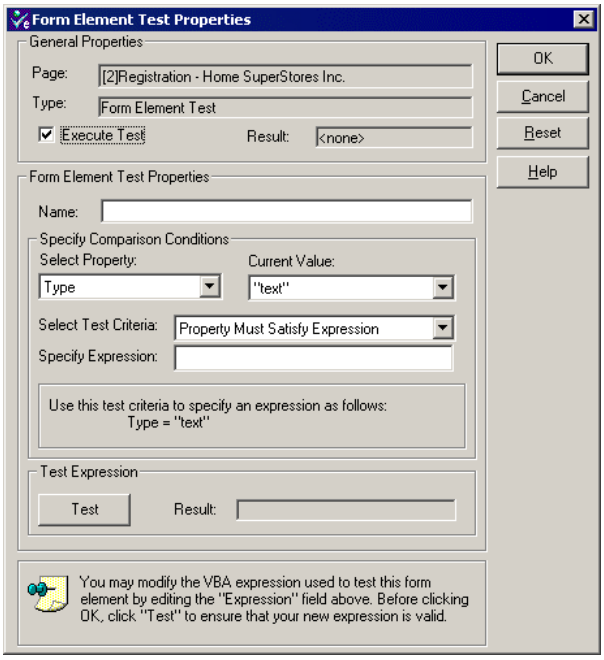
14. Select **Options → Current Script** and select **Content Tests** in the **Functional Test** section. Make sure the **Forms** and **Form Elements** check box are selected and click **OK**.

15. Select the [2] Registration - Home Superstores, Inc. item in the Visual Script.

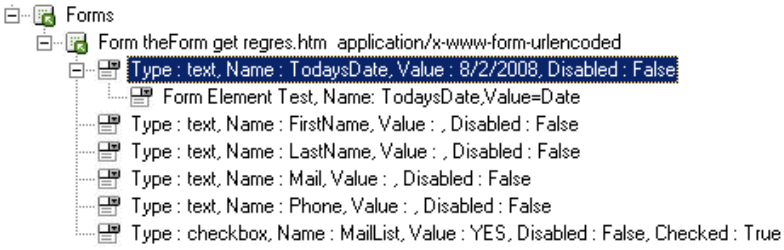
16. Expand the page and select the TodaysDate element of the regres.htm form.



16. Select **Tests → Insert Form Element Test**. Oracle Functional Testing for Web Applications opens the Form Element Test dialog box.



17. Type TodaysDate as the test case name.
18. Set the **Select Property** option to Value.
19. Set the **Select Test Criteria** to Property Must Satisfy Expression.
20. Change the **Specify Expression** field to Value = Date.
21. Click the **Test** button. Oracle Functional Testing for Web Applications should return True in the **Result** field.
22. 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.



Insert Table Test Test Case

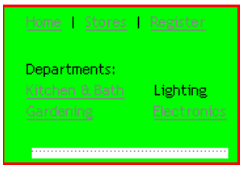
Table test test cases let you define a custom test on a Web page table object. The Table Test Wizard lets you select the table object directly from the Web page by highlighting it with the mouse. The wizard also lets you specify the table object property to test and the type of test to perform.

23. Select [1] Welcome - Home Superstores, Inc. in the Visual Script.
24. Click the right mouse button and select **Goto Page** to open the Welcome - Home Superstores, Inc page in the Browser.

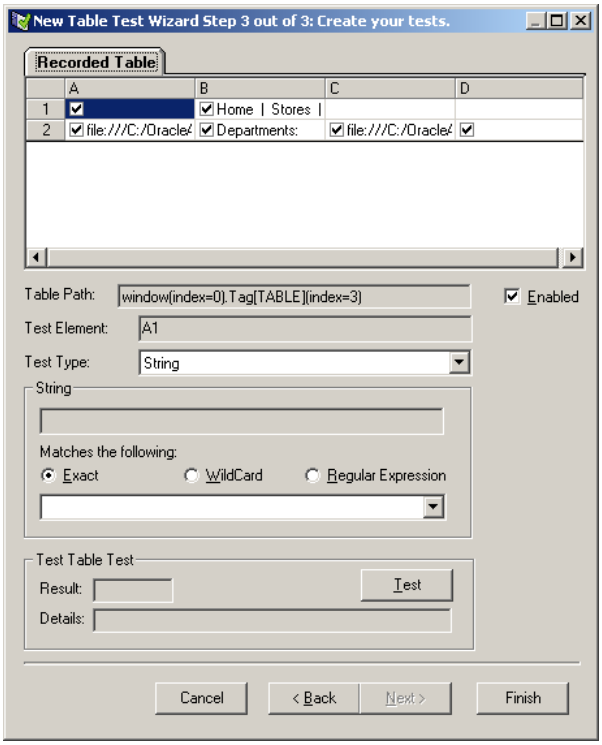
25. Select **Tests → Insert Table Test**. Oracle Functional Testing for Web Applications opens the Table Test Wizard.

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

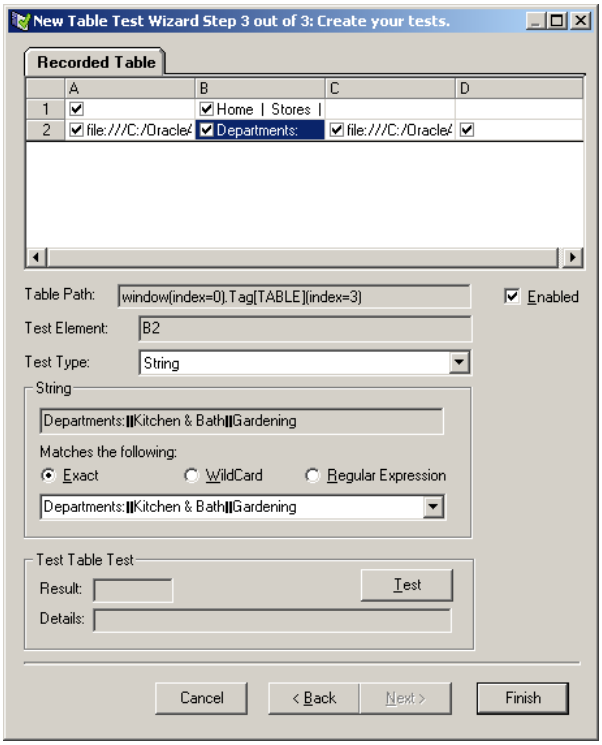
28. Moving the cursor over the web page highlights the tables on the page. Highlight the Departments area in the upper left-hand corner of the page.



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



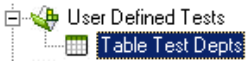
30. This dialog box displays the table in a spreadsheet format, showing the data that is in each table cell. The dialog box options change depending on the type of data in the cell. Cells that are checked will be tested. Select cell B2.



31. Make sure that Exact is selected in the Matches the following field.

32. Click **Test** to display the Last Played Table Tab. The result is displayed in the Result field.

33. 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.



34. Save the script as tutor2.

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 **File → New Script** 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.

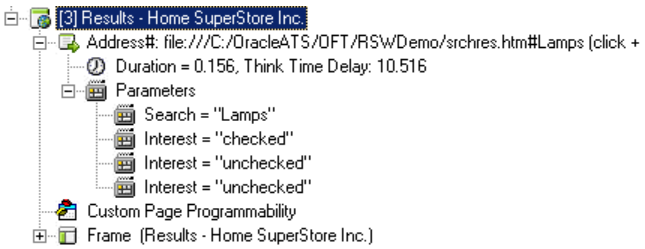
Recording a Search

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.

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.

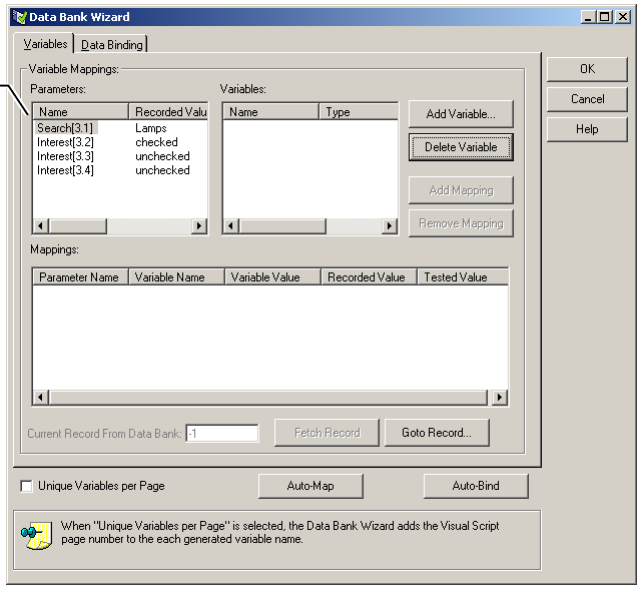


Using the Data Bank Wizard to Map Variables

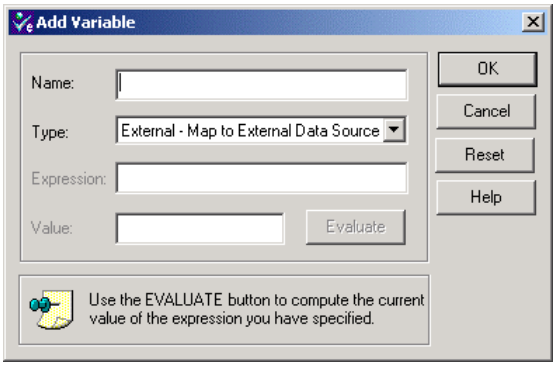
8. Select **Edit → Data Bank Wizard**.

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

Script[3.1] of the script

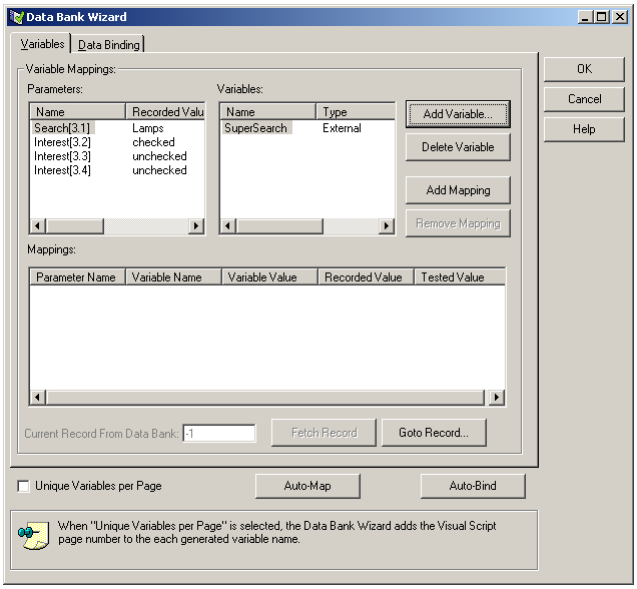


9. Click the **Add Variable** button. The Data Bank Wizard opens a dialog box for specifying a variable name.



10. Type SuperSearch as the variable name, and then click **OK**.

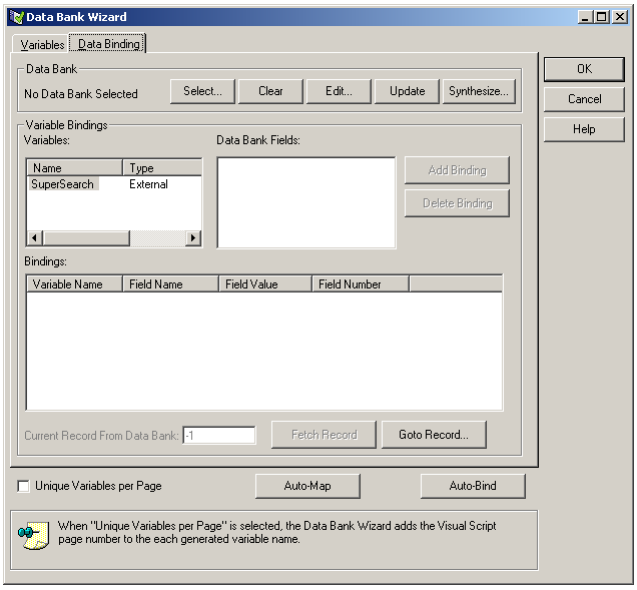
    The Data Bank Wizard adds the name to the **Variables** list.



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.

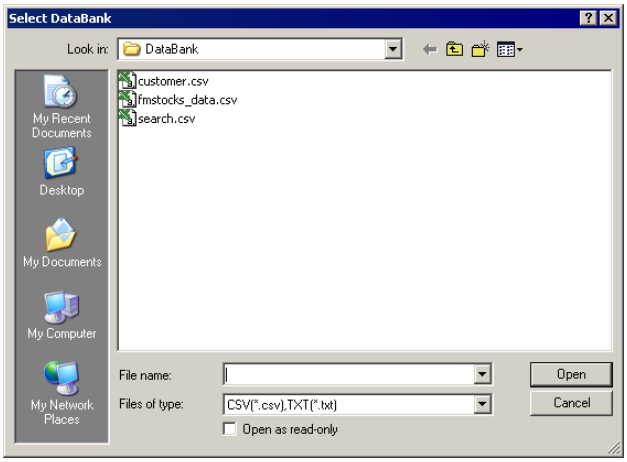
Using the Data Bank Wizard to Bind to a Data Bank

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



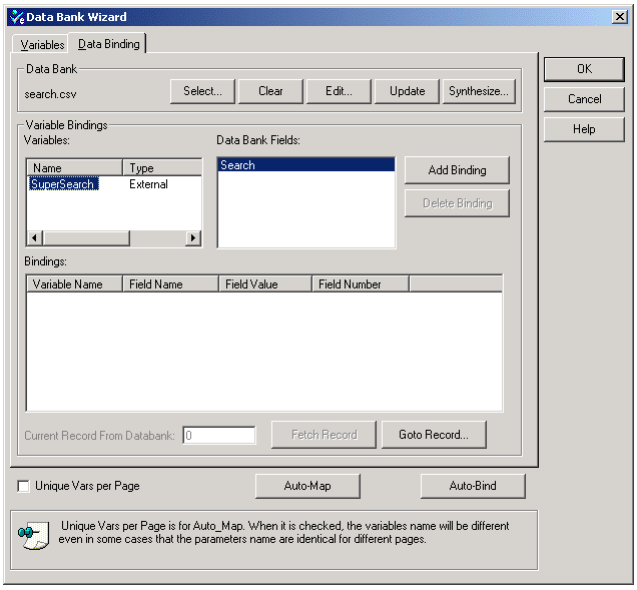
Now we want to select the Data Bank file that contains the values we want to use for iterative testing.

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



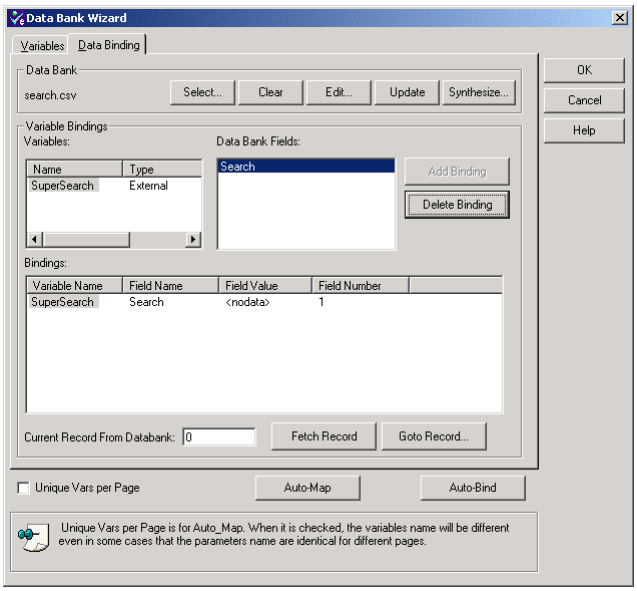
NOTE: The .csv file name extension may or may not be visible depending upon your system settings.

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

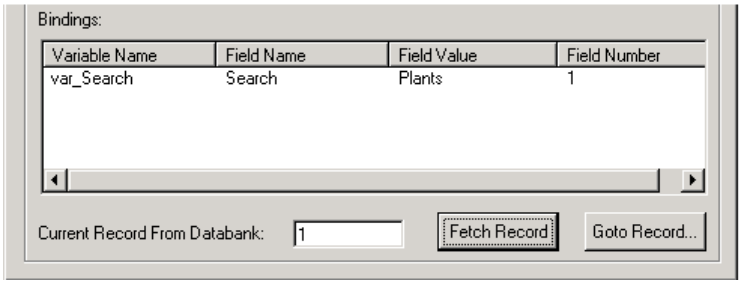


NOTE: The Data Bank file is a comma-delimited ASCII file with the field names as column headers on the first line of the file. Subsequent lines of the file contain data. You can view the contents of any of the sample files in the c:\OracleATS\OFT\DataBank directory using Notepad or any other ASCII editor.

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.



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

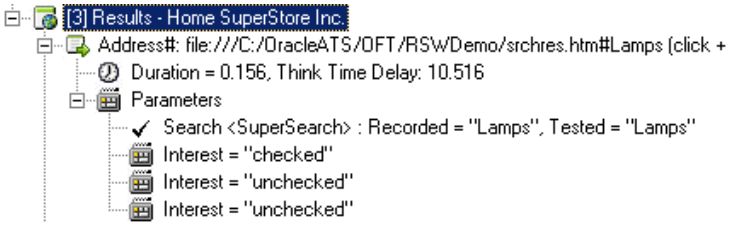


17. Continue clicking the **Fetch Record** button to cycle through all of the records in the file. There are five records in the sample Data Bank file.

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

View the Data Bank Parameters in the Visual Script

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



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.

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**.

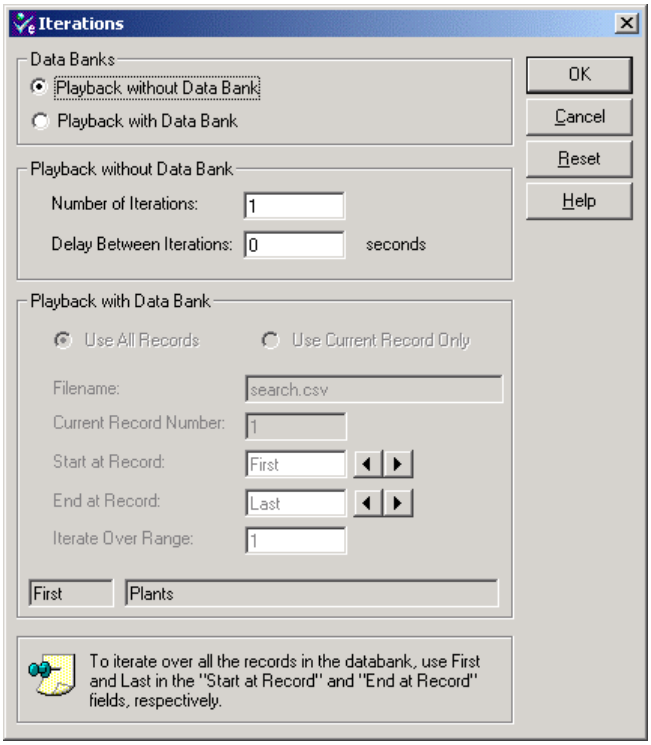
23. Select the HTML node in the Visual Script. Notice the yellow flag next to the HTML node. Oracle Functional Testing for Web Applications automatically turns off the HTML comparison test when you insert a Text Matching test.

NOTE: 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.

26. Save the Visual Script as tutor3.

Play Back the Script with Iterations

27. Select **Run → Playback → Iterate**. Oracle Functional Testing for Web Applications opens the Iterations dialog box.



28. Select **Playback with Data Bank** and **Use All Records**.

29. Click the **OK** button to playback the Visual Script.

30. Watch as Oracle Functional Testing for Web Applications plays back the script several times using a different data value for the search each time.

Analyzing a Playback Failure

31. 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.

32. Select **Edit → Data Bank Wizard**.

32. Click the **Goto Record** button. The Data Bank Wizard opens a dialog box for entering the record number.

33. Enter 3, and then click **OK**.

34. Click **OK** to close the Data Bank Wizard.

35. Click the Playback button on the toolbar to play back this one record.

36. 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.

37. Make sure the [3] Results - Home Superstore, Inc page is still selected in the Visual Script.

38. 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.

Save the Script and the Results Log

40. Save the Visual Script.

41. Save the Output Log as tutor3.log.

42. Select **View → Clear Results Window** to clear the results log for the next test.

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 **File → New Script** 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.

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.

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.

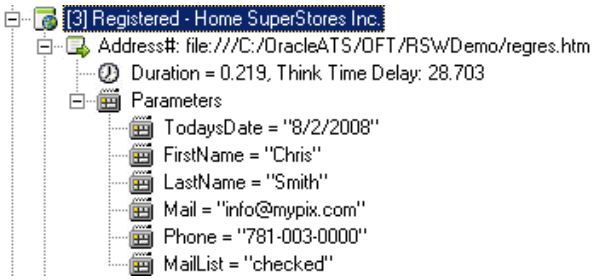
8. Type VerifyRegistration as the test case name, make sure **Pass when**: is set to **Selected text is present**, and then click **OK**.

9. Click the Stop Record button on the toolbar.

10. Select **File → Save Script As** and save the file as tutor4.

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.

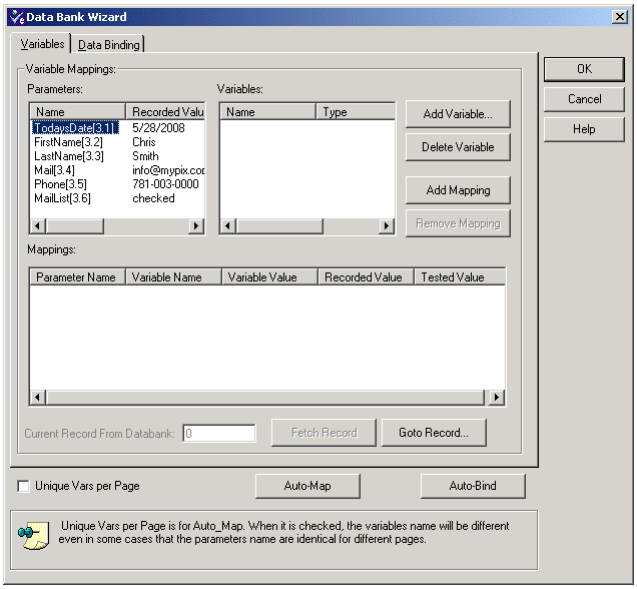


12. Make sure the [3] Registered - Home Superstore, Inc page is selected in the Visual Script.

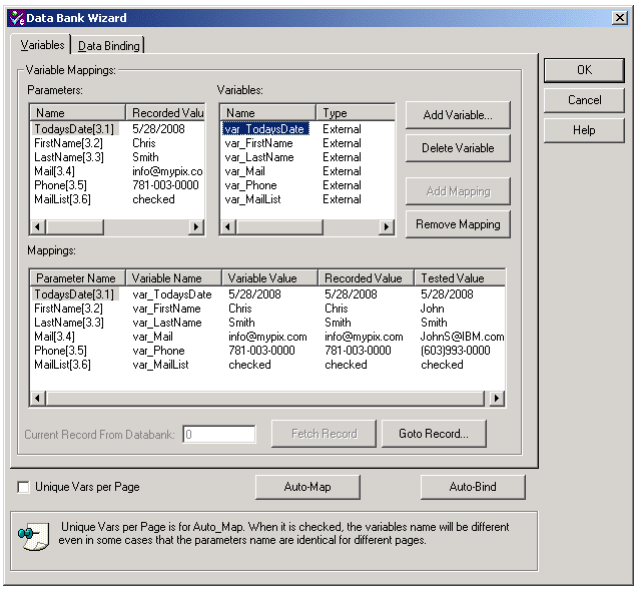
Using the Data Bank Wizard to Map Variables.

13. Select **Edit → Data Bank Wizard**.

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



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

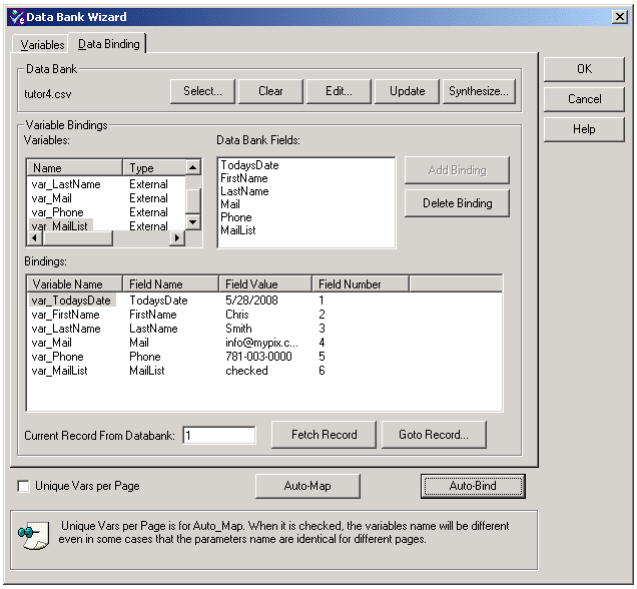


You now need to bind the variable names to fields in a Data Bank file.

Using the Data Bank Wizard to Bind to Data Source

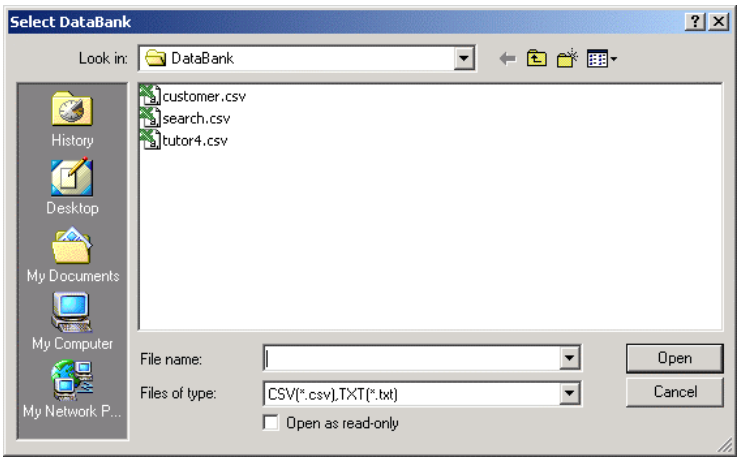
15. Click the **Auto Bind** button. The Data Bank Wizard opens the Data Binding options with the variable names in the **Variables** list.

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.



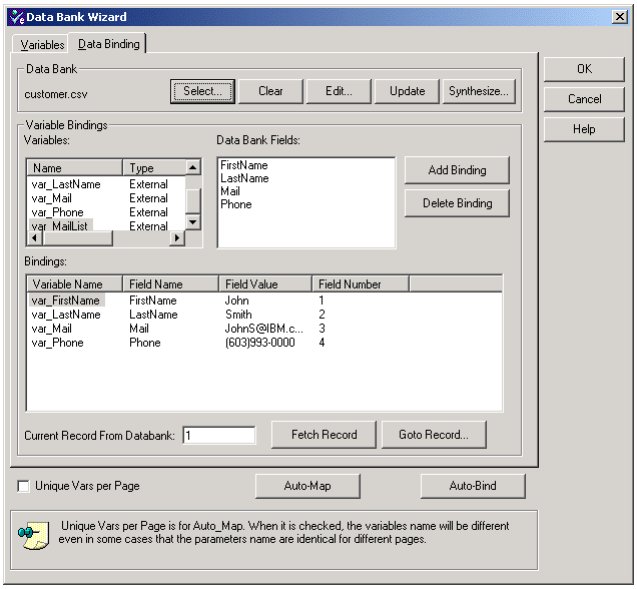
Now we want to select the data source that contains the values we want to use for iterative testing.

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

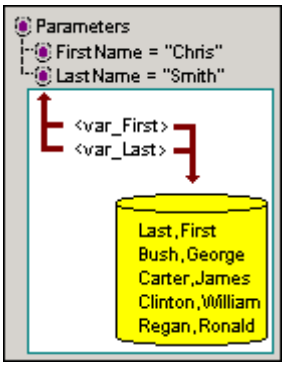


17. Select customer.csv, and then click **Open**.

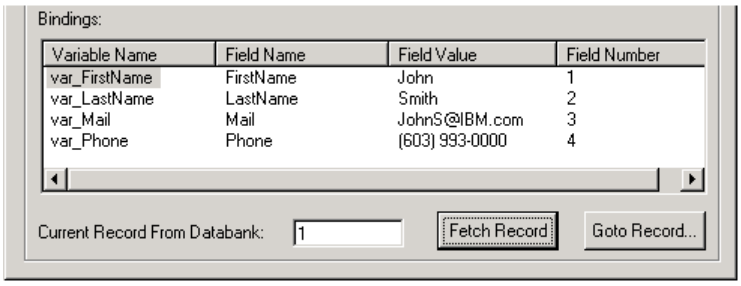
    The Data Bank Wizard automatically re-binds the appropriate data field names from the Data Bank file to the variable names.



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.



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

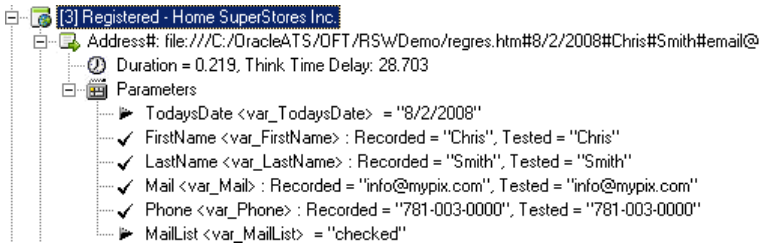


19. Continue clicking the **Fetch Record** button to cycle through all of the records in the Data Bank file.

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

View the Data Bank Parameters in the Visual Script

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



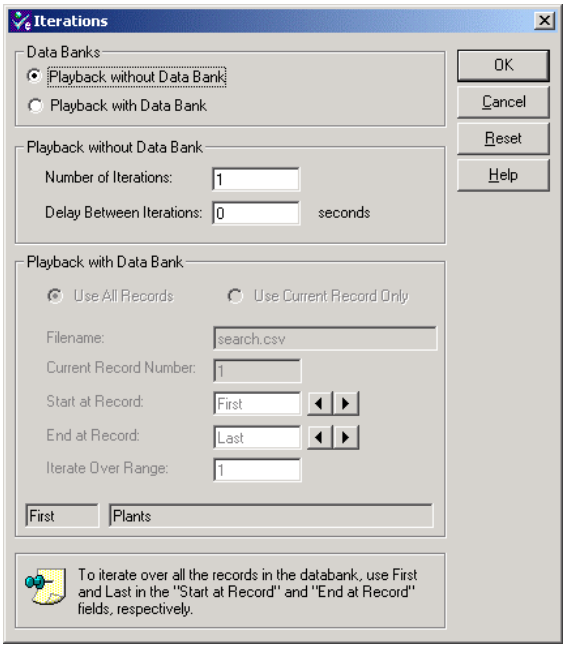
The Visual Script now includes the variable names as part of the Parameters. The check marks (✓) indicate the parameters that are mapped to variables and bound to fields in a Data Bank file. The triangles (▶) indicate variables that are mapped, but not bound to fields in a Data Bank file.

22. Save the Visual Script.

Play Back the Script with Data Iterations

23. Select **Options → Current Script → Results Log**, make sure the **Failures Only** radio button in the **Results** section is selected, and then click **OK**.

24. Select **Run → Playback → Iterate**. Oracle Functional Testing for Web Applications opens the 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.

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 **Edit → Data Bank Wizard**.
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.

> **NOTE**: You can play back the current record repeatedly using **Options → Playback**, and then selecting 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.

Save the Script and the Results Log

37. Save the Visual Script.
38. Save the Output Log as tutor4.log.
39. Select **View → Clear Results Window** to clear the results log for the next test.

Example 8

Using Custom Tests

This example explains how to use custom VBA tests with Visual Scripts. Custom VBA tests are Visual Basic code added to the Oracle Functional Testing for Web Applications Visual Scripts to extend testing capabilities. Oracle Functional Testing for Web Applications includes the Visual Basic for Applications development environment for developing custom test code.

Note: Microsoft's Visual Basic for Applications (VBA) will soon be deprecated from Oracle Application Testing Suite as Microsoft has announced (see readme.htm) the discontinuation of this technology. You can enable VBA in Oracle Application Testing Suite through the Oracle Functional Testing for Web Applications **Options → New Scripts (Global) → General** menu, but be advised that this technology will soon be replaced with alternate technologies that will provide scripting extensibility and bring more power and flexibility to test automation.

Enabling Custom VBA Testing

To enable custom VBA testing:

1. Select **Options → New Scripts (Global)**. Oracle Functional Testing for Web Applications opens the Custom Variable Test Wizard.
2. Select the top level General category.
3. Select the **Enable VBA support in Oracle Functional Testing for Web Applications** option in the **VBA Options**.
4. Click **OK**.

Note: When VBA support is disabled, the Custom Page Programmability node does not appear in the Visual Scripts. The **Enable VBA support in Oracle Functional Testing for Web Applications** option can be enabled for existing Visual Scripts that contain VBA code.

Using the Custom Test Wizard

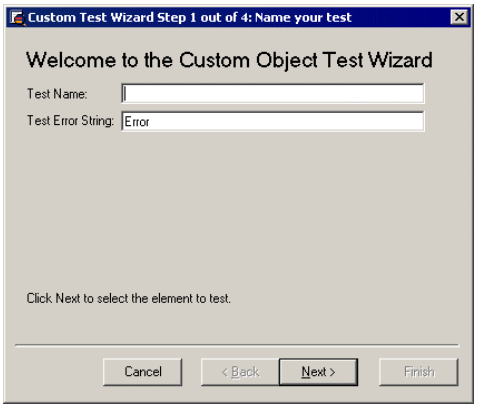
Oracle Functional Testing for Web Applications includes a Custom Test Wizard that lets you select Web page objects, specify test options, and automatically generate Visual Basic code.

1. Open the Visual Script Tutor1.

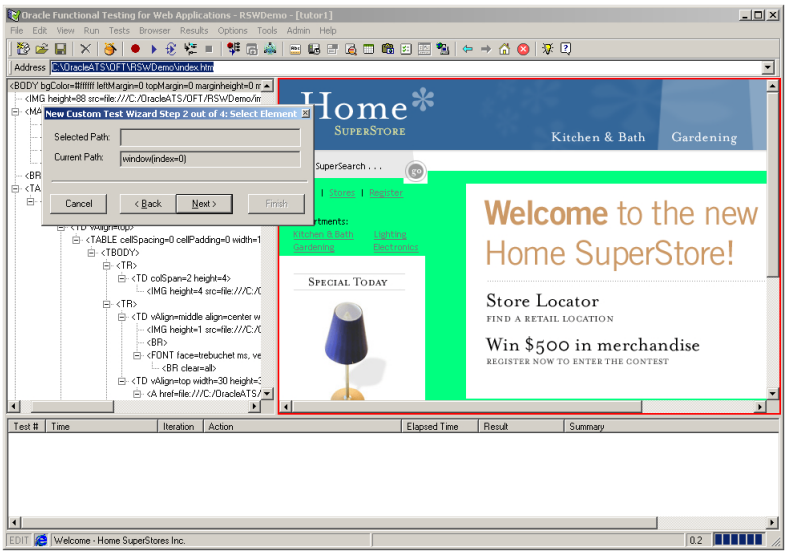
   Note: The page where you want to add the custom test must be the current page displayed in the Browser pane.

2. Select [1] Welcome - Home SuperStores page of the Visual Script.
3. Click the right mouse button and select **Goto Page**.

4. Select **Tests → Insert Custom Object Test**. The Oracle Functional Testing for Web Applications opens the Custom test Wizard. The following illustration shows the Custom Test Wizard:



5. Type CustomTest1 as the test name and click **Next**. Oracle Functional Testing for Web Applications opens a window over Oracle Functional Testing for Web Applications that shows the Selected and Current Path information.

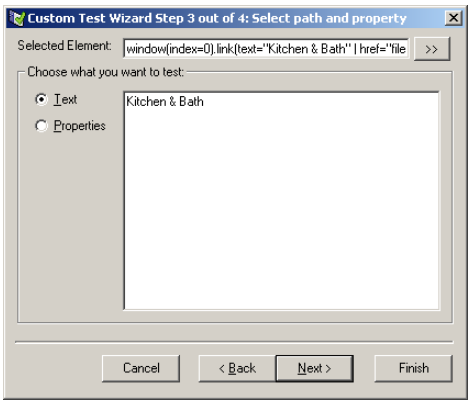


The Current Path is the object model location of the Web page object currently beneath the mouse cursor. The objects in the browser pane change color to highlight the current Web page object. The Visual Script pane shows the Web page hierarchy and highlights the current object.

6. Move the mouse cursor over the Web page to see how the Custom Test Wizard highlights Web page objects both in the Browser pane and the Visual Script pane.

7. Click the mouse cursor on the Kitchen & Bath link. The object model path is added to the Select Element window as the Selected path.

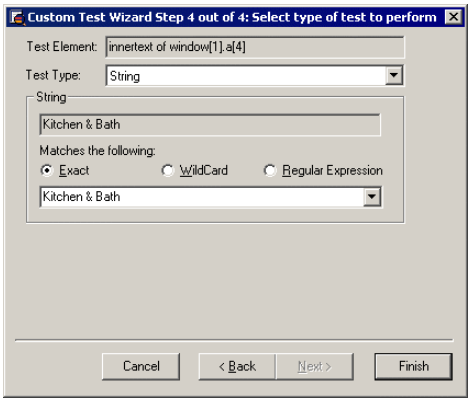
8. Select **Next** in the Select Element window. The Custom Test Wizard reopens.



9. Select the **Properties** option. The element properties are listed in the wizard. The Custom Test Wizard lets you specify tests for specific Web page element properties. For this tutorial, we'll create a test on the text rather than a property.

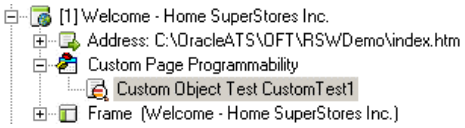
10. Select the **Text** option.

11. Click **Next**. The Custom Test Wizard opens a panel for selecting the type of test to perform.

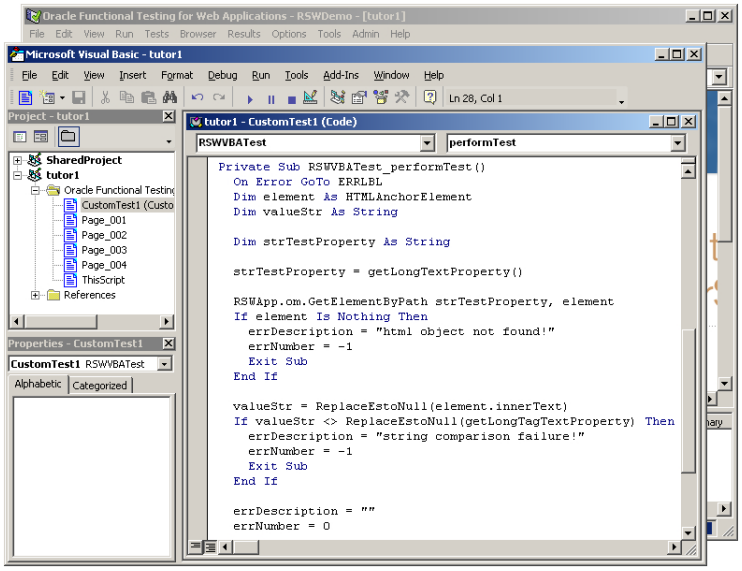


12. Leave the default values and click **Finish**.

13. The custom test is added as a node below the Custom Page Programmability node.



14. Right click the Custom Object Test CustomTest1 node and select **Edit VBA Code** to open the VBA development environment and generates the Visual Basic project and code for the test.



You can customize the Visual Basic code as necessary for specific testing purposes.

15. Select **File → Close and Return to Oracle Functional Testing for Web Applications**.

16. Playback the Visual Script. The custom test verifies the text of the Kitchen & Bath link during playback of the Visual Script.

17. Close the Resource Validation window.

Inserting Custom Page Tests

In addition to using the Custom Test Wizard, you can create your own custom Visual Basic tests that execute before or after playback of a Visual Script page.

Displaying Links

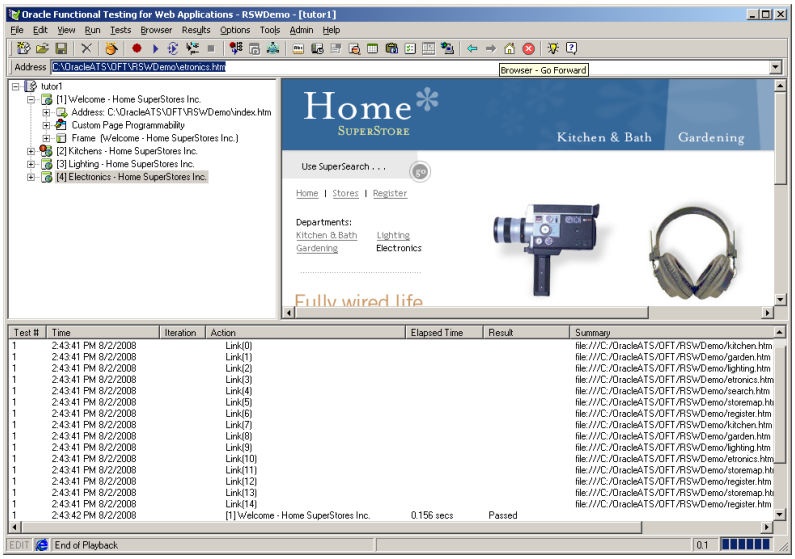
1. Expand the [1] Welcome - Home SuperStores page of the Visual Script.
2. Select the Custom Page Programmability node and click the right mouse button.
3. Select **Edit VBA Code**. Oracle Functional Testing for Web Applications opens the VBA development environment with the code window for the page.
4. If you have the Oracle Application Testing Suite CD-ROM, open the Sample EPI Scripts\VBA directory. Copy the Get All Links text from the Get All Links subdirectory into the `afterPlay` event of the `RSWVBAPage` object.

   If you do not have the Oracle Application Testing Suite CD-ROM, enter the following code into the `afterPlay` event of the `RSWVBAPage` object:

   ```
   Dim topdoc As Object
   Dim numlinks As Long
   Dim hrefVal As String
   Dim linkIndex As Long
   Dim indexVal As String

   Set topdoc = RSWApp.om.GetTopDocument(0)

   numlinks = topdoc.links.Length

   If numlinks <> 0 Then
      For linkIndex = 0 To numlinks - 1
          hrefVal = topdoc.links(linkIndex).href
          indexVal = "Link(" + CStr(linkIndex) + ")"
          Call RSWApp.WriteToLog(indexVal, "", hrefVal)
      Next
   End If
   ```

5. Select **File → Close and Return to Oracle Functional Testing for Web Applications**.

6. Playback the Visual Script and close the Resource Validation window. The sample custom test should display a list of all links in the Web page with an index value, as follows:



The code for Custom tests can access the Document Object Model of the Web page, evaluate and modify elements and properties, access Data Bank values, and change Visual Script page navigation programmatically based upon specific conditions.

Oracle Functional Testing for Web Applications includes the e-Test Programming Interface (e-PI) that provides several methods and properties that you can use when developing custom tests. See the *Oracle Functional Testing for Web Applications User's Guide* and *Application Programming Interface Reference* for more information about using custom tests to extend your Visual Script testing capabilities.

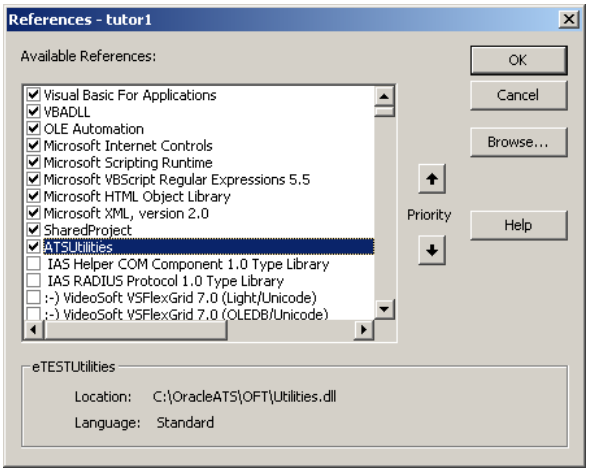
7. Save the Visual Script.

Accessing a DLL

This example shows how to access methods in a registered COM DLL. This example will access the Random method in ATSUtilities and

generate a random day in the Summary column of the results window and log.

1. Expand the [2] Kitchens - Home SuperStores page of the Visual Script.
2. Select the Custom Page Programmability node and click the right mouse button.
3. Select **Edit VBA Code**. Oracle Functional Testing for Web Applications opens the VBA development environment with the code window for the page.
4. Select **Tools → References** and select ATSUtilities.

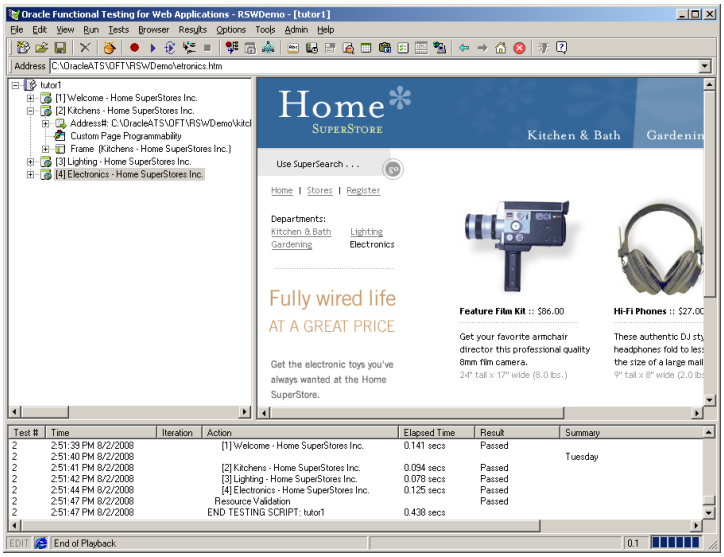


5. Click **OK**.
6. Enter the following code into the `afterPlay` event of the `RSWVBAPage` object:

```
Dim m_util As ATSUtilities.Random
Set m_util = New ATSUtilities.Random

Dim rday As String
rday = m_util.randomDay()

Call RSWApp.WriteToLog("", "", rday)

Set m_util = Nothing
```

7. Select **File → Close and Return to Oracle Functional Testing for Web Applications**.

8. Playback the Visual Script and close the Resource Validation window. The sample custom test should display a random day in the Summary column of the Results window.



9. Save the Visual Script and exit Oracle Functional Testing for Web Applications.

This completes the Oracle Functional Testing for Web Applications tutorial.

Chapter 4

Job Scheduler Tutorial

This tutorial walks you through the main features of Job Scheduler. Job Scheduler is a separate product in the Oracle Application Testing Suite, which you may or may not have purchased. If you have Job Scheduler version of Oracle Application Testing Suite, 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.
- **Adding Custom Code** – explains how to access the Visual Basic for Applications (VBA) editor and add custom code for Job Scheduler events.

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.

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.

Starting Job Scheduler

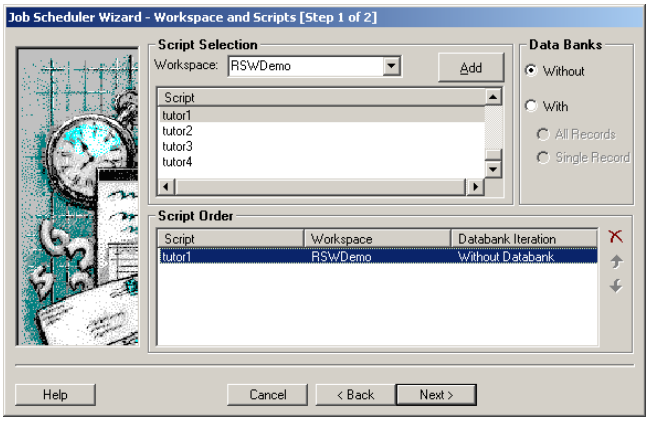
1. Select **Start → Programs → Oracle Application Testing Suite → Job Scheduler** to start Job Scheduler.

2. Select **File → New Job** 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 **Skip this screen in the future** check box.

3. Click the **Next** button to continue to the Workspace and Scripts [Step 1 of 2] screen.

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.

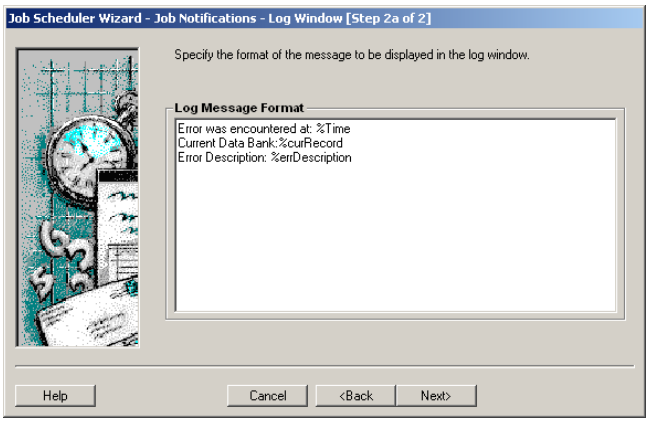


4. Select the RSWDemo workspace. Note that you can include scripts from more than one workspace in a job.

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.

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.



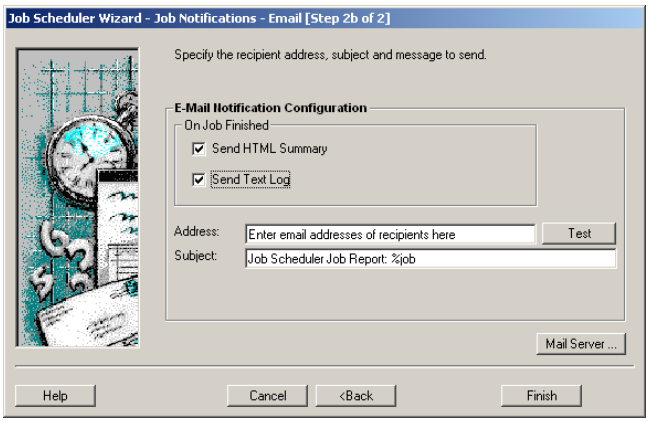
The default log message writes the following information to the results log if an error occurs during playback of a Visual Script:

- Time of the error
- Databank record
- An error number
- A description of the error

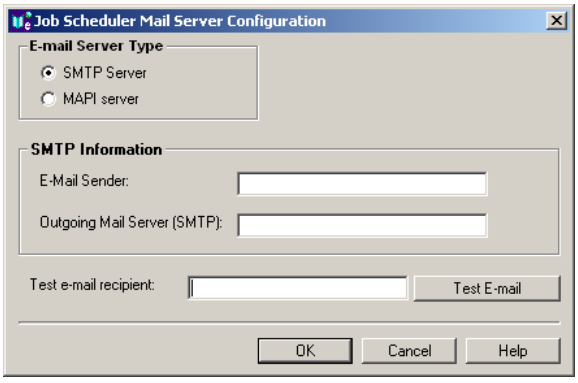
8. Click the **Next** button to continue to the Job Notifications Email [Step 2b of 2] screen of the Job Scheduler Wizard.

Specifying Email Notifications

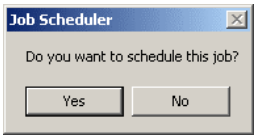
The Job Notifications Email [Step 2b of 2] screen of the Job Scheduler Wizard is where you specify who to notify when the job is finished and what information to send.



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.



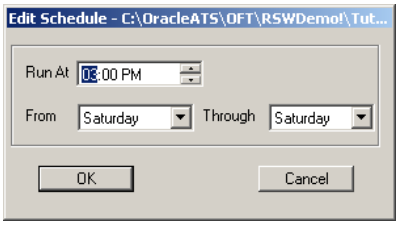
13. Select your email server type and enter the email information for your email system. Check with your system administrator if you are not sure of the information to enter for your email configuration. You can use the **Test email recipient** address and **Test Email** button to verify email capabilities 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**.

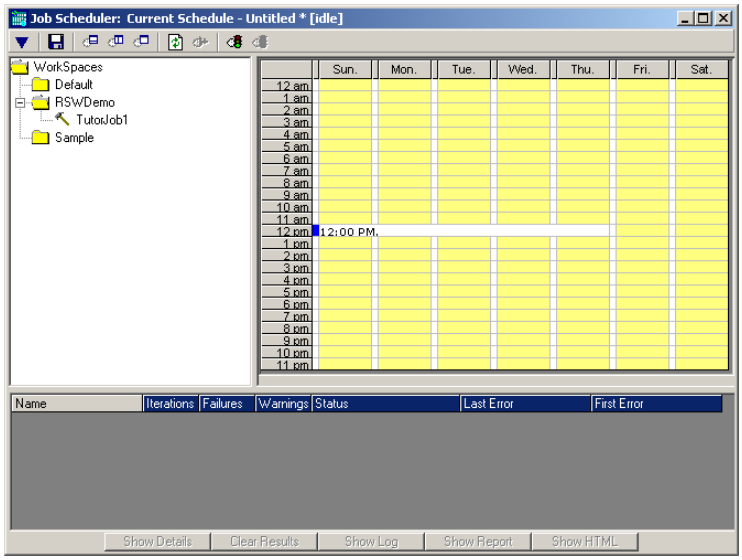
Scheduling the Job

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



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.**

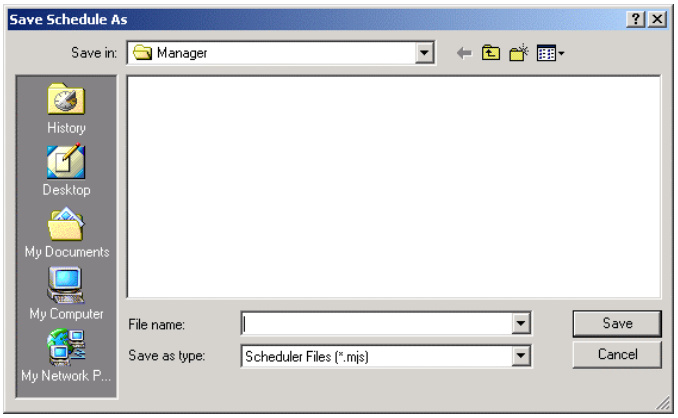
21. Click **OK** to schedule the job.



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 **File → Save Schedule As.**



2. Enter tutorSch1 for the schedule name and click **Save**.

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

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.

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 **Control → Activate Schedule** 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.

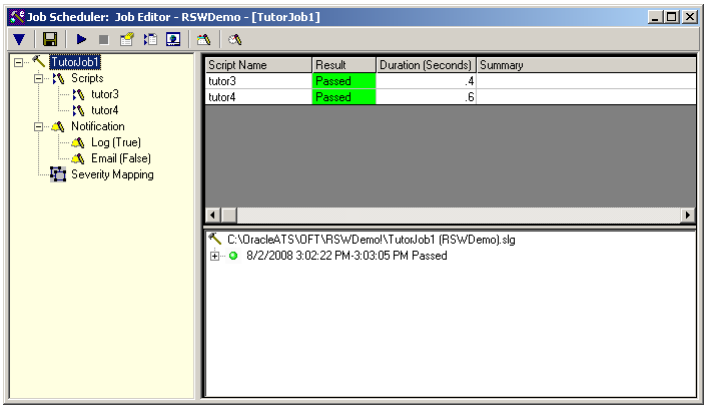
3. Select **Control → Deactivate Schedule** or click on the Deactivate Schedule toolbar button to deactivate the schedule.

Example 2

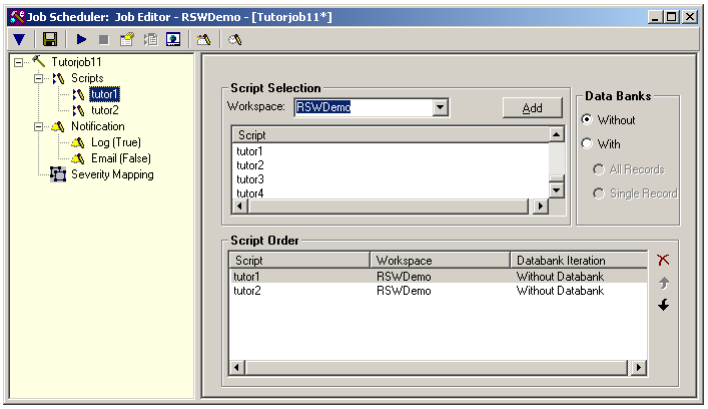
Editing a Job

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

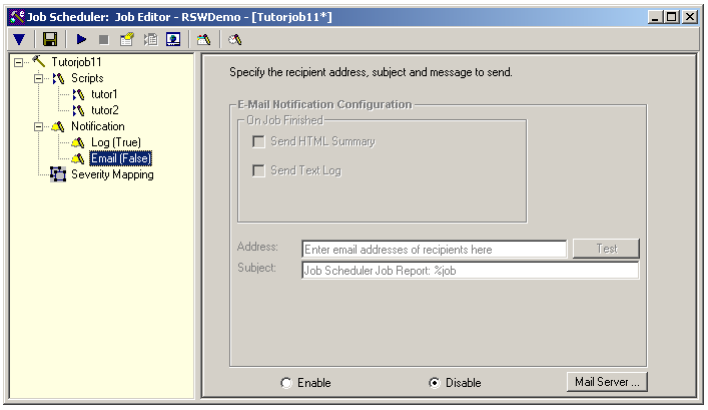


2. Click **Scripts** in the job tree to display the script selection options.



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.



5. Select the **Disable** option button to disable sending emails on error.

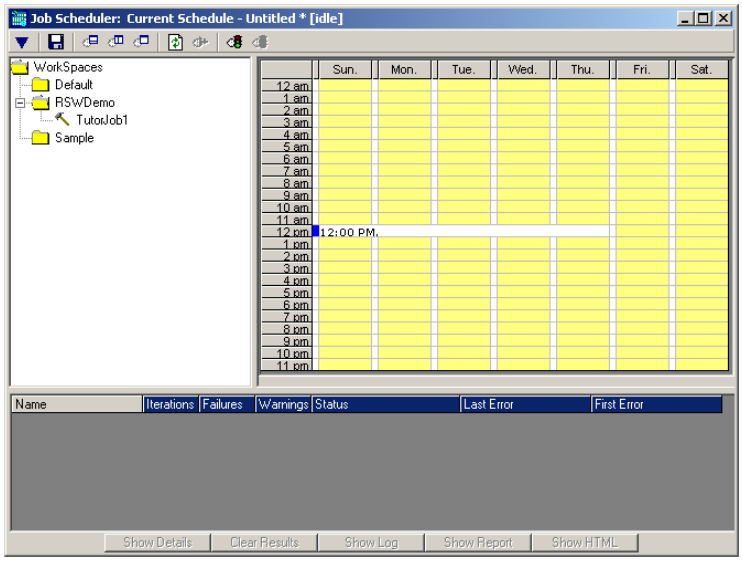
6. Save the job by selecting **File → Save tutorJob1**.

Example 3

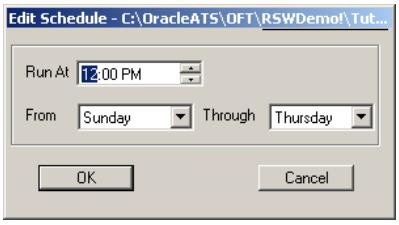
Editing a Schedule

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

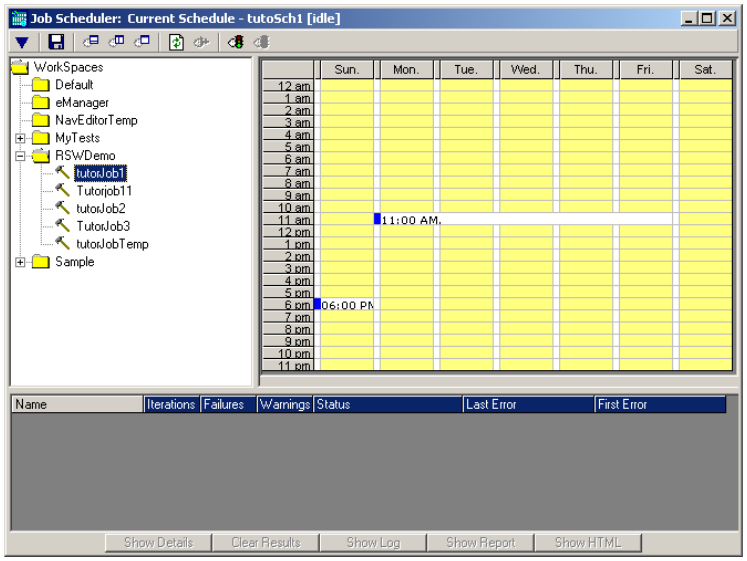


2. Double-click on the job in the calendar or right-click on the job and select **Edit Schedule**.



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.



5. Save the schedule by selecting **File → Save tutorSch1.**

Example 4

Adding Custom Code

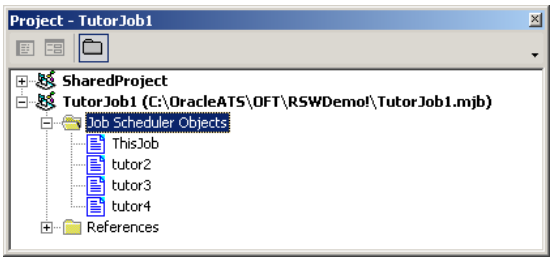
This example explains how to add Visual Basic custom code to your jobs.

Open the Visual Basic Editor

1. If it is not open, open the `tutorSch1.mjb` job by Selecting **File→ Open Job**. Select the file and click the **Open** button.

2. Select **Edit VBA** from the Job Editor menu or click the toolbar button to start the Visual Basic for Applications editor.

3. Expand the project in the Project – VBA Project dialog box and double-click on the ThisJob object to display the code window.



4. Select ThisJob from the object pulldown menu.

5. Select the Failure event from the event pulldown menu.

6. Enter the following code that will send a message to the log file and stop the job if it fails.

```
Private Sub ThisJob_Failure()
  'Send message to log and stop job
  Notifier.Log "Job Failed. Stopping Job."
  stopJob
End Sub
```

7. Select **File → Close and Return to Job Scheduler.**

8. Click on the **Run** Button in the Job Editor to test the script. The failure message displays in the Results Pane of Job Scheduler.
9. When finished, close Job Scheduler.

This completes the Job Schedular tutorial. Refer to the *Job Scheduler User's Guide* or online help for details about Job Scheduler objects and events.

[This page intentionally left blank]

Chapter 5

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 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.

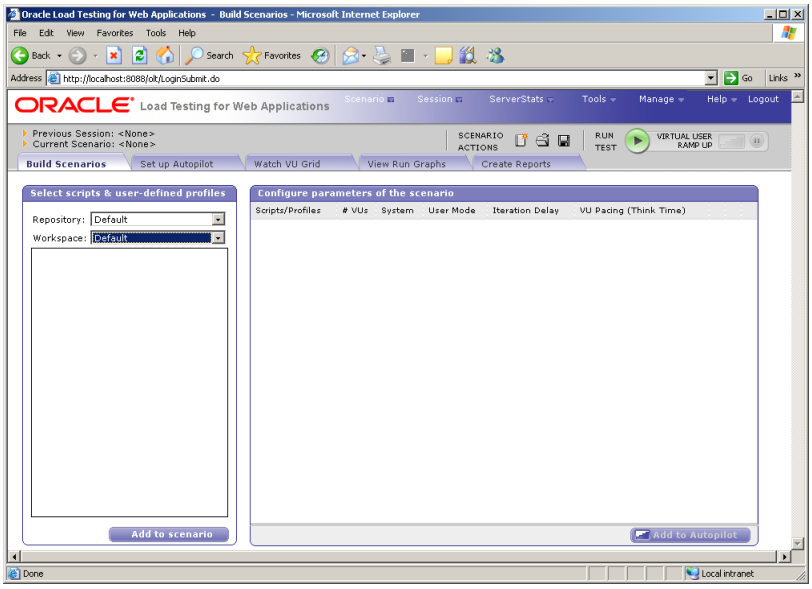
Example 1

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.

Starting Oracle Load Testing for Web Applications and Specifying the Workspace

1. Select **Start → Programs → Oracle Application Testing Suite → Oracle Load Testing for Web Applications** to start Oracle Load Testing for Web Applications.



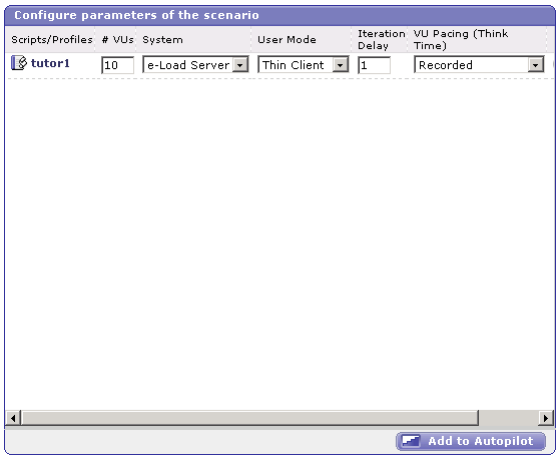
2. Select RSWDemo in the Workspace list.

Specifying a Scenario Profile

3. Make sure the **Build Scenario** tab is displayed in Oracle Load Testing for Web Applications.

4. Select tutor1 in the **Select scripts...** list. These are the Visual Scripts that you record using Oracle Functional Testing for Web Applications.

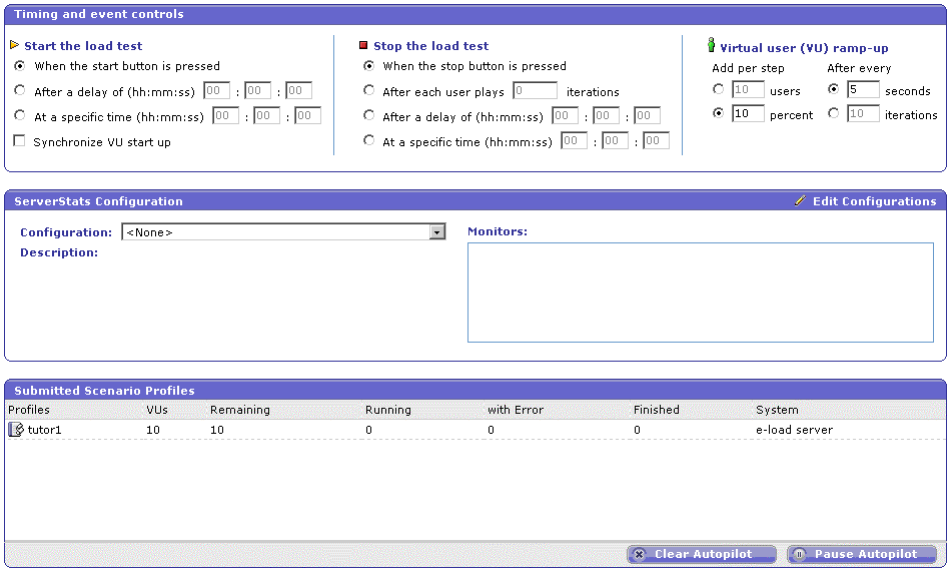
5. Click the **Add to scenario** button to add tutor1 to the **Configure Parameters** list. You can also double-click the script name to add it to the **Configure Parameters** list.



   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.

6. 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.

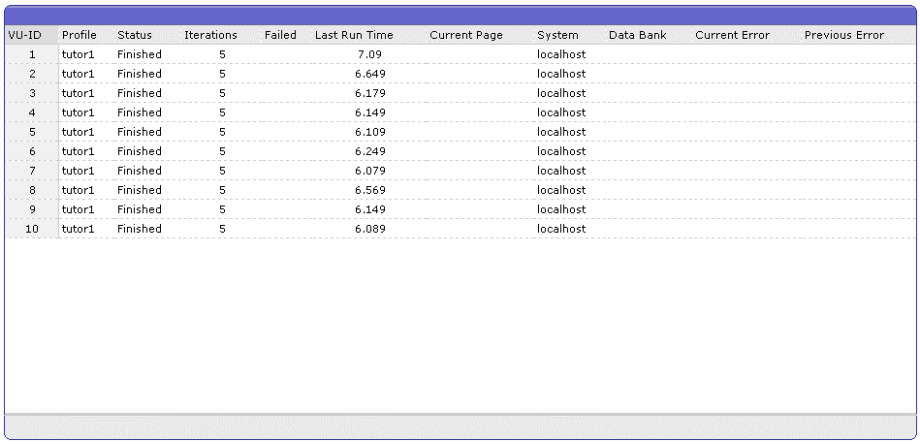


Running the Scenario Profile Using Autopilot

7. Select After each user plays [#] iteration option from the **Stop the load test** group of the Autopilot tab.
8. Enter 5 in the After each user plays edit box.
9. Select [#] users below **Add per step** in the Virtual User ramp up group.
10. Enter 1 in the Add per step [#] users edit box.
11. Select [#] seconds below **After every** in the Virtual User ram up group.
12. Enter 10 in the [#] seconds edit box.

14. Click the **Run Test** button on the Autopilot tab or the toolbar.
15. 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.



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			

16. 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.

Example 2

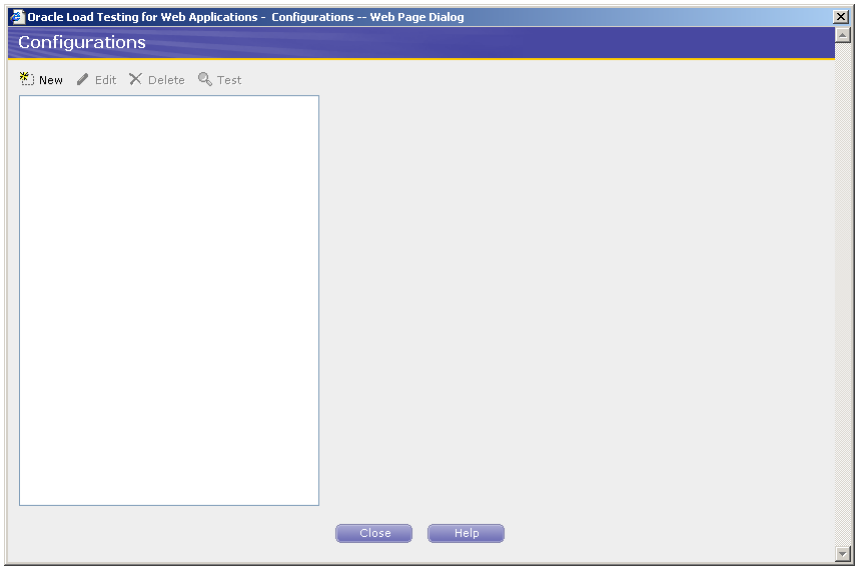
Adding Data Sources

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.

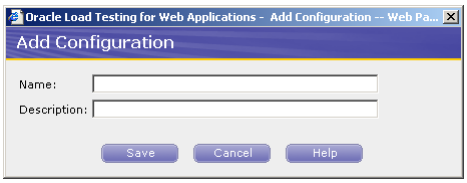
Note: Oracle Load Testing for Web Applications ServerStats can monitor statistics from a variety of systems and server types. This tutorial adds counters from your local Windows 200X/XP system to demonstrate the features of Oracle Load Testing for Web Applications ServerStats. If you are not running 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:

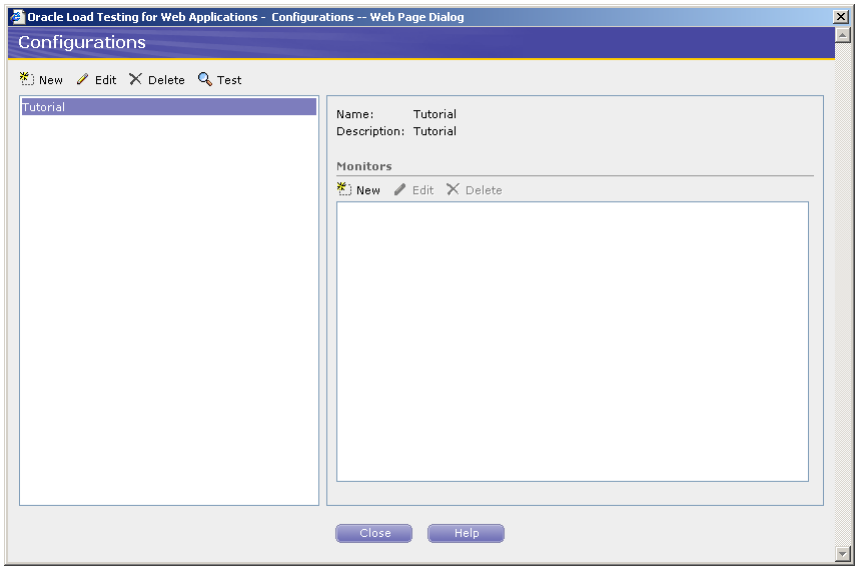
1. Select **ServerStats → Configurations**. Oracle Load Testing for Web Applications opens the ServerStats Configurations window.



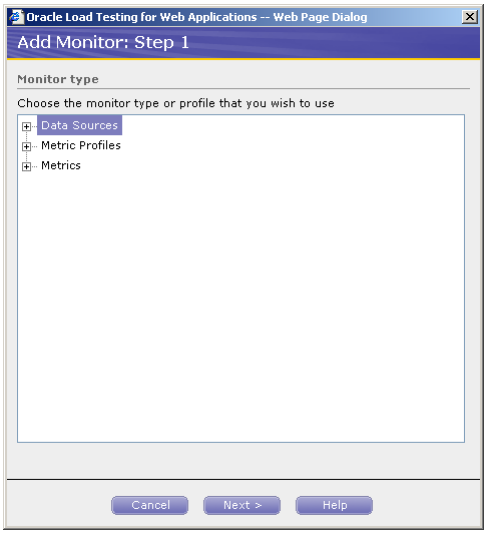
2. Click **New** to add a new configuration.



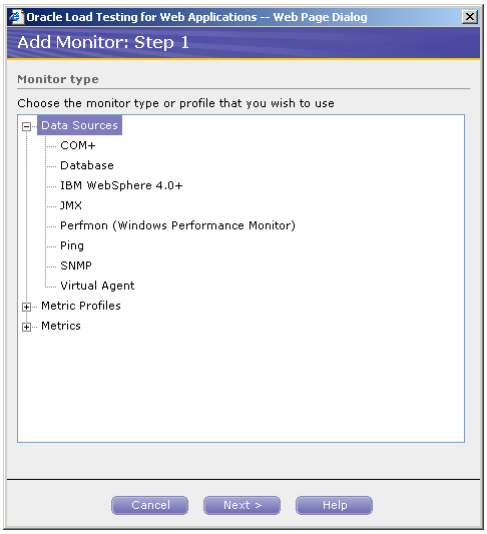
3. Type Tutorial for the Name and the Description.
4. Click **Save**. The Configuration window changes to include the Monitors configuration options.



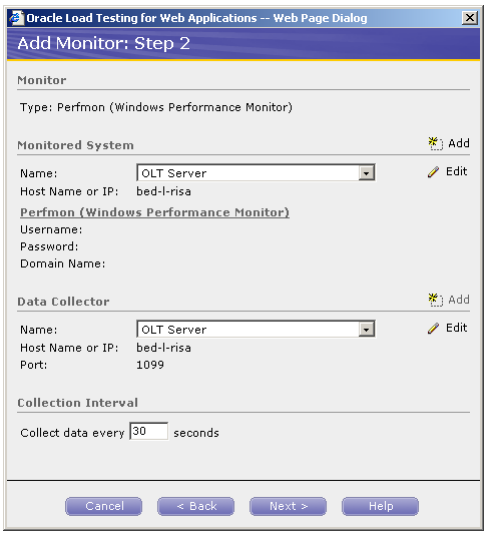
5. Click New in the Monitors pane to open the Add Monitor Step 1 window.



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



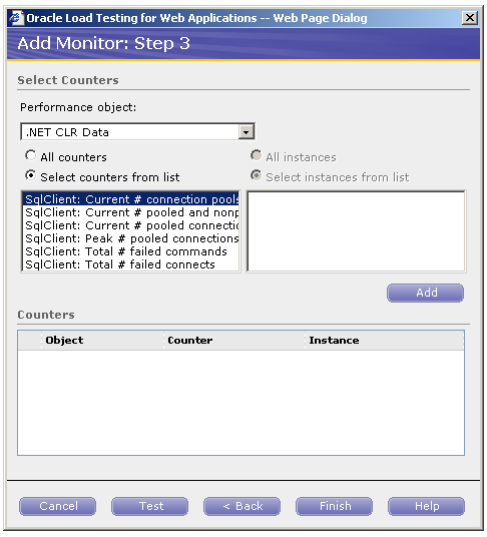
7. Select Perfmon (Windows Performance Monitor) and click **Next**.



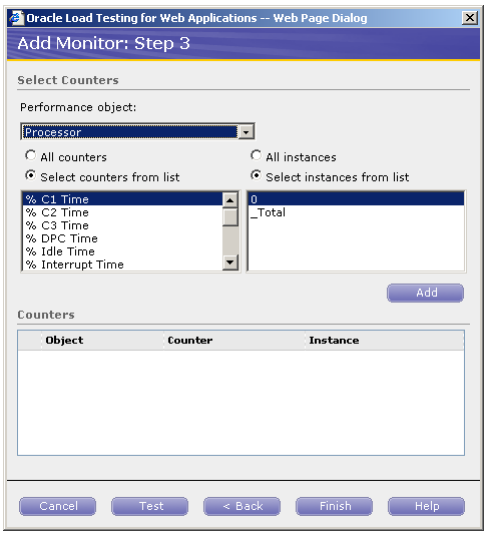
This step lets you specify which to monitor and which system to use for the data collector.

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

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



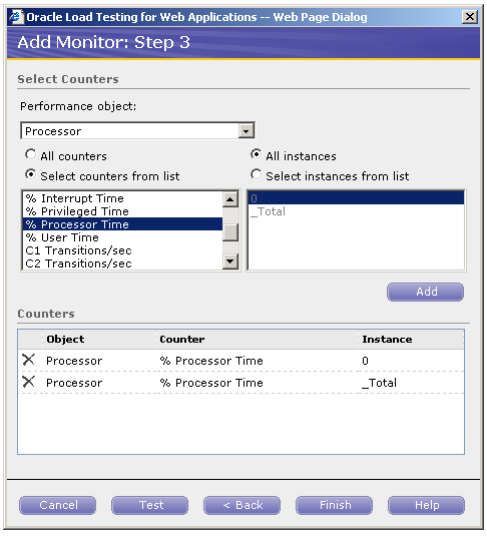
10. Select Processor in the **Performance object** list.



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

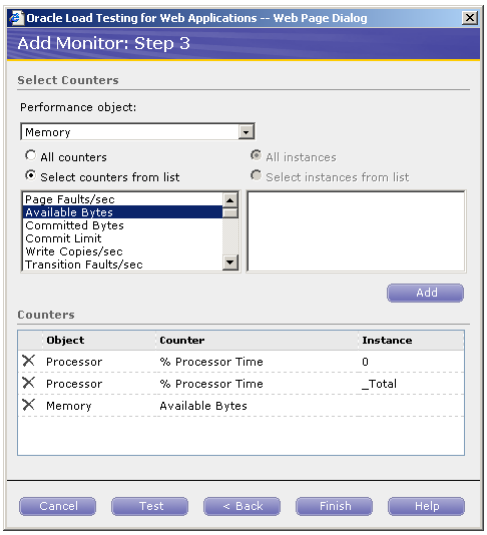
12. Select the **All instance** option.

13. Click **Add.** The counters are added to the **Counters** list.

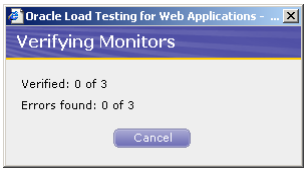


14. Select Memory in the **Performance object** list.

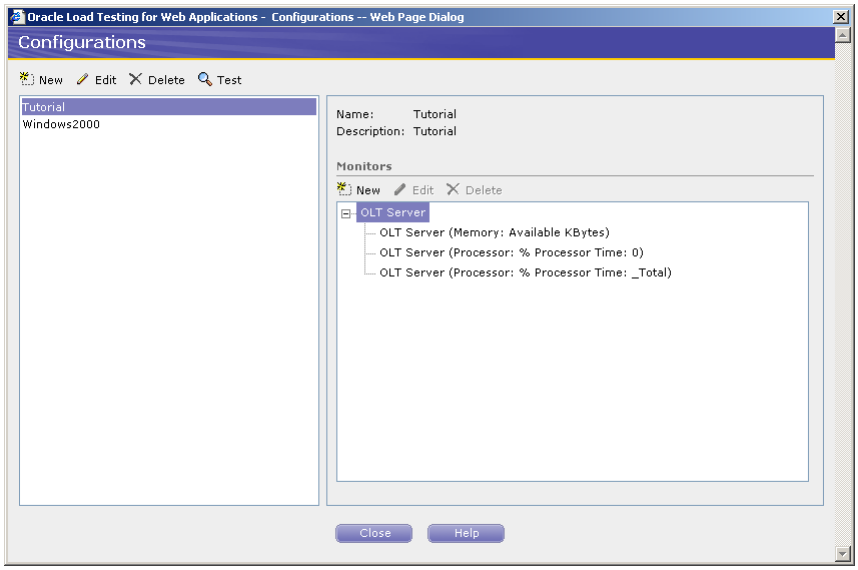
15. Select Available Kbytes and click **Add**.



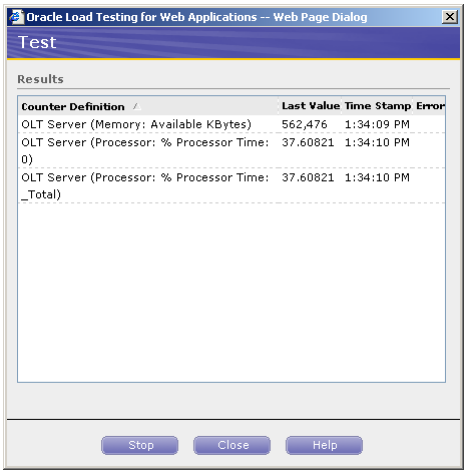
16. Click **Finish** to complete adding monitors. ServerStats display a status while it verifies the counters (this may take a few moments).



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



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



18. Review the results to verify the counters are working properly.
19. Click **Close** to exit the test results.

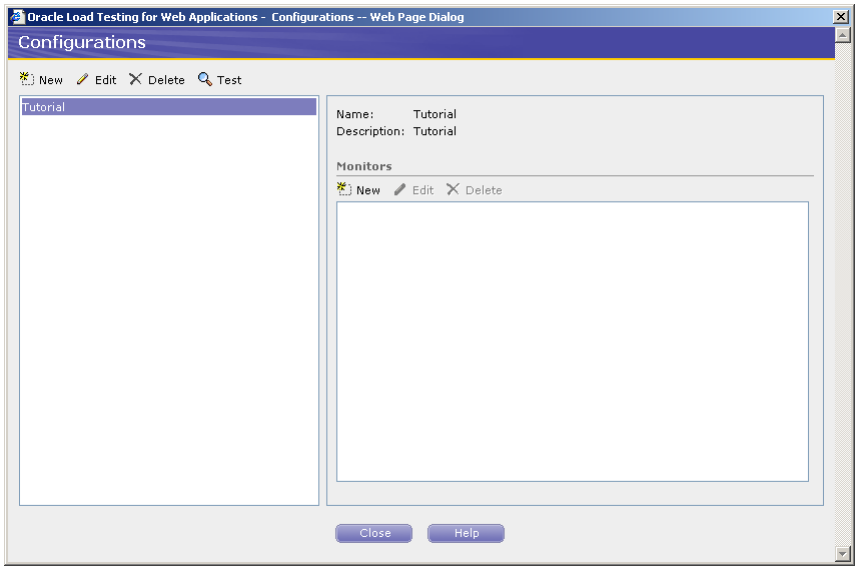
    The next example explains the procedures for editing existing ServerStats configurations.

Example 3

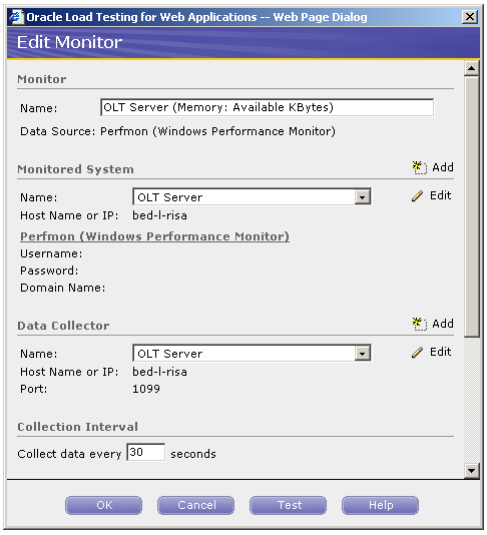
Editing Data Sources

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.

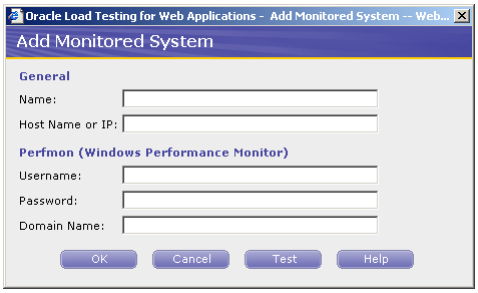
1. If not already open, select **ServerStats→ Configurations** to open the ServerStats Configurations window.



2. Select the Tutorial configuration.
3. Click the Memory (Available Kbytes) monitor and click **Edit.**



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



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 ServerStats.

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.

Adding a Virtual User Profile to the Scenario

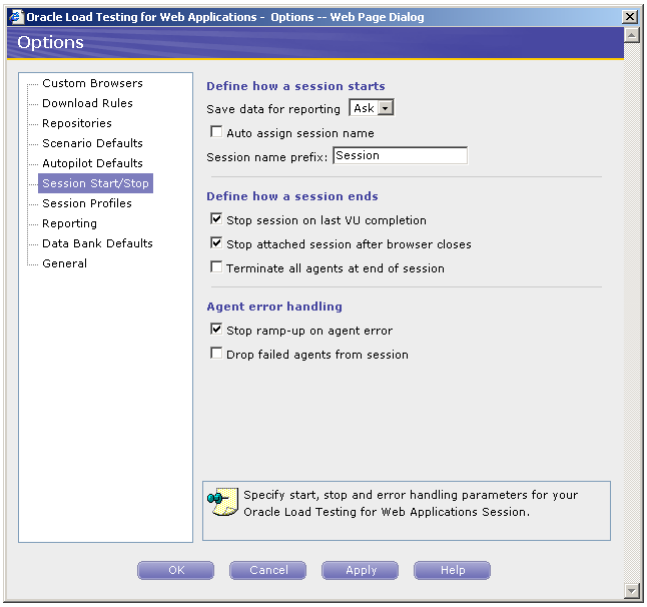
1. Click the **Build Scenarios** tab.
2. Double-click tutor3 in the **Default Profiles** list to add it to the **Configure parameters of the scenario** list.
3.  Click the **Configure all parameters** button on the tutor3 line to display the Edit Scenario Details dialog box.
4. Change the **# VUs** value 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.

Saving Data for Reporting

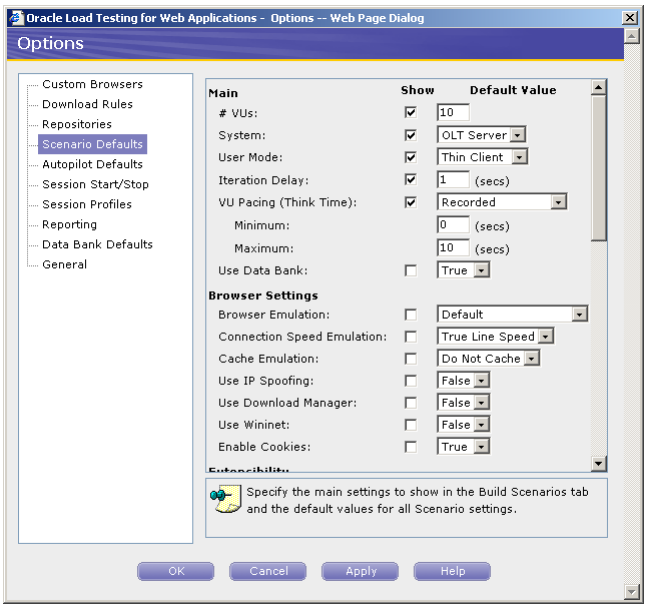
The data generated by a Oracle Load Testing for Web Applications Autopilot session can be saved to the Oracle Load Testing for Web Applications database for post-session analysis. The Session Start/Stop options let you specify if Oracle Load Testing for Web Applications should save the data.

13. Select **Tools → Options → Session Start/Stop**.



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**.



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

Saving the Scenario

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

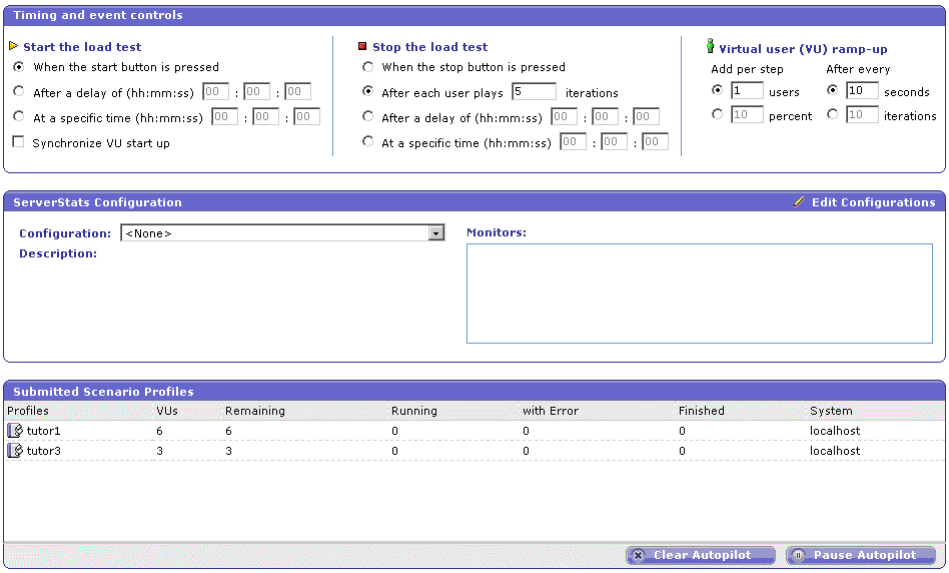
Example 5

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.

Running the Scenario Profiles Using Autopilot

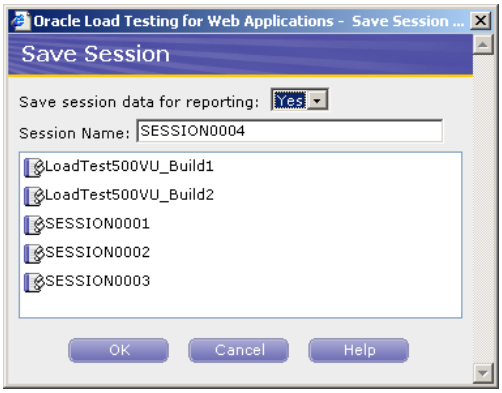
1. Make sure the Scenario from the previous example is still shown in the Build Scenarios tab.
2. Click the **Add to Autopilot** button on the Scenario tab or the toolbar.
3. Oracle Load Testing for Web Applications automatically opens the Set Up Autopilot tab with the tutor1 and tutor3 Scenario Profiles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



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.

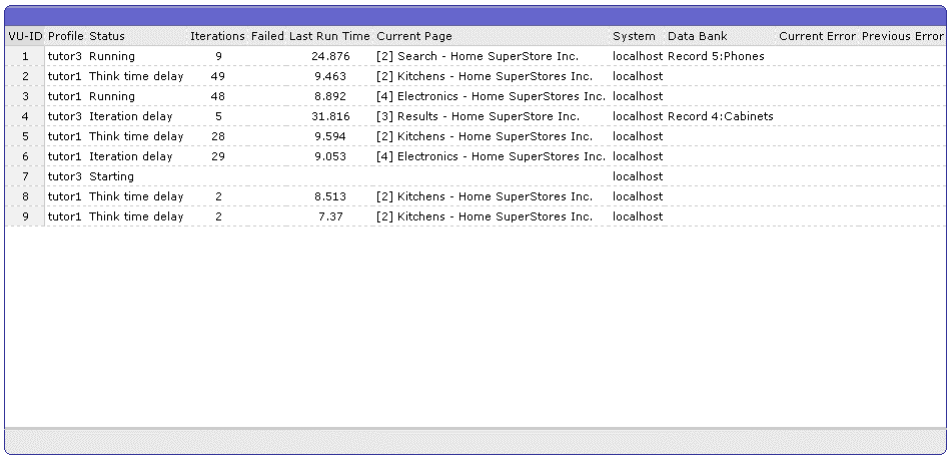
10. Click the **Run test** button on the Oracle Load Testing for Web Applications toolbar.
11. Oracle Load Testing for Web Applications opens the Save Session data dialog box.



12. Click **Ok**.

Note: The above dialog box appears because we used the Ask setting in **Tools → Options → Session Start/Stop**. 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.

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.

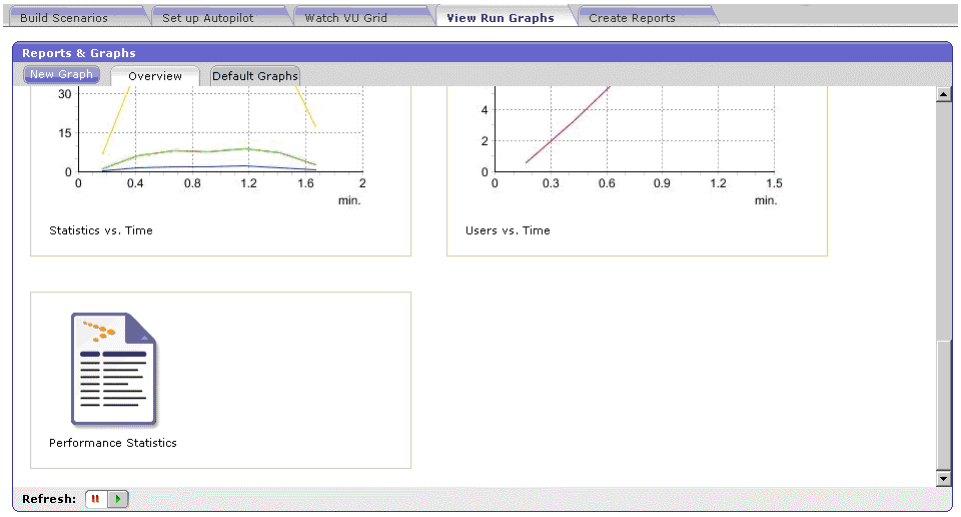


VU-ID	Profile	Status	Iterations	Failed	Last Run Time	Current Page	System	Data Bank	Current Error	Previous Error
1	tutor3	Running	9		24.876	[2] Search - Home SuperStore Inc.	localhost	Record 5:Phones		
2	tutor1	Think time delay	49		9.463	[2] Kitchens - Home SuperStores Inc.	localhost			
3	tutor1	Running	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			

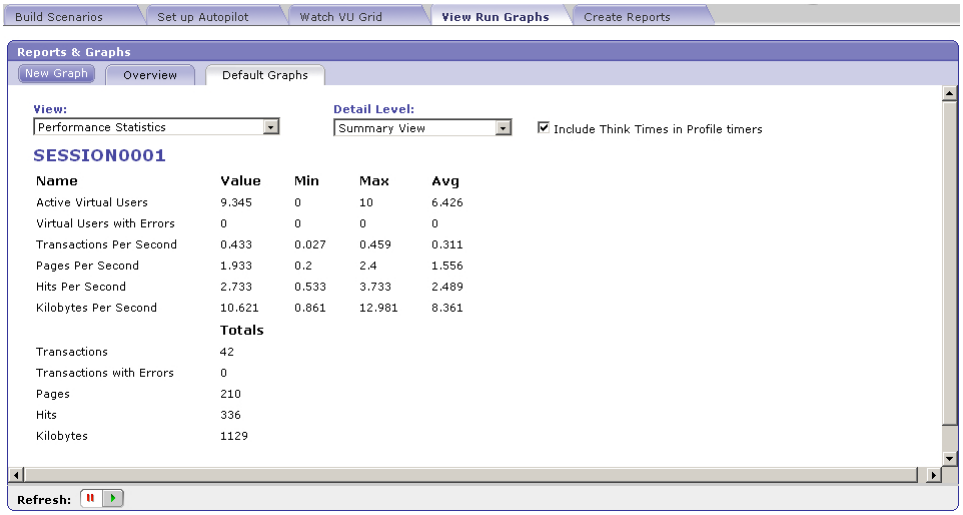
Initially, the Autopilot starts only three virtual users. After the first three have completed five iterations, the Autopilot starts another three virtual users. Once the second three virtual users have completed five iterations, the remaining three virtual users start. The **Virtual User (VU) Rampup**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

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.



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.



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

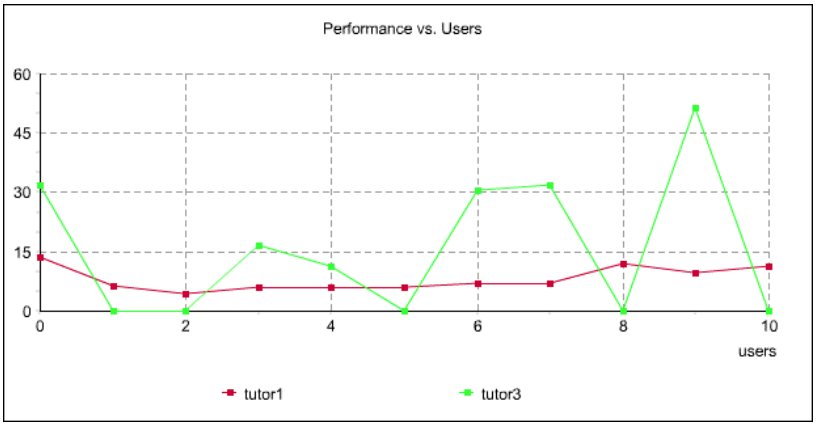
- **<Profile Name> # VUs** – shows the time it took to run the virtual user profile with the indicated number of virtual users running. When ramping up virtual users, Performance by Profile and VUs values are added when additional virtual users start running. Once additional Performance by Profile and 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.

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.

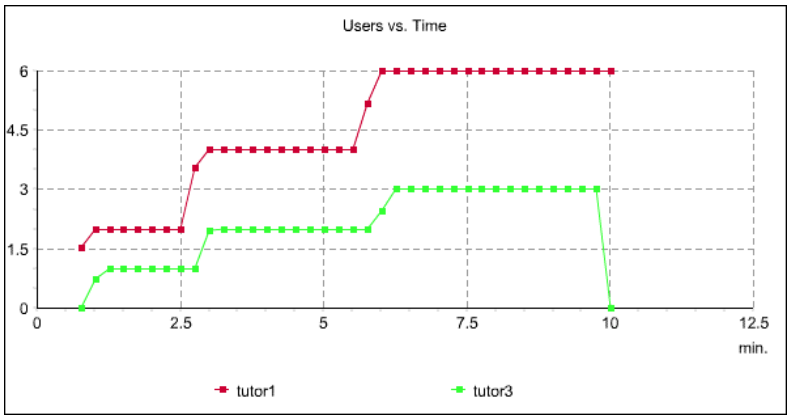


This graph shows the average run time for the number of running virtual users in each profile. The plot points represent the Autopilot rampup of virtual users. In the above graph, the first plot points for each profile shows the average run time while three virtual users were running. Once the Autopilot 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.

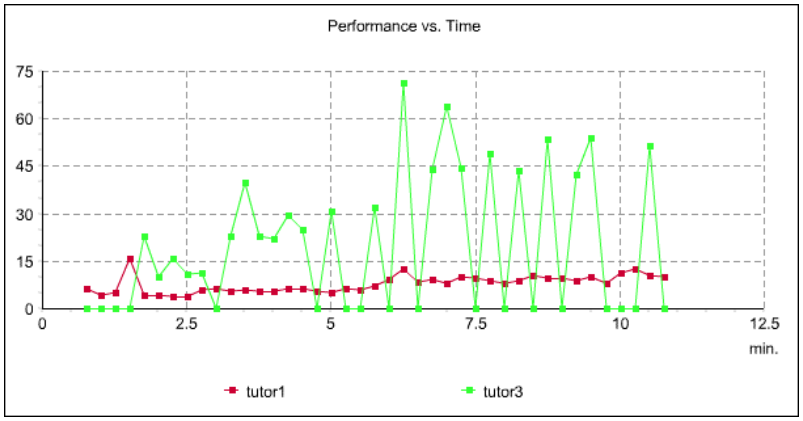
The third plot points show the average run time while nine virtual users are running. In this example, nine virtual users is the total number of virtual users the Autopilot ramps up to run. The third plot points will be updated continuously while the nine virtual users are running.

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



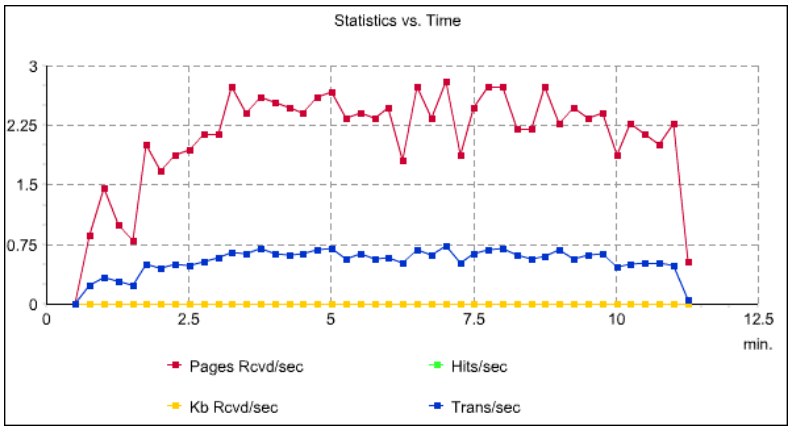
This graph shows the relative time when the virtual users for each profile started running. The graph represents the Autopilot ramp up times and the number of virtual users ramped up for each profile.

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



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.



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.

Example 6

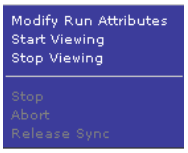
Controlling Virtual Users

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.

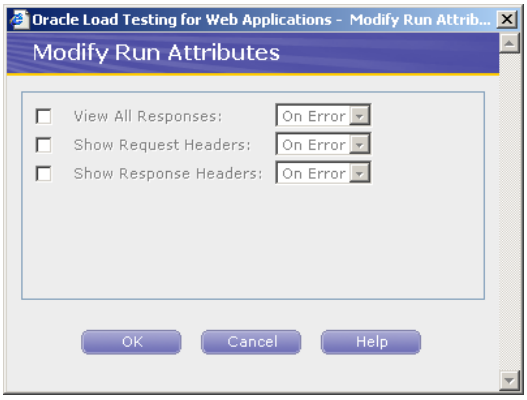
1. Make sure the virtual users from Example 3 are still running.

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.



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.

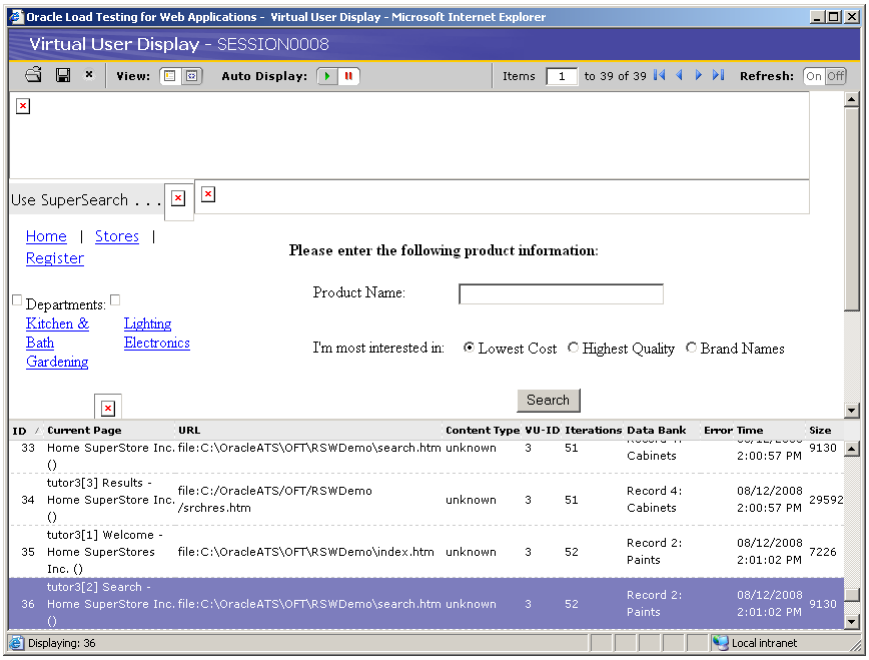


You can change the attributes of each virtual user individually.

5. Select **View All Responses** and select Always from the list.
6. Click **OK** to close the dialog box.

Viewing Virtual User Actions

6. Select **Tools → VU Display** 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.



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.

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.

Aborting an Individual Virtual User

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

Stopping All Virtual Users

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

Aborting All Virtual Users

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

Example 7

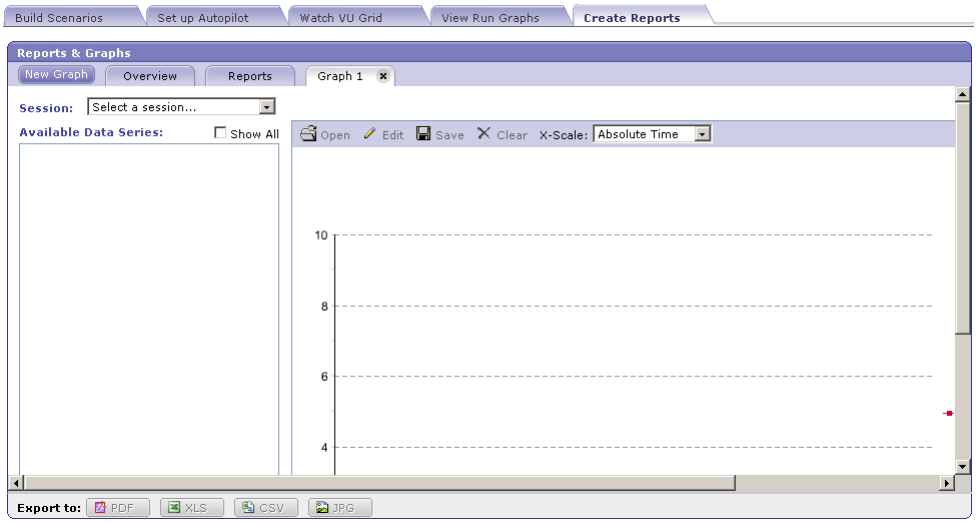
Generating Reports

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.

Generating Reports from Oracle Load Testing for Web Applications

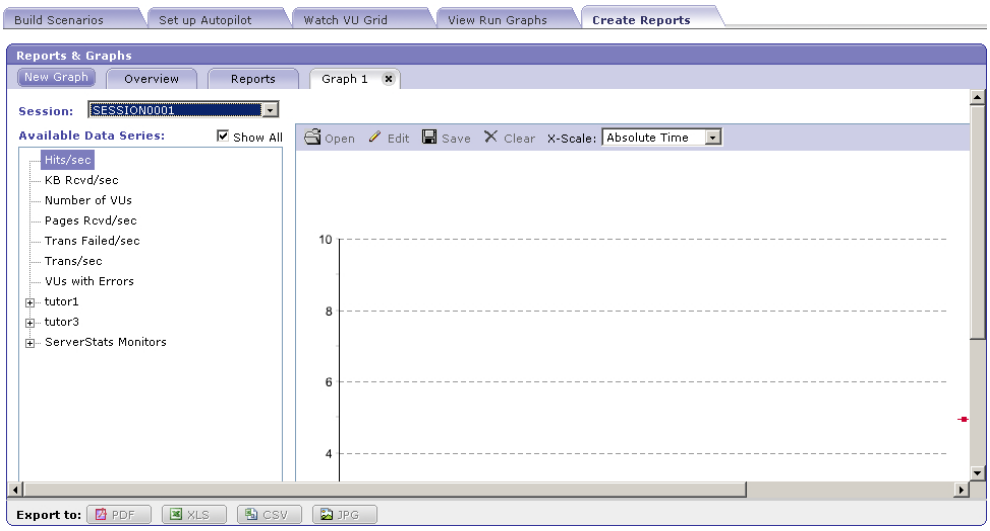
This example shows how to use previously saved data to create custom reports and to view Session and Scenario reports.

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



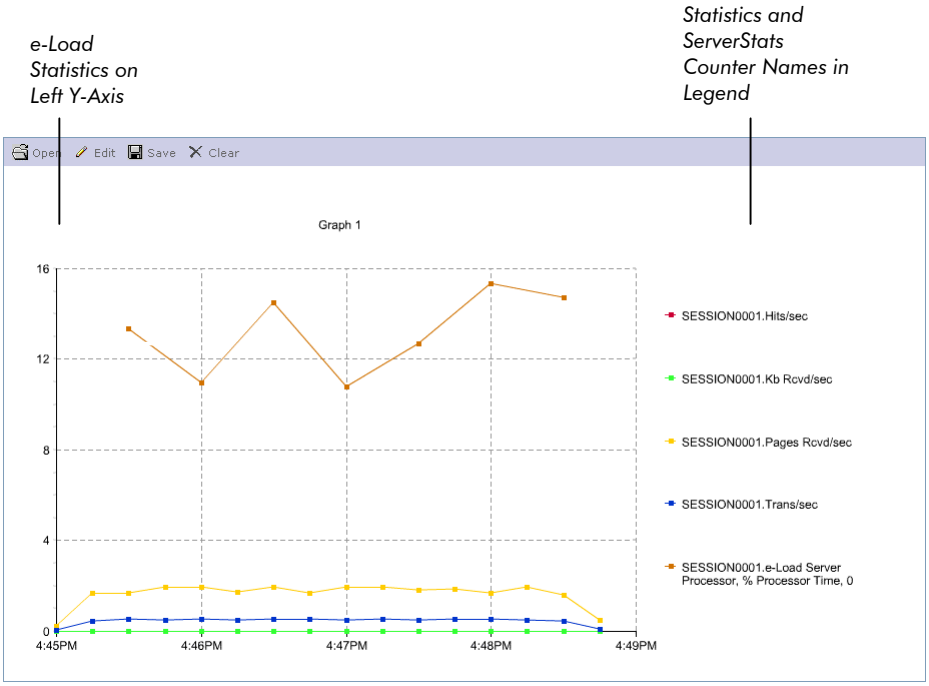
2. The Graph 1 tab is displayed with a blank graph. Select Session0001 from the **Session** list. The data categories appear in the Available Data Series list.

3. Click **Show All**.



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.



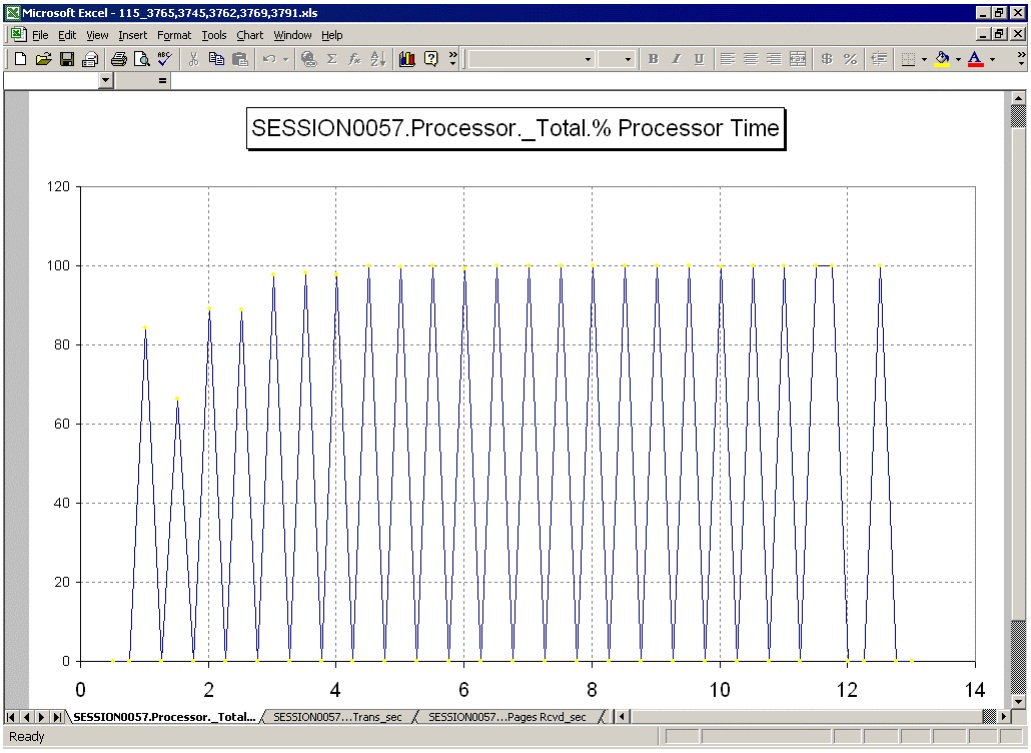
The legends show which color line represents which virtual user profile, Visual Script page, and Oracle Load Testing for Web Applications ServerStats counter. The legends for Oracle Load Testing for Web Applications data show the session, the virtual user profile, and the Visual Scripts page in the form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

Opening the Chart in Microsoft Excel

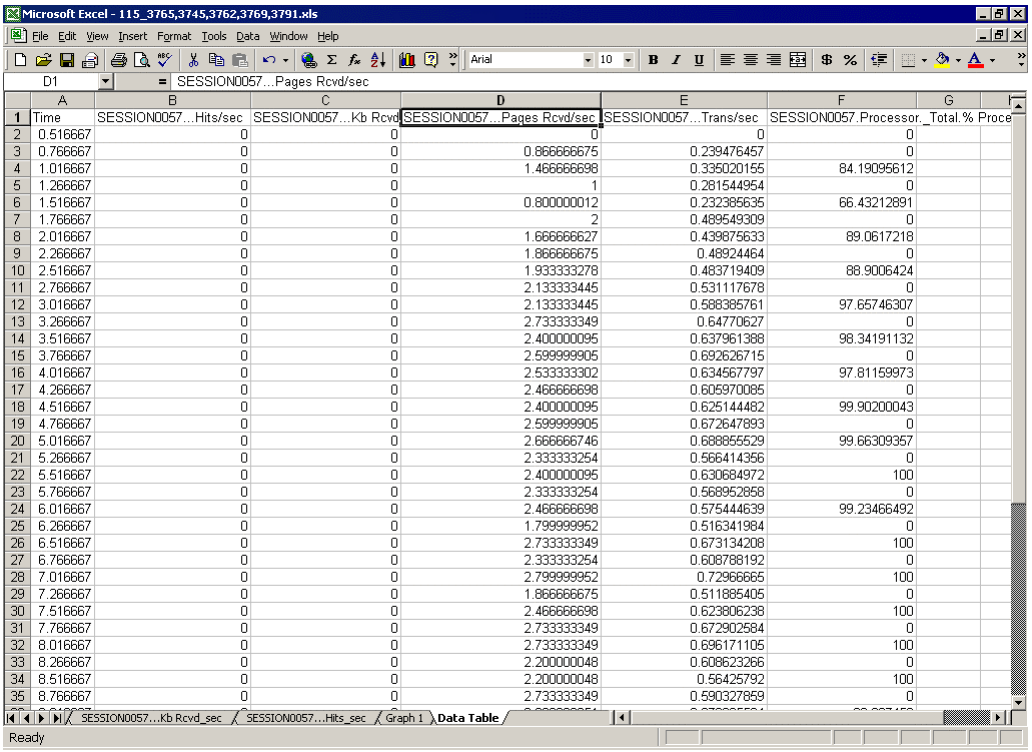
Note: Skip this section if you do not have Microsoft Excel installed on your system.

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



The workbook contains a chart sheet and a worksheet. You can use the features and capabilities of Microsoft Excel to change the chart format.

12. Click the Data Table tab to view the actual data values.



	A	B	C	D	E	F	G
1	Time	SESSION0057...Hits/sec	SESSION0057...Kb Rcvd	SESSION0057...Pages Rcvd/sec	SESSION0057...Trans/sec	SESSION0057.Processor...	_Total.% Proce
2	0.516667	0	0	0	0	0	
3	0.766667	0	0	0.866666675	0.239476457	0	
4	1.016667	0	0	1.466666698	0.335020155	84.19095612	
5	1.266667	0	0	1	0.281544954	0	
6	1.516667	0	0	0.800000012	0.232385635	66.43212891	
7	1.766667	0	0	2	0.489549309	0	
8	2.016667	0	0	1.666666627	0.439875633	89.0617218	
9	2.266667	0	0	1.866666675	0.48924464	0	
10	2.516667	0	0	1.933333278	0.483719409	88.9006424	
11	2.766667	0	0	2.133333445	0.531117678	0	
12	3.016667	0	0	2.133333445	0.588385761	97.65746307	
13	3.266667	0	0	2.733333349	0.64770627	0	
14	3.516667	0	0	2.400000095	0.637961388	98.34191132	
15	3.766667	0	0	2.599999905	0.692626715	0	
16	4.016667	0	0	2.533333302	0.634567797	97.81159973	
17	4.266667	0	0	2.466666698	0.605970085	0	
18	4.516667	0	0	2.400000095	0.625144482	99.90200043	
19	4.766667	0	0	2.599999905	0.672647893	0	
20	5.016667	0	0	2.666666746	0.688855529	99.66309357	
21	5.266667	0	0	2.333333254	0.566414356	0	
22	5.516667	0	0	2.400000095	0.630684972	100	
23	5.766667	0	0	2.333333254	0.568952858	0	
24	6.016667	0	0	2.466666698	0.575444639	99.23466492	
25	6.266667	0	0	1.799999952	0.516341984	0	
26	6.516667	0	0	2.733333349	0.673134208	100	
27	6.766667	0	0	2.333333254	0.608788192	0	
28	7.016667	0	0	2.799999952	0.72966665	100	
29	7.266667	0	0	1.866666675	0.511885405	0	
30	7.516667	0	0	2.466666698	0.623806238	100	
31	7.766667	0	0	2.733333349	0.672902584	0	
32	8.016667	0	0	2.733333349	0.696171105	100	
33	8.266667	0	0	2.200000048	0.608623266	0	
34	8.516667	0	0	2.200000048	0.56425792	100	
35	8.766667	0	0	2.733333349	0.590327859	0	

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

14. Select **File → Exit** to close Microsoft Excel.

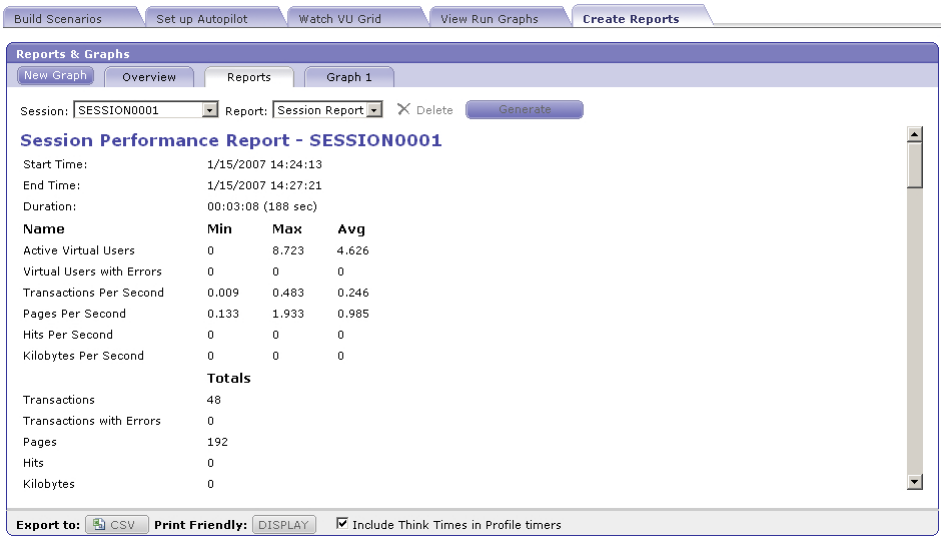
Viewing Scenario and Session Reports

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.

15. 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.

16. Select the **Reports** tab.

17. 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.



Scroll to the end of the Session Performance Report to view the Oracle Load Testing for Web Applications Scenario Report.

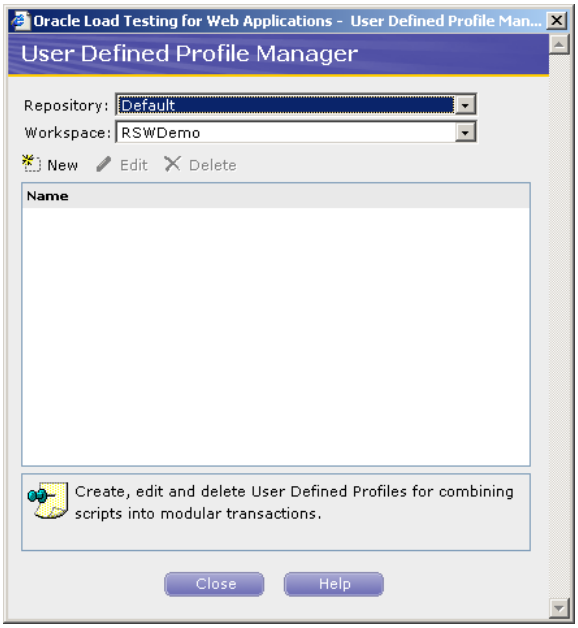
You can print the report by clicking the **Print Friendly** button and selecting **File → Print** from the browser window.

Example 8

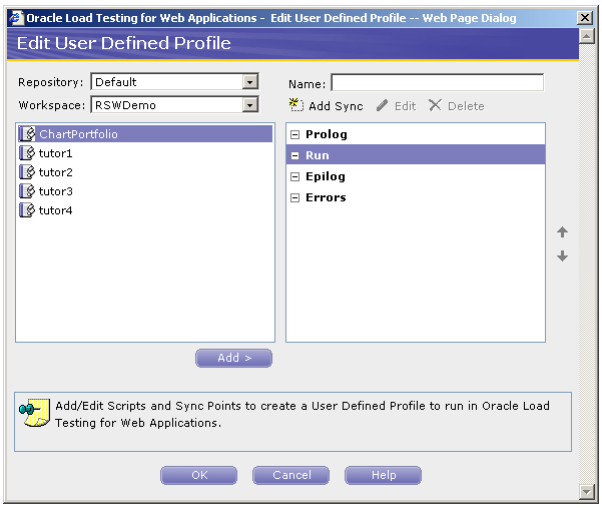
Creating User-Defined Profiles

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

1. Select **Manage → User Defined Profiles**. Oracle Load Testing for Web Applications opens a dialog for managing profiles.



2. Click **New**.



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.

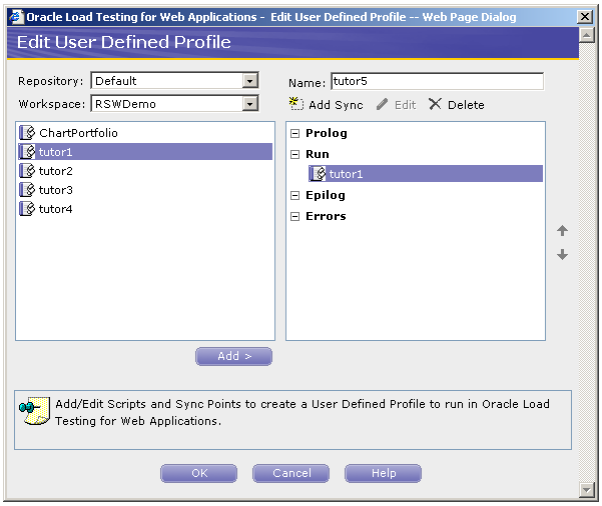
   Epilog – the Visual Scripts in this section play back only once at the end of the Scenario run. An example of what you add to this section is a logoff script.

   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.

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.



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

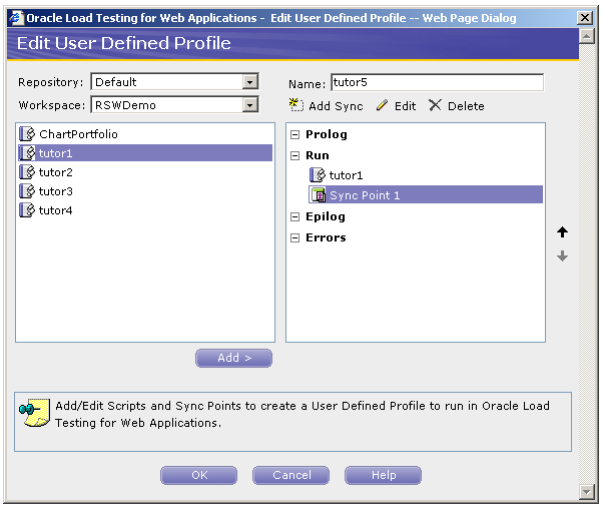
Adding Synchronization Points to the Sections Tree

Note: 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.

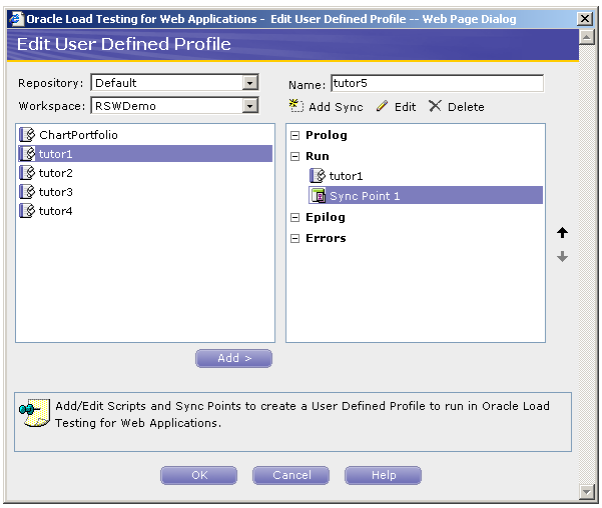


10. Repeat steps 7 and 8 to add additional Sync points to the Sections tree.

Moving Items in the Sections Tree

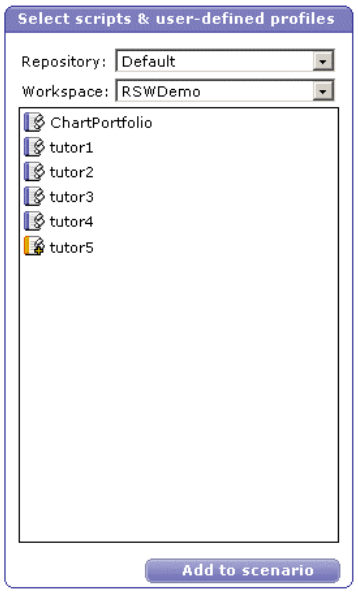
When you have multiple items under any section of the tree, you can move the items up and down within that section.

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



13. Click the **OK** button when you finish defining the profile.

14. The new profile appears in the **Select scripts & user-defined profiles** list of the **Build Scenarios** tab.

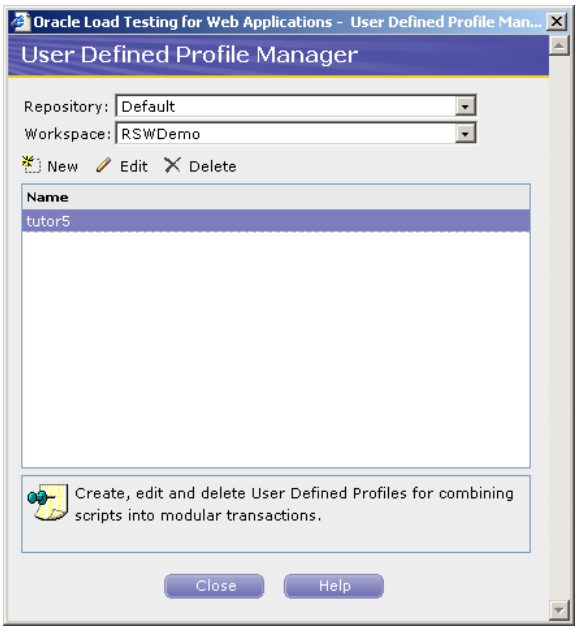


You can include user-defined profiles as part of the Scenario Profiles the same way you use the default profiles.

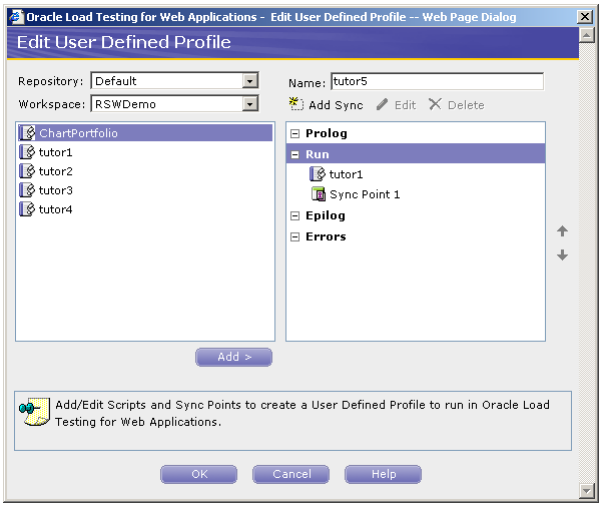
Editing User-Defined Profiles

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

15. Select **Manage → User Defined Profiles**.



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.



17. Click the ⊞ icons 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.

Index