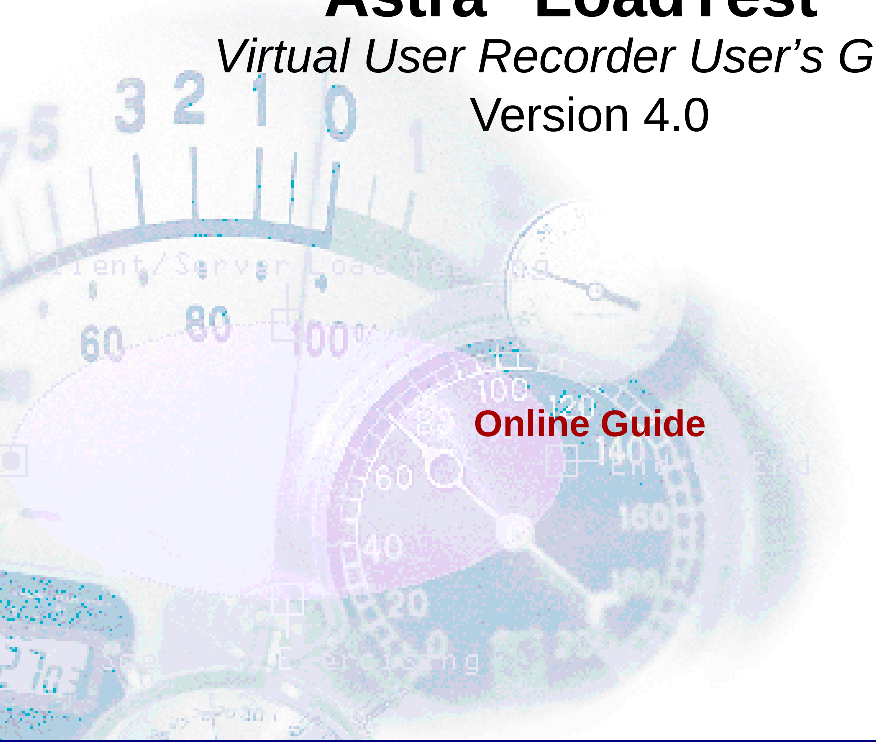
Astra® LoadTest™

Virtual User Recorder User’s Guide

Version 4.0



Online Guide

Find

Find Again

Table of Contents

PART I: STARTING THE TESTING PROCESS

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

PART II: CREATING TESTS

Find
Find Again
◀ ▶
Top of Chapter
Back

Find
Find Again
Top of Chapter
Back

Find
Find Again
◀ ▶
Top of Chapter
Back

PART III: RUNNING AND DEBUGGING TESTS

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

PART IV: ADVANCED FEATURES

Find
Find Again
Top of Chapter
Back

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

PART V: CONFIGURING THE VIRTUAL USER RECORDER

Find
Find Again
Top of Chapter
Back

PART VI: WORKING WITH TESTDIRECTOR

Welcome to Astra LoadTest, Mercury Interactive’s load testing tool for Web sites. Astra LoadTest provides everything you need to quickly create and run tests.

Using This Guide

This guide describes how to use Astra LoadTest to test your Web site. It provides step-by-step instructions to help you create, debug and run tests.

This guide contains 6 parts:

Part I: Starting the Testing Process

Provides an overview of Astra LoadTest and the main stages of the testing process.

Part II: Creating Tests

Describes how to create tests, insert checkpoints, assign parameters, set run-time settings, use regular expressions, actions, and handle unexpected events that occur during a test run.

Find
Find Again
◀ ▶
Top of Chapter
Back

Part III: Running and Debugging Tests

Describes how to run tests and analyze test results, and how to control test runs to identify and isolate bugs in test scripts.

Part IV: Advanced Features

Describes how to enhance your test in Expert View mode and introduces several programming techniques to create a more powerful test. It also describes how to streamline the testing process of your Web sites. **This section is recommended for advanced users of Astra LoadTest**.

Part V: Configuring the Virtual User Recorder

Describes how to change Astra LoadTest’s default settings, both globally and per test. It also describes how to customize the test script editor.

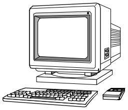
Part VI: Working with TestDirector

Describes how Astra LoadTest interacts with TestDirector, Mercury Interactive’s test management tool.

Find
Find Again
◀ ▶
Top of Chapter
Back

Online Resources

Astra LoadTest includes the following online resources:



Read Me First provides last-minute news and information about Astra LoadTest.

Astra LoadTest Tutorial teaches you basic Astra LoadTest skills and shows you how to start load testing your Web site.

Astra LoadTest QuickTour provides a basic overview of load testing with Astra LoadTest.

Mercury Tours sample Web site is the basis for many examples in this book. The URL for this Web site is *http://astra.mercuryinteractive.com/mercurytours*.

Astra LoadTest Context Sensitive Help describes dialog boxes and toolbar buttons, and provides procedural information.

Astra Function Reference gives you online access to the Astra VBScript functions, including a description of each object, a list of the functions (methods) associated with each object, and description syntax and usage of each function (method).

VBScript Reference describes Microsoft's VBScript language, and includes a tutorial and function reference.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Technical Support Online uses your default Web browser to open Mercury Interactive’s Customer Support Web site. The URL for this Web site is *http://support.mercuryinteractive.com*.

Support Information presents the locations of Mercury Interactive’s Customer Support Web site and home page, the e-mail address for sending information requests, the name of the relevant news group, the location of Mercury Interactive’s public FTP site, and a list of Mercury Interactive’s offices around the world.

Mercury Interactive on the Web uses your default Web browser to open Mercury Interactive’s home page. This site provides you with the most up-to-date information on Mercury Interactive and its products. This includes new software releases, seminars and trade shows, customer support, educational services, and more. The URL for this Web site is *http://www.mercuryinteractive.com*.

Find
Find Again
◀ ▶
Top of Chapter
Back

Typographical Conventions

This book uses the following typographical conventions:

1, 2, 3	Bold numbers indicate steps in a procedure.
●	Bullets indicate options and features.
>	The greater than sign separates menu levels (for example, **File > Open**).
Bold	**Bold** text indicates function names.
Italics	*Italic* text indicates variable names.
Helvetica	The Helvetica font is used for examples and statements that are to be typed in literally.
[ ]	Square brackets enclose optional parameters.
{ }	Curly brackets indicate that one of the enclosed values must be assigned to the current parameter.
...	In a line of syntax, an ellipsis indicates that more items of the same format may be included. In a program example, an ellipsis is used to indicate lines of a program that were intentionally omitted.
|	A vertical bar indicates that either of the two options separated by the bar should be selected.

Find
Find Again
Top of Chapter
Back

Starting the Testing Process

Find
Find Again
◀ ▶
Top of Chapter
Back

Welcome to Astra LoadTest, Mercury Interactive’s load testing tool for Web sites. This guide provides you with detailed descriptions of Astra LoadTest features and testing procedures.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Testing with Astra LoadTest

Astra LoadTest facilitates creating tests on your Web site by recording as you navigate. You record your test with the Virtual User Recorder. As you navigate through your site, the Virtual User Recorder records each step you perform and generates a test that graphically displays this step in an icon-based *test tree*. For example, clicking a link, selecting a check box, or submitting a form are all recorded in your test.

In addition, you can instruct Astra LoadTest to check the response of your site to specific Web objects, text strings, or tables. For example, you can instruct Astra LoadTest to check that a specific text string appears in a particular location on your Web page, or you can check that a hypertext link goes to the correct URL address.

After you record, you can further enhance your test by adding and modifying steps in the test tree. When you run the test, Astra LoadTest connects to your site and performs each step in your test. After you run your test, you can view a report detailing which steps in your test succeeded or failed.

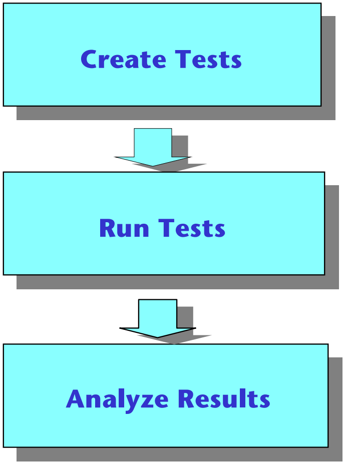
Note that by default, each test includes a single action, however a test can include multiple actions. Most of the chapters in this guide provide information on how to work with a single action. For information on why and how to work with multiple actions in a test, see Chapter 11, **Working with Actions**.

Once you test the validity of your test in the Virtual User Recorder, you incorporate it in a load testing scenario. You use the Astra LoadTest Controller to run load tests and analyze your Web application’s performance under load. Refer to the *Astra LoadTest Controller User’s Guide* for information about load testing scenarios.

Find
Find Again
Top of Chapter
Back

Astra LoadTest Testing Process

Testing with Astra LoadTest involves 3 main stages:



Create Tests

Run Tests

Analyze Results

Find
Find Again
Top of Chapter
Back

Creating Tests

You create a test by recording a Web session with the Virtual User Recorder. The test is used to load test your application.

To create a test:

- Record a session on your site.

  As you navigate through your site, the Virtual User Recorder graphically displays each *step* you perform in the form of a collapsible icon-based *test tree*. A step changes the content of a Web page in your site, for example, clicking a link or image, or submitting a data form. For more information, see Chapter 3, **Creating Tests**.

- Insert checkpoints into your test.

  A *checkpoint* searches for a specific value of a page, object or text string on a Web page and enables you to identify whether or not your Web site is functioning correctly. You can check a Web page for objects, text strings, and tables. For more information, see Chapter 4, **Creating Checkpoints**.

- Insert load testing elements into your test.

  You define *transactions* to mark the business processes that Astra LoadTest should measure. When you record a test, Astra LoadTest automatically marks each step you perform as a transaction. This means that when you run a load testing scenario, each step in your test tree is recognized as a transaction to be measured.

  You insert *rendezvous points* into a test to emulate heavy user load on the server. Rendezvous points allow you to select specific steps in your test and regulate the load they are subjected to. They are used to check high usage areas such as log in screens or form submissions. For more information, see Chapter 6, **Testing Load**.

Find
Find Again
Top of Chapter
Back

- Broaden the scope of your test by replacing fixed values with parameters.

When you test your site, you can *parameterize* your test to check how your Web site performs the same operations with multiple sets of data. The data is stored in a table in the Data pane. When you parameterize your test, Astra LoadTest substitutes the parameters in your test with values from the table. During each *iteration* of your test, Astra LoadTest changes the values in the parameterized statements. For more information, see Chapter 7, **Parameterizing Tests**.

You can also use output parameters to parameterize your test. An *output parameter* is a value retrieved from a parameter in your test during the test run, and entered into your table in the Data pane. You can subsequently use this output parameter as an input variable in your test. This enables you to use data retrieved during a test in other parts of the test. For more information, see Chapter 8, **Creating Output Parameters**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Running Tests

After you create your test, you run it using the Virtual User Recorder to debug it before you incorporate it into a load testing scenario.

You can:

- Run your test to check your site.

The test runs from the first line in your test and stops at the end of the test. While running, Astra LoadTest connects to your Web site and performs each operation in your test, checking any text strings, objects or tables you specified. If you parameterized your test, Astra LoadTest repeats the test for each set of data values you defined. For more information, see Chapter 14, **Running Tests in Stand-Alone Mode**.

- Run a test to debug your test.

  You can control your test run to help you identify and eliminate defects in your test. You can use the *Step* commands to run your test step by step. You can also set *breakpoints* to pause your test at pre-determined points. For more information, see Chapter 16, **Debugging Tests**.

Analyzing Test Results

After you run your test, you can view the test results.

You can:

- View the test results in the Test Results window.

  After you run your test, the Test Results window opens and displays the results of your test. You can view a summary of your test results or a detailed report. For more information, see **Viewing the Results of a Test Run** on page 252.

- Report defects detected during a test run.

  If a test run detects a defect in you site, you can report it using the Web Defect Manager, Mercury Interactive's system for reporting and tracking defects and errors over the World Wide Web.

  The Web Defect Manager is a scalable, cross-platform defect tracking system that helps you monitor defects closely from initial detection until resolution. The Web Defect Manager is tightly integrated with TestDirector, Mercury Interactive's test management tool.

  For more information on the Web Defect Manager, see Chapter 27, **Reporting Defects**. For more information on TestDirector , see Chapter 26, **Managing the Testing Process**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Expert View

You can use the Expert View tab to view a text-based version of your test. The test script is composed of VBScript statements (Microsoft's Visual Basic Scripting language) that correspond to the steps and checks displayed in your test tree. For more information, see Chapter 20, **Testing in the Expert View**.

Actions

You can divide your test into sections called "actions." This enables you to parameterize only part of your test. It also enables you to use an action in multiple tests by importing it from one test into another test. For more information on parameterizing tests, see Chapter 7, **Parameterizing Tests**. For more information on working with actions, see Chapter 11, **Working with Actions**.

Sample Site

Many examples in this guide use the Mercury Tours sample Web site. The URL for this Web site is *http://astra.mercuryinteractive.com/mercurytours*.

The first page of the Mercury Tours site is the login page. You must log in to begin using the site. To log in, enter "mercury" as your member name and "mercury" as your password.

Find
Find Again
◀ ▶
Top of Chapter
Back

Managing the Testing Process

Astra LoadTest works with TestDirector, Mercury Interactive's test management tool. You can use TestDirector to create a project (central repository) of manual and automated tests, build test cycles, run tests, and report and track defects. You can also create reports and graphs to help you review the progress of test planning, test runs, and defect tracking before a software release.

When you work in Astra LoadTest, you can create and save tests directly to your TestDirector project. You can also run Astra LoadTest tests from TestDirector and then use TestDirector to review and manage the results. For more information, see Chapter 26, **Managing the Testing Process**.

Find
Find Again
◀ ▶
Top of Chapter
Back

This chapter explains how to start the Virtual User Recorder and introduces the Virtual User Recorder window.

This chapter describes:

Find
Find Again
Top of Chapter
Back

Starting the Virtual User Recorder

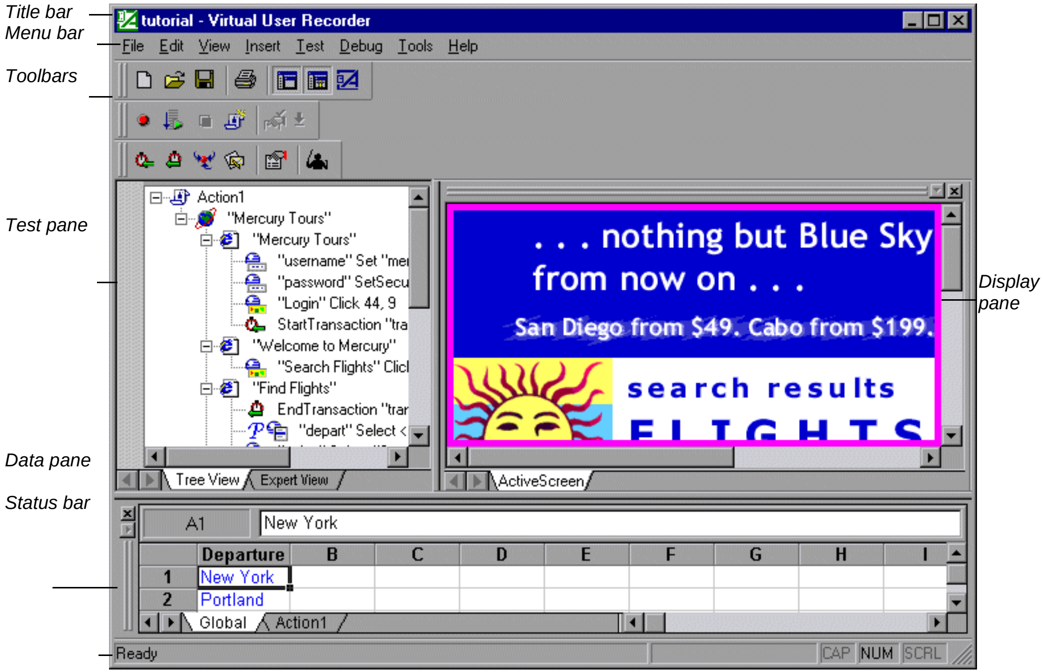
To start the Virtual User Recorder, click **Programs** > **Astra LoadTest** > **Products > Virtual User Recorder** in the **Start** menu. After several seconds, the Virtual User Recorder window is displayed on your desktop.

The Virtual User Recorder Window

The Virtual User Recorder window contains the following key elements:

- *Title bar,* displaying the name of the currently open test
- *Menu bar,* displaying menus of the Virtual User Recorder commands
- *File toolbar,* containing buttons to assist you in managing your test
- *Main toolbar,* containing buttons to assist you in the testing process
- *Debug toolbar,* containing buttons to assist you in debugging your test
- *Test pane*, containing two tabs to view your test—Tree View and Expert View
- *Display pane*, containing three tabs to assist you in the testing process—ActiveScreen, Defect Report, and Log
- *Data pane*, containing two tabs to assist you in parameterizing your test—Global and Action
- *Status bar*, displaying the status of the open test

Find
Find Again
◀ ▶
Top of Chapter
Back



```
Action1
  "Mercury Tours"
    "Mercury Tours"
      "username" Set "me
      "password" SetSecu
      "Login" Click 44, 9
      StartTransaction "tra
    "Welcome to Mercury"
      "Search Flights" Clicl
    "Find Flights"
      EndTransaction "trar
      "depart" Select <
```

. . . nothing but Blue Sky from now on . . .

San Diego from $49. Cabo from $199.

search results

FLIGHTS

A1 New York

	Departure	B	C	D	E	F	G	H	I
1	New York								
2	Portland								

Find
Find Again
Top of Chapter
Back

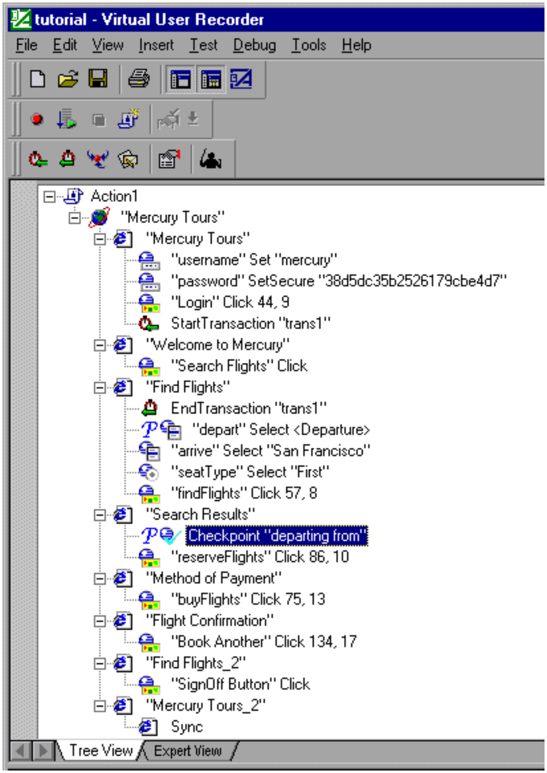
Test Pane

The Test pane contains two tabs to view your test—Tree View and Expert View.

Tree View Tab

In the Tree View tab (default mode), the Virtual User Recorder displays your test in the form of a collapsible icon-based *test tree*. Each operation performed on your Web application is recorded as an icon in your test tree. For every icon in the Tree View, the Virtual User Recorder displays a corresponding line of script in the Expert View.

Find
Find Again
◀ ▶
Top of Chapter
Back


tutorial - Virtual User Recorder
File Edit View Insert Test Debug Tools Help
Tree View
Expert View

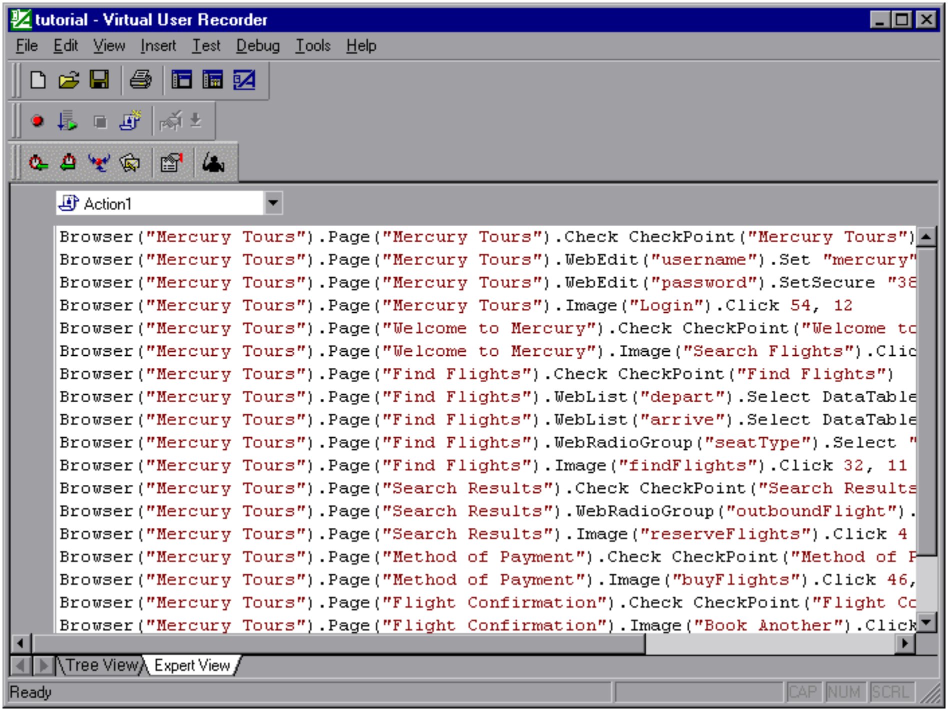
```
Action1
  "Mercury Tours"
    "Mercury Tours"
      "username" Set "mercury"
      "password" SetSecure "38d5dc35b2526179cbe4d7"
      "Login" Click 44, 9
      StartTransaction "trans1"
    "Welcome to Mercury"
      "Search Flights" Click
    "Find Flights"
      EndTransaction "trans1"
      "depart" Select <Departure>
      "arrive" Select "San Francisco"
      "seatType" Select "First"
      "findFlights" Click 57, 8
    "Search Results"
      Checkpoint "departing from"
      "reserveFlights" Click 86, 10
    "Method of Payment"
      "buyFlights" Click 75, 13
    "Flight Confirmation"
      "Book Another" Click 134, 17
    "Find Flights_2"
      "SignOff Button" Click
    "Mercury Tours_2"
      Sync
```

Find
Find Again
Top of Chapter
Back

Expert View Tab

In the Expert View tab, the Virtual User Recorder displays your test in the form of a test script instead of a test tree. Your test script is composed of VBScript statements. For every statement in the Expert View tab, a corresponding icon exists in the test tree in the Tree View tab. For more information on using the Expert View, see Chapter 20, **Testing in the Expert View**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back



tutorial - Virtual User Recorder
File Edit View Insert Test Debug Tools Help
Action1

```
Browser("Mercury Tours").Page("Mercury Tours").Check CheckPoint("Mercury Tours")
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Set "mercury"
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("password").SetSecure "38
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Click 54, 12
Browser("Mercury Tours").Page("Welcome to Mercury").Check CheckPoint("Welcome to
Browser("Mercury Tours").Page("Welcome to Mercury").Image("Search Flights").Clic
Browser("Mercury Tours").Page("Find Flights").Check CheckPoint("Find Flights")
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select DataTable
Browser("Mercury Tours").Page("Find Flights").WebList("arrive").Select DataTable
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seatType").Select "
Browser("Mercury Tours").Page("Find Flights").Image("findFlights").Click 32, 11
Browser("Mercury Tours").Page("Search Results").Check CheckPoint("Search Results
Browser("Mercury Tours").Page("Search Results").WebRadioGroup("outboundFlight").
Browser("Mercury Tours").Page("Search Results").Image("reserveFlights").Click 4
Browser("Mercury Tours").Page("Method of Payment").Check CheckPoint("Method of P
Browser("Mercury Tours").Page("Method of Payment").Image("buyFlights").Click 46,
Browser("Mercury Tours").Page("Flight Confirmation").Check CheckPoint("Flight Co
Browser("Mercury Tours").Page("Flight Confirmation").Image("Book Another").Click
```

Find
Find Again
Top of Chapter
Back

Display Pane

ActiveScreen Tab

The ActiveScreen tab displays the Web page or object corresponding to a highlighted step in your test. It provides you with an easy way to view your test, make modifications, and add checkpoints.

Data Pane

The Data pane contains two tabs to assist you in parameterizing your test—Global and Action. To view this pane, click the **Data Views** button or choose **View > Data Views**.

Global Tab

The Global tab contains variable values for the parameters defined in your parameterized test. The variable values are available for an entire test. When you run your parameterized test, Astra LoadTest inserts the data from the Global tab into the test. For more information, see Chapter 11, **Working with Actions**.

Action Tab

The Action tab contains variable values for the parameters defined in your parameterized test. The variable values are available only for a specific action and not for an entire test. When you run your parameterized test, Astra LoadTest inserts the data from the Action tab into the relevant action in the test. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Using Virtual User Recorder Commands

You can select Virtual User Recorder commands from the menu bar or from a toolbar. Certain Virtual User Recorder commands can also be executed by pressing shortcut keys.

Choosing Commands on a Menu

You can choose all Virtual User Recorder commands from the menu bar.

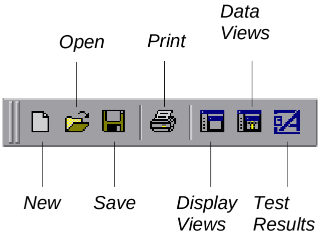
Clicking Commands on a Toolbar

You can execute some Virtual User Recorder commands by clicking buttons on the toolbars. The Virtual User Recorder has three built-in toolbars: the *File toolbar*, the *Main toolbar*, and the *Debug toolbar*.

Find
Find Again
Top of Chapter
Back

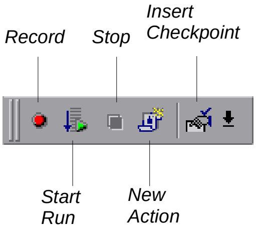
File Toolbar

The File toolbar contains buttons for managing a test. For more information on managing your test, see Chapter 3, **Creating Tests**. The following buttons appear on the File toolbar:



Main Toolbar

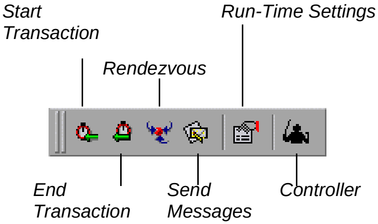
The Main toolbar contains buttons for the commands used when creating and maintaining your test. The following buttons appear on the Main toolbar:



Find
Find Again
Top of Chapter
Back

Load Toolbar

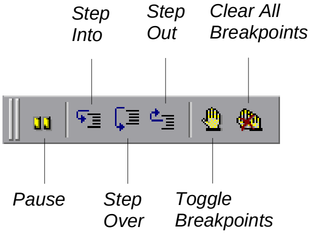
The Load toolbar contains buttons for the commands used when inserting transactions and rendezvous points into your test. You can also invoke the Controller when you are ready to run your scenario. The following buttons appear on the Load toolbar:



Find
Find Again
Top of Chapter
Back

Debug Toolbar

The Debug toolbar contains buttons for the commands used when debugging the steps in your test. The following buttons appear on the Debug toolbar:



Note: The debug toolbar is automatically enabled when you run your test in the Virtual User Recorder.

Executing Commands Using Shortcut Keys

You can execute some Virtual User Recorder commands by pressing shortcut keys. The following shortcut keys appear on the corresponding menu commands:

Command	Shortcut Key	Function
New	Ctrl + N	Creates a new test and closes your browser.
Open	Ctrl + O	Opens a test.
Save	Ctrl + S	Saves the active test.
Print	Ctrl + P	Prints the active test.
Function Arguments	Alt + Enter	Opens the Method tab in the Parameterization/Properties dialog box.
Object Properties	Ctrl + Enter	Opens the Object tab in the Parameterization/Properties dialog box.
Cut	Ctrl + X	Removes the selection from your test.

Find
Find Again
Top of Chapter
Back

Command	Shortcut Key	Function
Copy	Ctrl + C	Copies the selection from your test.
Paste	Ctrl + V	Pastes the selection to your test.
Delete	Del	Deletes the selection from your test.
Rename	F2	Changes the name of an action or a step (Tree View only).
Parameter Info	Ctrl + Shift + Space	Displays the syntax of a parameter (Expert View only).
Checkpoint	F12	Creates a checkpoint for a text string, an object, or a table.
Output Parameter	Ctrl + F12	Creates an output parameter for a text string, an object, or a table.
Run	F5	Runs the test from the beginning.
Stop	F4	Stops test recording or the test run.
Pause	PAUSE	Stops the test run after the statement has been executed. The test run can be resumed from this point.

Find
Find Again
Top of Chapter
Back

Command	Shortcut Key	Function
Step Into	F11	Runs only the current line of the test script, however, if the current line calls a test or function, the called test or function is displayed in the view but is not executed.
Step Over	F10	Runs only the current line of the test script. When the current line calls another test or a function, the called test or function is executed in its entirety but is not displayed in the view.
Step Out	Shift + F11	Runs to the end of the called test or function, returns to the calling test, and then pauses execution. (Available only after entering a test or function using Step Into.)
Toggle Breakpoint	F9	Sets or clears a breakpoint in the test.
Clear All Breakpoints	Ctrl + Shift + F9	Deletes all breakpoints in the test.

Find
Find Again
◀ ▶
Top of Chapter
Back

Creating Tests

Find

Find Again

Top of Chapter

Back

Creating Tests

You can quickly create a test by recording the operations you perform on your Web site.

This chapter describes:

Find

Find Again

◀ ▶

Top of Chapter

Back

About Creating Tests

The Virtual User Recorder enables you to generate an automated test by recording the typical processes that you perform on your Web site. As you navigate through your site, the Virtual User Recorder graphically displays each *step* you perform as an icon in a *test tree*. A step is anything a user does that changes the content of a Web page in your site, for example, clicking a link, or typing data into an edit box.

While recording, you can insert checkpoints into your test. A *checkpoint* compares the current value of the specified property with the expected one in order to help you determine whether or not your Web site is functioning correctly.

When you test your site, you may want to check how it performs the same operations with multiple sets of data. This is called *parameterizing* your test. The data is stored in a table in the Data pane. When you parameterize your test, the Virtual User Recorder substitutes the parameters in your test with values from the table. During each *iteration, or repetition,* of your test, the Virtual User Recorder inserts a different value in the parameterized statements. The Virtual User Recorder runs one iteration of your test for each set of values in the Data pane.

After recording, you can further enhance your test by adding and modifying steps in the test tree.

Find
Find Again
Top of Chapter
Back

Planning a Test

Before you start recording, you should plan your test. You should consider the following:

- Determine the steps you want to record to create your test. Realistic tests that check specific functions and load performance of the application are best.
- Decide which information you want to check during the test. A checkpoint can check for differences in the text strings, objects, and tables in your site. For more information, see Chapter 4, **Creating Checkpoints**.
- Evaluate the types of events you need to record. If you want to record more or fewer events than the Virtual User Recorder generally records by default, you can configure the events you want to record. For more information, see Chapter 18, **Configuring Event Recording**.
- Consider increasing the power and flexibility of your test by replacing fixed values with parameters. When you parameterize your test, you can check how it performs the same operations with multiple sets of data. For more information, see Chapter 7, **Parameterizing Tests**.
- You can change the way that the Virtual User Recorder identifies objects in your application. This is particularly helpful when running tests on objects that change frequently or are created using dynamic content, e.g. from a database. For additional information, see Chapter 13, **Understanding How the Virtual User Recorder Identifies Objects**.

If you are an advanced user, consider using actions to streamline the testing process. For additional information, see Chapter 11, **Working with Actions**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Recording a Test

You create a test by recording the typical processes that users perform on your Web site. The Virtual User Recorder records each step you perform and generates a test tree.

Note that by default, each test includes a single action, but a test can include multiple actions. This chapter describes how to record a test with a single action. For information on why and how to work with multiple actions, see Chapter 11, **Working with Actions**.

Consider the following guidelines when recording a test:

- Before you start to record, close all applications not required for the test.
- Determine the security zone of your site. When you record your test, the Web browser may prompt you with security alert dialog boxes. You may choose to disable/enable these dialog boxes.
- You can control how the Virtual User Recorder records and displays your tests by setting testing options in the Options dialog box. For more information, see Chapter 23, **Setting the Virtual User Recorder Testing Options**.

To record a test:

1 Open the Virtual User Recorder. For more information, see **Starting the Virtual User Recorder** on page 25.

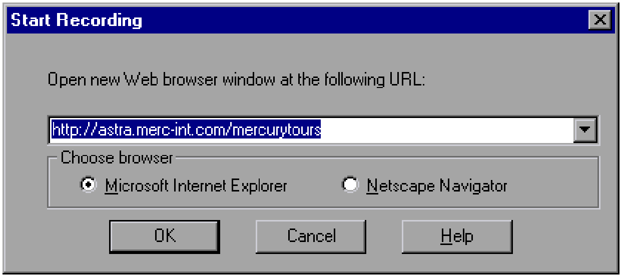
Find
Find Again
Top of Chapter
Back

2 Open a test:

- To create a new test, click the **New** button or choose **File > New**.
- To open an existing test, click the **Open** button or choose **File > Open**. In the **Open** dialog box, select a test and click **Open**.

For more information, see **Managing a Test** on page 52.

3 Click the **Start Record** button or choose **Test > Record**. The Start Recording dialog box opens.



Start Recording

Open new Web browser window at the following URL:

http://astra.merc-int.com/mercurytours

Choose browser

● Microsoft Internet Explorer ○ Netscape Navigator

Choose which browser to use and specify a URL. By default, the URL for the Mercury Tours site appears as the address.

Note: If you choose to use an existing Web browser window, then you must start the browser session after starting the Virtual User Recorder.

Find
Find Again
Top of Chapter
Back

4. Navigate through your site. The Virtual User Recorder records each step you make in the test tree in the Tree View tab.

5. You can insert text checkpoints and object checkpoints to compare the current value of the specified property with the expected one, in order to determine whether or not a site is functioning correctly. For more information, see Chapter 4, **Creating Checkpoints**.

6. You can insert load testing elements to measure how your application functions under load. For more information, see Chapter 6, **Testing Load**.

7. You can parameterize your test to check how it performs the same operations with multiple sets of data. For more information, see Chapter 7, **Parameterizing Tests**.

8. When you complete your Web session, click the **Stop** button or choose **Test > Stop**.

9. To save your test, click the **Save** button or choose **File > Save**. Assign a name to the test. For more information, see **Managing a Test** on page 52.

Find
Find Again
Top of Chapter
Back

Creating Checkpoints

You use the Virtual User Recorder to add checks to your automated test. A *checkpoint* is a verification point on a Web page that compares a current value for a specified property with the expected value for that property. This enables you to identify whether or not your Web site is functioning correctly.

You can create checkpoints to check various objects in a Web site. You can check Web page statistics, text strings, objects, and tables. You can also use formulas to check that the data displayed on a Web page is valid. For information, see Chapter 5, **Checking Web Objects**.

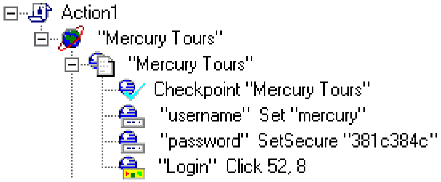
For general information about creating checkpoints, see Chapter 4, **Creating Checkpoints**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Understanding Your Test

While recording, the Virtual User Recorder creates a *test tree*—a graphical representation of the navigation you perform on your site. The test tree appears in the Tree View tab. Each step in the tree represents a step performed on your site and browser.

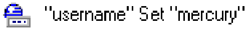
The following is a sample test of a login procedure to the Mercury Tours site, Mercury Interactive's sample Web site.



```
Action1
  "Mercury Tours"
    "Mercury Tours"
      Checkpoint "Mercury Tours"
      "username"  Set "mercury"
      "password"  SetSecure "381c384c"
      "Login"  Click 52, 8
```

Find
Find Again
Top of Chapter
Back

The table below provides an explanation of each step in the tree.

Step	Description
Action1	The action number.
"Mercury Tours"	The browser invokes the *Mercury Tours* site.
"Mercury Tours"	*The name of the Web page*.
Checkpoint "Mercury T	A checkpoint that checks for broken links in the *Mercury Tours* page.
 "username" Set "mercury"	*username* is the name of the edit box. *Set* is the method performed on the edit box. *mercury* is the value of the edit box.
"password" SetSecure "381c384c"	*password* is the name of the edit box. *SetSecure* is an encrypt method performed on the edit box. *381c384c* is the encrypted value of the password.
"Login" Click 52, 8	*Login* is the name of the image. *Click* is the method performed on the image. *52, -8* are the x- and y-coordinates where the image was clicked.

Find
Find Again
Top of Chapter
Back

Modifying Steps in Your Test

As the content of your Web site changes, you can continue to use tests you developed previously. Your test includes all steps you perform, such as clicking hypertext and image links. As Web sites change, the objects in the steps may also change.

Suppose an object in a step in your test changes. You would need to modify the step in your test containing the object so that Astra LoadTest can continue to identify it. You can modify the step by modifying one or more property values of the object in the Object Properties dialog box.

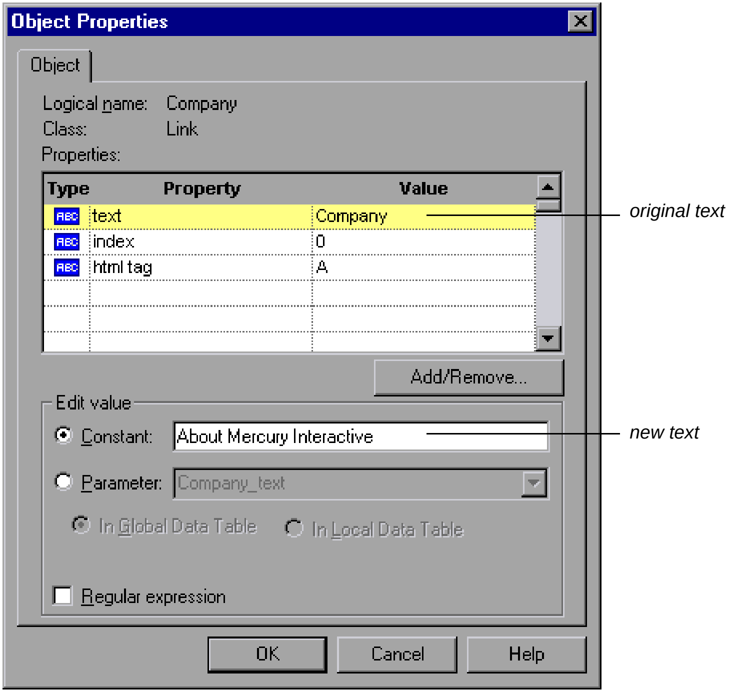
For example, the Mercury Interactive Web site (*www.mercuryinteractive.com*) has a “Home” hypertext link. Suppose that the text string in this link is changed to “About Mercury Interactive”. You need to update your test so that Astra LoadTest will identify the link properly.

To modify a step in your test:

1 Right-click the step containing the object that changed, and choose **Object Properties**.

Find
Find Again
Top of Chapter
⇦ Back

The Object Properties dialog box for the selected object opens and displays the properties Astra LoadTest uses to identify the object.



Object Properties

Logical name: Company
Class: Link
Properties:

Type	Property	Value
ABC	text	Company
ABC	index	0
ABC	html tag	A

Edit value
● Constant: About Mercury Interactive
○ Parameter: Company_text
● In Global Data Table ○ In Local Data Table
□ Regular expression

2 Highlight the property and value to modify.

3 In the **Constant** box, enter a new value for the property.

4 Click **OK** to close the dialog box.

Find
Find Again
Top of Chapter
Back

Changing the ActiveScreen

As the content of your Web site changes, you can continue to use tests you developed previously. You simply change the ActiveScreen display so that the Virtual User Recorder can continue to find the objects in your modified site.

For example, suppose that one of the screens in the Mercury Tours site now includes a new object and you want add a checkpoint that checks for this object. You can use the Change ActiveScreen command to replace the screen in your ActiveScreen tab and then proceed to create a checkpoint for this object.

To change the ActiveScreen:

1 Make sure that your Web browser displays the page that you want to use to replace with the current ActiveScreen tab display.

2 In the test tree, click a step that you want to change, the page is displayed in the ActiveScreen tab.

3 Choose **Tools > Change ActiveScreen**. The Virtual User Recorder window is minimized, the mouse pointer becomes a pointing hand.

4 Click the page displayed in your browser. A message asks you whether you want to change your current ActiveScreen display.

5 Click **Yes**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Managing a Test

You can use the File toolbar to create, open, save, and print recorded tests.

Creating a New Test

To create a new test, click the **New** button or choose **File > New**. A new test opens. You are ready to start recording your test.

Opening an Existing Test

You can open an existing test in order to enhance or run it.

To open an existing test:

1. Click the **Open** button or choose **File > Open**. The **Open** dialog box opens.
2. Select a test and click **Open**. The test opens and the title bar displays the test name.

Find
Find Again
◀ ▶
Top of Chapter
Back

Saving a Test

You can save a new test or save changes to an existing test.

To save a new test:

1. Click the **Save** button or choose **File > Save** to save the test. The **Save** dialog box opens.
2. Choose the folder in which you want to save the test.
3. Type a name for the test in the **File** name box.
4. Click **Save**. The Virtual User Recorder displays the test name in the title bar.

To save changes to an existing test:

- Click the **Save** button or choose **File > Save** to save changes to the test.
- Choose **File > Save As** to save an existing test to a new name or a new location.

Printing a Test

You can print your test.

To print a test:

1. Click the **Print** button or choose **File > Print**. The **Print** dialog box opens.
2. Click **OK** to print.

Find
Find Again
Top of Chapter
Back

You can check objects in your Web site to ensure that your Web site functions as desired.

This chapter describes:

- Checking Objects
- Adding Checkpoints to a Test
- Understanding the Checkpoint Properties Dialog Box
- Modifying Checkpoints

Find
Find Again
Top of Chapter
Back

About Creating Checkpoints

The Virtual User Recorder enables you to add checks to your test. A *checkpoint* is a verification point that compares a current value for a specified property with the expected value for that property. This enables you to identify whether or not your Web site is functioning correctly.

When you add a checkpoint, the Virtual User Recorder adds a checkpoint icon under the highlighted step in the test tree. When you run the test, Astra LoadTest compares the expected results of the checkpoint to the current results. If the results do not match, the checkpoint fails. You can view the results of the checkpoint in the Test Results window.

Find
Find Again
◀ ▶
Top of Chapter
Back

Checking Objects

You can create checkpoints to check various objects in a Web site. You can check Web page statistics, text strings and objects.You can also use formulas to check that the data displayed on a Web page is valid. For information, see Chapter 5, **Checking Web Objects**.

Adding Checkpoints to a Test

You can add checkpoints during or after recording a test. It is generally more convenient to define checks once the initial test has been recorded.

There are several ways to add checkpoints:

- Use the commands on the Insert menu or click the arrow beside the **Insert Checkpoint** button on the Main toolbar. This displays a menu of checkpoint options that are relevant to the selected step in the test tree.
- Right-click the step where you want to add the checkpoint and choose **Insert Checkpoint**.
- Right-click the ActiveScreen and choose **Insert Checkpoint**. This option can be used only after you record a test.

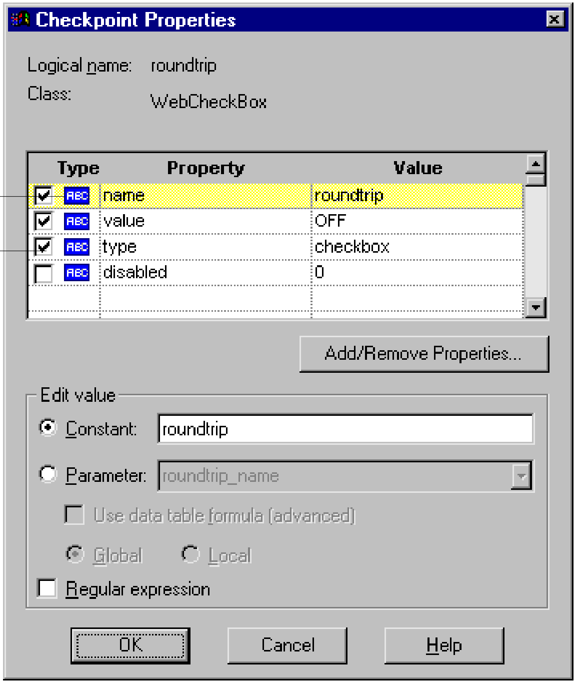
Find
Find Again
Top of Chapter
Back

Understanding the Checkpoint Properties Dialog Box

While the Checkpoint Properties dialog box varies slightly depending on the type of object you are checking, the Checkpoint Properties dialog box generally includes the following basic elements:

Note: The Text Checkpoint Properties dialog box is quite different from the Checkpoint Properties dialog box described below. For more information see **Understanding the Text Checkpoint Properties Dialog Box** on page 79.

Find
Find Again
◀ ▶
Top of Chapter
Back



Checkpoint Properties

Logical name: roundtrip

Class: WebCheckBox

	Type	Property	Value
☑	ABC	name	roundtrip
☑	ABC	value	OFF
☑	ABC	type	checkbox
□	ABC	disabled	0

Edit value

● Constant: roundtrip

○ Parameter: roundtrip_name

□ Use data table formula (advanced)

● Global ○ Local

□ Regular expression

This icon indicates that the value of the property to check is a constant.

The selected check box indicates that this property will be checked.

Identifying the Object

The top part of the dialog box displays basic information about the object to check such as the name and type of object.

Find
Find Again
Top of Chapter
Back

Choosing which Property to Check

The next part of the dialog box displays the available properties for the object, and enables you to select which properties to check.

You can also create checkpoints on objects with variable descriptions. For more information, see Chapter 13, **Understanding How the Virtual User Recorder Identifies Objects**.

Modifying the Expected Value

In the **Edit value** section, you can edit the value of any property that you want to check.

If the expected value of one of the properties you are checking changes, you don’t have to rerecord the object. You can modify the expected value by selecting the relevant property and changing its value in the **Constant** box.

You can parameterize the expected value of an object by entering a parameter name in the **Parameter** box and entering the expected values in the appropriate column in the data table. For more information on parameterization, see Chapter 7, **Parameterizing Tests**.

You can also enter the expected value as regular expression in the **Constant** box or in the data table cells for the parameter. If you choose to enter a regular expression for the expected value, select **Regular expression** at the bottom of the Edit value section.

For more information data tables, see Chapter 9, **Working with Data Tables**. For more information on regular expressions, see Chapter 10, **Using Regular Expressions**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Modifying Checkpoints

If you already created a checkpoint, or if the Virtual User Recorder automatically created page checkpoints, you can modify the settings. For example, you can choose to use parameters, or you can use filters to specify which image sources and links to check.

To modify a checkpoint:

1 Right-click an existing checkpoint in the test tree and choose **Function Arguments**. A checkpoint dialog box opens.

2 Modify the properties and click **OK**.

Tip: You can also open a checkpoint dialog box by double-clicking a checkpoint in your test tree.

Find
Find Again
◀ ▶
Top of Chapter
Back

Checking Web Objects

By adding Web object checkpoints to your tests, you can compare Web object in different versions of your Web site.

This chapter describes:

Find

Find Again

◀ ▶

Top of Chapter

Back

About Checking Web Objects

You can check the Web objects in your Web site to determine whether your Web site functions as desired. Web object checkpoints compare the expected values of object properties to the object’s current values during a test run. You can perform checks on Web page properties, text, tables and other Web objects such as images and form elements.

Find
Find Again
Top of Chapter
Back

Checking Pages

You can check statistical information on your Web pages by adding page checkpoints to your test. These checkpoints check the links and the source of images on a Web page. You can also instruct page checkpoints to include a check for broken links.

When you record, by default the Virtual User Recorder creates automatic page checkpoints in your test tree. If you instruct the Virtual User Recorder not to create automatic page checkpoints, you can insert them for pages you want to check, as described in **Creating a Page Checkpoint** on page 64.

Find
Find Again
Top of Chapter
⇦ Back

Note: You can instruct the Virtual User Recorder not to create automatic page checkpoints in your test tree while you record by clearing the **Add automatic checks for each page during record** check box in the Page Verification tab of the Options dialog box. You can also instruct the Virtual User Recorder not to perform automatic page checkpoints when you run your test by clearing the **Don’t perform automatic checkpoints during test run** check box in the Page Verification tab of the Options dialog box. For more information, see Chapter 23, **Setting the Virtual User Recorder Testing Options**.

You cannot create a page checkpoint if one has already been created automatically. However, you can modify a page checkpoint that has already been created, as described in **Understanding the Page Checkpoint Properties Dialog Box** on page 65.

Creating a Page Checkpoint

If you did not choose to add page checkpoints automatically while recording, you can add a page checkpoint to your script to check the links and the image sources on a selected Web page either while recording or after recording.

To add a page checkpoint while recording:

1 Click the page step in your test tree where you want to add a checkpoint. The ActiveScreen displays the Web page corresponding to the highlighted step.

2 Choose **Insert > Checkpoint** and click in the page you want to check. The **Object Selection - Checkpoint Properties** dialog box opens.

3 Select the page item and click **OK**. The **Page Checkpoint Properties** dialog box opens.

4 Modify the settings for the checkpoint in the Page Checkpoint Properties dialog box, as described in **Understanding the Page Checkpoint Properties Dialog Box** on page 65.

5 When you are done, click **OK** to close the dialog box.

A tree item with a checkpoint ✓ icon is added to your test tree.

To add a page checkpoint after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Right-click anywhere on the ActiveScreen and choose **Insert Checkpoint.** The **Object Selection - Checkpoint Properties** dialog box opens.

Find
Find Again
Top of Chapter
Back

3 Select the page item and click **OK**. The **Page Checkpoint Properties** dialog box opens.

4 Specify the settings for the checkpoint in the Page Checkpoint Properties dialog box, as described in **Understanding the Page Checkpoint Properties Dialog Box** on page 65.

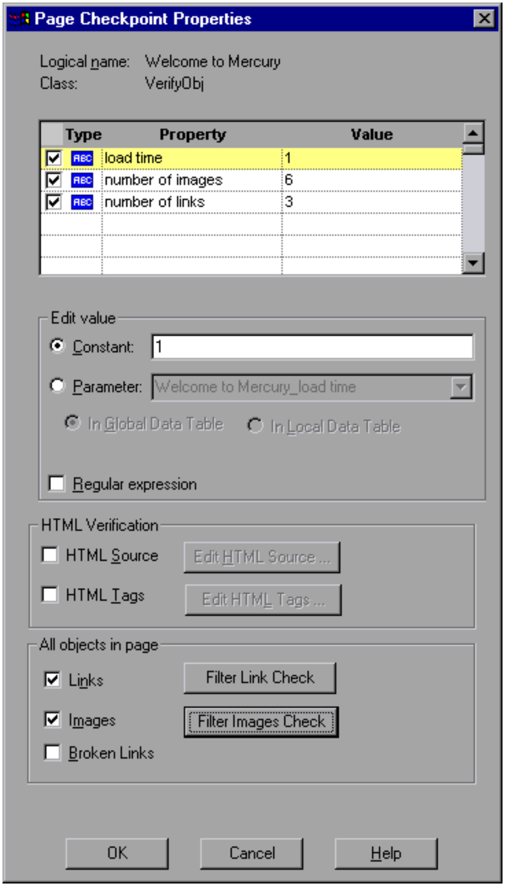
5 When you are done, click **OK** to close the dialog box.

A tree item with a checkpoint ✓ icon is added to your test tree.

Understanding the Page Checkpoint Properties Dialog Box

The Page Checkpoint Properties dialog box enables you to choose which properties to check.

Find
Find Again
◀ ▶
Top of Chapter
Back



Page Checkpoint Properties

Logical name: Welcome to Mercury
Class: VerifyObj

	Type	Property	Value
☑	ABC	load time	1
☑	ABC	number of images	6
☑	ABC	number of links	3

Edit value

● Constant: 1

○ Parameter: Welcome to Mercury_load time

● In Global Data Table ○ In Local Data Table

□ Regular expression

HTML Verification

□ HTML Source

□ HTML Tags

All objects in page

☑ Links

☑ Images

□ Broken Links

Find
Find Again
Top of Chapter
Back

Identifying the Object

The top part of the dialog box displays information about the object to check:

Information	Description
Logical name	The title of the Web page as defined in the HTML code.
Class	The type of object.

Choosing which Property to Check

The default properties for the object are listed in the **Properties** pane of the dialog box. The pane includes the properties, their values, and their types:

Pane Element	Description
check box	For each object class, the Virtual User Recorder recommends default property checks. You can accept the default checks or modify them accordingly. To include a property check, select the corresponding check box. To exclude a property check, clear the corresponding check box.
Type	The ABC icon indicates that the value of the property is a constant. The ▦ icon indicates that the value of the property is a parameter.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Pane Element	Description
Property	The name of the property to check.
Value	The value of the property to check. Note that unless you edit this value, the value in the page will be the expected value of the property when you run your test. For information about editing the value of a property, see **Editing the Value of a Property** on page 69.

Find
Find Again
◀ ▶
Top of Chapter
Back

Editing the Value of a Property

In the **Edit value** section, you use the following options to edit the value of the property to check.

Option	Description
Constant (default)	Sets the value of the property.
Parameter	Sets the property value as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.
Global	Adds a parameter to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds a parameter to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Use data table formula (advanced)	Inserts two columns in the table in the Data pane: the first column contains a formula that checks the validity of output in the second column. uses the data in the second (output) column to compute the formula, and inserts a value of TRUE or FALSE in the table cell of the first (formula) column. For more information, see Chapter 9, **Working with Data Tables**.
Regular expression	Sets the value as a regular expression. For more information, see Chapter 10, **Using Regular Expressions**.

Find
Find Again
Top of Chapter
Back

Checking the HTML Source

In the **HTML Source** section, you can check the HTML source and tags of the page.

Option	Description
HTML Source	Checks that the source in the web page being tested matches the expected HTML code (the source code of the page at the time that the test is recorded).
Edit HTML Source (enabled only when the HTML Source check box is selected)	Opens the **HTML Source** dialog box, which displays the expected HTML code. Note that you can also use regular expressions when editing the expected HTML source code if you click the regular expression check box at the bottom of the page. Edit the expected HTML source code and click **OK**.
HTML Tags	Checks that the HTML tags in the web page being tested match the expected HTML tags (the HTML tags on the page at the time that the test is recorded).
Edit HTML Tags (enabled only when the HTML Tags check box is selected)	Opens the **HTML Tags** dialog box, which displays the expected HTML tags. Note that you can also use regular expressions when editing the HTML tags if you click the regular expression check box at the bottom of the page. Edit the expected HTML tags and click **OK**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Checking All the Objects in a Page

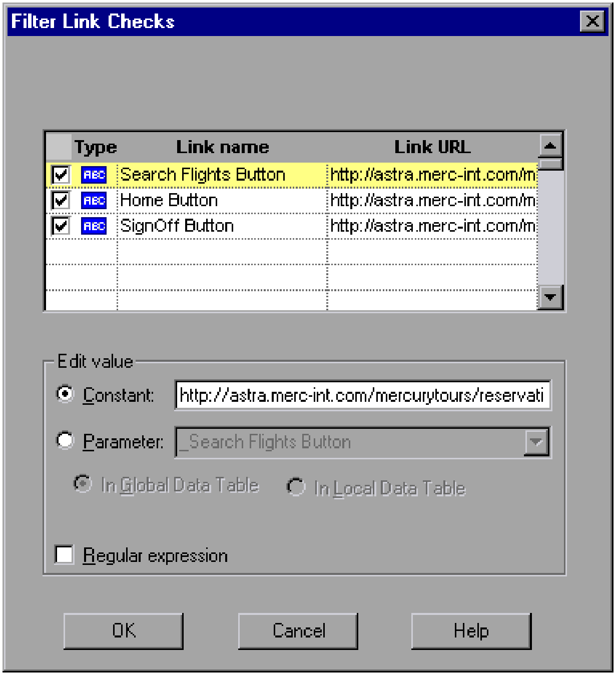
In the **All objects in page** section, you can check all the links, images, and broken links in a page. You can use the following options to check the objects in a page:

Option	Description
Links	Checks the functionality of the links in the page.
Filter Link Check	Opens the Filter Link Checks dialog box, which enables you to specify which hypertext links to check in the page. For additional information, see **Filtering Hypertext Links** on page 71.
Images	Checks that the images are displayed on the page.
Filter Images Check	Opens the Filter Image Check dialog box, which enables you to specify which image links to check in the page. For additional information, see **Filtering Image Sources** on page 74.
Broken Links	Checks for broken links in the page.

Filtering Hypertext Links

You can filter which hypertext links to check in a page checkpoint using the Filter Link Checks dialog box. You open this dialog box from the Page Checkpoint Properties dialog box. For additional information, see **Checking All the Objects in a Page** on page 71.

Find
Find Again
◀ ▶
Top of Chapter
Back



Filter Link Checks

	Type	Link name	Link URL
☑	ABC	Search Flights Button	http://astra.merc-int.com/m
☑	ABC	Home Button	http://astra.merc-int.com/m
☑	ABC	SignOff Button	http://astra.merc-int.com/m

Edit value

● Constant: http://astra.merc-int.com/mercurytours/reservati

○ Parameter: _Search Flights Button

● In Global Data Table ○ In Local Data Table

□ Regular expression

In this dialog box, you can specify which hypertext links to check.

Find
Find Again
Top of Chapter
Back

Choosing which Hypertext Links to Check

You can choose which hypertext links to check in a page checkpoint.

Pane Element	Description
check box	Each link on the page has a corresponding check box. To check a link, select the corresponding check box (by default all links are selected). To exclude a link from the page checkpoint, clear the corresponding check box.
Type	The ABC icon indicates that the target URL is a constant. The ▦ icon indicates that the target URL is a parameter.
Link name	The text in the hypertext link.
Link URL	The target URL.

Find
Find Again
Top of Chapter
Back

Editing the Value of the Target URL

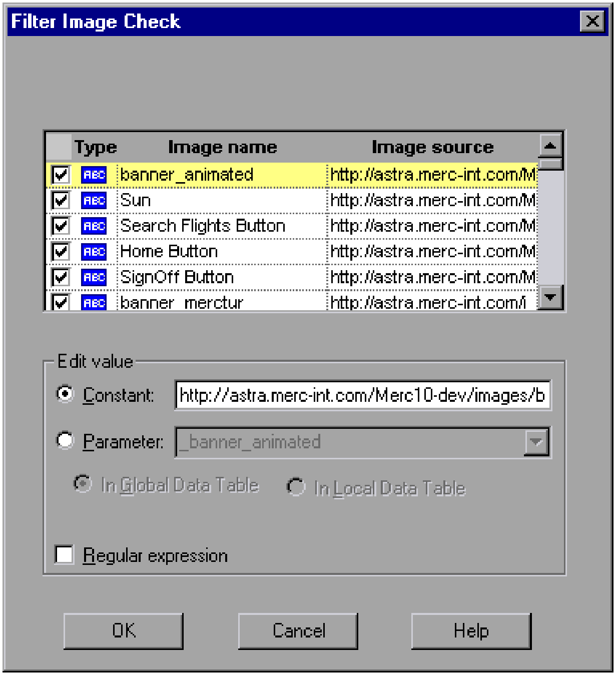
In the Edit value section, you use the following options to edit the value of the target URL to which the hypertext links

Option	Description
Constant (default)	Sets the value of the target URL.
Parameter	Sets the target URL as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.
Global	Adds a parameter to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds a parameter to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Regular expression	Sets the value as a regular expression. For more information, see Chapter 10, **Using Regular Expressions**.

Filtering Image Sources

You can filter which image sources to check in a page checkpoint using the Filter Image Check dialog box. You open this dialog box from the Page Checkpoint Properties dialog box. For additional information, see **Checking All the Objects in a Page** on page 71.

Find
Find Again
Top of Chapter
Back



Filter Image Check

	Type	Image name	Image source
☑	ABC	banner_animated	http://astra.merc-int.com/M
☑	ABC	Sun	http://astra.merc-int.com/M
☑	ABC	Search Flights Button	http://astra.merc-int.com/M
☑	ABC	Home Button	http://astra.merc-int.com/M
☑	ABC	SignOff Button	http://astra.merc-int.com/M
☑	ABC	banner_merctur	http://astra.merc-int.com/

Edit value

● Constant: http://astra.merc-int.com/Merc10-dev/images/b

○ Parameter: _banner_animated

● In Global Data Table ○ In Local Data Table

□ Regular expression

In this dialog box, you can specify which image sources to check.

Find
Find Again
Top of Chapter
Back

Choosing which Image Sources to Check

You can choose which image sources to check in a page checkpoint.

Pane Element	Description
check box	Each image source on the page has a corresponding check box. To check an image source, select the corresponding check box (by default all image sources are selected). To exclude an image source from the page checkpoint, clear the corresponding check box.
Type	The ABC icon indicates that the image source is a constant. The ▦ icon indicates that the image source is a parameter.
Image name	The name of the image.
Image source	The image source file and path.

Find
Find Again
◀ ▶
Top of Chapter
Back

Editing the Value of the Path of the Image Source File

In the Edit value section, you use the following options to edit the path of the image source file

Option	Description
Constant (default)	Sets the value of the path of the image source file.
Parameter	Sets the path of the image source file as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.
Global	Adds a parameter to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds a parameter to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Regular expression	Sets the value as a regular expression. For more information, see Chapter 10, **Using Regular Expressions**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Checking Text

You can check that a specified text string appears on your Web page by adding a text checkpoint to your test. To add a text checkpoint to your test, you use the Text Checkpoint Properties dialog box.

Creating a Text Checkpoint

You can add a text checkpoint while recording or afterward.

To add a text checkpoint while recording:

1 Highlight a text string on the Web page.

2 Choose **Insert > Text Checkpoint**.

The mouse pointer turns into a pointing hand.

3 Click the text string. The **Text Checkpoint Properties** dialog box opens.

4 Specify the settings for the checkpoint. For more information, see **Understanding the Text Checkpoint Properties Dialog Box** on page 79.

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

A tree item with a checkpoint ✓ icon is added to your test tree.

Find
Find Again
Top of Chapter
Back

To add a text checkpoint after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Click a step in your test where you want to add a checkpoint. The ActiveScreen displays the Web page corresponding to the highlighted step.

3 Highlight a text string on the ActiveScreen.

4 Right-click the text string and choose **Insert Text Checkpoint**. The **Text Checkpoint Properties** dialog box opens.

5 Specify the settings for the checkpoint. For more information, see **Understanding the Text Checkpoint Properties Dialog Box** on page 79.

6 Click **OK** to close the dialog box.

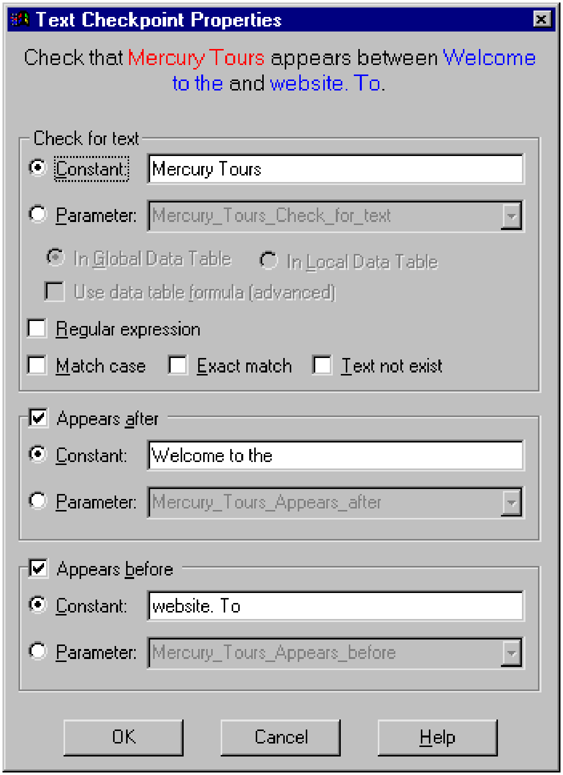
A tree item with a checkpoint ✓ icon is added to your test tree.

Understanding the Text Checkpoint Properties Dialog Box

In the Text Checkpoint Properties dialog box, you can specify which text to check as well as which text appears before and after the text to check. This is particularly helpful when the text string you want to check appears several times in the same Web page. For example, suppose you want to check that the "Mercury Tours" text string appears in a specific location in the first page of the sample Web site,

Find
Find Again
◀ ▶
Top of Chapter
Back

Mercury Tours. This text string actually appears three times on that Web page. To check for the text string in a specific location, you can specify which text precedes and/or follows the text string you are checking.



Check that Mercury Tours appears between Welcome to the and website. To.

Check for text
● Constant: Mercury Tours
○ Parameter: Mercury_Tours_Check_for_text
● In Global Data Table ○ In Local Data Table
□ Use data table formula (advanced)
□ Regular expression
□ Match case □ Exact match □ Text not exist

☑ Appears after
● Constant: Welcome to the
○ Parameter: Mercury_Tours_Appears_after

☑ Appears before
● Constant: website. To
○ Parameter: Mercury_Tours_Appears_before

Find
Find Again
Top of Chapter
Back

Specifying which Text to Check

In the **Check for text** section, you use the following options to specify which text to check:

Option	Description
Constant (default)	The text the Virtual User Recorder checks when running the test.
Parameter	Sets the text string as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.
Use data table formula (advanced)	Inserts two columns in the table in the Data pane. The first column contains a formula that checks the validity of output in the second column. uses the data in the second (output) column to compute the formula, and inserts a value of TRUE or FALSE in the table cell of the first (formula) column. For more information, see Chapter 9, **Working with Data Tables**.
Global	Adds a parameter to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds a parameter to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Regular expression	Sets the text string as a regular expression. For more information, see Chapter 10, **Using Regular Expressions**.
Match case	Conducts a case sensitive search.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Option	Description
Exact match	Checks according to the exact expected text.
Text not exist	Checks that text string does not appear.

Specifying What Appears After the Text to Check

In the **Appears after** section, you use the following options to specify which text, if any, should appear after the text to check:

Option	Description
Appears after (default)	Checks that the text to check appears after the text in this box. To ignore the text that appears after the text to check, clear this check box.
Constant (default)	Displays the text that appears after the text to check.
Parameter	Sets the text string as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Specifying What Appears Before the Text to Check

In the **Appears before** section, you use the following options to specify which text, if any, should appear before the text to check:

Option	Description
Appears before (default)	Checks that the text to check appears after the text in this box. To ignore the text that appears before the text to check, clear this check box.
Constant (default)	Displays the text that appears before the text to check.
Parameter	Sets the text string as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.

Find
Find Again
Top of Chapter
Back

Checking Objects

You can check that a specified object appears on your Web page by adding an object checkpoint to your test. To add an object checkpoint to your test, you use the Checkpoint Properties dialog box.

Creating an Object Checkpoint

You can add an object checkpoint while recording or afterward.

To add an object checkpoint while recording:

1 Click the **Insert Checkpoint** button or choose **Insert > Checkpoint**.

The mouse pointer turns into a pointing hand.

2 Click the object to check. The **Object Selection - Checkpoint Properties** dialog box opens.

3 Select the last tree item. The tree item name depends on the object's class, for example:

Object	Class
Check box	WebCheckBox
Edit box	WebEdit

Find
Find Again
Top of Chapter
Back

Object	Class
Image	Image
Radio button	WebRadio

4 Click **OK**. The **Checkpoint Properties** dialog box opens.

5 Specify the settings for the checkpoint. For more information, see **Understanding the Checkpoint Properties Dialog Box** on page 88.

6 Click **OK** to close the dialog box.

A tree item with a checkpoint ✓ icon is added to your test tree.

Find
Find Again
Top of Chapter
Back

To add an object checkpoint after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Click a step in your test where you want to add a checkpoint. The ActiveScreen displays the Web page corresponding to the highlighted step.

3 Right-click an object on the ActiveScreen and choose Select **Insert Checkpoint**.

4 Click the object to check. The **Object Selection - Checkpoint Properties** dialog box opens.

5 Select the last tree item. The tree item name depends on the object’s class, for example:

Object	Class
Check box	WebCheckBox
Edit box	WebEdit
Image	Image
Radio button	WebRadio

6 Click **OK**. The **Checkpoint Properties** dialog box opens.

7 Specify the settings for the checkpoint. For more information, see **Understanding the Checkpoint Properties Dialog Box** on page 88.

8 Click **OK** to close the dialog box.

Find
Find Again
Top of Chapter
Back

A tree item with a checkpoint ✓ icon is added to your test tree.

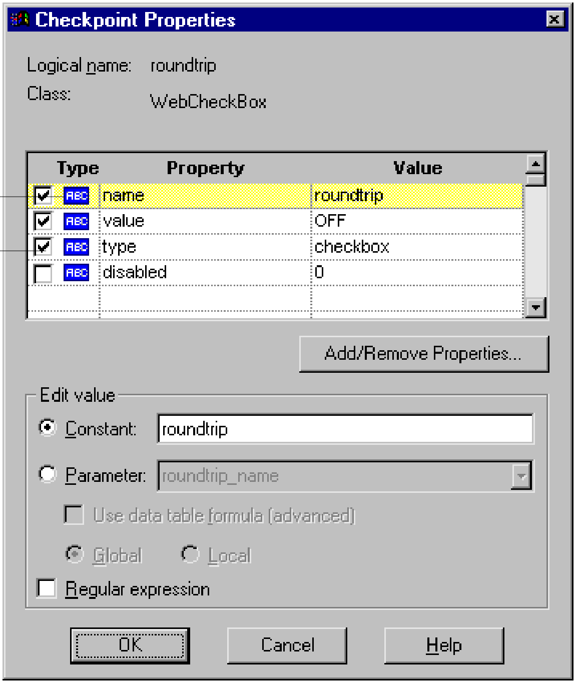
Find
Find Again
◀ ▶
Top of Chapter
Back

Understanding the Checkpoint Properties Dialog Box

In the Checkpoint Properties dialog box, you can specify which properties of the

Find
Find Again
Top of Chapter
Back

object to check, and edit the values of these properties.



This icon indicates that the value of the property to check is a constant.

The selected check box indicates that this property will be checked.

Checkpoint Properties

Logical name: roundtrip

Class: WebCheckBox

	Type	Property	Value
☑	ABC	name	roundtrip
☑	ABC	value	OFF
☑	ABC	type	checkbox
□	ABC	disabled	0

Add/Remove Properties...

Edit value

● Constant: roundtrip

○ Parameter: roundtrip_name

□ Use data table formula (advanced)

● Global ○ Local

□ Regular expression

OK | Cancel | Help

Find
Find Again
Top of Chapter
Back

Identifying the Object

The top part of the dialog box displays information about the object to check:

Information	Description
Logical name	The name of the object as defined in the HTML code of the Web page.
Class	The type of object. In this example, the "WebCheckBox" class indicates that the object is a check box.

Find
Find Again
Top of Chapter
Back

Choosing which Property to Check

The dialog box also displays the default properties of the object you can check in the **Properties** pane, which lists the properties, their values, and their types:

Pane Element	Description
check box	For each object class, the Virtual User Recorder recommends default property checks. You can accept the default checks or modify them accordingly. To check a property, select the corresponding check box. To exclude a property check, clear the corresponding check box.
Type	The ABC icon indicates that the value of the property is a constant. The ▦ icon indicates that the value of the property is a parameter.
Property	The name of the property to check.
Value	The value of the property to check. Note that unless you edit this value, the listed value will be the expected value of the property when you run your test. For information about editing the value of a property, see **Editing the Value of an Object Property** on page 91.

Find
Find Again
Top of Chapter
Back

Editing the Value of an Object Property

In the **Edit value** section, you use the following options to edit the value of the property to check.

Option	Description
Constant (default)	Sets the value of the property.
Parameter	Sets the property value as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.
Global	Adds a parameter to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds a parameter to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Use data table formula (advanced)	Inserts two columns in the table in the Data pane. The first column contains a formula that checks the validity of output in the second column. The Virtual User Recorder uses the data in the second (output) column to compute the formula, and inserts a value of TRUE or FALSE in the table cell of the first (formula) column. For more information, see Chapter 9, **Working with Data Tables**.
Regular expression	Sets the property value as a regular expression. For more information, see Chapter 10, **Using Regular Expressions**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Creating Tests
Testing Load

Today's Web applications are accessed by multiple application clients over complex architectures. With Astra LoadTest, you can emulate the load of real users interacting with your application and measure system performance.

The Virtual User Recorder enables you to customize your test to accurately measure the performance of your Web application under load.

This chapter describes:

Find

Find Again

◀ ▶

Top of Chapter

Back

Inserting Transactions

To measure the performance of the server, you define *transactions*. A transaction represents an action or a set of actions that you are interested in measuring. You define transactions within your Vuser script by enclosing the appropriate sections of the script with *start* and *end* transaction statements. For example, you can define a transaction that measures the time it takes for the server to process a request to reserve a seat on a flight and for the confirmation to be displayed at the application client’s terminal.

When you record a test, Astra LoadTest automatically marks each page you browse as a transaction. This means that when you run a scenario, each page in your test tree is recognized as a transaction to be measured.

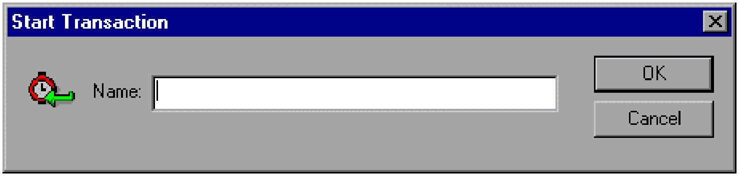
The automatic transactions create a great deal of general analysis information. As you refine your test, you may want to remove the automatic transactions and insert transactions that will measure the performance of specific business processes. You manually insert a transaction to mark a group of steps that make up the business process that you want to measure.

Find
Find Again
◀ ▶
Top of Chapter
Back

To insert a transaction:

1 In the test tree, click the step where you want your transaction timing to begin. The page opens in the ActiveScreen.

2 Click the **Start Transaction** button. The **Start Transaction** dialog box opens.

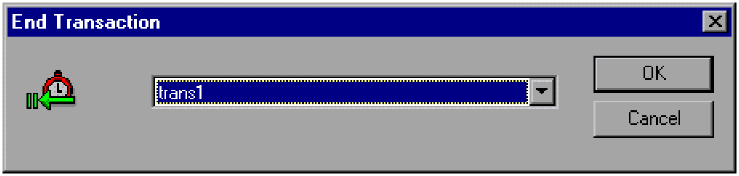


3 Enter a meaningful name in the **Name** box. Click **OK**.

The **Strart Transaction** icon is added to the test tree below the highlighted step.

4 In the test tree, click the step where you want the transaction timing to end. The page opens in the ActiveScreen.

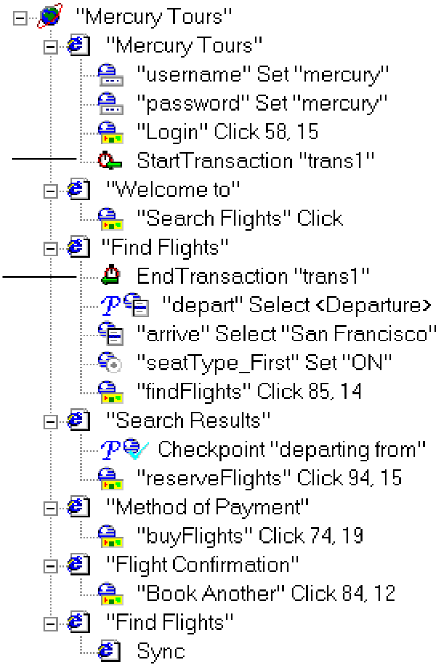
5 Click the **End Transaction** button. The **End Transaction** dialog box opens.



6 The Name box contains the transaction name you entered in the Start Transaction dialog box. Click **OK**.

Find
Find Again
Top of Chapter
Back

The End Transaction icon is added to the test tree above the selected step. A sample test tree with transactions is shown below:



```
"Mercury Tours"
  "Mercury Tours"
    "username" Set "mercury"
    "password" Set "mercury"
    "Login" Click 58, 15
    StartTransaction "trans1"
  "Welcome to"
    "Search Flights" Click
  "Find Flights"
    EndTransaction "trans1"
    "depart" Select <Departure>
    "arrive" Select "San Francisco"
    "seatType_First" Set "ON"
    "findFlights" Click 85, 14
  "Search Results"
    Checkpoint "departing from"
    "reserveFlights" Click 94, 15
  "Method of Payment"
    "buyFlights" Click 74, 19
  "Flight Confirmation"
    "Book Another" Click 84, 12
  "Find Flights"
    Sync
```

Start Transaction

End Transaction

Note: There is no limit to the number of transactions you can add to a test.

Find
Find Again
Top of Chapter
Back

Inserting Rendezvous Points

During the scenario run, you instruct multiple Vusers to perform tasks simultaneously by creating a rendezvous point. This ensures that:

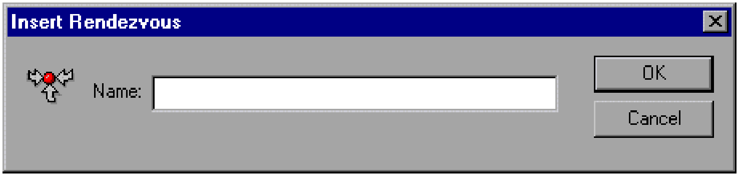
- intense user load is emulated
- transactions are measured under the load of multiple Vusers

A rendezvous point is a meeting place for Vusers. To designate the meeting place, you insert rendezvous statements into your Vuser scripts. When the rendezvous statement is interpreted, the Vuser is held by the Controller until all the members of the rendezvous arrive. When all the Vusers have arrived (or a time limit is reached), they are released together and perform the next task in their Vuser scripts.

To insert a rendezvous point:

1 In the test tree, click the step where you want your intense user load emulated. The page opens in the **ActiveScreen**.

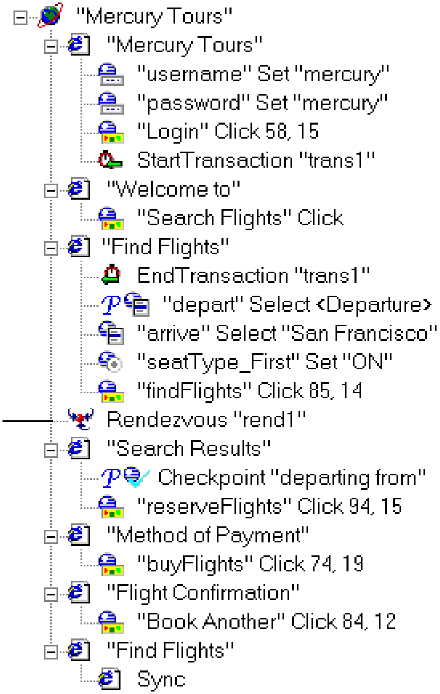
2 Click the **Rendezvous** button. The **Rendezvous** dialog box opens.



Find
Find Again
Top of Chapter
Back

3 Enter a meaningful name in the **Name** box. Click **OK**.

A test tree containing a rendezvous point is shown below:



```
"Mercury Tours"
    "Mercury Tours"
        "username" Set "mercury"
        "password" Set "mercury"
        "Login" Click 58, 15
        StartTransaction "trans1"
    "Welcome to"
        "Search Flights" Click
    "Find Flights"
        EndTransaction "trans1"
        "depart" Select <Departure>
        "arrive" Select "San Francisco"
        "seatType_First" Set "ON"
        "findFlights" Click 85, 14
    Rendezvous "rend1"
    "Search Results"
        Checkpoint "departing from"
        "reserveFlights" Click 94, 15
    "Method of Payment"
        "buyFlights" Click 74, 19
    "Flight Confirmation"
        "Book Another" Click 84, 12
    "Find Flights"
        Sync
```

Find
Find Again
Top of Chapter
Back

Setting Run-Time Options

Astra LoadTest run-time options affect how your test runs in a load testing scenario. For example, you can set think time options that Astra LoadTest will use when running a test or set the output messages sent by Astra LoadTest.

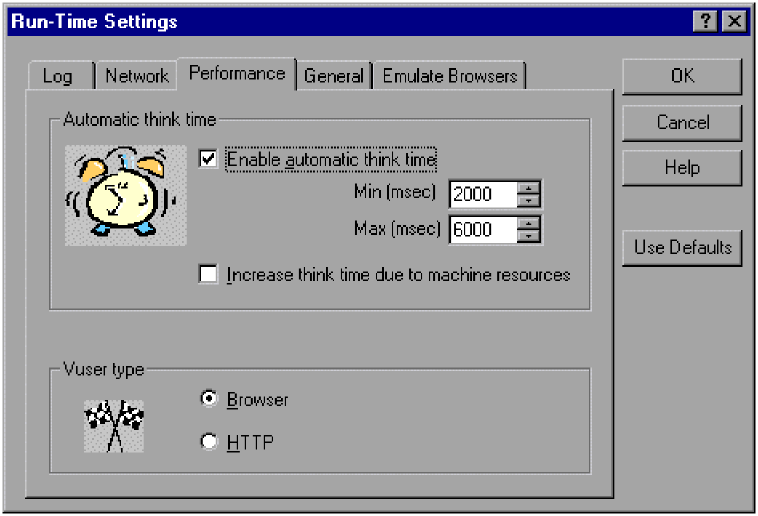
Before you run your test, you can use the Run-time Settings dialog box to modify your testing options. The values you set remain in effect for all tests.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

To set run-time options:

1 Choose **Test > Run-Time Settings**.

The **Run-Time Settings** dialog box opens. It is divided by subject into five tabbed pages.



Automatic think time

☑ Enable automatic think time

Min (msec) 2000

Max (msec) 6000

□ Increase think time due to machine resources

Vuser type

● Browser

○ HTTP

2 Set an option, as described in **Run-Time Settings** on page 101.

3 When you are done, click **OK** to apply your changes and close the dialog box.

Find
Find Again
Top of Chapter
Back

Run-Time Settings

The Run-Time Settings dialog box contains the following tabbed pages:

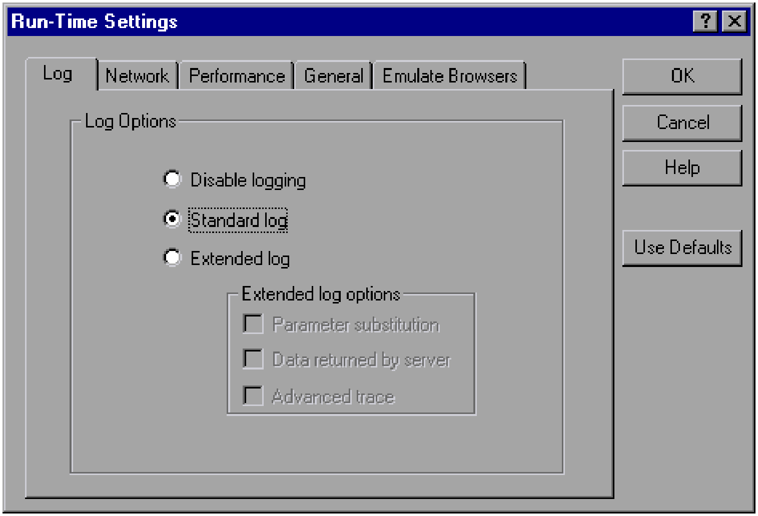
Tab Heading	Subject
Log	Options for output messages
Network	Options on how to handle Web images, Java applets and scripts
Performance	Options for think time
General	Options for run mode and global settings
Emulate Browsers	Options for browser emulation during the running of a scenario.

This section lists the testing options you can set using the Run-Time Settings dialog box.

Find
Find Again
◀ ▶
Top of Chapter
Back

Log Settings

The Log tab options indicate what type of output messages Astra LoadTest should send to the output file.



Log Options

○ Disable logging

● Standard log

○ Extended log

Extended log options

□ Parameter substitution

□ Data returned by server

□ Advanced trace

Find
Find Again
Top of Chapter
Back

The Log tab includes the following options:

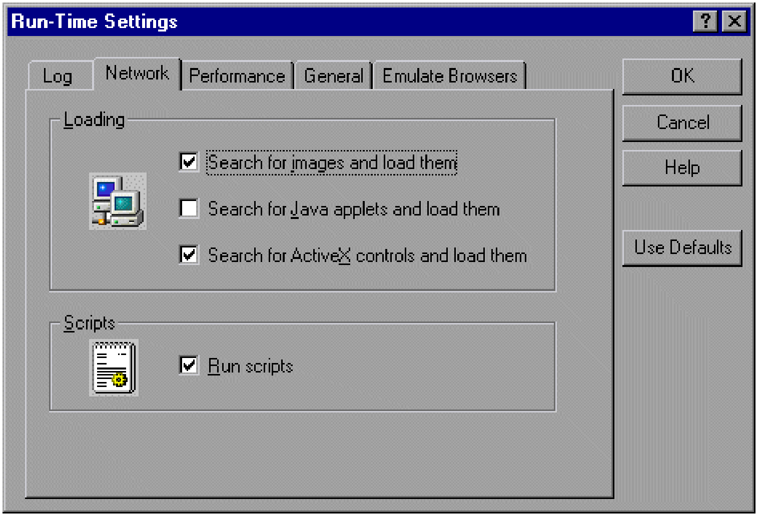
Option	Description
Disable logging	Instructs Astra LoadTest not to send output messages.
Standard log	Instructs Astra LoadTest to send standard output messages. This is the default setting.
Extended log	Instructs Astra LoadTest to send detailed output messages to the output file.
Parameter substitution	Instructs Astra LoadTest to log all the parameters that are replaced while a script runs, and the values that replace the parameters.
Data returned by server	*Instructs* Astra LoadTest *to log all the data that is returned by the server*.
Advanced trace	Instructs Astra LoadTest to log all functions called and messages sent by the Vuser during the session.

Note: The use of the log option results in a higher usage of resources, therefore, you should disable the log options to improve scalability.

Find
Find Again
◀ ▶
Top of Chapter
Back

Network Settings

The Network tab specifies how the Vuser handles Web images, Java applets, ActiveX controls, and scripts.



Loading

☑ Search for images and load them

□ Search for Java applets and load them

☑ Search for ActiveX controls and load them

Scripts

☑ Run scripts

Find
Find Again
Top of Chapter
Back

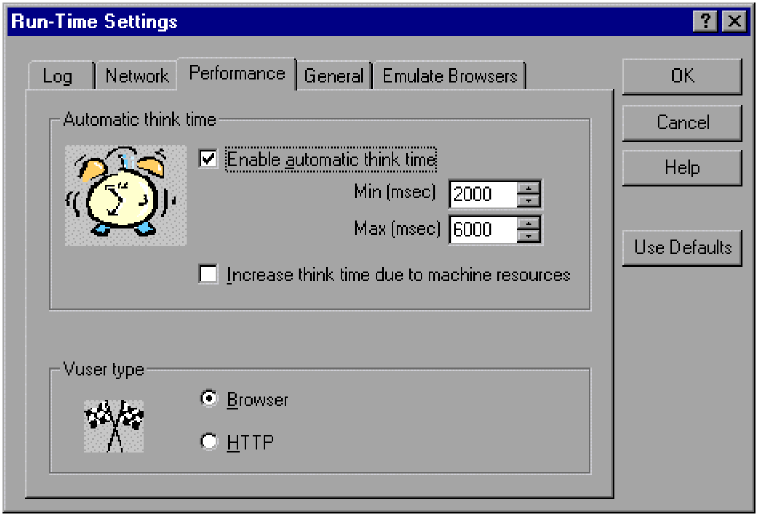
The Network tab includes the following options:

Option	Description
Search for images and load them	Instructs a Vuser to load the graphics associated with a Web page when the Vuser accesses the page during script execution.
Search for Java applets and load them	Instructs a Vuser to load the Java applets associated with a Web page when the Vuser accesses the page during script execution.
Search for ActiveX controls and load them	Instructs a Vuser to load the ActiveX controls associated with a Web page when the Vuser accesses the page during script execution.
Run Scripts	Instructs a Vuser to run scripts associated with a Web page when the Vuser accesses the page during script execution.

Find
Find Again
◀ ▶
Top of Chapter
Back

Performance Settings

The Performance tab gives you the option to set *think time*. Think time emulates the time that a real user waits between actions.



Automatic think time

☑ Enable automatic think time

Min (msec) 2000

Max (msec) 6000

□ Increase think time due to machine resources

Vuser type

● Browser

○ HTTP

Find
Find Again
Top of Chapter
Back

The Performance tab includes the following options:

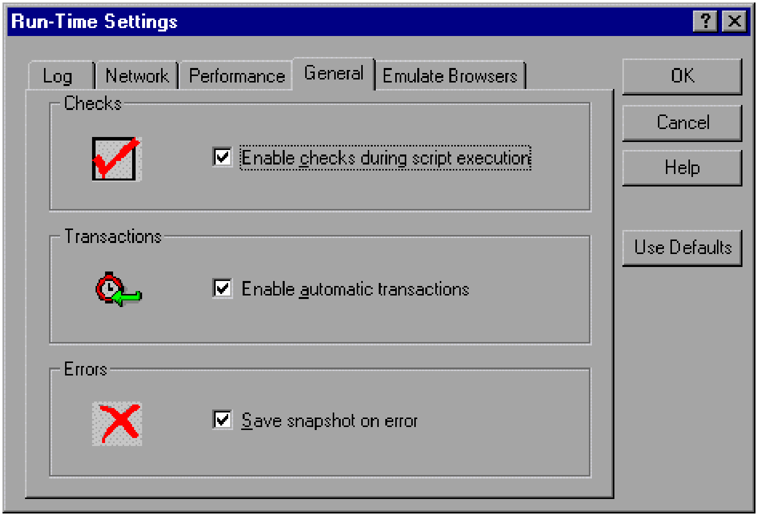
Option	Description
Enable automatic think time	Allows you to designate a range, in milliseconds, for the Vuser to wait between steps. Within the range, the think times are random.
Min (msec)	Sets the minimum wait time in milliseconds.
Max (msec)	Sets the maximum wait time in milliseconds.
Increase think time due to machine resources	Increases the think time to compensate for slower processing times due to hardware limitations.
Browser*	Sets the browser as the replay mode. This is the default setting and is used for standard testing .
HTTP*	Sets HTTP as the replay mode. This setting is used when there is a need for increased scalability and performance when running a test.

Note: The Browser and HTTP choices are only visible when HTTP mode is enabled. For further information on HTTP, see Chapter 17, **Testing in HTTP Mode**.

Find
Find Again
◀ ▶
Top of Chapter
Back

General Settings

The General tab enables Astra LoadTest to perform checks or transactions during a test run.



Run-Time Settings

Checks

☑ Enable checks during script execution

Transactions

☑ Enable automatic transactions

Errors

☑ Save snapshot on error

Find
Find Again
Top of Chapter
Back

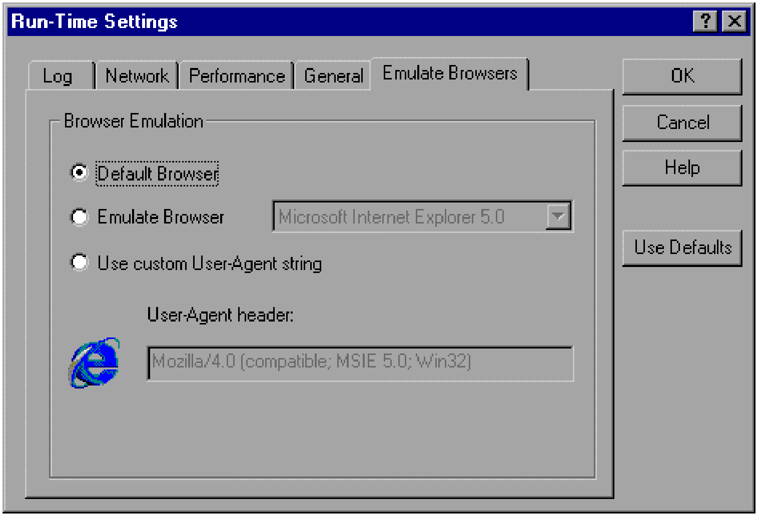
The General tab includes the following options:

Option	Description
Enable checks during script execution	Instructs Astra LoadTest to perform checks during script execution.
Enable automatic transactions	Instructs Astra LoadTest to measure each browsed page as a transaction.
Save snapshot on error	Instructs Astra LoadTest to save an image of a Web page where an error is detected.

Find
Find Again
Top of Chapter
Back

Emulate Browsers Settings

The Emulate Browsers tab enables Astra LoadTest to emulate specific browsers during the playback of the scenario.



Browser Emulation

● Default Browser

○ Emulate Browser   Microsoft Internet Explorer 5.0

○ Use custom User-Agent string

User-Agent header: Mozilla/4.0 (compatible; MSIE 5.0; Win32)

Find
Find Again
Top of Chapter
Back

The Emulate Browsers tab includes the following options:

Option	Description
Default Browser	Instructs Astra LoadTest to emulate the default browser of the host machine during playback.
Emulate Browser	Instructs Astra LoadTest to emulate the specified browser during playback.
Use custom User-Agent string	Instructs Astra LoadTest to use a specified header string during playback.
User-Agent header	Supplies Astra LoadTest with the header string associated with the custom User-Agent.

Find
Find Again
Top of Chapter
Back

Sending Messages to Output

When you run a scenario, the Controller's Output window displays information about test execution. You can include statements in a Vuser test to send error and notification messages to the Controller. The Controller displays these messages in the Output window. For example, you could insert a message that displays the current state of the Web application. You can also save these messages to a file.

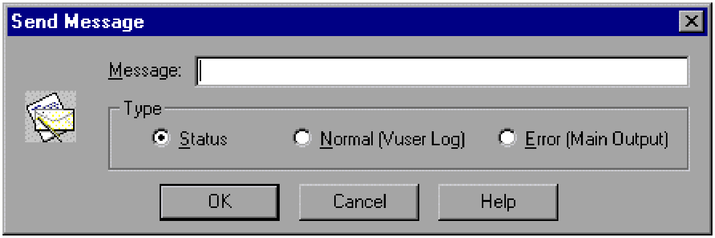
Note: Do not send messages from within a transaction. Doing so lengthens the transaction execution time and may skew the actual transaction results.

Find
Find Again
Top of Chapter
Back

To send messages to the output window:

1 Click a step in your test where you want to send a message.

2 Click the **Send Message** button or choose **Insert > Send Message.** The **Send Message** dialog box opens.



3 Type the message into the **Message box**.

Find
Find Again
Top of Chapter
Back

4 Specify the message type.

The message type includes the following options:

Option	Description
Status	Instructs Astra LoadTest to generate and print formatted output to the Controller Vuser status area.
Normal (Vuser Log)	Instructs Astra LoadTest to send the output message directly to a file.
Error (Main Output)	Instructs Astra LoadTest to send an error message to the Output window.

5 Click **OK** to close the dialog box. The **Send Message** icon is added above the highlighted step.

Find
Find Again
Top of Chapter
Back

Parameterizing Tests

Astra LoadTest enables you to expand the scope of a basic test by replacing fixed values with parameters. This process, known as *parameterization*, greatly increases the power and flexibility of your tests.

This chapter describes:

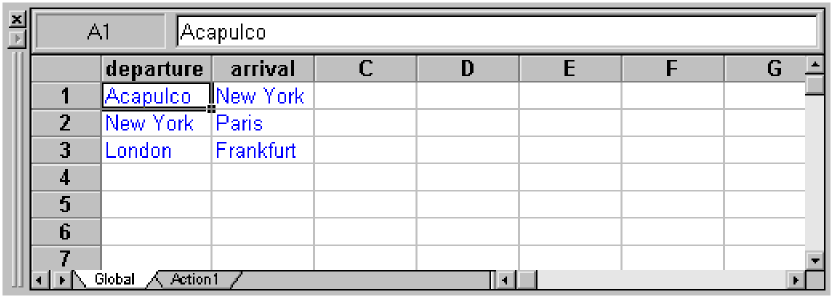
Find
Find Again
Top of Chapter
Back

About Parameterizing Tests

You can use the Virtual User Recorder to enhance your tests by parameterizing values in the test. A *parameter* is a variable that is assigned a value from outside the test in which it is defined.

You start by recording a test that performs a set of actions. After you finish recording, you can parameterize certain constants in the test so that the test will run the same set of actions many times. In each repetition, or *iteration*, Astra LoadTest substitutes the constant value with a parameter value. You supply the list of possible values for a parameter in a table in the Data pane.

Find
Find Again
Top of Chapter
Back



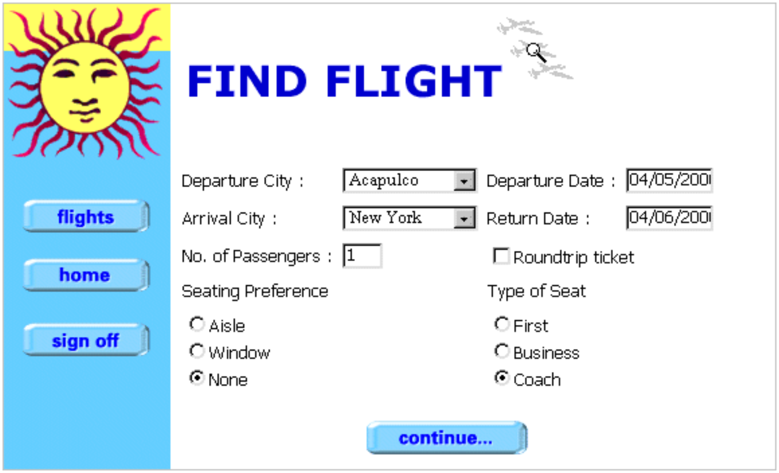
A1 Acapulco

	departure	arrival	C	D	E	F	G
1	Acapulco	New York					
2	New York	Paris					
3	London	Frankfurt					
4							
5							
6							
7							

Each *column* in the table represents the list of values for a single parameter. The column header is the parameter name.

Each *row* in the table represents a set of values that Astra LoadTest submits for all the parameters during a single iteration of the test. When you run your test, Astra LoadTest runs one iteration of the test for each set of data in the table. Thus, a test with ten-rows in the data table will run ten times.

For example, consider the sample Web flight site, “Mercury Tours,” which enables you to book flight requests. To book a flight, you supply the flight itinerary and click the Continue button.



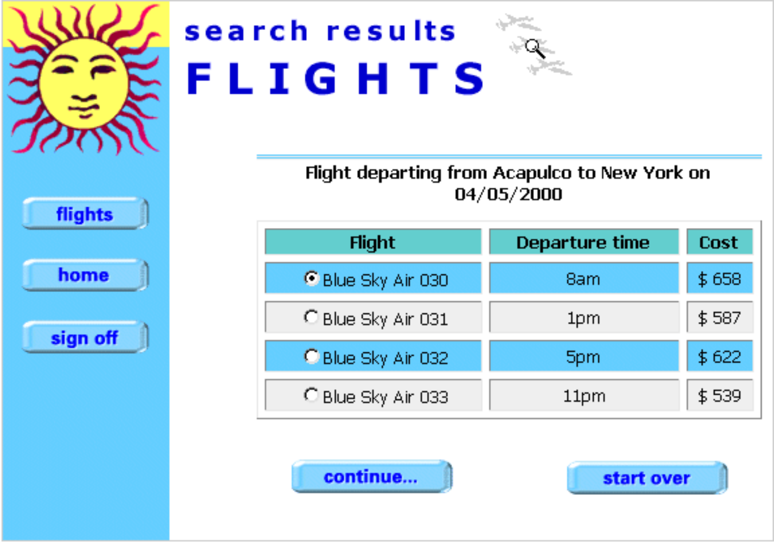
Departure City : Acapulco
Departure Date : 04/05/200
Arrival City : New York
Return Date : 04/06/200
No. of Passengers : 1
□ Roundtrip ticket

Seating Preference
○ Aisle
○ Window
● None

Type of Seat
○ First
○ Business
● Coach

Find
Find Again
Top of Chapter
Back

The site returns the available flights for the requested itinerary.

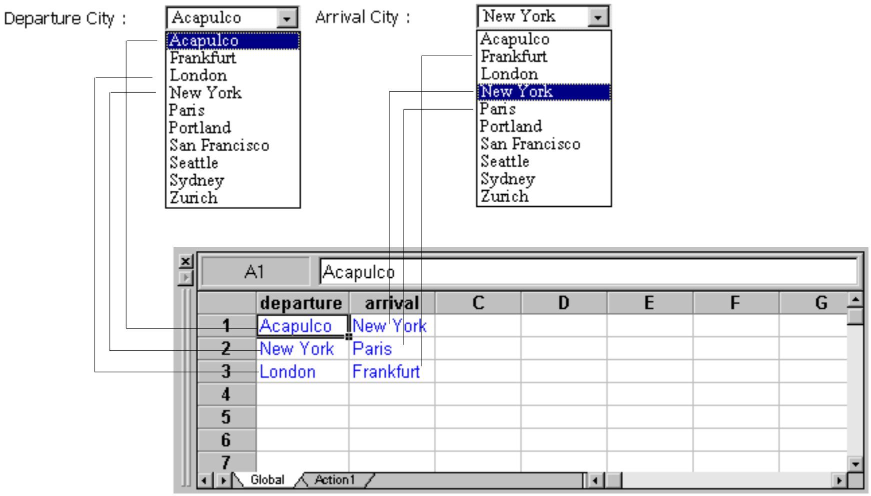


Flight departing from Acapulco to New York on 04/05/2000

Flight	Departure time	Cost
● Blue Sky Air 030	8am	$ 658
○ Blue Sky Air 031	1pm	$ 587
○ Blue Sky Air 032	5pm	$ 622
○ Blue Sky Air 033	11pm	$ 539

You could conduct the test by accessing the Web site and recording the submission of numerous queries. This is a slow, laborious, and inefficient solution. When you parameterize your test, you first record a test that accesses the Web site and checks for the available flights for one requested itinerary. You then substitute the recorded itinerary with a parameter, and add multiple sets of data, one for each itinerary, into the table linked to the test. When you run the test, Astra LoadTest submits a separate query for each itinerary.

Find
Find Again
Top of Chapter
Back



	departure	arrival	C	D	E	F	G
1	Acapulco	New York					
2	New York	Paris					
3	London	Frankfurt					
4							
5							
6							
7							

When you add parameters to your test, you can parameterize a step recorded in your test or a checkpoint added to your test. When you parameterize a step, you parameterize either the *object* that you navigate in your Web page or the *method* by which you navigate. When you parameterize a checkpoint, instead of checking how your Web site performs an operation on a single text string or object, you can check how it performs with multiple sets.

Find
Find Again
Top of Chapter
Back

You can also parameterize your test by creating output parameters, which retrieve variables from the test while it runs and insert them into a table in the Data pane so that you can use them as input later in the test. For more information, see Chapter 8, **Creating Output Parameters**.

Note: After running, you can view the results of the values used in a parameterized test in the Runtime Data table. For more information, see **Viewing the Runtime Data Table for a Parameterized Test** on page 258.

Find
Find Again
◀ ▶
Top of Chapter
Back

Parameterizing Steps

You can parameterize a step while recording your test or afterward. You parameterize a step in your test tree. A step is made up of an *object* that you navigate in your Web page, and/or a *method* by which you navigate the step. When you parameterize a step, you are actually parameterizing either the object or the method. For an example of a parameterized step, see **Example of a Parameterized Test** on page 131.

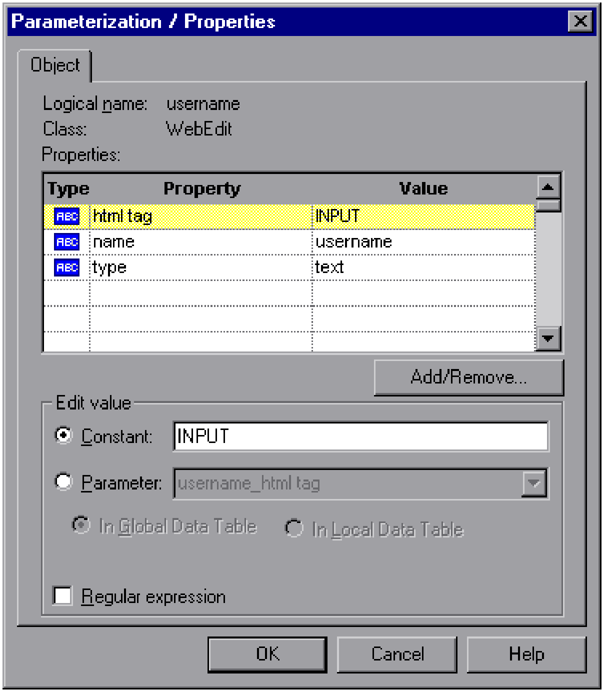
Parameterizing an Object in a Step

You can parameterize the object that you navigate in a step. For example, your Web site may include a form in which the user can choose to click one of several radio buttons. You may want to test how your site responds when different radio buttons are selected. Rather than record a separate test for clicking each radio button, you can parameterize your test so that during each iteration of the test run, Astra LoadTest clicks a different radio button.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

To parameterize the object in a step:

1 Right-click a step in the test tree and choose **Object Properties**. The **Parameterization/Properties** dialog box opens and displays the properties of the object in the step.



Parameterization / Properties

Object

Logical name: username
Class: WebEdit
Properties:

Type	Property	Value
ABC	html tag	INPUT
ABC	name	username
ABC	type	text

Edit value

● Constant: INPUT

○ Parameter: username_html tag

● In Global Data Table ○ In Local Data Table

□ Regular expression

Find
Find Again
Top of Chapter
Back

The Object tab displays information about the object in the step:

Information	Description
Logical name	The logical name of the object.
Class	The type of object. In this example, the "WebEdit" class indicates that the object is an edit box.

The dialog box displays the default properties you can parameterize, in a pane listing the properties, their values, and their types:

Pane Element	Description
Type	The ABC icon indicates that the property value is a constant. The ▦ icon indicates that the property value is a parameter.
Property	The name of the property whose value will be parameterized.
Value	The value of the property to parameterize.
Add/Remove Properties	Opens the **Add/Remove Properties** dialog box, to enable you to modify the list of properties that you can parameterize. To add/remove a property, select/clear a check box and click **OK**. To set the default, click the **Default** button and click **OK**.

2 Click the property to parameterize in the **Properties** section. The property is highlighted.

3 In the **Edit value** section, click **Parameter**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

4 In the **Parameter** box, choose a parameter from the list or enter a new name.

- To use a parameter that you already created, select it from the list.
- To create a new parameter, either use the default parameter name or enter a descriptive name for the parameter.

5 Select **Global** or **Local**.

- To add the parameter to the Global tab in the Data pane, click **Global**.
- To add the parameter to the Action tab, click **Local**.

  For more information, see Chapter 11, **Working with Actions**.

6 If you want to set the property value of the step as a regular expression, select the **Regular expression** check box. For more information, see Chapter 10, **Using Regular Expressions**.

7 Click **Close** to save the parameter and close the dialog box.

8 If you created a new parameter, the **Astra Parameters** dialog box prompts you to add the new parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new parameter.

In your test tree, the P icon next to the step indicates that the step has been parameterized.

Note: You can specify additional data values for the parameter by entering them directly into the table in the Data pane. For more information, see Chapter 9, **Working with Data Tables**.

Find
Find Again
◀ ▶
Top of Chapter
Back

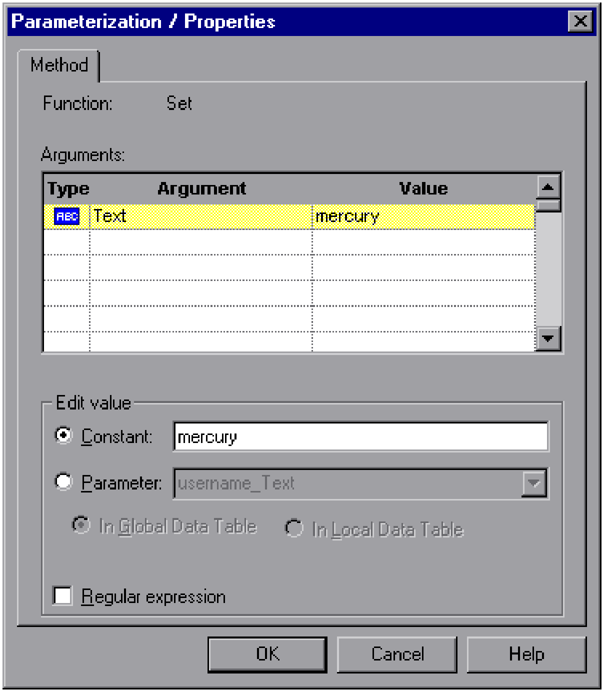
Parameterizing a Method in a Step

You can parameterize the method you use to navigate a step. For example, your Web site may include a form with an edit field into which the user types a text string. You may want to test how your site responds to different data in the form. Rather than record a separate test for each text string typed, you can parameterize your test so that during each iteration of the test run, Astra LoadTest enters a different text string into the edit field.

Find
Find Again
Top of Chapter
Back

To parameterize the method in a step:

1 Right-click a step in the test tree and choose **Function Arguments**. The **Parameterization/Properties** dialog box opens and displays the method arguments in the step.



Parameterization / Properties

Method

Function: Set

Arguments:

Type	Argument	Value
ABC	Text	mercury

Edit value

● Constant: mercury

○ Parameter: username_Text

● In Global Data Table ○ In Local Data Table

□ Regular expression

Find
Find Again
Top of Chapter
Back

The Method tab displays the name of the function performed in the step:

Information	Description
Function	The name of the function performed.

The dialog box displays the default arguments you can parameterize, in a pane listing the arguments, their values, and their types:

Pane Element	Description
Type	The ABC icon indicates that the argument value is a constant. The ▦ icon indicates that the argument value is a parameter.
Argument	The name of the argument whose value will be parameterized.
Value	The value of the argument to parameterize.

2 Click an argument in the **Arguments** section. The argument is highlighted.

3 In the **Edit value** section, click **Parameter**.

4 In the **Parameter** box, choose a parameter from the list or enter a new name.

- To use a parameter that you already created, select it from the list.
- To create a new parameter, either use the default parameter name or enter a descriptive name for the parameter.

Find
Find Again
◀ ▶
Top of Chapter
Back

5 Select **Global** or **Local**.

- To add the parameter to the Global tab in the Data pane, click **Global**.
- To add the parameter to the Action tab, click **Local**.

  For more information, see Chapter 11, **Working with Actions**.

6 If you want to set the argument value of the step as a regular expression, select the **Regular expression** check box. For more information, see Chapter 10, **Using Regular Expressions**.

7 Click **Close** to save the parameter and close the dialog box.

8 If you created a new parameter, the **Astra Parameters** dialog box prompts you to add the new parameter to a table in the Data pane. Click **OK**. A new column is highlighted in the table for the new parameter.

In your test tree, the P icon next to the step indicates that the step has been parameterized.

Note: You can specify additional data values for the parameter by entering them directly into the table in the Data pane. For more information, see Chapter 9, **Working with Data Tables**. .

Find
Find Again
◀ ▶
Top of Chapter
Back

Parameterizing Checkpoints

You can parameterize a checkpoint while recording your test or afterward. For information on parameterizing checkpoints while creating them, see Chapter 4, **Creating Checkpoints**.

When you test your Web site, you may want to check how it performs the same operations with multiple sets of data. For example, if you are testing the sample flight Web site, “Mercury Tours,” you may create a checkpoint to check that once you book a ticket, it is booked correctly. Suppose that you want to check that flights are booked correctly for a variety of different destinations. Rather than create a separate test with a separate checkpoint for each destination, you can parameterize the destination information: for each iteration of the test, Astra LoadTest checks the flight information for a different destination. For an example of a parameterized checkpoint, see **Example of a Parameterized Test** on page 131.

To parameterize a checkpoint either while recording your test or afterward:

1 Right-click a checkpoint in the test tree and choose **Function Arguments**.

- For a page checkpoint, the **Page Checkpoint Properties** dialog box opens.
- For a text checkpoint, the **Text Checkpoint Properties** dialog box opens.
- For an object checkpoint, the **Checkpoint Properties** dialog box opens.
- For a table checkpoint, the **Table Checkpoint Properties** dialog box opens.

Find
Find Again
Top of Chapter
Back

2. In the dialog box, click **Parameter** to set the value as a parameter.
3. In the **Parameter** box, choose a parameter from the list or enter a new name.
   - To use a parameter that you already created, select it from the list.
   - To create a new parameter, either use the default parameter name or enter a descriptive name for the parameter.

   You can create the parameter in the global or a local data table. For additional information, see Chapter 9, **Working with Data Tables**.
4. To use a regular expression with the parameter, select the **Regular expression** check box. For additional information, see Chapter 10, **Using Regular Expressions**.
5. Click **OK** to save the parameter and close the dialog box.
6. If you created a new parameter, the **Astra Parameters** dialog box prompts you to add the new parameter to the data. Click **OK**. A new column is highlighted in the table for the new parameter.

   In your test tree, the 𝒫 icon next to the checkpoint indicates that the checkpoint has been parameterized.

Note: You can specify additional data values for the parameter by entering them directly into the table in the Data pane. For more information, see Chapter 9, **Working with Data Tables**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

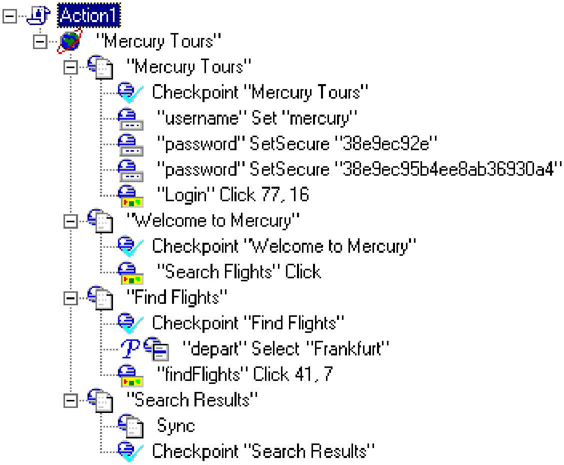
Example of a Parameterized Test

The following example shows how to parameterize a step object, step method, and checkpoint.

When you test your Web site, you may want to check how it performs the same operations with multiple sets of data. For example, if you are testing the sample flight Web site, “Mercury Tours,” you may want to check that the correct departure and the arrival cities are selected before you book a particular flight. Suppose that you want to check that the flights are booked correctly for a variety of different locations. Rather than create a separate test with a separate checkpoint for each location, you can parameterize the location information: for each iteration of the test, Astra LoadTest checks the flight information for a different locations.

Find
Find Again
◀ ▶
Top of Chapter
Back

The following is a sample test of a flight booking procedure. The departure city is "Frankfurt" the arrival city is "Acapulco".



```
Action1
  "Mercury Tours"
    "Mercury Tours"
      Checkpoint "Mercury Tours"
      "username" Set "mercury"
      "password" SetSecure "38e9ec92e"
      "password" SetSecure "38e9ec95b4ee8ab36930a4"
      "Login" Click 77, 16
    "Welcome to Mercury"
      Checkpoint "Welcome to Mercury"
      "Search Flights" Click
    "Find Flights"
      Checkpoint "Find Flights"
      "depart" Select "Frankfurt"
      "findFlights" Click 41, 7
    "Search Results"
      Sync
      Checkpoint "Search Results"
```

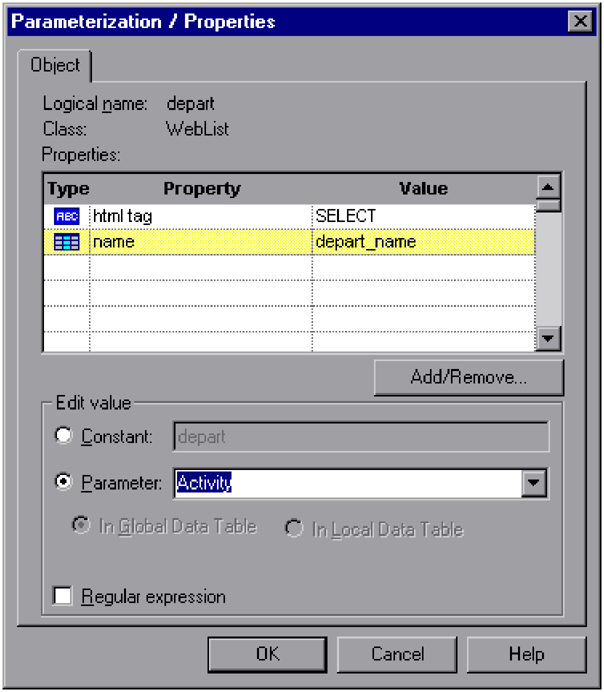
Find
Find Again
Top of Chapter
Back

Parameterize a Step

Parameterize the object and the method of the following step:

"depart" Select "Frankfurt"

In the **Parameterization/Properties** dialog box, select the “name” property. In the Parameter box, rename the “depart_name” to Activity. Close the dialog box. The Activity row is added to the data table. Note that the Activity row will be used for both departures and arrivals.



Parameterization / Properties

Object

Logical name: depart
Class: WebList
Properties:

Type	Property	Value
ABC	html tag	SELECT
▦	name	depart_name

Add/Remove...

Edit value

○ Constant: depart

● Parameter: Activity

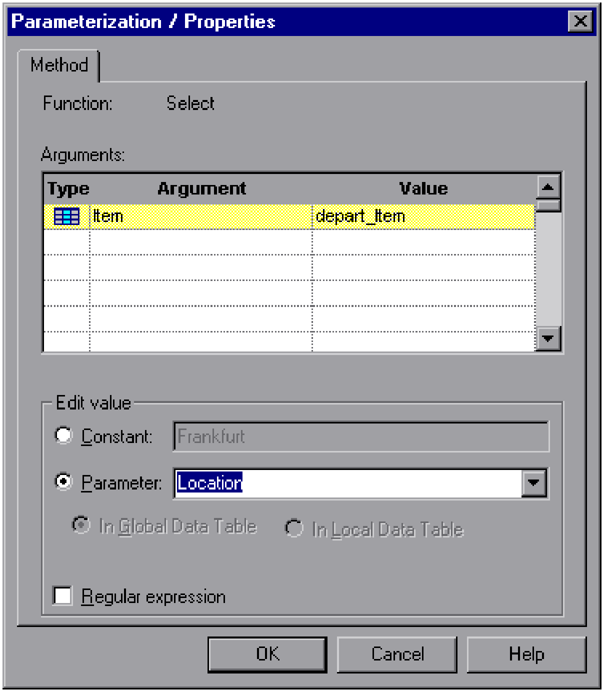
● In Global Data Table ○ In Local Data Table

□ Regular expression

OK | Cancel | Help

Find
Find Again
Top of Chapter
Back

In the following example, parameterize the method of the above step. In the **Parameterization/Properties** dialog box, select the “item” property. In the Parameter box, rename the “depart_item” to `Location`. Close the dialog box. The Location row is added to the data table. Note that the Location row will be used for both departure cities and arrival cities.



Parameterization / Properties

Method

Function: Select

Arguments:

Type	Argument	Value
▦	Item	depart_Item

Edit value

○ Constant: Frankfurt

● Parameter: Location

● In Global Data Table ○ In Local Data Table

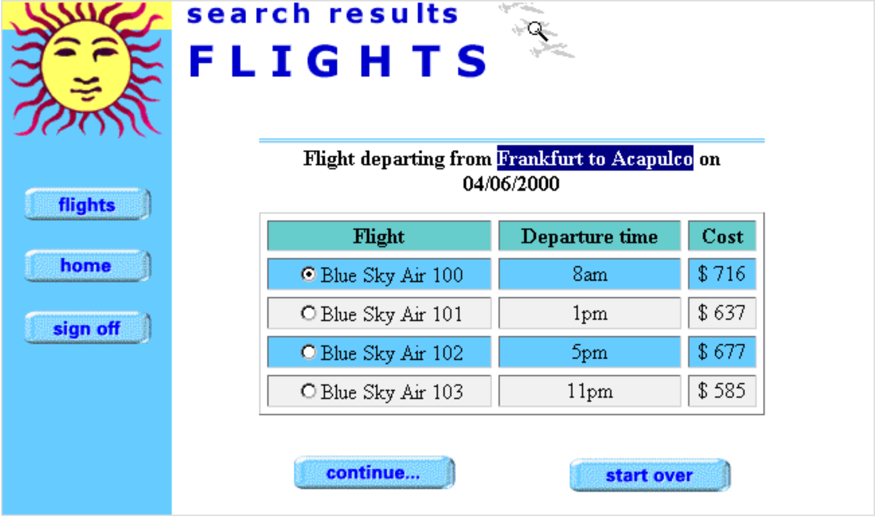
□ Regular expression

For more information on parameterizing a step, see **Parameterizing Steps** on page 121.

Find
Find Again
Top of Chapter
Back

Parameterized Checkpoint

In the following example, a parameterized text checkpoint is added to check that the correct locations were selected before you book a flight. A text checkpoint is created for the following text:

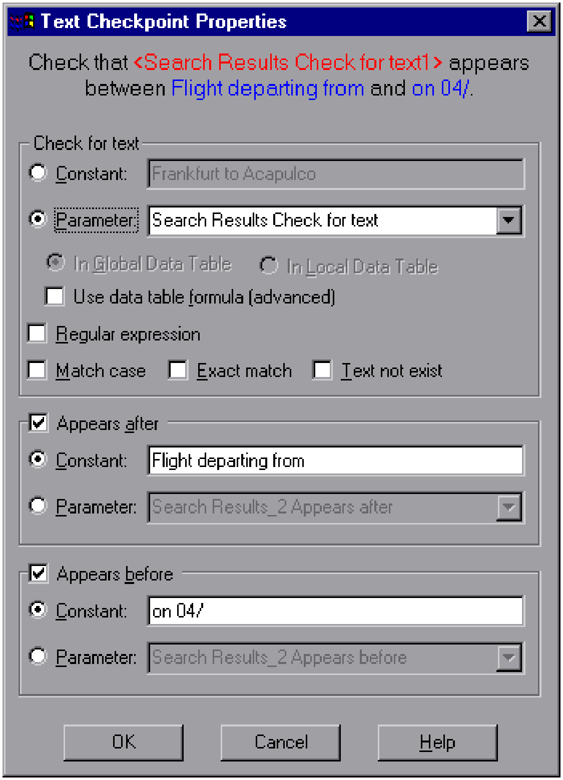


Flight departing from Frankfurt to Acapulco on 04/06/2000

Flight	Departure time	Cost
● Blue Sky Air 100	8am	$ 716
○ Blue Sky Air 101	1pm	$ 637
○ Blue Sky Air 102	5pm	$ 677
○ Blue Sky Air 103	11pm	$ 585

Find
Find Again
Top of Chapter
Back

In the **Text Checkpoint Properties** dialog box, the text is parameterized. Close the dialog box. A text checkpoint parameter row is added to the data table.



Text Checkpoint Properties

Check that <Search Results Check for text1> appears between Flight departing from and on 04/.

Check for text

○ Constant: Frankfurt to Acapulco

● Parameter: Search Results Check for text

● In Global Data Table ○ In Local Data Table

□ Use data table formula (advanced)

□ Regular expression

□ Match case □ Exact match □ Text not exist

☑ Appears after

● Constant: Flight departing from

○ Parameter: Search Results_2 Appears after

☑ Appears before

● Constant: on 04/

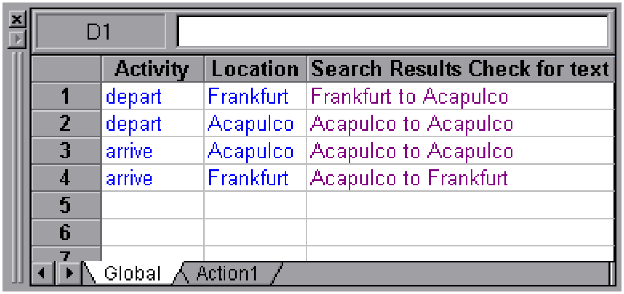
○ Parameter: Search Results_2 Appears before

For more information on parameterizing a checkpoint, see **Parameterizing Checkpoints** on page 129.

Find
Find Again
Top of Chapter
Back

Enter Data in the Data Table

Complete the table in the Data pane. The data table may appear as follows:



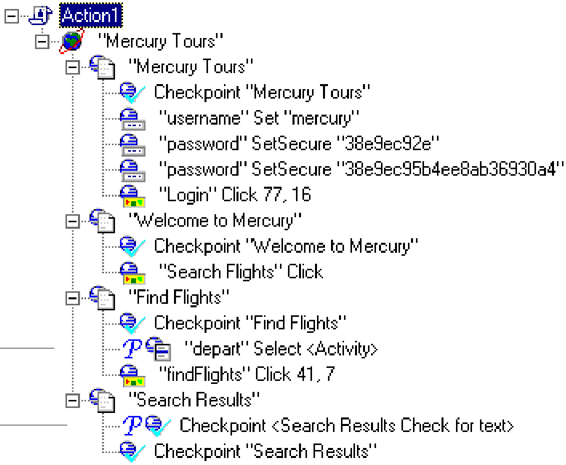
	Activity	Location	Search Results Check for text
1	depart	Frankfurt	Frankfurt to Acapulco
2	depart	Acapulco	Acapulco to Acapulco
3	arrive	Acapulco	Acapulco to Acapulco
4	arrive	Frankfurt	Acapulco to Frankfurt
5			
6			
7			

For more information on data tables, see Chapter 9, **Working with Data Tables**.

Find
Find Again
Top of Chapter
Back

Modified Test

The following example shows the test after parameterizing the step and creating a parameterized text checkpoint.



```
Action1
  "Mercury Tours"
    "Mercury Tours"
      Checkpoint "Mercury Tours"
      "username" Set "mercury"
      "password" SetSecure "38e9ec92e"
      "password" SetSecure "38e9ec95b4ee8ab36930a4"
      "Login" Click 77, 16
    "Welcome to Mercury"
      Checkpoint "Welcome to Mercury"
      "Search Flights" Click
    "Find Flights"
      Checkpoint "Find Flights"
      P "depart" Select <Activity>
      "findFlights" Click 41, 7
    "Search Results"
      P Checkpoint <Search Results Check for text>
      Checkpoint "Search Results"
```

Parameterized step

Parameterized text checkpoint

Find
Find Again
Top of Chapter
Back

Creating Output Parameters

Astra LoadTest enables you to parameterize your test by retrieving a variable value from your test and entering it in your table in the Data pane, as an output parameter. You can subsequently use this output parameter as an input variable in your test. This enables you to use data retrieved during a test in other parts of the test.

This chapter describes:

Find

Find Again

◀ ▶

Top of Chapter

Back

About Creating Output Parameters

You parameterize your test by adding values to a table in the Data pane that replace variables in the test. When you run the test, Astra LoadTest runs one iteration of the test for each set of values from your table, as discussed in Chapter 7, **Parameterizing Tests**.

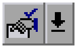
You can also use output parameters to parameterize your test. An *output parameter* is a value that is retrieved from a parameter in your test while the test runs and is entered into your table.

For example, consider a flight reservation site. You design a test to create a new reservation and then view the reservation details. Every time you run the test, the site generates a unique order number for the new reservation. To view the reservation, the site requires the user to input the same order number. You cannot know the order number before you run the test, however.

To solve this problem, you create an output parameter for the unique order number that the site generates when creating a new reservation. In the view reservation screen, you parameterize the order number input field. You use the same output parameter of the unique order number.

When you run the test, Astra LoadTest retrieves the unique order number generated by the site for the new reservation and inserts it in the table for the order number output parameter. When the test reaches the order number input field required to view the reservation, Astra LoadTest uses the unique order number stored in the table for the order number input field parameter.

Find
Find Again
◀ ▶
Top of Chapter
Back



You can add an output parameter by using commands on the Insert menu or by clicking the arrow beside the Insert Checkpoint button on the Main toolbar. This displays a menu of output parameter options that are relevant to the selected step in the test tree.

Note: After running, you can view the output parameters retrieved during a test run in the Runtime Data table. For more information, see **Viewing the Runtime Data Table for a Parameterized Test** on page 258.

Find
Find Again
Top of Chapter
Back

Creating Page Output Parameters

When you create a page output parameter, you parameterize a constant page property by replacing it with a variable. When you run the test, Astra LoadTest retrieves the value in the output parameter and inserts it in the data table. For example, the number of links on a page may vary based on the selections a user makes on a form on the previous page. You could make an output parameter to return the number of links on the page during each run. To parameterize a page property, you open the Page Output Parameter Properties dialog box.

Find
Find Again
◀ ▶
Top of Chapter
Back

Adding a Page Output Parameter

You can create a page output parameter while recording your test or afterward.

To create a page output parameter while recording:

1 Choose **Insert > Output Parameter**.

The mouse pointer turns into a pointing hand.

2 Click the page to parameterize. The **Object Selection - Output Parameter Properties** dialog box opens.

3 Select the page item and click **OK**. The **Page Output Parameter Properties** dialog box opens.

4 Specify the settings for the output parameter. For more information, see **Understanding the Page Output Parameter Properties Dialog Box** on page 144.

5 Click **OK** to close the Page Output Parameter Properties dialog box.

6 If you created a new output parameter, the **Astra Parameter** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item is added to your test tree.

To create a page output parameter after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Click a page step in your test tree where you want to add an output parameter. The ActiveScreen displays the Web page corresponding to the highlighted step.

3 Right-click the page to parameterize in the ActiveScreen and choose **Output**. The **Object Selection - Output Parameter Properties** dialog box opens.

4 Select the page item and click **OK**. The **Page Output Parameter Properties** dialog box opens.

5 Specify the settings for the output parameter. For more information, see **Understanding the Page Output Parameter Properties Dialog Box** on page 144.

6 Click **OK** to close the Page Output Parameter Properties dialog box.

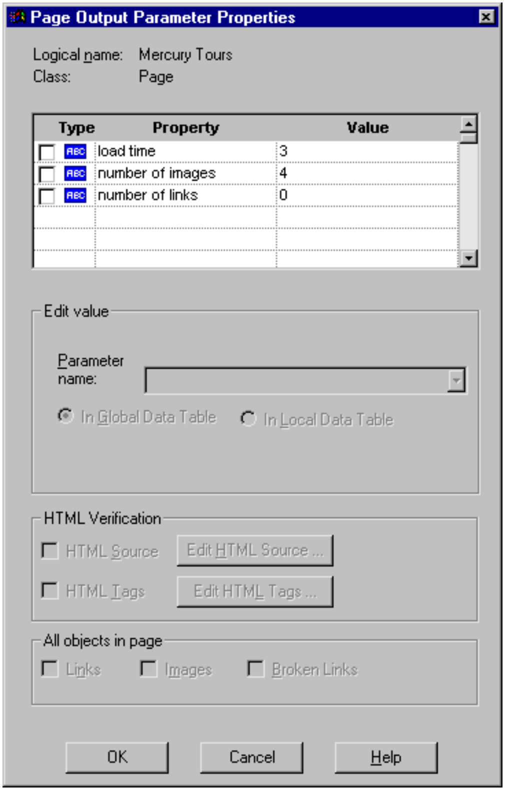
7 If you created a new output parameter, the **Astra Parameter** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item is added to your test tree.

Find
Find Again
Top of Chapter
Back

Understanding the Page Output Parameter Properties Dialog Box

In the Page Output Parameter Properties dialog box, you can specify which property of the page to parameterize, and edit the values of this property.



Page Output Parameter Properties

Logical name: Mercury Tours
Class: Page

Type	Property	Value
□ ABC	load time	3
□ ABC	number of images	4
□ ABC	number of links	0

Edit value

Parameter name:

● In Global Data Table ○ In Local Data Table

HTML Verification

□ HTML Source
□ HTML Tags

All objects in page

□ Links □ Images □ Broken Links

Find
Find Again
Top of Chapter
Back

Identifying the Page

The top part of the dialog box displays information about the page to parameterize:

Information	Description
Logical name	The name of the page as defined in the HTML code of the Web page.
Class	The type of object.

Choosing which Property to Parameterize

The dialog box also displays the page properties that you can parameterize, in a pane listing the properties, their values, and their types:

Pane Element	Description
Check box	To parameterize a property, select the corresponding check box.
Type	The ABC icon indicates that the value of the property is a constant. The ▦ icon indicates that the value of the property is a parameter.
Property	The name of the property to check.
Value	The value of the property to parameterize.

Find
Find Again
◀ ▶
Top of Chapter
Back

Choosing an Output Parameter

In the **Edit value** section, you use the following options to specify the output parameter name:

Option	Description
Parameter name	Specifies the output parameter name. To use an output parameter that you already created, select an output parameter from the list. To create a new output parameter, you can use the default output parameter name, or type a descriptive name for the output parameter.
Global (default)	Adds the output parameter name to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds the output parameter name to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Creating Text Output Parameters

When you create a text output parameter, you parameterize a constant text string by replacing it with a variable. To parameterize a text string, you open the Text Output Parameter Properties dialog box.

Adding a Text Output Parameter

You can create a text output parameter while recording your test or afterward.

To create a text output parameter while recording:

1 Highlight the text string you want to parameterize.

2 Choose **Insert > Text Output Parameter**.

The mouse pointer turns into a pointing hand.

3 Click the text string to parameterize. The **Text Output Parameter Properties** dialog box opens.

4 Specify the settings for the output parameter. For more information, see **Understanding the Text Output Parameter Properties Dialog Box** on page 149.

5 Click **OK** to close the Text Output Parameter Properties dialog box.

6 If you created a new output parameter, the **Astra Parameter** dialog box prompts you to add the new output parameter to the table in Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item is added to your test tree.

Find
Find Again
Top of Chapter
Back

To create an output parameter after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Click a step in your test where you want to create an output parameter.

The ActiveScreen displays the Web page corresponding to a highlighted step.

3 Highlight and then right-click the text string to parameterize in the ActiveScreen.

4 Choose **Insert Text Output**. The **Text Output Parameter Properties** dialog box opens.

5 Specify the settings for the output parameter. For more information, see **Understanding the Text Output Parameter Properties Dialog Box** on page 149.

6 Click **OK** to close the Text Output Parameter Properties dialog box.

7 If you created a new output parameter, the **Astra Parameter** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

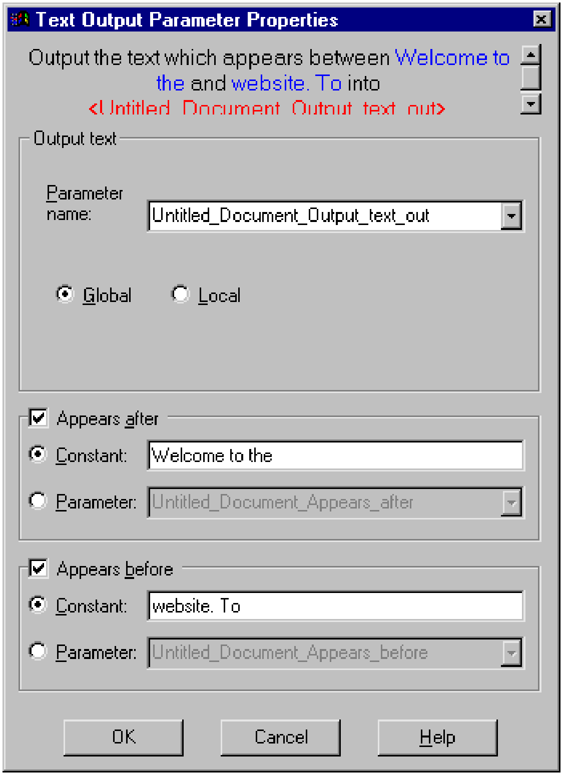
An output parameter tree item P is added to your test tree.

Find
Find Again
◀ ▶
Top of Chapter
Back

Understanding the Text Output Parameter Properties Dialog Box

In the Text Output Parameter Properties dialog box, you can specify which text to parameterize as well as which text appears before and after the parameter. This is particularly helpful when the text string you want to parameterize appears several times in the same Web page. For example, suppose you want to parameterize the “Mercury Tours” text string in a specific location in the first page of the sample Mercury Tours Web site. This text string actually appears three times on that Web page. To parameterize the text string in a specific location, you can specify which text precedes and/or follows the text string you are parameterizing.

Find
Find Again
Top of Chapter
Back



Text Output Parameter Properties

Output the text which appears between Welcome to the and website. To into <Untitled_Document_Output_text_out>

Output text

Parameter name: Untitled_Document_Output_text_out

● Global ○ Local

☑ Appears after

● Constant: Welcome to the

○ Parameter: Untitled_Document_Appears_after

☑ Appears before

● Constant: website. To

○ Parameter: Untitled_Document_Appears_before

Find
Find Again
Top of Chapter
Back

Specifying which Text to Parameterize

In the **Output text** section, you use the following options to specify the output parameter name for the highlighted text string:

Option	Description
Parameter name	Specifies the output parameter name. To use an output parameter that you already created, select an output parameter from the list. To create a new output parameter, you can use the default output parameter name, or type a descriptive name for the output parameter.
Global (default)	Adds the output parameter name to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds the output parameter name to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
Top of Chapter
Back

Specifying What Appears After the Text to Parameterize

In the **Appears after** section, you use the following options to specify which text, if any, should appear after the text output parameter:

Option	Description
Appears after (default)	Specifies that the output parameter text appears after the text in this box. To ignore the text that appears after the parameter, clear this check box.
Constant (default)	Displays the text that appears after the parameter.
Parameter	Sets the text string as a parameter.

Specifying What Appears Before the Text to Parameterize

In the **Appears before** section, you use the following options to specify which text, if any, should appear before the text output parameter:

Option	Description
Appears before (default)	Specifies that the output parameter text appears before the text in this box. To ignore the text that appears before the parameter, clear this check box.
Constant (default)	Displays the text that appears before the parameter.
Parameter	Sets the text string as a parameter.

Find
Find Again
◀ ▶
Top of Chapter
Back

Creating Object Output Parameters

You can parameterize an object on your Web page to create an object output parameter. To parameterize an object, you open the Object Output Parameter dialog box.

Adding an Object Output Parameter

You can create an object output parameter while recording your test or afterward.

To create an object output parameter while recording:

1 Choose **Insert > Output Parameter**.

The mouse pointer turns into a pointing hand.

2 Click the object to parameterize. The **Object Selection - Output Parameter Properties** dialog box opens.

3 Select the last tree item. The tree item name depends on the object's class, for example:

Object	Class
Check box	WebCheckBox
Edit box	WebEdit
Image	Image
Radio button	WebRadio

Find
Find Again
Top of Chapter
Back

4 Click **OK**. The **Output Parameter Properties** dialog box opens.

5 Specify the settings for the output parameter. For more information, see **Understanding the Output Parameter Properties Dialog Box** on page 156.

6 Click **OK** to close the Output Parameter Properties dialog box.

7 If you created a new output parameter, the **Astra parameters** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item is added to your test tree.

To create an object output parameter after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Click a step in your test where you want to create an output parameter.

The ActiveScreen displays the Web page corresponding to a highlighted step.

3 Right-click the object to parameterize in the ActiveScreen and choose **Output**. The **Object Selection - Output Parameter Properties** dialog box opens.

4 Select the last tree item. The tree item name depends on the object’s class, for example:

Object	Class
Check box	WebCheckBox
Edit box	WebEdit
Image	Image
Radio button	WebRadio

Find
Find Again
Top of Chapter
Back

5 Click **OK**. The **Output Parameter Properties** dialog box opens.

6 Specify the settings for the output parameter. For more information, see **Understanding the Output Parameter Properties Dialog Box** on page 156.

7 Click **OK** to close the Output Parameter Properties dialog box.

8 If you created a new output parameter, the **Astra parameters** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item P is added to your test tree.

Find

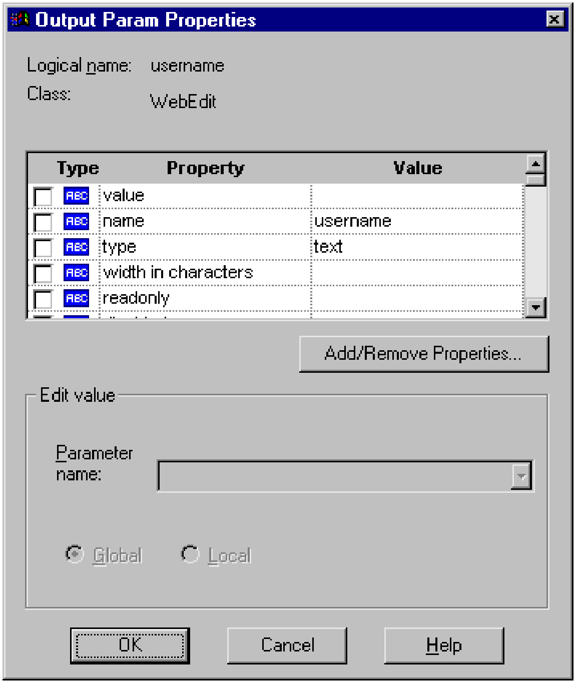
Find Again

Top of Chapter

Back

Understanding the Output Parameter Properties Dialog Box

In the Output Parameter Properties dialog box, you can specify which property of the object to parameterize, and edit the values of this property.



Output Param Properties

Logical name: username

Class: WebEdit

Type	Property	Value
□ ABC	value	
□ ABC	name	username
□ ABC	type	text
□ ABC	width in characters	
□ ABC	readonly	

Find
Find Again
Top of Chapter
Back

Identifying the Object

The top part of the dialog box displays information about the object to parameterize:

Information	Description
Logical name	The name of the object as defined in the HTML code of the Web page.
Class	The type of object. In this example, the "WebEdit" class indicates that the object is an edit field.

Choosing which Property to Parameterize

The dialog box also displays the properties of the object you can parameterize, in a pane listing the properties, their values, and their types:

Pane Element	Description
Check box	To parameterize a property, select the corresponding check box.
Type	The ABC icon indicates that the value of the property is a constant. The ▦ icon indicates that the value of the property is a parameter.
Property	The name of the property to check.
Value	The value of the property to parameterize.

Find
Find Again
Top of Chapter
Back

Choosing an Output Parameter

In the **Edit value** section, you use the following options to specify the output parameter name:

Option	Description
Parameter name	Specifies the output parameter name. To use an output parameter that you already created, select an output parameter from the list. To create a new output parameter, you can use the default output parameter name, or type a descriptive name for the output parameter.
Global (default)	Adds the output parameter name to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds the output parameter name to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
Top of Chapter
Back

Creating Table Output Parameters

You can parameterize a text string in your table to create a table output parameter. To parameterize a text string in a table, you open the Table Output Parameter Properties dialog box.

Find
Find Again
◀ ▶
Top of Chapter
Back

Adding a Table Output Parameter

You can create a table output parameter while recording your test or afterward.

To create a table output parameter while recording:

1. Choose **Insert > Output Parameter**.

   The mouse pointer turns into a pointing hand.

2. Click the table to parameterize. The **Object Selection - Output Parameter Properties** dialog box opens.

3. Select a table item and click **OK**. The **Table Output Parameter Properties** dialog box opens.

4. Specify the settings for the output parameter. For more information, see **Understanding the Table Output Parameter Properties Dialog Box** on page 161.

5. Click **OK** to close the Table Output Parameter Properties dialog box.

6 If you created a new output parameter, the **Astra parameters** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item is added to your test tree.

To create a table output parameter after recording:

1 Make sure the **Display Views** button and the **ActiveScreen** tab are selected.

2 Click a step in your test where you want to create an output parameter.

The ActiveScreen displays the Web page corresponding to a highlighted step.

3 Highlight a text string in a table on the ActiveScreen tab.

4 Right-click the table to parameterize in the ActiveScreen and choose **Output**. The **Object Selection - Output Parameter Properties** dialog box opens.

5 Select a table item and click **OK**. The **Table Output Parameter Properties** dialog box opens.

6 Specify the settings for the output parameter. For more information, see **Understanding the Table Output Parameter Properties Dialog Box** on page 161.

7 Click **OK** to close the Table Output Parameter Properties dialog box.

8 If you created a new output parameter, the **Astra parameters** dialog box prompts you to add the new output parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new output parameter.

An output parameter tree item is added to your test tree.

Find
Find Again
Top of Chapter
Back

Understanding the Table Output Parameter Properties Dialog Box

In the Table Output Parameter Properties dialog box, you can specify the name of the output parameter.

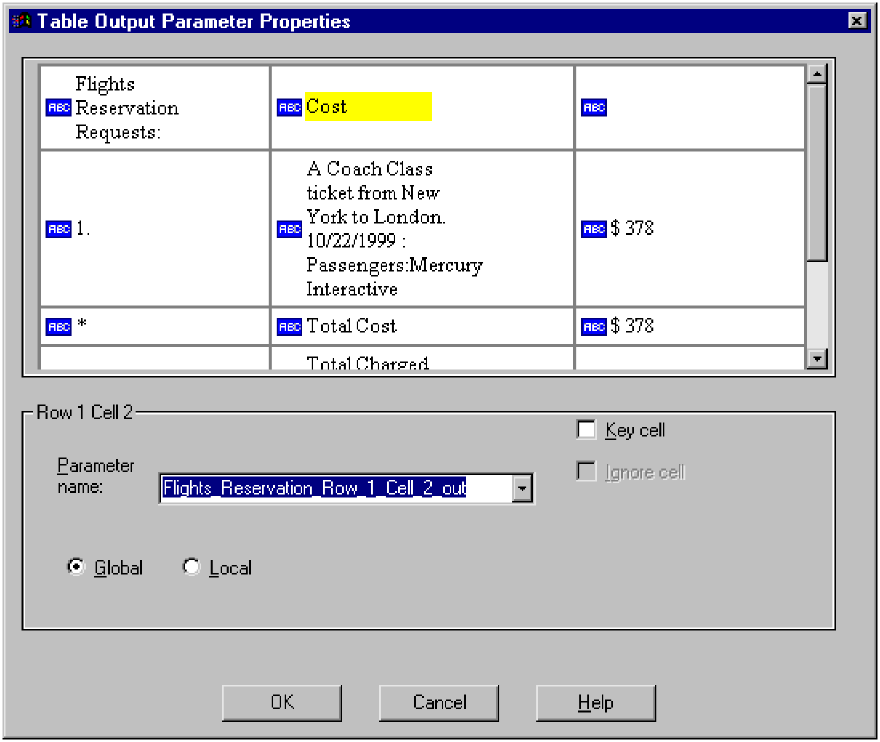


Table Output Parameter Properties

Flights Reservation Requests:	Cost	
1.	A Coach Class ticket from New York to London. 10/22/1999 : Passengers:Mercury Interactive	$ 378
*	Total Cost	$ 378
	Total Charged	

Row 1 Cell 2

Parameter name: Flights_Reservation_Row_1_Cell_2_out

□ Key cell

□ Ignore cell

● Global ○ Local

Find
Find Again
Top of Chapter
Back

The dialog box displays rows and columns of a table. Your highlighted text string appears in a cell.

Use the following options to specify the output parameter name:

Option	Description
Parameter name	Specifies the output parameter name. To use an output parameter that you already created, select an output parameter from the list. To create a new output parameter, you can use the default output parameter name, or type a descriptive name for the output parameter.
Global (default)	Adds the output parameter name to the Global tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.
Local	Adds the output parameter name to the Action tab in the Data pane. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
Top of Chapter
Back

Working with Data Tables

The Virtual User Recorder enables you to create and run tests that are driven by data stored in the data table.

This chapter describes:

Find
Find Again
Top of Chapter
Back

About Working with Data Tables

You can parameterize your test with input and output parameters so that it will run several times on different sets of data. The data your test uses is stored in the data table, which appears in the Data pane at the bottom of the site. The data table has the characteristics of a Microsoft Excel spreadsheet, meaning that you can also execute mathematical formulas within the cells.

After you run a test, the data you enter in the data table is displayed in the Runtime data table within the Test Results window. For more information on the Runtime data table, see **Viewing the Runtime Data Table for a Parameterized Test** on page 258.

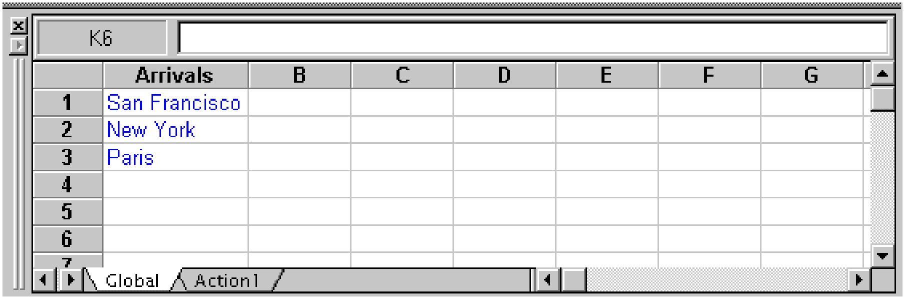
Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Global and Local Sheets

There are two types of sheets within the data table: *Global* and *Local*. You can access the different sheets by clicking the appropriate tabs below the data table.

Global Sheet

The Global sheet contains the data that replaces parameters in each iteration of the test. If you create a Global parameter called Arrivals, the Global sheet might look like this:



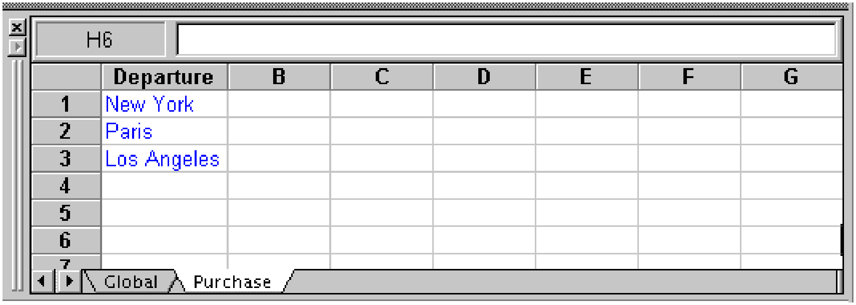
K6

	Arrivals	B	C	D	E	F	G
1	San Francisco						
2	New York						
3	Paris						
4							
5							
6							
7							

Find
Find Again
Top of Chapter
Back

Local Sheets

Each time you add a new action to the test, a new *Local* sheet is added to the data table. Local sheets are automatically labeled with the exact name of the corresponding action. The data contained in a local sheet is relevant for the corresponding action only. For example, if a test had the data table below, the Virtual User Recorder would only use the data contained in the Departure column when running iterations on the *Purchase* action:



	Departure	B	C	D	E	F	G
1	New York						
2	Paris						
3	Los Angeles						
4							
5							
6							
7							

Editing the Data Table

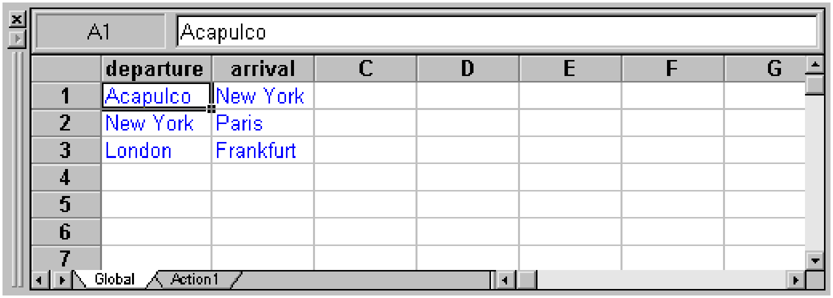
The data table contains the values that Astra LoadTest substitutes for parameters when you run a test. Whenever you save your test, Astra LoadTest automatically saves the test's data table. Astra LoadTest automatically saves the data table for a test in the test folder and assigns it an *.xls* extension.

You can edit information in the table by typing directly into the table. You can also import data in Excel 95, Excel 97, or ASCII format. You use the table in the same way as an Excel spreadsheet, including inserting formulas into cells.

To edit the data table:

1 Open your test.

2 Make sure the **Data Views** button is enabled.



A1 Acapulco

	departure	arrival	C	D	E	F	G
1	Acapulco	New York					
2	New York	Paris					
3	London	Frankfurt					
4							
5							
6							
7							

Find
Find Again
Top of Chapter
Back

- Each *row* in the table represents the set of values that Astra LoadTest submits for the parameterized arguments during a single iteration of the test or action. The number of iterations that a test runs is equal to the number of rows in the table.
- Each *column* in the table represents the list of values for a single parameterized argument. The column header is the parameter name.

Note: You must enter data in rows from top to bottom, i.e., you cannot enter data in a cell in a row until you have entered data in a previous row.

3 To change the name of a column, double-click the column heading cell. The **Change Parameter** dialog box opens. Type a parameter name and click **OK**. If you change the name in the table, you must also change the corresponding parameter name in the test pane.

4 Use the data table menu commands described below to edit the table. To open the data table menu, right-click a cell. The following menus are available: File, Edit, Data, and Format.

Find
Find Again
Top of Chapter
Back

File Menu

The following commands are available in the File menu:

File Command	Description
Import	Imports an existing table file into the table. Note that the new table file replaces all data currently in any sheet of the table.
Export	Saves the table as a file.
Print	Prints the table.

Edit Menu

The following commands are available in the Edit menu:

Edit Command	Description
Cut	Cuts the table selection and puts it on the Clipboard.
Copy	Copies the table selection and puts it on the Clipboard.
Paste	Pastes the contents of the Clipboard to the current table selection.
Paste Values	Pastes values from the Clipboard to the current table selection. Any formatting applied to the values is ignored. In addition, only formula results are pasted; formulas are ignored.

Find
Find Again
Top of Chapter
Back

Edit Command	Description
Clear	Clears formats or contents from the current selection. You can clear only formats, only contents (including formulas), or both formats and contents.
Insert	Inserts empty cells at the location of the current selection. Cells adjacent to the insertion are shifted to make room for the new cells.
Delete	Deletes the current selection. Cells adjacent to the deleted cells are shifted to fill the space left by the vacated cells.
Fill Right	Copies data in the left-most cell of the selected range to all cells to the right of it within the selected range.
Fill Down	Copies data in the top cell of the selected range to all cells below it within the selected range.
Find	Finds a cell containing specified text. You can search by row or column in the table and specify to match case or find entire cells only.
Replace	Finds a cell containing specified text and replaces it with different text. You can search by row or column in the table and specify to match case and/or to find entire cells only. You can also replace all.
Go To	Goes to a specified cell. This cell becomes the active cell.

Find
Find Again
◀ ▶
Top of Chapter
Back

Data Menu

The following commands are available in the Data menu:

Data Command	Description
Recalc	Recalculates any formula cells in the table.
Sort	Sorts a selection of cells by row and/or column and keys.
AutoFill List	Creates, edits, or deletes an autofill list. An autofill list contains frequently-used series of text such as months and days of the week. When adding a new list, separate each item with a semi-colon. To use an autofill list, enter the first item into a cell in the table. Drag the cursor across or down and the Virtual User Recorder automatically fills in the cells in the range according to the autofill list.

Find
Find Again
◀ ▶
Top of Chapter
Back

Format Menu

The following commands are available in the Format menu:

Format Command	Description
General	Sets format to General. General displays numbers with as many decimal places as necessary and no commas.
Currency(0)	Sets format to currency with commas and no decimal places.
Currency(2)	Sets format to currency with commas and two decimal places.
Fixed	Sets format to fixed precision with commas and no decimal places.
Percent	Sets format to percent with no decimal places. Numbers are displayed as percentages with a trailing percent sign (%).
Fraction	Sets format to fraction.
Scientific	Sets format to scientific notation with two decimal places.
Date: (M/d/yy)	Sets format to Date with the M/d/yy format.
Time: h:mm AM/PM	Sets format to Time with the h:mm AM/PM format.
Custom Number	Sets format to a custom number format that you specify.
Validation Rule	Sets validation rule to test data entered into a cell or range of cells. A validation rule consists of a formula to test, and text to display if the validation fails.

Find
Find Again
◀ ▶
Top of Chapter
Back

Using Formulas in the Data Table

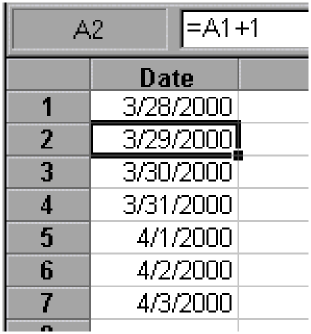
You can use any Excel formula in your data table. This enables you to create contextually relevant data during the test run. You can also use formulas as part of a checkpoint to check that objects from a page created on-the-fly (dynamically generated) or other variable objects in your Web page have the values you expect for a given context.

Using Formulas to Create Input Parameterization Data

You can enter formulas rather than fixed values in the cells of an input parameter column.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

For example, suppose you want to parameterize the value for a WebEdit object that requires a date value no earlier than today's date. You can set the cells in the Date column to the date format, and enter the =NOW() Excel formula into the first row in order to set the value to today's date for the first iteration. Then you can use another formula in the rest of the rows in order to enter the above date plus one day, as shown below. By using this formula you can run the test on any day, and the dates will always be valid.



A2 | =A1+1

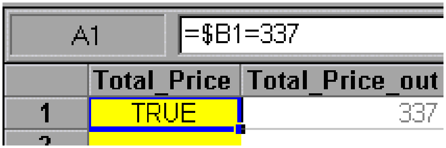
	Date
1	3/28/2000
2	3/29/2000
3	3/30/2000
4	3/31/2000
5	4/1/2000
6	4/2/2000
7	4/3/2000

Find
Find Again
Top of Chapter
Back

Using Formulas in Checkpoints

You can use a formula in a checkpoint in order to confirm that an object from a page created on-the-fly (dynamically generated) or another variable object in your Web page contains the value it should for a given context. For example, suppose a shopping cart Web site displays a price total. You can create a text checkpoint on the displayed total value and use a data table formula to check whether the site properly computes the total, based on the individual prices of the products selected for purchase in each iteration.

When you use the data table formula option with a checkpoint, Astra LoadTest creates two columns in the data table. The first column contains a default checkpoint formula. The second column contains the value to be checked in the form of an output parameter. The result of the formula is Boolean: TRUE or FALSE.



A1 =$B1=337

	Total_Price	Total_Price_out
1	TRUE	337
2		

A FALSE result in the checkpoint column during a test run causes the test to fail.

Once you finish adding the checkpoint, you can modify the default formula in the first column to perform the check you need.

To use a formula in a checkpoint:

1 Select the page, text, or object for which you want to create a checkpoint and open the Insert Checkpoint dialog box as described in Chapter 4, **Creating Checkpoints**.

2 Click **Parameter** and specify a logical name for the parameter.

3 Select the **Use data table formula** check box.

4 Specify your other checkpoint setting preferences as described in Chapter 4, **Creating Checkpoints**.

Find
Find Again
◀ ▶
Top of Chapter
Back

5 Click **OK.**

6 Confirm the addition of two columns in the data table. The two columns are added to the table, and a checkpoint ✓ icon is added to your test tree.

7 Highlight the value in the first (formula) column to view the formula and modify the formula to fit your needs.

8 If you want to run several iterations, add the appropriate formula in subsequent rows of the formula column for each iteration in the test or action.

Find
Find Again
◀ ▶
Top of Chapter
Back

Using Data Table Scripting Functions

Astra LoadTest provides several data table functions that enable you to retrieve information about the data table and to set the value of cells in the data table.

You enter these statements manually in the Expert View. For more information about working in the Expert View see Chapter 20, **Testing in the Expert View**.

For more details on the data table functions, refer to the *Astra LoadTest Function Reference*.

From a programming perspective, the data table is made up of three types of objects: DataTable, Sheet (sheet), and Parameter (column). Each object has several functions and properties that you can use to retrieve or set values.

The functions and properties available for each type of object are described below.

The DataTable Object

The table below summarizes the functions and properties of the DataTable object. For these functions and properties, use the syntax:

DataTable.PropOrFunc (*params* **)**

For example, the following statement returns the MySheet sheet.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

DataTable.GetSheet ("MySheet")

Function or Property	Description
DataTable (*ParameterID* , *SheetID* **)** or **DataTable.Value (** *ParameterID* , *SheetID* **)**	Retreives the value of the cell in the specified parameter and the current row.
DataTable (*ParameterID* , *SheetID* **)=***newvalue* or **DataTable.Value (** *ParameterID* , *SheetID* **)=***newvalue*	Sets the value of the cell in the specified parameter and the current row.
RawValue (*ParameterID* , *SheetID* **)**	Retrieves the *raw value* of the specified parameter and current row. The *raw value* is the actual string written in the cell before it has been computed, such as the actual text of a formula.
GetSheetCount	Returns the total number of sheets in the data table.
GetSheet (*SheetID* **)**	Returns the specified sheet.
GlobalSheet	Returns the Global sheet.
LocalSheet	Returns the current local sheet.

Find
Find Again
Top of Chapter
Back

Function or Property	Description
AddSheet (*SheetName* **)**	Adds the specified sheet and returns it so that you can directly set or return properties of the new sheet in the same statement.
DeleteSheet (*SheetID* **)**	Deletes the specified sheet. Note that deleting the sheet will cause the test to fail if the corresponding action has parameterized values.
GetRowCount	Returns the total number of rows in the longest column of the Global sheet.
GetCurrentRow	Returns the current row in the Global sheet.
SetCurrentRow (*RowNumber* **)**	Sets the specified row as the current row in the Global sheet.
SetNextRow	Sets the row after the currently active row as the new current row in the Global sheet.
SetPrevRow	Sets the row above the currently active row as the mew current row in the Global sheet.
Import (*FileName* **)**	Imports the specified Excel file. Note that the imported table completely replaces all data in the existing data table.
Export (*FileName* **)**	Saves a copy of the data table in the specified location.

Find
Find Again
Top of Chapter
Back

The Sheet Object

The table below summarizes the functions and properties of the Sheet object. For these functions and properties, use the syntax:

```
...Sheet.PropOrFunc ( params ).
```

For example, the following statement returns the name that Astra LoadTest assigned to the newly added Sheet, which may be different than the name specified, if the specified name already exists in the data table or if the name contains an invalid character.

```
DataTable.AddSheet ("MySheet").Name
```

In the above example, if the sheet, ‘MySheet’ already exists, Astra LoadTest returns *MySheet1* as the name of the new sheet.

Function or Property	Description
GetParameterCount	Returns the total number of parameters (columns) in the sheet.
GetParameter (*ParameterID* **)**	Returns the specified parameter from the sheet.
Name	Returns the name of the sheet.

Find
Find Again
Top of Chapter
Back

Function or Property	Description
AddParameter (*ParameterName, value* **)**	Adds the specified parameter to the sheet, sets the value of the first row to the specified value, and returns the parameter so that you can directly set or return properties of the new parameter in the same statement.
DeleteParameter (*ParameterID* **)**	Deletes the specified parameter (column) from the sheet. Note that deleting a parameter from the data sheet will cause the test to fail if a corresponding parameter exists in the test.
GetRowCount	Returns the total number of rows in the longest column of the sheet.
GetCurrentRow	Returns the current row in the sheet.
SetCurrentRow (*RowNumber* **)**	Sets the specified row as the current row in the sheet.
SetNextRow	Sets the row after the currently active row as the new current row in the sheet.
SetPrevRow	Sets the row above the currently active row as the new current row in the sheet.

Find
Find Again
Top of Chapter
Back

The Parameter Object

The table below summarizes the functions and properties of the Parameter (column) object. For these functions and properties, use the syntax:

...Parameter.PropOrFunc (*params* **)**

For example, the following statement returns the name that Astra LoadTest assigned to the newly added Parameter (column), which may be different than the name specified, if the specified parameter name already exists in the sheet or if the name contains an invalid character.

DataTable.GetSheet("MySheet").AddParameter("ParamA", 2).Name

Function or Property	Description
Parameter **or** **Parameter.value**	Retrieves the value of the cell in the current row of the parameter.
Parameter=*newvalue* or **Parameter.value=***newvalue*	Sets the value of the cell in the current row of the parameter.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Function or Property	Description
RawValue	Retrieves the *raw value* of the current row of the parameter. The *raw value* is the actual string written in the cell before it has been computed, such as the actual text of a formula.
ValueByRow (*RowNum* **)**	Returns the value of the cell in the specified row of the parameter
Name	Returns the name of the parameter.

Find
Find Again
Top of Chapter
Back

Creating Tests
Using Regular Expressions

You can use regular expressions to increase the flexibility and adaptability of your tests. This chapter describes:

Find

Find Again

◀ ▶

Top of Chapter

Back

About Regular Expressions

When you run your test, regular expressions enable Astra LoadTest to identify Web objects and text strings with varying values. You can use regular expressions when defining the properties of a step or when parameterizing a step, and when creating checkpoints with varying values. For example, when you create a checkpoint on a text string with a varying date, you can define the date as a regular expression.

A regular expression is a string that specifies a complex search phrase. By using special characters such as a period (.), asterisk (*), caret (^), and brackets ([ ]), you define the conditions of the search. When one of these special characters is preceded by a backslash (\), Astra LoadTest searches for the literal character.

Find
Find Again
◀ ▶
Top of Chapter
Back

Using Regular Expressions in Steps

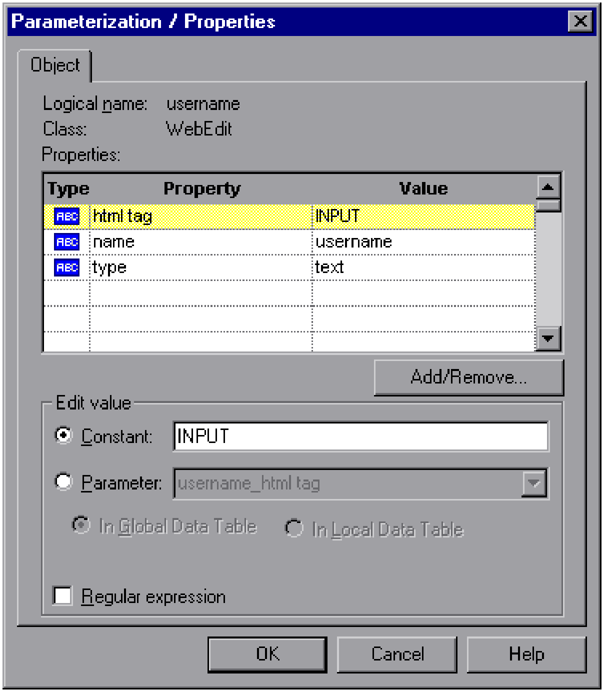
You can use regular expressions when defining or parameterizing a step in your test tree. A step is made up of an *object* that you navigate in your Web page, and/or a *method* by which you navigate the step. You can use regular expressions when defining or parameterizing the object of a step.

For example, your site may include a form in which the user inputs data and clicks the Send button to submit the form. When a required field is not completed, the form reappears for the user to complete. When resubmitting the form, the user clicks the Resend button. You can define the value of the button’s “name” property as a regular expression, so that Astra LoadTest ignores variations in the button name when clicking the button.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

To define a property value as a regular expression:

1 Right-click a step in the test tree and choose **Object Properties**. The **Parameterization/Properties** dialog box opens.



Parameterization / Properties

Logical name: username

Class: WebEdit

Properties:

Type	Property	Value
ABC	html tag	INPUT
ABC	name	username
ABC	type	text

Edit value

● Constant: INPUT

○ Parameter: username_html tag

● In Global Data Table ○ In Local Data Table

□ Regular expression

Find
Find Again
Top of Chapter
Back

The Object tab displays information about the object in the step:

Information	Description
Logical name	The name of the object as defined in the HTML code of the Web page.
Class	The type of object. In this example, the "WebButton" class indicates that the object is a button.

The Object tab displays the properties of the object in the step:

Option	Description
Type	The ABC icon indicates that the property value is a constant. The ▦ icon indicates that the property value is a parameter.
Property	The name of the property value.
Value	The value of the property.
Add/Remove	Opens the **Add/Remove** dialog box to enable you to modify the list of properties that you can check. To add/remove a property, select/clear a check box and click **OK**. To set the default, click the **Default** button and click **OK**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

2. Click the property you want to set as a regular expression in the **Properties** section. The property is highlighted.

3. In the **Edit value** section, set the property value as a regular expression.

   - To set the property value as a constant, click **Constant**.

     In the **Constant** box, set the value as a regular expression. For information on regular expression syntax, see **Regular Expression Syntax** on page 199.

   - To set the property value as a parameter, click **Parameter**.

     In the **Parameter** box, choose a parameter from the list or enter a new name. To use a parameter that you already created, select it from the list. To create a new parameter, either use the default parameter name or enter a descriptive name for the parameter. For more information on parameterization, see Chapter 7, **Parameterizing Tests**.

     To add the parameter to the Global tab in the Data pane, select **Global**. To add the parameter to the Action tab, select **Local**. For more information, see Chapter 11, **Working with Actions**.

Note: The property value in the **Parameter** box should not be defined as a regular expression. When you add additional values for the parameter into the table in the Data pane, you specify the values as regular expressions.

For information on regular expression syntax, see **Regular Expression Syntax** on page 199. For information on editing the table, see Chapter 7, **Parameterizing Tests**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

4 Select the **Regular Expression** check box. You are prompted to normalize the regular expression.

5 By default,Astra LoadTest treats all characters in a regular expression literally, except for a period (**.**), asterisk (*), caret (**^**), brackets (**[ ]**), parentheses (**()**), dollar sign (**$**), vertical line (**|**), plus sign (**+**), question mark (**?**), and backslash (****). When one of these special characters is preceded by a backslash (****), Astra LoadTest treats it as literal character.

- Click **Yes** to instruct Astra LoadTest to treat a special character literally. The special character is now preceded by a backslash (****).
- Click **No** to instruct Astra LoadTest to treat all the characters literally, except for special characters.

6 Click **OK** to save and close the Parameterization/Properties dialog box.

If you created a new parameter, the **Astra parameters** dialog box prompts you to add the new parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new parameter.

In your test tree, the 𝒫 icon next to the step indicates that the step has been parameterized.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Using Regular Expressions in Object Checkpoints

When creating an object checkpoint to verify that an object appears on your Web site, you can also set the property value of the object as a regular expression, so that the object with varying name can be verified.

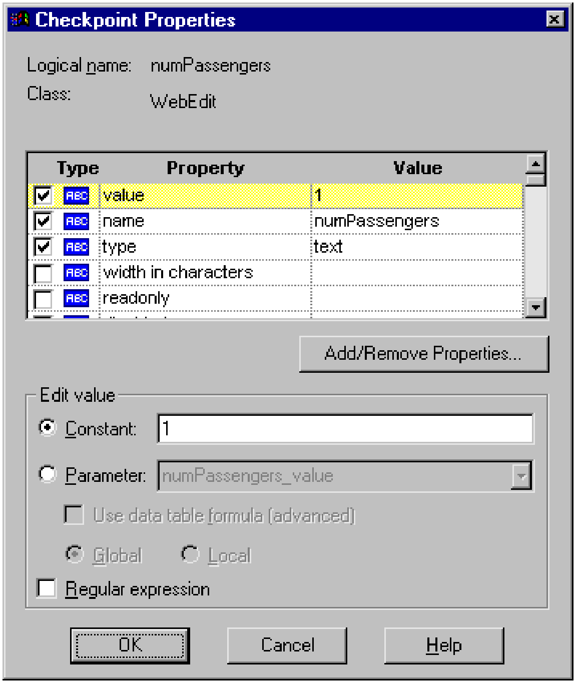
For example, suppose you want to check that when booking the number of passengers for a flight reservation in the “Mercury Tours” sample site, whole numbers are used. Astra LoadTest will ignore variations in the object’s property value as long as the value is a whole number.

To define a regular expression in an object checkpoint:

1 Right-click the object on the ActiveScreen and choose **Insert Checkpoint**. The **Object Selection - Checkpoint Properties** dialog box opens.

Find
Find Again
◀ ▶
Top of Chapter
Back

2 Select the object in the **Object Selection - Checkpoint Properties** dialog box and click **OK**. The **Checkpoint Properties** dialog box opens.



Checkpoint Properties

Logical name: numPassengers

Class: WebEdit

Type	Property	Value
☑ ABC	value	1
☑ ABC	name	numPassengers
☑ ABC	type	text
□ ABC	width in characters	
□ ABC	readonly	

Add/Remove Properties...

Edit value

● Constant: 1

○ Parameter: numPassengers_value

□ Use data table formula (advanced)

● Global ○ Local

□ Regular expression

The Checkpoint Properties dialog box enables you to specify which properties of the object to check, and to edit the values of these properties. For more information, see Chapter 4, **Creating Checkpoints**.

Find
Find Again
Top of Chapter
Back

3 Select the check box of a property to be set as a regular expression. The property is highlighted.

4 In the **Edit value** section, set the property value as a regular expression.

- To set the property value as a constant, click **Constant**.

  In the **Constant** box, set the value as a regular expression. For information on regular expression syntax, see **Regular Expression Syntax** on page 199.

- To set the property value as a parameter, click **Parameter**.

  In the **Parameter** box, choose a parameter from the list or enter a new name: To use a parameter that you already created, select it from the list. To create a new parameter, either use the default parameter name or enter a descriptive name for the parameter. For more information on parameterization, see Chapter 7, **Parameterizing Tests**.

  To add the parameter to the Global tab in the Data pane, select **Global**. To add the parameter to the Action tab, select **Local**. For more information, see Chapter 11, **Working with Actions**.

Note: The property value in the **Parameter** box should not be defined as a regular expression. When you add additional values for the parameter into the table in the Data pane, you specify the values as regular expressions.

For information on regular expression syntax, see **Regular Expression Syntax** on page 199. For information on editing the table, see Chapter 7, **Parameterizing Tests**.

Find
Find Again
◀ ▶
Top of Chapter
Back

5 Select the **Regular Expression** check box. You are prompted to normalize the regular expression.

6 By default,Astra LoadTest treats all characters in a regular expression literally, except for a period (**.**), asterisk (*), caret (**^**), brackets (**[ ]**), parentheses (**()**), dollar sign (**$**), vertical line (**|**), plus sign (**+**), question mark (**?**), and backslash (****). When one of these special characters is preceded by a backslash (****), Astra LoadTest treats it as literal character.

- Click **Yes** to instruct Astra LoadTest to treat a special character literally. The special character is now preceded by a backslash (****).
- Click **No** to instruct Astra LoadTest to treat all the characters literally, except for special characters.

7 Click **OK** to save and close the Checkpoint Properties dialog box.

If you created a new parameter, the **Astra parameters** dialog box prompts you to add the new parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new parameter.

In your test tree, the P icon next to the step indicates that the step has been parameterized. The ✓ icon indicates a checkpoint.

Find
Find Again
◀ ▶
Top of Chapter
Back

Using Regular Expressions in Text Checkpoints

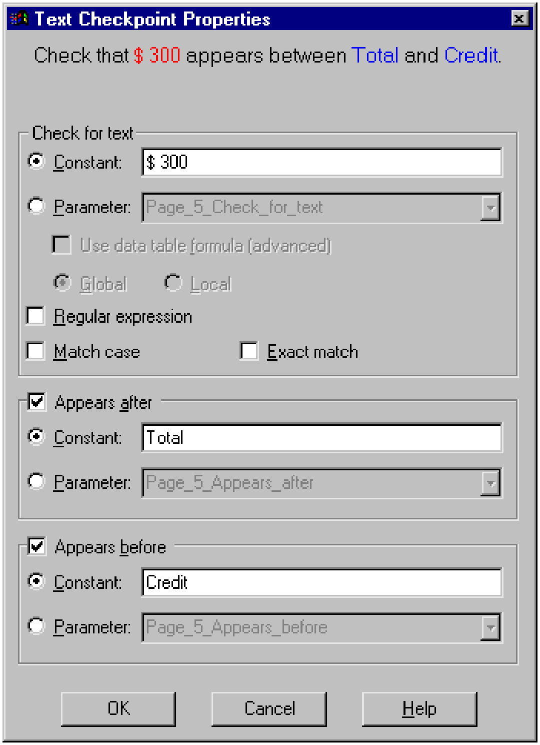
When creating a text checkpoint to check that a varying text string appears on your Web site, you define the text string as a regular expression.

For example, when booking a flight in the “Mercury Tours” sample site, the total cost charged to a credit card number should not be less than $300. You define the amount as a regular expression, so that Astra LoadTest will ignore variations in the text string as long as the value is not less than $300.

Find
Find Again
Top of Chapter
Back

To define a regular expression in a text checkpoint:

1 Open the **Text Checkpoint Properties** dialog box.



Text Checkpoint Properties

Check that $ 300 appears between Total and Credit.

Check for text
- ● Constant: $ 300
- ○ Parameter: Page_5_Check_for_text
- □ Use data table formula (advanced)
- ● Global ○ Local
- □ Regular expression
- □ Match case □ Exact match

☑ Appears after
- ● Constant: Total
- ○ Parameter: Page_5_Appears_after

☑ Appears before
- ● Constant: Credit
- ○ Parameter: Page_5_Appears_before

The Text Checkpoint Properties dialog box enables you to specify which text to check as well as which text appears before and after the text to check. For more information, see Chapter 4, **Creating Checkpoints**.

Find
Find Again
Top of Chapter
Back

2 In the **Check for text** section, define the text string as a regular expression.

- To set the text string as a constant, click **Constant**.

  In the **Constant** box, define the text string as a regular expression. For information on regular expression syntax, see **Regular Expression Syntax** on page 199.

- To set the text string as a parameter, click **Parameter**.

  In the **Parameter** box, choose a parameter from the list or enter a new name: To use a parameter that you already created, select it from the list. To create a new parameter, either use the default parameter name or enter a descriptive name for the parameter. For more information on parameterization, see Chapter 7, **Parameterizing Tests**.

  To add the parameter to the Global tab in the Data pane, select **Global**. To add the parameter to the Action tab, select **Local**. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Note: The name in the **Parameter** box should not be defined as a regular expression. When you add additional values for the parameter into the table in the Data pane, you specify the values as regular expressions.

For information on regular expression syntax, see **Regular Expression Syntax** on page 199. For information on editing the table, see Chapter 7, **Parameterizing Tests**.

3 Select the **Regular Expression** check box. You are prompted to normalize the regular expression.

4 By default,Astra LoadTest treats all characters in a regular expression literally, except for a period (**.**), asterisk (*), caret (**^**), brackets (**[ ]**), parentheses (**()**), dollar sign (**$**), vertical line (**|**), plus sign (**+**), question mark (**?**), and backslash (\). When one of these special characters is preceded by a backslash (\), Astra LoadTest treats it as literal character.

- Click **Yes** to instruct Astra LoadTest to treat a special character literally. The special character is now preceded by a backslash (\).
- Click **No** to instruct Astra LoadTest to treat all the characters literally, except for special characters.

5 Click **OK** to save and close the Text Checkpoint Properties dialog box.

If you created a new parameter, the **Astra parameters** dialog box prompts you to add the new parameter to the table in the Data pane. Click **OK**. A new column is highlighted in the table for the new parameter.

In your test tree, the P icon next to the step indicates that the step has been parameterized. The ✓ icon indicates a checkpoint.

Find
Find Again
◀ ▶
Top of Chapter
Back

Regular Expression Syntax

Astra LoadTest searches for all characters in a regular expression literally, except for a period (**.**), asterisk (*****), caret (**^**), brackets (**[ ]**), parentheses (**()**), dollar sign (**$**), vertical line (**|**), plus sign (**+**), question mark (**?**), and backslash (\) as described below. When one of these special characters is preceded by a backslash (\), Astra LoadTest searches for the literal character.

The following options can be used to create regular expressions:

Matching Any Single Character

A period (.) instructs Astra LoadTest to search for any single character. For example:

```
welcome.
```

matches welcomes, welcomed, or welcome followed by a space or any other single character. A series of periods indicates the same number of unspecified characters.

Find
Find Again
Top of Chapter
Back

Matching Any Single Character within a Range

In order to match a single character within a range, you can use square brackets ([ ]). For example, to search for a date that is either 1968 or 1969, write:

```
196[89]
```

You can use a hyphen (-) to indicate an actual range. For instance, to match any year in the 1960s, write:

```
196[0-9]
```

A hyphen does not signify a range if it appears as the first or last character within brackets, or after a caret (^).

A caret (^) instructs Astra LoadTest to match any character except for the ones specified in the string. For example:

```
[^A-Za-z]
```

matches any non-alphabetic character. The caret has this special meaning only when it appears first within the brackets.

Note that within brackets, the characters “.”, “*”, “[“ and “\” are literal. If the right bracket is the first character in the range, it is also literal. For example:

```
[]g-m]
```

matches the right bracket, and g through m.

Find
Find Again
Top of Chapter
Back

Matching Specific Characters

An asterisk (*) instructs Astra LoadTest to match one or more occurrences of the preceding character. For example:

```
me*
```

causes Astra LoadTest to match Mercury and menu, but not welcome.

A period “.” followed by an asterisk “*” instructs Astra LoadTestto locate the preceding characters followed by any combination of characters. For example:

```
O.*
```

causes Astra LoadTest to locate any string that starts with “O” (for example, On or Off).

You can also use a combination of brackets and an asterisk to limit the label to a combination of non-numeric characters. For example:

```
"!O[a-zA-Z]*"
```

Find
Find Again
Top of Chapter
Back

Combining Special Characters

You can combine special characters in a regular expression. The most common use of this is the combination of the '.' and '*' characters in order to find zero or more occurrences of any character.

For example,

```
start.*
```

matches start, started, starting, starter, etc.

Matching End of String

A dollar sign ($) instructs Astra LoadTest to match the end of a string. For example:

```
book
```

matches both book and bookend, while a string that is followed by ($), matches only that string. For example,

```
book$
```

matches only book.

Find
Find Again
Top of Chapter
Back

Matching either the Preceding or Following Expression

A vertical line (|) instructs Astra LoadTest to match either the preceding or following expression. For example:

```
l|book
```

matches l or book, while the following expression:

```
(l|b)ook
```

matches look or book.

Matching One or More of the Preceding Expression

A plus sign (+) instructs Astra LoadTest to match the preceding character one or more times. For example,

```
zo+
```

matches zoo but not z.

Matching the Preceding Character Zero or One Time

A question mark (?) instructs Astra LoadTest to match zero or more times the preceding character one or more times. For example,

```
ca?r
```

matches both car and cr, but nothing else.

Find
Find Again
◀ ▶
Top of Chapter
Back

Grouping Regular Expressions

Parentheses (()) instruct Astra LoadTest to match the pattern and remember the match. The matched substring can be retrieved using [0]...[n].

Using the Backslash Character

A backslash (\) instructs Astra LoadTest to treat the next character as either a special character or a literal. Examples are as follow:

- n matches the character n
- \n matches a newline character
- \\ matches \
- \(matches (

Find
Find Again
Top of Chapter
Back

Creating Tests
Working with Actions

The Virtual User Recorder enables you to divide your test into actions in order to streamline the testing process of your Web sites.

This chapter describes:

Find
Find Again
Top of Chapter
Back

About Working with Actions

When you use the Virtual User Recorder, you can utilize the action feature to design more modular and efficient tests. Actions enable you to parameterize specific components of a test and to reuse components created in other tests. Actions also enable you to easily re-record one action so that you don’t have to re-record the entire test when part of your web site or application changes .

When you divide your test into actions, you can:

- parameterize one action in a test without parameterizing the remainder of the test
- have unique sets of parameters and data in the table in the Data pane for each action
- copy an action from another test
- link to an action from another test

Find
Find Again
◀ ▶
Top of Chapter
Back

Using Multiple Actions in a Test

When you create a test, each test includes one action by default. All the steps you record and all the modifications you make after recording are part of a single action.

When you divide your test into multiple actions, you can parameterize each one separately. For information on parameterizing tests, see Chapter 7, **Parameterizing Tests**, and Chapter 8, **Creating Output Parameters**.

When you run a test with actions, the The Test Results are divided by actions within each test iteration so that you can see if each action passed, and you can view the results for each action individually. For more information on the Test Results window, see Chapter 15, **Analyzing Test Results in Stand-Alone Mode**.

Suppose you want to test how a flight reservation system handles multiple bookings. You may want to parameterize the test to check how your site responds to multiple sets of customer flight itineraries. When you plan your test, you plan the following procedures:

1 The travel agent logs into the flight reservation system.

2 The travel agent books five sets of customer flight itineraries.

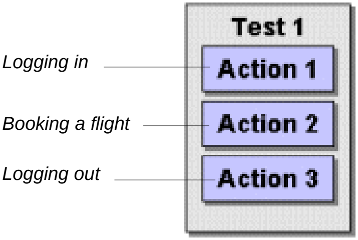
3 The travel agent logs out of the flight reservation site.

Find
Find Again
Top of Chapter
Back

When you consider these procedures, you realize that it is necessary to parameterize only the second step: after all, the travel agent logs into the flight reservation system only once, at the beginning and logs out of the system only once, at the end. Therefore, it is not necessary to parameterize the login and logout procedures in your test.

By creating three separate actions within your test—one for logging in, another for booking a flight, and a third for logging out—you can parameterize the second action in your test without parameterizing the others.

When you divide your test into three actions, it is structured as shown:



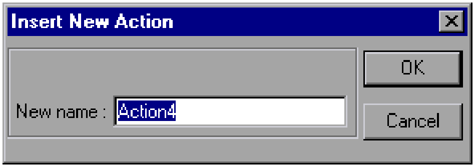
Find
Find Again
Top of Chapter
Back

Creating a New Action

You can create a new action in your test.

To create a new action in your test:

1 Choose **Insert > New Action** or click the **New Action** button. The Insert New Action dialog box opens.



New name : Action4

2 Type a new action name or use the default name.

3 Click **OK**.

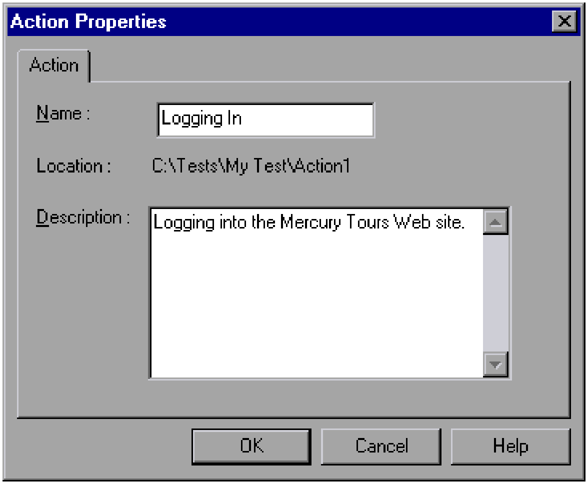
A new action is added to your test and is displayed at the bottom of the test tree.

You can also add a description for an action.

Find
Find Again
Top of Chapter
Back

To add a description to an action:

1 Right-click the action in the test tree and select **Action Properties**. The Action Properties dialog box opens.



Name : Logging In

Location : C:\Tests\My Test\Action1

Description : Logging into the Mercury Tours Web site.

2 Enter a description of the action in the **Description** box.

For every action in your test, the Virtual User Recorder creates a corresponding sheet in the Data pane so that you can enter parameters that are specific to that action only. For more information on parameterizing actions, see **Parameterizing an Action** on page 212.

Find
Find Again
Top of Chapter
Back

Copying an Action within a Test

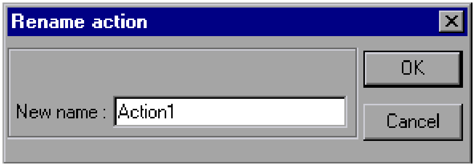
If you want to create two very similar actions, you can copy an action from within your test.

To copy an action within a test:

1 Right-click the action to copy.

2 Choose **Copy Action** in the menu.

3 Right-click the action and choose **Paste Action** in the menu. The Rename Action dialog box opens.


Rename action

New name : Action1

4 Give the action a logical name.

5 Click **OK**.

A new action is added to your test and is displayed at the bottom of the test tree.

Find
Find Again
Top of Chapter
Back

Parameterizing an Action

You parameterize an action by parameterizing the steps within that action. When you add a parameter to your test, you specify whether it is a *global* parameter, for the entire test, or a *local* parameter, for a specific action within the test.

In the parameterization dialog boxes:

- Choosing **Global** creates parameters in the **Global** sheet in the Data pane.
- Choosing **Local** creates parameters in the corresponding **Action** sheet in the Data pane. When there are parameters in an Action sheet, the Virtual User Recorder runs the relevant number of iterations on that action before continuing with the test.

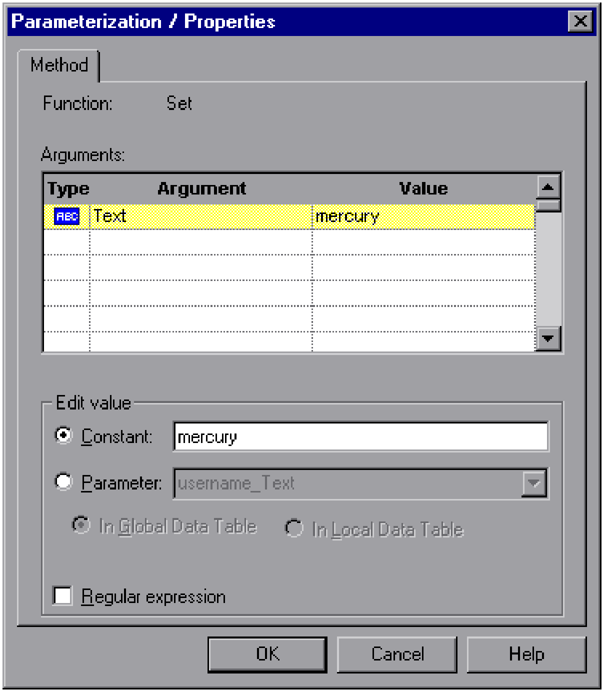
Note: You can choose to run iterations on specific rows within the action sheet rather than on all rows, from the Run Tab of the Test Settings dialog box. For more information, see Chapter 24, **Setting Testing Options for a Single Test**.

Find
Find Again
Top of Chapter
Back

To parameterize an action:

1 In the test tree, right-click a step. A menu opens.

2 Select **Function Arguments**. The **Parameterization/Properties** dialog box opens and displays the properties of the object in the step.



Function: Set

Arguments:

Type	Argument	Value
ABC	Text	mercury

Edit value

● Constant: mercury

○ Parameter: username_Text

● In Global Data Table ○ In Local Data Table

□ Regular expression

Find
Find Again
Top of Chapter
Back

3 In the **Edit value** section, click **Parameter**.

4 In the **Parameter** box, choose a parameter from the list or enter a new name.

- To use a parameter that you already created, select it from the list.
- To create a new parameter, accept the default parameter name or enter a descriptive name for the parameter.

5 Select **In Local Data Table** to add the parameter to the Action sheet.

6 Click **OK** to close the dialog box.

7 If you created a new parameter, the **Astra Parameter** dialog box prompts you to add the new parameter to the relevant action sheet in the Data pane. Click **OK**. A new column is highlighted in the table for the new parameter.

In your test tree, the P icon next to the step indicates that the step has been parameterized.

Note: You can specify additional data values for the parameter by entering them directly into the relevant action sheet in the Data pane. For more information, see Chapter 9, **Working with Data Tables**.

Find
Find Again
◀ ▶
Top of Chapter
Back

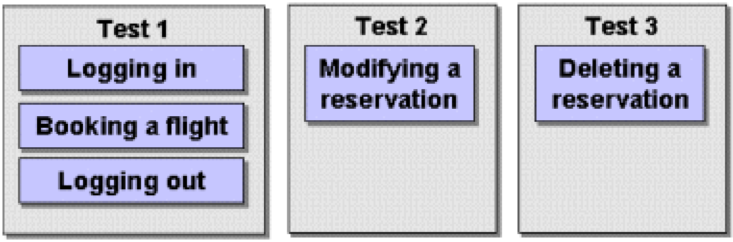
Inserting Actions from Another Test

When you plan a suite of tests, you may realize that each test requires one or more identical activities, such as logging in. Rather than record the login process three times in three separate tests, and enhancing this part of the script (with checkpoints and parameterization) separately for each test, you can use the **Insert > New Action** command to create an action in one test that logs into the flight reservation system. Once you are satisfied with the action you recorded and enhanced, you can insert it into other tests. You can insert an action from another test by pasting a copy of the action into your test, or by linking to the action in the original test.

For more information on creating a new action in a test, see **Using Multiple Actions in a Test** on page 207.

Find
Find Again
◀ ▶
Top of Chapter
Back

Suppose you wanted to record the following three tests in the Mercury Tours site: Booking a flight, Modifying a reservation and Deleting a reservation. While planning your test you realize that for each test you need to log in and log out of the site. If you plan to use the insert action option for the repeated activities, then you would initially record the three tests as shown:



Once you have set up your tests, you must choose whether you want to paste a copy of the action or paste a link to it.

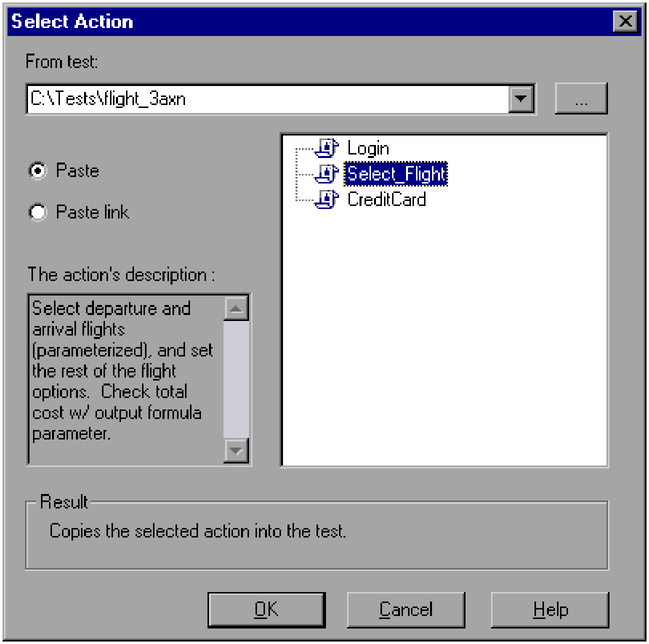
When you paste a copy of an action into a test, the action is copied in its entirety, including checkpoints, parameterization, and the corresponding action tab in the Data pane. Once it is pasted in the test, you can add to, delete from, or modify the action just as you would with any other recorded action. Any changes you make to this action after you insert it affect only this action, and changes you make to the action in the original test do not affect the action in this test.

When you paste a link to the action, the action is inserted in read-only format. You can view the components of the action, but you cannot modify them. If you modify the action in the original test, the modifications also take effect in any tests that are linked to the action.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

To paste a copy of an action into a test:

1 Choose **Insert > Action from test...**. The **Select Action** dialog box opens.



Select Action

From test:

C:\Tests\flight_3axn

● Paste

○ Paste link

The action's description :

Select departure and arrival flights (parameterized), and set the rest of the flight options. Check total cost w/ output formula parameter.

Result

Copies the selected action into the test.

2 Use the browse button to find the test from which you want to insert the action.

Find
Find Again
Top of Chapter
Back

3 Select the action you want to insert from the list. When you highlight an action, its description, if one exists, appears in the action's description box. This helps you identify the action you want to insert. For more information about inserting action descriptions see **page 210**.

4 Click **Paste**.

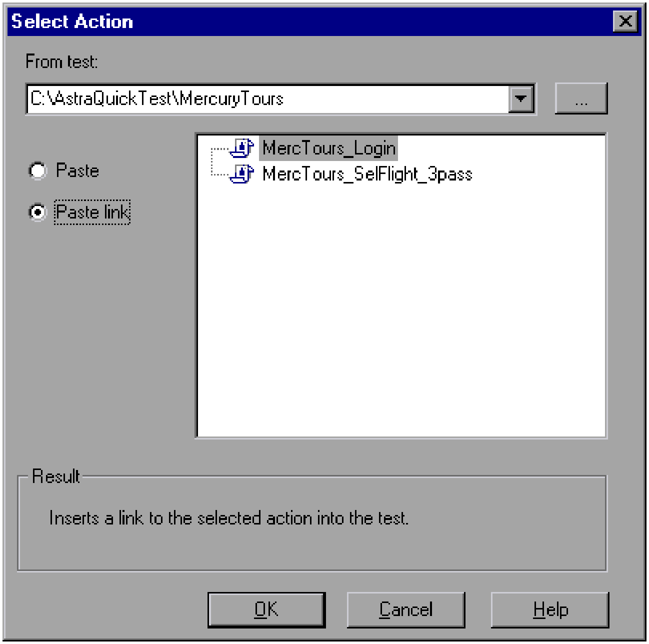
5 Click **OK**. The action is pasted into the test and appears at the bottom of the test tree. You can move the action to another location in the test by dragging it to the desired location in the test tree.

Note: If you try to insert an action in a test that already contains an action with the same name, you are prompted to rename the action. You can rename the action by choosing **Edit > Rename**.

Find
Find Again
◀ ▶
Top of Chapter
Back

To paste a link of an action into a test:

1 Choose **Insert > Action from test...**. The **Select Action** dialog box opens.



Select Action
From test:
C:\AstraQuickTest\MercuryTours
...
○ Paste
● Paste link
MercTours_Login
MercTours_SelFlight_3pass
Result
Inserts a link to the selected action into the test.
OK
Cancel
Help

2 Use the browse button to find the test from which you want to insert the action.

3 Select the action you want to insert from the list.

Find
Find Again
Top of Chapter
Back

4 Click **Paste link**.

5 Click **OK**. A link to the action is pasted into the test and appears at the bottom of the test tree with a special linked action icon. You can move the action to another location in the test by dragging it to the desired location in the test tree.

Find
Find Again
Top of Chapter
Back

Guidelines for Working with Actions

Consider the following guidelines when working with actions:

- When you parameterize an action in your test, the action must ‘clean up after itself’. In other words, the action must end at the same point it started, so that it can run again without interruption. For example, suppose you are testing the sample flight reservation site. If the action starts with a blank flight reservation form, it should conclude with a blank flight reservation form.
- A single test may include both global parameterization and action parameterization. For example, you can create a test in which a travel agent logs into the flight reservation system, books three flights, and logs out; the next travel agent logs into the flight reservation system, and books three flights and logs out, etc. To parameterize the ‘book a flight’ action, you choose **Local** in the parameterization dialog box and enter the three flights into the relevant **Action** tab in the Data pane. To parameterize the entire test, you choose **Global** in the parameterization dialog box and enter the login names and passwords for the different agents into the **Global** tab in the Data pane.
- You should rename the actions in your test with descriptive names to help you identify them. This facilitates inserting actions from one test to another. You can rename the action by choosing **Edit > Rename**.

Find
Find Again
◀ ▶
Top of Chapter
Back

- If you plan to use an identical procedure in more than one test, you should consider inserting an action from another test. If you want to make modifications to the action in only one test, or to make different modifications in different tests, you should use the **Paste** option to paste a copy of the action. If you want modifications to affect all tests containing the action, you should use the **Paste link** option to insert a link to the action in the original test.
- If you expect certain elements of your Web site to change regularly, it is a good idea to divide the steps related to changeable element into a separate action so that it will be easy to re-record the required steps when the site is modified.

Find
Find Again
◀ ▶
Top of Chapter
Back

Handling Unexpected Events and Errors

You can instruct the Virtual User Recorder to handle unexpected events and errors that occur in your testing environment during a test run.

This chapter describes:

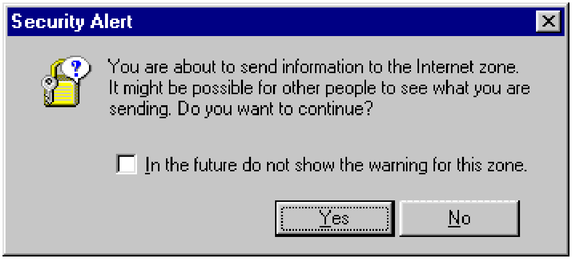
Find
Find Again
Top of Chapter
Back

About Handling Unexpected Events and Errors

Unexpected events and errors during a test run can disrupt your test and distort test results. This is a problem particularly when running tests unattended: the tests are suspended until you perform the action needed to recover.

You can use the *Exception Editor* to instruct the Virtual User Recorder to detect and handle the appearance of a specific dialog box and act to recover the test run.

For example, if a Security Alert dialog box is displayed during a test run, you can instruct the Virtual User Recorder to recover the test run by clicking the default button. In this particular case, the Yes button is the default button.



Security Alert

You are about to send information to the Internet zone. It might be possible for other people to see what you are sending. Do you want to continue?

□ In the future do not show the warning for this zone.

The Exception Editor contains a list of exceptions that the Virtual User Recorder supports. Each exception is associated with a handler function that is activated when there is a need to recover the test run. You can modify the list of exceptions and configure additional types of dialog box exceptions that you would like the Virtual User Recorder to support.

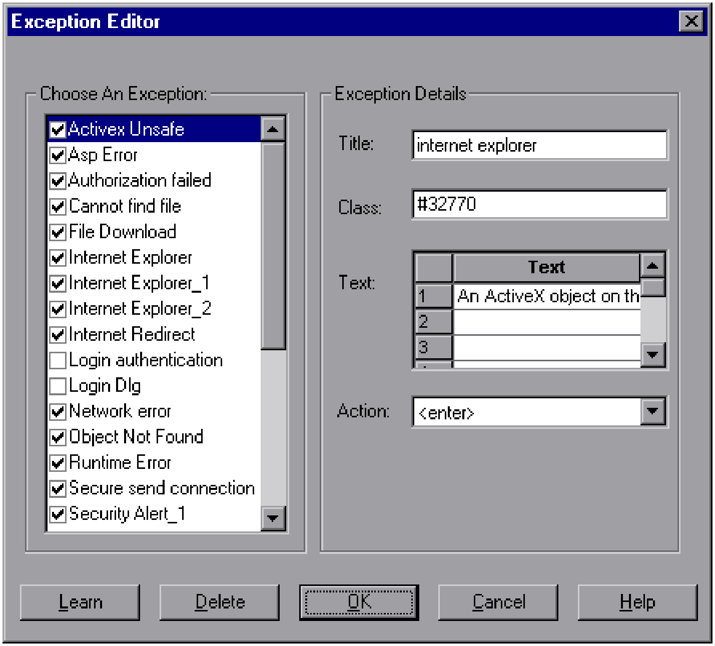
Find
Find Again
Top of Chapter
Back

Changing the Status of Exceptions

The Exception Editor includes a list of all the available exceptions. You can choose to activate or deactivate any exception in the list.

To change the status of an exception:

1 Choose **Tools > Exception Editor**. The **Exception Editor** opens.



☑ Activex Unsafe
☑ Asp Error
☑ Authorization failed
☑ Cannot find file
☑ File Download
☑ Internet Explorer
☑ Internet Explorer_1
☑ Internet Explorer_2
☑ Internet Redirect
☐ Login authentication
☐ Login Dlg
☑ Network error
☑ Object Not Found
☑ Runtime Error
☑ Secure send connection
☑ Security Alert_1

Title: internet explorer

Class: #32770

Text:

	Text
1	An ActiveX object on th
2	
3	

Action: <enter>

Find
Find Again
Top of Chapter
Back

2 In the **Choose An Exception** list, click an exception.

The exception is highlighted. The current description of the exception appears in the Exception Details area.

3 To activate an exception, select its check box. To deactivate the exception, clear its check box.

4 Click **OK** to save the changes.

Find
Find Again
◀ ▶
Top of Chapter
Back

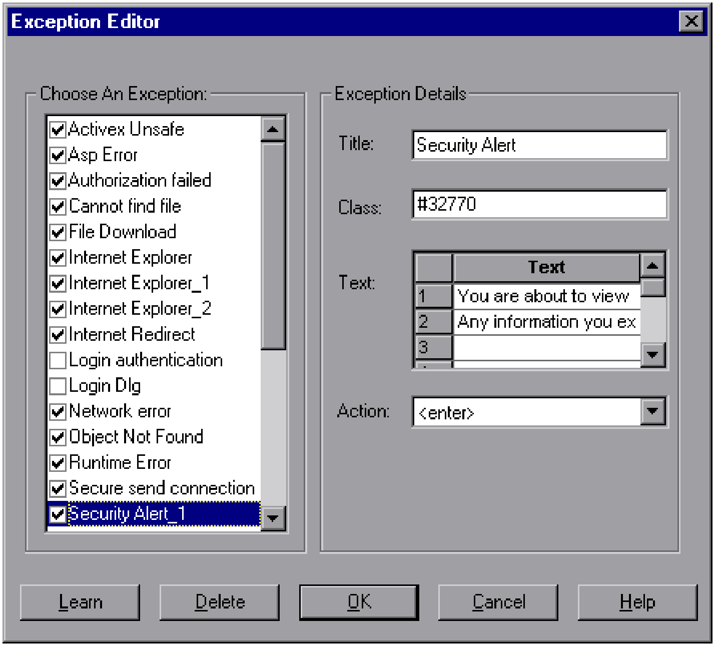
Modifying Exceptions

You can modify the details of an exception listed in the Exception Editor.

To modify the details of an exception:

1 Choose **Tools > Exception Editor**. The **Exception Editor** opens.

2 In the **Choose An Exception list**, click an exception.



Choose An Exception:

- ☑ Activex Unsafe
- ☑ Asp Error
- ☑ Authorization failed
- ☑ Cannot find file
- ☑ File Download
- ☑ Internet Explorer
- ☑ Internet Explorer_1
- ☑ Internet Explorer_2
- ☑ Internet Redirect
- □ Login authentication
- □ Login Dlg
- ☑ Network error
- ☑ Object Not Found
- ☑ Runtime Error
- ☑ Secure send connection
- ☑ Security Alert_1

Title: Security Alert

Class: #32770

Text:

	Text
1	You are about to view
2	Any information you ex
3	

Action: <enter>

The exception is highlighted. The current description of the exception appears in the Exception Details area.

Find
Find Again
Top of Chapter
Back

3 You can modify the title or the content of the exception dialog box, or which handler function to associate with it.

- To modify the title of the exception dialog box, edit the text in the **Title** box.
- To modify the text that appears in the exception dialog box, edit a text line in the **Text** box.
- To change the handler function associated with this exception dialog, choose a function from the **Action** list. This function recovers the test run.

Function	Description
<enter>	Presses the **Enter** key.
<login>	Uses the user name and password you supply in the **User Name** and **Password** edit boxes.
<press button>	Clicks the button you select from the **Button Name** list.

4 Click **OK** to save the changes.

Find
Find Again
Top of Chapter
Back

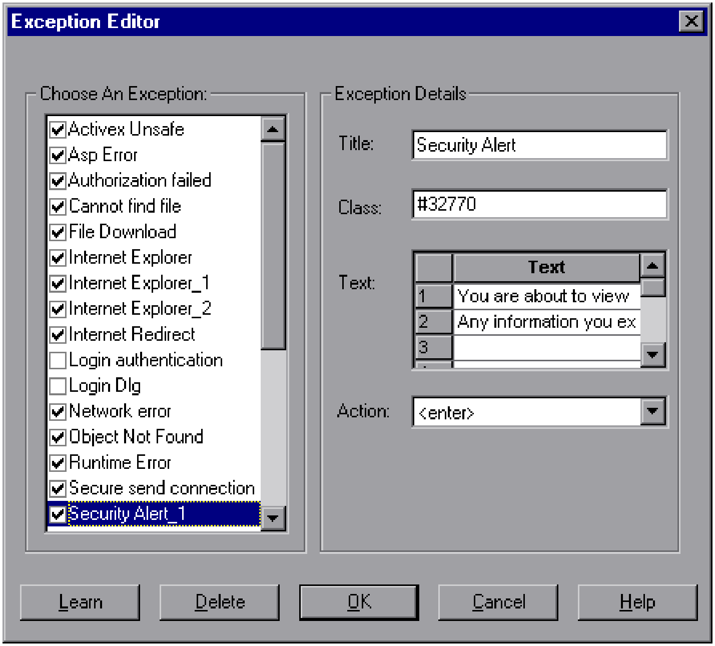
Adding New Exceptions

You can add a new exception to the list of exceptions in the Exception Editor.

To add a new exception:

1 Choose **Tools > Exception Editor**. The **Exception Editor** opens.

2 Click the **Learn** button. The mouse pointer becomes a pointing hand. Click the dialog box. A new exception is added to the list.



Choose An Exception:

- ☑ Activex Unsafe
- ☑ Asp Error
- ☑ Authorization failed
- ☑ Cannot find file
- ☑ File Download
- ☑ Internet Explorer
- ☑ Internet Explorer_1
- ☑ Internet Explorer_2
- ☑ Internet Redirect
- □ Login authentication
- □ Login Dlg
- ☑ Network error
- ☑ Object Not Found
- ☑ Runtime Error
- ☑ Secure send connection
- ☑ Security Alert_1

Exception Details

Title: Security Alert

Class: #32770

Text:

	Text
1	You are about to view
2	Any information you ex
3	

Action: <enter>

Find
Find Again
Top of Chapter
Back

The Exception Editor displays the title of the exception dialog box, the property class of the exception, the text that appears in the dialog box, and the handler function that is responsible for recovering test execution. To modify these fields, refer to **Modifying Exceptions** on page 227.

3 Click **OK** to save the exception.

Find
Find Again
◀ ▶
Top of Chapter
Back

Deleting Exceptions

You can delete an exception from the list of exceptions in the Exception Editor.

To delete an exception:

1 Choose **Tools > Exception Editor**. The **Exception Editor** opens.

2 In the **Choose An Exception list**, click an exception to delete.

The exception is highlighted. The current description of the exception appears in the Exception Details area.

3 Click the **Delete** button. A warning dialog box opens.

4 Click **Yes** to delete the exception.

5 Click **OK** to exit the Exception Editor.

Find
Find Again
◀ ▶
Top of Chapter
Back

Understanding How the Virtual User Recorder Identifies

This chapter explains how the Virtual User Recorder identifies objects in your application. It also describes how to modify the way the Virtual User Recorder identifies an object, which is useful when working with objects that change dynamically.

This chapter describes:

- Understanding Dynamic Descriptions of Objects
- Modifying How the Virtual User Recorder Identifies Objects

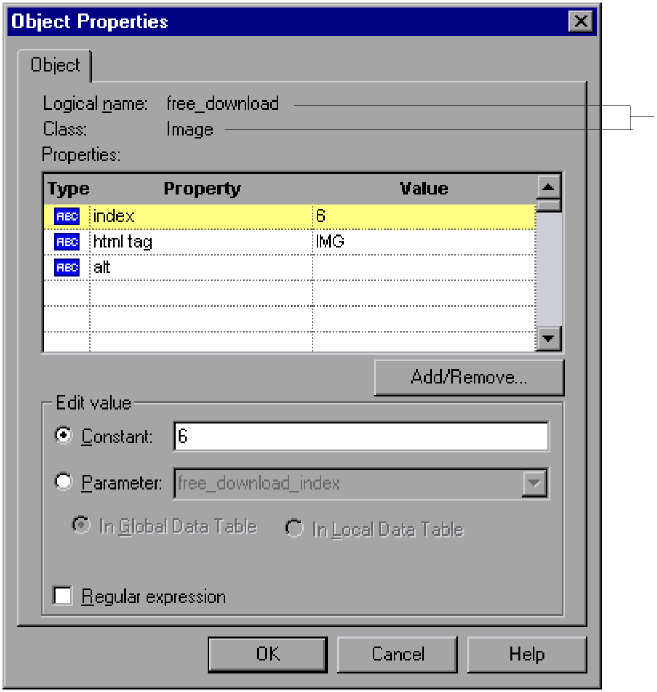
Find
Find Again
Top of Chapter
Back

About How the Virtual User Recorder Identifies Objects

The Virtual User Recorder identifies each object in your application by its *physical description*: a list of physical properties and their assigned values. The properties in an object's physical description are its logical name and its class. The *logical name* is the object's label. The *class* indicates the object's type. Each object belongs to a different class, according to its functionality, such as Browser, Image, Link (hypertext link), WebList (drop-down list box).

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

You can open the Object Properties dialog box for an object to see its physical description. The example below displays the physical description for an image in a Web page.



Object Properties

Object

Logical name: free_download

Class: Image

Properties:

Type	Property	Value
ABC	index	6
ABC	html tag	IMG
ABC	alt	

Edit value

● Constant: 6

○ Parameter: free_download_index

● In Global Data Table ○ In Local Data Table

□ Regular expression

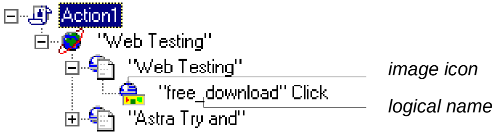
physical description

Find
Find Again
Top of Chapter
Back

For each object class, the Virtual User Recorder learns a set of default properties, which it uses to identify objects of that class in your application.

The Virtual User Recorder learns the values of an object's default properties when it records a test, and it uses this information to identify the object when it runs the test.

For example, by default, the Virtual User Recorder identifies an image by its physical description, the image's HTML tag, the alternate text (displayed if the image is not available or as a tooltip for the image), and its index. The index is a unique number that the Virtual User Recorder uses to identify an object (in this case the image) on the Web page. The index is numbered in the order in which the object appears in the Web page, starting from top to bottom, and from left to right.



Find
Find Again
Top of Chapter
Back

Understanding Dynamic Descriptions of Objects

You can change the properties that the Virtual User Recorder uses to identify an object. This is useful when you want to create and run tests on an object that changes dynamically. An object may change dynamically if it is frequently updated or if it is created using dynamic content, e.g. from a database. You can also change the properties that identify an object in order to reference objects using properties that were not automatically learned while recording.

For example, suppose you are testing a Web site that contains news headlines. This site includes a hypertext link to each current news story. The text in the hypertext link changes as the current news stories change. Suppose you always want to create a step in your test in which you always click the same hypertext link in the News section of your Web page. Since the news is always changing, the text in the hypertext link keeps changing. You need to modify how the Virtual User Recorder identifies this hypertext link so that it can continue to find it.

The default properties for a Link object (hypertext link) are “text” and “HTML tag”. The text property is the text inside the link. The HTML tag property is always, “A”, which indicates text.

You can modify the default properties for a hypertext link, so that you can identify it by where it appears on the page, rather than by the text in the link. You can use the “source_image” property to check the link by a unique index identifier in the source HTML file instead of using the “text” property to check the link by the text in the link. This index does not change, even if the text within the link changes.

Find
Find Again
◀ ▶
Top of Chapter
Back

Modifying How the Virtual User Recorder Identifies Objects

Suppose you are testing a Web page that contains an image that is an advertisement. Clicking this image opens the advertiser’s Web page. You do not want to identify the image by its name, since the image changes whenever the advertiser changes. Therefore, the value of the name property changes. You would want to create a test that will run no matter what the image’s name is. Therefore, you would want the Virtual User Recorder to identify your image using properties other than the name property.

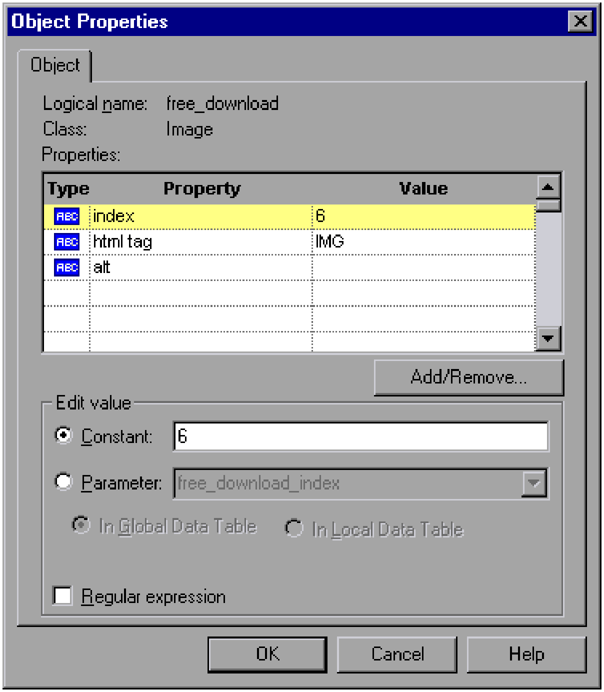
You can use the Add/Remove Properties dialog box to instruct the Virtual User Recorder to learn properties of the object other than the default properties.

Find
Find Again
◀ ▶
Top of Chapter
Back

To modify how the Virtual User Recorder identifies an object:

1 In the test tree, right-click the object for which you want to modify the identifying properties and choose **Object Properties**.

The Object Properties dialog box opens.



Logical name: free_download

Class: Image

Properties:

Type	Property	Value
ABC	index	6
ABC	html tag	IMG
ABC	alt	

● Constant: 6

○ Parameter: free_download_index

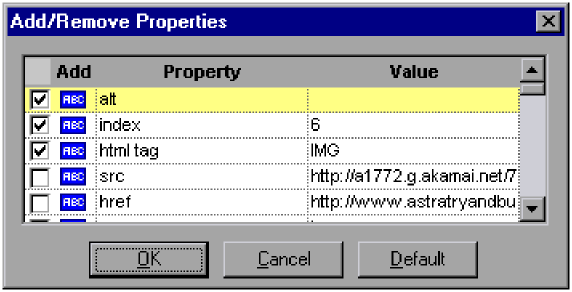
● In Global Data Table ○ In Local Data Table

□ Regular expression

Find
Find Again
Top of Chapter
Back

2 Click the **Add/Remove** button.

The Add/Remove Properties dialog box opens, listing the properties that can be used to identify the object. A selected check box next to a property indicates a property that the Virtual User Recorder uses to identify the object. The value for each property is displayed in the Value column.



Add/Remove Properties

Add		Property	Value
☑	ABC	alt	
☑	ABC	index	6
☑	ABC	html tag	IMG
□	ABC	src	http://a1772.g.akamai.net/7
□	ABC	href	http://www.astratryandbu

3 Modify the default properties that the Virtual User Recorder uses to identify the object:

- To add a property, select the corresponding check box.
- To remove a property, clear the corresponding check box.

4 Click **OK** to close the Add/Remove Properties dialog box.

The Object Properties dialog box is reactivated.

Tip: After you add a new property, you can modify its value in the Edit Value box in the Object Properties dialog box.

Find
Find Again
Top of Chapter
Back

Running and Debugging Tests

Find
Find Again
Top of Chapter
Back

Running Tests in Stand-Alone Mode

In order to perform load testing with a recorded test, you use the Controller to run the test within a scenario. Before integrating the test into a scenario, you check its functionality by running the test in stand-alone mode.

Running the test in stand-alone mode means running the test without using the Controller. This is done to establish how the test will execute when run from the Controller.

This chapter describes:

- **Running a Test**
- **Optional Steps**

Find
Find Again
◀ ▶
Top of Chapter
Back

About Running Tests

When you run a test, the Virtual User Recorder navigates through your Web site, performing the steps you recorded. If your test does not contain parameterized values, the Virtual User Recorder runs the test once. If the test contains global parameters (in the Global tab of the Data pane), the Virtual User Recorder runs the test for each row in the table, using the parameters you specified.

If the test contains local parameters (in one or more Action tabs in the Data pane), you can specify whether to run the action in iterations, or to run the action in iterations for a range of data sets. For additional information, see Chapter 11, **Working with Actions**.

Once the test run is complete, the Virtual User Recorder displays a report detailing the test results. After you run your test, you can update your ActiveScreen with the screens of your current test results. By updating, you refresh the screens in your ActiveScreen to include any changes to your Web site.

You can also run tests on objects with dynamic descriptions. For additional information, see Chapter 13, **Understanding How the Virtual User Recorder Identifies Objects**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Running a Test

When you run a test, the Virtual User Recorder performs the steps you recorded on your Web site and displays each Web page in your browser. The Virtual User Recorder always runs a test from the first step in the test.

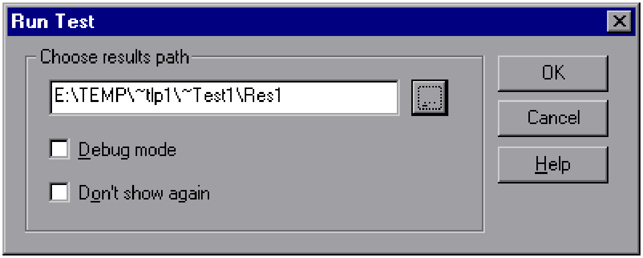
If your test does not contain parameterized values, the Virtual User Recorder runs the test once. If the test does contain parameters, the Virtual User Recorder runs the test for each row in the table in the Data pane, using the parameters you specified. If an action within your test is parameterized, you can choose to set and run only certain data sets. For more information, see Chapter 11, **Working with Actions**.

Find
Find Again
Top of Chapter
Back

To run a test:

1 If your test is not already open, choose **File > Open** or click the **Open** button to open the test.

2 Click the **Start Run** button on the toolbar, or choose **Test > Run**. The **Run Test** dialog box opens, displaying the path and a default test run name for the test results.



E:\TEMP\~tlp1\~Test1\Res1

□ Debug mode

□ Don't show again

3 To save the test results under a different name, type it in the text box or click **Browse** to locate the folder.

To run the test and overwrite the previous test results, select the **Debug mode** check box.

If you do not want to display the Run Test dialog box the next time you run your test, select the **Don’t show again** check box.

Find
Find Again
Top of Chapter
Back

4 Click **OK**. The Run Test dialog box closes and the Virtual User Recorder begins running the test. The Virtual User Recorder always runs a test from the first step in the test. As the Virtual User Recorder runs the test, it highlights each step in the test tree.

When the test stops running, the Test Results window opens.

Note: If you want to interrupt a test that is running, you can:

Click the **Pause** button or choose **Debug > Pause**. The test pauses. To resume running a paused test, click the **Run** button or choose **Test > Run**.

Click the **Stop** button or choose **Test > Stop**. The test stops running and the **Test Results** window opens.

Find
Find Again
Top of Chapter
Back

Optional Steps

When running a test, if a step does not open a particular dialog box, does not interrupt the test run. It bypasses this step and continues to run the test. A bypassed step is an *optional* step. By default, sets some dialog boxes as optional steps. You can also set a step as optional.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Setting Optional Steps

When running a test, you can bypass certain steps if they are not required, by setting them as optional steps. For example, when recording a test, the site you are testing may prompt you to enter your user name and password in a logon window. However, when you run the test, the site does not prompt you to enter your user name and password, because it has retained the information that was previously entered. In this case, the steps that were recorded for entering the logon information are not required and should be marked as optional. When runs the test, it will use these optional steps when needed.

To set an optional step:

Right-click a step in the test tree and choose **Optional Step**. The Optional Step ? icon is added next to the selected step.

Note: You can also add an optional step from the Expert View using a `OptionalStep.Browser("browser_name").Page("page_name").Link("link_name")` VBScript statement. For information on working in Expert View, see Chapter 20, **Testing in the Expert View**. For information on the OptionalStep object, refer to the *Astra Function Reference*.

Find
Find Again
Top of Chapter
Back

Default Optional Steps

When running a test, if a step fails to open a particular dialog box, automatically bypasses this step and continues to run the test. When the test run is completed, a message is displayed for the step that failed to open the dialog box.

 automatically recognizes the following dialog boxes:

Logical Name	Title
Auto Complete	Auto Complete
File Download	File Download
IE message	Internet Explorer
Netscape message	Netscape
Password	Enter Network Password
Runtime error	Error
Security Alert	Security Alert
Security Information	Security Information
Security Warning	Security Warning
Username and Password Required	Username and Password Required

Find
Find Again
Top of Chapter
Back

Analyzing Test Results in Stand-Alone Mode

After you run a test, you can view a report of all the major events that occurred during the test run.

This chapter describes:

Find
Find Again
Top of Chapter
Back

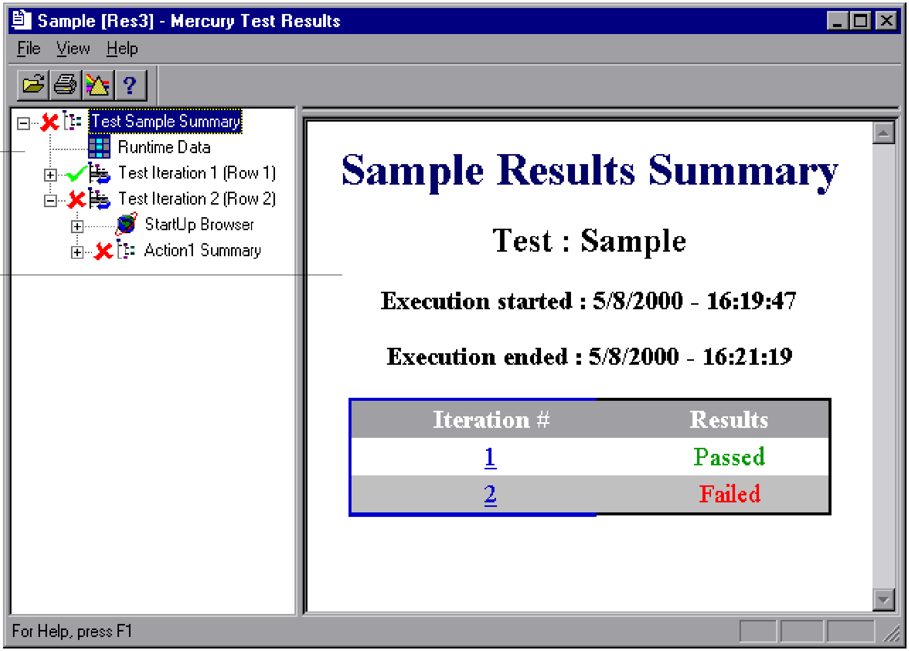
About Analyzing Test Results

After you run your test, the test results are displayed in the Test Results window. This window contains a description of every step performed during the test run. If the test does not contain parameterized values, the Test Results window shows a single test iteration result. If the test does contain parameters, the Test Results window shows a test iteration for each row in the table in the Data pane.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

The Test Results Window

After a test run, you can view the results in the Test Results window. By default, the Test Results window automatically opens when a test run is completed. For more information on opening the Test Results window, see **Viewing the Results of a Test Run** on page 252.



Sample Results Summary

Test : Sample

Execution started : 5/8/2000 - 16:19:47

Execution ended : 5/8/2000 - 16:21:19

Iteration #	Results
1	Passed
2	Failed

Test results tree

Test results details

Find
Find Again
Top of Chapter
Back

Test Results Tree

The left pane in the Test Results window displays the *test results tree*—a graphical representation of the test results. This includes the ✓ icon for a successful iteration, and the ✗ icon for a failed iteration. In the example above, the tree includes two iterations. The test results tree also includes the ▦ icon that displays the *Runtime Data*—a table that shows the values used to run a parameterized test, or the values retrieved from a parameterized test while it runs (output parameters). Note that your test results are organized by action.

You can collapse or expand a branch in the test results tree in order to change the level of detail that the tree displays.

Test Results Details

The right portion of the window displays the *test results details*—additional information for a selected branch of the report tree.

By default, when the Test Results window opens, a test summary appears. It indicates the test name, the date and time of the test run, and whether a test iteration passed or failed.

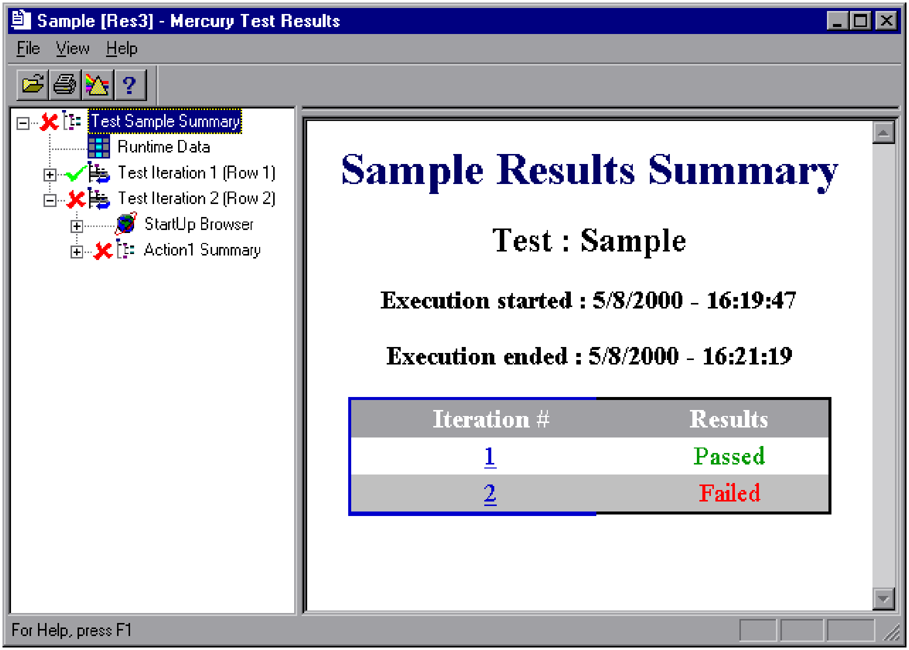
Viewing the Results of a Test Run

After a test run, you can view the results in the Test Results window. By default, the Test Results window automatically opens when a test run is completed. For more information, see **The Virtual User Recorder Testing Options** on page 376.

Find
Find Again
◀ ▶
Top of Chapter
Back

To view the results of a test run:

1 If the Test Results window is not already open, then click the **Test Results** button or choose **Test > Results**. The **Select Results File** dialog box opens. Select a results file (*.qtp* extension). Click **Open**. The **Test Results** window opens.



Sample Results Summary

Test : Sample

Execution started : 5/8/2000 - 16:19:47

Execution ended : 5/8/2000 - 16:21:19

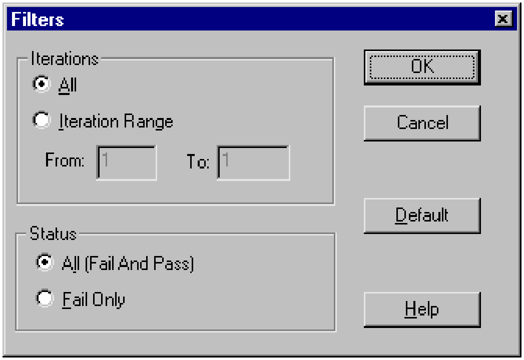
Iteration #	Results
1	Passed
2	Failed

Find
Find Again
Top of Chapter
Back

2 You can collapse or expand a branch in the test results tree in order to change the level of detail that the tree displays.

- To collapse a branch, click the Collapse (-) sign to the left of the branch icon. The report tree hides the details for the branch and the Collapse sign changes to Expand.
- To collapse all the branches in the report tree, choose **View > Collapse All**.
- To expand a branch, click the Expand (+) sign to the left of the branch icon. The tree displays the details for the branch and the Expand sign changes to Collapse.
- To expand all the branches in the report tree, choose **View > Expand All**.

3 To filter the information contained in your test results report, click the **Filter** button or choose **View > Filters**. The **Filters** dialog box opens.



Find
Find Again
Top of Chapter
Back

The default filter options are displayed above.

- Click **Iteration Range** to limit the test results to a specified range of test iterations.
- Click **Fail Only** to limit the test results to test iterations that failed.

4 To view other test run results, click the **Open** button or choose **File > Open.** The **Open** dialog box opens. Select a results file (.*qtp* extension) and click the **Open** button**.**

5 To print the test results, see **Printing Test Results** on page 260.

6 Choose **File > Exit** to close the Test Results window.

Note: You can open the Test Results window as a standalone application from the **Start** menu. To open the Test Results window, choose **Start > Programs > Astra LoadTest > Products > Report Viewer**.

Find
Find Again
Top of Chapter
Back

Viewing the Results of a Checkpoint

By adding checkpoints to your tests, you can compare pages, text strings, objects, and tables in different versions of your Web site. This enables you to ensure that your Web site functions as desired.

When you run the test, Astra LoadTest compares the expected results of the checkpoint to the current results. If the results do not match, the checkpoint fails. You can view the results of the checkpoint in the Test Results window.

For more information on checkpoint, see Chapter 4, Creating Checkpoints.

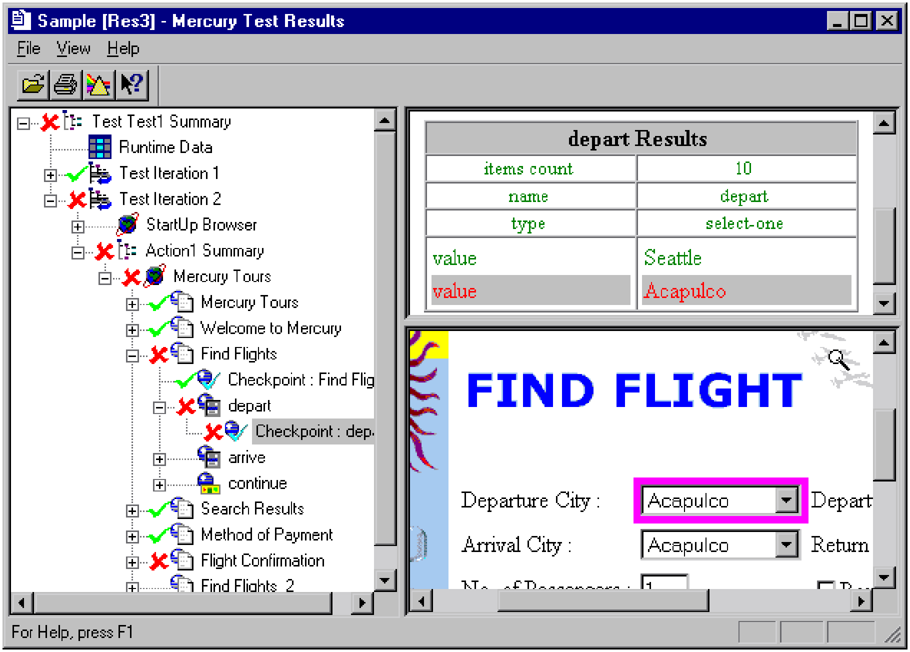
To view the results of a checkpoint:

1 If the Test Results window is not already open, then click the **Test Results** button or choose **Test > Results**. The **Select Results File** dialog box opens. Select a results file (*.qtp* extension). Click **Open**. The **Test Results** window opens.

2 In the left pane of the Test Results window, expand the branches of a test iteration.

Find
Find Again
Top of Chapter
Back

3 Click a checkpoint branch. The right pane displays detailed results of the selected checkpoint.



depart Results	
items count	10
name	depart
type	select-one
value	Seattle
value	Acapulco

Departure City : Acapulco

Arrival City : Acapulco

For Help, press F1

In the above example, the detailed results of the failed checkpoint indicates that the expected results and the current results do not match. The expected value of the flight departure is “Seattle”, but the actual value is “Acapulco”.

4 Choose **File > Exit** to close the Test Results window.

Find
Find Again
Top of Chapter
Back

Viewing the Runtime Data Table for a Parameterized Test

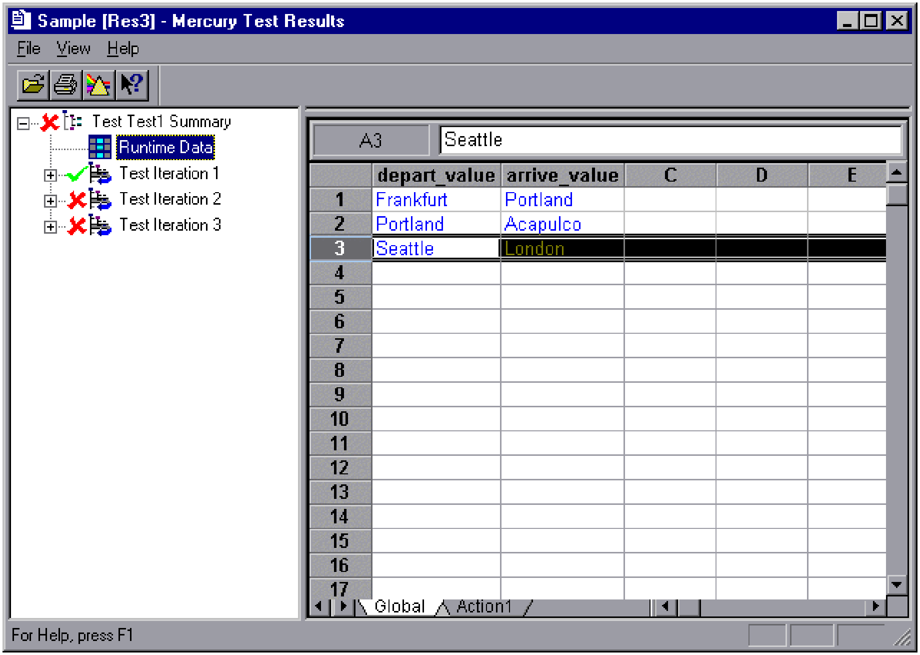
After you run a parameterized test, the Runtime data table displays the values used to run a parameterized test, or the values retrieved from a parameterized test while it runs (output parameters). For more information on parametrization, see Chapter 7, **Parameterizing Tests**. For more information on output parameterization, see Chapter 8, **Creating Output Parameters**. For more information on the test Data Table, see Chapter 9, **Working with Data Tables**.

To view the Runtime Data Table:

1 If the Test Results window is not already open, then click the **Test Results** button or choose **Test > Results**. The **Select Results File** dialog box opens. Select a results file (*.qtp* extension). Click **Open**. The **Test Results** window opens.

Find
Find Again
◀ ▶
Top of Chapter
Back

2 In the left pane of the Test Results window, click the **Runtime Data** icon. The right pane displays the Runtime data table.



A3 Seattle

	depart_value	arrive_value	C	D	E
1	Frankfurt	Portland			
2	Portland	Acapulco			
3	Seattle	London			
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					

In the above example, the Runtime data table contains the parameterized flight departure values and the flight arrival values.

3 Choose **File > Exit** to close the Test Results window.

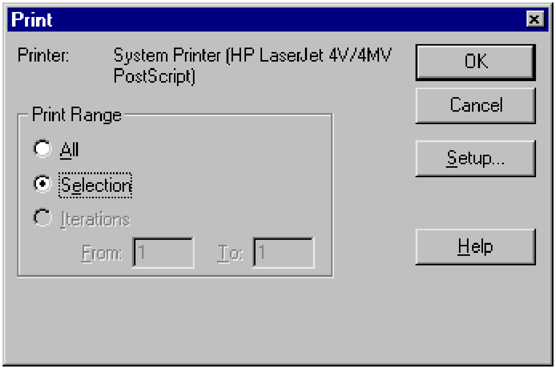
Find
Find Again
Top of Chapter
Back

Printing Test Results

You can print your test results from the Test Results window.

To print the test results:

1 To print the report, click the **Print** button or choose **File > Print**. The **Print** dialog box opens.



Printer: System Printer (HP LaserJet 4V/4MV PostScript)

Print Range

○ All

● Selection

○ Iterations

From: 1 To: 1

2 Select a **Print Range** option:

- Select **All** to print the entire results report.
- Select **Selection** to only print a selected branch in the report tree.
- Select **Iterations** to only print a specified range of test iterations.

3 Click **OK** to print.

Find
Find Again
Top of Chapter
Back

Debugging Tests

Controlling test runs can help you to identify and eliminate defects in your tests.

This chapter describes:

- Using the Step Commands
- Pausing Test Runs
- Setting Breakpoints
- Deleting Breakpoints

Find
Find Again
Top of Chapter
Back

About Debugging Test Scripts

After you create a test script you should check that it runs smoothly, without errors in syntax or logic. In order to detect and isolate defects in a test, you can use the Step and Pause commands to control how it runs. In addition, you can also control how the test runs by setting breakpoints.

The following Step commands are available:

- The Step Into command calls a function or displays another test.
- The Step Out command—used in conjunction with Step Into—completes the execution of a function or called test.
- The Step Over command executes a function or a called test.

You can also use the Pause command to temporarily suspend a test run. When you resume running the test, it continues from the point where you invoked the Pause command.

In addition, you can also control test runs by setting breakpoints. A breakpoint pauses a test run at a pre-determined point, enabling you to examine the effects of the test on your Web site.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Using the Step Commands

You can run a single line of a test using the Step Into, Step Out, and Step Over commands.

Find
Find Again
Top of Chapter
Back

Step Into

Choose **Debug > Step Into** or click the **Step Into** button to run only the current line of the active test script. If the current line of the active test calls another test or a function, the called test or function is executed in its entirety, and the called test or function is displayed in the Virtual User Recorder window.

Step Out

Choose **Debug > Step Out** or click the **Step Out** button only after entering a test or a user-defined function using Step Into. Step Out runs to the end of the called test or user-defined function, returns to the calling test, and then pauses the test run.

Step Over

Choose **Debug > Step Over** or click the **Step Over** button to run only the current step in the active test. When the current step calls another test or a user-defined function, the called test or function is executed in its entirety, but the called test script is not displayed in the Virtual User Recorder window.

Pausing Test Runs

You can temporarily suspend test runs by choosing **Debug > Pause** or clicking the **Pause** button. A paused test stops running when all previously interpreted steps have been run.

To resume running a paused test, click the **Run** button or choose **Test > Run**. The test run continues from the point that you invoked the Pause command.

Find
Find Again
◀ ▶
Top of Chapter
Back

Setting Breakpoints

By setting a breakpoint you can stop a test run at a specific place in a test. A breakpoint is indicated by a red-colored hand in the left margin of the test window. The Virtual User Recorder pauses the test run when it reaches a breakpoint. You can examine the effects of the test run up to the breakpoint, make any necessary changes, and then continue running the test from the breakpoint.

You can use breakpoints to:

- suspend a test run and inspect the state of your site
- mark a point from which to begin stepping through a test script using the Step commands

To set a breakpoint:

1 Click a step or a line in the test where you want the test run to stop.

2 Choose **Debug > Toggle Breakpoint** or click the **Toggle Breakpoint** button. The breakpoint symbol appears in the left margin of the Virtual User Recorder window.

Note: The breakpoints you define are active only during your current Astra LoadTest session. If you terminate your Astra LoadTest session, you must redefine breakpoints to continue debugging the script in another session.

Find
Find Again
Top of Chapter
Back

Deleting Breakpoints

You can delete a single breakpoint or all breakpoints defined for the current test using the Debug menu.

- To delete a single breakpoint, click a line in your test with the breakpoint symbol and choose **Debug > Toggle Breakpoint,** or click the **Toggle Breakpoint** button.

  The breakpoint symbol is removed from the left margin of the Virtual User Recorder window.

- To delete all breakpoints, choose **Debug > Clear All Breakpoints** or click the **Clear All Breakpoints** button.

  All breakpoint symbols are removed from the left margin of the Virtual User Recorder window.

Find
Find Again
◀ ▶
Top of Chapter
Back

Advanced Features

Testing in HTTP Mode

The Virtual User Recorder enables you to record and replay tests in HTTP mode. This provides the user with a high performance, highly scalable testing mode.

This chapter describes:

- Using HTTP Recording Mode
- Correlating Your Scripts
- Setting HTTP Testing Options
- Setting HTTP Run-Time Options
- About HTTP Vuser Warnings
- Editing Your HTTP Script

Find
Find Again
Top of Chapter
Back

About HTTP Mode

To increase the scalability of your Browser mode test, you can record and replay your test in HTTP mode. Unlike tests recorded in Browser mode which generate scripts relying on links and images, HTTP mode records your test as plain protocol actions.

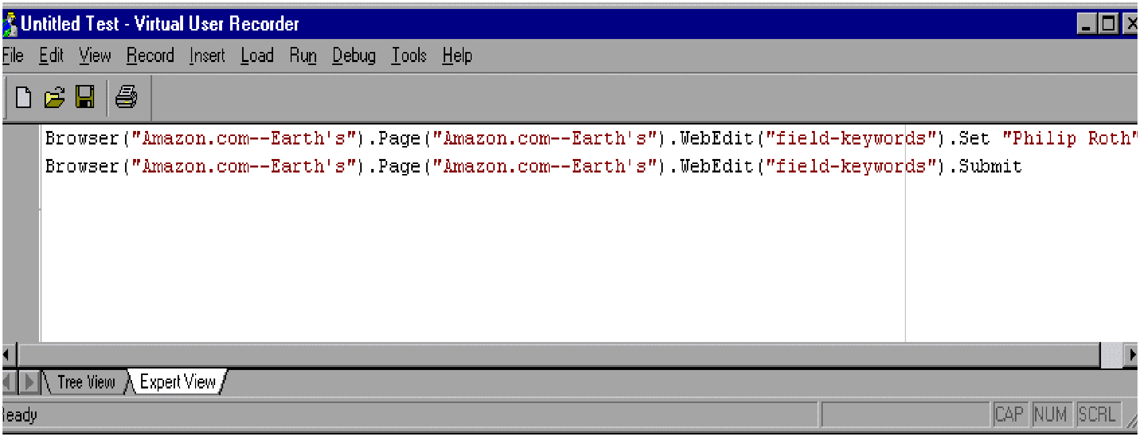
Use HTTP mode to record and replay scripts when:

- You want to increase scalability
- Content of a Web site is not HTML (Java, ActiveX)

Note: A test recorded in Browser mode can be replayed in both Browser mode and HTTP mode.

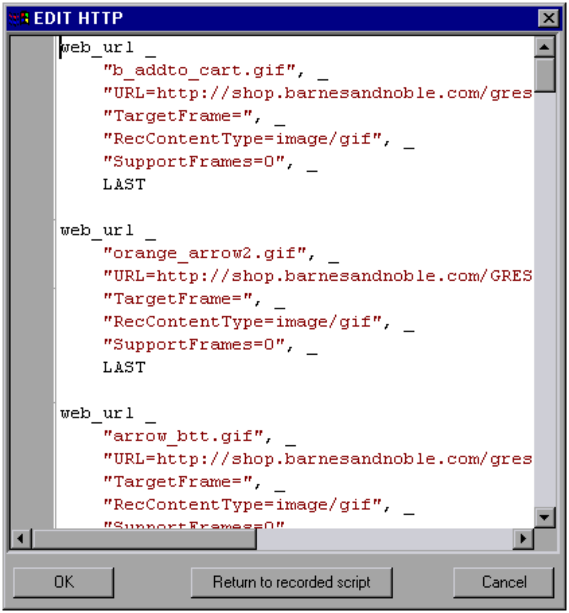
The following is a comparison between an Browser mode test and a portion of a test recorded in HTTP mode:

Find
Find Again
Top of Chapter
Back


Untitled Test - Virtual User Recorder
File Edit View Record Insert Load Run Debug Tools Help
Tree View
Expert View
Ready
CAP NUM SCRL

```
Browser("Amazon.com--Earth's").Page("Amazon.com--Earth's").WebEdit("field-keywords").Set "Philip Roth"
Browser("Amazon.com--Earth's").Page("Amazon.com--Earth's").WebEdit("field-keywords").Submit
```

Find
Find Again
Top of Chapter
Back



```
web_url _
    "b_addto_cart.gif", _
    "URL=http://shop.barnesandnoble.com/gres
    "TargetFrame=", _
    "RecContentType=image/gif", _
    "SupportFrames=0", _
    LAST

web_url _
    "orange_arrow2.gif", _
    "URL=http://shop.barnesandnoble.com/GRES
    "TargetFrame=", _
    "RecContentType=image/gif", _
    "SupportFrames=0", _
    LAST

web_url _
    "arrow_btt.gif", _
    "URL=http://shop.barnesandnoble.com/gres
    "TargetFrame=", _
    "RecContentType=image/gif", _
    "SupportFrames=0"
```

As the above example illustrates, a script recorded in HTTP mode is much longer and more detailed than a script recorded in Browser mode. This is due to the complexity of the dynamic data captured in HTTP mode.

Find
Find Again
Top of Chapter
Back

Correlating Your Scripts

In a standard Browser mode script containing linkage and image information, the output received at recording time does not change. You can therefore use the output of a request as input for another request.

However, HTML pages often contain dynamic data, such as:

- a URL that changes each time you access the associated Web page
- a field (sometimes hidden) recorded during a form submission
- session IDs

When you record a script using HTTP mode, the dynamic data is recorded, but cannot be re-used during replay because the output is different from what it was when the test was recorded.

To record and replay tests in HTTP mode, you may need to correlate the generated values from application servers. Correlation enables the Virtual User Recorder to capture the data from the output of one request and use it as input to another.

Astra LoadTest can correlate dynamic data within known application environments. You can instruct Astra LoadTest to correlate dynamic information by setting testing options before you record a test in HTTP mode.

Find
Find Again
Top of Chapter
Back

Setting HTTP Testing Options

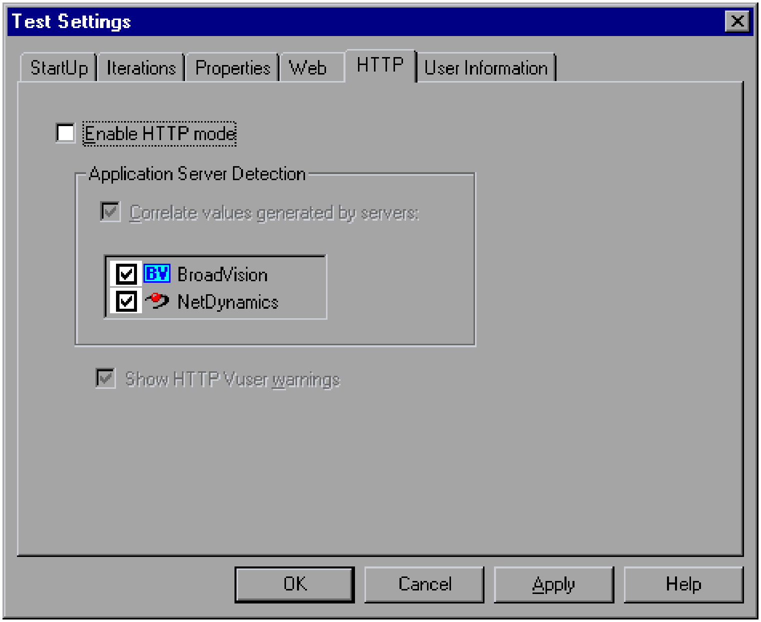
Before recording a test for HTTP play back, you must set the HTTP testing options from the **Test Settings** dialog box. The values you set remain in effect for all tests.

To set the HTTP testing options:

1 Choose **Test > Settings**. The **Test Settings** dialog box opens.

Find
Find Again
Top of Chapter
Back

2 Select the **HTTP** tab.



□ Enable HTTP mode

Application Server Detection

☑ Correlate values generated by servers:

☑ BroadVision

☑ NetDynamics

☑ Show HTTP Vuser warnings

3 Enable HTTP mode, correlation options, and Vuser warnings as shown.

Find
Find Again
Top of Chapter
Back

The HTTP tab includes the following options:

Option	Description
Enable HTTP Mode	Instructs Astra LoadTest to record in HTTP mode.
Correlate values generated by servers:	Enables automatic detection of known Application Servers and correlates dynamic information to allow successful test runs.
BroadVision	Instructs Astra LoadTest to correlate values generated from a Broadvision server.
NetDynamics	Instructs Astra LoadTest to correlate values generated from a NetDynamics server.
Show HTTP Vuser warnings	Enables the suppression of HTTP Vuser warnings.

Find
Find Again
◀ ▶
Top of Chapter
Back

Setting HTTP Run-Time Options

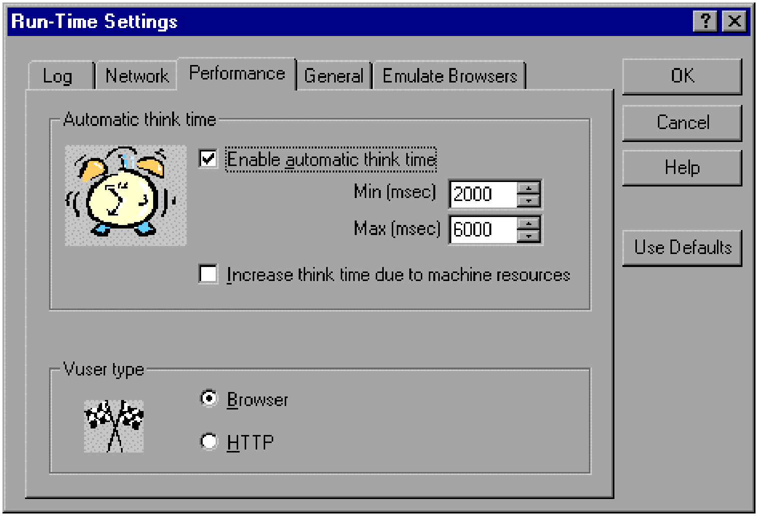
To enable your test to run in HTTP mode, you must modify your run-time settings.

To set HTTP run-time options:

1 Choose **Test > Run-Time Settings**.

The **Run-Time Settings** dialog box opens. It is divided by subject into seven tabbed pages.

2 Select the **Performance** tab.



Run-Time Settings

Log | Network | Performance | General | Emulate Browsers

Automatic think time

☑ Enable automatic think time

Min (msec) 2000

Max (msec) 6000

□ Increase think time due to machine resources

Vuser type

● Browser

○ HTTP

Find
Find Again
Top of Chapter
Back

3 Select **HTTP** from the Vuser type box.

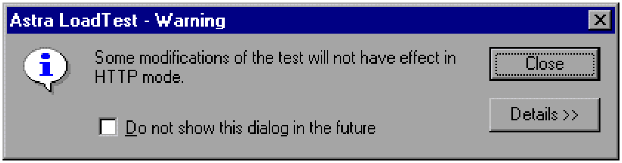
4 When you are done, click **OK** to apply your changes and close the dialog box.

Note: An HTTP Vuser can only be run using a test has been recorded in HTTP mode.

Find
Find Again
Top of Chapter
Back

About HTTP Vuser Warnings

Browser mode has certain features which HTTP mode does not possess. When you save HTTP scripts that have been recorded with these Browser mode features, the following warning message appears:



Some modifications of the test will not have effect in HTTP mode.

□ Do not show this dialog in the future

Click the Details button for an explanation of the problem Astra LoadTest has encountered with your script. The following is a list of reasons why the test may not run effectively in HTTP mode:

Description
This kind of parameterization has no effect in HTTP mode. Reasons: 1) The element of HTML form is not recognized. 2) HTML form is not submitted..
Parameterization of property X is not supported in HTTP mode.
Expressions are not allowed in HTTP mode.
Regular expressions are not supported in HTTP mode (property X).
This parameter will be ignored.
Checkpoints are not supported in HTTP mode.
Output parameters are not supported in HTTP mode.

Find
Find Again
Top of Chapter
Back

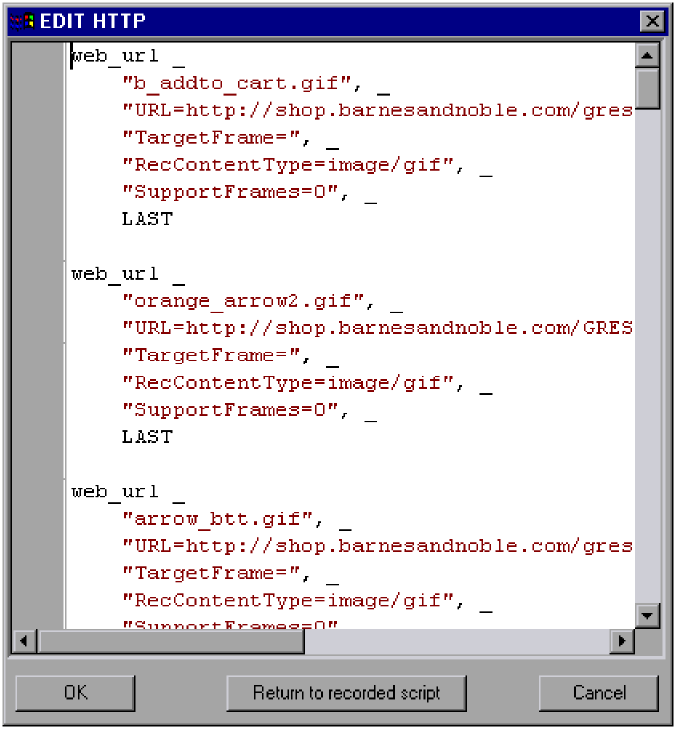
Editing Your HTTP Script

It is often necessary to add a step to your test or modify the script that you recorded. In HTTP mode, you can edit your script before running your test using simple Visual Basic programming commands.

Find
Find Again
Top of Chapter
Back

To edit an HTTP script:

1 In the Tree View, right-click the step that you want to modify and select **Edit the HTTP Script**. The **EDIT HTTP** dialog box opens.

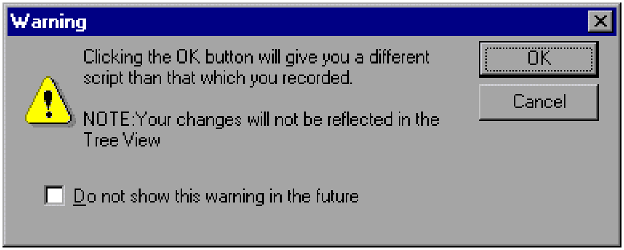


```
web_url _
     "b_addto_cart.gif", _
     "URL=http://shop.barnesandnoble.com/gres
     "TargetFrame=", _
     "RecContentType=image/gif", _
     "SupportFrames=0", _
     LAST

web_url _
     "orange_arrow2.gif", _
     "URL=http://shop.barnesandnoble.com/GRES
     "TargetFrame=", _
     "RecContentType=image/gif", _
     "SupportFrames=0", _
     LAST

web_url _
     "arrow_btt.gif", _
     "URL=http://shop.barnesandnoble.com/gres
     "TargetFrame=", _
     "RecContentType=image/gif", _
     "SupportFrames=0"
```

Find
Find Again
Top of Chapter
Back

2 Edit the line or lines that you want to modify.

3 When you have finished editing your script, click **OK**. The following warning message appears:



Warning

Clicking the OK button will give you a different script than that which you recorded.

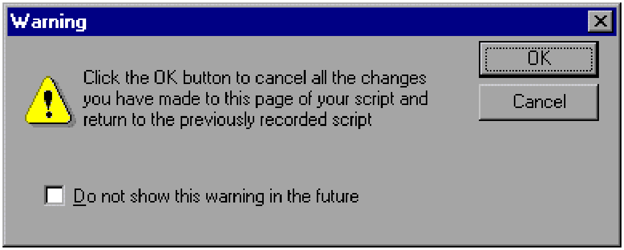
NOTE:Your changes will not be reflected in the Tree View

□ Do not show this warning in the future

4 Click **OK** to proceed with the modification of your script. Your script now differs from the one which you recorded and is no longer reflected in the Tree View. The automatic synchronization between the Browser mode script and your HTTP script is discontinued and the portion of the script which you modified will no longer be correlated with the Browser mode script.

Find
Find Again
Top of Chapter
Back

5 If, while editing your script, you want to cancel your modifications, click the **Return to recorded script** button. The following warning message appears:



Click the OK button to cancel all the changes you have made to this page of your script and return to the previously recorded script

□ Do not show this warning in the future

6 Click **OK** to cancel all the changes you have made to this page of your script and return to the previously recorded script. The automatic synchronization between the Browser mode script and your HTTP script will continue and the two scripts will remain correlated.

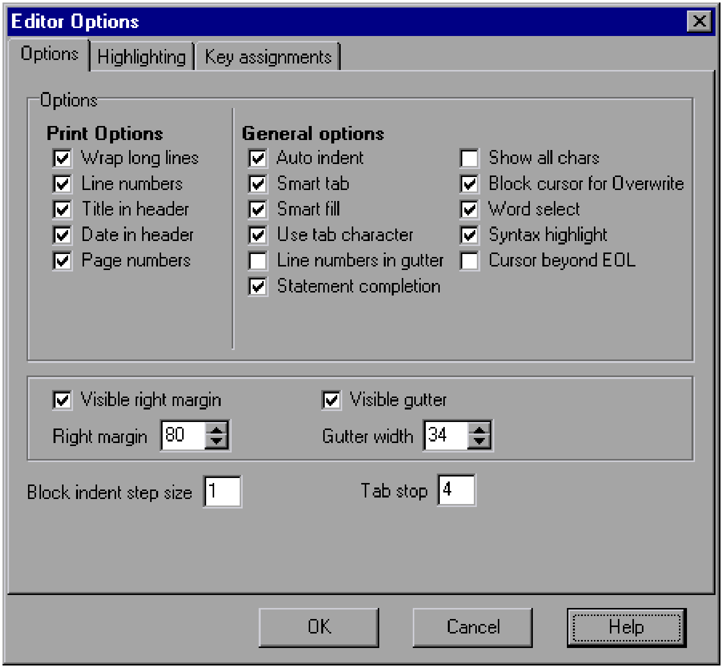
Find
Find Again
Top of Chapter
Back

Setting HTTP Script Editor Options

The HTTP Script Editor displays your VB Script using a set of default options. You change the appearance of your script by setting **Editor Options**.

To set Editor Options:

1 Select **Tools > Editor Options**. The **Editor Options** dialog box opens.



Options

Print Options
- ☑ Wrap long lines
- ☑ Line numbers
- ☑ Title in header
- ☑ Date in header
- ☑ Page numbers

General options
- ☑ Auto indent
- ☑ Smart tab
- ☑ Smart fill
- ☑ Use tab character
- ☐ Line numbers in gutter
- ☑ Statement completion
- ☐ Show all chars
- ☑ Block cursor for Overwrite
- ☑ Word select
- ☑ Syntax highlight
- ☐ Cursor beyond EOL

☑ Visible right margin

Right margin: 80

☑ Visible gutter

Gutter width: 34

Block indent step size: 1

Tab stop: 4

Find
Find Again
Top of Chapter
Back

The Editor Options dialog box includes the following tabs:

Tab	Description
Options	Options that determine the page layout. These include print, tab and margin parameters.
Highlighting	Options that determine the visual appearance of the page. These include background and text colors, as well as font information.
Key Assignments	Options that allow you to assign Hot Key status.

For further information on Script editing, see Chapter 22, **Customizing the Test Script Editor**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Creating LoadRunner Tests

The Virtual User Recorder allows you to convert your HTTP scripts for use with LoadRunner.

To convert your Astra LoadTest test to a LoadRunner script:

1 Open the desired test in the Virtual User Recorder.

2 Select **Tools > Create LoadRunner Test**. Your test is automatically generated.

3 Save your test to the desired location.

Find
Find Again
◀ ▶
Top of Chapter
Back

If the Virtual User Recorder does not record all the events you need, you can configure the events you want to record for each type of Web object.

This chapter describes:

Find
Find Again
Top of Chapter
Back

About Configuring Event Recording

The Virtual User Recorder records your test by recording the *events* you perform on your Web site. An event is a notification that occurs in response to an action, such as a change in state, or as a result of the user clicking the mouse or pressing a key while viewing the document. You may find that you need to record more or fewer events than the Virtual User Recorder automatically records by default. You can modify the default event recording settings by using the Event Configuration Settings dialog box to select one of three standard configurations, or you can customize the individual event recording configuration settings to meet your specific needs.

For example, the Virtual User Recorder does not generally record mouseover events on link objects. If, however, you have a mouseover *behavior* connected to a link, it may be important for you to record the mouseover event. In this case, you could customize the configuration to record mouseover events on link objects only if they are connected to a behavior.

Note that event configuration is a global setting and therefore affects all tests that are recorded after you change the settings.

Note: Changing the event configuration settings does not affect tests that have already been recorded. If you find that the Virtual User Recorder recorded more or less than you need, change the event recording configuration and then re-record the part of your test that is affected by the change.

Find
Find Again
◀ ▶
Top of Chapter
Back

Selecting a Standard Event Recording Configuration

By default, the Virtual User Recorder uses the *Basic* recording configuration level. If the Virtual User Recorder does not record all the events you need, you may require a higher event configuration level.

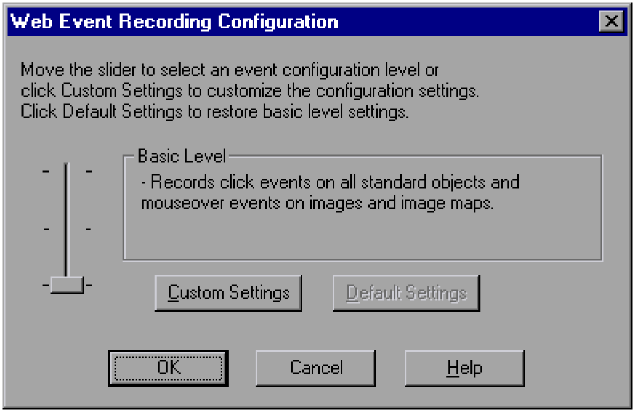
The Event Configuration Settings dialog box offers three standard event configuration levels.

Level	Description
Basic	**Default** - Always records click events on standard web objects. - Always records reset and submit events within forms. - Records click events on other objects with a handler or behavior connected. - Records the event following a mouseover event on images and image maps.
Medium	Records click events on the <DIV>, <SPAN>, and <TD> HTML tag objects in addition to the objects recorded in the basic level.
High	Records mouseover, mousedown, and double-click events on objects with *handlers* or *behaviors* attached in addition to the objects recorded in the basic level. For more information on handlers and behaviors, see **Listening Criteria** on page 297.

Find
Find Again
Top of Chapter
Back

To set a standard event recording configuration:

1 Choose **Tools > Web Event Recording Configuration**. The **Web Event Recording Configuration Settings** dialog box opens.



Web Event Recording Configuration

Move the slider to select an event configuration level or click Custom Settings to customize the configuration settings. Click Default Settings to restore basic level settings.

Basic Level

- Records click events on all standard objects and mouseover events on images and image maps.

2 Use the slider to select your preferred standard event recording configuration.

3 Click **OK**.

Find
Find Again
Top of Chapter
Back

Customizing the Event Recording Configuration

If the standard event configuration levels do not exactly match your recording needs, you can customize the event recording configuration using the Custom Event Configuration dialog box.

The Custom Event Configuration dialog box enables you to customize event recording in several ways. You can:

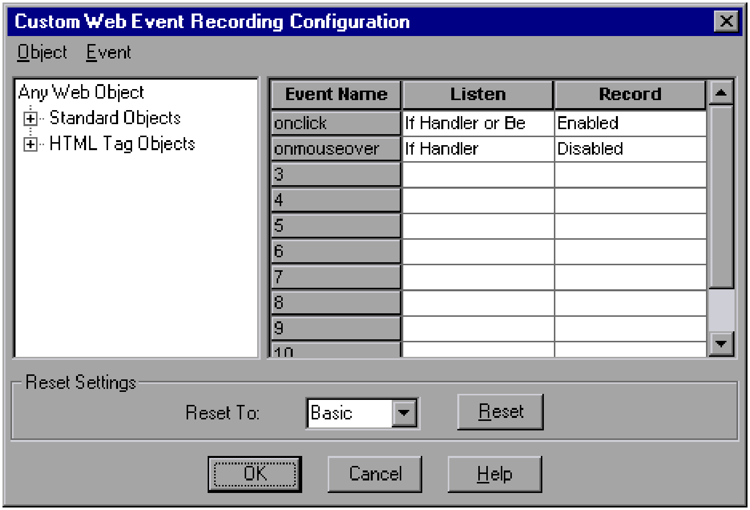
- add or delete objects to which the Virtual User Recorder should apply special listening or recording settings.
- add or delete events to which the Virtual User Recorder should listen for all objects.
- add or delete events to which the Virtual User Recorder should listen for one or more specific objects.
- modify the listening or recording settings of an event to which the Virtual User Recorder listens for all objects.
- modify the listening or recording settings of an event for one or more specific objects.

Find
Find Again
◀ ▶
Top of Chapter
Back

To customize the event recording configuration:

1 Choose **Tools > Web Event Recording Configuration**. The **Web Event Recording Configuration** dialog box opens.

2 Click the **Custom Settings** button. The **Custom Web Event Recording Configuration** dialog box opens.



Any Web Object
- Standard Objects
- HTML Tag Objects

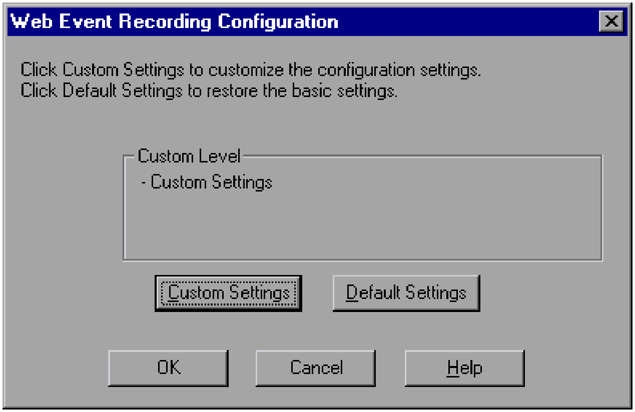
Event Name	Listen	Record
onclick	If Handler or Be	Enabled
onmouseover	If Handler	Disabled
3		
4		
5		
6		
7		
8		
9		
10		

Reset To: Basic

3 Set the event recording configuration options you want. The sections below describe the event recording configuration options in detail.

Find
Find Again
Top of Chapter
Back

4 Click **OK**. The Custom Event Configuration dialog box closes. The slider on the Event Configuration Settings dialog box disappears and the configuration description displays: Custom Settings.



Click Custom Settings to customize the configuration settings.
Click Default Settings to restore the basic settings.

Adding and Deleting Objects in the Custom Configuration Object List

The Custom Event Configuration dialog box lists objects in an object hierarchy. The top of the hierarchy is Any Web Object. The settings for Any Web Object apply to any object on the Web page being tested, for which there is no specific settings. Below this are the Standard and HTML Tag Objects categories, each of which contains a list of objects.

Find
Find Again
Top of Chapter
Back

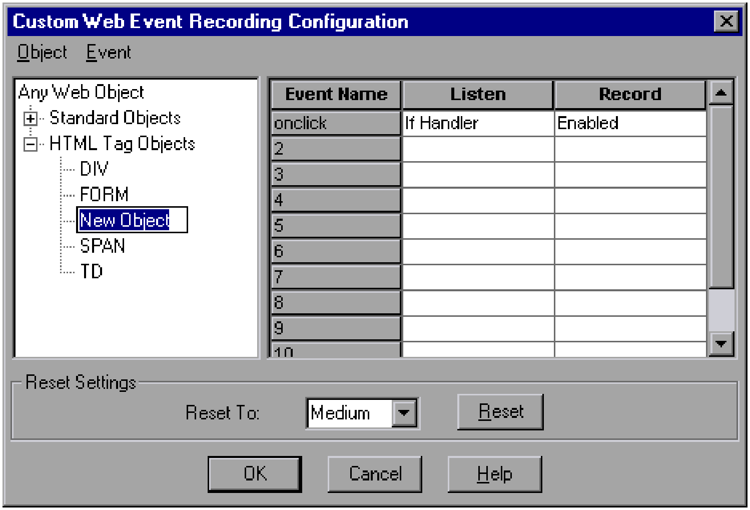
When working with the objects in the Custom Event Configuration dialog box, keep the following principles in mind:

- If an object is listed in the Custom Event Configuration dialog box, then the settings for that object override the settings for Any Web Object.
- The Virtual User Recorder always listens to the objects listed under standard objects. Therefore, you cannot delete or add to the objects in this category.
- You can add any HTML Tag object in your web page to the HTML Tag Objects category.

Find
Find Again
Top of Chapter
Back

To add objects to the event configuration object list:

1 From the Custom Event Configuration dialog box, choose **Object > Add**. A “New Object” object appears in the HTML Tag Objects list.



Event Name	Listen	Record
onclick	If Handler	Enabled
2		
3		
4		
5		
6		
7		
8		
9		
10		

Reset To: Medium

2 Click **New Object** to rename it. Enter the exact HTML Tag name.

By default the new object is set to listen and record *onclick* events with handlers attached.

For more information on adding or deleting events, see **Adding and Deleting Listening Events for an Object** on page 295.

For more information on listening and recording settings, see **Modifying the Listening and Recording Settings for an Event** on page 297.

Find
Find Again
Top of Chapter
Back

To delete objects from the HTML Tag Objects list:

1 From the Custom Event Configuration dialog box, select the object in the HTML Tag Objects category that you want to delete.

2 Choose **Object > Delete**. The object is deleted from the list.

Note: You cannot delete objects from the standard objects category.

Adding and Deleting Listening Events for an Object

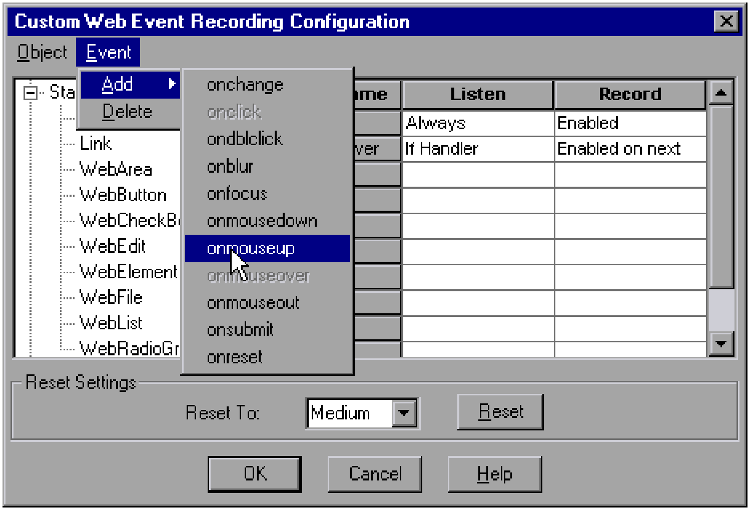
You can modify the list of events that trigger the Virtual User Recorder to listen to an object.

To add listening events for an object:

1 From the Custom Event Configuration dialog box, select the object to which you want to add the event, or select **Any Web Object**.

Find
Find Again
◀ ▶
Top of Chapter
Back

2 Choose **Event > Add**. A list of available events opens.



me	Listen	Record
	Always	Enabled
ver	If Handler	Enabled on next

Reset To: Medium

3 Select the event you want to add. The event appears in the Event Name column in alphabetical order. By default, the event is set to record when a handler is attached to the object.

For more information on listening and recording settings, see **Modifying the Listening and Recording Settings for an Event** on page 297.

Find
Find Again
Top of Chapter
Back

To delete listening events for an object:

1. From the Custom Event Configuration dialog box, select the object from which you want delete an event, or select **Any Web Object**.
2. Select the event you want to delete from the Event Name column.
3. Choose **Event > Delete**. The event is deleted from the Event Name column.

Modifying the Listening and Recording Settings for an Event

You can select the listening criteria and set the recording status for each event listed for each object.

Listening Criteria

For each event, you can instruct the Virtual User Recorder to listen every time the event occurs on the object, if an event handler is attached to the event and/or if a DHTML behavior is attached to the event.

An event *handler* is code in a web page, typically a function or routine written in a scripting language, that receives control when the corresponding event occurs.

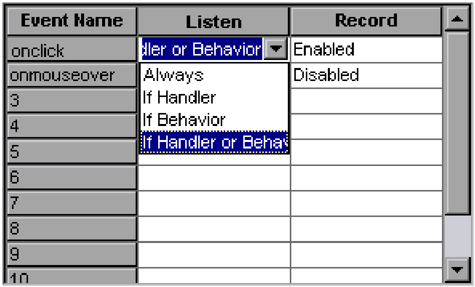
A DHTML *behavior* is a simple, lightweight component that encapsulates specific functionality or behavior on a page. When applied to a standard HTML element on a page, a behavior enhances that element's default behavior.

Find
Find Again
Top of Chapter
Back

To specify the listening criterion for an event:

1 From the Custom Event Configuration dialog box, select the object for which you want to modify the listening criterion or select **Any Web Object**.

2 In the row of the event you want to modify, select the listening criterion you want from the Listen column.



Event Name	Listen	Record
onclick	dler or Behavior	Enabled
onmouseover		Disabled
3		
4		
5		
6		
7		
8		
9		
10		

You can select **Always, If Handler**, **If Behavior**, or **If Handler or Behavior**.

Find
Find Again
Top of Chapter
Back

Recording Status

For each event to which the Virtual User Recorder listens, you can enable recording, disable recording, or enable recording only if the next event is dependant on this event.

- **Enabled** - records the event each time the listening criterion is met.
- **Disabled** - does not record the specified event and ignores event *bubbling* where applicable.

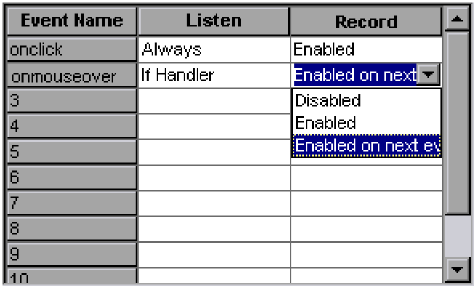
  Bubbling is the process whereby, when an event occurs on a child object, the event can travel up the chain of hierarchy within the HTML code until it encounters an event handler to process the event.
- **Enabled on next** - records the event only if the subsequent event occurs on the same object and is dependant on this event. For example, suppose a mouseover behavior modifies an image link. You may not want to record the mouseover event each time you happen to move the mouse over this image. Because only the image that appears after the mouseover event enables the link event, however, it is essential that the mouseover event is recorded before a click event on the same object. This option applies only to the Image and WebArea standard objects.

Find
Find Again
Top of Chapter
Back

To set the recording status for an event:

1 From the Custom Event Configuration dialog box, select the object for which you want to modify the recording status or select **Any Web Object**.

2 In the row of the event you want to modify, select the recording status you want from the Record column.



Event Name	Listen	Record
onclick	Always	Enabled
onmouseover	If Handler	Enabled on next
3		
4		
5		
6		
7		
8		
9		
10		

Find
Find Again
Top of Chapter
Back

Resetting Standard Event Recording Configuration Settings

If you want to restore standard settings after you have set Custom settings, there are two ways to reset the settings.

- You can reset the event recording configuration settings to the basic level from the Event Configuration Settings dialog box.
- You can reset the settings to any one of the standard settings from within the Custom Event Configuration dialog box so that you can begin customizing from that point.

Note: When you choose to reset standard settings, your custom settings are cleared completely.

To reset basic level configuration settings from the Event Configuration Settings dialog box:

1. Choose **Tools > Web Event Configuration**. The **Event Configuration Settings** dialog box opens.
2. Click **Default**. The standard configuration slider re-appears and all event settings are restored to the *Basic* event recording configuration level.
3. If you want to select a different standard configuration level, see **Selecting a Standard Event Recording Configuration** on page 288.

Find
Find Again
Top of Chapter
Back

To reset standard settings from the Custom Event Configuration dialog box:

1. Choose **Tools > Web Event Configuration**. The **Event Configuration Settings** dialog box opens.
2. Click the **Customize** button. The **Custom Event Configuration** dialog box opens.
3. In the **Reset To** box, select the standard event recording level you want.
4. Click **Reset**. All event settings are restored to the defaults for the level you selected.

Find
Find Again
Top of Chapter
Back

Enhancing Your Tests with Programming

After recording a test, you can use the Virtual User Recorder to enhance your test using a few simple programming techniques.

This chapter describes:

Find
Find Again
Top of Chapter
Back

About Enhancing Your Tests with Programming

When recording, a test is generated by recording the typical processes that you perform on your Web site. As you navigate through your site, the Virtual User Recorder graphically displays each *step* you perform as an icon in a *test tree*.

Once you record your test, you can increase the power and flexibility of your tests by programming. The Virtual User Recorder includes the *Function wizard*, a programming tool which helps you to quickly and easily add recordable and non-recordable functions to your test. You can use the wizard to add functions that perform operations on Web objects or retrieve information from your site. For example, you can add a step that checks that an object exists, or you can retrieve the return value of a function.

The Virtual User Recorder also enables you to incorporate decision-making into your test. You can add conditional statements to control the logical flow of your test.

In addition, you can define messages in your test that Astra LoadTest sends to your test results. To improve the readability of your tests, you can also add comments to your test.

This chapter introduces some programming concepts and shows you how to use simple programming techniques in the Tree View in order to create more powerful tests. For information on how to use programming concepts in the Expert View, see Chapter 20, **Testing in the Expert View**.

Find
Find Again
Top of Chapter
Back

Inserting Functions

After recording, you can add additional functions to your tests using the Function wizard. With the wizard you can add recordable and non-recordable functions that perform operations on objects or retrieve information from your site. For example, the **QueryValue** function enables you to query the method argument value. You can use the return value of the function as an output parameter or as part of a conditional statement.

To insert a function in a test:

1 In the Tree View, click a step in the test tree. A menu opens. Choose **Insert > Step > Function**.

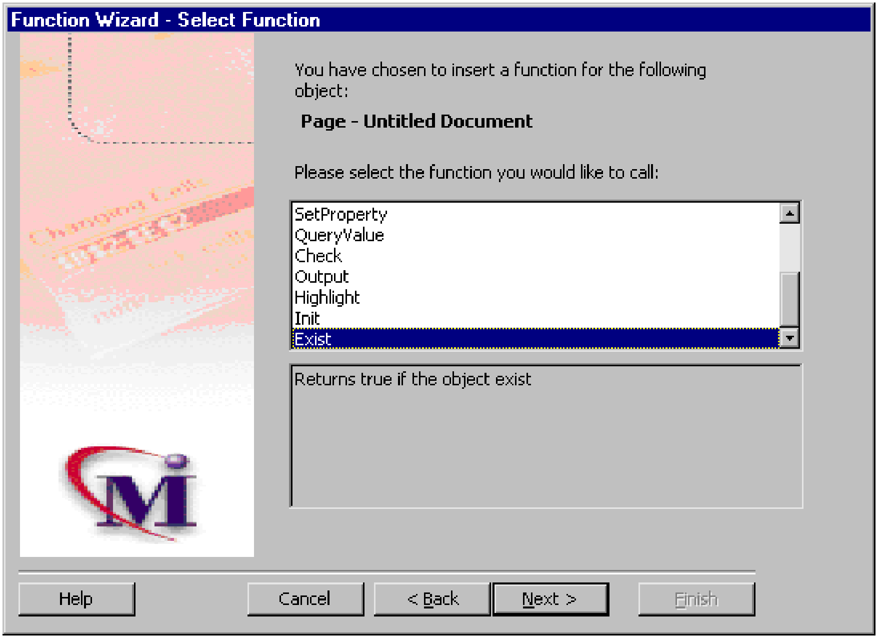
Tip: To add a function from the Expert View, click a statement in the test script. Right-click the highlighted object in the ActiveScreen and choose **Insert Function**. The **Object Selection - Insert Function** opens. Select an object and click **OK**.

---

2 The **Function Wizard - Introduction** screen opens. Click **Next**.

Find
Find Again
◀ ▶
Top of Chapter
Back

3 The **Function Wizard - Select Function** screen opens.



Function Wizard - Select Function

You have chosen to insert a function for the following object:

Page - Untitled Document

Please select the function you would like to call:

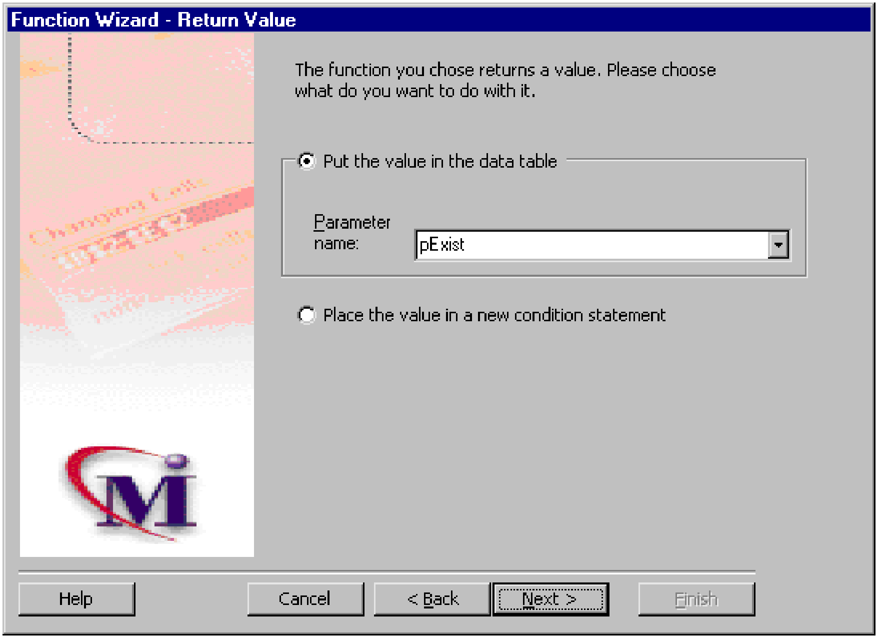
SetProperty
QueryValue
Check
Output
Highlight
Init
Exist

Returns true if the object exist

Select a function and click **Next**.

Find
Find Again
Top of Chapter
Back

4 If the function you chose returns a value, the **Function Wizard - Return Value** screen opens. Otherwise, proceed to the next step.



Function Wizard - Return Value

The function you chose returns a value. Please choose what do you want to do with it.

● Put the value in the data table

Parameter name: pExist

○ Place the value in a new condition statement

Find
Find Again
Top of Chapter
Back

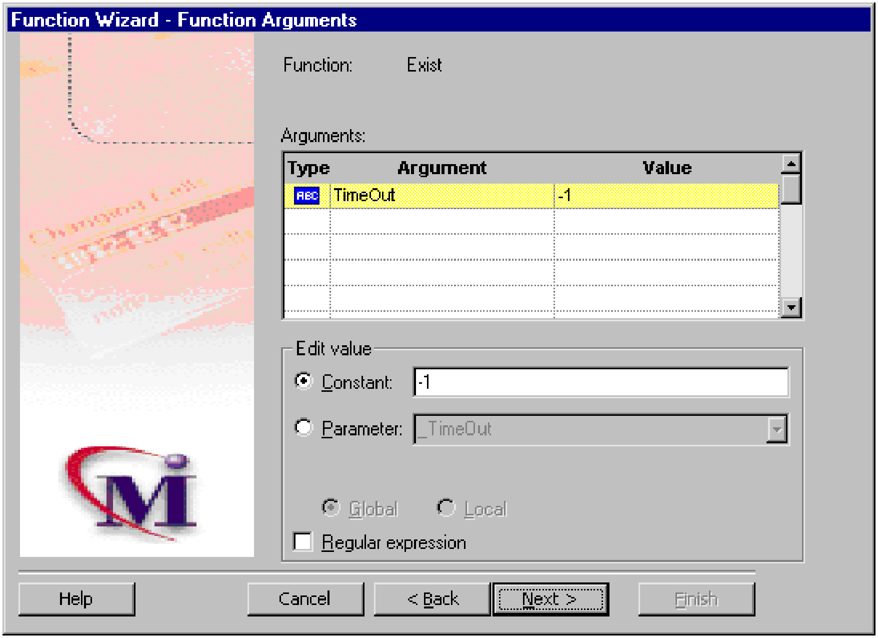
The following options are available:

Option	Description
Put the value in the data table (default)	Inserts the return value of the function as an output parameter into your Data pane. For more information, see Chapter 8, **Creating Output Parameters**.
Output parameter	Sets the name for the output parameter. You can accept the default name, select from the list, or enter a new name. For more information, see Chapter 8, **Creating Output Parameters**.
Place the value in a new condition statement	Inserts the return value of the function into a conditional statement. For more information, see **Using Conditional Statements** on page 312.

Click **Next** to continue.

Find
Find Again
Top of Chapter
Back

5 If the function you chose has function arguments, the **Function Wizard - Function Arguments** screen opens. Otherwise, proceed to the next step.



Type	Argument	Value
ABC	TimeOut	-1

● Constant: -1

○ Parameter: _TimeOut

● Global ○ Local

□ Regular expression

Find
Find Again
Top of Chapter
Back

The screen displays the arguments you can parameterize, in a pane listing the arguments, their values, and their types:

Option	Description
Type	The ABC icon indicates that the argument value is a constant. The ▦ icon indicates that the argument value is a parameter.
Argument	The name of the argument whose value will be parameterized.
Value	The value of the argument to parameterize.

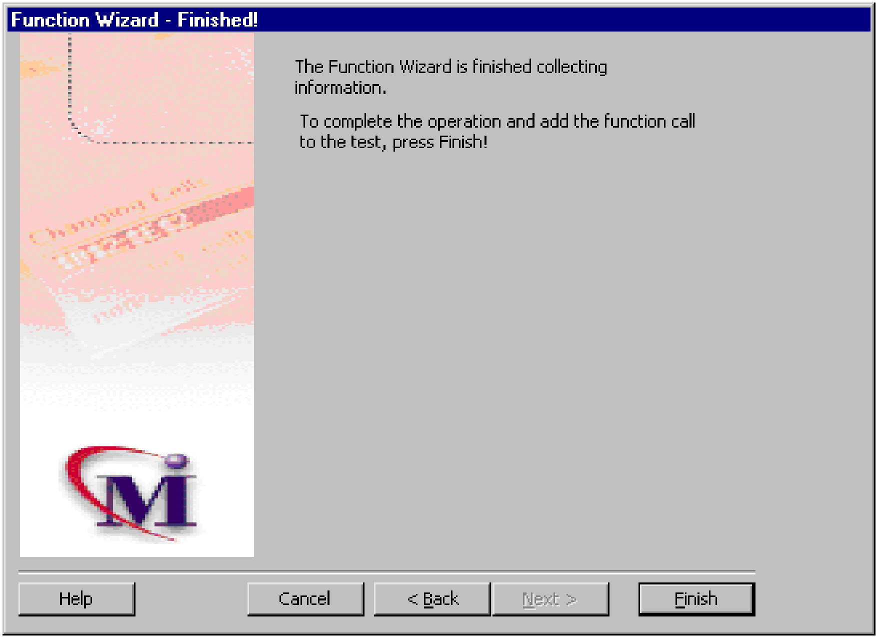
In the **Edit value** section, you use the following options to edit the argument value.

Option	Description
Constant (default)	Sets the argument value as a constant.
Parameter	Sets the argument value as a parameter. For more information, see Chapter 7, **Parameterizing Tests**.

Click **Next** to continue.

Find
Find Again
◀ ▶
Top of Chapter
Back

6 The **Function Wizard - Finished** screen opens.



The Function Wizard is finished collecting information.

To complete the operation and add the function call to the test, press Finish!

Click **Finish** to complete the process and add the function to your test.

Find
Find Again
Top of Chapter
Back

Using Conditional Statements

You can control the flow of your script with conditional statements. Using conditional statements, you can incorporate decision-making into your tests using *If...Then...Else* statements.

The *If...Then...Else* statement is used to evaluate whether a condition is true or false and, depending on the result, to specify one or more statements to run. Usually the condition is an expression that uses a comparison operator to compare one value or variable with another. The following comparison operators are available: *less than* <, *less than or equal to* <=, *greater than* >, *greater than or equal to* >=, *not equal* <>, *and equal* =.

Your *If...Then...Else* statement can be nested to as many levels as you need. It has the following syntax:

If *condition* **Then** *statements* [**Else** elsestatements]

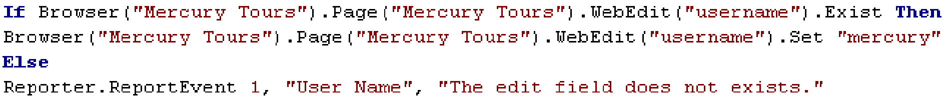
Or, you can use the block form syntax:

```
If condition Then
    [statements]
[ElseIf condition-n Then
    [elseifstatements]] . . .
[Else
    [elsestatements]]
End If
```

Find
Find Again
Top of Chapter
Back

Part of Statement	Description
condition	One or more expressions that evaluate to true or false. If the condition is null, it is treated as false.
statements	One or more statements, separated by colons, that are executed if the condition is true.
condition-n	Same as *condition*.
elseifstatements	One or more statements, separated by colons, that are executed if the associated *condition-n* is true.
elsestatements	One or more statements, separated by colons, that are executed if no previous *condition-n* expression is true.

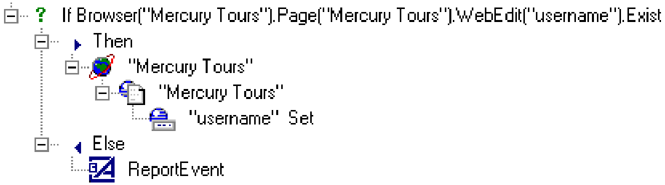
For example, the statement below (as it appears in the Expert View) checks that the user name edit box exists in the Mercury Tours site. **If** the edit box exists, **then** a user name is entered; **else** a message is sent to test results.



```
If Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Exist Then
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Set "mercury"
Else
Reporter.ReportEvent 1, "User Name", "The edit field does not exists."
```

Find
Find Again
Top of Chapter
Back

The same example is displayed in the Tree View as follows:



```
? If Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Exist
    ▶ Then
        "Mercury Tours"
            "Mercury Tours"
                "username" Set
    ◀ Else
        ReportEvent
```

In the test tree, the following icons are used to indicate the different levels of *If...Then...Else* statements:

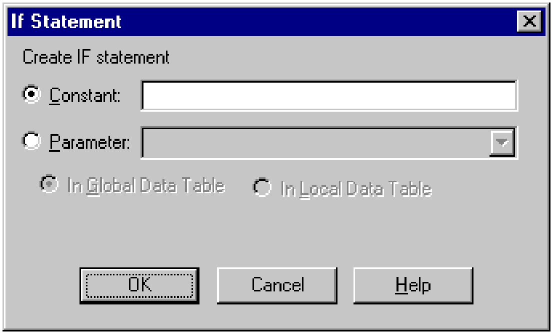
Icon	Description
?	Starts an *If* statement.
▶	Starts a *Then* statement.
▶?	Starts an *Elseif* statement.
◀	Starts an *Else* statement.

To add a conditional statement:

1 In the Tree View, click a step in the test tree.

Find
Find Again
Top of Chapter
Back

2 Choose **Insert > Step > If...Then...Else > If...Then**. The **If Statement** dialog box opens.


If Statement

Create IF statement

● Constant:

○ Parameter:

● In Global Data Table ○ In Local Data Table

3 Set the expression as a constant or a parameter. Note that the expression must be a Boolean expression.

- To set the expression as a constant, select **Constant**, and type the expression in the box. For example, type i>5.
- To set the expression as a parameter, select **Parameter**, and choose a parameter from the list or enter a new parameter. For example, type a>c for a formula in a table. For more information, see Chapter 7, **Parameterizing Tests**.
- To add the parameter to the Global tab in the Data pane, select **Global**. To add the parameter to the Action tab, select **Local**. For more information, see Chapter 7, **Parameterizing Tests**.

Find
Find Again
Top of Chapter
Back

4. Click **OK** to close the dialog box.
5. To complete the **Then** statement you can:
   - Record a new step and then use the Cut/Paste commands to add it to your **Then** statement.
   - Copy an existing step and paste it in your **Then** statement.
   - Click and drag a step to move it to your **Then** statement.
6. To nest an additional level to your statement, click the **Then** statement and choose one of the following options:

To add:	**Choose:**
an **If** statement	**Insert > Step > If...Then...Else > If...Then**
an **Elseif** statement	**Insert > Step > If...Then...Else > Elseif...Then**
an **Else** statement	**Insert > Step > If...Then...Else > Else**

To complete the new statement you can:

- Record a new step and then use the Cut/Paste commands to add it to your statement.
- Copy an existing step and paste it in your statement.
- Click and drag a step to move it to your statement.

Find
Find Again
◀ ▶
Top of Chapter
Back

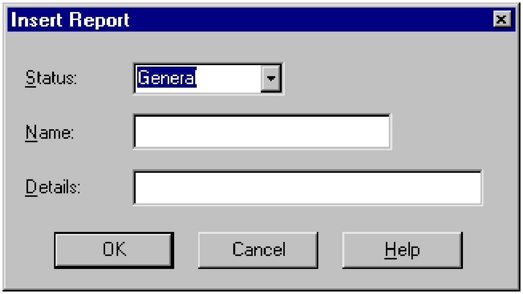
Sending Messages to Your Test Results

You can define a message in your test that Astra LoadTest sends to your test results. For example, suppose your want to check that a password edit box exists in the Mercury Tours site. If the edit box exists, then a password is entered. Otherwise, Astra LoadTest sends a message to the test results indicating that the object is not found.

To send a message to your test results:

1 In the test tree, click a step. A menu opens.

2 Choose **Insert > Step > Report**. The **Insert Report** dialog box opens.



Find
Find Again
Top of Chapter
Back

3 Select a status from the **Status** list.

Status	Description
Passed	sends a message if the step passes
Failed	sends a message if the step fails
General	sends a message regardless of the status of the step

4 In the **Name** box, type a step name to associate with the message. For example, “Password edit box”.

5 In the **Details** box, type a message to insert in your test results. For example, “Password edit box does not exist”.

6 Click **OK**.

After your run the test, the ⚠ icon indicates in the Test Results window indicates the message sent.

Find
Find Again
◀ ▶
Top of Chapter
Back

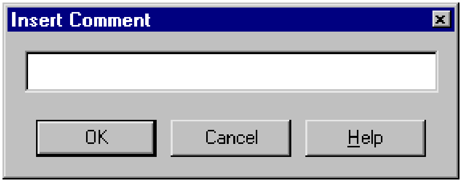
Adding Comments

While programming, you can add comments to your tests. A *comment* is an explanatory remark in a program. When you run a test, Astra LoadTest does not process comments. Use comments to explain sections of a test in order to improve readability and to make tests easier to update.

To add a comment:

1 In the test tree, click a step. A menu opens.

2 Choose **Insert > Step > Comment**. The **Insert Comment** dialog box opens.



3 Type a comment and click **OK**.

A comment statement is added to your test. If you are working in Tree View, the ✎ icon indicates a comment. In the Expert View, a comment is specified as *Rem.*

Find
Find Again
Top of Chapter
Back

Advanced Features

Testing in the Expert View

In the Virtual User Recorder, test scripts are composed of statements coded in Microsoft's programming language, VBScript. This chapter provides a brief introduction to VBScript and shows you how to enhance your test scripts using a few simple programming techniques.

This chapter describes:

- **Understanding the Expert View**
- **Programming in the Expert View**
- **Enhancing Tests with Comments, Calculations, and Control-Flow Statements**

Find
Find Again
◀ ▶
Top of Chapter
Back

About Testing in the Expert View

The Expert View provides an alternative to the Tree View for testers who are familiar with VBScript. In the Expert View, you can view the recorded test in VBScript and enhance it with programming.

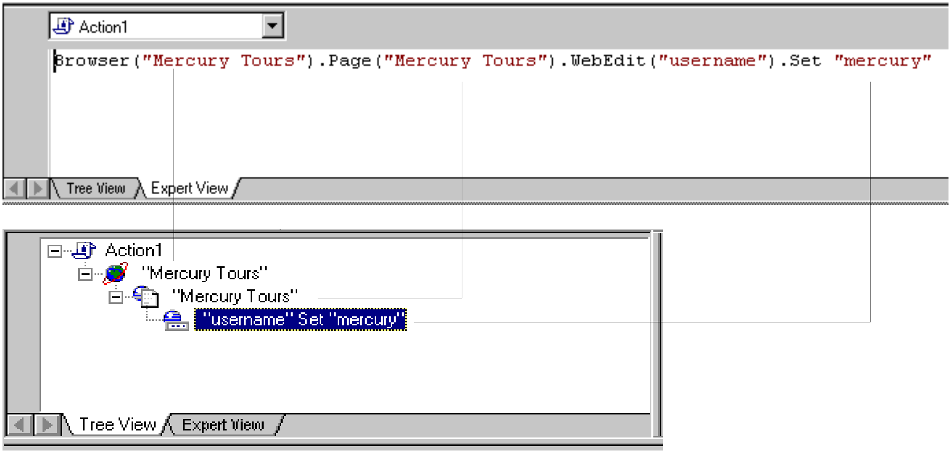
In the Expert View you can also add functions manually instead of from the Function wizard. For information on using the Function wizard, see Chapter 19, **Enhancing Your Tests with Programming**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Understanding the Expert View

The Virtual User Recorder can display tests you record in two formats:

- the Tree View, where the Virtual User Recorder displays the object hierarchy in an icon-based tree
- the Expert View, where the Virtual User Recorder displays the object hierarchy in VBScript



```
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Set "mercury"
```

Note that in the diagram above, the object hierarchy is identical in both views.

Find
Find Again
Top of Chapter
Back

Each line of VBScript in the Expert View represents a step in the test. The example above represents a step in the test in which the user inserts the name “mercury” into an edit field. The hierarchy of the step enables you to see the name of the site, the name of the page, the name of the object in the page, and the name of the method performed on the object. When you record your test, the Virtual User Recorder records the operations you perform on your Web site in terms of the objects on the page. It identifies the objects on a Web page by specific names (e.g. a button or a list) and the function performed on the object (e.g. click or select).

To further understand how a step in the Expert View corresponds with a step in the Tree View, examine the table below:

Tree View	Expert View	Description
"Mercury Tours"	Browser (“Mercury Tours”)	The name of the Web site is “Mercury Tours”.
"Mercury Tours"	Page (“Mercury Tours”)	The name of the current page in the Web site is “Mercury Tours”.
"username"	WebEdit(“username”)	The name of the edit field upon which the action is performed is “username”.
Set "mercury"	Set “mercury”	The name of the function performed on the edit box is “Set”. The name inserted into the edit box is “mercury”.

Find
Find Again
◀ ▶
Top of Chapter
Back

An object's logical name appears in parentheses following the object type. In the following example, the object type is Browser, and the logical name of the Browser is "Mercury Tours":

```
Browser ("Mercury Tours")
```

The object types in the object hierarchy are separated by a period. In the following example, Browser and Page are two separate objects:

```
Browser("Mercury Tours"). Page("Mercury Tours")
```

The function performed on the object always appears at the end of the line of script. In the following example, the word "mercury" is inserted in the "username" edit box using the **Set** function:

```
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Set
    "mercury"
```

For a complete list of objects and their associated functions, choose **Help** > **Function Reference** to open the *Astra Function Reference*.

Find
Find Again
Top of Chapter
Back

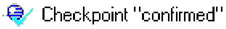
Checkpoints

In the Virtual User Recorder, you create checkpoints on pages, text strings, and objects. When you create a checkpoint in the Tree View, the Virtual User Recorder creates a corresponding line in VBScript in the Expert View. It uses the **Check** function to perform the checkpoint.

For example, in the following statement the Virtual User Recorder performs a check on the word “confirmed”:

```
Browser("Mercury Tours").Page("Flight Confirmation").
    Check Checkpoint("confirmed")
```

The corresponding step in the Tree View appears as follows:



Checkpoint "confirmed"

Note: You cannot insert checkpoints into your test while in the Expert View. Use the Tree View to insert and modify checkpoints. For more information on inserting and modifying checkpoints, see Chapter 4, **Creating Checkpoints**.

Find
Find Again
Top of Chapter
Back

Parameters

You can use the Virtual User Recorder to enhance your tests by parameterizing values in the test. A *parameter* is a variable that is assigned a value from outside the test in which it is defined. When you create a parameter in the Tree View, the Virtual User Recorder creates a corresponding line in VBScript in the Expert View.

The Virtual User Recorder calls the values of a parameterized object from the data table using the following syntax:

function_name **DataTable (** *parameterID***,** *sheetID* **)**

function_name	The name of the function that the Virtual User Recorder executes on the parameterized object.
DataTable	The data table object.
parameterID	The name of the column in the data table.
sheetID	The name of the sheet. If the parameter is a global parameter, the word “global” appears.

Note: You cannot create parameters from the Expert View. Use the Tree View to create parameters. For more information on parameterization, see Chapter 7, **Parameterizing Tests**.

Find
Find Again
Top of Chapter
Back

For example, suppose you are creating a test on the Mercury Tours site, and you select “Paris” as your destination. The following statement is inserted into your test in the Expert View:

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select
"Paris"
```

Now suppose you want to parameterize the destination, and you create a “Departure” column in the data table. The previous statement is modified to the following:

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select
    DataTable("Departure",dtGlobalSheet)
```

where **Select** is the function name, DataTable is the object, Departure is the name of the column in the data table, and dtGlobalSheet is the name of the sheet in the data table.

Find
Find Again
Top of Chapter
⇦ Back

Programming in the Expert View

The Expert View displays in VBScript the steps you executed while recording your test. After you record your test, you can increase the power and flexibility of your test by adding recordable and non-recordable VBScript statements. You can add statements that perform operations on objects or retrieve information from your site. For example, you can add a step that checks that an object exists, or you can retrieve the return value of a function.

Most objects have corresponding functions. For example, the **Back** function is associated with the Browser object. Some functions are associated with more than one object. For example, the **Click** function may be used on the WebArea object as well as the WebButton object.

The objects in the Virtual User Recorder are divided by environment.

The Virtual User Recorder environments include Web objects, DataTable objects and Utility objects.

Find
Find Again
Top of Chapter
Back

Functions Associated with Web Objects

The following table summarizes the web objects and the functions with which they are associated:

Object	Description	Functions
Browser	the site	**AddCookie**, **Back**, **Check**, **Close**, **Exist**, **Forward**, **FullScreen**, **GetProperty**, **Highlight**, **Home**, **Init**, **Navigate**, **QueryValue**, **Set**, **SetProperty**, **Stop**, **Sync**
Page	the page	**Check**, **EndTransaction**, **Exist**, **GetProperty**, **Highlight**, **Init**, **QueryValue**, **SetProperty**, **StartTransaction**, **Sync**
Image	an image with or without a target URL	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **Output**, **QueryValue**, **SetProperty**
Link	a hypertext link	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **Output**, **QueryValue**, **SetProperty**
WebArea	a client-side image map	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **QueryValue**, **SetProperty**

Find
Find Again
Top of Chapter
Back

Object	Description	Functions
WebButton	a button	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **Output**, **QueryValue**, **SetProperty**
WebCheckBox	a check box with "ON" and "OFF" states	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **QueryValue**, **Set**, **SetProperty**
WebEdit	an edit field	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **QueryValue**, **Set**, **SetProperty**, **SetSecure**, **Submit**
WebFile	an edit field with a “Browse” button attached	**FireEvent**, **GetProperty**, **Highlight**, **QueryValue**, **Set**, **SetProperty**
WebList	a dropdown or multi-selection list	**Check**, **Click**, **DeSelect**, **Exist**, **ExtendSelect**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **QueryValue**, **Select**, **Set**, **SetProperty**

Find
Find Again
Top of Chapter
Back

Object	Description	Functions
WebRadioGroup	a radio button	**Check**, **Click**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **Output**, **QueryValue**, **Select**, **Set**, **SetProperty**
WebTable	a table	**CellText**, **CellTextByContext**, **Check**, **ChildItem**, **ChildItemCount**, **Click**, **ColumnCount**, **Exist**, **FireEvent**, **GetProperty**, **Highlight**, **Init**, **MouseOver**, **Output**, **QueryValue**, **RowCount**, **RowText**, **RowTextByContext**, **SetProperty**, **TextCellExist**, **TextRowExist**

Find
Find Again
Top of Chapter
Back

In the following example, the user inserts “mercury” in the User Name edit box while recording. The following line is recorded in the Expert View:

```
Browser ("Mercury_Tours"). Page ("Mercury_Tours"). WebEdit ("username").
    Set"Mercury"
```

Browser object	Mercury_Tours
Page object	Mercury_Tours
WebEdit object (edit box)	username

The **Set** function sets the “Mercury” text into the WebEdit object.

In the following example, the user selects “Paris” from the Departure City drop-down list while recording. The following line is recorded in the Expert View:

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select
"Paris"
```

Browser object	Mercury Tours
Page object	Find Flights
WebList object (drop-down list)	depart

The **Select** function selects Paris in the WebList object.

Note: For the syntax for Astra functions, refer to the *Astra Function Reference*. To open this, choose **Help** > **Function Reference**.

Find
Find Again
Top of Chapter
Back

Functions Associated with DataTable Objects

The following table summarizes the DataTable objects and the properties and functions with which they are associated:

Object	Description	Functions
DataTable	a data table	**GetSheetCount, GetSheet, GlobalSheet, LocalSheet, AddSheet, DeleteSheet, GetRowCount, GetCurrentRow, SetCurrentRow, SetNextRow, SetPrevRow, Import, Export, Value, RawValue**
Sheet	a sheet in the data table	**GetParameterCount, GetParameter, AddParameter, DeleteParameter, GetRowCount, GetCurrentRow, SetCurrentRow, SetNextRow, SetPrevRow, Name**
Parameter	a parameter (column) in the sheet of a data table	**Value, RawValue, ValueByRow**

Find
Find Again
Top of Chapter
Back

Functions Associated with Utility Objects

The following table summarizes the Utility objects and the properties and functions with which they are associated

Object	Description	Functions
Crypt	The object used to encrypt strings.	**Encrypt**
OptionalStep	object that causes the step to which it is attached to be bypassed, if the object referred to in the step cannot be found. A bypassed step is an optional step.	N/A
Reporter	the object used to send information to the test report	**ReportEvent**

Find
Find Again
Top of Chapter
Back

Object	Description	Functions
Setting	the object used to modify test settings during the test run.	**Add, AutomaticLinkCheck, CheckBrokenLinks, CheckHtmlContent, CheckHtmlTag, CheckImagesSource, CheckLinksUrl, CheckLoadtime, CheckNumberOfImages, CheckNumberOfLinks, DefaultLoadTime, DefaultTimeOut, LocalLinks, Remove, WebTimeOut**
WebPackage	the object used to modify test settings specific to Web objects during the test run.	**ReplayType**

Functions Not Associated with Objects

In addition to the functions discussed above, there are functions that can be inserted into the test script that are not associated with any object.

Find
Find Again
Top of Chapter
Back

Adding a Pause During a Test Run

In the following example,the user records a test on the Mercury Tours sample site, which selects an arrival city and a departure city. The following statements are recorded in the Expert View:

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").
    Select"New York"
Browser("Mercury Tours").Page("Find Flights").WebList("arrive").Select
"Paris"
```

In order to instruct the Virtual User Recorder to pause between these two steps, the user inserts the **Wait** function between the steps. The **Wait** function has the following syntax:

Wait (*seconds***)**

In order to pause the test for 10 seconds, the user inserts 10 as the value for the *seconds* argument. The script now appears as follows:

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").
    Select "New York"
wait(10)
Browser("Mercury Tours").Page("Find Flights").WebList("arrive").Select
"Paris"
```

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

When the test runs, pauses for 10 seconds between selecting the departure city and the arrival city.

Setting the Run Mode

By default, the Virtual User Recorder runs the test by events. If it is necessary to run the test, or part of the test, directly by mouse operations, you can change this setting in the test script from the Expert view with the **WebPackage ("ReplayType")** function.

To set the test to run by mouse operations, enter:

```
Setting.WebPackage("ReplayType") = 2 '
```

To return to the event run mode, enter:

```
Setting.WebPackage("ReplayType") = 1
```

Generating a Function for an Object

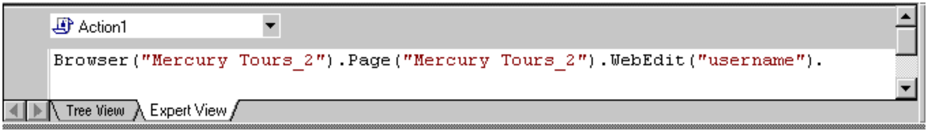
In the Expert View you can generate functions. You can also generate functions in the Tree View using the Function wizard. For additional information, see Chapter 19, **Enhancing Your Tests with Programming**.

By default, the Virtual User Recorder displays the syntax for functions as you type. You can disable or enable this **Statement Completion** option in the Editor Options dialog box. For additional information, see Chapter 22, **Customizing the Test Script Editor**.

Find
Find Again
◀ ▶
Top of Chapter
Back

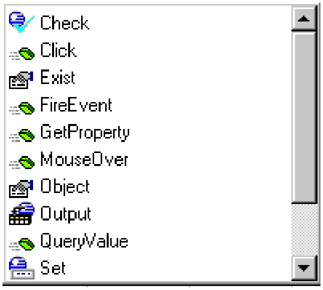
To generate a function in the Expert View:

1 In the Expert View, type a period after the object upon which you want to perform the function.



```
Browser("Mercury Tours_2").Page("Mercury Tours_2").WebEdit("username").
```

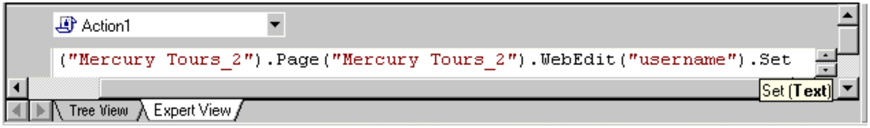
2 A list of the available functions for the object is displayed.



Double-click a function in the list. The Virtual User Recorder inserts the function into the line of script.

Find
Find Again
Top of Chapter
Back

3 If the function contains arguments, and the Statement Completion option is enabled, the Virtual User Recorder displays the syntax of the function.

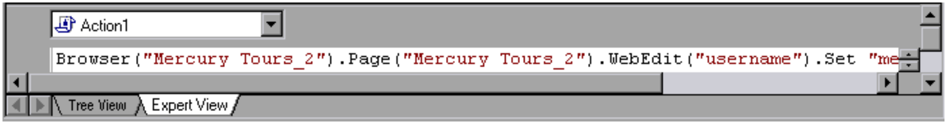


```
("Mercury Tours_2").Page("Mercury Tours_2").WebEdit("username").Set
```

```
Set (Text)
```

In the above example, the **Set** function has one argument, called *Text*. The argument name represents the text to enter in the edit box.

4 Insert an argument after the function.



```
Browser("Mercury Tours_2").Page("Mercury Tours_2").WebEdit("username").Set "me
```

Note: To find the syntax for a function, you can also refer to the *Astra Function Reference*.

Find
Find Again
Top of Chapter
Back

Enhancing Tests with Comments, Calculations, and Control-Flow Statements

The Virtual User Recorder enables you to incorporate decision-making into your test by adding conditional statements that control the logical flow of your test. In addition, you can define messages in your test that Astra LoadTest sends to your test results. To improve the readability of your tests, you can also add comments to your test.

For information on how to use these programming concepts in the Tree View, see Chapter 19, **Enhancing Your Tests with Programming**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Comments

A comment is a line or part of a line in a test script that is preceded by an apostrophe (‘). When you run a test, the Virtual User Recorder does not process comments. Use comments to explain sections of a test script in order to improve readability and to make tests easier to update. Comments are displayed in the color green. For example:

```
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").
    Set "mercury"
‘Sets the word “mercury” into the “password” edit field.
```

Note: You can also add a comment line using VBScript’s **Rem** function. For additional information, refer to the *VBScript Reference*.

Performing Calculations

You can create tests that perform simple calculations using mathematical operators. For example, you can use a multiplication operator to multiply the values displayed in two text boxes in your site. VBScript supports the following mathematical operators:

+	addition
-	subtraction
-	negation (a negative number - unary operator)
*	multiplication
/	division
^	exponent

In the following example, the multiplication operator is used to calculate the total luggage weight of the passengers at 100 pounds each.

Find
Find Again
◀ ▶
Top of Chapter
Back

```
passenger = Browser ("Mercury_Tours"). Page ("Find_Flights").
    WebEdit("numPassengers"). QueryValue("value")

‘Retrieves the number of passengers from the edit box using the QueryValue
    function

weight = passenger * 100

‘Multiplies the number of passengers by 100

msgbox("The maximum weight for the party is "& weight &"pounds.")

‘Inserts the maximum weight into a message box.
```

For...Next Statement

A **For...Next** loop instructs Astra LoadTest to execute one or more statements a specified number of times. It has the following syntax:

```
for counter = start to end [Step step]
    statement
Next
```

counter	variable used as a counter
start	the start number of the counter
end	the last number of the counter
step	number to increment at the end of each loop. The default is 1.
statement	statement to be executed during the loop

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

In the following example, the Virtual User Recorder queries the value of the “numpassengers” edit field using the QueryValue function. The Virtual User Recorder executes a loop ten times using the **For** statement. Each time the loop runs, the Virtual User Recorder increments the number of passengers by one.

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numpassengers").QueryValue("value")
For i = 1 to 10
    passengers = passengers + 1
Next
```

Find
Find Again
Top of Chapter
Back

For...Each Statement

A **For...Each** loop instructs the Virtual User Recorder to execute one or more statements for each element in an array. It has the following syntax:

```
For Each item In array
    statement
Next
```

item	a variable representing the element in the array
array	the name of the array
statement	a statement or series of statements to be executed during the loop

Do...Loop Statement

The **Do...Loop** statement instructs the Virtual User Recorder to execute a statement or series of statements while a condition is true or until a condition becomes true. It has the following syntax:

```
Do [{while}{until}condition]
    statement
Loop
```

condition	a condition to be fulfilled
statement	a statement or series of statements to be executed during the loop

In the following example, the Virtual User Recorder performs a loop until the number of passengers is more than four. Within each loop, the Virtual User Recorder increments the number by one.

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numpassengers").QueryValue("value")
do until passengers > 4
    passengers = passengers + 1
```

Find
Find Again
◀ ▶
Top of Chapter
Back

While Statement

A **While** statement instructs the Virtual User Recorder to execute a statement or series of statements while a condition is true. It has the following syntax:

```
While condition
    statement
Wend
```

In the following example, the Virtual User Recorder performs a loop using the **While** statement while the number of passengers is fewer than four. Within each loop, the Virtual User Recorder increments the number by one.

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numpassengers").QueryValue("value")
while passengers < 4
    passengers = passengers + 1
wend
```

Find
Find Again
◀ ▶
Top of Chapter
Back

If...Then...Else Statement

The **If...Then...Else** statement instructs the Virtual User Recorder to execute a statement or a series of statements based on specified conditions. If a condition is not fulfilled, the next **elseif** or **else** statement is examined. It has the following syntax:

```
If condition Then
    statement
ElseIf
    statement
Else
    statement
EndIf
```

condition	condition to be fulfilled
statement	statement to be executed

In the following example, if the number of passengers is fewer than four, the Virtual User Recorder closes the browser.

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numpassengers").QueryValue("value")
if (passengers < 4) then
    Browser("Mercury Tours").close
Else
    Browser("Mercury Tours").Page("Find Flights").Image("continue").Click
69,5
End If
```

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

The Dim Statement

The **Dim** statement is used to declared variables of all types, including strings, integers, and arrays. Use the **Dim** statement at the beginning of the procedure. It has the following syntax:

Dim *variable* **[(**_subscript_**)]**

variable	the name of the variable
subscript	the dimensions of the array

In the following example, the Dim statement is used to declare the “passengers” variable.

```
Dim passengers
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numpassengers").QueryValue("value")
```

Note: The Virtual User Recorder includes Microsoft’s *VBScript Language Reference*. The VBScript Language reference describes VBScript in detail. To open this reference, choose **Help** > **VBScript Reference**.

Find
Find Again
◀ ▶
Top of Chapter
Back

This chapter answers some of the questions that are asked most frequently by *advanced users* of Astra LoadTest. The questions and answers are divided into the following sections:

Find
Find Again
◀ ▶
Top of Chapter
Back

Recording and Running Tests

- **How does the Virtual User Recorder capture user processes?**

  The Virtual User Recorder hooks the browser (Netscape or Microsoft Internet Explorer). As the user navigates the Web site, the Virtual User Recorder intercepts and records all steps as they enter the browser. The Virtual User Recorder can then run the test by running the steps as they originally occurred.

- **How does the Virtual User Recorder record and identify objects on Web pages?**

  The Virtual User Recorder can record all Web objects on a Web page. Each HTML tag is considered a Web object. The Virtual User Recorder identifies each object by its HTML tag and logical name and stores these descriptions of each object in the memory.

Find
Find Again
◀ ▶
Top of Chapter
Back

Working with Dynamic Web Content

- **How can I record and run tests on objects that change dynamically from viewing to viewing?**

  Sometimes the content of objects in a Web page changes due to dynamic content. You can create dynamic descriptions of these objects so that Astra LoadTest will recognize them when it runs the test. For more information, see Chapter 13, **Understanding How the Virtual User Recorder Identifies Objects**.

- **How can I check that a spawned window exists (or does not exist)?**

  Sometimes a link in one window spawns another window. Use the **Exist** function to check whether or not a spawned window exists. For example:

  ```
  Brower("Window_logical_name").Exist
  ```

  For additional information about the **Exist** function, refer to the *Astra LoadTest Function Reference*.

- **How does the Virtual User Recorder record on dynamically generated URLs and Web pages?**

  The Virtual User Recorder actually clicks on links as they appear on the page. Therefore, the Virtual User Recorder records how to find a particular object, such as a link on the page, rather than the object itself. For example, if the link to a dynamically generated URL is an image, then the Virtual User Recorder records the “IMG” HTML tag, and the name of the image. This enables Astra LoadTest to find this image in the future and click on it.

Find
Find Again
◀ ▶
Top of Chapter
Back

Advanced Web Issues

- **How does Astra LoadTest handle cookies?**

  Server side connections, such as CGI scripts, can use cookies both to store and retrieve information on the client side of the connection.

  Astra LoadTest stores cookies in the memory for each Vuser, and the browser handles them as it normally would. In Astra LoadTest, a log is made every time a cookie is set or accessed.

- **How does Astra LoadTest handle session IDs?**

  The server, not the browser, handles session IDs, usually by a cookie or by embedding the session ID in all links. This does not affect **Astra LoadTest**.

- **How does Astra LoadTest handle server redirections?**

  When the server redirects the client, the client generally does not notice the redirection, and misdirections generally do not occur. In most cases, the client is redirected to another script on the server. This additional script produces the HTML code for the subsequent page to be viewed. This has no effect on **Astra LoadTest**or the browser.

- **How does Astra LoadTest handle meta tags?**

  Meta tags do not affect how the page is displayed. Generally, they contain information only about who created the page, how often it is updated, what the page is about, and which keywords represent the page’s content. Therefore, **Astra LoadTest** has no problem handling meta tags.

Find
Find Again
◀ ▶
Top of Chapter
Back

- **Does Astra LoadTest work with .asp?**

  Dynamically created Web pages that utilize ActiveX scripting have an .asp extension (Active Server Pages). This technology is completely server-side and has no bearing on **Astra LoadTest**.

- **Does Astra LoadTest work with COM?**

  Astra LoadTest complies with the COM standard.

  Astra LoadTest supports COM objects embedded in Web pages (which are currently accessible only using Microsoft Internet Explorer).

- **Does Astra LoadTest work with XML?**

  XML is eXtensible Markup Language, a pared-down version of SGML for Web documents, that enables Web designers to create their own customized tags.

  Astra LoadTest supports XML and recognizes XML tags as objects. Note that XML is not currently completely supported by Microsoft Internet Explorer and Netscape.

Find
Find Again
◀ ▶
Top of Chapter
Back

Test Maintenance

- **How do I maintain my test when my application changes?**

  The way to maintain a test when your application changes depends on how much your application changes. This is one of the main reasons you should create a small group of tests rather than one large test for your entire application. When your application changes, you can rerecord part of a test. If the change is not significant, you can manually edit a test to update it.

  You can also use **Astra LoadTest**'s action feature to design more modular and efficient tests. While recording, you divide your test into several actions, based on functionality. When your application changes, you can rerecord an action, without changing the rest of the test. For additional information, see Chapter 11, **Working with Actions**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Load Testing Questions

- **How does Astra LoadTest handle think times?**

  Think times can be added to the script at any place; they can also be adjusted. Generally random think times are between 2 and 6 seconds, this margin can be changed to whatever you like.

- Can I use the same scripts for functional/regression, load testing and monitoring? Do I need to re-record them or edit the scripts?

  Yes and no. It depends highly on how you create the scripts. It is possible to write one script that does everything, though it generally isn't recommended. We recommend that you create smaller scripts that have a specific functionality for a specific area of your Web site, then link all of these together in larger, more dynamic scripts. This allows you to do a wide variety of testing. Astra LoadTest Version 4.0, which has recently come out, has added functionality that makes modular scripts more viable than previously. Also, Astra QuickTest scripts can be used in LoadTest, so you could use them later for load testing if you determined that they were a good representation of your user actions. It's very much up to you what you do with the scripts.

Find
Find Again
◀ ▶
Top of Chapter
Back

- At what point do I consider using LoadRunner over the Astra LoadTest tools? Can I integrate the scripts between products?

  Astra LoadTest scripts can be used in LoadRunner. LoadRunner should probably be used if you have a site that uses Java or other complex components. LoadRunner can run more Vusers and can run on the HTTP only level, whereas Astra LoadTest runs over a browser layer and thus is more dynamic. Astra LoadTest doesn't offer quite as much functionality as LoadRunner. You can perform database load testing with LoadRunner, whereas Astra LoadTest is just Web based. LoadRunner is generally for larger enterprise applications.

- **Availability testing: Is there a monitoring tool integrated with Astra tools?**

  Astra LoadTest has numerous graphs and monitors that allow you to watch how the server is doing under load.

- **Do JavaScript or session management require me to create scripts in more than one phase.** (Record the initial interaction and then program the script to make it playback correctly with session management)?

  No. We use an internal browser that handles all of these complexities. You just record the script, and play it back in the Controller.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Configuring the Virtual User Recorder

Find
Find Again
Top of Chapter
Back

Configuring the Virtual User Recorder

Customizing the Test Script Editor

The Virtual User Recorder includes a powerful and customizable script editor. This enables you to set the size of margins in test windows, change the way the elements of a test script appear, and create a list of typing errors that will be automatically corrected by Astra LoadTest.

This chapter describes:

- Setting Display Options
- Personalizing Editing Commands

Find
Find Again
◀ ▶
Top of Chapter
Back

About Customizing the Test Script Editor

The Virtual User Recorder’s script editor lets you set display options, and personalize script editing commands.

Setting Display Options

Display options let you configure the Virtual User Recorder’s test windows and how your test scripts will be displayed. For example, you can set the size of test window margins, and activate or deactivate word wrapping.

Display options also let you change the color and appearance of different script elements. These include comments, strings, Astra LoadTest reserved words, operators and numbers. For each script element, you can assign colors, text attributes (bold, italic, underline), font, and font size. For example, you could display all strings in the color red.

Finally, there are display options that let you control how the hard copy of your scripts will appear when printed.

Personalizing Script Editing Commands

The Virtual User Recorder includes a list of default keyboard commands that let you move the cursor, delete characters, cut, copy, and paste information to and from the clipboard. You can replace these commands with commands you prefer. For example, you could change the Set Bookmark [#] command from the default CTRL + K + [#] to CTRL + B + [#].

Find
Find Again
◀ ▶
Top of Chapter
Back

Setting Display Options

The Virtual User Recorder's display options let you control how test scripts appear in test windows, how different elements of test scripts are displayed, and how test scripts will appear when they are printed.

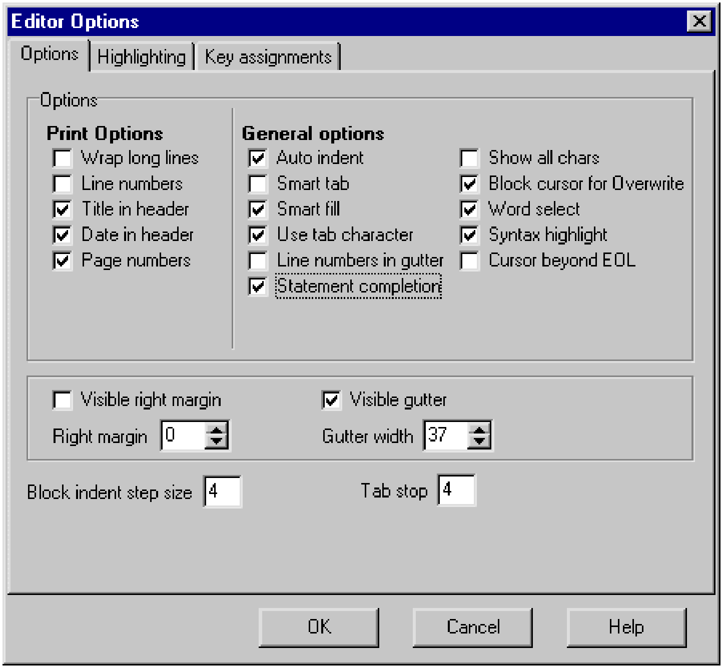
Customizing Test Scripts and Windows

You can customize the appearance of the Virtual User Recorder's test windows and how your scripts are displayed. For example, you can set the size of the test window margins, highlight script elements, and show or hide text symbols.

Find
Find Again
◀ ▶
Top of Chapter
Back

To customize the appearance of your script:

1 In the Expert View, choose **Tools > Editor Options**. The **Editor Options** dialog box opens.



Editor Options

Options

Print Options
- □ Wrap long lines
- □ Line numbers
- ☑ Title in header
- ☑ Date in header
- ☑ Page numbers

General options
- ☑ Auto indent
- □ Smart tab
- ☑ Smart fill
- ☑ Use tab character
- □ Line numbers in gutter
- ☑ Statement completion
- □ Show all chars
- ☑ Block cursor for Overwrite
- ☑ Word select
- ☑ Syntax highlight
- □ Cursor beyond EOL

□ Visible right margin ☑ Visible gutter

Right margin 0 Gutter width 37

Block indent step size 4 Tab stop 4

2 Click the **Options** tab.

3 Under the **General options** choose from the following options:

Options	Description
Auto indent	Causes lines following an indented line to automatically begin at the same point as the previous line. You can click the Home key on your keyboard to move the cursor back to the left margin.
Smart tab	A single press of the tab key will insert the appropriate number of tabs and spaces in order to align the cursor with the text in the line above.
Smart fill	Insert the appropriate number of tabs and spaces in order to apply the Auto indent option. When this option is not selected, only spaces are used to apply the Auto indent. (Both **Auto indent** and **Use tab character** must be selected to apply this option).
Use tab character	Inserts a tab character when the tab key on the keyboard is used. When this option is not enabled, the appropriate number of space characters will be inserted instead.
Line numbers in gutter	Displays a line number next to each line in the script. The line number is displayed in the test script window's gutter.

Find
Find Again
Top of Chapter
Back

Options	Description
Statement completion	Opens a list box displaying all available matches to the function prefix whenever the user presses the Ctrl and Space keys simultaneously, the Underscore key, or chooses **Edit > Complete Word**. Select an item from the list to replace the typed string. To close the list box, press the Esc key. Displays a tooltip with the function parameters once the complete function name appears in the editor. The function parameters are displayed also whenever the user presses the Ctrl, Shift, and Space keys simultaneously or choose **Edit > Parameter Info**.
Show all chars	Displays all text symbols, such as tabs and paragraph symbols.
Block cursor for Overwrite	Displays a block cursor instead of the standard cursor when you select overwrite mode.
Word select	Selects the nearest word when you double-click on the test window.
Syntax highlight	Highlights script elements such as comments, strings, or reserved words.
Cursor beyond EOL	Enables the Virtual User Recorder to display the cursor after the end of the current line.
Visible right margin	Displays a line that indicates the test window's right margin.

Find
Find Again
Top of Chapter
Back

Options	Description
Right margin	Sets the position, in characters, of the test window's right margin (0=left window edge).
Visible gutter	Displays a blank area (gutter) in the test window's left margin.
Gutter width	Sets the width, in pixels, of the gutter.
Block indent step size	Sets the number characters that the selected block of TSL statements will be moved (indented) when the INDENT SELECTED BLOCK softkey is used. For more information on editor softkeys, see **Personalizing Editing Commands** on page 369.
Tab stop	Sets the distance, in characters, between each tab stop.

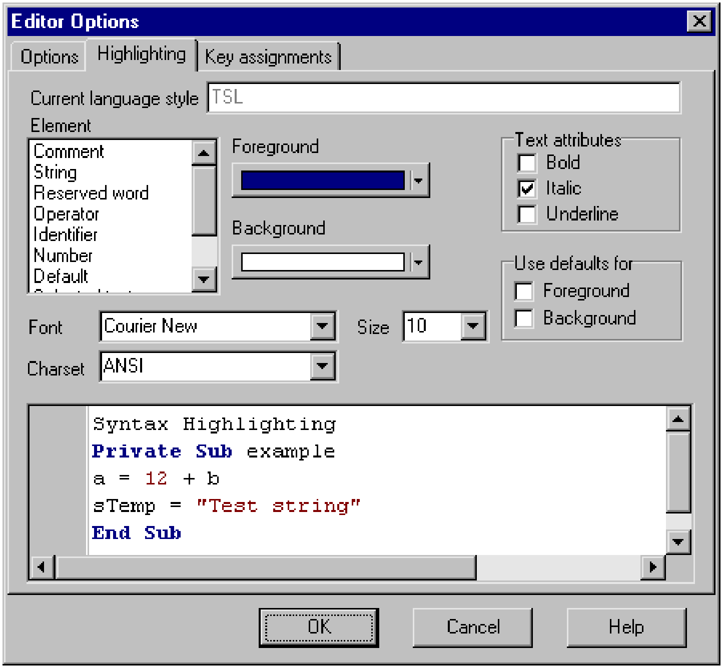
Find
Find Again
◀ ▶
Top of Chapter
Back

Highlighting Script Elements

The Virtual User Recorder tests contain many different elements, such as comments, strings, Astra LoadTest reserved words, operators and numbers. Each element of a Virtual User Recorder test is displayed in a different color and style. You can create your own personalized color scheme and style for each script element. For example, all comments in your scripts could be displayed as italicized, blue letters on a yellow background.

To edit script elements:

1 In the Expert View, choose **Tools > Editor Options**. The **Editor Options** dialog box opens to the **Highlighting** tab.



Current language style: TSL

Element:
- Comment
- String
- Reserved word
- Operator
- Identifier
- Number
- Default

Foreground

Background

Text attributes: ☐ Bold ☑ Italic ☐ Underline

Use defaults for: ☐ Foreground ☐ Background

Font: Courier New Size: 10

Charset: ANSI

```
Syntax Highlighting
Private Sub example
a = 12 + b
sTemp = "Test string"
End Sub
```

2 Select a script element from the **Elements** list.

Find
Find Again
Top of Chapter
Back

3 Choose from the following options:

Options	Description
Foreground	Sets the color applied to the text of the script element.
Background	Sets the color that appears behind the script element.
Text Attributes	Sets the text attributes applied to the script element. You can select bold, italic, or underline or a combination of these attributes.
Use defaults for	Applies the font and colors of the “default” style to the selected style.
Font	Sets the typeface of the script element.
Size	Set the size, in points, of the script element.
Charset	Sets the character subset of the selected font.

An example of each change you apply will be displayed in the pane at the bottom of the dialog box.

4 Click **OK** to apply the changes.

Find
Find Again
◀ ▶
Top of Chapter
Back

Customizing Print Options

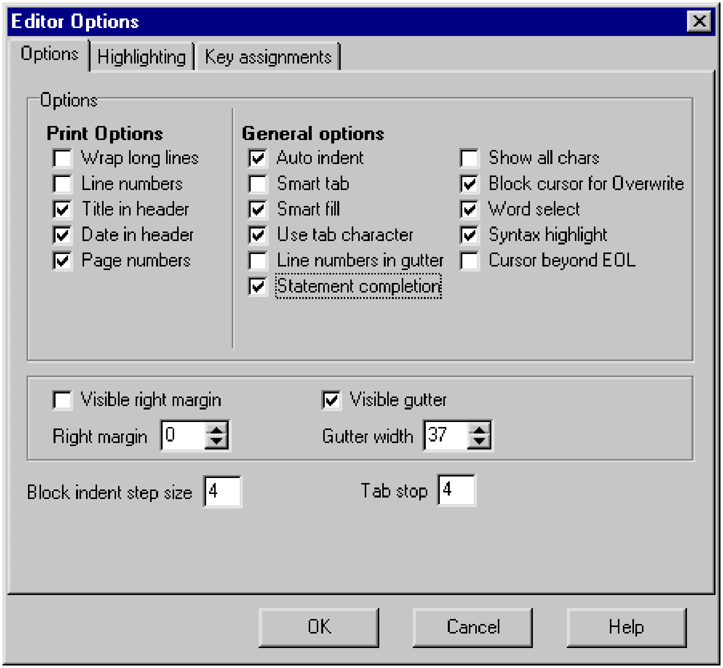
You can set how the hard copy of your script will appear when it is sent to the printer. For example, your printed script can include line numbers, the name of the file, and the date it was printed.

To customize your print options:

1 In the Expert View, choose **Tools > Editor Options**. The **Editor Options** dialog box opens.

Find
Find Again
Top of Chapter
Back

2 Click the **Options** tab.



Options

Print Options

- ☐ Wrap long lines
- ☐ Line numbers
- ☑ Title in header
- ☑ Date in header
- ☑ Page numbers

General options

- ☑ Auto indent
- ☐ Smart tab
- ☑ Smart fill
- ☑ Use tab character
- ☐ Line numbers in gutter
- ☑ Statement completion
- ☐ Show all chars
- ☑ Block cursor for Overwrite
- ☑ Word select
- ☑ Syntax highlight
- ☐ Cursor beyond EOL

☐ Visible right margin

☑ Visible gutter

Right margin 0

Gutter width 37

Block indent step size 4

Tab stop 4

Find
Find Again
Top of Chapter
Back

3 Choose from the following Print options:

Option	Description
Wrap long lines	Automatically wraps a line of text to the next line if it is wider than the current printer page settings.
Line numbers	Prints a line number next to each line in the script.
File name in header	Inserts the file name into the header of the printed script.
Date in header	Inserts today's date into the header of the printed script.
Page numbers	Numbers each page of the script.

4 Click **OK** to apply the changes.

Find
Find Again
◀ ▶
Top of Chapter
Back

Personalizing Editing Commands

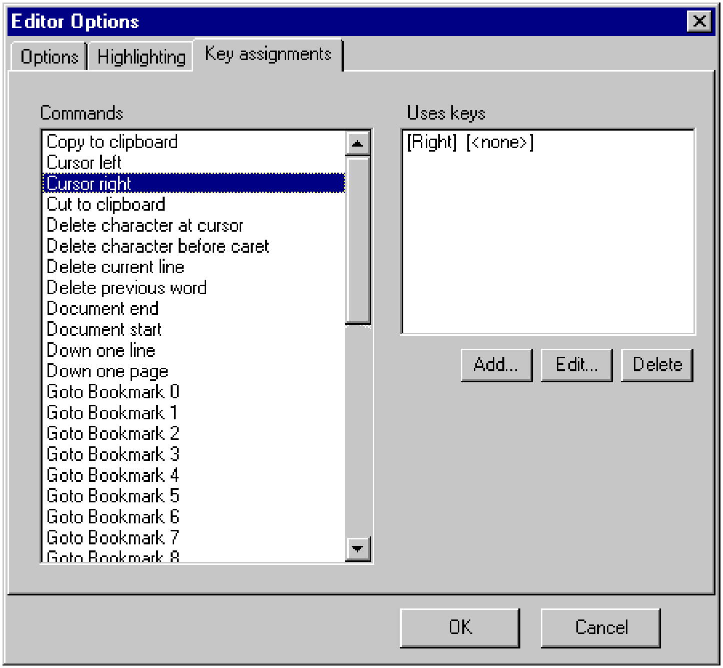
You can personalize the default keyboard commands you use for editing test scripts. The Virtual User Recorder includes keyboard commands that let you move the cursor, delete characters, cut, copy, and paste information to and from the clipboard. You can replace these commands with your own preferred commands. For example, you could change the Paste command from the default CTRL + V to CTRL + P.

To personalize editing commands:

1 In the Expert View, choose **Tools > Editor Options**. The **Editor Options** dialog box opens.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

2 Click the **Key Assignments** tab.



Editor Options

Commands

- Copy to clipboard
- Cursor left
- Cursor right
- Cut to clipboard
- Delete character at cursor
- Delete character before caret
- Delete current line
- Delete previous word
- Document end
- Document start
- Down one line
- Down one page
- Goto Bookmark 0
- Goto Bookmark 1
- Goto Bookmark 2
- Goto Bookmark 3
- Goto Bookmark 4
- Goto Bookmark 5
- Goto Bookmark 6
- Goto Bookmark 7
- Goto Bookmark 8

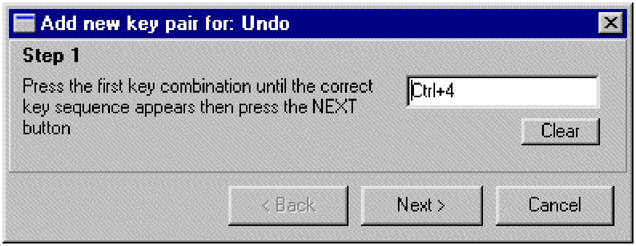
Uses keys

[Right] [<none>]

3 Select a command from the **Commands** list.

Find
Find Again
Top of Chapter
Back

4 Click **Add** to create an additional key assignment or click **Edit** to modify the existing assignment. The Add/Edit key pair for dialog box opens. Press the keys you want to use. For example, CTRL + 4.



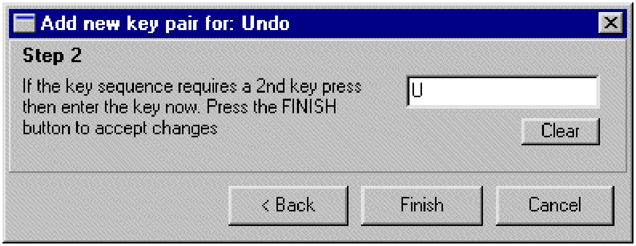
Add new key pair for: Undo

Step 1

Press the first key combination until the correct key sequence appears then press the NEXT button

Ctrl+4

5 Click **Next**. To add an additional key sequence, press the keys you want to use. For example, U.



Add new key pair for: Undo

Step 2

If the key sequence requires a 2nd key press then enter the key now. Press the FINISH button to accept changes

U

6 Click **Finish** to add the key sequence(s) to the **Use keys** list.

If you want to delete a key sequence from the list, highlight the keys in the **Uses Key** list and click **Delete**.

7 Click **OK** to apply the changes.

Find
Find Again
Top of Chapter
Back

Setting the Virtual User Recorder Testing Options

You can control how the Virtual User Recorder records and runs tests by setting testing options.

This chapter describes:

Find
Find Again
Top of Chapter
Back

About Setting the Virtual User Recorder Options

The Virtual User Recorder testing options affect how you record and run tests. For example, you can set the speed at which The Virtual Recorder runs a test, or set the timing-related settings used by The Virtual Recorder. The values you set remain in effect for all tests and for subsequent testing sessions.

You can also set testing options for that effect only the test currently open in The Virtual User Recorder. For more information, see Chapter 24, **Setting Testing Options for a Single Test**.

Find
Find Again
◀ ▶
Top of Chapter
Back

Setting the Virtual User Recorder Testing Options

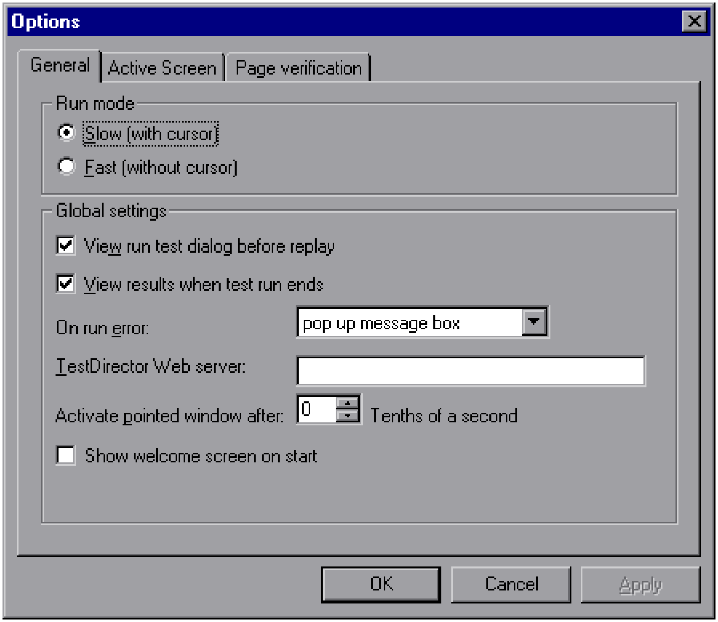
Before you record or run a test, you can use the Options dialog box to modify your testing options. The values you set remain in effect for all tests.

To set the Virtual User Recorder testing options:

1 Choose **Tools > Options**.

Find
Find Again
Top of Chapter
Back

The **Options** dialog box opens. It is divided by subject into three tabbed pages.



Run mode

● Slow (with cursor)

○ Fast (without cursor)

Global settings

☑ View run test dialog before replay

☑ View results when test run ends

On run error: pop up message box

TestDirector Web server:

Activate pointed window after: 0 Tenths of a second

□ Show welcome screen on start

2 To choose a page, click a tab.

3 Set an option, as described in The Virtual User Recorder Testing Options on page 376.

4 To apply your changes and keep the Options dialog box open, click **Apply**.

5 When you are done, click **OK** to apply your changes and close the dialog box.

Find
Find Again
Top of Chapter
Back

The Virtual User Recorder Testing Options

The Test Settings dialog box contains the following tabbed pages:

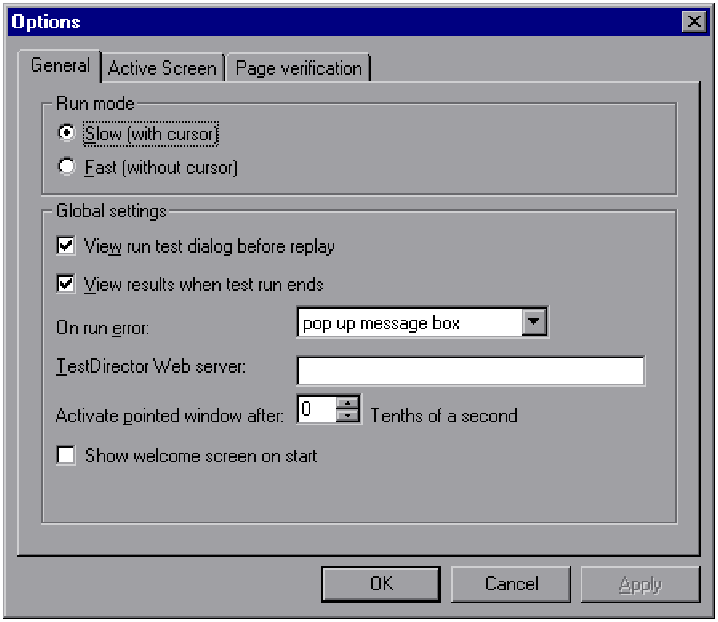
Tab Heading	Subject
General	options for run mode and global settings
Active Screen	options for displaying Web pages in the ActiveScreen tab in the Display pane
Page Verification	options for adding automatic checkpoints

This section lists the testing options you can set using the Options dialog box.

Find
Find Again
Top of Chapter
Back

General Testing Options

The General tab options affect how the Virtual User Recorder runs tests, displays test results, and specifies the Web server for reporting defects.



Run mode

● Slow (with cursor)

○ Fast (without cursor)

Global settings

☑ View run test dialog before replay

☑ View results when test run ends

On run error: pop up message box

TestDirector Web server:

Activate pointed window after: 0 Tenths of a second

□ Show welcome screen on start

Find
Find Again
Top of Chapter
Back

The General tab includes the following options:

Option	Description
Run mode - Slow (with cursor)	Instructs Astra LoadTest to run your test with the execution arrow in the left margin of the test, marking each step or statement as it is interpreted.
Run mode - Fast (without cursor)	Instructs Astra LoadTest to run your test without the execution arrow in the left margin of the test, marking each step or statement as it is interpreted.
View results when test run ends	Instructs Astra LoadTest to display the test results automatically following the test run.
On run error	Determines how Astra LoadTest responds to an error during a test run. Choose an option from the list: **pop up message box** displays an error message dialog box when an error occurs. **proceed to next iteration** jumps to the next iteration when an error occurs. **stop run** stops the test run when an error occurs.
TestDirector Web server	Designates the HTTP address to use to report and maintain defects in the Web Defect Manager. For example, if the TestDirector Web Defect Manager server is accessed using the *http://testdirector.mycompany.com/defects/bugs.htm* page, then the HTTP address should be specified as *http://testdirector.mycompany.com/defects/*.

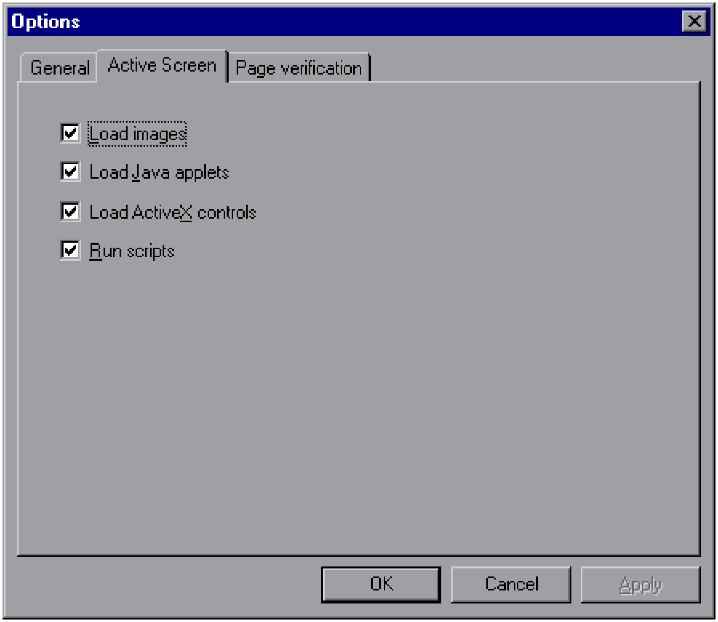
Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Option	Description
Activate pointed window after	Specifies the time (in tenths of a second) that Astra LoadTest waits before it sets focus on the Web browser.
Show welcome screen on start	Determines whether the Welcome screen is displayed when starting Astra LoadTest.

Find
Find Again
Top of Chapter
Back

Active Screen Testing Options

The Active Screen tab options affect how the Virtual User Recorder displays Web pages in the ActiveScreen view in the Display pane.



☑ Load images

☑ Load Java applets

☑ Load ActiveX controls

☑ Run scripts

Find
Find Again
Top of Chapter
Back

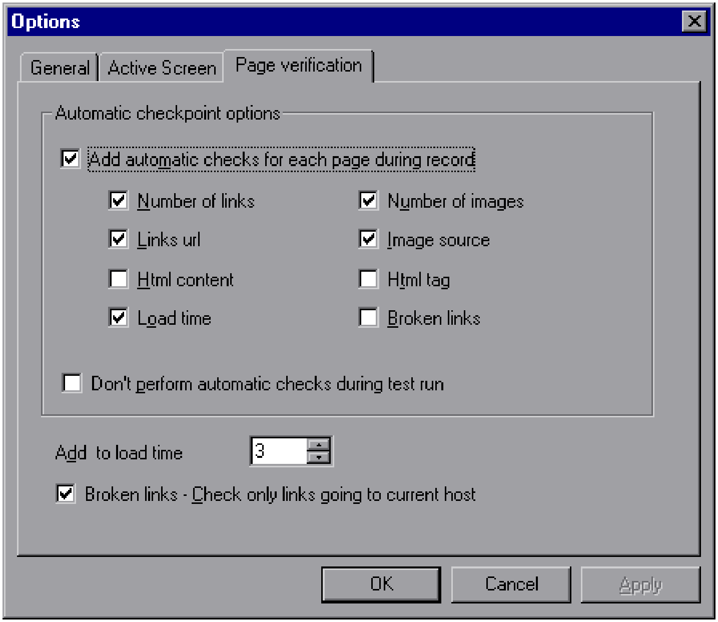
The Active Screen tab includes the following options:

Option	Description
Load images	Instructs Astra LoadTest to load images from your browser page to the ActiveScreen pane.
Load Java applets	Instructs Astra LoadTest to load Java applets from your browser page to the ActiveScreen pane.
Load ActiveX controls	Instructs Astra LoadTest to load ActiveX controls from your browser page to the ActiveScreen pane.
Run scripts	Instructs Astra LoadTest to run scripts while loading your browser page on the ActiveScreen pane.

Find
Find Again
Top of Chapter
Back

Page Verification Options

The Page Verification tab options affect how the Virtual User Recorder performs automatic checkpoints on a Web page.



Automatic checkpoint options

☑ Add automatic checks for each page during record

☑ Number of links

☑ Number of images

☑ Links url

☑ Image source

□ Html content

□ Html tag

☑ Load time

□ Broken links

□ Don't perform automatic checks during test run

Add to load time: 3

☑ Broken links - Check only links going to current host

Find
Find Again
Top of Chapter
Back

The Page Verification tab includes the following options:

Option	Description
Add automatic checks for each page during record	Instructs the Virtual User Recorder to add a page checkpoint for each page navigated to in your site.
Number of links	Checks that the number of links which appeared during the record session and the run session are identical.
Links URL	Checks that the URL links which appear during the record session and the run session are identical.
HTML content	Checks that the source code which appears during the record session and the run session are identical.
Load time	Checks that the amount of time it takes for the page to load during the record session and the run session are identical.
Number of images	Checks that the number of images which appear during the record session and the run session are identical.
Image source	Checks that the source files of the images which appear during the record session and the run session are identical.
HTML tag	Checks that the HTML tags which appear in the source code during the record session and the run session are identical.

Find
Find Again
Top of Chapter
Back

Option	Description
Broken links	Displays the number of broken links which appear during the run session.
Don’t perform automatic checks during test run	Instructs Astra LoadTest to ignore the automatically added page checkpoints when running your test.
Add to load time	Instructs Astra LoadTest to add a specified number of seconds to the load time of the page. This option is a safeguard which prevents the test from failing in the event that the amount of time it takes for a page to load during the run session is higher than the amount of time it took during the record session.
Broken links - Check only links going to current host	Instructs Astra LoadTest to check for broken links that are targeted to your current host.

Find
Find Again
◀ ▶
Top of Chapter
Back

Setting Testing Options for a Single Test

You can control how Astra LoadTest records and runs specific tests by setting testing options.

This chapter describes:

Find
Find Again
Top of Chapter
Back

About Setting Testing Options for a Single Test

You can set testing options that affect how you record and run a specific test. For example, you can instruct Astra LoadTest to run a parameterized action for only certain lines in the table in the Data pane. You can also teach Astra LoadTest to recognize a specific object in your test as a standard object. These testing options are saved when you save the test.

You can set testing options from within a test, for a part of the test, using a test script. For more information, see Chapter 25, **Setting Testing Options from a Test Script**.

You can also set testing options that affect all tests. For more information, see Chapter 23, **Setting the Virtual User Recorder Testing Options**.

Find
Find Again
Top of Chapter
Back

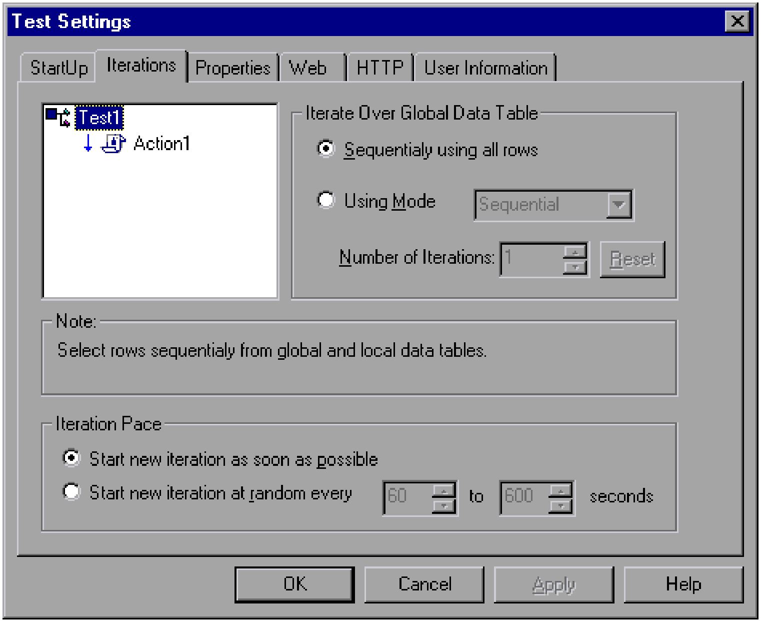
Setting Testing Options for a Single Test

Before you record or run a test, you can use the Test Settings dialog box to modify your testing options.

To set testing options for a single test:

1 Choose **Test > Settings**.

The **Test Settings** dialog box opens. It is divided by subject into six tabbed pages.



Test1
Action1

Iterate Over Global Data Table

● Sequentialy using all rows

○ Using Mode Sequential

Number of Iterations: 1 Reset

Note:

Select rows sequentialy from global and local data tables.

Iteration Pace

● Start new iteration as soon as possible

○ Start new iteration at random every 60 to 600 seconds

Find
Find Again
Top of Chapter
Back

2. To choose a page, click a tab.
3. Set an option, as described in **Testing Options for a Single Test** on page 389.
4. To apply your changes and keep the Test Settings dialog box open, click **Apply**.
5. When you are done, click **OK** to apply your changes and close the dialog box.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Testing Options for a Single Test

The Test Settings dialog box contains the following tabbed pages:

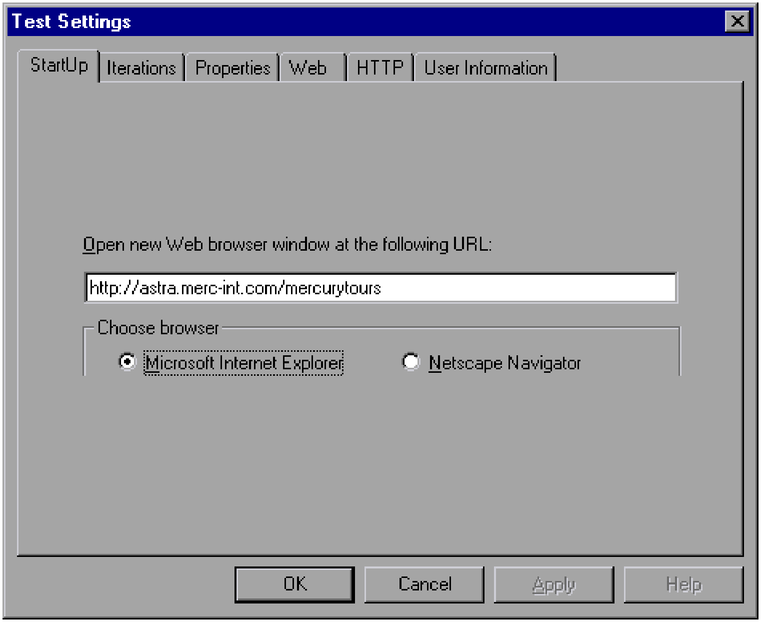
Tab Heading	Subject
StartUp	Options for setting a Web browser for recording tests
Iterations	Options for setting test run logic for tests and actions
Properties	Options for setting the properties of tests
Web	Options for recording tests
HTTP	Options for HTTP testing
User Information	Options to handle network and active screen passwords

This section lists the testing options you can set using the Test Settings dialog box.

Find
Find Again
◀ ▶
Top of Chapter
Back

StartUp Testing Options

The StartUp tab options set which browser to use while recording and whether to use an existing Web browser window or to open a new browser window to a specified location.



Test Settings

StartUp | Iterations | Properties | Web | HTTP | User Information

Open new Web browser window at the following URL:

http://astra.merc-int.com/mercurytours

Choose browser

● Microsoft Internet Explorer ○ Netscape Navigator

Find
Find Again
Top of Chapter
Back

The StartUp tab includes the following options:

Option	Description
Open new Web browser window at the following URL	Instructs Astra LoadTest to open a new browser session to record a test using the specified Web location address.
Choose browser	Instructs Astra LoadTest to use the specified browser type to record a browser session.

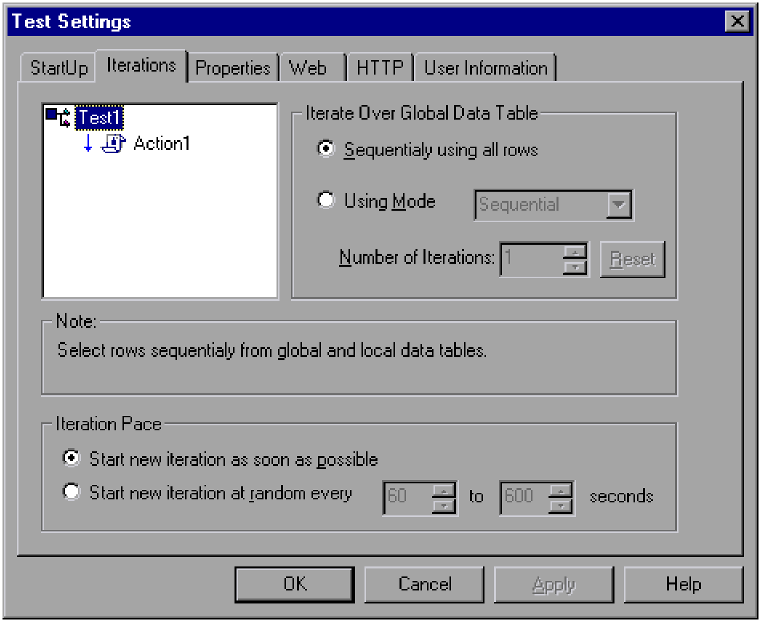
Note: You can also set the **Open new Web browser window at the following URL** options for a specific test in the **Start Recording** dialog box, which opens when you start recording a new test.

Iterations Testing Options

When you run a test, Astra LoadTest performs the steps you recorded on your Web site. When you run a test with global parameters, Astra LoadTest runs the test for each row in the table in the Data pane, using the parameters you specified. If your test includes local parameters, you can choose to run a range of data sets. For more information on global and local parameters, see Chapter 9, **Working with Data Tables**.

Find
Find Again
Top of Chapter
Back

You can use the Iterations tab to instruct Astra LoadTest to parameterize a test or an action for only certain lines in the local tab in the Data pane.



Iterate Over Global Data Table

● Sequentialy using all rows

○ Using Mode Sequential

Number of Iterations: 1 Reset

Note:

Select rows sequentialy from global and local data tables.

Iteration Pace

● Start new iteration as soon as possible

○ Start new iteration at random every 60 to 600 seconds

Find
Find Again
Top of Chapter
Back

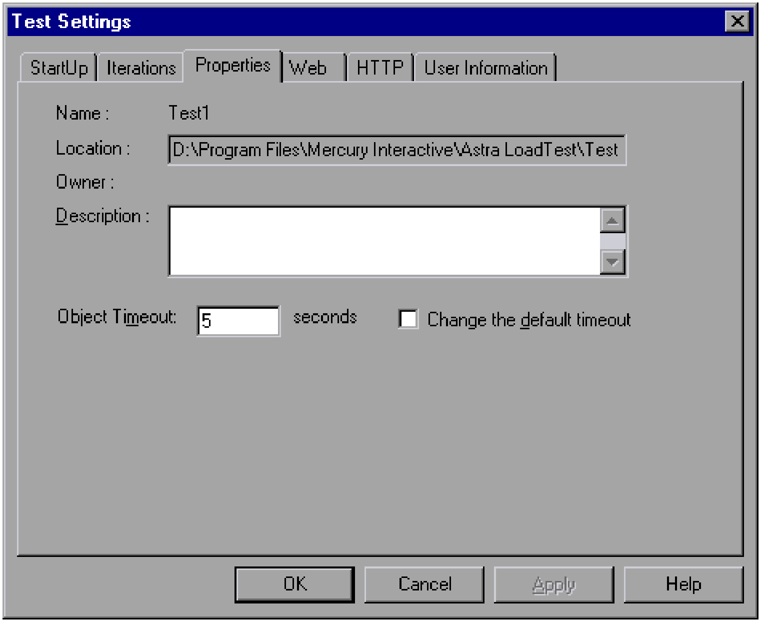
The Iterations tab includes the following options:

Option	Description
Iterate Over Global Data Table	Runs the test or action (depending on which is highlighted) using all the values in the local data table (for an action) or in the global data table (for a test). Use **Sequentially using all rows** to run as many iterations as there are rows in the data table. Alternatively, you can specify the **Number of Iterations**, regardless of the number of rows in the data table.
Using Mode	Specifies how Astra LoadTest should assign data to the Vusers for each iteration. Sequential assigns the data values to a Vuser in a sequential order. Random assigns the data values in a random order. Unique indicates that Astra LoadTest should make sure that each Vuser uses a unique data value for each iteration.
Iteration Pace	Specifies how long the test should wait before continuing with the next iteration.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Properties Testing Options

The Properties tab option defines general test information.



Test Settings

Name : Test1

Location : D:\Program Files\Mercury Interactive\Astra LoadTest\Test

Owner :

Description :

Object Timeout: 5 seconds

□ Change the default timeout

Find
Find Again
Top of Chapter
Back

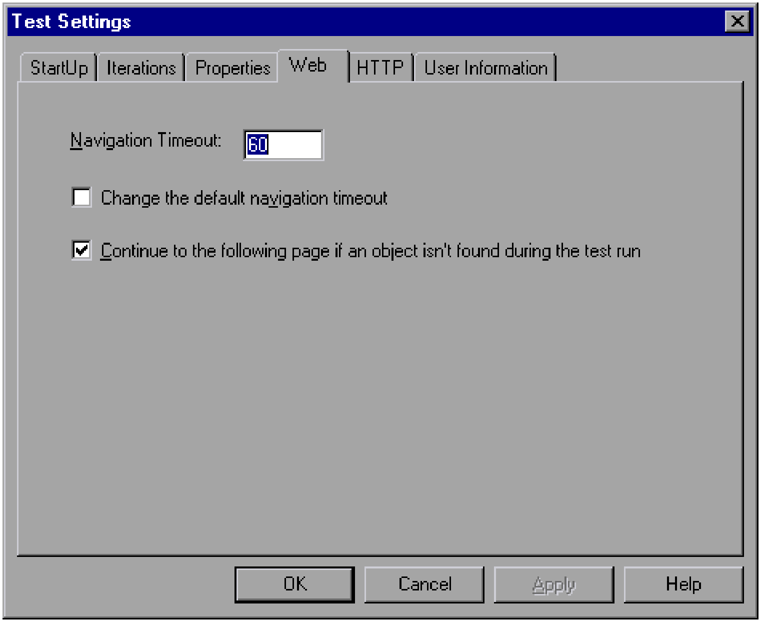
The Properties tab includes the following options:

Option	Description
Name	Indicates the name of the test.
Location	Indicates the path of the test.
Owner	Indicates the user name.
Description	Indicates the test description.
Object Timeout	Instructs Astra LoadTest not to exceed the specified time while loading an object.
Change the default timeout	Instructs Astra LoadTest not to use the default timeout.

Find
Find Again
◀ ▶
Top of Chapter
Back

Web Testing Options

The Web tab options affect the recording in the Virtual User Recorder.



Navigation Timeout: 60

□ Change the default navigation timeout

☑ Continue to the following page if an object isn't found during the test run

Find
Find Again
Top of Chapter
Back

The Web tab includes the following options:

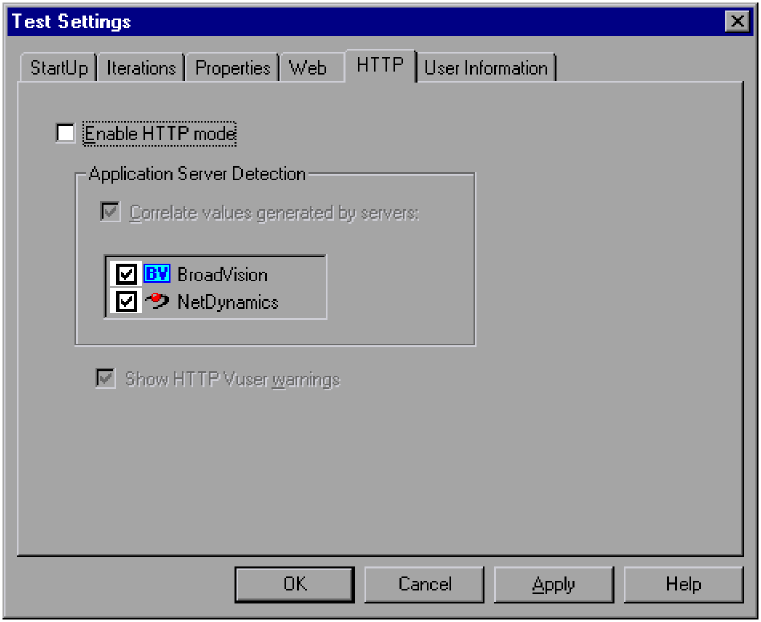
Option	Description
Navigation timeout	Indicates the interval (in seconds) Astra LoadTest waits for the Web page to load before running a test step.
Change the default navigation timeout	Changes the default **Navigation timeout** globally.
Continue to the following page if an object not found during the test run	Specifies that Astra LoadTest should continue running the test, even if an object is not found.

Note: You can also set the **Use existing Web browser window** and **Open new Web browser window at the following URL** options for a specific test in the **Start Recording** dialog box, which opens when you start recording a new test.

Find
Find Again
Top of Chapter
Back

HTTP Testing Options

The HTTP tab options affect how Astra LoadTest correlates dynamic information and issues HTTP warnings.



☐ Enable HTTP mode

Application Server Detection

☑ Correlate values generated by servers:

☑ BroadVision

☑ NetDynamics

☑ Show HTTP Vuser warnings

Find
Find Again
Top of Chapter
Back

The Windows tab includes the following options:

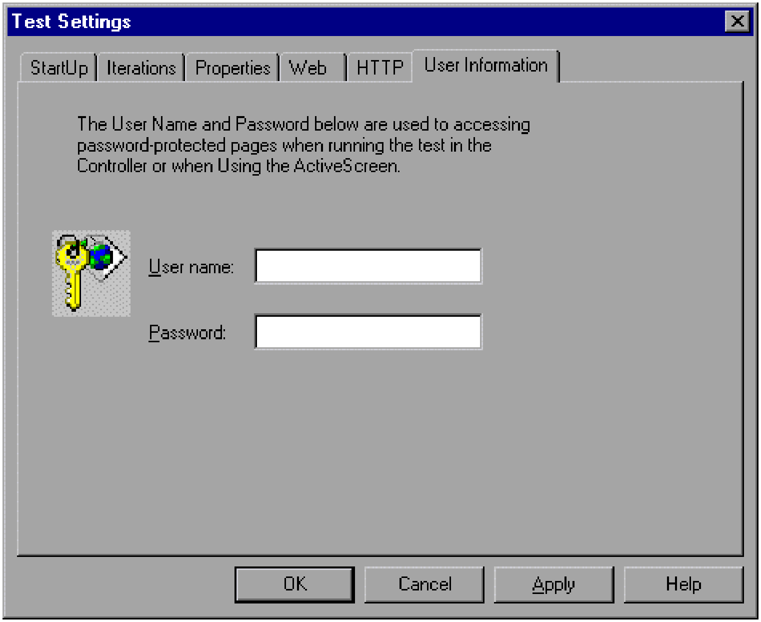
Option	Description
Enable HTTP mode	Instructs the Virtual User Recorder to record in HTTP mode. A test must be recorded in this mode for a HTTP replay to run.
Correlate values generated by servers:	Enables automatic detection of known Application Servers and correlates dynamic information to allow successful test runs.
Broadvision	Instructs Astra LoadTest to correlate values generated from a Broadvision server.
NetDynamics	Instructs Astra LoadTest to correlate values generated from a NetDynamics server.
Show HTTP Vuser warnings	Enables the suppression of HTTP Vuser warnings.

Note: For further information on HTTP mode, Chapter 17, **Testing in HTTP Mode**.

Find
Find Again
◀ ▶
Top of Chapter
Back

User Information Settings

The User Information tab allows you to enter networkor active screen passwords for processing tests in applications where passwords are required.



The User Name and Password below are used to accessing password-protected pages when running the test in the Controller or when Using the ActiveScreen.

User name:

Password:

Find
Find Again
Top of Chapter
Back

The User Information tab includes the following options:

Option	Description
User	The Network or ActiveScreen user name.
Password	The Network or ActiveScreen password.

Find
Find Again
Top of Chapter
Back

Setting Testing Options from a Test Script

You can control how Astra LoadTest records and runs tests by setting and retrieving testing options from within a test script.

This chapter describes:

- **Setting Testing Options**
- **Retrieving Testing Options**
- **Controlling the Test Run**
- **Adding and Removing Run-Time Settings**
- **Test Script Testing Options**

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

About Setting Testing Options from a Test Script

Astra LoadTest testing options affect how you record test scripts and run tests. For example, you can set the maximum time that Astra LoadTestallows for finding an object in a page.

You can set and retrieve the values of testing options from within a test script using the **Setting Object** function in the Expert View. For more information on Programming in the Expert View, see Chapter 20, **Testing in the Expert View**.

By retrieving and setting testing options in a test script using the **Setting Object**, you can control how Astra LoadTest executes a test.

You can also set many testing options using the Options dialog box (global testing options) and the Test Settings dialog box (test-specific settings). For more information on setting global testing options using the Options dialog box, see Chapter 23, **Setting the Virtual User Recorder Testing Options**. For more information on setting options for a single test, see Chapter 24, **Setting Testing Options for a Single Test**.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Setting Testing Options

You can use the **Setting Object** to set the value of a testing option from within the test script. To set the option, use the following syntax:

Setting (*testing_option* **)** = *new_value*

In this function, *testing_option* may specify any one of the following:

Testing Option	Possible Values	Setting Type
AutomaticLinkCheck	1 (ON) 0 (OFF)	Global
CheckBrokenLinks	1 (ON) 0 (OFF)	Global
CheckHtmlContent	1 (ON) 0 (OFF)	Global
CheckHtmlTag	1 (ON) 0 (OFF)	Global
CheckImagesSource	1 (ON) 0 (OFF)	Global
CheckLinksUrl	1 (ON) 0 (OFF)	Global
CheckLoadtime	1 (ON) 0 (OFF)	Global

Find
Find Again
Top of Chapter
Back

Testing Option	Possible Values	Setting Type
CheckNumberOfImages	1 (ON) 0 (OFF)	Global
CheckNumberOfLinks	1 (ON) 0 (OFF)	Global
DefaultLoadTime	0-9999	Global
DefaultTimeOut	1-1000	Test-specific
LocalLinks	1 (ON) 0 (OFF)	Global
WebTimeOut	1-1000	Test-specific

For example, if you execute the following statement:

```
Setting("AutomaticLinkCheck")=0
```

The Virtual User Recorder disables the **Add automatic checks for each page during record** testing option. The setting remains in effect during the testing session until it is changed again, either with another **Setting** statement or from the corresponding **Add automatic checks for each page during record** check box in the Page Verification tab of the Options dialog box (**Tools > Options**).

Using the **Setting** object with a global testing option changes a testing option globally, and this change is reflected in the Options dialog box. You can also use the **Setting** object to set testing options for a specific test, or even for part of a specific test. For more information see Controlling the Test Run on page 407.

Find
Find Again
◀ ▶
Top of Chapter
Back

Retrieving Testing Options

You can also use the **Setting** object to retrieve the current value of a testing option. To retrieve the value of a testing option, use the following syntax:

Setting (*testing_option* **)**

To store the value in a variable, use the syntax:

new_var = **Setting (** *testing_option* **)**

To display the value in a message box, use the syntax:

MsgBox (Setting (*testing_option* **))**

In this function, *testing_option* may specify any of the setting values listed in the table on **page 404**.

For example:

LinkCheckSet = Setting("AutomaticLinkCheck")

assigns the current value of the AutomaticLinkCheck setting to the user-defined variable LinkCheckSet.

Find
Find Again
Top of Chapter
Back

Controlling the Test Run

You can use the retrieve and set capabilities of the **Setting** object together to control a test run without changing global settings. For example, if you want to change the *DefaultTimeOut* testing option to 5 seconds for objects on one Web page only, insert the following statement after the Web page opens in your test script:

```
Rem Keep the original value of the DefaultTimeOut testing option
old_delay = Setting ("DefaultTimeOut")

Rem Set temporary value for the DefaultTimeOut testing option
Setting("DefaultTimeOut")= 5
```

To change back the *DefaultTimeOut* testing option to its original value at the end of the Web page, insert the following statement just before linking to the next page in the script:

```
Rem Change the DefaultTimeOut testing option back to its original value.
Setting("DefaultTimeOut")=old_delay
```

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

Adding and Removing Run-Time Settings

In addition to the global and test-specific settings, you can also add, modify, and remove your own run-time settings. These settings are applicable during the test run only.

To add a new run-time setting, use the syntax:

Setting.Add (*testing_option*, *value* **)**

Find
Find Again
Top of Chapter
Back

For example, you could create a setting that indicates the name of the current tester and writes the name in the report.

```
Setting.Add (“Tester Name”, “Mark Train”)
Reporter.ReportEvent 1, "Test Run By:", paramcount
```

To modify a run-time setting that has already been initialized, use the same syntax you use for setting any standard setting option:

Setting (*testing_option* **) =** *new_value*

For example:

```
Setting(“Tester Name”)=Alice Wonderlin
```

To remove a run-time setting, use the following syntax:

Setting.Remove (*testing_option* **)**

For example:

```
Setting.Remove ("Tester Name")
```

Test Script Testing Options

This section describes the Virtual User Recorder testing options that can be used with the **Setting** object from within a test script. The corresponding dialog box option is listed where applicable.

Find

Find Again

◀ ▶

Top of Chapter

⇦ Back

AutomaticLinkCheck

Sets or retrieves the setting for the Add automatic checks for each page during record option.

AutomaticLinkCheck is a global testing option.

Note that you may also set this option using the **Add automatic checks for each page during record** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckBrokenLinks

Sets or retrieves the setting for the check broken links option in an automatic page check.

CheckBrokenLinks is a global testing option.

Note that you may also set this option using the **Broken links** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckHtmlContent

Sets or retrieves the setting for the check HTML content option in an automatic page check.

CheckHtmlContent is a global testing option.

Note that you may also set this option using the **Html content** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckHtmlTag

Sets or retrieves the setting for the check HTML tag option in an automatic page check.

CheckHtmlTag is a global testing option.

Note that you may also set this option using the **Html tag** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckImagesSource

Sets or retrieves the setting for the check image source option in an automatic page check.

CheckImagesSource is a global testing option.

Find
Find Again
Top of Chapter
Back

Note that you may also set this option using the **Image source** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckLinksUrl

Sets or retrieves the setting for the check links URL option in an automatic page check.

CheckLinksUrl is a global testing option.

Note that you may also set this option using the **Links url** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckLoadtime

Sets or retrieves the setting for the check load time option in an automatic page check.

CheckLoadtime is a global testing option.

Note that you may also set this option using the **Load time** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

CheckNumberOfImages

Sets or retrieves the setting for the check number of images option in an automatic page check.

CheckNumberOfImages is a global testing option.

Note that you may also set this option using the **Number of images** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

CheckNumberOfLinks

Sets or retrieves the setting for the check number of links option in an automatic page check.

CheckNumberOfLinks is a global testing option.

Note that you may also set this option using the **Number of links** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

DefaultLoadTime

Sets or retrieves the setting for the add to load time option.

DefaultLoadTime is a global testing option.

Note that you may also set this option using the **Add to load time** option in the Page Verification tab of the Options dialog box as described in **Page Verification Options** on page 382.

Find
Find Again
Top of Chapter
Back

DefaultTimeOut

Sets or retrieves the delay for finding objects.

DefaultTimeOut is a test-specific option.

Note that you may also set this option using the **Object Timeout** option in the Properties tab of the Test Settings dialog box.

LocalLinks

Sets or retrieves the setting for the check broken links going to local host option.

LocalLinks is a global testing option.

Note that you may also set this option using the **Broken links - Check only links going to current host** option in the Page Verification tab of the Options dialog box as described in Page Verification Options on page 382.

WebTimeOut

Sets or retrieves the delay for navigating to a URL address.

WebTimeOut is a test-specific option.

Note that you may also set this option using the **Navigation Timeout** option in the Web tab of the Test Settings dialog box.

Find
Find Again
Top of Chapter
Back

Working with TestDirector

Managing the Testing Process

Web site testing typically involves creating and running thousands of tests. TestDirector, Mercury Interactive’s test management tool, can help you organize and control the testing process.

This chapter describes:

Find

Find Again

Top of Chapter

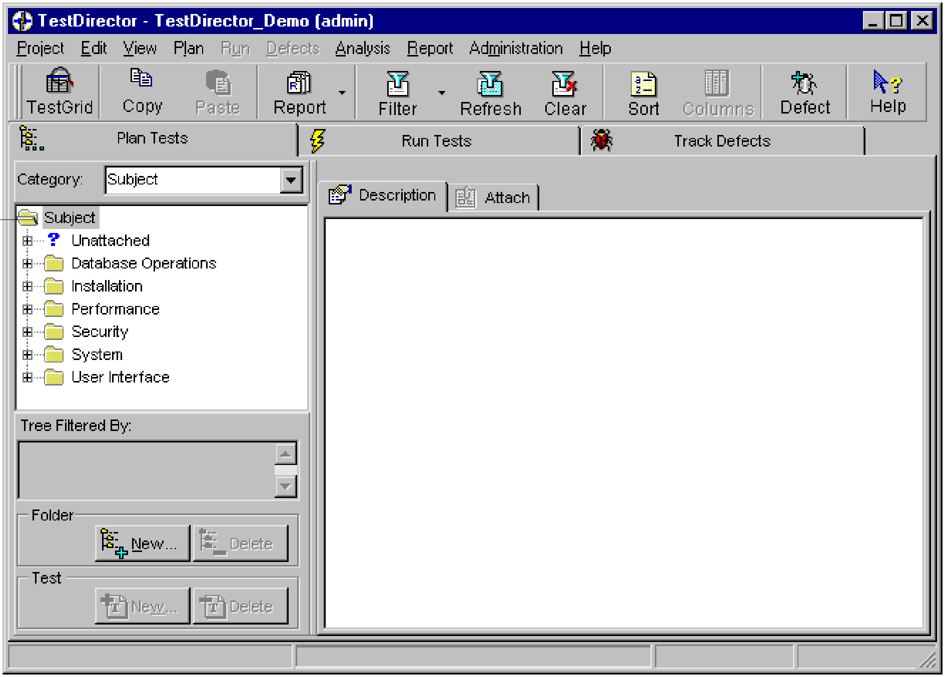
Back

About Managing the Testing Process

TestDirector is a powerful test management tool that enables you to manage and control all phases of software testing. It provides a comprehensive view of the testing process so you can make strategic decisions about the human and material resources needed to test a Web site and repair defects.

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

TestDirector divides testing into three modes of operation: Plan Tests, Run Tests, and Track Defects. In Plan Tests mode, you begin the testing process by dividing your site into test subjects and building a *test plan tree*. This is a graphical representation of your test plan, displaying your tests according to the hierarchical relationship of their functions.



Test plan tree

Find
Find Again
Top of Chapter
Back

After you build the test plan tree, you plan tests for each subject. You then use Astra LoadTest to record your tests and save them under the test plan tree.

In Run Tests mode, you define *test sets*. A test set is a group of tests designed to meet a specific testing goal. For example, to verify that your site is functional and stable, you create a sanity test set that checks the site's basic features. You could then create other test sets to test the advanced features.

To build a test set, you select tests from the TestDirector test repository. Once you build a test set, you can schedule test runs. You can run tests on your own computer (locally), or on multiple remote hosts. A host is any computer connected to your network. After TestDirector runs a test in Astra LoadTest, it displays the results and marks the test as passed, failed, or not completed.

In Track Defects mode, you report defects that were detected in the Web site you are testing. Information about defects is stored in a defect database. The defects are assigned to developers to be fixed, and then they are tracked until they are corrected.

In all stages of test management, you can create detailed reports and graphs to help you analyze testing data and review the progress of testing on your site.

For more information on working with TestDirector, refer to the *TestDirector User's Guide*.

Find
Find Again
◀ ▶
Top of Chapter
Back

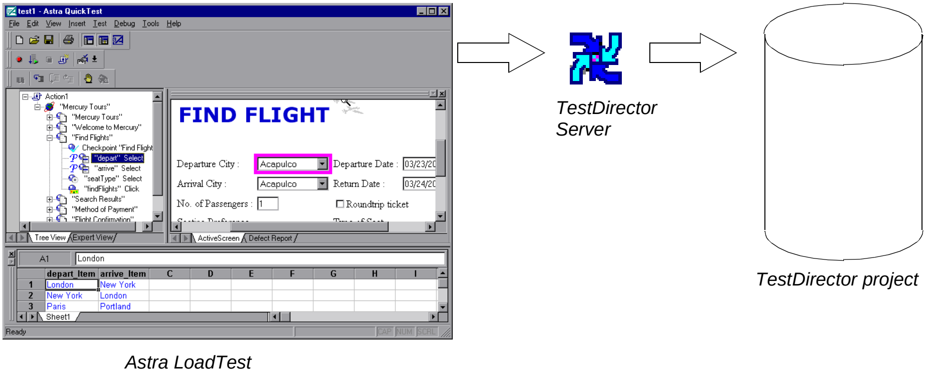
Using Astra LoadTest with TestDirector

TestDirector and Astra LoadTest work together to integrate all aspects of the testing process. In Astra LoadTest, you can create tests and save them in your TestDirector project. After a test is run, the results are viewed in TestDirector.

TestDirector stores test and defect information in a project. TestDirector project databases can be either file-based (Microsoft Access) or client/server (Oracle, Sybase, and Microsoft SQL). A file-based database resides on your local file system or in a shared network directory. Client/server databases always reside on a central database server. You create individual project databases within TestDirector. These project databases store information related to the current testing project, such as tests, test run results, and reported defects.

In order for Astra LoadTest to access the project, you must connect it to the TestDirector server. This is a program that handles the communication between Astra LoadTest and the TestDirector project. Note that the TestDirector server usually runs on your TestDirector machine but you can also install it on any computer connected to the network.

Find
Find Again
◀ ▶
Top of Chapter
Back



When Astra LoadTest is connected to TestDirector, you can save a test by associating it with a subject in the test plan tree, instead of assigning the test to a folder in the file system. This makes it easy to organize tests by subject for your Web site. When you open a test, you search for it according to its position in the test plan tree. After you run the test, results are sent directly to your TestDirector project.

Find
Find Again
Top of Chapter
Back

Note: The integration of Astra LoadTest with TestDirector (described in this chapter) is valid only for TestDirector 6.0.

Find
Find Again
Top of Chapter
Back

Connecting to and Disconnecting from a Project

If you are working with both Astra LoadTest and TestDirector, Astra LoadTest can communicate with your TestDirector project. You can connect or disconnect Astra LoadTest from a TestDirector project at any time during the testing process. However, do not disconnect Astra LoadTest from TestDirector while an Astra LoadTest test is opened from TestDirector.

The connection process has two stages. First, you connect Astra LoadTest to the TestDirector server. This server handles the connections between Astra LoadTest and the TestDirector project. Next, you choose the project you want Astra LoadTest to access. The project stores tests and test run information for the Web site you are testing. Note that TestDirector project databases are password protected, so you must provide a user name and a password.

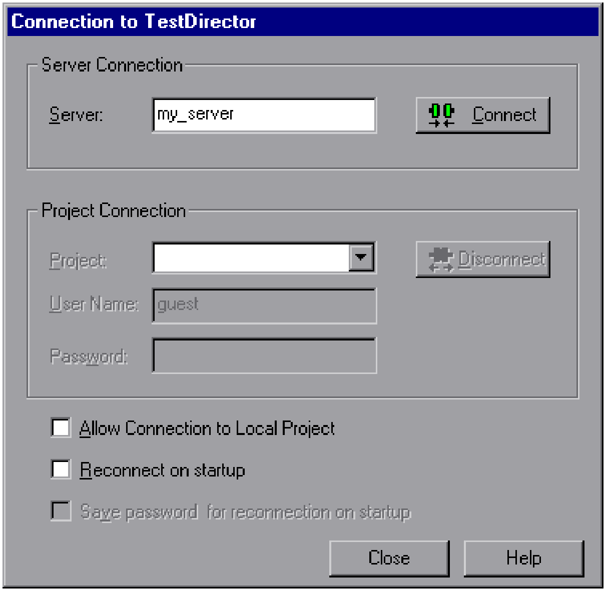
Connecting Astra LoadTest to a TestDirector Server and Project

You must connect Astra LoadTest to the TestDirector API server before you connect Astra LoadTest to a project. For more information, see **Using Astra LoadTest with TestDirector** on page 419.

Find
Find Again
Top of Chapter
Back

To connect Astra LoadTest to a TestDirector server and project:

1 Choose **Tools > TestDirector Connection**. The **Connection to TestDirector** dialog box opens.



Server: my_server

User Name: guest

□ Allow Connection to Local Project

□ Reconnect on startup

□ Save password for reconnection on startup

2 In the **Server Connection** section, in the **Server** box, enter the name of the host where the TestDirector server runs.

3 Click **Connect**.

Find
Find Again
Top of Chapter
Back

Once the connection to the server is established, the server's name is displayed in read-only format in the Server box.

4 In the **Project Connection** section, select a TestDirector project from the **Project** box.

5 Type a user name in the **User Name** box.

6 Type a password in the **Password** box.

7 Click **Connect** to connect Astra LoadTest to the selected project.

Once the connection to the selected project is established, the project's name is displayed in read-only format in the Project box.

To automatically reconnect to the TestDirector server and the selected project on startup, select the **Reconnect on Start Up** check box.

If the **Reconnect on Start Up** check box is selected, then the **Save Password for Reconnection on Start Up** check box is enabled. To save your password for reconnection on startup, select the **Save Password for Reconnection on Start Up** check box. If you do not save your password, you will be prompted to enter it when Astra LoadTest connects to TestDirector on startup.

8 Click **Close** to close the Connection to TestDirector dialog box.

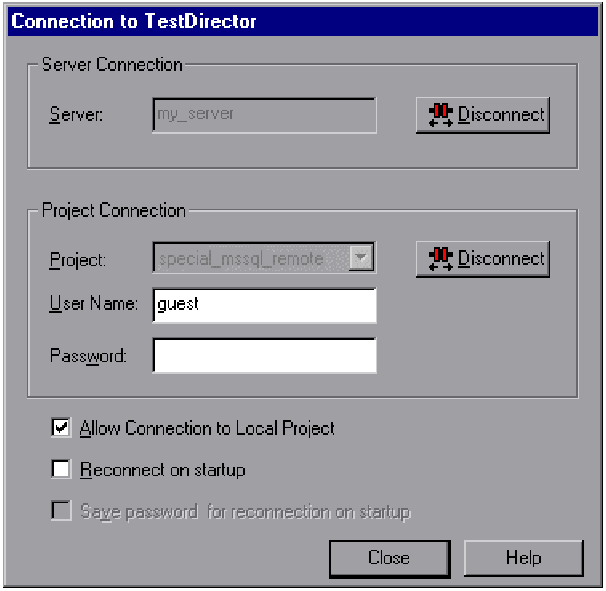
Find
Find Again
◀ ▶
Top of Chapter
Back

Disconnecting from a TestDirector Project

You can disconnect from a TestDirector project. This enables you to select a different project while using the same TestDirector server.

To disconnect Astra LoadTest from a TestDirector project:

1 Choose **Tools > TestDirector Connection**. The **Connection to TestDirector** dialog box opens.



Connection to TestDirector

Server Connection

Server: my_server

Project Connection

Project: special_mssql_remote

User Name: guest

Password:

☑ Allow Connection to Local Project

□ Reconnect on startup

□ Save password for reconnection on startup

Find
Find Again
Top of Chapter
Back

2 In the **Project Connection** section, click **Disconnect** to disconnect Astra LoadTest from the selected project.

3 Click **Close** to close the Connection to TestDirector dialog box.

Disconnecting from a TestDirector Server

You can disconnect from a TestDirector server. This enables you to select a different TestDirector server and a different project.

To disconnect Astra LoadTest from a TestDirector server:

1 Choose **Tools > TestDirector Connection**.

The **Connection to TestDirector** dialog box opens.

2 In the **Server Connection** section, click **Disconnect** to disconnect Astra LoadTest from the TestDirector server.

3 Click **Close** to close the Connection to TestDirector dialog box.

Note: If you disconnect Astra LoadTest from a TestDirector server without first disconnecting from a project, Astra LoadTest's connection to that database is automatically disconnected.

Find
Find Again
◀ ▶
Top of Chapter
Back

Saving Tests to a Project

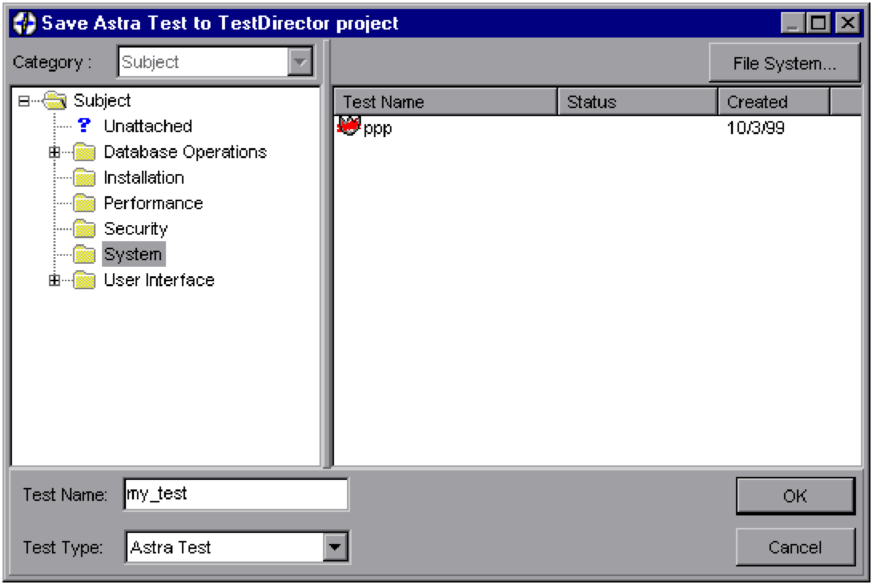
When Astra LoadTest is connected to a TestDirector project, you can create new tests in Astra LoadTest and save them directly to your project. To save a test, you give it a descriptive name and associate it with the relevant subject in the test plan tree. This helps you to keep track of the tests created for each subject and to quickly view the progress of test planning and creation.

To save a test to a TestDirector project:

1 In Astra LoadTest, click **Save** or choose **File > Save** to save the test.

The **Save Astra Test to TestDirector Project** dialog box opens and displays the test plan tree.

Find
Find Again
◀ ▶
Top of Chapter
Back



Test Name	Status	Created
ppp		10/3/99

Test Name: my_test

Test Type: Astra Test

Note that the Save Astra Test to TestDirector Project dialog box opens only when Astra LoadTest is connected to a TestDirector project.

Find
Find Again
Top of Chapter
Back

To save a test directly in the file system, click the **File System** button to open the Save Astra LoadTest Test dialog box. (From the Save Astra Test dialog box, you may return to the Save Astra Test to TestDirector Project dialog box by clicking the TestDirector button.)

Note: If you save a test directly in the file system, your test will not be saved in the TestDirector project.

---

2 Select the relevant subject in the test plan tree. To expand the tree and view a sublevel, double-click a closed folder. To collapse a sublevel, double-click an open folder.

3 In the **Test Name** text box, enter a name for the test. Use a descriptive name that will help you easily identify the test.

4 Click **OK** to save the test and to close the dialog box.

The next time you start TestDirector, the new test will appear in TestDirector's test plan tree. Refer to the *TestDirector User's Guide* for more information.

Find
Find Again
Top of Chapter
Back

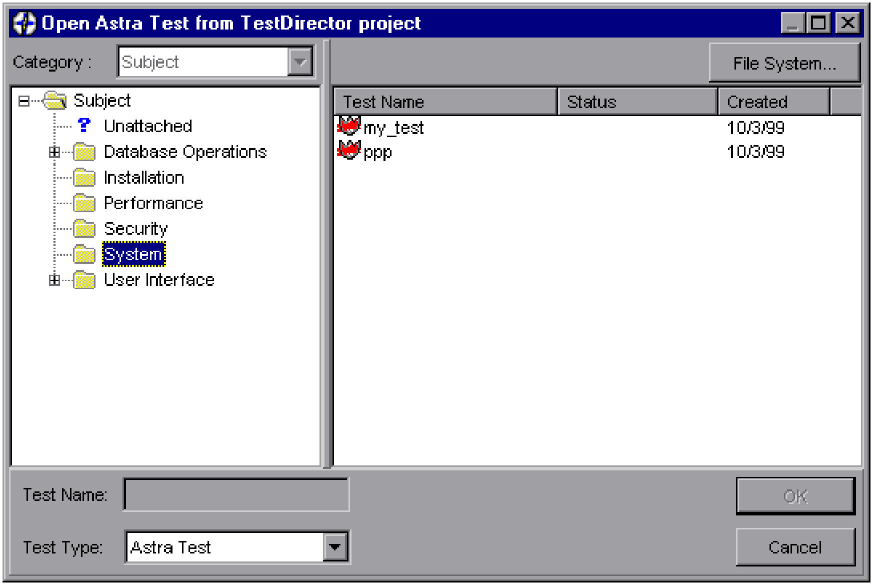
Opening Tests in a Project

If Astra LoadTest is connected to a TestDirector project, you can open automated tests that are a part of your database. You locate tests according to their position in the test plan tree, rather than by their actual location in the file system.

Find
Find Again
◀ ▶
Top of Chapter
Back

To open a test saved to a TestDirector project:

1 In Astra LoadTest, click **Open** or choose **File > Open** to open the test. The **Open Astra Test** dialog box opens. The **Open Astra Test from TestDirector Project** dialog box opens and displays the test plan tree.



Test Name	Status	Created
my_test		10/3/99
ppp		10/3/99

Note that the Open Astra Test from TestDirector Project dialog box opens only when Astra LoadTest is connected to a TestDirector project.

Find
Find Again
Top of Chapter
Back

To open a test directly from the file system, click the **File System** button to open the Open Astra LoadTest Test dialog box. (From the Open Astra Test dialog box, you may return to the Open Astra Test from TestDirector Project dialog box by clicking the TestDirector button.)

2 Click the relevant subject in the test plan tree. To expand the tree and view sublevels, double-click closed folders. To collapse the tree, double-click open folders.

Note that when you select a subject, the tests that belong to the subject appear in the Test Name list.

3 Select a test in the **Test Name** list. The test appears in the read-only Test Name box.

4 Click **OK** to open the test. The test opens in a window in Astra LoadTest.

Find
Find Again
◀ ▶
Top of Chapter
Back

Running Tests from TestDirector

You can run an Astra LoadTest test from a TestDirector project.

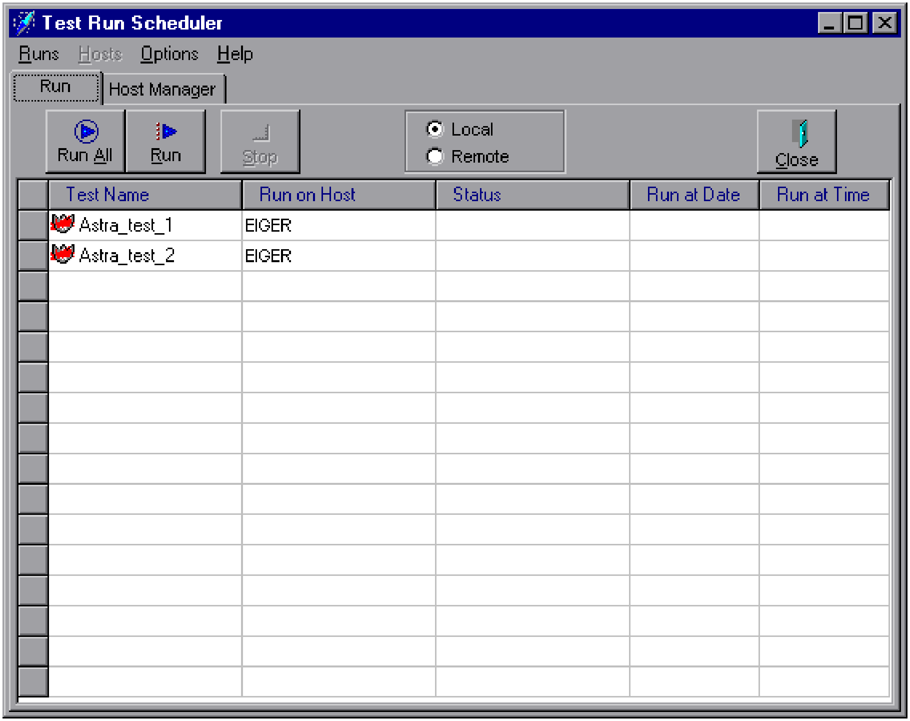
To run a test from a TestDirector project:

1 In TestDirector, click the **Run Tests** tab.

2 Select the tests you want to run from the Run Tests grid.

Find
Find Again
Top of Chapter
Back

3 Click the **Automated** button. The Test Run Scheduler opens with the selected tests displayed in the Test Run Scheduler grid.



Test Name	Run on Host	Status	Run at Date	Run at Time
Astra_test_1	EIGER			
Astra_test_2	EIGER			

Find
Find Again
Top of Chapter
Back

4 To run a single test, select a test from the grid and click the **Run** button. Alternatively, click the **Run All** button to run all the tests in the **Test Run Scheduler**. Click the **Stop** button if you need to terminate the test run before it is complete.

After the test run has finished running, you can view a summary of test results in TestDirector. The updated status for each test run appears in the **Run Tests** grid. Results for each test step appear in the **Last Run** grid. You must click the **Refresh** button before you can view the latest test results in the **Run Tests** grid.

Find
Find Again
Top of Chapter
Back

Working with TestDirector
Reporting Defects

You can report defects detected in your Web site using TestDirector’s Web Defect Manager.

This chapter describes:

About Reporting Defects

When you click the Defect Report tab in Astra LoadTest’s Display pane, TestDirector’s Web Defect Manager opens. It enables you to report defects detected in your site. You provide detailed information about the defect and then add it to a central repository (TestDirector project), so that the defect can be tracked until it is fixed.

Before you can launch the Web Defect Manager, you must ensure that a Web Defect Manager server is installed on your Web server, and that a TestDirector Web server is specified in the Test Settings dialog box. For more information about specifying the TestDirector Web server, see Chapter 27, **Reporting Defects**. For more information about installing the Web Defect Manager Server, refer to the *TestDirector Installation Guide*.

Find
Find Again
Top of Chapter
Back

Using the Web Defect Manager

The Web Defect Manager is Mercury Interactive’s system for reporting and tracking software defects and errors over the World Wide Web. The Web Defect Manager is a scalable, defect tracking system that helps you monitor defects closely from initial detection until resolution.

The Web Defect Manager is tightly integrated with TestDirector, Mercury Interactive's test management tool. Multiple users can share defect-tracking information stored in a central repository (TestDirector project). Several projects can be stored on a database server. This ensures that all software development, Quality Assurance, and Information Systems personnel can share defect-tracking information. For more information about TestDirector projects, refer to the *TestDirector User’s Guide*.

When you detect a defect in your site, you report it to a TestDirector project. For example, suppose you are testing a flight reservation site. You discover that errors occur when you try to order an airline ticket. You open the Web Defect Manager and report the defect. This includes a summary and detailed description of the defect, where it was discovered, and if you are able to reproduce it. You can also include screen captures, text documents, and other files relevant to understanding and repairing the problem. For information on using the Web Defect Manager, refer to the online *Web Defect Manager User’s Guide*.

Find
Find Again
◀ ▶
Top of Chapter
Back

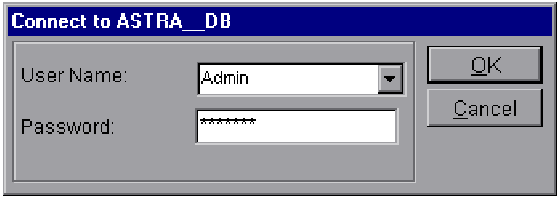
Reporting New Defects

Once the Web Defect Manager is set up, you can use it to report defects.

To report new defects:

1 In Astra LoadTest, click the **Defect Report** tab. The Web Defect Manager window opens and displays a list of TestDirector projects.

2 Select a project from the **TestDirector Projects** list. The **Connect to** dialog box opens.



Connect to ASTRA__DB

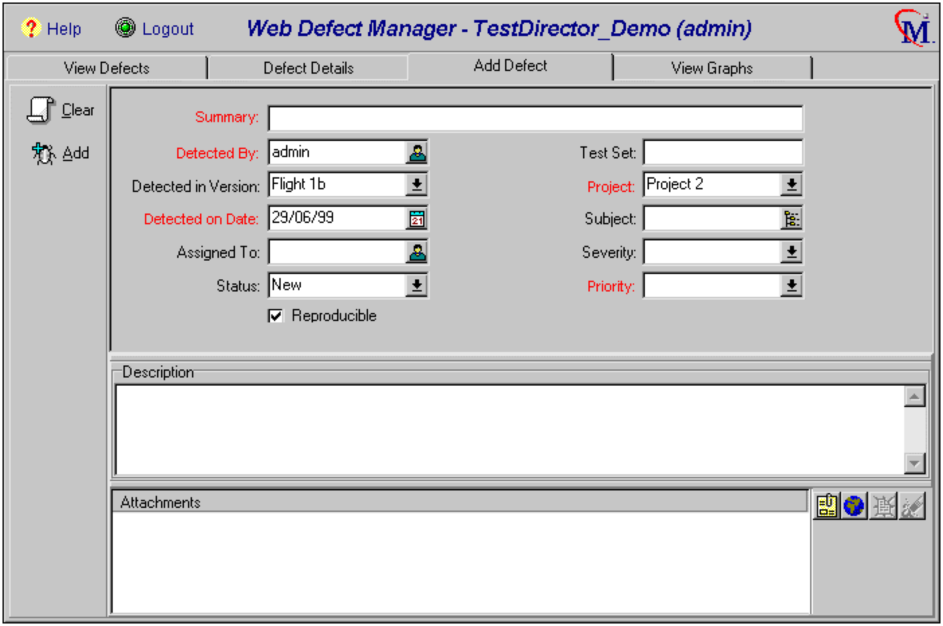
User Name: Admin

Password: ********

3 Type your user name and password and click **OK**. If you do not know your password, check with your TestDirector administrator. The **Web Defect Manager** opens and displays the **View Defects** tab.

Find
Find Again
Top of Chapter
Back

4 Click the **Add Defect** tab.



Summary:
Detected By: admin
Detected in Version: Flight 1b
Detected on Date: 29/06/99
Assigned To:
Status: New
☑ Reproducible
Test Set:
Project: Project 2
Subject:
Severity:
Priority:

5 In the **Summary** box, type in a brief summary of the defect.

6 In **Detected By**, click the name of the person who detected the defect. Note that the current date appears automatically.

Find
Find Again
Top of Chapter
Back

7 Enter information in the rest of the defect fields. Note that you must enter information in all the text boxes with red labels.

8 In the **Description** section, type in a detailed description of the defect. You may also type in other information, such as suggestions for working around the defect.

9 You can also choose from the following options:

- If the defect can be recreated under the same conditions by which it was detected, select the **Reproducible** check box.
- To associate a file with a defect record. Use attachments to reference relevant files such as detailed defect descriptions and captured screen images, click the **New Attachments** button.
- To associate an address of a document on the World Wide Web (URL) with a defect record, click the **New Web Link** button.

10 Click the **Add** button to save the defect record in the database. TestDirector assigns the new defect a unique record number.

11 Click the **Logout** button to exit the Web Defect Manager.

Find
Find Again
◀ ▶
Top of Chapter
Back

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
◀ ▶
Top of Chapter
⇦ Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
◀ ▶
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
◀ ▶
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
Top of Chapter
Back
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
Top of Chapter
Back
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
◀ ▶
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
◀ ▶
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find
Find Again
◀ ▶
Top of Chapter
Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Find

Find Again

Top of Chapter

Back

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Astra LoadTest User’s Guide, Version 4.0

Mercury Interactive may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents except as expressly provided in any written license agreement from Mercury Interactive.

WinRunner, XRunner, LoadRunner, TestDirector, TestSuite, and WebTest are registered trademarks of Mercury Interactive Corporation in the United States and/or other countries.Astra SiteManager, Astra SiteTest, Astra QuickTest, Astra LoadTest, Topaz, RapidTest, QuickTest, Visual Testing, Action Tracker, Link Doctor, Change Viewer, Dynamic Scan, Fast Scan, and Visual Web Display are trademarks of Mercury Interactive Corporation in the United States and/or other countries.

Online Guide

Find

Find Again

Back

This document also contains registered trademarks, trademarks and service marks that are owned by their respective companies or organizations. Mercury Interactive Corporation disclaims any responsibility for specifying which marks are owned by which companies or organizations.

If you have any comments or suggestions regarding this document, please send them via e-mail to documentation@mercury.co.il.

Mercury Interactive Corporation
1325 Borregas Avenue
Sunnyvale, CA 94089
Tel. (408) 822-5200 (800) TEST-911
Fax. (408) 822-5300

ALTUG4.0/01

Find
Find Again
◀ ▶
Top of Chapter
Back