WebTest® User's Guide Version 5.0

3210

Server

Online Guide



Find

Table of Contents

Chapter 1: Introduction	4
Using WebTest to Test Web Sites	5
How WebTest Identifies GUI Objects	6
Setting Up WebTest	8
Starting WebTest	10
Chapter 2: Creating Tests	11
About Creating Tests	
Planning Tests	
Recording Tests	
Understanding Your Test Script	
Enhancing WebTest Scripts with TSL	21
Chapter 3: Checking Web Pages	25
About Checking Web Pages	
Checking Contents of Cells or Frames	27
Checking Links	
Checking the Source, Type, or Url of an Image	
Checking All Images in a Cell or Frame	
Checking Table Contents	
Checking Table Size	
Checking Standard Properties	







Chapter 4: Running Tests	.48
About Running Tests	.49
WinRunner Test Run Modes	50
Running a Test to Check Your Web Site	52
Chapter 5: Analyzing Test Results	53
About Analyzing Test Results	.54
Viewing Results of a Test Run	55
Viewing Results of a Contents Check	57
Viewing Results of a Check on Links, Images, or Tables	.60
Viewing Results of a Check on Source, Type, or Url of an Image	65
Viewing Results of a Table Size Check	.67
Viewing Results of a Standard Properties Check	69

Index	71
	11



Click a page

Introduction

Welcome to WebTest, Mercury Interactive's automated software testing tool for the Netscape Navigator web browser and the Microsoft Internet Explorer web browser.

This chapter describes:

- Using WebTest to Test Web Sites
- How WebTest Identifies GUI Objects
- Setting Up WebTest
- Starting WebTest





Using WebTest to Test Web Sites

WebTest enables you to test and verify the functionality of your web site while using the Netscape Navigator web browser or the Microsoft Internet Explorer web browser. WebTest is an add-in to WinRunner, Mercury Interactive's automated GUI testing tool for Microsoft Windows applications.

With WebTest you can create a suite of tests, and then run the tests each time you change your web site. To create a test, use WinRunner to record the operations you perform on your web site. As you click on hypertext and hypergraphic links, WinRunner generates a test script in TSL, Mercury Interactive's C-like test script language.

You can further enhance the recorded test script by inserting GUI checkpoints. A GUI checkpoint captures the text content, links, images, tables, and standard properties of a web page or a frame. This information is stored in the test's expected results directory.

When you run a test, WinRunner performs the operations you recorded earlier. For each GUI checkpoint it compares the expected results with the actual results on the page. If WinRunner detects any discrepancies, it sends this information to a report which you can view after the test run is completed.

This guide explains how to use WebTest for the Netscape Navigator web browser and the Microsoft Internet Explorer web browser. For additional information on using WinRunner, refer to the **WinRunner User's Guide** and the **TSL Online Reference**.





How WebTest Identifies GUI Objects

WinRunner learns a set of default properties for each object you operate on while recording a test. These properties enable WinRunner to obtain a unique identification for every object that you test. This information is stored in the GUI map. WinRunner uses the GUI map to help it locate frames and objects during a test run.

WinRunner identifies each HTML page and frame that it encounters as a separate window. The *class* property indicates the class of the GUI object. The *html_name* property indicates the name assigned to the HTML page or frame. The *html_url* property indicates the Internet address of the page in which it is displayed.





Introduction

For example, a web page may have the following information in the GUI map:

```
{
class: window,
html_name: "Mercury Interactive Home Page",
html_url: "http://www.merc-int.com/index.html"
}
```

WinRunner assigns the *html_name* property to standard objects. For example, a search button may have the following information in the GUI map:

```
{
class: push_button,
html_name: Search
}
```

Non-standard objects are assigned the *value* property. For example, a hypertext link may have the following information in the GUI map:

```
{
class: object,
MSW_class: html_text_link,
value: Mercury Interactive's Products
}
```

You can view the contents of your GUI map files in the GUI Map Editor, by choosing Tools > GUI Map Editor. The GUI Map Editor displays the logical names and the physical descriptions of GUI objects. For more information, refer to the section "Understanding the GUI Map," in the *WinRunner User's Guide*.





WebTest User's Guide

Chapter 1, page 7

Setting Up WebTest

Before you install WebTest, you must first install WinRunner version 5.01. For more information, refer to the *WinRunner Installation Guide*.

After the WinRunner installation is complete, install the WebTest software. Webtest supports Netscape Navigator 4.0 and higher, and Microsoft Internet Explorer 4.01.

If you have an existing installation of WebTest, you will need to overwrite it during the installation process.

Note: If your web sites contain Java applets, you need to install Java add-in support for WinRunner.





Introduction

To install WebTest from disk:

- 1 Insert WebTest Disk 1 into a floppy drive.
- 2 Click Start on the Task Bar to open the Start menu. Choose Run.
- 3 In the **Open** box, type the drive name and **setup.exe**. For example, type a:\setup.exe.
- 4 Click **OK**. The WebTest setup program starts. Follow the instructions on your screen.
- 5 When the WebTest installation is completed, restart WinRunner.

To install WebTest from CD:

- 1 Insert WebTest CD into the CD drive.
- 2 Click Install.
- **3** The WebTest setup program starts. Follow the instructions on your screen.

Note: If WinRunner was installed previously, clear the GUI map before running any WebTest scripts. In WinRunner, choose **Tools > GUI Map Editor** to open the GUI Map Editor. Then, choose **Edit > Clear All** and close the GUI Map Editor.





Starting WebTest

Before you begin testing your web site, make sure that you have installed all the necessary files and made any necessary configuration changes. For more information, see **Setting Up WebTest** on page 8, and the *WinRunner Installation Guide*.



To start WebTest, click **Programs > WinRunner > WinRunner** in the Start menu. After several seconds, the WinRunner window opens on your desktop. You are now ready to launch a browser.



Creating Tests

You can quickly create a test script by recording the operations you perform in your web browser.

This chapter describes:

- Planning Tests
- Recording Tests
- Understanding Your Test Script
- Enhancing WebTest Scripts with TSL





About Creating Tests

You can create a test by recording, programming, or a combination of both methods. The easiest way to create a test is by recording. When you record a test, the operations that you perform on a web site are recorded in the test script as statements in Test Script Language (TSL). Usually you create a script by recording, and then you use programming to enhance the recorded script.

You can increase the power of your test script by adding GUI checkpoints. GUI checkpoints help you examine GUI objects in your web site and detect defects. When you run a test, a GUI checkpoint compares the current state of the GUI objects in your web site to the expected results captured when you created the test. If any differences are detected, WinRunner sends this information to the test report. For example, you can create checkpoints to compare an old version of a web page with a new version.





Planning Tests

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

- The functionality you want to test. Short tests that check specific functions of the application are better than long tests that perform several tasks.
- The types of GUI checkpoints that you want to use. A GUI checkpoint can check for differences in the HTML content, links, tables, and standard attributes of a page. For more information, see Chapter 3, Checking Web Pages.

For more information on planning tests, refer to the section "Creating Tests", in the *WinRunner User's Guide*.





Recording Tests

After planning your test, you are ready to start recording your test script using WinRunner's Context Sensitive recording mode. In this mode, WinRunner records the operations you perform on your web site by uniquely identifying Graphical User Interface (GUI) objects.

To create a test script:

- 1 Start WinRunner.
- 2 Start your web browser.

Note: You must start WinRunner before you start your browser. Otherwise, WinRunner will not record your test script properly.

	l

3 In WinRunner, choose File > New, or click the New button to create a new test.



- 4 Choose Create > Record-Context Sensitive, or click the Record button. WinRunner starts recording your operations.
- 5 Perform the sequence in your web browser that you want to record.

As you record, each operation you perform generates a TSL statement in your test script.

You can insert checkpoints in your test by choosing **Create > Check GUI > Object/Window**. For more information, see Chapter 3, **Checking Web Pages**. Find

Creating Tests



- 6 To stop recording, choose Create > Stop Recording or click the Stop button.
- 7 To save your test, choose File > Save As and assign the test a name.

Note: To prevent valuable GUI information from being overwritten during a new recording session, save the temporary GUI map file for your test script in a permanent GUI map file. For more information, refer to Chapter 4, "Creating the GUI Map", in the *WinRunner User's Guide*.





Understanding Your Test Script

As you record, each operation you perform generates a statement in Mercury Interactive's Test Script Language (TSL), in your test script. The following is a sample of a recorded WinRunner test script.

set_window("Mercury Interactive Home Page", 10);
web_link_click("Products");
set_window("Mercury Interactive Products", 10);
web_image_click("WebTest", 48, 6);
win_check_gui("Mercury Interactive Products-WebTest", "list1.ckl", "gui1", 1);





set_window Function

Each time you click a link in a page, WinRunner generates a **set_window** statement. The **set_window** statement directs input to the currently displayed page in the browser. It has the following syntax:

set_window (window, time);

window is the name of the current web page (the title of the page as it was written in HTML).

time is the maximum interval in seconds that WinRunner waits for the page to open when your run the test, plus the timeout value defined in the WinRunner Set Options dialog box. If it takes longer than the default timeout to load the page (the default is 10+10=20 seconds), WinRunner waits up to an additional 10 seconds for the page to finish loading in the browser.

For example, the statement:

set_window ("Mercury Interactive Home Page", 10);

indicates that "Mercury Interactive Home Page" is the name of the current web page. WinRunner waits a maximum of 10 seconds for the page to open, plus the timeout value defined in the WinRunner Set Options dialog box.





web_link_click Function

WinRunner records this function when you click a hypertext link. It has the following syntax:

web_link_click (link_name);

link_name is the logical name of the GUI object as it appears in the GUI map. For example, the statement:

web_link_click ("Products");

indicates that the "Products" hypertext link is clicked.





web_image_click Function

WinRunner records this function when you click a hypergraphic link or an image. It has the following syntax:

web_image_click (image_name, x, y);

image_name is the source name of the image. The position of the mouse click is expressed as x and y (pixel) coordinates. Coordinates are relative to the upper left corner of the image. For example, the statement:

web_image_click("WinRunner", 48, 6);

indicates that a bitmap called "WinRunner" is clicked. This image can be a hypergraphic link or a bitmap. The coordinates of the image are 48 and 6.





win_check_gui Function

The **win_check_gui** function captures and compares GUI data for a window. It has the following syntax:

win_check_gui (window, checklist, GUI_file, time);

window is logical name of the window.

checklist is the name of the checklist specifying the checks to perform.

GUI_file is the name of the file storing the expected GUI data.

time is the interval marking the maximum delay between the previous input and the capture of the current GUI data, in seconds. This interval is added to the timeout test option before the next statement is executed.

For example, the statement:

win_check_gui("Mercury Interactive Products-WinRunner", "list1.ckl", "gui1", 1);

indicates that a GUI checkpoint is added to verify the objects in the "Mercury Interactive Products-WinRunner" page.

For more information on TSL statements, refer to the TSL Online Reference.





Enhancing WebTest Scripts with TSL

You can enhance your recorded test scripts by adding TSL statements. In the Function Generator, the following functions are located in the *Web* category.

 The web_browser_invoke function invokes the browser and opens the specified site. It has the following syntax:

web_browser_invoke (browser, site);

 The web_cursor_to_image function directs the cursor to move to an image on a page. It has the following syntax:

web_cursor_to_image (image_name, x, y);

 The web_cursor_to_label function directs the cursor to move to a label on a page. It has the following syntax:

web_cursor_to_label (label_name, x, y);

 The web_cursor_to_link function directs the cursor to move to a hypertext link on a page. It has the following syntax:

web_cursor_to_link (link_name, x, y);

 The web_file_browse function directs the cursor to click on a browse button. It has the following syntax:

web_file_browse (web_object);





 The web_file_set function sets the text value in a file type object. It has the following syntax:

web_file_set (web_object, value);

 The web_frame_get_text function retrieves the text content of a page. It has the following syntax:

web_frame_get_text (frame_name, text);

 The web_get_timeout function returns the maximum time that WinRunner waits for response from the web. It has the following syntax:

```
web_get_timeout ( out_timeout );
```

 The web_image_click function is recorded when you click a hypergraphic link or an image. It has the following syntax:

web_image_click (image_name, x, y);

 The web_label_click function performs a click on a specified label. It has the following syntax:

web_label_click (label);

 The web_link_click function is recorded when you click a hypertext link. It has the following syntax:

web_link_click (link_name);



Find



• The **web_obj_get_child_item** function returns the description of the children in an object. It has the following syntax:

web_obj_get_child_item (web_object, table_row, table_column, object_type, index, object);

 The web_obj_get_child_item_count function returns the count of the children in an object. It has the following syntax:

 The web_obj_get_info function returns the value of an object property. It has the following syntax:

web_obj_get_info (web_object, property_name, property_value);

• The **web_set_timeout** function sets the maximum time WinRunner waits for a response from the web. It has the following syntax:

web_set_timeout (timeout);

 The web_sync function is recorded when you navigate between links within the same frame. It has the following syntax:

web_sync (timeout);

 The web_text_exists function returns text value if it is found in the frame. It has the following syntax:

web_text_exists (frame, text_to_find, parent_object, table_row, table_column
);





Creating Tests

When testing tables, you can also use the following **tbl_get** functions:

 The tbl_get_cell_data function retrieves the contents of the specified cell from a table. It has the following syntax:

tbl_get_cell_data (table, row, column, out_text);

• The **tbl_get_cols_count** function retrieves the number of columns in a table. It has the following syntax:

tbl_get_cols_count (table, out_cols_count);

• The **tbl_get_column_name** function retrieves the column header name of the specified column in a table. It has the following syntax:

tbl_get_column_name (table, col_index, out_col_names);

• The **tbl_get_rows_count** function retrieves the number of rows in the specified table. It has the following syntax:

tbl_get_rows_count (table, out_rows_count);

In the Function Generator, the **tbl_get** functions are located in the *Table Functions* category.

For more information on TSL functions, refer to the TSL Online Reference.





Checking Web Pages

You can check the contents of a web page by inserting GUI checkpoints into the test script.

This chapter describes:

- Checking Contents of Cells or Frames
- Checking Links
- Checking the Source, Type, or Url of an Image
- Checking All Images in a Cell or Frame
- Checking Table Contents
- Checking Table Size
- Checking Standard Properties





About Checking Web Pages

Use GUI checkpoints in your test script to help you examine your web site and detect defects. Using the Web Check GUI dialog box, you can create GUI checkpoints that check for differences between test runs in the text content, links, images, tables, and standard properties of a web page.

You can define GUI checkpoints according to default properties recommended by WinRunner, or you can define custom checks by selecting other properties.





Checking Contents of Cells or Frames

You can use a GUI checkpoint to check the text contents of a cell or a frame. To check the contents of a table, see see **Checking Table Contents**, on page 39.

To check the contents of a cell or a frame:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click the page. The Web Check GUI dialog box opens.

1	Web Check GUI			×
	Objects	Checks		
	🖃 🛅 Automated Testing Solutions - Mercury Interac	Check	Value	
	⊨	Content		
	Ė-∰ (Cell(6,1))	🔒 Images		
	🗄 🎬 h_seminar_ebiz_sm (Table)	💋 Links		
	🛲 (Cell(1,1))			
	(Cell(1,2))			
	•	•	Þ	
	Highlight Selected Object			
	OK Add All	Cancel	Help	





- **3** In the **Objects** column, the object you clicked on the web page is highlighted. You can check the contents for that object or click on any other object in the Objects column.
- 4 In the Checks column, click the Content check.
- 5 Click OK.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.





Checking Links

You can use a GUI checkpoint to check the HTML links in a frame or a cell.

To check the HTML links:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click a web page. The Web Check GUI dialog box opens.





- 3 In the **Objects** column, the object you clicked on the page is highlighted. You can check the links for that object or click on any object.
- 4 In the Checks column, click the Links check and click OK.

If you are checking links for a cell, the Cell Links Verification dialog box opens. If you are checking links for a frame, the HTML Links Verification dialog box opens.

<u>-</u>	TML Links Verification		_ 🗆 ×
	Text	Href 🔺	ОК
1	Company Profile	http://www.merc-int.com/company/bri	
2	Directions and Map	http://www.merc-int.com/company/tra	Cancel
3	Articles and Quotes	http://www.merc-int.com/company/pr/	
4	News Releases	http://www.merc-int.com/company/pr/	Insert
5	Investor Relations	http://www.merc-int.com/company/fina	Delete
6	Employment Opportunities	http://www.merc-int.com/company/jot	Delete
7	Company Offices	http://www.merc-int.com/company/off	
8	Worldwide Distributors	http://www.merc-int.com/company/dis	
9	LoadRunner	http://www.merc-int.com/products/loa	
10	WinRunner	http://www.merc-int.com/products/win	
11	XRunner	http://www.merc-int.com/products/xrui	
12	TestDirector	http://www.merc-int.com/products/tes	
13	TestSuite	http://www.merc-int.com/products/tes	





The HTML Links Verification dialog box and the Cell Links Verification dialog box, both list the links according to the order they appear in your source code. The dialog boxes include the following columns:

- Text indicates the name of the link.
- Href indicates the address of the link.

5 Select the type of check from the Verification Type list.

The following verification types are available:

- **Case Sensitive** (default), checks the text content of the links, ignoring any differences in font size, type, and color. Any difference in case or text content between the expected and actual data results in a mismatch.
- **Case Insensitive** checks the text content of the links, ignoring any differences in font size, type, color, and case. Only differences in text content between the expected and actual data result in a mismatch.
- Numeric Content evaluates the selected data according to numeric values. WinRunner recognizes, for example, that "2" and "2.00" are the same number. It also ignores the alignment of numerals within a cell.
- **Numeric Range** compares the selected data against a numeric range. This verification type is not relevant for the links verification check.
- 6 Specify the parts of the table that you want to check.
 - To include all links, click the upper left cell and click **Insert**.
 - To include one or more full columns, click the column header(s) and click **Insert**.
 - To include one or more full rows, click the row number(s) and click Insert.
 - To include any block of cells, highlight the block and click **Insert**.

A description of the checks inserted appears in the List of Checks field at the bottom of the dialog box.





- 7 To add more checks, repeat steps 4, 5, and 6.
- 8 Select a verification method for the entire list of checks, using the **Verification Method** check boxes.
 - Row Sensitive checks the rows in the list of checks according to a unique identifier, called a key. A shift in the position of a row does not result in a mismatch. For example, suppose you select Row Sensitive and define the key as the Href column. When running a test, WinRunner looks for the rows according to the Href column, ignoring any changes in the position of the row.
 - Column Names checks the selected cells according to their physical position. The column names are included in the verification. This option is not relevant for the links verification check.
- 9 Click OK to close the dialog box.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.





Checking the Source, Type, or Url of an Image

You can use a GUI checkpoint to check the source, image type, and the Url of a single image in your web page.

To check the source, type, or url of an image:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click an images on a page. The Web Check GUI dialog box opens. The image you clicked on the page is highlighted in the **Objects** column.

🙀 Web Check GUI		×
Objects	Checks	
[⊡-∰ (Cell(4,2))	Check	Value
🖻 🎹 web.gif (Table)	Source	http://www.merc-int.com/pics/cs.gif
🖻 🛲 (Cell(1,1))	👰 Туре	Image Link
庄 🛲 (Cell(2,1))	🍓 URL	http://www.merc-int.com/products
⊞ • 🗰 (Cell(3,1))		
🖕 🛲 (Cell(4,1))		
🔄 🤐 cs.gif (Image)		
🖶 🛲 (Cell(6,1))		
	•	
Highlight Selected Object	,	
OK Add All	Cancel	Help





- 3 In the **Checks** column, click one of the following checks:
 - **Source** indicates the location of the image.
 - Type indicates that the image is a plain image, image map, or an image link.
 - Url indicates the address of an image link.
- 4 Click **OK** to close the dialog box.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.





Checking All Images in a Cell or Frame

You can use a GUI checkpoint to check all images in a particular cell or frame.

To check all images in a cell or a frame:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click on a page. The Web Check GUI dialog box opens.

🚯 Web Check GUI	X
Objects	Checks
Ē-∰ (Cell(4,2))	Check Value
🗄 🚟 web.gif (Table)	Content
	Calmages
	💋 Links
🔒 🔒 cs.gif (Image)	
🛲 (Cell(5,1))	
🛲 (Cell(5,2))	
Highlight Selected Object	
OK Add All	Cancel Help

3 In the **Objects** column, the cell you clicked on the web page is highlighted. You can check that cell or click on another cell or a frame.

Find

4 In the **Checks** column, click the **Images** check and click **OK**. The HTML Images Verification dialog box opens.

🔒 HTML Images Verif	ication			_ 🗆 ×
Name	Туре	Width	Height	
l merc_logo.gif	Plain Image	100	116	
2 I_ball.gif	Plain Image	14	15	Cancel
3 I_ball.gif	Plain Image	14	15	
4 I_ball.gif	Plain Image	14	15	Insert
5 l_ball.gif	Plain Image	14	15	Delete
6 l_ball.gif	Plain Image	14	15	
7 l_ball.gif	Plain Image	14	15	
3 I_ball.gif	Plain Image	14	15	
3 na∨top.gif	Client Side	495	19	•
•				•
Verification Method	Key Sele	ction	Verification Type:	
	Name		Case Sensitive	-
🔽 <u>R</u> ow Sensitive	Туре		Todae Demaidve	
Column <u>N</u> ames	Width Height			
List Of Checks				


The HTML Images Verification dialog box displays information about each image included in a cell or a frame. It lists the images according to the order they appear in your source code.

The table includes the following columns:

- Name indicates the name of an image.
- Type indicates that the image is a plain image, image map, or an image link.
- Width and Height indicate the dimensions of an image, (only when the image size is defined in the source page).
- 5 Select the type of check from the Verification Type list.

The following verification types are available:

- **Case Sensitive** (default), checks the text content, ignoring any differences in font size, type, and color. Any difference in case or text content between the expected and actual data results in a mismatch.
- **Case Insensitive** checks the text content, ignoring any differences in font size, type, color, and case. Only differences in text content between the expected and actual data result in a mismatch.
- Numeric Content evaluates the selected data according to numeric values. WinRunner recognizes, for example, that "2" and "2.00" are the same number. It also ignores the alignment of numerals within a cell.
- **Numeric Range** compares the selected data against a numeric range. This verification type is not relevant for this verification check.





- 6 Specify the parts of the table that you want to check.
 - To include the entire table, click the upper left cell and click Insert.
 - To include one or more full columns, click the column header(s) and click Insert.
 - To include one or more full rows, click the row number(s) and click Insert.
 - To include any block of cells, highlight the block and click **Insert**.
 - A description of the checks inserted appears in the List of Checks field at the bottom of the dialog box.
- 7 To add more checks, repeat steps 4, 5, and 6.
- 8 Select a verification method for the entire list of checks, using the Verification Method check boxes.
 - Row Sensitive checks the rows in the list of checks according to a unique identifier, called a key. A shift in the position of a row does not result in a mismatch. For example, suppose you select Row Sensitive and define the key as the Src column. When running a test, WinRunner looks for the rows according to the Src column, ignoring any changes in the position of the row.
 - Column Names checks the selected cells according to their physical position. The column names are included in the verification. This option is not relevant for the images verification check.
- 9 Click OK to close the dialog box.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.



Find



Checking Table Contents

You can use a GUI checkpoint to check tables in a frame. You can check the contents of a table and the rows and columns count. Note that **y**ou can check only one table at a time. If a page contains several tables, you must create a new GUI checkpoint for each one.

To check the contents of a table:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click a page. The Web Check GUI dialog box opens.

🚜 Web Check GUI		×
Objects	Checks	
Ē-∰ (Cell(4,2))	Check	Value
⊨	ፍ Content	
	III Rows	5
	🛲 Columns	1
🕂 🤗 cs.gif (Image)		
🗰 (Cell(5,2))		
Highlight Selected Object		
OK Add All	Cancel	Help





WebTest User's Guide

- 3 In the **Objects** column, click a table.
- 4 In the Checks column, click **Contents**. The HTML Table Verification dialog box opens.

HTML Table Verification			_ 🗆 🗙
Col #1	Col #2	<u> </u>	ОК
 http://www.merc-int.com http://www.merc-int.com 	/pic:Australia Benel /pic:Canada Czech	ux Brazil — • Republic	Cancel
http://www.merc-int.com http://www.merc-int.com	/pic:Denmark Finlar /pic:Germany Gree	nd France - ce Hungary	Insert
http://www.merc-int.com http://www.merc-int.com	/picsIndia Israel Italy /picsKorea The Neth	/Japan - herlands -	Delete
	New Zealand N Singapore Sou	lorway J th Africa	
	Spain Sweden	Switzerland 🖵	
Verification Method	Key Selection	Verification Tv	me:
	Col #1	Case Sensitiv	ve 🔽
Column Nemos	Col #2		
1			



The data of the table is displayed in the dialog box. The dialog box, according to the columns found in the table.

5 Select the type of check from the Verification Type list.

The following verification types are available:

- **Case Sensitive** (default), checks the text content of the selection (ignoring any differences in font size, type, and color). Any difference in case or text content between the expected and actual data results in a mismatch.
- **Case Insensitive** checks the text content of the selection, ignoring any differences in font size, type, color, and case. Only differences in text content between the expected and actual data result in a mismatch.
- Numeric Content evaluates the selected data according to numeric values.
 WinRunner recognizes, for example, that "2" and "2.00" are the same number.
 It also ignores the alignment of numerals within a cell.
- Numeric Range compares the selected data against a numeric range. Both the minimum and maximum values are any real number that you specify. This comparison differs from text and numeric content verification in that the actual data is compared against the range that you defined and not against the expected results.
- 6 Specify the parts of the table that you want to check.
 - To include the entire table, click the upper left cell and click Insert.
 - To include one or more full columns, click the column header(s) and click Insert.
 - To include one or more full rows, click the row number(s) and click Insert.
 - To include any block of cells, highlight the block and click Insert.
 - A description of the checks inserted appears in the List of Checks field at the bottom of the dialog box.





WebTest User's Guide

Chapter 3, page 41

- 7 To add more checks, repeat steps 4, 5, and 6.
- 8 Select a verification method for the entire list of checks, using the Verification Method check boxes.

The following verification methods are available:

- Row Sensitive checks the rows in the selection according to a unique identifier, called a key. A shift in the position of a row does not result in a mismatch. For example, suppose you select Row Sensitive and define the key as the Product column. When running a test, WinRunner looks for the rows according to the Product column, ignoring any changes in the position of the row.
- **Column Names** checks the selected cells according to their physical position. The column names are included in the verification.

The following verification methods are available for a single-column table:

- By Position checks the selection according to the location of the selection.
- **By Content** checks the selection according to the content of the items, ignoring their location in the page.
- 9 Click OK to close the dialog box.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.





Checking Table Size

You can use a GUI checkpoint to check the number of rows and columns in a table. Note that **y**ou can check only one table at a time. If a page contains several tables, you must create a new GUI checkpoint for each one.

To check the table size:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click a page. The Web Check GUI dialog box opens.

bjects	Checks		
🋲 (Cell(4,2))	Check	Value	
🖮 🎹 web.gif (Table)	Content		
⊞ ## (Cell(1,1))	# Rows	5	
	🛲 Columns	1	
🔒 🔒 cs.qif (Image)			
🗰 (Cell(5,1))			
🗰 (Cell(5,2))			
🛲 (Cell(6,1))			
ງ 🛲 ແດລແຜ່ຂຶ້ນ			





- 3 In the **Objects** column, click a table.
- 4 In the Checks column, click Rows or Columns.
- 5 Click OK to close the dialog box.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.



Checking Standard Properties

You can use a GUI checkpoint to check standard properties of a page.

To check standard properties:

1 Choose Create > Check GUI > Object/Window.

The WinRunner window is minimized to an icon, the mouse pointer turns into a pointing hand, and a help window opens.

2 Double-click the page. The Web Check GUI dialog box opens.

🚯 Web Check GUI		×
Objects	Checks	
Automated Testing Solutions - Mercury Interactive	Check	Value
🚟 merc_logo.gif (Table)	🔁 Content	
	🔒 Images	
🕂 🔒 merc_logo.gif (Image)	💋 Links	
🖻 🚟 SITE MENU (Table)	🚟 Standard	
🛲 (Cell(1,1))		
🛲 (Cell(2,1))		
i ∰ ## (Cell(3,1))		
i ∰ ## (Cell(4,1))		
i (Cell(5,1))		
	1	
E Lieblich Cole start Obie st	<u> </u>	
	0	Usia I
OK Add All	Cancel	Неір





- 3 In the **Objects** column, click the frame.
- 4 In the Checks column, click Standard.
- 5 Click OK to close the dialog box. The Check GUI dialog box opens.

🖆 Check GUI - list1.ckl			×	
Add All Select All Clear All				
Objects	Properties		 /	Fied
🚽 🖂 🖾 Powerful Test Automation for	Name	Expected Value	上街	
	Count_objects Count_objects Catabled Catabled Catable			?
Highlight Selected Object	<u>O</u> K <u>C</u> a	ncel <u>H</u> elp		

The Check GUI dialog box displays the following standard properties:

- Count_objects counts the number of GUI objects in the window.
- Enabled checks whether the window can be selected.
- Focused checks whether keyboard input will be directed to this window.
- Label checks the window's label.
- **Minimizable** and **Maximizable** check whether the window can be minimized or maximized.
- Minimized and Maximized check whether the window is minimized or maximized.
- Resizable checks whether the window can be resized.
- System_menu checks whether the window has a system menu.
- Width and Height check the window's width and height, in pixels.
- X and Y check the x and y coordinates of the top left corner of the window.
- 6 Select the properties you want WinRunner to check.
- 7 Click OK to close the dialog box.

WinRunner captures the GUI information and stores it in the test's expected results directory. The WinRunner window is restored and a **win_check_gui** statement is inserted into your test script. For more information on **win_check_gui**, refer to the *TSL Online Reference*.





Running Tests

Once you have created a test script, you run the test to check the behavior of your web site.

This chapter describes:

- WinRunner Test Run Modes
- Running a Test to Check Your Web Site



About Running Tests

When you run a test, WinRunner interprets your test script, line by line, and performs the operations on your web site.

WinRunner provides three modes in which to run tests—Verify, Debug, and Update. You use each mode during a different phase of the testing process.

Use WinRunner's Run commands to run your tests. You can run an entire test or a portion of a test. For more information, refer to the *WinRunner User's Guide*.

Note: If you created permanent GUI map files for you tests, you must load the appropriate GUI map files before you run your tests. For more information, refer to Chapter 4, **Creating the GUI Map** in the *WinRunner User's Guide*.





WinRunner Test Run Modes

WinRunner provides three modes in which to run tests—Verify, Debug, and Update. You use each mode during a different phase of the testing process. You select a run mode from the list of modes on the WinRunner toolbar. The Verify mode is the default run mode.

Verify mode (default): Use the Verify mode to check your web site. WinRunner compares the *current* response of your web page to its *expected* response. Any discrepancies between the current and expected responses are captured and saved as *verification results*.

You can save as many sets of verification results as you need. To do so, save the results in a new directory each time you run the test. You specify the directory name for the results using the Run Test dialog box.

Debug mode: Use the Debug mode to help you identify bugs in a test script. Running a test in the Debug mode is the same as running a test in the Verify mode, except that debug results are always saved in the *debug* directory. Because only one set of debug results is stored, the Set Results Directory dialog box does not appear when you run a test in Debug mode.

Once you run a test in Debug mode, this mode remains the default run mode for the current WinRunner session, until you choose another mode.





Running Tests

Update mode: Use the Update mode to update the *expected results* of a test. For example, you might choose to update the expected results for a GUI checkpoint that checks a push button, if the default status of the push button changes from enabled to disabled. By default, WinRunner saves expected results in the *exp* directory, overwriting any existing expected results.

For more information, refer to the WinRunner User's Guide.



Running Tests

Running a Test to Check Your Web Site

When you run a test to check the behavior of your web site, WinRunner compares the current results with the expected results. You specify the directory in which the verification results for the test are saved.

To run a test to check your web site:

- 1 Open the test if it is not already open.
- 2 Make sure that **Verify** is selected from the dropdown list of run modes on the toolbar.



- 3 Choose Run > Run from Top, or click the Run from Top button.
- 4 In the Run Test dialog box, assign a name to the directory that will store the results, or accept the default name "res1".

Run Test	×
	<u>0</u> K
Test <u>R</u> un Name: resi 🚬	<u>C</u> ancel
□ Use Debug mode (don't display this dialog box)	<u>H</u> elp
☑ Display test results at end of run	

To instruct WinRunner to display the test results automatically following the test run (the default), select the **Display test results at end of run** check box.

Click **OK**. The Run Test dialog box closes and WinRunner runs the test.





WebTest User's Guide

Chapter 4, page 52

Analyzing Test Results

After you execute a test, you can view a report of all the major events that occurred during the test run in order to determine its success or failure.

This chapter describes:

- Viewing Results of a Test Run
- Viewing Results of a Contents Check
- Viewing Results of a Check on Links, Images, or Tables
- Viewing Results of a Check on the Source, Type, or Url of an Image
- Viewing Results of a Table Size Check
- Viewing Results of a Standard Properties Check





About Analyzing Test Results

When a test run is completed, you can view detailed test results in the WinRunner Test Results window. The window contains a description of the major events that occurred during the test run, such as errors and GUI checkpoints. You can view expected, debug, and verification results in the Test Results window. By default, the Test Results window displays the results of the most recently executed test run. For more information, refer to the *WinRunner User's Guide*.



Viewing Results of a Test Run

When a test run is completed, test results are displayed in the WinRunner Test Results window.

To view test results:



1 To open the WinRunner Test Results window, choose Tools > Test Results or click the Test Results button. Note, if the Display test results at end of run check box was selected in the Run Test dialog box before you ran the test, the Test Results window opens automatically.

1	Indicates whether the test passed or failed, and lists all checkpoints		WinF Eile ≽ @	Runner Options	Test Ro 	esults Wind	- [D:\Program Files dow	Mercury Interactive\WinRur	nner\t	mp\sample]		1× 1×
	performed during the test run.	sample					Test Result:	fail bitmap checkpoint0 GLII checkpoints: 3				
2	Failed GUI checkpoint.						General Informatio	n				
3	Successful GUI checkpoint.											
4	Checkpoint that					Line	Event	Details		Result	Time	
	verified the			1	start run	sample	run		00:00:00			
	content of a web	content of a web			1	start GUI checkpc	gui1	—		00:00:00		
	page.		Г		-	1	end GUI checkpo	gui1	misn	natch	00:00:13	
						3	start GUI checkpc	gui2	—		00:00:13	
					-	3	end GUI checkpo	gui2	ОК		00:00:14	
						5	start GUI checkpc	gui3	-		00:00:14	
						65	file compare	D:\Program Files\Mercury	misn	natch	00:00:15	-
		Î										
		(D (2) 3	4							

Find

WebTest User's Guide

- In the Test Results section you can see whether the test passed or failed, and how many GUI checkpoints were included in the test.
- In the test log, look for GUI check statements. Failed GUI checkpoints appear in red; passed GUI checkpoints appear in green.
- If you created a checkpoint to verify the text content of a cell or a frame, a file compare statement will appear above that GUI checkpoint statement.
- **2** By default, the Test Results window displays the results of the most recently executed test run.

To view other test run results, click the results directory box and select a test run.

- 3 To view a text version of the test results, choose Tools > Text Report from the Test Results window; the report opens in Notepad.
- 4 To view only specific types of results in the events column in the test log, choose Options > Filters or click the Filters icon.
- 5 To print test results directly from the Test Results window, choose File > Print or click the Print icon.

In the Print dialog box, choose the number of copies you want to print and click OK. Test results print in a text format.

6 To close the Test Results window, choose File > Exit.



Find



Viewing Results of a Contents Check

You can view the results of a contents check of a cell or a frame using the WDiff utility. This utility is accessed from the WinRunner Test Results window.

To view the results of a contents check:

- 1 Open the Test Results window.
- 2 Double-click a **file compare** entry in the **Event** column. The WDiff utility window opens. For an entry with no mismatch, the Notepad utility window opens.

The WDiff utility displays the expected and actual results. Differences are highlighted.

∰ ₩Diff	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>S</u> plit <u>O</u> ptions <u>I</u> nfo <u>H</u> elp	
D:\Program Files\Mercury Interactive\WinRunner\tmp	D:\Program Files\Mercury Interactive\WinRunner\tm;
WinRunner® is an enterprise functional user interactions automatically, WinRun remain reliable.	WinRunner© is an enterprise functional user interactions automatically, WinRun remain reliable.
Integrated with all of today's leading Java, and traditional GUI development t Includes sophisticated support for reco Powerful integrated scripting environme throughout your application lifecycle Organizes and structures test results f you to manage the complete testing proc	Information technology systems are revo want these systems not because they fea business processes, streamline operatio competitive advantage. IT systems provi mission-critical applications continue to business needs. In addition, the het connected to terminals, workgroups usin applications along with offsite Interne be responsible for testing all the diff enhancement, bug fix, etc.?
•	► //ı



Find



Chapter 5, page 57

- 3 To see the next mismatch in a file, choose View > Next Difference, or press Tab. The window scrolls to the next highlighted line. To see the previous difference, choose View > Previous Difference or press the Backspace key.
- 4 To view only the lines in the files that contain mismatches, choose Options > View > Hide Matching Areas, (a check mark at the left side of the menu identifies the current state). The window shows only the highlighted parts of both files.
- 5 To modify the way the actual and expected results text files are compared, choose Options > File Comparison.

File Comparison Options	×			
🗖 Ignore <u>s</u> paces on comparison				
☑ Ignore trailing blanks				
☑ <u>E</u> ×pand tabs before comparison				
□ <u>C</u> ase insensitive compare				
8 <u>T</u> absize				
Ok				





Note that when you modify any of the options, the two text files are read and compared again.

- Ignore spaces on comparison: Tab characters and spaces are ignored on comparison.
- **Ignore trailing blanks** (default): One or more blanks at the end of a line are ignored during the comparison.
- Expand tabs before comparison (default): Tab characters (hex 09) in the text are expanded to the number of spaces which are necessary to reach the next tab stop. The number of spaces between tab stops is specified in the **Tabsize** parameter. This **expand tabs before comparison** option will be ignored, if the **Ignore spaces on comparison** option is selected at the same time.
- Case insensitive compare: Uppercase and lowercase is ignored during comparison of text files.
- **Tabsize:** The tabsize (number of spaces between tab stops) is selected between 1 and 19 spaces. The default size is 8 spaces. The option influences the file comparison, if the **expand tabs before comparison** options is also set. Tabs are always expanded to the given number of spaces in the Difference Display form.
- 6 Choose File > Exit to close the WDiff utility.



Find



Viewing Results of a Check on Links, Images, or Tables

You can view the results of GUI checkpoints on links, images in cells or frames, or table contents using the Data Comparison Viewer and Expected Data Viewer. For a check with no mismatches, the Expected Data Viewer opens. When you have a check with a mismatch the Data Comparison Viewer opens.

To view the results of checks on links, images, or tables:

- 1 Open the Test Results window. In the test log, look for an **end GUI checkpoint** entry in the **Event** column.
- 2 Double-click on an end GUI checkpoint entry. The GUI Checkpoint Results dialog box opens.

	🖆 GUI Checkpoint Results		×
	Objects	Properties	7
	Powerful Test Automation for the Ente	Name Expected Value Actual Value	NERP RCT W
lian love the requilte		Kate Kate Kate Kate Kate Kate Kate Kate	ACT I
splays the results			
			<u>×</u>
			<u></u>
			==
		I F	
	Highlight Selected Object	<u>O</u> K <u>C</u> ancel <u>H</u> elp	



Find



3 In the Properties column, click the name of the check and click **Display**, or doubleclick the name.

For a check with no mismatches, the Expected Data Viewer opens. When you have a check with a mismatch the Data Comparison Viewer opens.

4 If the Expected Data Viewer opens, it shows expected results only. To close the viewer, choose File > Exit.





5 If the **Data Comparison Viewer** opens, it shows both expected and actual results. All cells are color coded, and all errors and mismatches are listed at the bottom of the viewer.







Use the following color codes to interpret the differences that are highlighted in your window:

- Blue on white background: Cell was included in the comparison and no mismatch was found.
- Cyan on ivory background: Cell was not included in the comparison.
- Red on yellow background: Cell contains a mismatch.
- Magenta on green background: Cell was verified but not found in the corresponding table.
- Background color only: cell is empty (no text).
- 6 By default, scrolling between the expected and actual tables in the Data Comparison Viewer is synchronized. When you click any cell, the corresponding cell in the other table flashes red.

To scroll through the tables separately, clear **Synchronize Scrolling** from the **Utilities** menu. Use the scroll bar as needed to view hidden parts of the table.





- **7** To filter a list of errors and mismatches that appear at the bottom of the Differences Display form, use the following options:
 - To view mismatches for a specific column only: Double-click a column heading (the column name) in either table.
 - To view mismatches for a single row: Double-click a row number in either table.
 - To view mismatches for a single cell: Double-click a cell with a mismatch.
 - To see the full list of mismatches: Click Full List in the Utilities menu or double-click the empty cell in the upper left corner of the table.
 - To clear the list: Double-click a cell with no mismatch.
 - To see the cell(s) that correspond to a listed mismatch: Click a mismatch in the list at the bottom of the dialog box to see the corresponding cells in the table flash red. If the cell with the mismatch is not visible, one or both table scroll automatically to display it.
- 8 Choose File > Exit to close the Data Comparison Viewer.

Note: The cells in the Data Comparison Viewer can display up to 1,000 characters.





Viewing Results of a Check on the Source, Type, or Url of an Image

You can view the results of a GUI checkpoint on an image source, type, or Url.

To view results of checks on the source, type, or Url of an image:

- 1 Open the Test Results window. In the test log, look for an **end GUI checkpoint** entry in the Event column.
- 2 Double-click an **end GUI checkpoint** entry in the Event column. The GUI Checkpoint Results dialog box opens.

	🖆 GUI Checkpoint Results		×
	Objects	Properties	**
	Automated Testing Solutions	Name Expected V Actual Value	RCT
Compare expected		Mage: [mage: [me Image: [me	<u>हेता।</u>
and actual values			<u>~</u>
			<u>*</u>
		,,	1
	I Highlight Selected Object	<u>OK</u> <u>C</u> ancel <u>H</u> elp	





3 In the Properties column, click the name of the check and click the **Compare expected and actual values** button. The Compare Expected and Actual Values dialog box opens.

Compare Expected and Actual V	alues 🗙
Image: [merc_logo], Source	Image: [merc_logo], Source
	Ж



5 Click **OK** to close the GUI Checkpoint Results dialog box.



Viewing Results of a Table Size Check

You can view the results of a GUI checkpoint on rows and columns count in a table.

To view results of table size check:

- 1 Open the Test Results window. In the test log, look for an **end GUI checkpoint** entry in the Event column.
- 2 Double-click an **end GUI checkpoint** entry in the Event column. The GUI Checkpoint Results dialog box opens.

	🖆 GUI Checkpoint Results		×
	Objects	Properties	**
	🗹 🚍 Automated Testing Solutions	Name Expected V Actual Value	RCT
Compare expected		HTML Table: [mer Table: [mer	
and actual values			<u>×</u>
			<u>×-</u>
			<u>1</u>
	☑ Highlight Selected Object	<u>O</u> K <u>C</u> ancel <u>H</u> elp	





3 In the Properties column, click the name of the check and click the Compare expected and actual values button. The Compare Expected and Actual Values dialog box opens.





5 Click **OK** to close the GUI Checkpoint Results dialog box.



Viewing Results of a Standard Properties Check

You can view the results of a GUI checkpoint on standard properties using the GUI Checkpoint Results dialog box.

1 Open the Test Results window. In the test log, look for an **end GUI checkpoint** entry in the Event column.



2 Double-click an **end GUI checkpoint** entry in the Event column. The GUI Checkpoint Results dialog box opens.

GUI Checkpoint Results				×
Objects Properties				*
The Mercury Interactive Intrans	Name ♥ © Count ♥ © Enabled ■ ● Height ♥ © Width ♥ © X ♥ © X	Expected Value 3 ON 299 488 176 414	Actual Value 3 ON 275 488 176 414	
Highlight Selected Object	<u>0</u> K	<u>C</u> ancel	<u>H</u> elp	

For a standard checkpoint, the GUI Checkpoint Results dialog box lists the properties checked. Each check is marked as either passed or failed, and the expected and actual results are shown.

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

Find

Index

A

Analyzing test results 53 images in a cell or frame 60 links 60 page contents 57 source, type, or Url of an image 65 standard properties 69 table contents 60 table size 67

С

Cell Links Verification dialog box 30 Check GUI dialog box 46 Compare Expected and Actual Values dialog box 66, 68 Creating tests 11

D

Data Comparison viewer 60, 61

Ε

Expected Data viewer 60, 61

G

GUI Checkpoint Results dialog box 60, 65, 67, 70 GUI checkpoints 25 images in a cell or frame 35 links 29 page contents 27 source, type, or Url of an image 33 standard properties 45 table contents 39 table size 43 GUI Map Editor 7, 9 GUI objects checking 25 identifying 6

Н

HTML Images Verification dialog box 36 HTML Links Verification dialog box 30 HTML Table Verification dialog box 40







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

WebTest User's Guide

Index

Images

checking all images in a cell or a frame 35 checking the source, type, or Url of an image 33 viewing results of a check on images in a cell or a frame 60 viewing results of a check on the source, type, or Url of an image 65 Installing WebTest 8

L

Links

checking 29 viewing test results 60

Ρ

Page contents checking 27 viewing results 57 Planning tests 13

R

Recording tests 14

Run modes Debug mode 50 Update mode 51 Verify mode 50 Run Test dialog box 52 Running tests 48

S

set_window function 17, 19 Standard properties checking 45 viewing results 69

Т

Tables checking table contents 39 checking table size 43 viewing results of a table size check 67 viewing results of contents check 60 tbl_get_cell_data function 24 tbl_get_cols_count function 24 tbl_get_column_name function 24 tbl_get_rows_count function 24







page

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

WebTest User's Guide
Index

Test

analyzing results 53 creating 11 planning 13 recording 14 running 48 Test Results window 54, 55

V

Viewing test results 53 images in a cell or frame 60 links 60 page contents 57 source, type, or Url of an image 65 standard properties 69 table contents 60 table size 67

W

WDiff utility 57 Web Check GUI dialog box 27, 29, 33, 35, 39, 43, 45 web_browser_invoke function 21 web_cursor_to_image function 21 web_cursor_to_label function 21 web_cursor_to_link function 21 web file browse function 21 web file set function 22 web_frame_get_text function 22 web_get_timeout function 22 web_image_click function 19, 22 web label click function 22 web link click function 18, 22 web obj get child item function 23 web obj get child item count function 23 web obj get info function 23 web set timeout function 23 web sync function 23 web text exists function 23 WebTest identifying GUI objects 6 installation 8 introduction 4 invoking 10 using to test web sites 5 win check gui function 20





Click a page

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

WebTest User's Guide

WebTest User's Guide

© Copyright 1998 by Mercury Interactive Corporation

All rights reserved. All text and figures included in this publication are the exclusive property of Mercury Interactive Corporation, and may not be copied, reproduced, or used in any way without the express permission in writing of Mercury Interactive. Information in this document is subject to change without notice and does not represent a commitment on the part of Mercury Interactive.

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, Astra SiteManager, Astra SiteTest, 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. Find

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

WRWTUG5.01/01