JavaTest™ Harness User's Guide

Graphical User Interface

JavaTest Harness, 3.2.2_01

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, California 95054
U.S.A. 1-650-960-1300

August 2005

Contents

Preface 9

1. **What is the JavaTest Harness Graphical User Interface?** 1
   - JavaTest Harness GUI Features 1
   - For the New JavaTest Harness User 2
     - Providing Configuration Information 2
     - Online Documentation and Context Sensitive Help 2
2. **Before Starting the JavaTest Harness** 3
3. **Starting the JavaTest Harness** 4
   - Run a Startup Script 5
   - Execute the JAR File 5
   - Double-click the Icon 5
   - Invoke the Class Directly 5
4. **JavaTest Harness Online Help** 7
   - Accessing Help 8
   - Navigation 9
   - Quick Start Wizard 10
5. **JavaTest Harness GUI** 11
   - JavaTest Harness Desktop 11
     - Desktop Styles 12

- Desktop Windows 13
- Desktop Menus 14
    - File Menu 15
    - Windows Menu 16
    - Help Menu 17

Setting JavaTest Harness Preferences 17

- Appearance Preferences 19
    - Change Window Styles 19
    - Set Tool Tip Options 21
    - Change Shutdown Options 22
- Test Manager Preferences 22
    - Toolbar 23
    - View Filters 23
- Configuration Editor Preferences 23
    - Set the Configuration Editor Mode 24
    - Display or Hide the More Info Pane 24

Arranging the Window Layout 25

- Opening a Test Suite 25
- Creating a Work Directory 25
- Opening a Work Directory 26
- Displaying Online Information 27
    - Help Menu 27
    - Help Buttons 27
    - F1 Key 27

Keyboard Access 27

- Keyboard Focus 28
- Keyboard Shortcuts 28
    - Hotkey Shortcuts 28
- Keyboard Navigation 29
    - Navigation in Hyperlinked Text 30
    - Navigation in Folder Pane Status Tabs 31
    - Navigation in the Test Tree 31
- Keyboard Activation 31

- Test Manager Window 32
  - Status Message 34
  - Configure Menu 34
  - Run Tests Menu 36
  - Report Menu 36
  - View Menu 37
  - Toolbar 38

6. **Configuring a Test Run** 39
- Configuration Editor Window 40
  - Question Mode 40
    - Menus 41
  - Quick Set Mode 46
- Creating a Configuration 47
- Creating a Configuration Template 48
  - Save Configuration Template 49
  - ▼ Create a Configuration Template from an Existing Configuration 50
  - Create a Configuration Template as a New Configuration 51
- Saving a Configuration 52
- Loading a Configuration 52
- Changing Configuration Values 53
  - Specifying Tests to Run 53
    - Using the Test Tree to Specify Tests 54
    - Using a Load Test List to Specify Tests 55
  - Using Exclude Lists 56
    - Latest Exclude List 58
    - Other Exclude List 58
  - Using Keywords 59
  - Using Prior Status 60
  - Specifying a Test Environment 62
    - Files Area 63
    - Name Area 64
  - Setting Concurrency and Timeout Factor 64

            Concurrency 65
            Time Factor 65
        Using Bookmarks in Configurations 66
            Set Bookmarks for Specific Questions 66
            Display Only Questions With Bookmarks 66
            Open Groups of Questions 67
            Close Groups of Questions 67
            Clear the Values of All Questions With Bookmarks 67
            Clear the Value of a Specific Question 67
            Remove All Bookmarks 68
            Remove a Bookmark From a Question 68
    Searching the Configuration 68
    Working with Multiple Configurations 69
    Displaying Configuration Information 70
        Displaying the Configuration Checklist 70
        Using the Test Environment Browser 71
        Using the Exclude List Browser 72
            Exclude List Contents 73
            Test Details 73
        Displaying the Question Log 73

7. **Running Tests** 74
    Starting a Test Run 75
    Monitoring a Test Run 76
        Using the Test Tree 77
        Using the Progress Indicator 78
            Elapsed Time 78
            Progress Bar 78
        Using the Progress Monitor 79
            Progress 79
            Tests in Progress 80
            Memory 81
            Time 81

        Agent Monitor Window 81
            Agent Pool 82
            Agents Currently In Use 83
    Stopping a Test Run 84
    Troubleshooting a Test Run 84
        Test Tree 85
        Folder View 85
        Test View 85

8. Browsing Test Information 87

    Test Tree 88
        Displaying Current Status and State of Tests 88
        Choosing Tests and Folders to Run, Refresh, or Clear Results 89
        Folder Icons 89
            Folder Run Status 89
            Folder Result Status 90
        Test Icons 91
            Test Run Status 91
            Test Result Status 91
        Test Tree Pop-up Menu 92
            Quick Pick Test Execution 93
            Refresh Test Suite Contents 93
            Clear Previous Test Results 95
    Folder View 96
        Summary Information 97
        Status Information 98
            Passed 99
            Failed 99
            Error 99
            Not Run 99
            Filtered Out 100
        Multi-Selection Panel 100
    Test View 100

- Test Description Pane 102
  - Name 102
  - Value 102
- Files Pane 102
- Configuration Pane 103
  - Name 103
  - Value 103
- Test Run Details Pane 103
  - Name 103
  - Value 104
- Test Run Messages Pane 104
  - Message List 104
  - Message Area 107
  - Test Result Status Bar 107

Using View Filters 107

- All Tests View Filter 108
- Last Test Run View Filter 109
- Current Configuration View Filter 109
- Custom View Filter 110
  - Edit the Custom Filter 111
  - Use a Custom View Filter 115

Viewing Test Suite Errors 115

Viewing Test Manager Properties 115

- Test Suite 116
- Work Directory 117
- Configuration 117
- Plug-Ins 117

9. Test Reports 118

Creating Reports 118

- Report Formats 120
- HTML Options 120
- HTML Files 121

Other Options 122
Displaying Reports 123
Display Reports in the Report Browser 123
Display Reports Offline 125

10. Auditing Test Results 126
Auditing Test Results in the GUI 126
Auditing Test Results from the Command-Line Interface 127
Setting Audit Options 127
Test Suite 127
Work Directory 128
Configuration File 128
Start Audit Button 129
Cancel Button 129
Help Button 129
Audit Test Results Window 129
Summary 131
Bad Result File 132
Bad Checksum 132
Bad Test Description 132
Bad Test Cases 132

11. Troubleshooting 133
Problems Using the JavaTest Harness 133
Problems Running Tests 133
Tests with Errors 134
Tests that Fail 135
Problems Viewing Test Results 135
Output Overflow Message Displayed in Test Run Messages 136
Problems Viewing Reports 137
Problems Writing Reports 137
Problems Moving Reports 137

Preface

This manual describes how to use the JavaTest harness graphical user interface (GUI) to run tests of the test suite, browse results, write reports, and audit test results. This User's Guide is a PDF version of the JavaTest GUI online help. It is provided in PDF format so that users can conveniently view and print the contents of the GUI online help without starting the JavaTest harness.

There are minor structural differences between the online help and the PDF document although the basic contents are the same. For example, the contents of the online help have been resequenced to be more useful in book format; in the PDF format, page references embedded in the text are hypertext links in the online help; and, extensive online help navigation links have been removed from the PDF format.

The *JavaTest User's Guide: Graphical User Interface* is one of two User's Guides that the JavaTest harness provides for users, the *JavaTest User's Guide: Graphical User Interface* and the *JavaTest User's Guide: Command-Line Interface.*

If your test suite provides the JavaTest agent for use in running tests, the *JavaTest Agent Users' Guide* may also be included.

Before You Read This Book

In order to fully use the information in this document, you must have thorough knowledge of the topics discussed in your TCK documentation.

How This Book Is Organized

Chapter 1 introduces the JavaTest harness

Chapter 2 describes the basic topics that users should be familiar with before starting the JavaTest harness on a test system.

Chapter 3 describes how to start the JavaTest harness Graphical User Interface.

Chapter 4 describes the JavaTest online help system.

Chapter 5 describes the JavaTest Graphical User Interface.

Chapter 6 describes how to create a configuration used by the JavaTest harness to run tests.

Chapter 7 describes how to use the GUI to run tests.

Chapter 8 describes how to use the GUI to browse test information.

Chapter 9 describes how to use the GUI to generate test reports.

Chapter 10 describes how to use the GUI to audit test results.

Chapter 11 describes how to use the GUI to troubleshoot a test run.

Using UNIX®Commands

This document may not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices.

See one or more of the following for this information:

- *Solaris Handbook for Sun Peripherals*
- AnswerBook2™ online documentation for the Solaris™ operating environment
- Other software documentation that you received with your system

Typographic Conventions

This User's Guide uses the following typographic conventions:

Typeface	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`

`AaBbCc123`	What you type, when contrasted with on-screen computer output	`%` **`su`**
		`Password:`

AaBbCc123	Book titles, new words or terms, words to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be superuser to do this.
	Command-line variable; replace with a real name or value	To delete a file, type `rm` *filename*.

Shell Prompts

Examples in this User's Guide might contain the following shell prompts:

Shell	Prompt
C shell	*machine_name*`%`
C shell superuser	*machine_name*`#`
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell superuser	`#`

Related Documentation

The following documentation provides additional detailed information about the JavaTest harness:

Application	Title
JavaTest command-line interface	*JavaTest Harness User's Guide: Command-Line Interface*
JavaTest Agent (optional)	*JavaTest Agent User's Guide*

Accessing Sun Documentation Online

The Java Developer Connection™ program web site enables you to access Java platform technical documentation at `http://java.sun.com/`.

Sun Welcomes Your Comments

We are interested in improving our documentation and welcome your comments and suggestions. Provide feedback to Sun at `http://java.sun.com/docs/forms/sendusmail.html`.

1

What is the JavaTest Harness Graphical User Interface?

The JavaTest harness is a powerful test harness that provides two interfaces, a Graphical User Interface (GUI) and a command-line interface. The GUI provides a desktop that contains a set of interactive windows and menus that you use to configure and execute tests, control and monitor agents, audit test results, and create reports.

Note – The JavaTest harness also provides a command-line interface that can be used to perform many of the same tasks as those performed in the GUI. See the *JavaTest Harness User's Guide: Command-Line Interface* for a description of the JavaTest harness command-line interface.

JavaTest Harness GUI Features

Features of the GUI include a desktop that contains a set of interactive windows and menus that you use to perform the following tasks:

- Configure, run, and monitor tests on a variety of test platforms (such as servers, workstations, browsers, and small devices) with a variety of test execution models (such as API compatibility tests, language tests, compiler tests, and regression tests).
- Evaluate and analyze test results.
- Create template configuration files.
- Generate and view test reports that summarize test runs.
- Audit test results and view audit results.
- Monitor agents (only enabled when the test suite provides an agent).The JavaTest harness supports the use of an agent to run tests on systems that can't run the JavaTest harness.
- Display online help that describes how to use the JavaTest harness. The JavaTest harness online help provides context sensitive help, full-text search, and keyword search.

For the New JavaTest Harness User

The JavaTest harness is designed to run test programs on a wide-variety of Java platforms. To run the tests, the JavaTest harness uses a configuration file that contains the required information about how the tests are run on a particular test platform. The JavaTest harness uses a specific work directory to contain all of the results from running the tests of a testsuite. See the Glossary for detailed descriptions of the terms **test suite**, **work directory**, and **configuration file**.

Providing Configuration Information

The JavaTest harness uses the GUI Configuration Editor to collect configuration information (both test environment and parameter values) in a single configuration file. While it is possible for existing test suites to use environment files (`.jte`) in the Configuration Editor, parameter files (`.jtp`) are no longer used. The JavaTest harness uses the Configuration Editor to include the parameter values with the values of the `.jte` file in a single configuration file (`.jti`).

For backwards compatibility, older test suites can continue to use environment and parameter files in the command line. See the Glossary for detailed descriptions of the terms `.jte` file, `.jtp` file, and `.jti` file.

Online Documentation and Context Sensitive Help

The JavaTest harness online help provides context sensitive help, full-text search, and keyword search. The online help is also available from the command line. See the *JavaTest Harness User's Guide: Command-Line Interface* for detailed information about searching for and displaying omline help without starting the JavaTest harness GUI.

2

Before Starting the JavaTest Harness

Before you start the JavaTest harness on a test system and run tests, you must have a valid test suite and Java Development Kit 1.4 or later installed on your test system. See your test suite documentation for information about installing the test suite and the JavaTest harness on your test system. Refer to `http://java.sun.com/products` for information about installing the current Java Development Kit on your test system.

You must also understand how your test group uses or intends to use the JavaTest harness in its test system. For example consider the following questions about your test groups use of the JavaTest harness:

- Does your test group use standard configuration files and templates from a central location, or does it use individual configuration files customized for each test run?
- Does your test group use the JavaTest harness and one or more agents to run distributed tests?
- Does your test group run the JavaTest harness from a central location or from local installations in the test system?

If you use the JavaTest harness agent to run tests, you must also install the JavaTest harness agent on the platform being tested. See *JavaTest harness Agent User's Guide* for detailed information about installing the JavaTest harness agent on a test platform.

3

Starting the JavaTest Harness

There are two interfaces that you can use to run the JavaTest harness, the graphical user interface (GUI) and the command-line interface. The following chapters provide detailed information about using the GUI. See the *JavaTest Harness User's Guide: Command-Line Interface* for detailed information about using the command-line interface.

When you run the harness with the GUI, it provides a desktop that contains an integrated set of interactive windows and controls for you to do the following tasks:

- Configure, run, and monitor tests
- Generate and view test reports
- Monitor agents
- Audit test results and view audit results

Note – To simplify the classpaths used to run tests, start the JavaTest harness from the writeable directory where you intend to create files and store test results. Refer to your test suite documentation for the specific instructions required to start the JavaTest harness for your installation.

You can use any of the following actions to start the JavaTest harness in GUI mode:

- Run a startup script
- Execute the JAR file
- Double-click the icon
- Invoke the class directly

Unless you are starting the harness for the first time or use a command-line option, the JavaTest harness restores your previous desktop when it starts the GUI. When the JavaTest harness cannot restore an existing desktop, it displays a Quick Start wizard that guides you through the startup.

Run a Startup Script

Use a startup script if your test suite includes one. Refer to your test suite documentation for details about using a startup script.

Execute the JAR File

If you have access to a command line, you can directly execute the JAR file from the directory where you intend to create files and store test results:

1. Make the directory where you intend to create files and store test results your current directory.
2. At the command prompt, type:

```
java -jar jt_dir/lib/JavaTest harness.jar
```

You must include the path of the directory where the `javatest.jar` file is installed (represented as *jt_dir/lib/* in the example). The `javatest.jar` file is usually installed in the TCK `lib` directory when the JavaTest harness is bundled with a TCK.

You can also change configuration values as part of the command line used to start the JavaTest harness GUI. See *JavaTest Harness User's Guide: Command-Line Interface* for information about using the command line to start the JavaTest harness GUI.

Double-click the Icon

If you are using a GUI, your system might support double clicking the `javatest.jar` file icon to launch the harness.

Invoke the Class Directly

If you have access to a command line and the JAR file is on the classpath, you can directly invoke the class from the directory where you intend to create files and store test results by making it the current directory and then, at the command prompt, typing the following command:

```
java com.sun.JavaTest harness.tool.Main
```

You must configure your classpath to include `javatest.jar`. See your TCK documentation.

4

JavaTest Harness Online Help

JavaTest harness online help describes how to use the JavaTest harness to run test suites and evaluate test results. This page describes how to access online help and how to navigate through the help topics to find the information you want.

The JavaTest harness online help is also provided in the following PDF format documents available at `doc/javatest/` in your TCK installation directory:

- *JavaTest Harness User's Guide: Graphical User Interface* contains JavaTest harness GUI information.
- *JavaTest Harness User's Guide: Command-Line Interface* contains JavaTest harness command-line and utility information.
- *JavaTest harness Agent User's Guide* contains JavaTest harness agent information (if supplied by the test suite).

Accessing Help

The following table describes user actions in the GUI that display online help and information about the JavaTest harness:

TABLE 1 Accessing JavaTest Harness Help and Information

Action	Description
F1 key	Press the F1 key to display information about the JavaTest harness window that has keyboard focus. You must establish keyboard focus in a window before pressing the F1 key. To establish focus, you might have to highlight something in the window.
Help menu	The Help menu lists the available online documentation. In addition to documentation provided by the JavaTest harness, test suites can also provide online documentation. The JavaTest harness lists the JavaTest harness documentation followed by any documentation provided by the test suite. Choose Help -> JavaTest Online Help from the menu bar to display the online documentation for the GUI. To display the help page for the window that has focus (such as the Test Manager window), choose Help -> *window_name* from the menu bar. Use the navigators in the Help viewer's left pane to locate specific information.
Help buttons	Click the Help button in a dialog box for information about how to use that dialog box.

You can also display online help without starting the JavaTest harness. See Information Commands for detailed information.

Navigation

The following table describes the tabs in the left pane of the help viewer that are used to navigate through the help information.

TABLE 2 Online Help Tabs

Tab	Description
Contents	Table of contents. Topics are organized by task (where possible). Topics displayed in the right pane are highlighted in the table of contents. You can also add pages from the Contents to the Bookmarks section of the Help viewer. See Bookmarks for detailed information.
Index	Keyword index. Index entries are listed in alphabetical order. To search for entries, enter a string of characters in the Find field and press Return. If the string is found in the index, it is highlighted (press Return again to repeat the search).
Glossary	Glossary. Glossary entries are listed in alphabetical order. You can scroll through the list or search for a term. To search for a term, enter a string of characters in the Find field and press Return. If the term is found in the glossary, it is highlighted in the list and its definition is displayed. Press Return to repeat the search.
Search	Full-text search. Type a natural language phrase in the Find field and press return. A ranked list of topics is returned with the following information: • The circle in the first column indicates the ranking of the matches for that topic. The more filled-in the circle is, the higher the ranking. There are five possible rankings (from highest to lowest): ● ◕ ◒ ◔ ○ • The number in the second column indicates the number of times the query was matched in the listed topic. • The title of the topic in which matches are found is listed as it appears in the table of contents.
Bookmarks	You can add a topic from the Contents to the Bookmarks section of the Help viewer. Click the topic name in the Contents tab and then click the Bookmarks button ▶ on the toolbar. The topic is listed in the Bookmarks panel. You can then use the Bookmarks list as a customized Table of Contents. Right click an item in the Bookmarks list to display a pop-up menu. Use the pop-up menu to manage items in the Bookmarks list.

Quick Start Wizard

The JavaTest harness displays the Quick Start Wizard whenever a user starts the GUI with a new desktop or chooses File **->** Open Quick Start Wizard from the menu bar.

The Quick Start Wizard allows users to rapidly accomplish the following tasks:

- Select a test suite, work directory, and a configuration to use for running tests
- Resume work on a previous test run
- Browse the contents of a test suite without specifying a work directory or a configuration

If the configuration is new or incomplete, the JavaTest harness allows the user to automatically open the configuration editor window after the Quick Start Wizard closes.

Users can also choose to automatically begin a test run after the Quick Start Wizard closes.

5

JavaTest Harness GUI

The JavaTest harness GUI provides a desktop containing a set of windows and menus used to configure and run tests, monitor test status, evaluate and analyze test results, and include or exclude tests from test runs.

Unless you are starting the harness for the first time or use a command-line option to start the harness, the JavaTest harness restores your previous desktop when it displays the GUI. See the *JavaTest Harness User's Guide: Command-Line Interface* for detailed information. If the JavaTest harness cannot display an existing desktop, it displays the JavaTest harness Quick Start Wizard. The Quick Start Wizard guides you through starting the GUI.

Note – For a description of the agent user interfaces provided by the JavaTest harness, see *JavaTest Harness Agent User's Guide*.

See JavaTest Harness Desktop for a detailed description of the windows and menus provided by the desktop. For information about using the JavaTest harness GUI to perform tasks, see the following topics:

- Configuring a Test Run
- Running Tests
- Browsing Test Information
- Test Reports
- Auditing Test Results

JavaTest Harness Desktop

The JavaTest harness desktop provides a set of windows and menus for you to use when loading, editing, or creating a configuration, when running and monitoring tests, when browsing test results, when creating and viewing reports, and when auditing test results. For detailed information about the windows and menus provided by the desktop, see the following topics:

- Desktop Windows

- Desktop Menus

Desktop Styles

You can display the desktop in one of three styles:

- **Tabbed** - A single top-level desktop displays the JavaTest harness windows as tabbed panes
- **Multiple Document Interface (MDI)** - A single top-level desktop contains separate JavaTest harness windows
- **Single Document Interface (SDI)** - A separate top-level JavaTest harness console and separate top-level JavaTest harness windows

For detailed information about setting display styles and functional options of the JavaTest harness, see Setting JavaTest Harness Preferences.

See the following topics for other detailed information about using the desktop:

- **Creating a Work Directory** - Describes how to use the desktop menus to create a work directory.
- **Opening a Work Directory** - Describes how to use the desktop menus to open a work directory.
- **Opening a Test Suite** - Describes how to use the desktop menus to open a test suite.
- **Arranging the Window Layout** - Describes how to change the layout of the open windows on the desktop when using the MDI and SDI window styles.
- **Displaying Online Information** - Describes how to use the desktop menus to display available online information, including JavaTest harness online help and additional documentation provided by the test suite.
- **Keyboard Access** - Describes how to use the keyboard to traverse and access desktop windows and components.
- **Test Manager Window** - Describes the window and menus used to run tests, write reports, and audit tests.

To locate specific information about a task, use the JavaTest harness online help full-text search and keyword search index tabs. A glossary of terms used in the documentation is also provided. See JavaTest Harness Online Help for a description of the online help features.

Desktop Windows

The following table describes the windows provided by the JavaTest harness desktop.

TABLE 3 Desktop Windows

Window	Description
Test Manager Window	Contains panels, menus, and controls that you use to do the following: • Open and create work directories • Create or modify information that the JavaTest harness uses when running your tests • Run the tests of a test suite • Monitor tests and test results while they are being run • View test environment settings of your configuration • View the contents of an exclude list • Browse completed tests and test results During a test run, the icons in the Test Manager window change to reflect the test status. You can also use the window during and after a test run to browse information about individual tests.
Audit Test Results Window	Contains panels and menus that you use to generate and view audit reports of the tests in a work directory.
Agent Monitor Window (optional)	This window is only available for use when the test suite is configured to use the JavaTest harness agent. See the *JavaTest Harness Agent User's Guide* for detailed information about using the agent and the Agent Monitor window.

Desktop Menus

The following table describes the two types of menus that the JavaTest harness provides for you to use when performing tasks.

TABLE 4 JavaTest Menus

Menu Type	Description
Desktop Menus	These menus are displayed by the top-level window or windows set by the current desktop style and are always available. They include the following menus: • File Menu • Windows Menu • Help Menu
Tool Menus	These menus are unique to specific tools and are only displayed in the appropriate tool window. For a description of the Test Manager menus, see Test Manager Window . For a description of the Audit Test Results menus, see Audit Test Results Window . If your test suite uses the JavaTest harness agent, see the *JavaTest harness Agent Monitor User's Guide* for a description of the Agent Monitor menus.

File Menu

Use the File menu to open files, set user preferences, close windows, and exit from the JavaTest harness. The contents of the File menu change dynamically, based on the context of the desktop. The JavaTest harness only enables menus when they can be used. The following table describes the contents of the File menu.

TABLE 5 File Menu Contents

Menu Item	Description
Open Quick Start Wizard	Opens the Quick Start Wizard for you to use for the following tasks: • Selecting a test suite, work directory, and a configuration file to use for running tests. If the configuration file is new or incomplete, the JavaTest harness opens the configuration editor window after the Quick Start Wizard closes. • Resuming work on a previous test run or browse a test suite without running tests. • Browsing the contents of a test suite.
Create Work Directory	Opens a file chooser dialog box for you to use in creating a new work directory for the current test suite. When you create a work directory, the JavaTest harness associates it with the current test suite. Each work directory is associated with a specific test suite and contain info about results from previous runs. The test result files contain all of the information gathered by the JavaTest harness during test runs.
Open Work Directory	Opens a file chooser dialog box you can use to choose an existing work directory. The JavaTest harness changes the icon in the file chooser for any recognized work directories. The JavaTest harness associates the work directory with an open test suite if the test suite is both a match and has no other work directory already open. If the JavaTest harness cannot associate the work directory with an open test suite, it opens a new Test Manager window and loads both the work directory and its associated test suite.
Open Test Suite	Opens a file chooser dialog box you can use to choose a test suite. The JavaTest harness changes the icon in the file chooser for any recognized test suites. When you choose a test suite, the JavaTest harness loads the test suite in an empty Test Manager window.
Preferences	Opens the JavaTest harness Preferences dialog box for you to set the display and functional options of the JavaTest harness.

TABLE 5 File Menu Contents

File History	Displays a list of work directories and test suites that have been opened. Choose a file from the list to open a new instance of it in the current session.
Close	Closes the current window without exiting from the JavaTest harness. Shown in the File menu only when the window style is set to Tabbed or SDI. See Change Window Styles for a description of the Tabbed and SDI window styles provided by the GUI. Closing a Test Manager window closes a test session. To start another session using the same test suite and work directory, use the Open Work Directory menu above.
Exit	Exits from the JavaTest harness. When you exit, your current desktop is saved so that all open windows can be restored in your next JavaTest harness session.

Windows Menu

The Windows menu contains an Open submenu that you can use to open additional windows required to perform specific tasks. The Windows menu also provides a list of all windows currently open in the JavaTest harness. You can navigate to any open window by clicking on its name in the list.

In MDI and SDI window styles, you can also use the Windows menu to manage the layout of the open windows. See Arranging the Windows Layout for a description of how you can use the Windows menu to manage the window layout. See Change Window Styles for a description of the MDI and SDI window styles provided by the GUI.

TABLE 6 Contents of the Windows Open Menu

Menu Item	Description
Agent Monitor	See the *JavaTest Harness Agent User's Guide* for a detailed description.
Test Results Auditor	Opens the Test Results Auditor window. Only one Test Results Auditor window can be open.

Help Menu

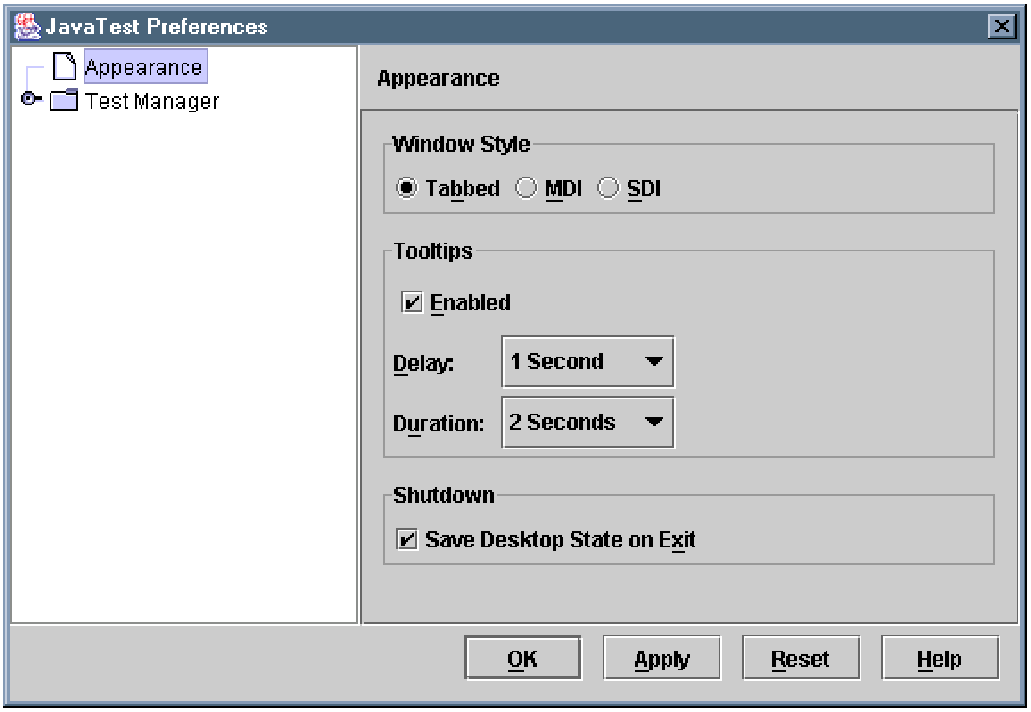
Use the Help menu to display online help for the window, JavaTest harness online help, available test suite documentation, JavaTest harness information, and current Java runtime information. Test suite supplied entries may also be listed in the Help menu.

TABLE 7 Contents of the Help Menu

Menu Item	Description
Active Window Name	Opens the help viewer and displays online help for the active window. This menu item is only available in an open tool window.
JavaTest harness Online Help	Opens the help viewer and displays online help for the JavaTest harness.
Test Suite Documentation (optional)	If the test suite provides online documentation, the JavaTest harness lists it here. Clicking the document name opens the help viewer and displays the test suite online document.
About the JavaTest Harness	Displays information about this release of the JavaTest harness.
About the Java Virtual Machine	Displays information about the runtime used to run the JavaTest harness.

Setting JavaTest Harness Preferences

You can use the JavaTest Preferences dialog box to set the display and functional options of the JavaTest harness.



Open the JavaTest Preferences dialog box by choosing File -> Preferences from the menu bar.

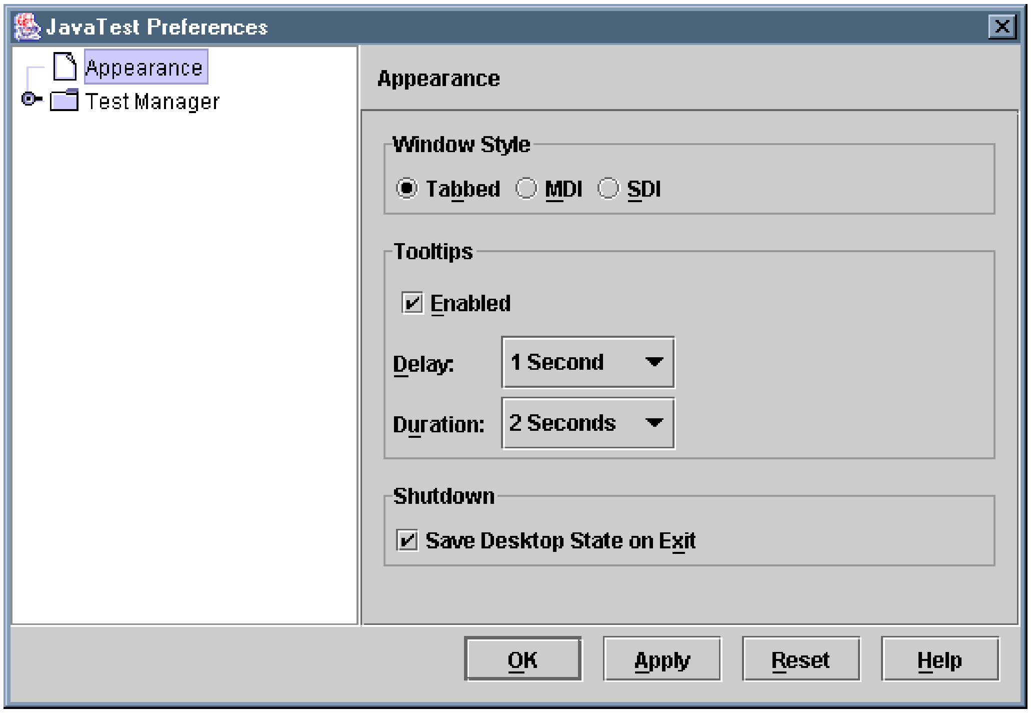
The following table describes the contents of the JavaTest Preferences dialog box.

TABLE 8 JavaTest Preferences Dialog Box Contents

Preference Category	**Description**
Appearance	Sets how the JavaTest harness graphical user interface displays its windows and tool tips. See Appearance Preferences for detailed information.
Test Manager	Sets the Test Manager toolbar property. See Test Manager Preferences for detailed information.
Configuration Editor	Sets the Configuration Editor properties. See Configuration Editor Preferences for detailed information.

Appearance Preferences

To change the appearance preferences for the JavaTest harness desktop, choose File -> Preferences from the menu bar to open the JavaTest Preferences dialog box.



Window Style

● Tabbed ○ MDI ○ SDI

Tooltips

☑ Enabled

Delay: 1 Second

Duration: 2 Seconds

Shutdown

☑ Save Desktop State on Exit

Click Appearances in the left panel and use the dialog box to set the following appearance preferences:

- Change Window Styles
- Set Tool Tip Options
- Change Shutdown Options

Change Window Styles

The following table describes the user-defined styles in which the JavaTest harness desktop displays its windows and menus.

TABLE 9 Window Styles

Window Style	Description

TABLE 9 Window Styles

Tabbed	A single top-level window that displays individual tool windows as tabbed panes
MDI	A single top-level desktop window that contains individual tool windows
SDI	A console window and individual tool windows displayed as top-level windows

Tabbed

When you choose the Tabbed window style, the JavaTest harness displays the opened tool windows as a set of tabbed panes within a single frame or desktop.

Tabs at the bottom of each pane allow you to choose the pane that is active and displayed on the top of the stack.

The desktop contains the complete set of menus including any tool menus used to perform tasks from the active pane. See JavaTest Harness Desktop Menus for a description of the two types of menus that the JavaTest harness provides.

The following are some advantages of using the Tabbed window style:

- Multiple open windows are managed from a single location
- The small footprint of the desktop window minimizes obstruction of windows from other applications
- If multiple windows of multiple applications are open, tabbed windows provide visual organization

The following are some disadvantages of using the tabbed style:

- Only one window can be viewed at a time
- Performing tasks on multiple open test suites might be confusing

MDI

When you choose the Multiple Document Interface (MDI) window style, all of the tool windows that the JavaTest harness opens to perform a task are contained within a single desktop.

The desktop contains the standard menus and each tool window contains only the tool menus used to perform tasks appropriate for that window. See JavaTest Harness Menus for a description of the two types of menus that the JavaTest harness provides.

The following are some advantages of using the MDI window style:

- Open windows are contained in a top-level window allowing simple window management
- Multiple windows can be viewed at the same time

- Multiple test suites can be open at the same time and easily monitored

The following are some disadvantages of using the MDI style:

- The single top-level desktop has a large footprint that might obstruct open windows of other applications and slightly decrease usable space
- Open windows cannot be positioned outside the boundary of the desktop

SDI

When you choose the Single Document Interface (SDI) window style, the JavaTest harness opens a console window and individual tool windows as separate top-level windows.

Each window contains the complete set of menus including any tool menus used to perform tasks from that window. See JavaTest Harness Menus for a description of the two types of menus that the JavaTest harness provides.

The following are some advantages of using the SDI window style:

- Individual windows can be positioned anywhere on the screen
- Obstruction of windows from other applications is minimized
- Multiple test suites can be opened and easily monitored at the same time

The following are some disadvantages of using the SDI style:

- Windows must be managed individually
- When multiple windows of multiple applications are open, the display might be visually confusing

Set Tool Tip Options

You can set the tool tip options from the Appearance category of the JavaTest Preferences dialog box.

The Tool Tips area contains combo boxes and a check box that you can use to specify how tool tips function in the desktop. The following table describes the available tool tip options.

TABLE 10 Setting Tool Tip Options

Option	Description
Enabled	Use the check-box to enable or disable tool tips for the desktop.
Delay	Use the combo box to select the delay interval before displaying tool tips.
Persistence	Use the combo box to select the duration that the GUI displays a tool tip.

Change Shutdown Options

This option is used each time you start the JavaTest harness.

Check the Save Desktop State on Exit option to use your current desktop in your next test session. Current view filter information is saved when you check Save Desktop State on Exit.

If you uncheck Save Desktop State on Exit and exit the JavaTest harness, the JavaTest harness uses the default Quick Start Wizard dialog and Test Manager window in your next test session. Current view filter information is not saved when you uncheck Save Desktop State on Exit and exit the JavaTest harness.

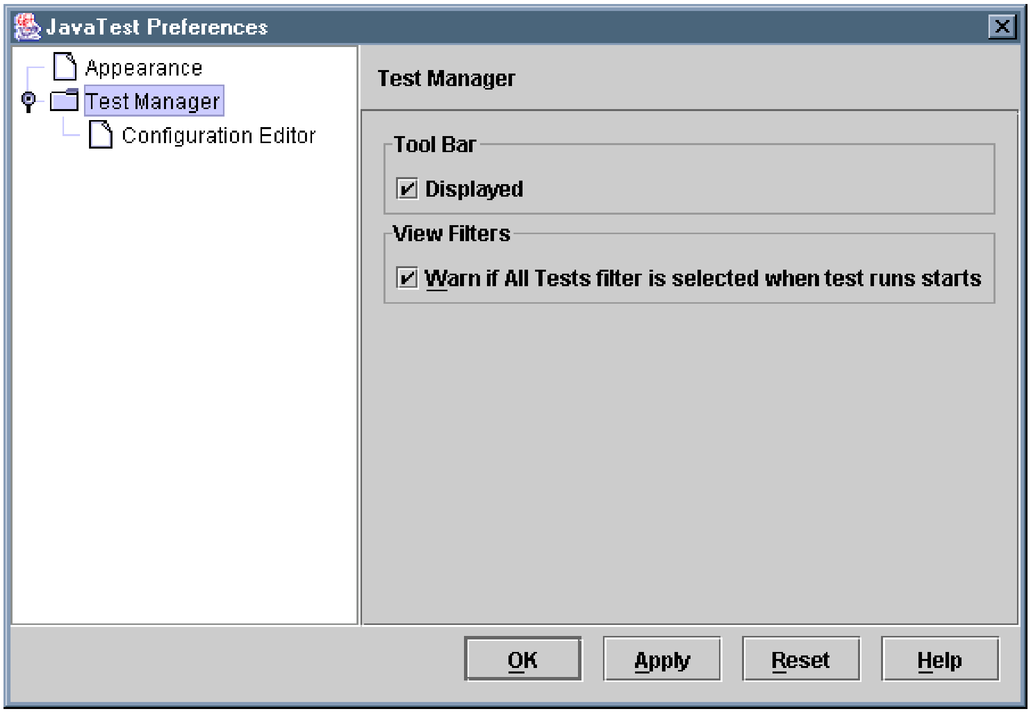
Test Manager Preferences

You can use the Test Manager preferences dialog box to display the Test Manger toolbar and to display a warning dialog when tests are run with the All Tests view filter.

To open the Test Manager preferences dialog box:

1. Choose File –> Preferences from the menu bar to open the JavaTest harness Preferences dialog box.

2. Click the Test Manager folder in the preferences tree on the left.



Test Manager

Tool Bar

☑ Displayed

View Filters

☑ Warn if All Tests filter is selected when test runs starts

Toolbar

To turn the Test Manager toolbar on or of, open the Test Manager preferences dialog box and check the Displayed box in the toolbar area to display the Test Manager toolbar. Clear the check box to hide the toolbar.

This preference is used each time you start the JavaTest harness.

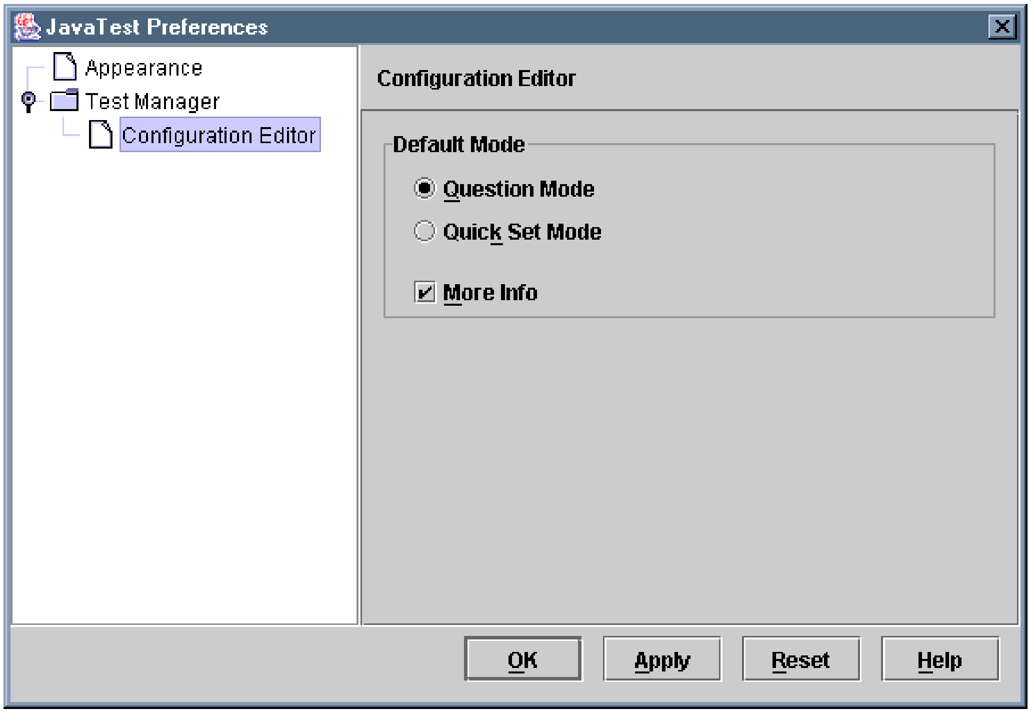
View Filters

To turn the All Tests filter warning dialog box on or off, open the Test Manager preferences dialog box and check the View Filters box to display a warning each time tests are run with the All Tests view filter. Clear the check box to turn off the warning dialog box.

This preference is used each time you start the JavaTest harness. See Using View Filters.

Configuration Editor Preferences

You can use the JavaTest Preferences dialog box to set the view the JavaTest harness uses when opening the Configuration Editor window and to show or hide the More Info pane.



Default Mode

● Question Mode

○ Quick Set Mode

☑ More Info

Set the Configuration Editor Mode

The JavaTest harness opens the Configuration Editor window in one of two user-specified modes:

- Question Mode
- Quick Set Mode

Choose either the Question Mode or the Quick Set Mode to set it as the mode used when opening the Configuration Editor window. This preference is used each time you start the JavaTest harness and open the Configuration Editor window.

Display or Hide the More Info Pane

The More Info pane contains detailed information about the questions and required settings presented in the Configuration Editor window. To display the More Info pane when the Configuration Editor window opens, More Info must be checked. To hide the More Info pane when the Configuration Editor window opens, More Info must not be checked.

Arranging the Window Layout

In MDI and SDI window styles, use the Windows menu to change the layout of the open windows on the desktop.The following table describes the available layout options.

TABLE 11 Managing JavaTest Windows

Menu Item	Description
Tile	Arranges the open windows edge-to-edge in a tiled pattern.
Cascade	Arranges the open windows so that their top-left corners form a cascading pattern from the top-left corner to the bottom-right corner of the desktop.

Opening a Test Suite

You can open any valid test suite by choosing File `->` Open Test Suite from the desktop menu.

The JavaTest harness displays the Open Test Suite dialog box. Use the dialog box to navigate to the location of the test suite.

Click or enter the name of the test suite. When the name of the test suite is displayed in the text field, click the Open button. When you open a test suite, the JavaTest harness loads the test suite in either a new or an empty Test Manager window.

Note – Before running tests, you must open or create a work directory for the test suite and provide a configuration.

Creating a Work Directory

Each work directory is associated with a specific test suite and stores its test result files. The test result files contain all of the information gathered by the JavaTest harness during test runs. You can create a new work directory for a new test suite or a new configuration. In some cases you might also choose to create a new work directory instead of using an existing work directory.

The following are some examples of occasions when you might choose to create a new work directory instead of using an existing work directory:

- The original work directory was created for the same test suite but with a different configuration
- The original work directory was created for a test suite similar to the current test suite (an old version of the test suite or a copy of the test suite from another directory)
- The original work directory was created for another test suite
- The original work directory is not a valid work directory

To create a new work directory, choose File -> Create Work Directory from the desktop menu.

The JavaTest harness displays the Create Work Directory dialog box. Use the dialog box to navigate to the location of the new work directory.

Enter the name of the new work directory in the text field and click the Create button.

When you create a new work directory, the JavaTest harness opens a new Test Manager window and that associates the new work directory with the current test suite.

Opening a Work Directory

You can open an existing work directory by choosing File -> Open Work Directory from the desktop menu.

The JavaTest harness displays the Open Work Directory dialog box. Use the dialog box to navigate to the location of the work directory.

Note – Each work directory is associated with a specific test suite and stores its test result files. The test result files contain all of the information gathered by the JavaTest harness during test runs.

Click or enter the name of the new work directory. When the name of the work directory is displayed in the text field, click the Create button.

If the JavaTest harness cannot associate the work directory with the open test suite, it opens a *new* Test Manager window and loads both the work directory and its test suite.

If the test suite is already associated with the work directory **and** has no other work directory already open, the JavaTest harness loads the work directory for use with the current test suite.

Displaying Online Information

You can display available online information in the following ways:

- Help menu
- Help buttons
- F1 key

Help Menu

The JavaTest harness desktop provides a Help menu that you can use to open online help, open available test suite documentation, display information about the JavaTest harness, display information about the test suite, and display information about the current runtime.

When multiple test suites are opened, the Help menu displays a menu item for each document provided by the test suite.

Help Buttons

The desktop provides a Help button on all window toolbars and in all dialog boxes:

- Toolbar help buttons ? open the help viewer and display online help for that window.
- Help buttons on dialog boxes open the help viewer and display online help for that dialog box.

F1 Key

Press the F1 key to see information about the window that has keyboard focus. Establish keyboard focus in a window before pressing the F1 key. In some cases, you might have to highlight something in the window to establish focus.

Keyboard Access

The JavaTest harness uses standard Java programming language key bindings for keyboard traversal and access of the GUI. See the *Java Look and Feel Design Guidelines* at `http://java.sun.com/products/jlf/` for a detailed description of the standard key bindings for keyboard traversal and access of the GUI.

The window or component must have keyboard focus before you can use keyboard navigation, activation, or shortcuts. Keyboard navigation, activation, and shortcut operations are described in the following topics.

Keyboard Focus

When a component has focus, it is generally displayed with a colored border or changes color. However, some components in the GUI cannot be displayed with a focus indicator. In this case, you must continue to use the keyboard to traverse the GUI until focus is displayed.

Keyboard Shortcuts

Keyboard shortcuts perform both navigation and activation in the same action. The following table lists the keys that are used to access menus and online help.

TABLE 12 Keyboard Shortcuts fo Navigation and Activation

Activation Keys	GUI Action
F1	Activates the JavaTest harness online help.
F10	Activates the File menu.
Shift F10	Activates the pop-up menu if focus is on a folder or test in the test tree.

Hotkey Shortcuts

The JavaTest harness provides shortcuts throughout the application for accessing menu titles, menu items, text fields, checkboxes, radio buttons, and command buttons.

Shortcut keys are identified in the following ways:

- **Text at the end of tool tips -** For example, Start Running Tests Alt-S
- **Underlined letters in menus and text buttons -** For example: File
- **Underlined letters in labeled fields -** For example: Response:

Keyboard Navigation

Keyboard navigation enables you to move keyboard focus from one GUI component to another by using the keyboard without activating the component. The following table lists the keys used for keyboard navigation.

TABLE 13 Keyboard Navigation Keys

Navigation Key	GUI Action
Tab	Navigates to the next focusable component in the GUI. The tab traversal order is generally left to right and top to bottom.
Shift-Tab	Navigates back to the next focusable component.
Control-Tab	Navigates to the next focusable component even if the current component accepts the Tab key as input (such as a text area).
Control-Shift- Tab	Navigates back to the next focusable component even if the current component accepts the Tab key as input (such as a text area).
left arrow	Moves keyboard focus left one character or component. If focus is in the test tree, focus moves up the tree and closes the node. If focus is on the splitter bar (F8 moves focus to the splitter bar), it moves the splitter bar left.
right arrow	Moves keyboard focus right one character or component. If focus is in the test tree, focus moves sequentially down the tree, opening the node and traversing all tests in a folder. If focus is on the splitter bar (F8 moves focus to the splitter bar), it moves the splitter bar right.
up arrow	Moves keyboard focus up one line or component. If focus is in the test tree, focus moves sequentially up the tree but does not open any closed folders. If focus is on the splitter bar (F8 moves focus to the splitter bar), it moves the splitter bar left.

TABLE 13 Keyboard Navigation Keys

down arrow	Moves keyboard focus down one line or component. If focus is in the test tree, focus moves sequentially down the tree but does not open any closed folders. If focus is on the splitter bar (F8 moves focus to the splitter bar), it moves the splitter bar right.
Page Up	Navigates up one pane of information within a scroll pane.
Page Down	Navigates down one pane of information within a scroll pane.
Home	Moves to the beginning of the data. In a table, moves to the beginning of a row. If focus is in the test tree, moves to the top of the tree.
End	Moves to the end of the data. In a table, moves to the last cell in a row. If focus is in the test tree, moves to the bottom of the tree.
Control-F1	Displays the tool tip information for the GUI object that has focus. Can be used to determine which GUI object has focus.
F6	Shifts focus between left and right panes.
F8	Shifts focus to the splitter bar if focus is in the left or right pane.
Control-T	Shifts focus to the next link in a topic or in a report.
Control-Shift-T	Shifts focus to the previous link in a topic or in a report.

Navigation in Hyperlinked Text

Navigating hyperlinks in text areas such as the More Info pane and report viewer requires that you establish focus in the pane itself. After you have established focus inside the pane the keyboard navigation keys listed in Keyboard Navigation above to navigate in the pane. Because some components in the GUI cannot display focus, you have to use the keyboard to traverse the GUI until you can determine that focus is established inside the pane.

After focus is established in the pane, use Control-T and Shift-Control-T to navigate to the next and previous link in the document. Use Control and Spacebar to select (follow) the hyperlink. See Keyboard Activation for a list of keys used for keyboard activation.

Navigation in Folder Pane Status Tabs

Navigating in the folder pane test status tabs requires that you establish focus in the pane itself.

After focus is established in the pane, use the arrow-up, arrow-down keys listed in Keyboard Navigation to navigate in the pane. When only a single entry is present in the folder pane, you must use the Home or End key to select the item.

Because some components in the GUI cannot be displayed with focus indicators, you may have to use the keyboard to traverse the GUI until you can determine that focus is established inside the folder pane.

Use the Return or Enter key to navigate to the selected item. See Keyboard Activation for a list of keys used for keyboard activation.

Navigation in the Test Tree

Navigating in a test tree requires that you establish focus in the test tree itself.

After focus is established in the pane, use arrow-up, arrow-down, arrow-left, and arrow-right keys listed in Keyboard Navigation to navigate in the pane.

Keyboard Activation

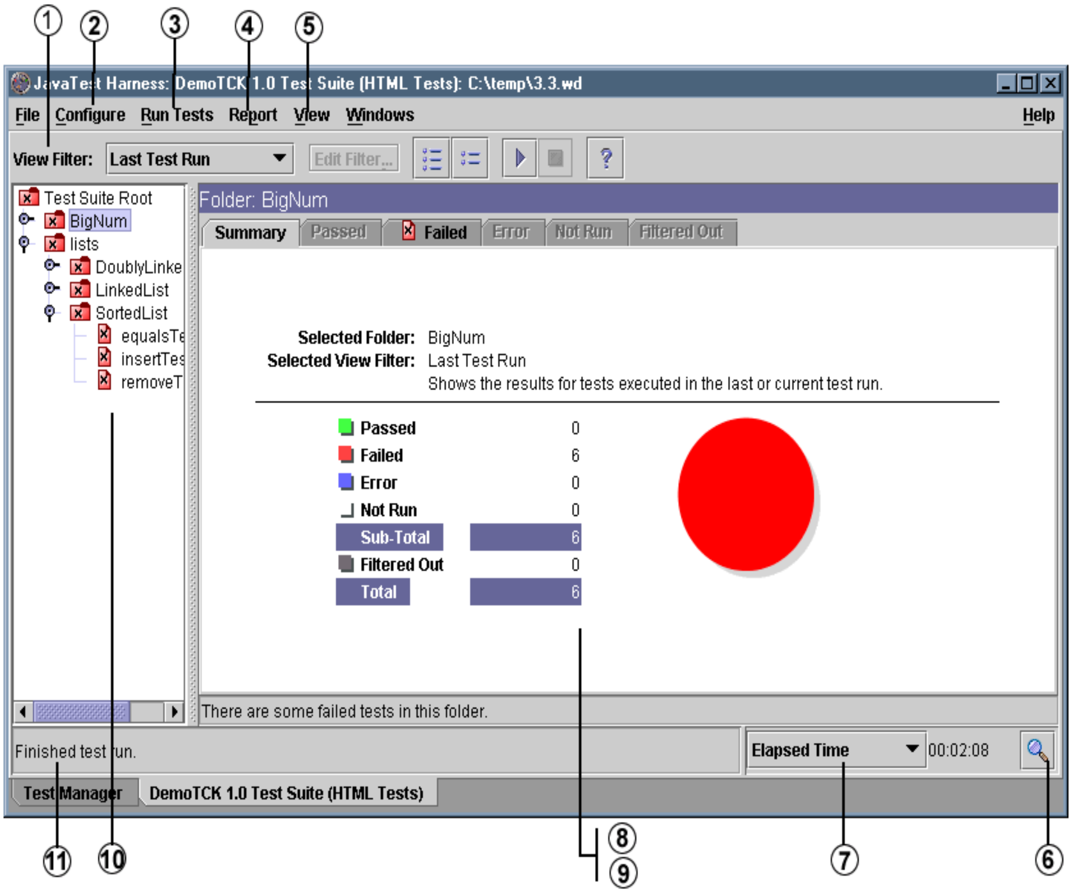
After navigating to a component, you can then use the keyboard to activate the component. The following table lists the key that are used to activate GUI components.

TABLE 14 Keyboard Activation

Activation Keys	GUI Action
Enter or Return	Activates the default command button.
Escape	Dismisses a menu or dialog box without changes.
spacebar	Activates the toolbar button that has keyboard focus.
Shift-spacebar	Extends the selection of items in a list.
Control-spacebar	If the item with focus is in a list, it toggles the selection state of the item without affecting any other list selections. If the item is a link, it follows the link.

Test Manager Window

Use the Test Manager window to load, edit, or create a configuration, to run and monitor tests, to browse test information, and to troubleshoot a test run.



Selected Folder: BigNum
Selected View Filter: Last Test Run
Shows the results for tests executed in the last or current test run.

🟩 Passed	0
🟥 Failed	6
🟦 Error	0
Not Run	0
Sub-Total	6
Filtered Out	0
Total	6

There are some failed tests in this folder.

Finished test run.

The Test Manager window contains the following items:

1. toolbar
2. Configure Menu
3. Run Tests Menu
4. Report Menu
5. View Menu
6. Progress Monitor

7. Progress Meter
8. Folder View (Information Area)
9. Test View (Information Area)
10. Test Tree
11. Status Message

Depending on the window style that you use, desktop menus may also be displayed in the menu bar. See JavaTest Harness Menus for a detailed description of the desktop menus that might also be displayed in a window.

See Appearance Preferences for a description of available window styles.

The following table describes the tasks that you can use the Test Manager to perform.

TABLE 15 Test Manager Tasks

Task	Description
Load, edit, or create a configuration	Provide configuration information required to run your test suite. Configuration as part of running tests is described in Configuring a Test Run.
Run tests	Start test runs by choosing the Run Tests -> Start menu item or click the ▶ button on the toolbar. See Running Tests for a detailed description of how to run tests.
Monitor test runs	Use the test tree with the folder and test views to monitor the status of the test run. For more information about monitoring test runs, see Monitoring a Test Run.
Browse test information	Use the test tree with the folder and test views to browse information about overall test status as well as what occurred during a test run. See Browsing Test Information for details about browsing test run information.
Generate and view test reports	Use the Report menu to generate and view reports of test run information. See Test Reports for details about generating and viewing test run information.
Troubleshoot a test run	Use the test tree with the folder and test views to troubleshoot a test run. See Troubleshooting a Test Run for help with troubleshooting test run problems.

Status Message

The status message is a resizable text area that displays information about current Test Manager activity, such as the state of a test run and the name of the test being run.

Note – If more than one test is running at a time, only the name of the last test started is diplayed in the status message.

Configure Menu

Use the Configure menu to load, create, modify, and view configuration data used for a test run. The following table describes the menu items in the Configure menu.

TABLE 16 Configure Menu

Menu Item	**Description**
New Configuration	Sets the current configuration to an empty configuration and opens the Configuration Editor. The JavaTest harness uses the Configuration Editor to create configuration data containing both test environment and standard values required to run a test suite. See Creating a New Configuration for detailed information.
Load Configuration	Opens the Load Configuration File dialog box. Use the dialog box to load an existing configuration into the Test Manager for use in running the test suite. The JavaTest harness does not require you to use the Configuration Editor when loading an existing configuration interview. See Loading an Existing Configuration for detailed information.
Change Configuration	Opens an additional menu containing Tests to Run, Exclude List (optional), Keywords (optional), Environment (optional), Prior Status, Concurrency, and Timeout Factor menu items that you can use to change specific configuration values. Use the Other Values menu item to change the remaining values in the configuration. The JavaTest harness opens the configuration editor window in the most appropriate mode (Question mode or Quick Set mode) for you to change the configuration value. See Changing Configuration Values for detailed information.

TABLE 16 Configure Menu

Show Checklist (optional)	Enabled only if your test suite supports the configuration checklist. Displays a checklist of tasks to be performed before running tests. See Displaying the Configuration Checklist for detailed information.
Show Exclude List	Opens an Exclude List dialog box that contains the exclude list used to run the test suite. You can use the Exclude List dialog box to review but not edit the contents of the exclude list. If you use multiple exclude lists, the Exclude List dialog box displays the merged set of exclude lists. Use the configuration editor window to add or remove exclude lists. See Using Exclude Lists for detailed information. See Using the Exclude List Browser for detailed information about viewing the contents of the exclude list.
Show Test Environment	Opens a Test Environment browser that displays the configuration values used when running the test suite You can browse but not change values in the Test Environment browser. Use the configuration editor to change the values. See Changing Configuration Values for detailed information. See Using the Test Environment Browser for detailed information about viewing the contents of the test environment.
Show Question Log	Displays a log of the current configuration interview questions and answers. See Displaying the Question Log for detailed information.
Configuration History	Displays a list of configuration files that have been opened. Choose a configuration from the list to use it as the current configuration.

Run Tests Menu

The Run Tests menu is used to start, stop, and monitor a test run. The following table describes the items in te Run Tests menu.

TABLE 17 Run Tests Menu

Menu Item	Description
▶ Start	The JavaTest harness enables the Start menu item when it is not running tests. Choose the Start menu item to start a test run. Only one test run at a time can be active in the Test Manager window. See Starting a Test Run for detailed information about starting a test run.
■ Stop	The JavaTest harness enables the Stop menu item when it is running tests. Choose the Stop menu item to end a test run after the current test is completed. See Stopping a Test Run for detailed information about stopping a test run.
Monitor Progress	When the JavaTest harness is running tests, you can use the Test Manager Progress Monitor to display the progress of the test run and current resource information about the test system. Choose the Monitor Progress menu item to open the Test Manager Progress Monitor. See Using the Progress Monitor for detailed information about using the Test Manager Progress Monitor.

Report Menu

The Report menu contains menu items that create and view reports about test run information. The following table describes the items in the Report menu.

TABLE 18 Report Menu

Menu Item	Description

TABLE 18 Report Menu

Create Report	Opens the Create a New Report dialog box for you to generate reports of test results. See Creating Reports for detailed information.
Open Report	Opens the Report dialog box used to specify a report. See Displaying Reports for detailed information.
Report History	Displays a list of available reports. Choose a report from the list to open a new instance of it in the Report browser. See Displaying Reports for detailed information about the Report browser.

View Menu

The View menu contains menu items that display information about a test run. The following table describes the item in te View menu.

TABLE 19 View Menu

Menu Item	Description
Filters...	Displays the available view filters, the active filter, and allows you to modify the custom view filter. See Using View Filters for detailed description.
Properties	Click the Properties menu item to display the Test Manager Properties dialog box containing the current settings of the Test Manager window. See Viewing Test Manager Properties for detailed information about using the Test Manager Properties dialog box.
Test Suite Errors	The JavaTest harness only enables the Test Suite Errors menu item when it detects that the test suite itself (not the tests in the test suite) contains errors. Click the Test Suite Errors menu item to display a dialog box identifying the errors detected in the test suite. See Viewing Test Suite Errors for detailed information.

Toolbar

The toolbar contains buttons to perform routine tasks that are also available as menu items from the menu bar. The JavaTest harness provides the following tool tips describing each button on the toolbar.

- **View Filter** - Displays the current view filter. Selecting a view filter from the list only filters the status (folder colors and counters) displayed in the Test Manager window, not the tests that are run. To filter the tests that are run, use the configuration editor. See Using View Filters for detailed inforamtion about each filter.
- **Edit Filter** - Enabled only when Custom is selected from the view filter list. See Using View Filters for detailed inforamtion about creating a custom view filter.
- **Question Mode** - Opens the Configuration Editor in Question Mode. See Question Mode for a detailed description of this mode.
- **Quick Set Mode** - Opens the Configuration Editor in Quick Set Mode. See Question Mode for a detailed description of the mode.
- **Run Tests** - Starts a test run. See Starting a Test Run for a detailed description.
- **Stop Test Run** - Stops a test run. See Stopping a Test Run for a detailed description.
- **Window Help** - Displays online help for the Test Manager window.

6

Configuring a Test Run

Before the JavaTest harness can execute the tests in a test suite, it requires information about how your computing environment is configured. You provide the JavaTest harness with this information by loading an existing configuration, creating a new configuration, or changing the values in a configuration.

Note – You can also specify configuration values from the command line when starting the JavaTest harness GUI.

The quantity and scope of information required to run tests depends on the test suite. Some test suites run in diverse environments (different platforms and networks), while others run in very specific, well-defined environments. The test suite may provide a configuration interview, configuration template, or a configuration file (`.jti`) for you to use in creating a configuration for your test run. If your test suite does not provide any of these, consult the test suite documentation for directions about how you can supply the required configuration information.

This chapter contains the following topics:

- **Opening the Configuration Editor Window** - Describes how to open the Configuration Editor window used to create a configuration or change the values in a configuration.
- **Creating a New Configuration** - Describes how to use the Configuration Editor window to create a configuration.
- **Creating a Configuration Template** - Describes how to use the Configuration Editor window to create a configuration template.
- **Saving a Configuration** - Describes how to save the current configuration in a configuration file.
- **Loading a Configuration** - Describes how to load an existing configuration or configuration template for use in running tests.
- **Changing Configuration Values** - Describes how to use the Configuration Editor window to change the values in a configuration or configuration template.
- **Searching the Configuration** - Describes how to search the current configuration for specific characters or values.

- **Working with Multiple Configurations** - Describes how to use multiple configuration files to switch configurations between test runs.
- **Displaying Configuration Information** - Describes how to view information derived from the current configuration.

Configuration Editor Window

The Configuration Editor window provides you with the following modes for displaying and changing configuration values:

- Question Mode
- Quick Set mode

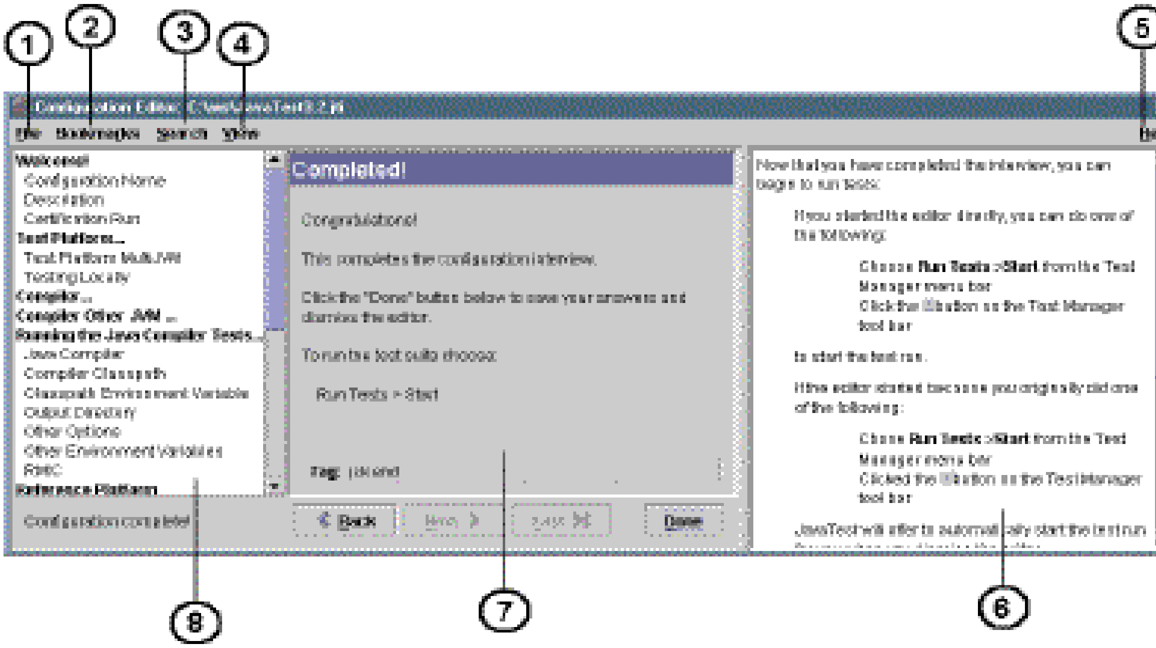
The JavaTest harness opens the Configuration Editor window in the mode best suited to perform the task that you choose from the menu bar. After the Configuration Editor window opens, use the View menu from the menu bar at any time to change modes.

You can also use the Test Manager toolbar to open the Configuration Editor window in a specific mode as follows:

- Click the ≡ button to open the Configuration Editor window in Question Mode. See Question Mode for a detailed description.
- Click the = button to open the Configuration Editor window in Quick Set mode. See Quick Set Mode for a detailed description.

Question Mode

The Question Mode displays the complete configuration, allowing you to create a new configuration, create a configuration template, change the values in a configuration, or search for character and value strings in a configuration.



Congratulations!

This completes the configuration interview.

Click the "Done" button below to save your answers and dismiss the editor.

To run the test suite choose:

Run Tests > Start

In Question Mode, the Configuration Editor window consists of a menu bar with five menus and three panes, as follows:

1. File Menu
2. Bookmarks Menu
3. Search Menu
4. View Menu
5. Help Menu
6. More Info Pane (may be hidden)
7. Question Pane
8. Index Pane

Menus

In Question Mode, the Configuration Editor window contains menus used to load, create, display, and change a configuration.

File Menu

The File menu contains items to open, save, and restore configuration files. The following table describes the items in the Configuration Editor File menu.

TABLE 20 Configuration Editor File Menu

Menu Item	Description
New Configuration	Clears the current configuration and starts a new configuration. See Creating a Configuration for a detailed description. CAUTION : If you accidentally choose New Configuration from the menu bar, do not close the Configuration Editor window or click the Done button. Either action immediately replaces the previous configuration with the new, empty configuration. Instead, choose the previous configuration file name from the Configuration History list displayed in the File menu. The Configuration Editor window then loads the previous configuration in place of the empty configuration. After the configuration is loaded, you can close the Configuration Editor window.
Load	Opens an existing configuration file and makes it the current configuration. See Loading a Configuration for a detailed description.
Save	Saves the current configuration. Choose File **->** Save at any time to save your answers and position in the configuration file. If the configuration is new, the editor opens the file chooser for you to use in naming and saving the current configuration. If you do not provide the `.jti` extension when you name the file, the editor adds the extension when it saves the file.
Save As	Opens a dialog box that you can use to save a configuration with a new name. The Configuration Editor makes the saved configuration the current configuration. If you do not provide the `.jti` extension when you name the configuration file, the editor adds the extension when it saves the file.
Save As Template	Opens the Configuration Template dialog box used to create a template file. See Creating a Configuration Template for a detailed description.
Revert	Discards any changes to the current configuration and restores the last saved version of configuration file.
Configuration History	Displays a list of configuration files that have been opened in the Configuration Editor window. Choose a configuration file from the list to open it in the Configuration Editor window.
Close	Closes the Configuration Editor window.

Bookmarks Menu

The Bookmarks menu contains items to use bookmarks in the configuration. The following table describes the items in the Bookmarks menu.

TABLE 21 Configuration Editor Bookmarks Menu

Menu Item	Description
Enable Bookmarks	Enables and disables bookmarking in the configuration.
Show Only Bookmarked Questions	Display only the marked questions or all questions.
Mark Current Question	Clears the bookmark from a selected question in the Configuration Editor window.
Unmark Current Question	Enabled only if the selected question is not bookmarked. Bookmarks the selected question in the Configuration Editor window.
Clear Answer For Current Question	Enabled only if the selected question is bookmarked. Clears the answer for a selected question in the Configuration Editor window.
Open ... Group	Expands a selected set of questions in the Configuration Editor window.
Clear Answers to Bookmarked Questions	Clear the values in all marked questions.
Remove Bookmarks	Remove all bookmarks from the configuration.

Using bookmarked questions allows the user to display only those configuration questions that must be answered. See Setting Markers for a description of how the Bookmarks menu can be used to specify the questions that the Configuration Editor window displays.

Search Menu

Use the Search menu items to find the occurrence in a configuration of a specific character or value string. When troubleshooting a test run, you can use the Search menu to quickly locate an answer that needs to be changed. The following table describes the items in the Search menu.

TABLE 22 Configuration Editor Search Menu

Menu Item	Description
Find	Opens a dialog box used to search the configuration for a specific character or value string.
Find Next	Searches the configuration for the next occurrence of a specific character or value string.

See Searching the Configuration for a detailed description of how to search for character and value strings in a configuration.

View Menu

Use the View menu to display the Configuration Editor window in Question Mode or in Quick Set mode, to hide or display the More Info pane, and to display question tag field at the bottom of each question panel. The following table describes the items in the View menu.

TABLE 23 Contents of the Search Menu

Menu Item	Description
Question Mode	Displays the Configuration Editor window in Question Mode.
Quick Set Mode	Displays the Configuration Editor window in Quick Set mode.
More Info	Displays and hides the More Info pane in the Configuration Editor window.
Question Tag	Displays and hides the Question Tag field in the Configuration Editor Question Pane.
Refresh	Updates the values and questions displayed in the Configuration Editor window.

Help Menu

Use the Help menu to display the online help for the Configuration Editor window and both editor modes. The following table describes the items in the Help menu.

TABLE 24 Configuration Editor Help Menu

Menu Item	Description
Configuration Editor	Displays online help for the Configuration Editor window.
Question Mode	Displays online help for the Question Mode.
Quick Set Mode	Displays online help for the Quick Set mode.

Index Pane

When completing a configuration, the index pane lists the titles of the questions you have answered, are currently answering, or that the editor determines might need to be answered. The current question is highlighted.

In completed configurations, the questions that are displayed in the index can be controlled by setting bookmarks in the configuration. See Bookmarks Menu.

Note – The title is also displayed at the top of the question pane when you are answering a question.

Click on any question in the index list to make it the current question. Clicking on a question does not cause the list to change. If you change an answer that alters the configuration options, the Configuration Editor window updates the questions in the list to reflect the change in options. If a previously answered question is no longer displayed in the index list, the Configuration Editor saves its answer until you either save the configuration or change its value.

You can also use the buttons at the bottom of the Question pane to navigate through the configuration file. See Question Pane for a description of the navigation buttons.

Question Pane

The Configuration Editor window displays configuration questions in the main text area of the editor. You answer the questions using controls such as text boxes, radio buttons, or combo boxes located beneath the question. After you answer each question, click Next at the bottom of the panel to proceed to the next question.

The buttons at the bottom of the Question pane control the following functions:

- **Back** - Returns to the previous question.
- **Last** - Advances as far as possible through the configuration.
- **Next** - Proceeds to the next question.
- **Done** - Saves your answers as a configuration and closes the Configuration Editor window.

See Creating a New Configuration for information about using the Question pane to create a configuration.

See Changing Configuration Values for information about using the Question pane to edit the current configuration.

More Info Pane

To open and close the More Info pane, choose View **->** More Info from the menu bar.

The More Info pane provides additional information about each question, including the following:

- Background information about the question
- Information about choosing an answer
- Examples of answers

See Keyboard Access for a description of how the keyboard can be used to navigate the More Info pane.

Quick Set Mode

Quick Set mode uses tabbed panes to display and change the standard values in the configuration. The standard values are the runtime values of a configuration. These values specify how tests are run and choose the tests that are run and can change from test run to test run.

Depending on your test suite, Quick Set mode can display four, five, or six tabbed panes and a menu bar.

If you choose to only generate a text report in an existing directory containing HTML reports, you must use an external text editor or web browser to view the `summary.txt` file. To use the Report Browser when viewing text reports, generate the text only report either in a new report directory or in an existing report directory that only contains plain text reports.

- Tests
- Exclude List
- Keywords (optional)
- Prior Status
- Environment (optional)
- Execution

To use Quick Set mode, a configuration must be loaded in the Configuration Editor. See Changing Configuration Values for a description of how to change the runtime values of the current configuration.

Creating a Configuration

If your test group or test suite does not provide an existing configuration or a configuration template, use the Configuration Editor window to create a configuration for the test run.

If your test suite only provides an environment file (`.jte`), the JavaTest harness uses the Configuration Editor window to include the environment file in the new configuration.

1. Choose Configure -> New Configuration from the Test Manager menu bar or click the ☰ button on the toolbar and choosing File -> New Configuration from the Configuration Editor menu bar.

Note – If you accidentally choose New Configuration, do not close the Configuration Editor window or click the Done button. Either action immediately replaces the previous configuration with the new, empty configuration. Instead, choose the previous configuration from the Configuration History list to reload the previous configuration.

---

2. Answer the questions displayed in the Configuration Editor window.

   The Configuration Editor window displays questions in the center pane (the Question Pane). Use the text box, radio button, or combo box controls located beneath the question to provide the required configuration information. After you answer each question, click the Next button to proceed to the next question.

Note – Some questions provide information and do not require an answer. In these cases, click the Next button to proceed to the next question.

You can go backward and forward to any question to review or change your answer by doing one of the following:

- Choosing a question directly from the Index pane
- Clicking the Back button, the Next button, or the Last button

  As you move backward and forward, the Configuration Editor window saves all answers until you either save the configuration or change the answers.

  Choose File **->** Save at any time to save your answers and position in a configuration file. See Saving a Configuration.

3. After you complete the configuration, click the Done button to save the configuration and close the Configuration Editor window.

Creating a Configuration Template

The JavaTest harness provides users and site administrators with the ability to create configuration templates that can be used to run tests on multiple test systems or test platforms with minimal changes. Users can save any configuration as a template (including configurations with unanswered questions).

See Create a Configuration Template from an Existing Configuration or Create a Configuration Template as a New Configuration.

For example, if a test group uses a central location to provide and manage the resources required to run tests (the test suite, report directories, configuration files, or the JavaTest harness), a site administrator or user can create a configuration template that contains all known configuration values required by the test group to run tests. Each user can load the configuration template from the central site, provide only those values unique to their test environment or test run, and run their tests using their completed configuration.

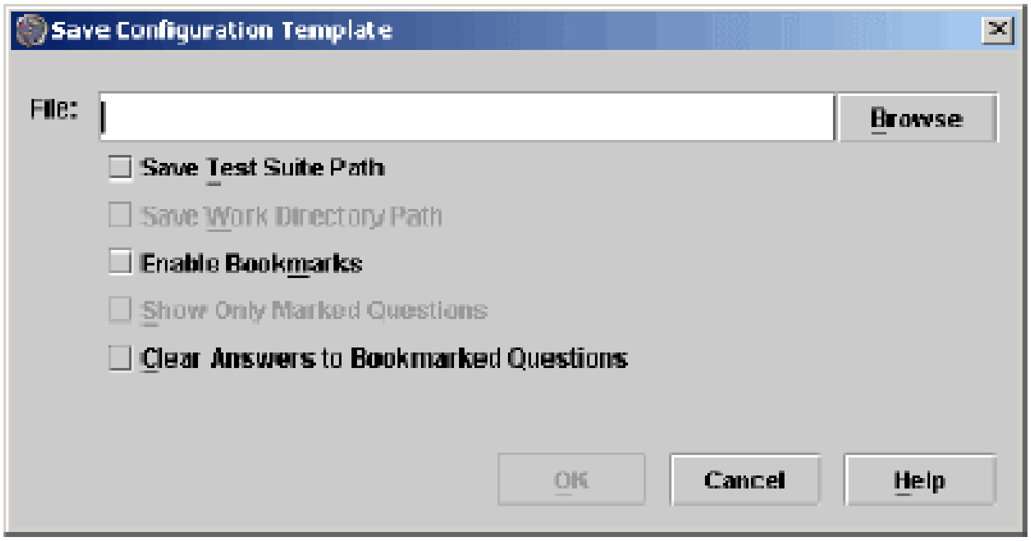
Note – Use Bookmarks to simplify a configuration template by displaying only the questions that require information from the user. Normally long interviews can often be reduced to a small set of required questions through the effective use of Bookmarks within a configuration template. See Setting Bookmarks in a Configuration.

Configuration templates are created by choosing File and Save As Template from the Configuration Editor menubar. The Configuration Editor opens the Save Configuration Template dialog box.

See Save Configuration Template for detailed information about the dialog box.

Save Configuration Template

The Save Configuration Template dialog box allows site administrators or users to creat configuration templates.


Save Configuration Template
File:
Browse
☐ Save Test Suite Path
☐ Save Work Directory Path
☐ Enable Bookmarks
☐ Show Only Marked Questions
☐ Clear Answers to Bookmarked Questions
OK
Cancel
Help

The Save Configuration Template dialog box enables site administrators or users to set the following options when creating a template:

- **File** - A text field that specifies the location and name of the saved template. Click the Browse button to open a file chooser that enables navigation to a template directory. Templates are saved using the same `.jti` extension as a standard configuration file.
- **Save Test Suite Path** - If selected, the path to the test suite remains associated with the configuration template. Each time the template is loaded, the JavaTest harness also loads the associated test suite. If Save Test Suite Path is not selected, the template is not associated with a specific test suite. Each time the template is loaded, the user must specify a test suite.
- **Save Work Directory Path** - This option is only enabled when the Save Test Suite Path option is selected. If selected, the work directory remains associated with the configuration template. Each time the template is loaded, the JavaTest harness also loads the contents of the work directory unless the work directory is already open or specified. If Save Work Directory Path is not selected, the template is not associated with a work directory. Each time the template is loaded, the user must specify a work directory before running tests.

- **Enable Bookmarks** - Bookmarks allow site administrators or users to mark questions that require additional information from the user or that should receive attention. If selected, all bookmarks are saved with the template. If Enable Bookmarks is not selected, any bookmarks set in the configuration are not saved with the template.
- **Show Only Marked Questions** - This option is only available when the Enable Bookmarks option is selected. Bookmarks allow site administrators or users to simplify the appearance of a template by displaying only those questions that require additional information. If selected, each time the template is loaded the configuration interview only displays questions that are bookmarked. Users can display all questions in the configuration interview by changing the settings in the Configuration Editor. This option is useful in reducing lengthy interviews to a small number of questions and values.

  If Show Only Marked Questions is not selected, all questions are displayed in the Configuration Editor.
- **Clear Answers to Bookmarked Questions** - If selected, the answers to all bookmarked questions are cleared when the template is saved. This allows site administrators or users to require that users provide answers for bookmarked questions before running tests. If Clear Answers to Bookmarked Questions is not selected, any answers in bookmarked questions are saved with the template.

  The cleared state may appear blank, unanswered, or may also be the default answer provided by the test suite.

▼ Create a Configuration Template from an Existing Configuration

Users can create a configuration template from an existing configuration by performing the following steps:

1. Click the ☰ button on the toolbar or choose Configure **->** Change Configuration **->** Other Values from the Test Manager menu bar.

2. If the configuration is not displayed in the Configuration Editor window, choose File **->** Load from the menu bar to load the configuration to be used in creating the configuration template.

3. The Configuration Editor window displays questions in the center pane (the Question Pane). Click the Next button to proceed to the next question in the configuration. Use the text box, radio button, or combo box controls located beneath the question to change any configuration information required for use as template.

4. If a user is only expected to answer or change the values of certain questions, use the Bookmarks menu to mark each question that a user is expected to answer. See Setting Bookmarks in a Configuration.

5. After all questions have been answered and marked, choose Bookmarks **->** Show Only Marked Questions from the menu bar to display only the questions that have bookmarks.

Examine the result and verify that the bookmarked questions are the minimum set of questions a user must answer. This is what the user sees when the template is loaded in the Configuration Editor.

1. After all changes have been made to the configuration values, choose File **->** Save as Template from the menu bar.
2. In the Save Configuration Template dialog box choose the options for the configuration template and click the Save button. See Save Configuration Template for detailed information about the available template options.

Create a Configuration Template as a New Configuration

Users can create a configuration template from a new configuration by performing the following steps:

1. Open an empty Configuration Editor window by choosing Configure **->** New Configuration from the Test Manager menu bar. You can also open an empty Configuration Editor window by clicking the ☰ button on the toolbar and choosing File **->** New Configuration from the menu bar.
2. Create the new configuration file by answering the questions displayed in the Configuration Editor window.
   The Configuration Editor window displays questions in the center pane (the Question Pane). Use the text box, radio button, or combo box controls located beneath the question to provide the required configuration information. After you answer a question, click the Next button to proceed to the next question.
3. If a user is only expected to answer or change the values of certain questions, use the Bookmarks menu to mark each question that a user is expected to answer. See Setting Bookmarks in a Configuration.
4. After all questions have been answered and marked, choose Bookmarks **->** Show Only Marked Questions from the menu bar to display only the questions that have bookmarks.

Examine the result and verify that the bookmarked questions are the minimum set of questions a user must answer. This is what the user sees when the template is loaded in the Configuration Editor.

1. Choose File **->** Save as Template from the menu bar.
2. In the Save Configuration Template dialog box choose the options for the configuration template and click the Save button. See Save Configuration Template for detailed information about the available template options.

3. After the template file is created, use the operating system to mark it read-only so that users cannot accidentally overwrite it.

Note – When a user loads a configuration template, the Configuration Editor does not save it as a different file. If it is loaded from a central location the user must either copy the `.jti` file before loading it or save it to a new location immediately after it is loaded.

Saving a Configuration

To save your current answers and position in the configuration file, perform one of the following actions:

- If you are creating a configuration template, choose File -> Save As Template from the Configuration Editor menu bar. See Creating a Configuration Template for detailed information about creating configuration templates.
- If you are using a configuration template or an existing configuration file to create a new configuation for use in running tests, choose File -> Save As from the Configuration Editor menu bar. A dialog box opens for you to use in setting the location and name of the new configuration file.
- If you are modifying an existing configuration file, choose File -> Save from the Configuration Editor menu bar. The Configuration Editor saves the changes in the configuration file.

At completion of the configuration interview, click the Done button at the bottom of the Configuration Editor window to save the current configuration in a `.jti` file and close the window.

Note – If you use different configurations to run a test suite, choose File -> Save As to save each configuration by using a different file name. See Working with Multiple Configurations for more information.

Loading a Configuration

If your test group or test suite provides a configuration template or an existing configuration, you can use it to run tests. Perform the following steps to load a configuration template or an existing configuration:

1. Choose Configure -> Load Configuration from the Test Manager menu bar, or, if you have an open Configuration Editor window, choose File -> Load from the menu bar.

The JavaTest harness opens the Load Configuration File chooser.

2. Use the file chooser to locate the configuration file (`.jti`), select it in the file chooser, and click the Load File button.

The JavaTest harness loads the configuration file (`.jti`) and closes the Load Configuration File chooser. If the configuration file is a template or if its values must be modified to run tests, use the Configuration Editor window to set values that are appropriate for your test environment. See Changing Configuration Values.

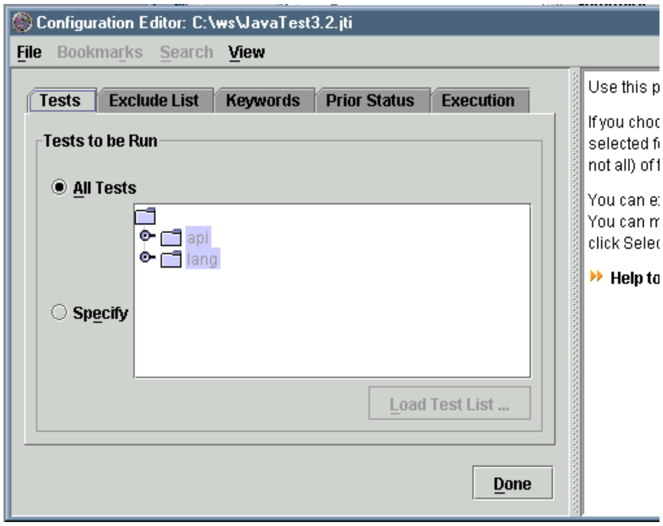
Changing Configuration Values

The following topics describe how values in the configuration can be changed for a test run.

- **Specifying Tests to Run** - Describes how to specify the individual tests or groups of tests in a test run.
- **Using Exclude Lists** - Describes how to specify the exclude list used for a test run.
- **Using Keywords** - Describes how to filter the tests that are run by specifying keywords.
- **Using Prior Status** - Describes how to filter the tests that are run based on their prior results.
- **Specifying a Test Environment** - Describes how to specify the test environment file (only enabled if required by the test suite) used to run tests.
- **Setting Concurrency and Timeout Factor** - Describes how to set the concurrency and timeout values of a test run.
- **Changing Other Values** - Describes how to change additional values in a configuration.

Specifying Tests to Run

You can use either the Tests tab in Quick Set Mode or the Specify Tests to Run question in Question Mode to specify which tests in a test suite are run.



In Quick Set Mode , choose Specify to enable the Configuration Editor test tree and the Load Test List button. In Question Mode, use the How to Specify Tests question to specify whether to use the test tree or a load list.

See the following topics for detailed information about using the test tree or load list to specify tests :

- Using the Test Tree to Specify Tests
- Using a Load Test List to Specify Tests

Using the Test Tree to Specify Tests

In the Configuration Editor test tree (not the Test Manager test tree) you can choose individual tests and folders of tests for the JavaTest harness to run. The JavaTest harness walks the test tree starting with the sub-branches or tests you specify and executes all tests not filtered out by the exclude list, keyword, or prior status.

Note – Restrictions are applied cummulatively. For example, you can specify the tests in a test suite, then restrict the set of tests using an exclude list, and then further restrict the set to only those tests that passed on a prior run.

If you choose a test folder, the JavaTest harness selects all tests in the test suite under that location for the test run.

If you choose one or more tests, the JavaTest harness selects those individual tests for the test run.

The Tests pane highlights all folder and test icons selected for the test run. You can make individual selections in the test tree by pressing the Control key when you click an icon or name in the test tree.

To select a series or sequence of tests or folders, press the Shift key and then click the first and the last icon or name in the sequence.

When you select some (but not all) of the tests in and under a folder, the Test pane partially highlights the folder icon.

Using a Load Test List to Specify Tests

Click the Load Test List to open a dialog box for locating and selecting a `.txt` file containing a list of tests to run. See Creating a Test List File for a detailed description of the test list file.

When the file is loaded, the test tree is updated to indicate the tests that have been specified.

Creating a Test List File

The format requirements of the test list file are as follows:

- Line-oriented
- Relative test paths
- Blank lines and lines beginning with the pound symbol (#) are ignored
- On all other lines, the first item (up to the first whitespace) is taken as a test name and the remainder of the line is ignored

The format of the load list file is compatible with the format of the `summary.txt` generated as part of a report.

The following sequence of steps describe one way in which a test list file might be created.

1. Select the lines from `summary.txt` that define the tests you want to run. On a Solaris platform you might use `awk`, `grep`, or similar utilities to identify lines that specify the tests with failed or error results.

Example lines from `summary.txt`:

```
BigNum/compareTest.html Failed. exit code 1
 BigNum/equalsTest.html Failed. exit code 1
 BigNum/longConstrTest.html Failed. exit code 1
 BigNum/subtractTest.html Failed. exit code 1
 lists/DoublyLinkedList/appendTest.html Failed. exit code 1
 lists/DoublyLinkedList/equalsTest.html Failed. exit code 1
 lists/DoublyLinkedList/insertTest.html Failed. exit code 1
 lists/DoublyLinkedList/removeTest.html Failed. exit code 1
 lists/LinkedList/appendTest.html Failed. exit code 1
 lists/LinkedList/equalsTest.html Failed. exit code 1
 lists/LinkedList/insertTest.html Failed. exit code 1
 lists/LinkedList/removeTest.html Failed. exit code 1
 lists/SortedList/equalsTest.html Failed. exit code 1
 lists/SortedList/insertTest.html Failed. exit code 1
 lists/SortedList/removeTest.html Failed. exit code 1
```

1. Copy and paste the relative paths into a `.txt` file.

2. Remove all text from the path names that is not part of the test path.

Example lines in the `.txt` file:

```
BigNum/compareTest.html
 BigNum/equalsTest.html
 BigNum/longConstrTest.html
 BigNum/subtractTest.html
 lists/DoublyLinkedList/appendTest.html
 lists/DoublyLinkedList/equalsTest.html
 lists/DoublyLinkedList/insertTest.html
 lists/DoublyLinkedList/removeTest.html
 lists/LinkedList/appendTest.html
 lists/LinkedList/equalsTest.html
 lists/LinkedList/insertTest.html
 lists/LinkedList/removeTest.html
 lists/SortedList/equalsTest.html
 lists/SortedList/insertTest.html
 lists/SortedList/removeTest.html
```

1. Save the `.txt` file using a descriptive name.

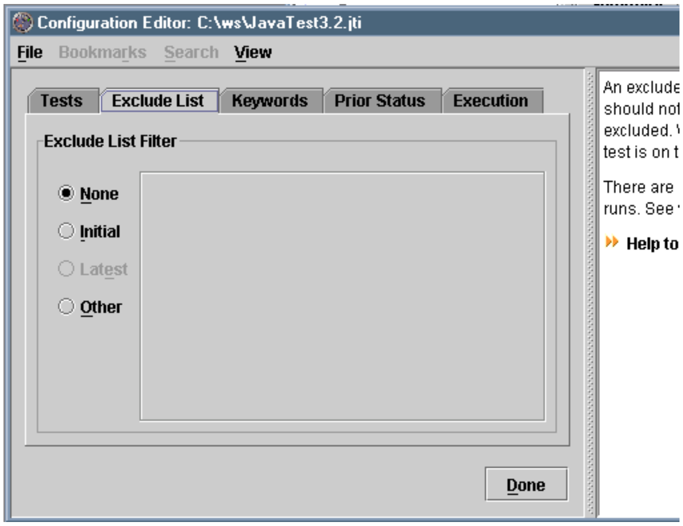
Example `.txt` file name:

```
BigNum_ListsfailedTests.txt
```

Using Exclude Lists

Exclude list files contain a list of tests in a test suite that are not run by the JavaTest harness. You can specify specify one or more exclude list files in one of two ways, as follows:

- Choose Configure -> Change Configuration -> Exclude List from the Test Manager menu bar.
- Click the ≡ button in the Test Manager toolbar to open the Configuration Editor window in Quick Set mode and click the Exclude List tab (the optional More Info pane is not fully shown in the following illustration).



Note – You can also use Question Mode view to specify exclude lists.

Use the following to specify the exclude list option used to run tests:

- **None** - an Exclude List is not used.
- **Initial** - only enabled if the test suite provides an exclude list. If you choose Initial, the tests are run using the exclude list provided by the test suite.
- **Latest** - only enabled if the test suite provides a location for updated exclude lists. If you choose Latest, additional options are displayed. See Latest Exclude List below for detailed information.
- **Other** - a custom exclude list can be used. See Other Exclude List below for detailed information.

If a complete test is added to the exclude list, the JavaTest harness updates the test result status without requiring the test be rerun. However, if only a test case from a test is added to the exclude list, the JavaTest harness requires that you rerun the test using the updated exclude list before updating the test result status.

Latest Exclude List

If your test suite provides a URL for the latest test suite, the JavaTest harness enables the Latest exclude list option. The following table describes the text and controls displayed when you choose the Latest exclude list option.

TABLE 25 Latest Exclude List Dialog Box Contents

Text and Controls	Description
Location:	Displays the location of the exclude list specified by the test suite. This is a non-editable field.
Last updated:	Displays the date that the exclude list was last updated. This is a non-editable field.
Check For Updates Automatically	Causes the JavaTest harness to automatically check the location of the exclude list and compare the date-time stamps of the remote and local exclude lists. The JavaTest harness then displays a dialog box advising you of the results. If a new exclude list is available, you can choose to download it.
Every _ Days	Sets an interval for the JavaTest harness to automatically check the remote location of the exclude list for updates.
Every Test Run	Causes the JavaTest harness to check the remote location of the exclude list for updates before each test run.
Check Now	Causes the JavaTest harness to check the remote location of the exclude list for an update.

Other Exclude List

The following table describes the buttons displayed when you choose the Other exclude list option.

TABLE 26 Other Exclude List Dialog Box Contents

Button	Description
Add	Selects an exclude list file for your test suite. As you make selections with the file chooser dialog box, they are added to the list. After you add an exclude list, you can modify the list.
Remove	Clears an item from the list. Select an item in the list and click Remove.
Move Up	Moves an item one position higher in the list. Select an item in the list and click Move Up.
Move Down	Moves an item one position lower in the list. Select an item in the list and click Move Down.

Using Keywords

Note – You can also use the Configuration Editor window in Question Mode to specify test run restrictions based on the keywords.

To specify the keywords and how they are used to restrict the tests in a test run, click Select tests that match. The JavaTest harness enables the Expression, Insert Operator, and Insert Keyword buttons.

Build an expression in the text field by using any logical combination of the following actions:

- Click the Expression button to display a list of expressions that can be constructed. From the list, choose the type of expression that you are building.
- Click the Insert Keyword button to display the list of keywords provided by the test suite for use in filtering tests (this is only available if the test suite has information). From the list, choose the keywords used in the expression.
- Click the Insert Operator button to display a list of operators that you can use to construct boolean expressions in the text field. From the list, choose the operator used in the expression.

The following table provides descriptions and examples of keyword expressions that can be constructed.

TABLE 27 Keyword Expressions

Expression	Description

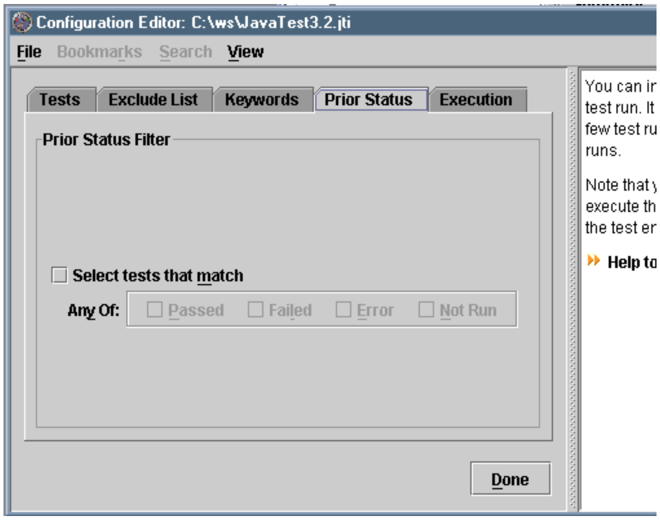
TABLE 27 Keyword Expressions

Any Of	Runs all tests in the test suite having any of the keywords entered in the text field. Example: A test suite uses the keyword `interactive` to identify tests that require human interaction, and `color` to identify tests that require a color display. To execute only the tests containing the `interactive` keyword, choose Any Of and then use the Insert Keyword button to choose the `interactive` keyword.
All Of	Runs all tests in the test suite having all of the keywords entered in the text field. Example: To execute only the tests containing both the `interactive` and `color` keywords, choose All Of and then use the Insert Keyword button to choose the `interactive` and `color` keyword.
Expression	Runs all tests in the test suite having the expression entered in the text field. Use the Insert Keyword and the Insert Operator buttons to construct a Boolean expression in the text field. Keywords stand as Boolean predicates that are true if, and only if, the keyword is present in the test being considered. A test is accepted if the overall value of the expression is true. All other tests are rejected by the restriction. Example: A test suite uses the keyword `interactive` to identify tests that require human interaction, and `color` to identify tests that require a color display. To execute only the tests with the `color` keyword that do not also contain the `interactive` keyword, choose Expression and then use the Insert Keyword button to choose the `color` keyword, the Insert Operator button to choose the ! operator, and the Insert Keyword button to choose the `interactive` keyword.

Using Prior Status

You can use prior status to restrict the set of tests to be run in one of two ways, as follows:

- Choose Configure -> Change Configuration -> Prior Status from the Test Manager menu bar.
- Click the ☰ button in the Test Manager toolbar to open the Configuration Editor window in Quick Set mode and click the Prior Status tab.



Note – You can also use the Configuration Editor window in Question Mode to specify test run restrictions based on results from a prior test run.

By choosing Select tests that match, you can run tests with restrictions based on their result from a prior test run. The following table describes the available Prior Status filter selections.

TABLE 28 Using Prior Status as a Filter

Prior Status	Action
Passed	Selects tests that passed the last time the test was executed.
Failed	Selects tests that failed the last time the test was executed.
Error	Selects tests that the JavaTest harness could not execute the last time it was included in a test run.
Not Run	Selects tests without results in the current work directory.

Prior status is evaluated on a test-by-test basis using information stored in the result files (`.jtr`) that are written in the work directory. Unless overridden by a test suite, a result file is written in the work directory for every test that is executed. When you change work directories between test runs, the result files in the previous work directory are no longer used and, if the new work directory is empty, the JavaTest harness behaves as though each test in the test suite was not run.

You can also use the Prior Status setting in combination with the Current Configuration view filter to display only those test and folder status icons that match the specified prior status. The test tree displays all other tests and folders as gray, filtered out 📁folder and 📄test icons. During a test run, when a test result no longer matches the prior status filter, the test tree changes the test and folder icons to gray, filtered out 📁folder and 📄test icons.

For example, if you only want to monitor tests in a test suite that had failed results you would set the Prior Status filter to Any Of Failed and repeat the test run. As tests pass, the test tree changes their icons from failed to filtered out, indicating that they no longer match the Prior Status filter Any Of Failed.

Note – It is often useful to choose all of the status values except Passed for the first few test runs, then refine the filtering to reduce the number of tests in subsequent runs.

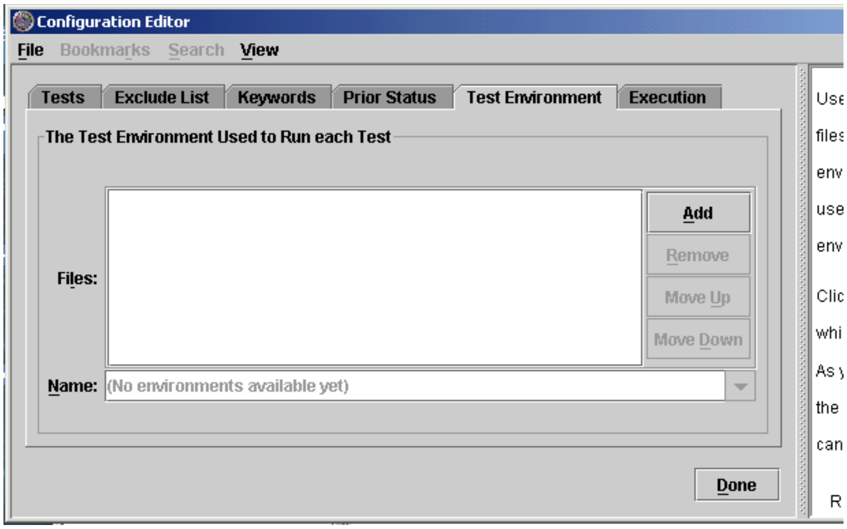
Note –

Specifying a Test Environment

The JavaTest harness displays this tab only if your test suite uses test environment (`.jte`) files to run tests.

You can list the environment files used by your test suite and include them in the configuration in one of two ways, as follows:

- Choose Configure -> Change Configuration -> Test Environment from the Test Manager menu bar.
- Click the ≡ button on the Test Manager toolbar to open the Configuration Editor window in Quick Set mode and click the Test Environment tab.



Note – You can also use the Configuration Editor window in Question Mode to specify the test environment.

The Test Environment pane contains following areas:

- [Files Area](#files-area)
- [Name Area](#name-area)

Files Area

Use the buttons described in the following table to create or modify a list of environment files used to run tests in your computing environment.

TABLE 29 Files Area Buttons

Button	Description
Add	Adds an environment file to the list. To select an environment file (`.jte`) for your test suite, click Add. As you make selections with the file chooser dialog box, they are added to the list. After you add an environment file, you can modify the list.
Remove	Clears an environment file from the list. Select it in the list and click Remove.
Move Up	Moves an environment file one position higher in the list. Select it in the list and click Move Up.
Move Down	Moves an environment file one position lower in the list. Select it in the list and click Move Down.

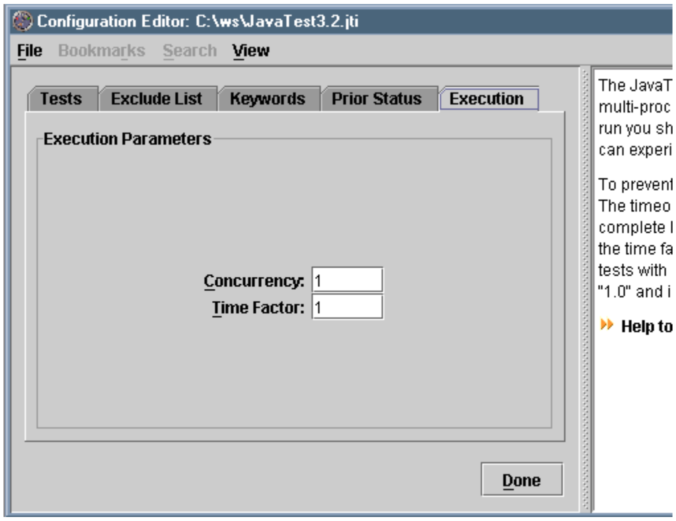
Name Area

Click the drop-down arrow on the text field to see the list of test environments in the environment files listed in the Env Files list. Click the Show button to view the contents of the selected test environment.

Setting Concurrency and Timeout Factor

You can set the concurrency and the timeout factor in one of three ways, as follows:

- Choose Configure -> Change Configuration -> Concurrency from the Test Manager menu bar.
- Choose Configure -> Change Configuration -> Timeout Factor from the test Manager menu bar.
- Click the ☰ button on the Test Manager toolbar to open the Configuration Editor window in Quick Set mode and click the Execution tab.



Concurrency: 1
Time Factor: 1

Note – You can also use the Configuration Editor window in Question Mode to specify the concurrency and the timeout factor.

Use the Execution pane to set the following:

- Concurrency
- Time Factor

Concurrency

The JavaTest harness can run tests concurrently. If you are running the tests on a multiprocessor computer or are using multiple agents on a test system, concurrency can reduce the time required to run tests. For detailed information about using agents to run tests, refer to your test suite documentation and to the *JavaTest Harness Agent User's Guide* if it is proviided by the test suite.

When using multiple agents to run tests, the concurrency value must not exceed the number of agents. If the concurrency value exceeds the total number of available agents, an error will occur in the test run.

If you have unexpected test failures, run the tests again, one at a time. Some test suites may not work correctly if you run tests concurrently. The default range of values used by the JavaTest harness is from `1` to `50`.

For your first test run, leave this field set to `1`. After the tests run properly, you can increase this value. Unless your test suite restricts concurrency, the maximum *number* of threads specified by the `concurrency` command is `50`. See your test suite documentation for additional information about using concurrency values greater than `1`.

This field is disabled for some test suites.

Time Factor

To prevent a stalled test from stopping a test run, most test suites set a timeout limit for each test. The timeout limit is the amount of time that the JavaTest harness waits for a test to complete before moving on to the next test.

If you are running the tests on a particularly slow CPU or slow network, you can change the time limit by specifying a floating point value in the time factor field. Each test's timeout limit is multiplied by the time factor value. The default range of values used by the JavaTest harness is from `0.1` to `100.0`.

Note – In the Time Factor field, the JavaTest harness uses the form of floating point values that is specific to the locale in which it is run. For example, if your locale uses floating point values in the form of `x,x`, JavaTest harness uses that form of floating point value in the Time Factor field. In setting the timeout factor in the following example, specify values of `2,0` and `0,5` if your locale uses floating point values in the form of `x,x`.

Example:

If you specify a value of `2.0`, the timeout limit for tests with a basic 10-minute time limit becomes 20 minutes. Specifying a value of `0.5` for tests with a 10-minute limit produces a 5-minute timeout limit.

At first, use the default value of `1.0` to run tests and then, if necessary, increase the value. The actual timeout calculation for any particular test suite might vary.

Using Bookmarks in Configurations

In Question Mode, you can use Bookmarks to restrict the set of questions displayed in the Configuration Editor window (Bookmarks are not used in Quick Set Mode). Bookmarks are persistent and are saved in the `.jti` file, and reloaded when the configuration is used again. When bookmarks are disabled, they are preserved in program memory but cannot be manipulated.

Set Bookmarks for Specific Questions

To set bookmarks for specific questions in the configuration, perform the following steps:

1. Click the Bookmarks **->** Enable Bookmarks check box in the menu bar.
2. Click a question in the Configuration Editor.
3. Set a bookark for the highlighted question by performing one of the following actions:

- Click to the left of the highlighted question text to set a bookmark for the question.
- Choose Bookmarks **->** Mark Current Question in the menu bar.
- Right click the highlighted question and choose the Mark Current Question menu item in the pop-up menu.

Display Only Questions With Bookmarks

To display only the questions with bookmarks, click the Bookmarks **->** Show Only Bookmarked Questions checkbox in the menu bar.

The list of configuration questions displays only those questions with bookmarks.

The set of questions displayed in the list is changed as follows:

- The first question in the interview is displayed.
- Questions with bookmarks are displayed.
- If the interview is complete (all questions have valid answers) the final question is displayed.
- If the interview is incomplete (one or more questions have invalid or incomplete answers), the questions from the last marked question to the first question with an invalid or incomplete answer are displayed.

- Sequences of questions not described by a previous category are grouped and represented by three dashes (---) in the list. These groups can be opened to display the complete sequence of questions.

Open Groups of Questions

To open groups of hidden questions in the configuration without bookmarks, which are represented by three dashes (---) in the list, perform the following steps:

1. Click the --- section in the question list.
2. Click the Bookmarks **->** Open --- Group menu item from the menu bar or right click the highlighted --- section and choose Open --- Group menu item from the pop-up menu.

Close Groups of Questions

To close groups of questions in the configuration without bookmarks, perform the following steps:

1. Click the question in the question list.
2. Click the Bookmarks **->** Close --- Group menu item from the menu bar or right click the highlighted --- section and choose Close --- Group menu item from the pop-up menu.

Clear the Values of All Questions With Bookmarks

To clear the values of all questions with bookmarks, perform the following steps:

1. Click a question in the question list.
2. Click the Bookmarks **->** Clear Answers to Bookmarked Questions menu item from the menu bar.

Clear the Value of a Specific Question

To clear the value of a specific question, perform the following steps:

1. Click the question in the question list.
2. Click the Bookmarks **->** Clear Answer For Current Question menu item or or right click the highlighted question and choose Clear Answer For Current Question from the pop-up menu.

Remove All Bookmarks

To remove the bookmarks from the configuration, choose Bookmarks **->** Remove Bookmarks. The configuration list reverts to the full configuration list.

Remove a Bookmark From a Question

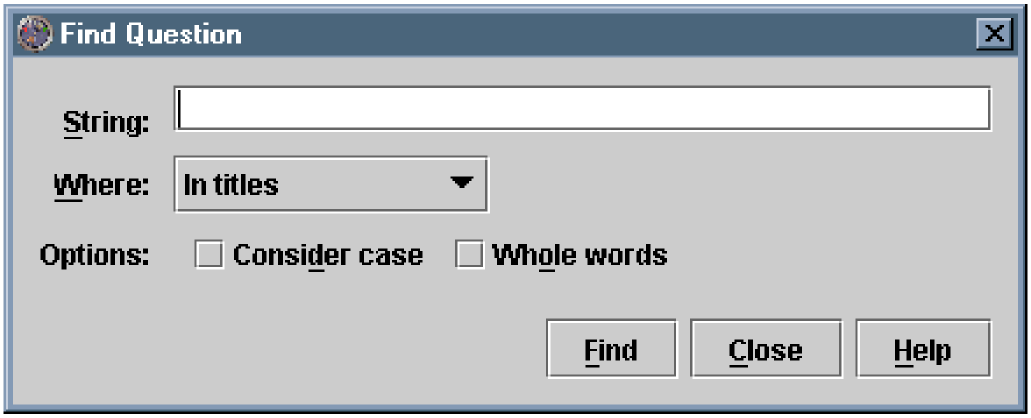
To remove the bookmarks from a question in the configuration, perform the following steps:

1. Click the question in the question list.
2. Click the Bookmarks **->** Unmark Current Question menu item or or right click the highlighted question and choose Unmark Current Question menu item from the pop-up menu.

Searching the Configuration

When using the Configuration Editor in Question Mode, you can locate and display the panes containing a specific character string by choosing Search -> Find. The Find Question dialog box is used in locating and displaying a specified character string.

The JavaTest harness can search titles, questions, and answers for matching characters. It does not search the More Info.



The following table decribes the search criteria used in the Find Question dialog box.

TABLE 30 Find Question Dialog Box Search Criteria

Item	Description
String	Enter the character string that you are trying to find.
Where	Choose where you want to search: • In titles • In text of questions • In answers • Anywhere
Options: Consider case	Specifies that the search pattern match the case of the characters in the Find text field.
Options: Whole words	Specifies that the search pattern only match whole words from the Find text field.

Click the Find button to search for the character string.

To repeat the search, either click the Find button again or use the Search **->** Find Next menu item from the Configuration Editor menu bar.

Click the Help button to display context-sensitive help.

Working with Multiple Configurations

In some testing situations it is useful to use separate configuration files to switch between different configurations for different test runs. For example, if you routinely test your product under different versions of the JVM amchine, you could specify a different Java launcher command in each configuration.

Prior to running tests, use the Configure menu to load the required configuration file. See Loading an Existing Configuration for detailed information. These configuration files can be loaded from a central resource provided by your test group. See your test group for the name and location of the configuration files.

If your group does not provide an existing set of configuration files, you can create them by using the Configuration Editor to edit an existing configuration and then save each variation to a file name of your choosing. See Editing the Current Configuration for detailed information.

You can save these configuration files anywhere in your file system. Generally, however, the work directory should not be used to save configuration files. Clearing the work directory would delete the configuration file.

After these configuration files have been created, they can be used by any JavaTest harness in your test group to run tests.

Displaying Configuration Information

The JavaTest harness provides special browsers for displaying configuration checklists, test environment values, exclude list contents, and configuration log information derived from the current configuration.This section contains the following topics:

- **Displaying the Configuration Checklist** - Describes how to display the checklist that a test suite's interview may produce. The checklist is optional and may not be produced by all test suites.
- **Using the Test Environment Browser** - Describes how to view the configuration values used to run a test suite. During troubleshooting you can use the Test Environment browser to view the configuration values and their sources that were derived from the configuration file and used by test suite specific plugin code to execute and run tests.
- **Using the Exclude List Browser** - Describes how to view the list of tests excluded from a test run. You can use the Exclude List browser to display the list of tests that were excluded from a test run. The browser also displays details about individual tests selected in the list.
- **Displaying the Question Log** - Describes how to browse (in HTML format) the text of all the completed questions asked in the configuration interview and their answers.

Displaying the Configuration Checklist

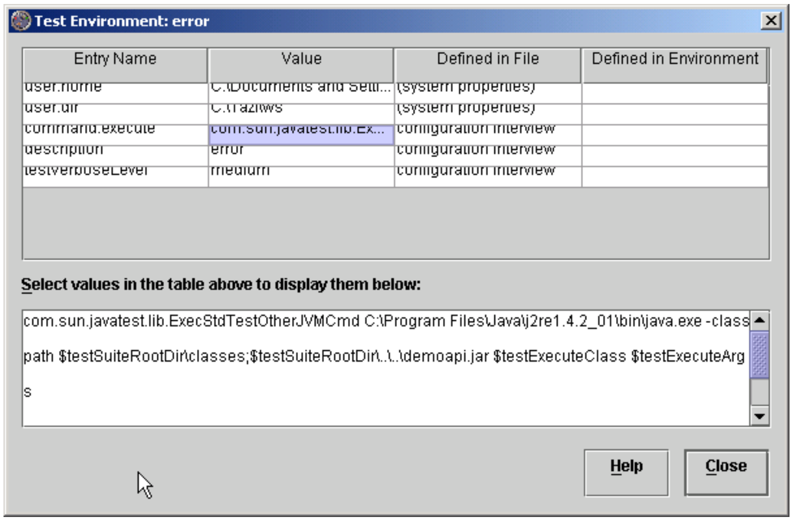
Your test suite might produce a checklist of steps that must be performed before running tests. The checklist is dynamically generated by the JavaTest harness from the current configuration. Changes to the values in the current configuration can produce different checklist items.

If the tests suite produces a checklist, the JavaTest harness enables the Show Checklist menu item. To view the Configuration Checklist of the current configuration interview choose Configure -> Show Checklist.

To close both the Configuration Checklist and the viewer, click the Close button at the bottom of the window.

Using the Test Environment Browser

The Test Environment browser displays the configuration values used to run a test suite.



Entry Name	Value	Defined in File	Defined in Environment
user.home	C:\Documents and Setti...	(system properties)	
user.dir	C:\raziws	(system properties)	
command.execute	com.sun.javatest.lib.Ex...	configuration interview	
description	error	configuration interview	
testVerboseLevel	medium	configuration interview	

Select values in the table above to display them below:

```
com.sun.javatest.lib.ExecStdTestOtherJVMCmd C:\Program Files\Java\j2re1.4.2_01\bin\java.exe -class
path $testSuiteRootDir\classes;$testSuiteRootDir\..\..\demoapi.jar $testExecuteClass $testExecuteArg
s
```

Open the Test Environment browser by choosing Configure **->** Show Test Environment from the Test Manager menu bar.

The Test Environment browser contains a four-column table that displays information derived from the configuration file and used in configuring the test run. Useful features of the Test Environment browser include the following:

- Click and drag the column headers or their separators to rearrange the order and change the size of the columns.
- Click inside a table cell to display its contents in the text box below the table. This is a useful feature when the contents of a cell are too long for the table to effectively display them.
- Click a column header to alpha-numerically sort the contents of the table. This is a useful feature when searching for a specific name or value in a lengthy table.

The following table describes the contents of the Test Environment browser.

TABLE 31 Test Environment Dialog Box Contents

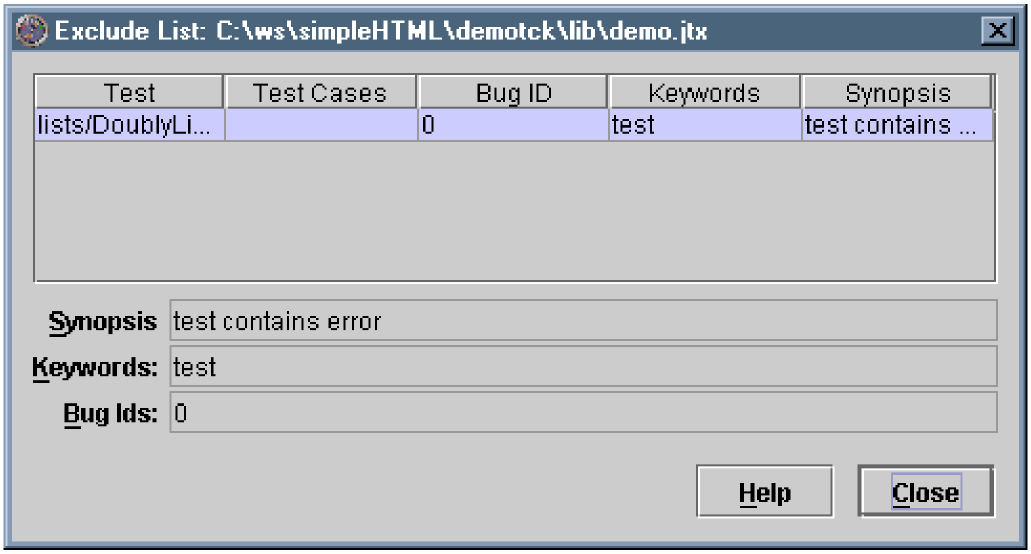
Name	Description
Entry Name	Identifies a name-value pair derived from the configuration file and used by test suite specific plug-in code to execute and run tests.

TABLE 31 Test Environment Dialog Box Contents

Value	A value specified by the user in the configuration file used to configure a test run.
Defined in file	Identifies the source of the configuration information used to run the tests. If your test suite uses a configuration (`.jti`) file to run tests, the source of the values is displayed. For test suites that use parameter (`.jtp`) and environment (`.jte`) files, the path and name of the appropriate environment file are displayed.
Defined in environment	When appropriate, this field displays the environment name from the configuration.

Using the Exclude List Browser

The Exclude List browser displays the list of tests excluded from the test run by the exclude list.



Test	Test Cases	Bug ID	Keywords	Synopsis
lists/DoublyLi...		0	test	test contains ...

Synopsis: test contains error

Keywords: test

Bug Ids: 0

Open the Exclude List browser by choosing Configure **->** Show Exclude List from the Test Manager menu bar.

JavaTest harness opens the Exclude List browser and displays the contents of the exclude list used to run tests and details about each test. You can use the browser to view but not edit the contents of the exclude list.

Use the Configuration Editor to add or remove exclude lists in a configuration. See Configuring a Test Run for detailed information.

Exclude List Contents

Information about individual tests in the exclude list is displayed in single, multi-column rows. Click a test in the list to display specific details in the text fields at the bottom of the panel.

Test Details

The following table describes the details displayed about individual tests highlighted in the Exclude List contents area.

TABLE 32 Exclude List Test Details

Field	Description
Synopsis	Provides annotated information about the excluded test
Keywords	Provides a list of keywords that describe why the test was excluded
Bug Ids	Lists the bug tracking IDs for the excluded test

Displaying the Question Log

The JavaTest harness creates a log file of all the completed questions in the current, saved configuration file and their answers. The JavaTest harness does not update the question log until you save the configuration file or click the **Done** button in the Configuration Editor.

Open the Question Log of the current, saved configuration file by choosing Configure -> Show Question Log from the Test Manager menu bar.

The log provides a list of all questions in the saved configuration with links to the following details about each question:

- Question
- Question tag
- Question description
- Response (if appropriate)

7

Running Tests

In the JavaTest harness GUI, you can use the Run Tests menu or the Test Manager toolbar to start a test run of the tests specified in your current configuration. See Test Manager Window for a description of the Run Tests menu and the Test Manager toolbar. See Configuring a Test Run for a description of how to use the Configuration Editor to specify configuration information used to run tests.

You can also use the test tree pop-up menu to start a test run of one or more tests selected from the test tree. See Test Tree Pop-up Menu for a description of how to use the test tree pop-up menu to run individual tests in a test tree.

Note – If you do not want to use the JavaTest harness GUI to run tests, you can use the command line. See Command-Line Overview in the *JavaTest Harness User's Guide: Command-Line Interface* for information about running tests from the command line.

The JavaTest harness saves all test results after running tests but does not automatically generate reports of test results. You must generate test reports from the Test Manager window or from the command line. See Generating and Viewing Reports for detailed information about reports.

If you are using the All Tests view filter and begin a test run, the JavaTest harness displays an advisory dialog box. Using the All Test view filter displays the results of all tests in a test suite regardless of whether or not they are included in the test run.

You can choose to disable the dialog box by setting JavaTest harness Preferences, by choosing a different view filter, or by using the check box in the dialog to stop the dialog from being displayed in the future.

This chapter contains the following topics, presented in a sequence that you can use when running tests:

Starting a Test Run - Describes how to use the Test Manager window to start a test run using the current configuration.

Monitoring a Test Run - Describes how to use the Test Manager window to monitor a test run.

Stopping a Test Run - Describes how to use the Test Manager window to stop a test run.

Troubleshooting a Test Run - Describes how to use the Test Manager window to troubleshoot a test run.

Starting a Test Run

When the JavaTest harness is not running tests, it enables both the ▶ button on the toolbar and the Run Tests -> Start menu item.

Note – Only one test run at a time can be active in each Test Manager window.

To start a test run using the current configuration, either click the ▶ button or choose Run Tests -> Start. You can also use the test tree popup menu to run a specific test or group of tests in a folder. See Test Tree Popup Menu.

Before the JavaTest harness attempts to run the test suite, it verifies that the required configuration information is complete. You can view the configuration state in the Test Manager Properties browser.

If the configuration information is not provided or is incomplete, the JavaTest harness opens a dialog box advising you that the configuration must be completed before it can begin running tests. You can choose to open the configuration editor window or cancel the test run. If you choose to open the configuration, the JavaTest harness opens the configuration at the incomplete section. You can also complete the configuration by choosing Configure -> Change Configuration -> Other values directly from the Changing Other Configuration Values.

To change the test suite or work directory before running tests, refer to the following topics:

- Opening a Test Suite
- Opening a Work Directory
- Creating a Work Directory

If a JavaTest harness Agent is used to run the tests for your product, you must start the agent before you begin the test run. See *JavaTest Harness Agent User's Guide* for detailed information about available JavaTest harness Agents.

If the JavaTest harness starts the test run and issues a request before the active agent starts running, the harness waits for an available agent until the timeout period ends. If the timeout period ends before an agent is available, the JavaTest harness reports an error for the test.

Monitoring a Test Run

After the test run begins, the Test Manager window displays the test run progress in five areas as described in the following table.

TABLE 33 Monitor a Test Run

Area	Description
Test Tree	The test tree uses colored icons to display the current run and test results status of the folders and tests in the work directory. As the JavaTest harness completes running individual tests, it updates each test tree icon to indicate the test result status. See Using the Test Tree for detailed information about using the test tree to monitor the progress of the test run.
Status Messages	Below the test tree area is a resizable text area that displays information about current Test Manager activity. In this area the JavaTest harness displays status messages about the state of the test run and the name of the test being run.
Progress Indicator	The Information Area contains a progress indicator at the bottom left of the pane. The indicator displays the elapsed time of the previous test run when tests are not running. When tests are running, it automatically changes to the progress bar of the current test run. At the completion of the test run, the indicator changes to display the elapsed time. See Using the Progress Meter for detailed information about using the test progress meter to monitor the progress of the test run.
Progress Monitor	A separate Progress Monitor is available that displays current, detailed information about the progress of the test run. See Using the Progress Monitor for detailed information.
Information Area	As tests run, the JavaTest harness displays information about the run in the information area to the right of the test tree. The information area provides two views: • Folder view - When you click a folder icon in the test tree, the JavaTest harness displays a Summary tab, five status tabs, and a status field containing information from the work directory about a folder and its descendants. See Folder View for detailed information about browsing folder information. • Test view - When you click a test icon in the test tree or double click its name in the Folder view, the JavaTest harness displays five tabbed panes that contain detailed information about the test. See Test View for detailed information about browsing test information.

The JavaTest harness also provides a web server that you can use to remotely monitor and control batch mode test runs. The HTTP Server provides two types of output:

- HTML Formatted Output allowing users to remotely monitor batch mode test runs in a web browser
- Plain Text Output intended for use by automated testing frameworks

See the *JavaTest Harness User's Guide: Command-Line Interface* for details about running tests from the command line and using the web server.

Using the Test Tree

The test tree uses folder icons, test icons, and two independent types of filtering (run filtering and view filtering) to simultaneously display the following conditions:

- Progress of a test run
- Current test results in the work directory

When a test run begins, you can track its progress in the test tree by observing the folder and test icons. The test tree displays an arrow at the left of each folder and test icon as it is loaded and run.

After the JavaTest harness completes running a test, it writes the test results to the work directory and updates the folder and test icons in the test tree. The test tree displays folder and test icons based on the view filter specified in the Test Manager window.

The JavaTest harness supports using status colors specified by the user instead of the default color settings. See Specifying Status Colors in the *JavaTest Harness User's Guide: Command-Line Interface*.

Regardless of whether or not a test was run, the test tree displays filtered out folder and test icons for those tests and folders filtered out by the view filter. All other icons are updated to reflect their current result status from the work directory.

Changing either the run or the view filter settings causes the JavaTest harness to immediately update the folder and test icons displayed in the test tree.

See Using View Filters for a description of how to specify which test results from the work directory are displayed in the test tree.

See Test Tree for a detailed description of the icons, filters, and other features used in the test tree.

The goal of a test run, when using the appropriate view filter, is for the root test suite folder to display the passed folder icon. The passed root test suite folder icon signifies that all tests in the test suite not filtered out of the test run (by specifying tests that are run, exclude lists, keywords, and prior status) have passed test results. See Change Standard Values for a description of the run filters that can be used to include or remove tests from a test run.

Click the test suite icon in the test tree to display status information for the test suite in the Test Manager information area. The view filter used in the test tree is also used to display folder status information. See Folder View for a detailed description of the folder information displayed in this area.

By browsing the tabbed pane and the test tree, you can find the folders that contain tests that do not have passed test results.

Using the Progress Indicator

The progress indicator located at the bottom of the Information Area displays the elapsed time of the previous test run when tests are not running. When tests are running, it automatically changes to the progress bar of the current test run. At the completion of the test run, the indicator changes to display the elapsed time.

You can use the ▼button at any time to select and display either the elapsed time or the progress bar.

The 🔍button opens and closes a separate Progress Monitor dialog box. The Progress Monitor provides additional detailed information about the test run that is not provided by the progress indicator. See Using the Progress Monitor for a detailed description.

Elapsed Time

When a test run starts, the elapsed time display resets to 00.00.00. At the end of the test run, the total elapsed time for the test run is displayed.

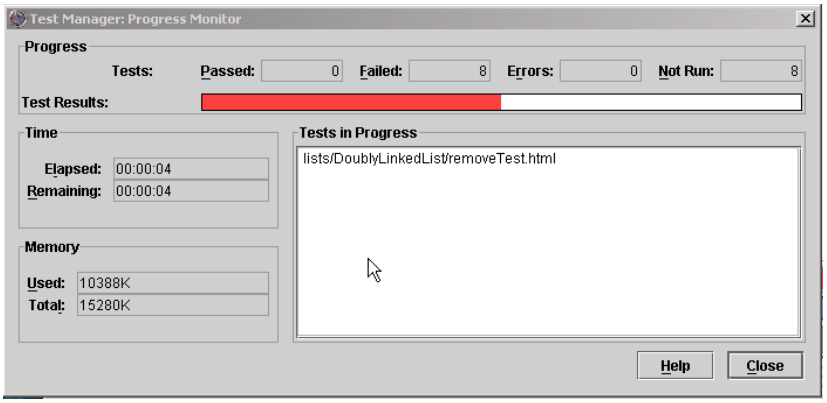
Progress Bar

The JavaTest harness displays a progress bar and a percentage complete value. As the JavaTest harness completes a test, it updates the progress bar and the percentage complete value.

Using the Progress Monitor

The Progress Monitor is a dialog box that displays information about the current configuration only when the harness is running tests. The information displayed is equivalent to the Last Test Run view filter set in the Test manager window. Changing view filters in the Test Manager window does not change the information displayed by the Progress Monitor.

Choose Run Tests **->** Monitor Progress from the Test Manager menu bar or click the 🔍 icon at the bottom of the Information Area to open the Progress Monitor.



Tests: Passed: 0 Failed: 8 Errors: 0 Not Run: 8

Elapsed: 00:00:04
Remaining: 00:00:04

Used: 10388K
Total: 15280K

Tests in Progress
lists/DoublyLinkedList/removeTest.html

The following areas in the Progress Monitor display information about the test run:

- Progress
- Tests in Progress
- Memory
- Time

Progress

The following table describes the information displayed in the Progress area.

TABLE 34 Progress Area Contents

Name	**Description**
Passed	Displays the number of tests in the test suite that were run and had passing results.
Failed	Displays the number of tests in the test suite that were run and had failing results.

TABLE 34 Progress Area Contents

Errors	Displays the number of tests in the test suite that could not be run.
Not Run	Displays the number of tests in the test suite have not yet been run.
Test Results	A colored progress bar representing the results of the tests in the test suite. As the JavaTest harness runs tests, the test results are displayed as colored segments in the progress bar. The colors used in the progress bar represent the current status of the test results. The colors below are the JavaTest harness default settings. The JavaTest harness supports using colors other than the default settings. See Specifying Status Colors in the *JavaTest Harness User's Guide: Command-Line Interface*. The progress bar is the same as that displayed in the test progress display at the bottom of the Test Manager window.

The following table describes the colors used in the progress bar. The colors are displayed from left to right in the order in which they are presented in the table.

TABLE 35 Order of Colors Displayed in the Progress Bar

Color	Status	Description
Green	Passed	Tests in the test run having passing results when they were executed.
Red	Failed	Tests in the test run having failed results when they were executed.
Blue	Error	Tests in the test run that the JavaTest harness could not execute. Errors usually occur because the test environment is not properly configured.
White	Not yet run	Tests in the test run that the JavaTest harness has not yet executed. Tests excluded from the test run are not included.

You can display the progress bar in the Test Manager window by clicking the ▼button and choosing Run Progress Meter from the selectable list. See Using the Progress Meter for detailed description.

Tests in Progress

The Tests in Progress text box displays either the names of the tests that the JavaTest harness is currently running or the set of tests that have been distributed for execution. It is empty when the JavaTest harness is not running tests. Concurrency settings and agents in use determine the number of items displayed in the text box. See your test suite documentation for additional information.

Clicking on this list displays the appropriate test view in the Test Manager window.

Memory

The Memory area contains two text fields and a bar graph. The following table describes the contents of the Memory area in the Progress Monitor.

TABLE 36 Memory Area Contents

Name	Description
Used:	The memory used to run the test.
Total:	The total memory available for use by the Java virtual machine.

Time

The Time area contains two fields that are continuously updated throughout a test run. The following table describes the contents of the Time area in the Progress Monitor.

TABLE 37 Time Area Contents

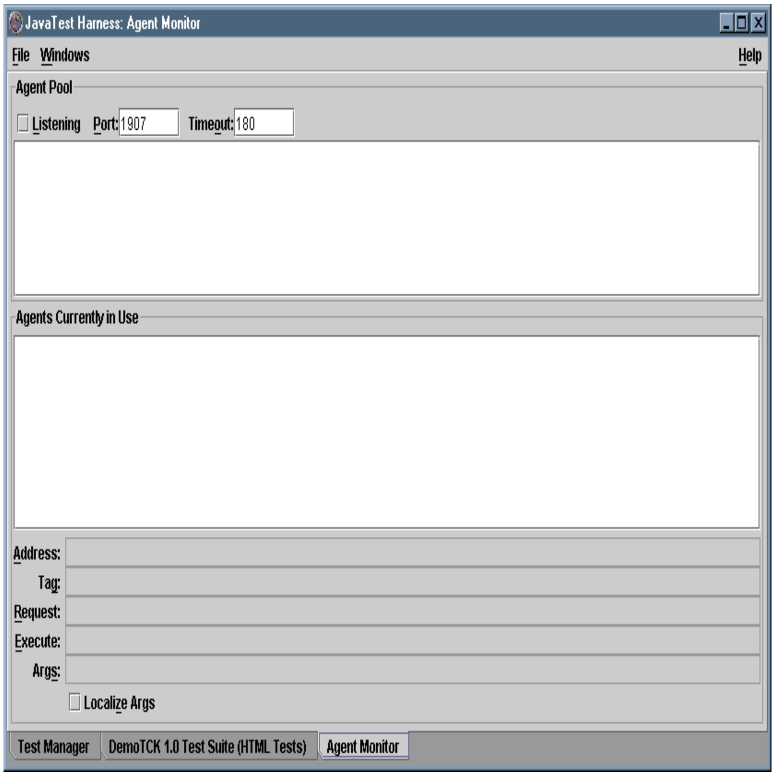
Name	Description
Elapsed:	The time that has elapsed since the test run was started.
Remaining:	The estimated time required to run the remaining tests.

Tests that timeout or have execution times significantly longer than the other tests being run can cause the JavaTest harness to display inaccurate times.

Agent Monitor Window

Open the Agent Monitor window by using the JavaTest harness GUI Test Manager to choose Window **->** Open **->** Agent Monitor.

The Agent Monitor window contains two sections, Agent Pool and Agents Currently In Use.



☐ Listening Port: 1907 Timeout: 180

☐ Localize Args

Agent Pool

Agent Pool lists the active agents that are available to run tests. When active agents connect to the JavaTest harness, they are added to the agent pool. When the JavaTest harness requires an active agent to run a test, it moves the agent from Agent Pool to Agents Currently In Use until the test is completed.

The following table lists and describes the contents of the Agent Pool GUI.

TABLE 38 Agent Pool GUI Contents

Field	Description

TABLE 38 Agent Pool GUI Contents

`Listening`	Click the check box to enable listening for active agents. If listening is not enabled when an agent starts, the agent issues a message that it cannot connect to the JavaTest harness and then waits for its timeout period to end before attempting to recontact the harness.
`Port`	Port 1907 is the default port used by active agents. If your agent uses a different port, you must either change the value used by the agent or change this value to match the agent.
`Timeout`	When the agent pool is empty, the timeout value sets the number of seconds that the JavaTest harness waits between tests for an available agent before reporting the test result as an error. If you run tests with one agent, a latent period might occur between the time when the agent completes the test and when it returns to the agent pool. The timeout value must be greater than the agent's latent period. The default value of 180 seconds is usually sufficient.

Agents Currently In Use

Agents Currently In Use lists all agents currently used by the JavaTest harness to run tests. When agents are not running tests they are removed from the list (active agents re-register with the agent pool). Click on an agent in the list to display detailed information about the agent and the test it is running. The detailed information is displayed in the text fields at the bottom and can be used to troubleshoot problems using an agent to run tests.

The following table lists and describes the contents of the Agents Currently In Use GUI.

TABLE 39 Agents Currently In Use GUI Contents

Field	Description
`Address`	Network address of the agent
`Tag`	Test executed by the agent
`Request`	Function executed by the agent
`Execute`	Class executed by the agent
`Args`	Arguments passed to the class executed by the agent
`Localize Args`	Checked if the agent uses a map file

Stopping a Test Run

When the JavaTest harness is running tests, it enables both the ■ button on the toolbar and the Stop menu item.

Either click the ■ button or choose Run Tests -> Stop to stop a test run.

As it completes each test, the JavaTest harness writes the test results (`.jtr` files) in the work directory. Stopping a test run causes the tests in progress to indicate an error.

When you stop a test run, the JavaTest harness does not generate reports of test results. You must generate the reports from the Test Manager window or from the command line. See Test Reports for detailed information about test reports.

Troubleshooting a Test Run

Normally, the goal of a test run is for all tests in the test suite that are not filtered out by the current configuration to have passing results. See Changing Configuration Values for a description of how the current configuration filters tests in a test run.

If the root test suite folder contains tests with errors or failing results, you must troubleshoot and correct the cause to satisfactorily complete the test run.

Tests with errors ! are tests that could not be executed by the JavaTest harness. These errors usually occur because the test environment is not properly configured or the software under test is defective. See Tests with Errors for a detailed description of troubleshooting tests with errors.

Tests that failed ☒ are tests that were executed but had failing results. See Tests that Fail for a detailed description of troubleshooting tests that failed.

The Test Manager window provides you with the following facilities to effectively troubleshoot a test run:

- Test Tree
- Folder View
- Test View

Test Tree

Use the test tree and view filters to identify specific folders and tests with errors or failing results. Open the red and blue folders until the specific tests that failed or had errors are displayed.

Folder View

When you click a folder icon in the test tree pane, the JavaTest harness displays a filtered summary of its test status in the Test Manager information area that matches the test tree.

Note – The View filter chosen in the Test Manager window might change the summary values displayed in the folder view, but does not change the test results written in the work directory.

Click the Error and the Failed tabs to display the lists of all tests in and under a folder that were not successfully run. You can double-click a test in the lists to view its detailed test information. Refer to Test View below for a description of the test information that the JavaTest harness displays.

Test View

When you click a test icon in the test tree or double-click its name in the folder view, the JavaTest harness displays unfiltered, detailed information about the test in the information area. The Test Manager displays the current information for that test from the work directory.

Note – Because the Test Manager does not use a view filter when displaying test information, the test status displayed in the information area may not match the filtered view in the test tree or the folder Summary view.

The test view contains five detailed test information panes and a brief status message at the bottom identifying the type of result. This message may be sufficient for you to identify the cause of an error or failure.

If you need additional information to identify the cause of the error or failure, use the following panes listed in order of their importance:

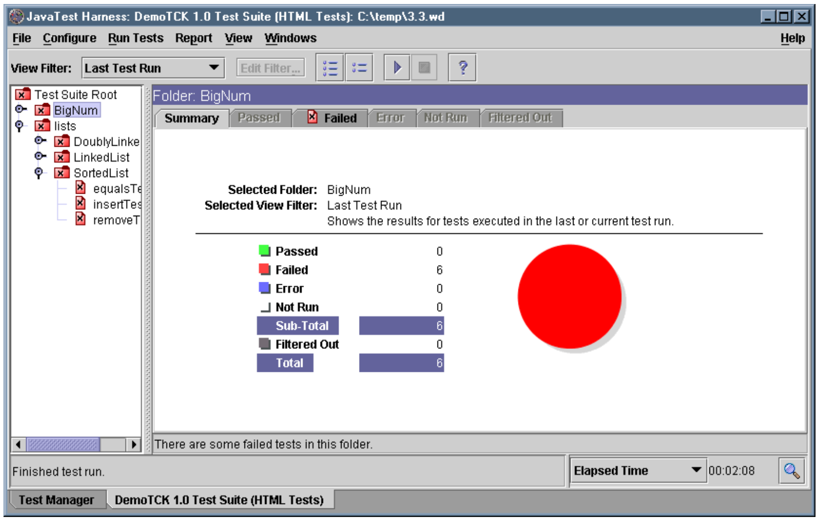
- Test Run Messages contains a Message list and a Message pane that display the messages produced during the test run
- Test Run Details contains a two-column table of name-value pairs recorded when the test was run

- Configuration contains a two-column table of the name-value pairs derived from the configuration data that were actually used to run the test.

8

Browsing Test Information

You can quickly browse test information in the Test Manager window by clicking folder and test icons in the test tree.



View Filter: Last Test Run

Selected Folder: BigNum
Selected View Filter: Last Test Run
Shows the results for tests executed in the last or current test run.

🟩 Passed	0
🟥 Failed	6
🟦 Error	0
Not Run	0
Sub-Total	6
⬛ Filtered Out	0
Total	6

There are some failed tests in this folder.

Finished test run.

Elapsed Time 00:02:08

This chapter is divided into the following topics:

- **Test Tree** - Describes the test tree and filtering used to display the test suite, its folders, tests and status icons.
- **Folder View** - Describes how to display filtered folder information in the Test Manager window.
- **Test View** - Describes how to display unfiltered test information in the Test Manager window.
- **Viewing Test Manager Properties** - Describes how to display the Test Manager Properties browser.
- **Viewing Test Suite Errors** - Describes how to display the Test Suite Errors dialog box.

Test Tree

The test tree is a multifunction panel that not only displays the current status and state of tests in the test suite but also enables users to choose any combination of tests and folders in the test suite for a test run and to refresh or to clear the test results. The test tree supports keyboard navigation. See Keyboard Access for a description of how the keyboard can be used to navigate the test tree pane.

Displaying Current Status and State of Tests

The test tree uses folder icons, test icons, and two independent types of filtering (run filtering and view filtering) to simultaneously display the progress of a test run and the current test results in the work directory.

The test tree indicates the progress of a test run by displaying an arrow to the left of each folder and test icon as it is loaded and run. See Using the Test Tree for detailed information about using the test tree to monitor a test run.

Based on the Test Manager view filter, the test tree updates the folder and test result icons to indicate the current test results in the work directory.

Example:

When you choose the Current Configuration view filter, the test tree only displays the results for the tests specified in your current configuration. If you choose the All Tests view filter, the test tree immediately redisplays all test results from the work directory regardless of the settings in your current configuration. To display filtered summary information about the test results in the work directory, click the test suite icon in the test tree.

See Using View Filters for a detailed description of how to specify which test results from the work directory are displayed in the test tree.

When you click a folder icon in the test tree, the JavaTest harness displays its folder view in the Test Manager information area. The Test Manager uses the view filter to filter results displayed in the folder view. See Folder View for additional information.

When you click a test icon in the test tree, the JavaTest harness displays its test view in the Test Manager information area. Unlike the folder view, the Test Manager does not filter results in the test view. See Test View for additional information.

Choosing Tests and Folders to Run, Refresh, or Clear Results

You can select any combination of tests and test folders and right click in the test tree to open a pop-up menu for performing the following actions:

- Execute highlighted folders and tests by performing a quick pick execution.
- Refresh the list of folders and tests without restarting the JavaTest harness by performing an on-demand refresh scan for new folders, new tests and updated test descriptions.
- Clear Results by performing an on-demand clearing of the contents of the highlighted folder, test, or entire work directory.

If a single test of folder is highlighted in the test tree, the Test Manager displays the appropriate Test View of Folder View. However, when multiple tests and folders are highlighted in the test tree, the JavaTest harness displays a Multiple Tree Nodes Selected pane that lists the highlighted tests and folders. Highlight multiple tests and folders in the test tree by using standard keystrokes (such as Shift and Control) and clicking the test and folder icons in the test tree.

The selection list in the Multiple Tree Nodes Selected pane displays the full path of each node highlighted in the test tree. Click a test or folder in the test tree a second time to remove the highlighting and to remove it from the selection list. The GUI displays the updated Multiple Tree Nodes Selected pane.

Folders and tests do not have to be highlighted in the test tree for you to use the pop-up menu. See Test Tree Pop-up Menu for detailed information about using the pop-up menu.

Folder Icons

The test tree uses colored icons to indicate both run status and result status. The colors of the icons shown below are the JavaTest harness default settings. The JavaTest harness enables you to use colors other than these default settings. See Specifying Status Colors in the *JavaTest Harness User's Guide: Command-Line Interface*.

Folder Run Status

When activity occurs in a folder, such as loading or running tests, the JavaTest harness displays an arrow to the left of the folder icon. The standard values of the current configuration, not the view filter, determine the run status displayed by the test tree.

Folder Result Status

The folder icon indicates the current test results in the work directory and does not change until its tests are completed. After the JavaTest harness completes the tests in a folder, it displays the appropriate result status icon.

The folder icon displayed in the test tree is determined by the result of all its tests (see Test Icons) and the current view filter (see Using View Filters). The folder icons displayed in the test tree indicate the highest priority result of any test hierarchically beneath it. The following table describes the folder icons in order of priority.

TABLE 40 Folder Icon Descriptions and Priority Order

Icon	**Result**	**Description**
!	Error	A blue folder containing an exclamation symbol (**!**) indicates that it and or one or more of its child folders contains tests with a result of Error. Note that this folder *might* also contain tests and folders that are Failed, Not Run, Passed, and Filtered out.
✗	Failed	A red folder containing a **x** symbol indicates that it and-or one or more of its child folders contains tests with a result of Failed. Note that this folder *might* also contain tests and folders that are Not Run, Passed, and Filtered out.
-	Not Run	A white folder containing a **-** symbol indicates that it and-or one or more of its child folders contains tests with a result of Not Run. Note that this folder *might* also contain tests and folders that are Passed and Filtered out.
✓	Passed	A green folder containing a ✓symbol indicates that it and all of its children have a result of Passed. Note that this folder *might* also contain tests and folders that are Filtered out.
▨	Filtered Out	A gray folder containing no symbols indicates that it and all of its children are filtered out.

Test Icons

The test tree uses colored icons to indicate both run status and result status. The colors of the icons shown below are the JavaTest harness default settings. The JavaTest harness enables you to use colors other than these default settings. See Specifying Status Colors in the *JavaTest Harness User's Guide: Command-Line Interface*.

Test Run Status

The JavaTest harness displays an arrow to the left of the test icon when running a test. The standard values of the current configuration, not the view filter, determine the run status displayed by the test tree.

Test Result Status

The test icon indicates the last known test result and does not change until the test is completed. After the JavaTest harness completes the test, it displays the appropriate result status icon.

The test result icons displayed in the test tree are determined by the results in the work directory and by the view filter set in the Test Manager window. See Using View Filters for a description of how to specify the test results displayed in the test tree. The following table describes the test icons used to indicate the result of each test.

TABLE 41 Test Icons

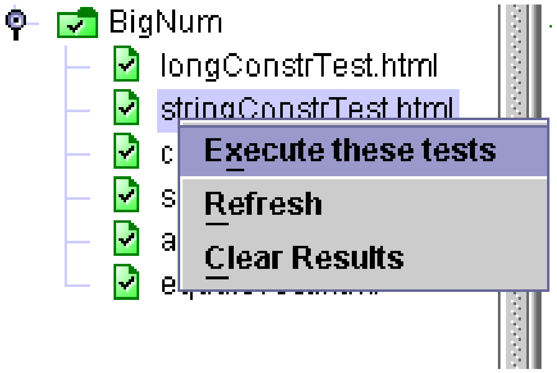
Icon	Result	Description
!	Error	A blue test containing an exclamation symbol (**!**) indicates that the test is not filtered out and that JavaTest harness could not execute it. These errors usually occur because the test environment is not properly configured.
✖	Failed	A red test containing a **x** symbol indicates that the test is not filtered out and had a Failed result the last time it was executed.

TABLE 41 Test Icons

-	Not Run	A white test containing a **-** symbol indicates that the test is not filtered out but has not yet been executed.
✓	Passed	A green test containing a ✓symbol indicates that the test is not filtered out and had a Passed result the last time it was executed.
▨	Filtered Out	A gray test containing no symbols indicates that the test is currently not selected to be run.

Test Tree Pop-up Menu

The test tree provides a pop-up menu. Displaying the pop-up menu for a folder or test is a platform specific operation (such as right clicking the folder or test icon in the test tree). You can also press Shift-F10 to display the pop-up menu for the test with focus.



The pop-up menu contains the following menu items:

- **Execute these tests** - Quick pick test execution.
- **Refresh** - Refresh test suite contents.
- **Clear Results** - Clear previous test results.

The JavaTest harness does not allow you to perform operations using the pop-up menu when it is running tests. The JavaTest harness displays an error dialog box if you attempt to perform an operation using the pop-up menu when tests are running.

See Keyboard Access for a description of how you can use the keyboard to open and navigate the pop-up menu.

Quick Pick Test Execution

You can use the Execute these tests menu item to run the tests and folders highlighted in the test tree.

The JavaTest harness does not automatically perform a refresh operation before running the tests. If changes are made to a test suite, you must perform a refresh before running tests. See Refresh Test Suite Contents for a description of the refresh operation.

To perform a quick pick test execution of specific folders or tests, complete the following actions:

1. Select one or more folders and tests in the test tree.

You must left click to select nodes. When more than one folder or test is highlighted in the test tree, the Test Manager displays a list of the selected tests and folders in the information area.

Selecting a test executes only that test.

Selecting a folder executes all tests currently known to the test manager in and below that folder.

1. Right click in the test tree pane to open the pop-up menu.

Opening the pop-up menu is a platform-specific operation (such as right clicking in the test tree pane).

1. Choose **Execute these tests** from the test tree pop-up menu.

In place of the Tests setting in the current configuration, the JavaTest harness uses the quick pick tree selection with the remaining values from the configuration to run the tests.

If the test manager does not contain a completed configuration, the JavaTest harness displays an advisory message and does not start the test run.

If the test manager contains a completed configuration, the JavaTest harness displays an advisory message to confirm the execute operation.

As it does during a routine test run, the JavaTest harness updates all icons and progress monitors during test execution.

Refresh Test Suite Contents

When developing tests, changes in a test suite are not automatically detected by the JavaTest harness. The first time tests are run, the JavaTest harness uses the test finder to read test descriptions. If the harness loads tests from an existing work directory, the test descriptions contained in those results are used by default.

The refresh operation enables test developers to load changes they might have made in a test suite without restarting the JavaTest harness or reloading the test suite.

The JavaTest harness does not require a work directory to perform a refresh of the test suite.

If you are viewing the test panel after refreshing a test or folder, update the test tree by choosing a different test or folder icon and repeating your test tree choice.

Refresh Tests and Folders

To refresh the contents of tests and folders, complete the following actions:

1. Select one or more folders and tests in the test tree.

You must left click to select a node. When more than one folder or test is highlighted in the test tree, the Test Manager displays a list of the selected tests and folders in the information area.

Selecting a test chooses only that test.

Selecting a folder chooses all tests currently known to the test manager in and below that folder.

1. Right click in the test tree pane to open the pop-up menu.

Opening the pop-up menu is a platform-specific operation (such as right clicking in the test tree pane).

1. Choose Refresh from the pop-up menu.

1. For files, the JavaTest harness checks the time stamp of the file containing the test description.

1. If the time stamp has changed, it compares the test descriptions.
2. If the properties of the test descriptions are different, the JavaTest harness removes the test result from the work directory and the test manager and loads a test containing the new test description into the test manager and displays it in the Not Run state.
3. For folders, the JavaTest harness checks the time stamps of the files in a folder and scans for new folders and tests. This operation may take place on any folder including the root folder.

1. If a time stamp has changed, the JavaTest harness compares the test descriptions.
2. If the properties of the test descriptions are different, the JavaTest harness removes the test result from the work directory and the test manager and loads the test containing the new test description into the test manager and displays it in the Not Run state.

Clear Previous Test Results

You can use the Clear Results menu item to remove existing test results for any combination of tests and folders.

To clear test results, you must have an open work directory.

Clear a Test Result

To clear a test result, complete the following actions:

1. Select one or more tests in the test tree.

You must left click to select a node. When more than one test is highlighted in the test tree, the Test Manager displays a list of the selected tests and folders in the information area.

1. Right click in the test tree pane to open the pop-up menu.

Opening the pop-up menu is a platform-specific operation (such as right clicking in the test tree pane).

1. Choose Clear Results from the test tree pop-up menu.

The JavaTest harness performs the following actions:

- Removes the .jtr file from the work directory for that test.
- Refreshes the test description for that test.
- Displays the test in the Not Run state.

Clear Test Results in Folders

To clear the test results in a folder, complete the following actions:

1. Select one or more folders in the test tree.

You must left click to select a node. When more than one folder is highlighted in the test tree, the Test Manager displays a list of the selected tests and folders in the information area.

1. Right click in the test tree pane to open the pop-up menu.

Opening the pop-up menu is a platform-specific operation (such as right clicking in the test tree pane).

1. Choose Clear Results from the test tree pop-up menu.

The JavaTest harness performs the following actions:

- Removes all .jtr files from the work directory for all tests in and below that folder.
- Deletes all other files in and below the folder in the work directory.

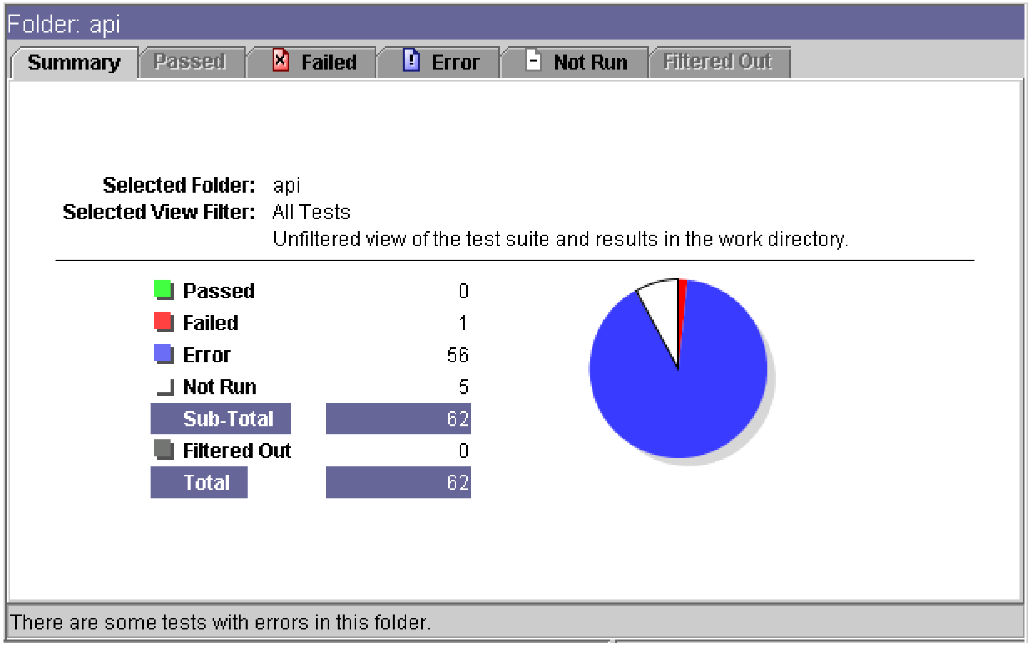
- Deletes all other directories corresponding to the folders in and below the folder in the work directory.
- Displays the folder and its tests as Not Run.

The JavaTest harness does not display an error message if it is unable to delete a folder or file from the work directory.

Folder View

The folder view contains filtered summary and status information about the tests in a test folder. This information is displayed both as values and as a pie chart. The folder view and test tree use the same view filter when displaying information.

To display the folder view, click a folder icon in the test tree. The folder view displays a Summary tab, five status tabs, and a status display.



Selected Folder: api
Selected View Filter: All Tests
Unfiltered view of the test suite and results in the work directory.

🟩 Passed	0
🟥 Failed	1
🟦 Error	56
Not Run	5
Sub-Total	62
▨ Filtered Out	0
Total	62

There are some tests with errors in this folder.

During a test run, you can use the folder view to monitor the status of a folder and its tests. You can also use the folder view during troubleshooting to quickly locate and open individual tests that had errors or failed during the test run. When a status pane is empty, the JavaTest harness disables its tab.

See Summary Information for a description of the information displayed in the Summary pane.

See Status Information for a description of the folder information displayed by clicking the following status tabs:

- ✓ Passed (green with a check mark)
- ✗ Failed (red with an x)
- ! Error (blue with an exclamation point)
- − Not Run (white with a dash)
- ▨ Filtered Out (gray or shaded)

The status message displayed at the bottom of the pane provides information about the selected tab. The messages indicate that tests in the folder are loading or provide detailed status information about a selected test.

Summary Information

When you click a folder icon in the test tree, the JavaTest harness uses the current view filter to display Summary information about the folder's test results.

The Summary pane contains header information that identifies the folder and the view filter presented in the Summary table and its associated pie chart. The pie chart displayed in the Summary pane is a graphical representation of the tabular data.

The following table describes the filtered work directory information that the Summary table and pie chart can display.

TABLE 42 Summary Pane Contents

Field	Description
Passed	The number of tests in a folder (including all of its subordinate folders) displayed in the test tree that were run and passed.
Failed	The number of tests in a folder (including all of its subordinate folders) displayed in the test tree that were run and failed.
Error	The number of tests in a folder (including all of its subordinate folders) displayed in the test tree that were run but had errors.
Not Run	The number of tests in a folder (including all of its subordinate folders) in the test tree that have not yet been run and were not filtered out.

TABLE 42 Summary Pane Contents

Sub-Total	The total number of tests that were selected to run.
Filtered Out	The total number of tests in a folder (including all of its subordinate folders) displayed in the test tree that were filtered out. Tests that were filtered out include tests that you omitted from the test run by using keywords, prior status, or exclude lists.
Total	Total number of tests in a folder and its subordinate folders.

Note – The GUI only displays results of those tests in the work directory that match your *view* filter setting. Changing the view filter used in the Test Manager window changes the values displayed in the Summary pane.

When using the All Tests view filter, new settings in the current configuration do not change the values displayed in the Summary pane.

When using the Current Configuration view filter, each time you make a change in the configuration, the values displayed in the Summary pane are recalculated and displayed based on the new settings in the current configuration. See the following example for a description of the use of the Current Configuration view filter and the All Tests view filter.

> Example:
> If you rerun a set of tests and use the Current Configuration view filter with Prior Status: Failed set in the configuration (see Using Prior Status), as actual test status in the work directory changes from Failed to either Passed or Error, the test status displayed in the Summary pane changes from Failed to Filtered Out.
>
> If you change the view filter to All Tests, the test tree and Summary pane immediately display the actual results of all tests in the work directory, regardless of the Prior Status settings in the configuration.

See Problems Viewing Test Results for additional information about viewing test status in the test tree and Summary pane.

Click the appropriate status information tab to identify the individual tests in a category.

Status Information

In addition to Summary information, the folder view contains five status tabs that use the view filter to group and list a folder’s tests by their results in the work directory.

The colors of the following icons are the JavaTest harness default settings. The JavaTest harness allows you to use colors other than these default settings. See Specifying Status Colors in the *JavaTest Harness User's Guide: Command-Line Interface*.

In the tabbed panes, click a test in the list to display the sumary message at the bottom of the pane. Double click the test name or press Enter to display the test in the test tree and to view its unfiltered test information.

Passed

Uses the view filter to display the test names of all tests in the folder (including all of its subordinate folders) displayed in the test tree that had passing results when they were run.

Failed

Uses the view filter to display the path names of all tests in the folder (including all of its subordinate folders) displayed in the test tree that were run and had failing results.

Error

Uses the view filter to display the path names of all tests in the folder (including all of its subordinate folders) displayed in the test tree with errors that prevented them from being executed.

Not Run

Uses the view filter to display the path names of all tests in the folder (including all of its subordinate folders) displayed in the test tree that are selected by the view filter but have not been run.

Filtered Out

Uses the view filter to display a two column list. The first column contains the names of tests in the selected folder whose results are filtered out by the selected view filter.

The second column contains a specific reason why the test result is filtered out by the view filter. The specific reason depends on the view filter criteria. See Using View Filters for a detailed description of filtering criteria.

Multi-Selection Panel

When multiple tests are selected in the test tree, the JavaTest harness displays a Multiple Selection panel in the Information Area.

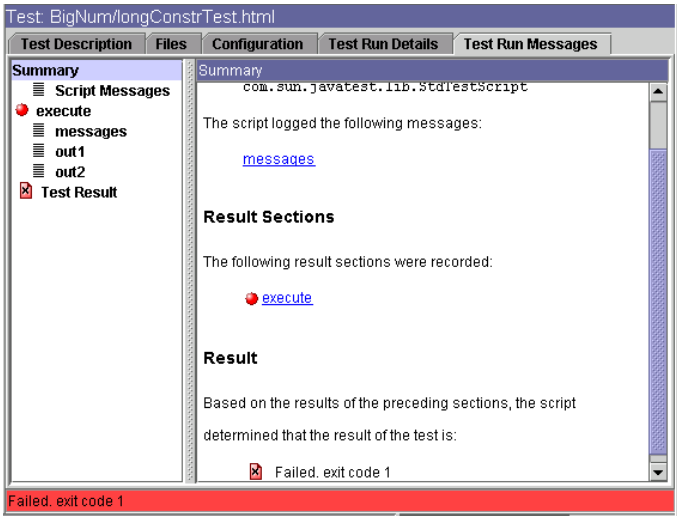
This panel contains a text description area at the top of the panel that describes the operations that can be performed using tests and folders selected in the test tree and listed below.

In the section below the text description area, the JavaTest harness displays a list of the tests and folders selected in the test tree.

Test View

The test view contains unfiltered, detailed information from the work directory about a specified test. To display the test view, click a test icon in the test tree or double-click a test name in status information list.

The test view contains five tabs and a colored status field.



com.sun.javatest.lib.StdTestScript

The script logged the following messages:

messages

Result Sections

The following result sections were recorded:

execute

Result

Based on the results of the preceding sections, the script determined that the result of the test is:

Failed. exit code 1

Failed. exit code 1

Note – The test view does not use view filtering to display information. If you are using a view filter other than All Tests, the status color displayed in this view might not match the test icon or the folder view tab.

The following table describes the contents of the test view panes. The status field at the bottom of the pane contains a description of the test result and is visible in all of the test view panes.

TABLE 43 Information Pane Contents

Tab	Description
Test Description Pane	Displays the name-value pairs contained in the test description. The contents are input data and always available.
Files Pane	Contains a drop-down list of source files from the test description. Click a file name from the drop-down list to display its contents. The contents are input data and always available.
Configuration Pane	Displays a table of configuration name-value pairs used to run a specific test. The contents are output data and only enabled if the test was run.
Test Run Details Pane	Displays the name-value pairs that were recorded when the test was run. The contents are output data and only enabled if the test was run.
Test Run Messages Pane	Contains a tree and message panel of output from sections of the test. Click a name to display its contents. The contents are output data and only enabled if the test was run.

When an information pane is empty, the JavaTest harness disables it.

Test Description Pane

Click the Test Description tab to display a two-column table of name-value pairs derived from the test descriptions cached in the work directory. Each test in a test suite has a test description.

When you open the test suite in the JavaTest harness, the test finder reads the test descriptions and caches them in the work directory. Test descriptions in the cache and the Test Description pane are not updated until you close and reopen the test suite. Refer to your test suite documentation for detailed descriptions of the names and values displayed in the Test Description pane.

Name

The names displayed in the table identify the attributes and properties contained in the test description.

Value

The values displayed in the table are the attribute and property values that the JavaTest harness used to run the test. The values are read from the files in the test suite.

Files Pane

Click the Files tab to display a drop-down list of source files and the test description file for the test.:

For compiler tests, the source files are those files that were compiled during the test run.

For runtime tests, the source files are those files that were previously compiled to create the test class files.

Choose a file from the drop-down list to display its contents. You can browse but not edit source files in this panel.

Configuration Pane

Click the Configuration tab to display a two-column table of the name-value pairs that were derived from the configuration file and actually used to run the test. The contents of this pane vary for each test suite. Because the table contains values that were used when the test was run, it may provide valuable information when troubleshooting a test run. Refer to your test suite documentation for detailed descriptions of the name-value pairs for your test.

Name

The names in the table identify test environment properties used by the JavaTest harness to run the test.

Value

The values displayed in the table were used to run the test.

Test Run Details Pane

Click the Test Run Details tab to display a two-column table of name-value pairs that were recorded when the test was run and may provide valuable information when troubleshooting a test run. Refer to your test suite documentation for detailed descriptions of the result property name-value pairs for your test.

Name

The JavaTest harness derives property names from the test results file and displays them in the table with the following defaults:

- Information about the version of the JavaTest harness used to run the test
- Information about the operating system used to run the test
- Date and time the test started
- Date and time the test ended
- Additional details recorded by the test script used to run the test

Because the properties listed in the table are a function of the test script that you are running, the contents vary for each test suite.

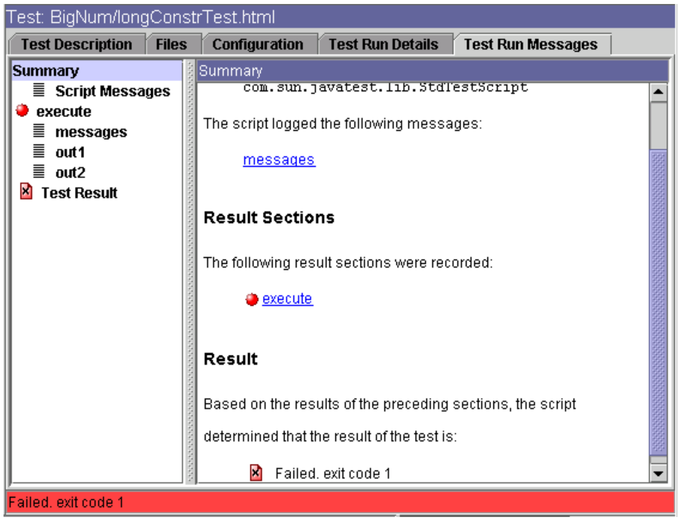
Information written by commands, tests, and scripts as they are executing is displayed in the Test Run Messages pane.

Value

The values displayed in the table are from the test results file created by the JavaTest harness after running the test.

Test Run Messages Pane

Click the Test Run Messages tab to display detailed messages describing what happened during the running of each section of the test. This information is useful when troubleshooting a test run.



com.sun.javatest.lib.StdTestScript

The script logged the following messages:

messages

Result Sections

The following result sections were recorded:

● execute

Result

Based on the results of the preceding sections, the script determined that the result of the test is:

☒ Failed. exit code 1

Failed. exit code 1

The Test Run Messages tab contains the following areas:

- **Message List** - The vertical area on the left side of the Test Run Messages tab.
- **Message Area** - The vertical area on the right side of the Test Run Messages tab.
- **Test Result Status Bar** - The horizontal area at the bottom of the Test Run Messages tab.

Message List

The message list provides a detailed list of messages issued during a test run. Click an item in the list to display its contents in the message area. The message list contains links to the following types of messages:

- Summary Message
- Output Summary and Result Messages
- Test Result Message

Summary Message

Only one per test, this message summarizes all of the messages generated during a test run and provides hypertext links to their detailed contents. The Summary Message contains the following information in the message area:

- Test script used to run the test
- Messages logged by the test script
- Individual test result sections
- Test result and its result icon

Script Messages

This message is passed up from the script that executed the test. There is only one script message per test. Script messages vary for each test script. Refer to your test suite documentation for detailed descriptions of its script messages when troubleshooting a test run.

Output Summary and Result Messages

Each test result section has an Output Summary and Result message that provides summary messages and hypertext links to its detailed messages. The name of the Output Summary message is a function of the test suite and varies for each test suite.

Some tests have only one result section, while others have multiple sections. Refer to your test suite documentation for detailed descriptions of the tests when troubleshooting a test run.

The following table lists and describes the message types.

TABLE 44 Output Summary and Result Messages

Message Type	Description
Output Summary	A two-column table listing the name and size of each output section. Each of the following output sections contains text generated while executing the test section: • **messages** - Provides the command string used by the test script to run the test section. Unlike ref and log, the messages field always exists in a section. • **ref** - The name of this message field is determined by the test and might be a name other than ref. A test can use this output stream to provide standard output information from the test section. • **log** - The name of this message field is determined by the test and might be a name other than log. A test can use this output stream to provide standard error information from the test section. Many tests only use the log stream and include tracing as well as standard error information when writing to the log output. The contents of each output section varies from test suite to test suite. Refer to your test suite documentation for detailed descriptions of the test section messages when troubleshooting a test run. If no details exist in an output section, the JavaTest harness does not create its hypertext link and indicates in the Size (chars) column that it is empty.
Result	Contains a colored status icon and a brief description of the results of the specific test section. The color of the circle indicates the result of the test section.

Test Result Message

The Test Result Message indicates the cumulative result of the test, determined by the test script from the results of the preceding test sections (if any). There is only one Test Result Message per test.

Note – For negative tests, the Test Result correctly indicates Passed when all of its test sections have failed.

Message Area

The area displays the messages issued during a test run. The number, names, and content vary for each test suite and may also vary for different tests in the same test suite.

Test Result Status Bar

The area displays an abbreviated form of the Test Result Message. See Test Result Message for detailed information.

Using View Filters

The JavaTest harness provides a special view filtering facility that enables you to filter the status (the colors and counters) of the folders and tests displayed in the Test Manager window. View filters function independently from the run filtering set in the configuration editor window.

See Changing Configuration Values for detailed information about specifying the tests that are run.

Note – You also use view filters when generating reports. See Creating New Reports for a description of using view filters when generating reports.

The JavaTest harness provides four Test Manager view filters:

- All Tests (default)
- Last Test Run
- Current Configuration
- Custom

An additional view filter, such as a Certification filter, can be added by the test suite. Refer to your test suite documentation for detailed descriptions of any additional filters displayed in the list of view filters.

To use a view filter, either choose View -> Filters from the menu bar or choose a view filter from the toolbar. See Custom View Filter for a description of how to create a custom view filter.

Current view filter information is saved when you exit if you checked the Save Desktop State on Exit option in the Preferences dialog box.

Current view filter information is not saved if you unchecked the Save Desktop State on Exit option in the Preferences dialog box, provide test suite or work directory information on the command line when starting the JavaTest harness, or use the `-newDesktop` option when starting the JavaTest harness.

All Tests View Filter

When you select the All Tests view filter, the Test Manager window immediately displays current totals in the Summary pane and status icons in the test tree for all folders and tests, regardless of the values set in the configuration. This is an unfiltered view of the complete work directory contents.

The JavaTest harness only overwrites the previous results for tests when they are rerun. The All Tests filter displays the last run status for all tests in the work directory, including the tests that were excluded from the current test run.

If you are using the All Tests view filter and begin a test run, the JavaTest harness displays an advisory dialog box. Using the All Test view filter will display the results of all tests in a test suite regardless of whether or not they are included in the test run.

You can choose to disable the dialog box by setting JavaTest harness preferences, by choosing a different view filter, or by using the check box in the dialog to stop the dialog from being displayed in the future.

The following examples provide descriptions of the use of the All Tests view filter.

- **Example 1 -** A test run had failed tests and you want to rerun only the failed tests. See Using Prior Status for a detailed description of setting the prior status value. The Test Tree and the Summary pane do not change.

When you use the All Tests view filter, the Test Manager window displays all totals in the Summary pane and status icons in the test tree regardless of the values set in th ecurrent configuration.

- **Example 2 -** A test run had failed tests and you want to use an Exclude List in the next run that excludes the failed tests from the test run. See Using Exclude Lists for a detailed description of specifying an exclude list. The Test Tree and the Summary pane do not change.

When you repeat the test run, the Test Manager window displays current status icons in the test tree and totals in the Summary pane for every test in the work directory, including results for any tests excluded from the current test run.

Last Test Run View Filter

When you select the Last Test Run view filter, the Test Manager window displays the current totals in the Summary pane and status icons in the test tree for all folders and tests included in the last test run even if you have exited the JavaTest harness since the last test run. The information displayed in the Test Manager window is associated with the work directory.

The following examples provide descriptions of the use of the Last Test Run view filter:

- **Example 1 -** A test run had failed tests and you want to rerun only the failed tests. See Using Prior Status for a detailed description of setting the prior status value. The Test Tree and the Summary pane continue to show all tests in the last test run regardless of their status.

When you repeat the test run, the Test Manager window clears the previous test results and only displays current status icons in the test tree and totals in the Summary pane for the tests and folders in the current test run.

- **Example 2 -** A test run had failed tests and you want to use an Exclude List in the next run that excludes the failed tests from the test run. See Using Exclude Lists for a detailed description of specifying an exclude list. The Test Tree and the Summary pane do not change when the exclude list is added to the configuration.

When you repeat the test run, the Test Manager window clears the previous test results and only displays current status icons in the test tree and totals in the Summary pane for the tests and folders in the current test run.

Current Configuration View Filter

When you select the Current Configuration view filter, the Test Manager window only displays the status of the folders and tests currently selected by the current configuration.

Select the Current Configuration view filter by either choosing View -> Filters -> Current Configuration from the menu bar or choosing the Current Configuration view filter from the toolbar.

The following examples provide descriptions of the use of the Current Configuration view filter:

- **Example 1 -** You only want to run and view the results for tests in the `api` folder. See Specifying Tests to Run for detailed description of specifying the tests that are run.

Either choose View **->** Filters **->** Current Configuration from the menu bar or choose the Current Configuration view filter from the toolbar.

Immediately the Test Manager window displays gray folder and test icons for all folders and tests except those under `api`. The Test Manager window changes the Summary totals to indicate that these tests are filtered out.

The current result icons are only displayed in the test tree for those folder and test icons under `api` and the Summary pane is updated to display their totals. When you run the tests, the test tree and folder view only display the results for those folders and tests under `api`.

- **Example 2 -** You only want to run and view results for failed tests. See Using Prior Status for detailed description of setting the prior status value.

Either choose View -> Filters -> Current Configuration from the menu bar or choose the Current Configuration view filter from the toolbar.

Immediately the Test Manager window displays gray folder and test icons for all folders and tests except those with Failed test results. The Test Manager window updates the Summary totals to indicate that these tests are filtered out and changes the Passed, Error, and Not Run totals to indicate 0.

The Test Manager window displays those folders and tests in the test tree with Failed test results as red folder and test icons. The Test Manager window updates the Summary information to indicate the number of tests that have Failed test results.

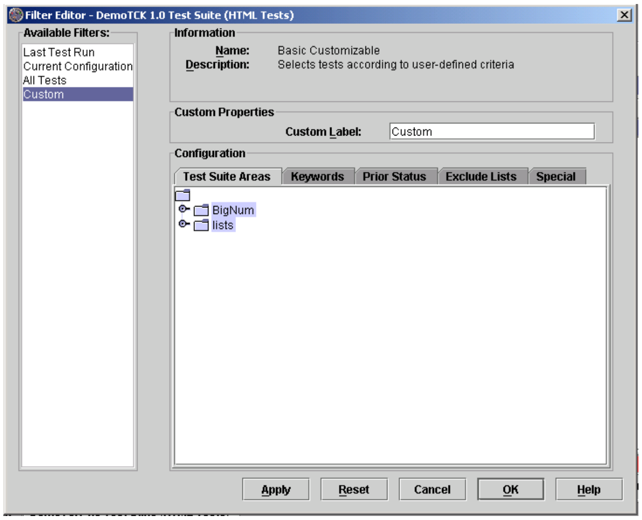
When you repeat the test run, as failed tests pass, the Test Manager window updates their icons from red folder and test icons to gray folder and test icons (they no longer match the configuration criteria of the view filter). The Test Manager window updates the totals in the Summary pane to match the results displayed in the test tree.

Custom View Filter

When the All Test, Last Test Run, and Current Configuration view filters do not provide a suitable view of the test tree, you can use the Filter Editor to create a Custom filter for the test suite.

The Custom filter is unique to the test suite. When the JavaTest harness opens the test suite in the Test Manager window, it restores the Custom filter, including any name that you assigned it.

You can also use the Custom filter to generate reports. See Creating New Reports for a description of using the Custom filter when generating reports.



Name: Basic Customizable
Description: Selects tests according to user-defined criteria

Custom Label: Custom

Edit the Custom Filter

Perform the following actions to edit the Custom filter:

1. Choose View -> Filters -> Configure Filters from the menu bar to open the Filter Editor. You can also choose Custom and then Edit Filter in the toolbar.
2. Choose Custom in the Available Filters panel. You can provide a name for the filter in the Custom Label field. The name is applied to the filter and restored each time the test suite is opened.
3. Use the Test Suite Areas, Keywords, Prior Status, Exclude Lists, and Special tabs to set the following view filter properties:
   - Specify Tests to View
   - Use the Exclude List as a Filter
   - Use Keywords as a Filter
   - Use Prior Status as a Filter
   - Use Special Settings as a Filter
4. Click one of the following buttons:
   - **Apply** - saves but does not dismiss the dialog box. Updates the GUI if the filter is selected.
   - **Reset** - discards all changes and restores the last saved Custom filter.
   - **Cancel** - closes the dialog box without saving any changes.
   - **OK** - saves the current changes, updates the Custom filter, and closes the dialog box.
   - **Help** - displays online help for the Filter Editor.

Using Test Suite Areas, Keywords, Prior Status, Exclude Lists, and Special settings in the view filter does not stop the JavaTest harness from running these tests. To filter the tests that are run, you must change the values in the configuration. See Changing Configuration Values for a detailed description.

Specify Tests to View

Click the Test Suite Areas tab and use the tree to choose the results of test folders or individual tests that you want displayed in the Test Tree. The JavaTest harness walks the test tree starting with the sub-branches and tests you specify and displays the results of all tests that it finds.

Use Keywords as a View Filter

If your test suite provides keywords, you can use the Keywords pane to restrict the set of test results displayed in the test tree and in the Summary pane.

To specify the keywords:

1. Click Match.

The JavaTest harness enables the Expression button.

1. Click Expression to display a list of expressions that can be constructed.
2. From the list, choose the type of expression that you are building.
3. In the text field, enter the keywords and operators used in the expression.

The following table provides descriptions and examples of keyword expressions that can be constructed.

TABLE 45 Keyword Expressions

Expression	Description
`Any Of`	Displays all tests in the test suite having *any of* the keywords entered in the text field. Example: A test suite uses the keyword `interactive` to identify tests that require human interaction, and `color` to identify tests that require a color display. To display only the results for tests containing the `interactive` keyword, choose `Any Of` and then use the `interactive` keyword.
`All Of`	Displays results for all tests in the test suite having *all of* the keywords entered in the text field. Example: To display results for only the tests containing both the `interactive` and `color` keywords, choose `All Of` and then use the `interactive` and `color` keywords.
`Expression`	Displays results for all tests in the test suite having the *expression* entered in the text field. Construct a Boolean expression in the text field. Keywords stand as Boolean predicates that are true if, and only if, the keyword is present in the test being considered. A test is accepted if the overall value of the expression is true; all other tests are rejected by the restriction. Example: A test suite uses the keyword `interactive` to identify tests that require human interaction, and `color` to identify tests that require a color display. To display results for only the tests with the `color` keyword that do not also contain the `interactive` keyword, choose `Expression` and then use the `color` keyword, the `!` operator, and the `interactive` keyword.

Use Prior Status as a View Filter

Click the Prior Status tab and choose the test results from the previous test run that you want displayed in the Test Tree.

TABLE 46 Using Prior Status as a Filter

Prior Status	Action
Passed	Displays tests that passed the last time the test was executed.
Failed	Displays tests that failed the last time the test was executed.
Error	Displays tests that the JavaTest harness could not execute the last time it was included in a test run.
Not Run	Displays tests without results in the current work directory.

Prior status is evaluated on a test-by-test basis using information stored in the result files (`.jtr`) that are written in the work directory. Unless overridden by a test suite, a result file is written in the work directory for every test that is executed. When you change work directories between test runs, the result files in the previous work directory are no longer used and, if the new work directory is empty, the JavaTest harness behaves as though the test suite was not run.

Use the Exclude List as a View Filter

To use the exclude list specified in the configuration interview as a filter, click the Exclude Lists tab and the Use settings in interview check box. The Exclude Lists pane displays the name of the exclude list file used by the current configuration.

Any test in the exclude list is filtered out and displayed as a 🗋icon in the test tree.

Use Special Settings as a Filter

If the test suite architect provides a default filter for the test suite, click the Special tab and the Enable test suite filter check box to select his filter.

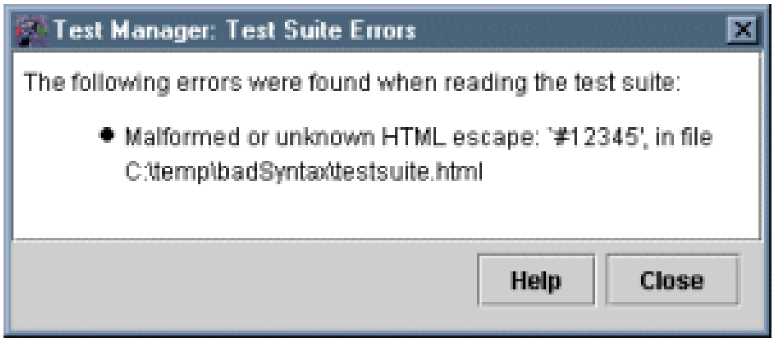
You must select this setting to correctly simulate the Current Configuration settings.

Use a Custom View Filter

To use a custom filter, you must choose it from the list of view filters below the test tree or from the View -> Filters menu. The Test Manager window updates the status of the folders and tests in the test tree that match the filter settings of the custom filter.

Viewing Test Suite Errors

The JavaTest harness only enables the View **->** Test Suite Errors menu item when errors are detected in the test suite. The JavaTest harness displays the Test Suite Errors dialog box when you choose View **->** Test Suite Errors from the Test Manager menu bar.



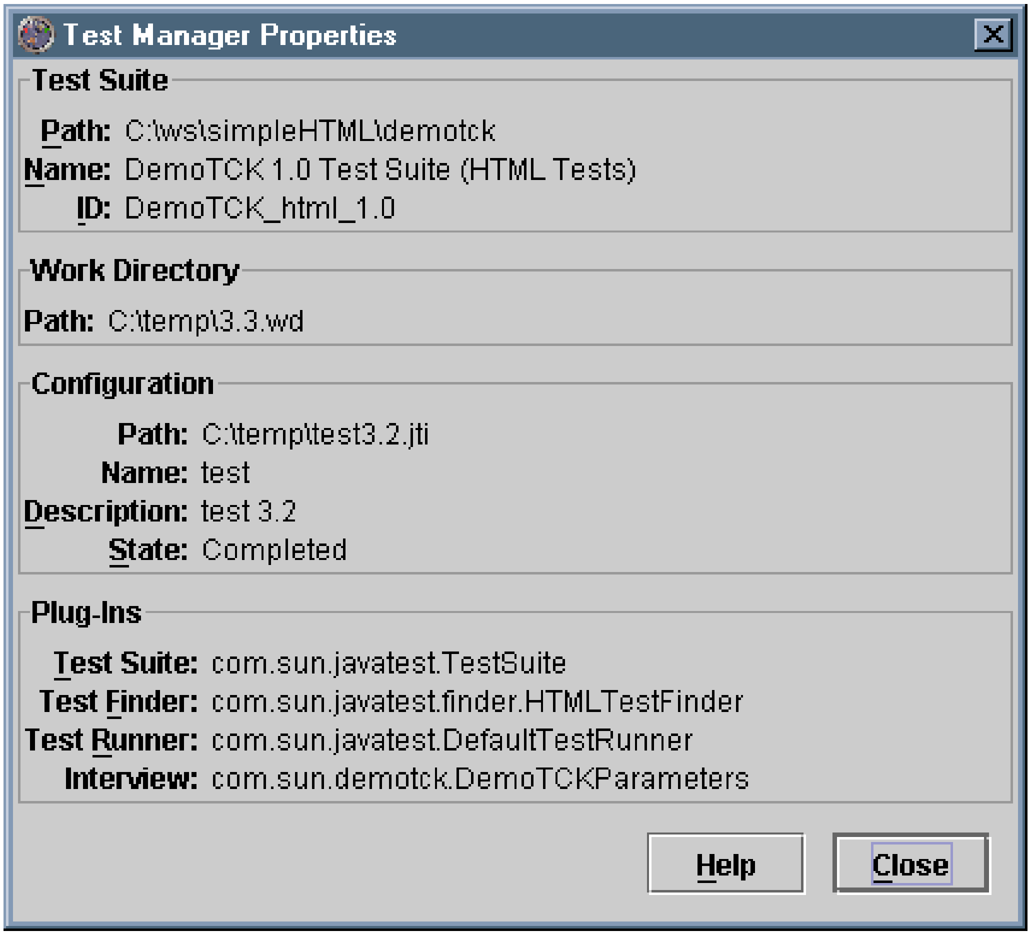
The following errors were found when reading the test suite:

- Malformed or unknown HTML escape: `#12345', in file C:\temp\badSyntax\testsuite.html

The dialog box displays a list of errors detected in the test suite. Unless instructed otherwise, report any test suite errors to the owner of the test suite.

Viewing Test Manager Properties

To view the properties of a test manager, choose View -> Properties. The JavaTest harness opens the Test Manager Properties browser.



Path: C:\ws\simpleHTML\demotck
Name: DemoTCK 1.0 Test Suite (HTML Tests)
ID: DemoTCK_html_1.0

Path: C:\temp\3.3.wd

Path: C:\temp\test3.2.jti
Name: test
Description: test 3.2
State: Completed

Test Suite: com.sun.javatest.TestSuite
Test Finder: com.sun.javatest.finder.HTMLTestFinder
Test Runner: com.sun.javatest.DefaultTestRunner
Interview: com.sun.demotck.DemoTCKParameters

The Test Manager Properties browser contains the following four areas:

- Test Suite
- Work Directory
- Configuration
- Plug-Ins

Test Suite

The Test Suite properties area displays the Path, Name, and ID of the current test suite opened by the test manager.

Work Directory

The Work Directory properties area displays the path of the current work directory opened by the test manager.

Configuration

The Configuration properties area displays the Path, Name, Description, and State of the current configuration interview opened by the test manager. The State field indicates whether the configuration is complete and tests can be run. It also identifies the availability of special filters.

Plug-Ins

The Plug-Ins properties area displays the name of the plug-ins used by the Test Manager. The plug-ins are provided by the test suite architect. The following table describes the properties that might be identified in the Plug-Ins properties area.

TABLE 47 Plug-Ins Area

Property	Description
Test Suite	The fully qualified name of the test suite class used by the test manager.
Test Finder	The fully qualified name of the test finder class used by the test manager.
Test Runner	The fully qualified name of the test runner class used by the test manager.
Interview	The fully qualified name of the interview class used by the test manager.

9

Test Reports

The JavaTest harness does not automatically generate reports at the end of a test run. Before viewing a report of your test run, you must use the JavaTest harness to generate a new report. You can generate and view reports containing the following test run information:

- Tests grouped by test status
- Configuration interview questions and answers
- Test environment used for the test run

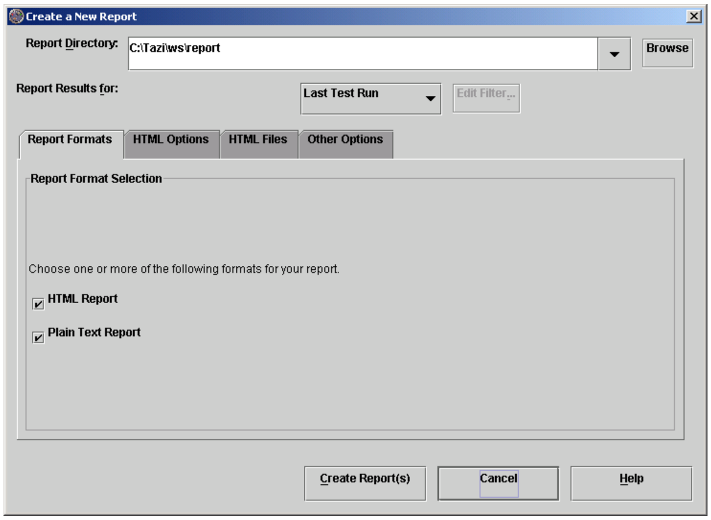
See Creating New Reports for a description of how to generate test reports.

To view reports in the JavaTest harness Report Browser, choose Report **->** Open Report from the menu bar. See Displaying Reports for a description of how to view reports.

Because JavaTest harness reports contain relative and fixed links to other files, you must update these links when moving reports to other directories. The JavaTest harness provides a command-line utility for you to use when moving reports to other directories. See Moving Test Reports in the *JavaTest Harness User's Guide: Command-Line Interface* for a description of how to use the `EditLinks` utility to move reports.

Creating Reports

The JavaTest harness does not automatically create reports of test results after a test run. You can create test reports either from the command line in batch mode (see Writing Reports in the *JavaTest Harness User's Guide: Command-Line Interface*) or from the JavaTest harness GUI. To create a test report from the JavaTest harness GUI, you must complete these steps:



Report Directory: C:\Tazi\ws\report

Report Results for: Last Test Run

Choose one or more of the following formats for your report.

☑ HTML Report

☑ Plain Text Report

1. Choose Report -> Create Report from the Test Manager menu bar.

The JavaTest harness opens the Create a New Report dialog box.

1. Type the name of a report directory in the Report Directory field or click the Browse button to specify where to put the new reports.

   You can either specify a new directory or an existing directory. If you ran reports earlier, it displays the directory from the previous run. If you use an existing report directory, the JavaTest harness can save the previous reports as backups when it writes the new reports. Use the settings in the Other Options tab to backup old reports and specify the number of backup reports to keep in the report directory.

   If you only generate a text report without html reports in an existing directory containing html reports, you must use an external text editor or web browser to view the `summary.txt` file. To use the Report Browser when viewing text reports, generate the text only report either in a new report directory or in an existing report directory that only contains plain text reports.

   The Report Results for field specifies the filter used to select the test results that are reported. You can choose Last Test Run, Current Configuration, All Tests, *Custom*, or Certification. If the *Custom* filter was previously renamed, the dialog box displays the current name of the custom filter. You can create or modify a custom filter for generating test reports for a specific set of test criteria.

   See Using View Filters for a description of the filters.

   See Custom View Filter for a description of how to create a custom view filter.

2. Use the tabbed panes to set printing formats and options. See Report Formats, HTML Options, HTML Files, and Other Options for a description of the formats and options that you can set.

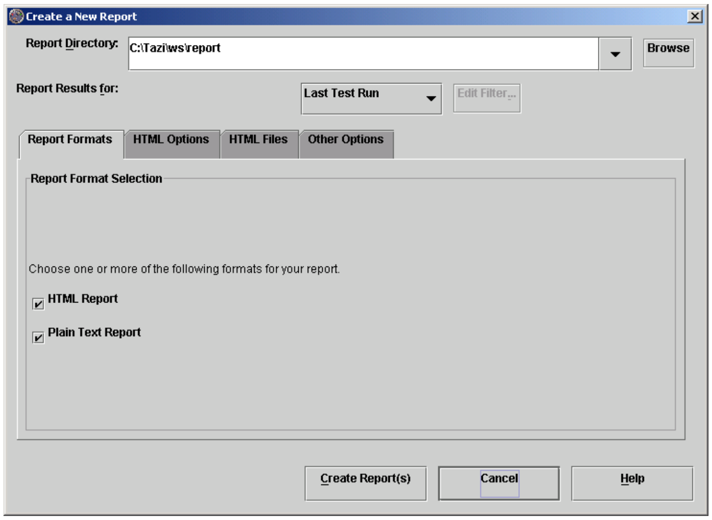
3. Click the Create Report(s) button.

The JavaTest harness writes the reports and displays a dialog box that gives you the option of either viewing the new reports in the report browser or returning to the Test Manager window.

The options set in this dialog are persistent. Once the options are set they are used each time a report is generated. These settings also apply to reports printed from the command-line interface.

Report Formats

In the Report Formats tab, choose the report formats generated by the JavaTest harness. If you choose to generate only the plain text report format in a report directory containing an existing HTML report, you must use an external text editor or web browser to view the summary.txt file. To use the Report Browser when viewing text reports, generate the text only report either in a new report directory or in an existing report directory that only contains plain text reports.



Report Directory: C:\Tazi\ws\report

Report Results for: Last Test Run

Report Format Selection

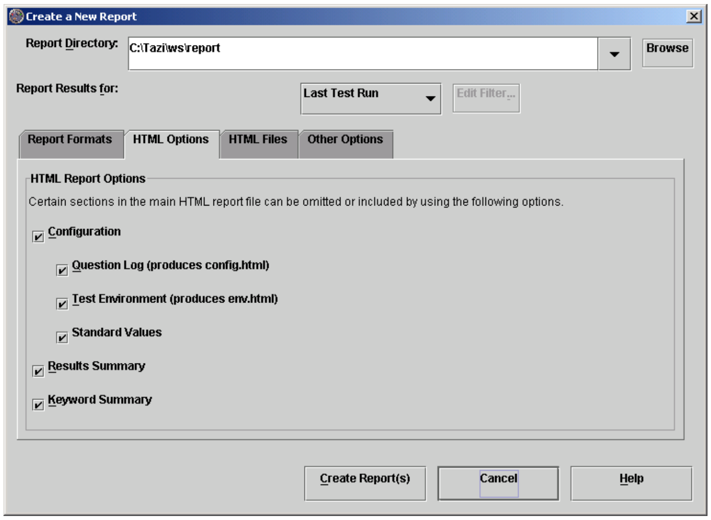
Choose one or more of the following formats for your report.

☑ HTML Report

☑ Plain Text Report

HTML Options

If you selected HTML Report in the Report Formats tab, use the HTML Options tab to select the sections of the main HTML report file that are generated.



Report Directory: C:\Tazi\ws\report

Report Results for: Last Test Run

HTML Report Options

Certain sections in the main HTML report file can be omitted or included by using the following options.

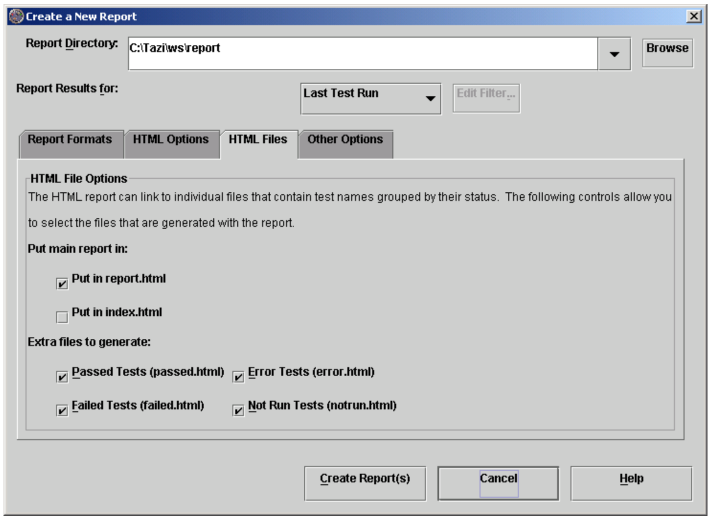
- ☑ Configuration
  - ☑ Question Log (produces config.html)
  - ☑ Test Environment (produces env.html)
  - ☑ Standard Values
- ☑ Results Summary
- ☑ Keyword Summary

The following options are available for generating HTML reports:

- **Configuration** - Selecting this option enables all of the following subordinate options:
- **Question Log** - Generates a report that is the equivalent of the Configuration Question Log.
- **Test Environment** - Generates a report that is the equivalent of the Show Environment dialog box.
- **Standard Values** - Generates a report that contains the standard values from the Quick set mode of the Configuration Editor window.
- **Results Summary** - Generates a report of the pass, Fail, Error, Not Run, and Total values. The HTML report provides hyperlinks to content in the other HTML files. If it is not selected the hyperlinks are not generated.
- **Keyword Summary** - Generates a report that provides a count of the number of occurrences of keywords that appeared in the selected tests.

HTML Files

If the HTML report format is selected, in the HTML Files tab, choose the main report location and choose the report files that you want generated.



Report Directory: C:\Tazi\ws\report

Report Results for: Last Test Run

HTML File Options

The HTML report can link to individual files that contain test names grouped by their status. The following controls allow you to select the files that are generated with the report.

Put main report in:

☑ Put in report.html

☐ Put in index.html

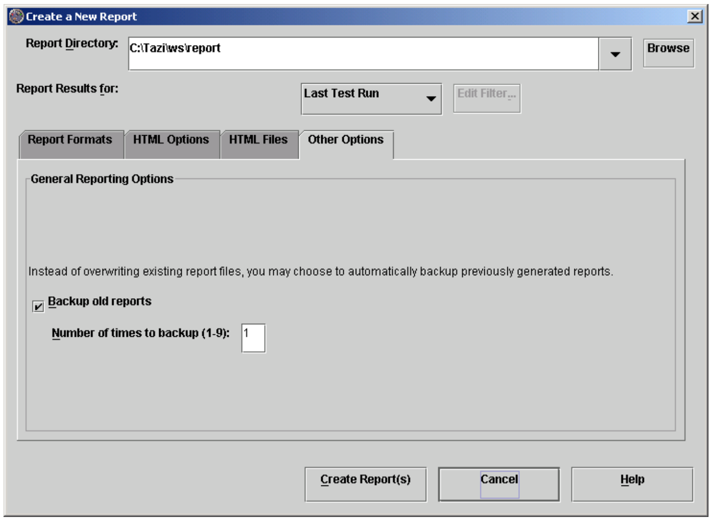
Extra files to generate:

☑ Passed Tests (passed.html) ☑ Error Tests (error.html)

☑ Failed Tests (failed.html) ☑ Not Run Tests (notrun.html)

Other Options

In the Other Options tab, you can choose to backup the previous report files. You can also choose the number of report backups that are maintained by the JavaTest harness.



Report Directory: C:\Tazi\ws\report

Report Results for: Last Test Run

General Reporting Options

Instead of overwriting existing report files, you may choose to automatically backup previously generated reports.

☑ Backup old reports

Number of times to backup (1-9): 1

If the Backup old reports option is selected, the JavaTest harness saves the previous reports as backups by appending a sequential numeric to the html extension (such as `failed.html~1~`). The JavaTest harness maintains the specified number of copies by deleting the oldest copy when the limit of backups is reached for a specific report. Changing this or any other setting in the report dialog box does not alter any previously saved backup reports. Existing backups are not deleted if backups are turned off and file names are not changed as old backups are deleted.

Displaying Reports

You can use the Report browser or a web browser to display JavaTest harness reports.

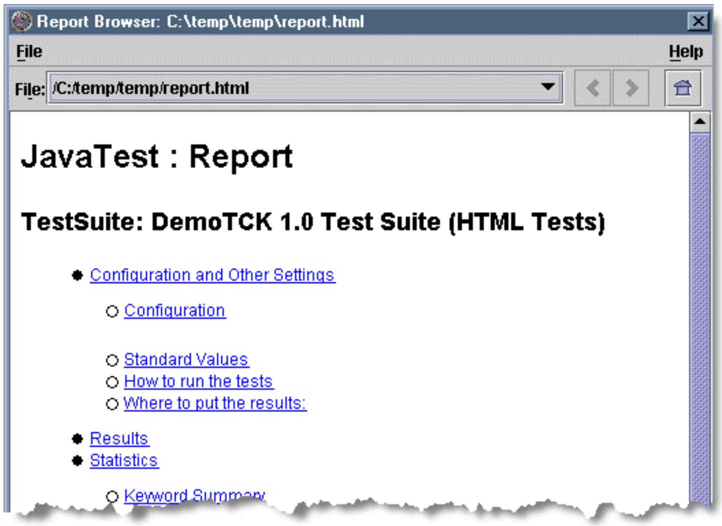
If you choose to only generate a text report in an existing directory containing HTML reports, you must use an external text editor or web browser to view the `summary.txt` file. To use the Report Browser when viewing text reports, generate the text only report either in a new report directory or in an existing report directory that only contains plain text reports.

Display Reports in the Report Browser

To display reports in the report browser, choose Report **->** Open Report from the menu bar. The JavaTest harness opens a file chooser dialog box for you to specify a report directory. When you specify a report directory, the JavaTest harness opens the Report Browser and displays the `report.html` file in that directory. If the report directory only contains the text version of the report, the browser will display the contents of the `summary.txt` file.

If you choose to display the `summary.txt` file the browser displays a hyperlink to the summary page. The summary page provides a text list of tests that were run and their test result status.

If you choose to display the HTML reports, a more sophisticated browser and set of reports are provided.


Report Browser: C:\temp\temp\report.html
File
Help
File:
/C:/temp/temp/report.html

JavaTest : Report

TestSuite: DemoTCK 1.0 Test Suite (HTML Tests)

- Configuration and Other Settings
    - Configuration
    - Standard Values
    - How to run the tests
    - Where to put the results:
- Results
- Statistics
    - Keyword Summary

The following table describes the contents of the Report Browser.

TABLE 48 Report Browser Contents

Component	Description
Menu bar	The menu bar contains a File and a Help menu. • Use the File menu to generate new reports, open existing reports, and to close the Report Browser. • Use the Help menu to display Report Browser online help.
toolbar	The toolbar contains a File field and three navigation buttons. The File field displays the name of the current report and provides a drop-down list of reports previously opened in the browser. As reports are opened, the JavaTest harness adds their names to the drop-down list enabling you to navigate to any previously displayed report. ‹ Returns to the previously displayed report page. › Opens the next report page that was displayed. ⌂ Returns to the `report.html` file. The `report.html` file is the root page and links to all of the other test report pages.
Contents area	The Report Browser displays the report file contents in the area below the toolbar. The Report Browser displays text files as well as HTML files. For HTML files, you can use hyperlinks in the report to display additional related reports.

See Keyboard Access for a description of how the keyboard can be used to navigate the Report Browser.

Display Reports Offline

If you choose to display the reports offline, use a web browser to open the `report.html` file located in the appropriate report directory.

The `report.html` file is the root file and links to all of the other HTML reports. The following table describes the HTML files.

TABLE 49 View Reports Offline

Report Files	Description
`config.html`	Contains the configuration interview questions and your answers used for the test run.
`env.html`	Contains the name-value pairs derived from the configuration file that were used for the test run.
`error.html`	Contains a list of the tests that had errors and could not be run.
`excluded.html`	Contains a list of the tests that were excluded from the test run.
`failed.html`	Contains a list of the tests that were executed during the test run but failed.
`notRun.html`	Contains a list of all tests that were not excluded from the test run but were not run.
`passed.html`	Contains a list of the tests that were executed during the test run and passed.
`report.html`	Contains links to all of the HTML files and additional information.

The JavaTest harness also generates a `summary.txt` report that contains a list of all tests that were run, their test results, and their status messages. You can open the `summary.txt` file in any text editor.

10

Auditing Test Results

The JavaTest harness includes an audit tool that you can use to analyze the test results in a work directory. The audit tool verifies that all tests in a test suite ran correctly and identifies any audit categories in a test run with errors.

You can use the GUI or the command-line interface to audit a test run.

Auditing Test Results in the GUI

Choose Windows -> Open -> Test Results Auditor from the menu bar.

The JavaTest harness opens the Test Results Auditor window. If you are running the audit for the first time, the JavaTest harness also opens the Options dialog box for you.

Open the Options dialog box and specify the reference test suite, the work directory to audit, and the reference configuration file. See Setting Audit Options for a description of the Options dialog box.

Click the Start Audit button at the bottom of the Options dialog box.

When the JavaTest harness completes the audit, it displays the results in the Test Results Auditor window.

To repeat the audit, choose Audit -> Options from the menu bar to open the Options dialog box and click the Start Audit button.

To close the Test Results Auditor window, choose File -> Close from the menu bar.

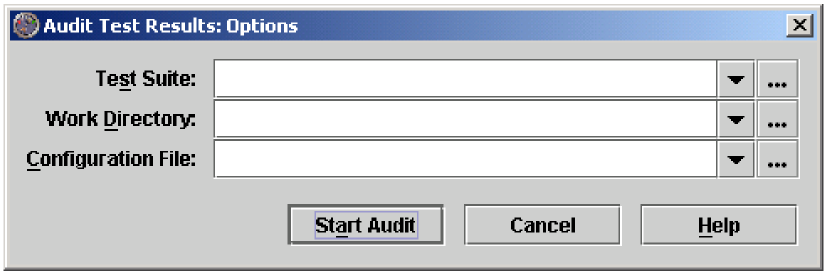
Auditing Test Results from the Command-Line Interface

See the *JavaTest Harness User's Guide: Command-Line Interface* for a detailed description of auditing a test run from the command line.

Setting Audit Options

Use the Options dialog box to specify the reference test suite, the audited work directory, the reference configuration, and to start the audit.

To display the Options dialog box, choose Audit -> Options from the menu bar.



The Options dialog box contains the following elements:

- Test Suite
- Work Directory
- Configuration File
- Start Audit button
- Cancel button
- Help button

Test Suite

You can use the drop-down list or the chooser to specify a test suite. You can also clear a previous entry in the Test Suite field by choosing the empty line from the drop-down list. A blank field indicates a test suite is not set.

Click the ▼ button to open the list of test suites currently loaded in the JavaTest harness. You are not limited to using these tests suites. You can either choose a test suite from the list or click the ... button to open the dialog box used to choose another test suite.

If you choose a reference test suite, the JavaTest harness sets the entries in the work directory drop-down list to the work directories that are currently loaded and matches the specified test suite. If you have multiple test suites and work directories, specifying a test suite can simplify choosing the options.

The JavaTest harness always uses the test suite associated with the work directory that you choose to audit. See Work Directory below for a description of how to choose a work directory to audit.

Work Directory

The JavaTest harness audits the work directory named in the Work Directory field. A blank field indicates the work directory is not set.

You can use the drop-down list or the chooser to specify the work directory to audit. You can also clear a previous entry in the work directory field by choosing the empty line in the drop-down list.

Click the ▼ button to open the list of work directories identified by the JavaTest harness. If you choose a reference test suite, the JavaTest harness only lists the work directories associated with it in the drop-down list.

You are not limited to using these work directories. You can either choose a work directory from the list or click the ... button to open the dialog box used to choose another work directory.

If you choose a work directory, the JavaTest harness uses the test suite associated with the work directory and sets the entries in the Configuration File drop-down list to those most recently used with the work directory.

If you do not choose a work directory, the JavaTest harness uses the work directory associated with the configuration file that you specify. See Configuration File below for a description of how to choose a reference configuration file.

Configuration File

You can use the drop-down list or the chooser to specify a reference configuration file. You can also clear a previous file from the Configuration File field by choosing the empty line in the drop-down list.

A blank field indicates that the default configuration file for the chosen work directory is used. If a work directory is not chosen, you can choose a reference configuration, and the JavaTest harness opens its work directory.

Click the ▼ button to open the list of configuration files identified by the JavaTest harness. The JavaTest harness lists the configuration files associated with the work directory.

Choose a file from the list or click the ... button to open the dialog box used in choosing a configuration file.

If you specify a configuration file, it must be associated with the work directory. If the configuration file is not associated with the work directory, the JavaTest harness displays an error message without performing the audit.

Start Audit Button

After you set the audit options, click the Start Audit button to audit the work directory. The JavaTest harness closes the Option dialog box and displays a message in the Audit Test Results window that it is performing the audit.

Cancel Button

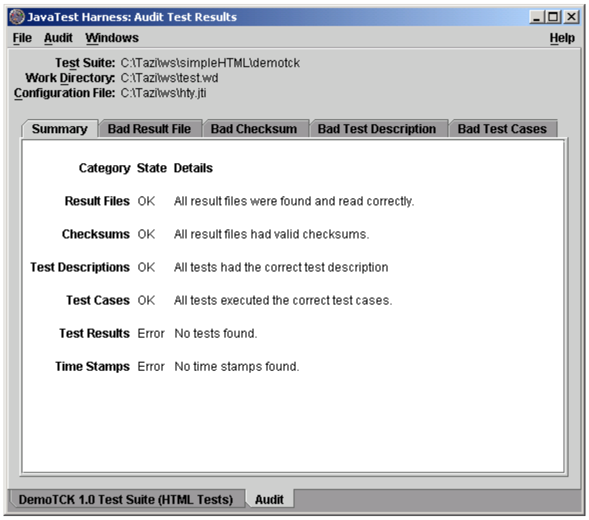
Closes the Options dialog box without accepting any changes to the option fields.

Help Button

Displays online help for the Options dialog box.

Audit Test Results Window

Use the Audit Test Results window to generate and view the audit report of the tests in a work directory. To display the Audit Test Results window, choose Windows -> Open -> Test Results Auditor from the menu bar.



Test Suite: C:\Tazi\ws\simpleHTML\demotck
Work Directory: C:\Tazi\ws\test.wd
Configuration File: C:\Tazi\ws\hty.jti

Category	State	Details
Result Files	OK	All result files were found and read correctly.
Checksums	OK	All result files had valid checksums.
Test Descriptions	OK	All tests had the correct test description
Test Cases	OK	All tests executed the correct test cases.
Test Results	Error	No tests found.
Time Stamps	Error	No time stamps found.

The following table describes the contents of the Test Results Auditor window.

TABLE 50 Test Results Auditor Window Contents

Component	Description
Menu bar	The Audit menu bar contains an Audit and a Help menu. The menu bar also contains JavaTest harness standard menus when the Tabbed and SDI window styles are used. See Change Window Styles for a description of the different styles of windows that the JavaTest harness provides. Use the Audit menu to set options used to generate an audit report. See Setting Audit Options for a description of the Options dialog box. Use the Help menu to display online help for the Test Results Auditor window. See JavaTest harness Menus for a description of the JavaTest harness standard menus that can also be displayed on the menu bar.
Text fields	The following three text fields are displayed at the top of the window: • **Test Suite** - Contains the name and location of the reference test suite. • **Work Directory** - Contains the name and location of the audited work directory. • **Configuration File** - Contains the name and location of the reference configuration interview.
Tabbed panels	The JavaTest harness displays the audit report in the following five tabbed panels: • Summary • Bad Result File • Bad Checksum • Bad Test Description • Bad Test Cases

Summary

The following table describes the audit category state and details displayed in the Summary panel.

TABLE 51 Summary Panel Contents

Category	Description
Result Files	Displays a summary of the audit for corrupted and missing test result files. See Bad Result File for a description of the detailed tab.
Checksums	Displays a summary of the audit of the test result files checksums. See Bad Checksum for a description of the detailed tab.
Test Descriptions	Displays a summary of the audit of the test descriptions. See Bad Test Description for a description of the detailed tab.
Test Cases	Displays a summary of the audit of the exclude list. See Bad Test Cases for a description of the detailed tab.
Test Results	Displays a summary of the test results audit. To pass the audit, all tests must have passing results.
Time Stamps	Displays the date and time when the first and the last tests were run.

Bad Result File

The Bad Result File panel lists any corrupted or missing test result files.

A corrupted or missing test result file indicates that the result file was edited.

Bad Checksum

The Bad Checksum panel lists all test result files with invalid checksums.

The result file checksums must match the reference checksums in the reference test suite. An invalid checksum indicates that a result file was edited.

Bad Test Description

The Bad Test Description panel lists all test files having edited test descriptions.

Test descriptions must match the reference test descriptions in the reference test suite. An invalid test description indicates that the test file was edited.

Bad Test Cases

The Bad Test Cases panel lists all tests failing to execute the required test cases.

The test cases run must match the exclude list for the reference test suite. Tests failing to execute the required test cases indicates that the exclude list was edited.

11

Troubleshooting

The JavaTest harness provides information you can use in troubleshooting problems. To troubleshoot problems using the JavaTest harness see the following topics:

- Problems Using the JavaTest Harness
- Problems Running Tests
- Problems Viewing Test Results
- Output Overflow Message Displayed in Test Run Messages
- Problems Viewing Reports
- Problems Writing Reports
- Problems Moving Reports

Problems Using the JavaTest Harness

If the JavaTest harness fails, you can use the `harness.trace` file in your work directory to help troubleshoot the problem. The `harness.trace` file is a plain-text file that contains a log of JavaTest harness activities during the test run. It is written in the work directory, is incrementally updated, and is intended primarily as a log of JavaTest harness activity.

Problems Running Tests

The goal of a test run is for all tests in the test suite that are not filtered out to have passing results.

If the root test suite folder contains tests with errors or failing results, you must troubleshoot and correct the cause to satisfactorily complete the test run. See Troubleshooting a Test Run for information about the resources that the JavaTest harness provides for troubleshooting.

Tests with Errors

Tests with errors are tests that could not be executed by the JavaTest harness. These errors usually occur because the test environment is not properly configured. Use the Test tabbed panes and the Configuration Editor to help determine the change required in the configuration.

The following is an example of how the Test Manager tabbed panes and the Configuration Editor can be used to identify and correct a configuration error:

1. Use the test tree to identify the folder containing test that had errors.
2. Click the folder icon to open its Summary tab in the Test Manager window.
3. Click the Error tab to display the list of tests in the folder that had errors.
4. Double-click a test in the list to display it in the test tree and view its detailed test information.
5. Click the Test Run Messages tab to display detailed messages describing what happened during the running of each section of the test. The contents of each output section vary from test suite to test suite. Refer to your test suite documentation for detailed descriptions of the test section messages when troubleshooting a test run.
6. Click the Configuration tab to display a two-column table of the name-value pairs that were derived from the configuration file and actually used to run the test. The names in the table identify test environment properties used by the JavaTest harness to run the test. The values displayed were used to run the test. Refer to your test suite documentation for detailed descriptions of the name-value pairs for your test.
7. Choose Configure -> Show Question Log to view the Question Log of the current, saved configuration. Use the question log to identify the configuration value that is incorrect and its configuration question.
8. Choose Configure -> Change Configuration -> Other Values from the menu bar or click either configuration editor button on the toolbar to open the Configuration Editor window.
9. Search the configuration file for the specific characters or character strings that must be changed. See Searching the Current Configuration for a detailed description of how the current configuration can be searched for a character or string of characters.

10. Click the Done button to save your changes to the configuration file
11. Rerun the tests.

Tests that Fail

Tests that fail are tests that were executed but had failing results. The test or the implementation might have errors.

The following is an example of how the Test Manager tabbed panes can be used to identify and correct a test failure:

1. Use the test tree to identify the folder containing test that had errors.
2. Click the folder icon to open its Summary tab in the Test Manager window.
3. Click the Error tab to display the list of tests in the folder that had errors.
4. Double-click a test in the list to display it in the test tree and view its detailed test information.
5. Click the Test Run Messages tab to display detailed messages describing what happened during the running of each section of the test. The contents of each output section vary from test suite to test suite. Refer to your test suite documentation for detailed descriptions of the test section messages when troubleshooting a test run.

Problems Viewing Test Results

Most problems in viewing test results in the Test Manager result from the use of view filters (other than the All Tests view filter) with a current configuration set to run specific tests (such as, running tests based on their prior status). When a view filter other than All Tests is used in the Test Manager, only the fields of the filtered category appear to be updated in the Test Manager Summary tab. This is normal behavior of the GUI when view filters are used.

The JavaTest harness *only* displays tests in the GUI that match the specified view filter criteria. All other tests are displayed as Filtered Out.

As test results change, the JavaTest harness moves the tests to the Filtered Out category (not to a test result category) in the Test Manager and turns the appropriate node in the test tree gray. Consequently, the fields for the other categories in the Test Manager Summary tab might not appear to be updated.

Example:
If you change the current configuration to run only tests with a prior status of Failed and use the Current Configuration view filter, the Test Manager displays all

tests that are not a Failed status as Filtered Out. This allows you to monitor only those tests with a previous Failed status. When you begin the test run, any tests with Passed or Error results are displayed in the Filtered Out category of the Test Manager. To view the actual test results, change to a different view filter. See Using View Filters for a detailed description of the view filters.

Output Overflow Message Displayed in Test Run Messages

If one of the output streams in the Test Run Messages Pane displays the following message, restart the JavaTest harness with the harness VM system property `JavaTest harness.maxOutputSize` value set to a greater number and rerun the test to ensure that all of the data is saved.

```
 Output overflow:JavaTest harness has limited the test output
of the text
 to that at the beginning and the end, so that you can see
how the
 test began, and how it completed.

 If you need to see more of the output from the test,
 set the system property JavaTest harness.maxOutputSize to a
higher
 value. The current value is 100000

 Set the system property of JavaTest harness.maxOutputSize by
using the syntax in
 java -DJavaTest harness.maxOutputSize=200000 -jar lib/
JavaTest harness.jar ....
```

The value of the system property sets the number of characters to accept.

The value of the system property is an integer subject to the maximum integer size of the VM.

Problems Viewing Reports

The JavaTest harness does not automatically generate reports of test results after a test run. You must generate test reports either from the command line or from the JavaTest harness GUI. See Creating Reports for detailed information.

Problems Writing Reports

Filters are used to write test reports for a specific set of test criteria. Verify that you are using the appropriate filter to generate reports of test results. See Creating Reports for detailed information.

Problems Moving Reports

Test reports contain relative and fixed links to other files that might be broken when you move reports to other directories.

You must update these links when moving reports to other directories. The JavaTest harness provides an `EditLinks` utility that updates the links in the reports for you when moving reports. See the *JavaTest Harness User's Guide: Command-Line Interface*.

Glossary

`.jte` Files

See environment files.

`.jti` Files

See configuration file.

`.jtp` Files

See parameter files.

`.jtr` File

See test result files.

`.jtx` Files

See exclude list.

A

All Tests View Filter

A test tree view filter that displays all folders and tests in the test suite.

Audit

The JavaTest harness includes an audit tool that you can use to analyze the test results in a work directory. The audit tool verifies that all tests in a test suite ran correctly and identifies any audit categories of a test run that had errors.

You can use the GUI or the command-line interface to audit a test run.

B

C

Class

The prototype for an object in an object-oriented language. A class might also be considered a set of objects that share a common structure and behavior. The structure of a class is determined by the class variables that represent the state of an object of that class and the behavior is given by a set of methods associated with the class.

Configuration

Information about the computing environment required to execute a test suite.

In the GUI, you can use the Configuration Editor to collect or modify configuration information or to load an existing configuration. See Configuration Editor. The Configuration Editor collects the following two types of data in an configuration file:

- Test environment
- Standard Values

In the command-line interface, you can perform the following tasks:

- Use the `EditJTI` utility to modify configuration information (see `EditJTI`).

- Set specific configuration values in the command line when starting the JavaTest harness.

Configuration Editor

The JavaTest harness Configuration Editor provides two editing modes, the Question mode and the Quick Set mode. Use the Question mode to create a configuration file for the test run and to search interview titles, questions, and answers for character strings. Use the Quick Set mode to modify specific runtime values.

To open the Configuration Editor, choose Configure -> Change Configuration from the Test Manager menu bar.

Configuration File

Contains all of the information collected by the configuration editor about the test platform.

The JavaTest harness derives the configuration values required to execute the test suite from environment entries in a configuration file (`.jti`).

Use the Configuration Editor or EditJTI to change configuration values in a `.jti` file.

You can also set specific values in the command line.

Configuration Value

A value specified by the user for the purpose of configuring a test run.

Configuration values are derived from environment entries in a configuration file and used by test suite specific plugin code to execute and run tests.

For test suites prior to the JavaTest harness, version 3.0, the configuration value is read from an environment file (`.jte`).

For JavaTest harness, version 3.0 (or later) test suites, the configuration value is derived from the configuration file (`.jti`).

Use the Configuration Editor or `EditJTI` to change the configuration values in the `.jti` file.

You can also set specific configuration values in the command line.

Current Configuration

The configuration containing the test environment and standard values currently loaded in the test manager or specified in the command line for use in running tests and displaying test status.

Current Configuration Filter

A filter that only displays or generates reports for the folders and tests that have not been filtered out in the configuration.

D

Desktop

The configuration and layout of the windows used by the JavaTest harness.

The desktop is saved when you exit from the harness and is automatically restored in your next session.

The JavaTest harness desktop is displayed in the following three user selectable styles:

- SDI
- MDI
- Tabbed

E

Environment

See Test Environment.

Environment Entry

A name-value pair derived from a configuration file and used by test suite specific plugin code to execute and run tests. These name-value pairs provide information (configuration values) about how to run tests of a test suite on a particular platform.

For test suites prior to the JavaTest harness, version 3.0, the name-value pairs are read from an environment file (`.jte`).

For JavaTest harness, version 3.0 (or later) test suites, the name-value pairs are derived from the configuration file (`.jti`).

Environment Files

Contain one or more test environments used by test suites prior to the JavaTest harness version 3.0. Environment files are identified by the `.jte` extension in the file name.

Error

The test is not filtered out and the JavaTest harness could not execute it. There are no test results for tests having errors. Errors usually occur because the test environment is not properly configured.

In the GUI, the JavaTest harness displays error icons for tests with errors and for folders containing any tests with errors. Folders marked with error icons can also contain tests and folders that are Failed, Not Run, Passed, and Filtered out.

Exclude List

Exclude list files (`*.jtx`), supply a list of invalid tests to be filtered out of a test run by the test harness. The exclude list provides a level playing field for all implementors by ensuring that when a test is determined to be invalid, then no implementation is required to pass it. Exclude lists are maintained by the technology specification Maintenance Lead and are made available to all technology licensees.

In the GUI, use the configuration editor to add or remove exclude lists from a test run. In the command line, you can specify an exclude list in the command.

To view the contents of an exclude list, choose Configure **->** Show Exclude List from the Test Manager menu bar. Exclude lists can only be edited or modified by the test suite Maintenance Lead.

F

Fail

Test results determined by the JavaTest harness that do not meet passing criteria.

In the GUI, the JavaTest harness displays Failed icons for tests that the test suite has determined have failing results and for folders containing any tests with fail results. Folders marked with Failed icons can also contain tests and folders that are Not Run, Passed, and Filtered out.

Filtered Out

Folders and their tests that are excluded from the test run by one or more test run filters.

In the GUI, Filtered Out folders and tests are identified in the test tree by grey 📁folder and 📄test icons.

Filters

A facility in the JavaTest harness that accepts or rejects tests based on a set of criteria. There are two types of filters in the JavaTest harness, view filters and run filters. View filters are set in the Test Manager to display the results for specific folders and tests and to create test reports. Run filters are set in the Configuration Editor or are specified as commands in the command-line to specify which tests are run.

G

Glossary

Use the Glossary tab in the left pane of the help viewer to open the glossary.

The glossary pane is divided into two areas. The top area displays the list of JavaTest harness terms and the bottom area displays their definition.

Glossary entries are listed in alphabetical order. You can scroll through the list or search for a specific term.

To search for a specific term, enter it in the Find field and press Return. If the term is in the glossary, the JavaTest harness highlights it in the list and displays its definition. Press Return to repeat the search.

H

I

Interview

For the JavaTest harness to execute a test suite, it requires information about how your computing environment is configured. The Configuration Editor uses an interview to simplify the process of collecting this information.

Because the quantity and scope of this information depends on the test suite, a test suite might include a specific interview for the Configuration Editor to use.

The information collected by the interview is written to a configuration file (`.jti`) and used by the JavaTest harness to derive the environment variables required execute the test suite.

Interview File

See configuration file.

J

JavaTest Harness Preferences

A dialog box that you can use to set the window style and tool tip options used by the JavaTest harness. To open the Preferences dialog box, choose File -> Preferences from the Test Manager menu bar.

JTI

Standard file extension for a configuration file. See configuration file.

K

Keywords

Special values in a test description that describe how the test is executed.

Keywords are provided by the test suite for use in the Configuration Editor or command line as a filter to exclude or include tests in a test run.

L

Last Test Run View Filter

A filter that displays the results of those tests and folders included in the last test run either in views of the test tree or in test reports.

M

MDI

See Multiple Document Interface.

Multiple Document Interface

A window style in which the JavaTest harness desktop is a single top-level window that contains all JavaTest harness windows opened to perform a task.

Use the Preferences dialog box to select the MDI window style. See JavaTest Harness Preferences.

N

Not Run

The test is not filtered out but the JavaTest harness has not yet run it.

The JavaTest harness displays Not Run icons for tests that have not *yet* been run. Folders containing tests that have not been run are displayed with the Not Run icon only when they do not also contain tests or folders with Error or Failed results.

O

P

Parameter Files

Files used prior to the JavaTest harness, version 3.0, to configure how the JavaTest harness runs the tests on your system. Parameter files have the file name extension of `.jtp`.

Use of parameter files is deprecated, however, the JavaTest harness provides support for those test suites that use parameter files.

Pass

Test results determined by the JavaTest harness to meet passing criteria.

The JavaTest harness displays Passed icons for tests that the test suite has determined have passing results and for folders containing only tests with passing results.

Port Number

A number assigned to the JavaTest harness that is used to link specific incoming data to an appropriate service.

Prior Status

A filter used to restrict the set of tests in a test run based on the last test result information stored in the test result files (`.jtr`) in the work directory.

Use the configuration editor or command line to enable the Prior Status filter for a test run.

Progress Monitor

A dialog box that displays detailed information about the current configuration of a test run. Information displayed in the Progress Monitor is not altered by view filter settings.

Q

R

Report Directory

The directory in which the JavaTest harness writes test reports.

The location of the report directory is set in the GUI or from the command line by the user when generating test reports.

S

SDI

See Single Document Interface.

Single Document Interface

A window style in which the JavaTest harness opens a console window and individual tool windows as separate top-level windows on an unbounded desktop.

Use the Preferences dialog box to select the SDI window style. See JavaTest Preferences.

Standard Values

The Quick Set mode of the Configuration Editor displays the standard values of a configuraion.

T

Tabbed

A window style in which the JavaTest harness desktop is a single top-level window that displays all JavaTest windows as tabbed panes.

Use the JavaTest Preferences dialog box to select the Tabbed window style. See JavaTest Preferences.

Test Description

Machine readable information that describes a test to the JavaTest harness so that it can correctly process and run the related test. The actual form and type of test description depends on the attributes of the test suite. When using the JavaTest harness, the test description is a set of test-suite-specific name-values pairs.

Each test in a test suite has a corresponding test description that is typically contained in an HTML file.

Test Environment

A collection of configuration values derived from environment entries in the configuration file that provide information used by test suite specific plugin code about how to execute and run each test on a particular platform.

When a test in a test suite is run, the JavaTest harness gives the script a test environment containing environment entries from configuration data collected by the configuration editor. See configuration.

Prior to the JavaTest harness, version 3.0, the environment entries were read from an environment file. Use of environment files is deprecated. However, the JavaTest harness continues to provide support for those test suites that use environment files. See environment file.

Test Manager

The JavaTest harness window used to configure, run, monitor, and manage tests from its panels, menus, and controls.

The Test Manager window is divided into two panes. It displays the folders and tests of a test suite in the tree pane on the left and provides information about the selected test or folder in the information panes on the right.

A new Test Manager window is used for each test suite that is opened.

Test Paths

The locations of the test folders and tests in the test tree specified in the Configuration Editor for running tests.

Test paths are used to specify the tests in a specific branch of the test tree or to specify a specific test to run. The JavaTest harness walks the test tree and runs all specified tests (unless otherwise filtered out) that it finds.

Use the configuration editor to set the test paths listed in the test run configuration.

Test Result Files

Contains all of the information gathered by the JavaTest harness during a test run.

The test result files (`.jtr`) are stored in a cache in the work directory associated with the test suite.

You can view the test result files in a web browser configured to use the JavaTest harness `ResultBrowser` servlet.

Test Run Filters

Include or exclude tests in a test run. Tests are included or excluded from test runs by the following means:

- Exclude lists
- Keywords
- Prior status

Test run filters are set using the Configuration Editor or the command-line interface.

Test Script

A script used by the JavaTest harness, responsible for running the tests and returning the status (pass, fail, error) to the harness. The test script must interpret the test description information returned to it by the test finder. The test script is a plug-in provided by the test suite. In the GUI, the Test Manager Properties dialog box lists the plug-ins that are provided by the test suite.

Test Suite

A collection of tests, used in conjunction with the JavaTest harness, to verify compliance of the licensee's implementation of the technology specifications.

A test suite must be associated with a work directory before the JavaTest harness can run its tests.

Test Tree

The hierarchical representation of the folders and tests in a test suite.

The test tree is displayed in the Test Manager window and uses colored status icons to indicate the test status of the folders and tests. Use view filters to specify the folders and tests whose test status are displayed in the test tree.

U

V

View Filters

Include or exclude the results of tests in views of the test tree.

View filter settings only change the Test Manager display and do not include or exclude tests from a test run. Use the test run filter settings to include or exclude tests from a test run.

W

Work Directory

A directory associated with a specific test suite and used by the JavaTest harness to store files containing information about the test suite and its tests.

Until a test suite is associated with a work directory, the JavaTest harness cannot run tests.

X

Y

Z

Index

Symbols
.jte files, 62
.jtx files, 56

A
accessing help, 7
Agent Monitor window, 81, 82, 83
agent pool, 82
agents currently used, 83
All Tests View Filter, 108
appearance preferences
  JavaTest desktop, 18
Audit Test Results window, 129
auditing test results, 125

B
Bad Checksum panel
  Test Results Auditor window, 132
Bad Result File panel
  Test Results Auditor window, 131
Bad Test Cases panel
  Test Results Auditor window, 132
Bad Test Description panel
  Test Results Auditor window, 132
Bookmarks, 66
browser
  exclude list, 72
    contents, 72
    test details, 73
  report, 123
  test environment, 70
    defined in environment, 72
    defined in file, 72
    Entry Name, 71
    Value, 72
  Test Manager Properties, 115
browsing test information, 86
button bar (same as toolbar), 37

C
changing configuration values, 53
changing window styles, 19
Clear Results
  test tree popup menu, 94
configuration
  creating a new, 47
  creating a template, 48
  test manager properties, 117
configuration checklist, 70
Configuration Editor
  Question Mode, 40
    configuration history, 42
    File menu, 41
    Help menu, 44
    index pane, 45
    More Info pane, 46
    question pane, 45
    Search menu, 43
    View menu, 44
  Question Mode view
    Keywords pane, 59
    setting test execution values, 64
  Standard Value, 46
  Test Environment pane, 62
Configuration Editor window
  Question Mode
    Bookmarks menu, 42
configuration file
  changing values, 53
Configure menu, 34
  configuration history, 35
configuring a test run, 38
creating a configuration, 47
creating a configuration template, 48
creating test reports, 118
Current Configuration view filter, 109

Custom view filter, 110
  edit, 111
  specify tests, 112
  use exclude lists, 114
  use keywords, 112
  use prior status, 113
  use test suite filter, 114
  using, 114

D

desktop
  GUI, 11
desktop menus, 13
displaying
  configuration checklist, 70
  online information, 26
  Question Log, 73
  test manager properties, 115
  test tree popup menu, 92
displaying reports, 123

E

environment files, 62
exclude list
  latest, 57
  other, 58
Exclude List browser, 72
Exclude List contents, 72
Exclude List files, 56
Exclude List test details, 73

F

file
  .jte, 62
File menu, 14
Files tab
  viewing, 102
filters
  browser
    view, 107
folder icons, 89
folder information
  Test Manager, 96
folder result status icon, 90
folder run status icon, 89

G

getting started, 10
GUI, 10
  desktop, 11

H

Help
  button, 27
  menu, 27
help
  accessing, 7
  F1 key, 27
  navigation, 8
Help menu, 16
help, context sensitive, 27

I

icons
  folder, 89
    result status, 90
    run status, 89
  test, 90
    result status, 91
    run status, 91
initial files, 53

J

JavaTest Desktop, 11
JavaTest desktop
  appearance preferences, 18
  window styles, 19
JavaTest GUI, 10
JavaTest online help, 6
JavaTest Preferences dialog box, 17

K

Keywords
  run filter, 59
Keywords pane
  Configuration Editor
    Question Mode view, 59

L

Last Test Run View Filter, 108
latest exclude list, 57

M

markers, 66
MDI window style, 20
menu
  Configure, 34
    configuration history, 35
  desktop
    File, 14
    Help, 16
    Windows, 16
  Help, 27
  test tree popup, 92
    clear previous test results, 94
    execute, 92
    refresh test suite contents, 93
menus
  desktop, 13
monitor, progress, 78
monitoring a test run, 76
multiple configurations, 69
Multi-Selection Tab, 100

N

navigating help, 8
number of
  failed tests, 97
  passed tests, 97
  tests filtered out, 98
  tests not run, 97
  tests selected to run, 98
  tests with errors, 97
number of tests
  total, 98

O

online help
  accessing, 7, 8
online information
  displaying, 26
options
  tool tip, 21

P

plug-ins
  test manager properties, 117
progress indicator, 78
  elapsed time, 78
  progress bar, 78
Progress Monitor, 78
  Memory, 81
  Progress area, 79
  Tests in Progress, 80
  Time, 81

Q

Question Log, 73
Question Mode view
  setting
    concurrency, 64
    time factor, 65
Quick Pick test execution
  test tree popup menu, 92
Quick Set Mode, 46
Quick Start Wizard, 9

R

Refresh Test Suite Contents
  test tree popup menu, 93
Report Browser, 123
reports
  test, 117
    creating, 118
run filter
  Keywords, 59
run tests
  start, 75
running tests
  Test Manager window, 73

S

SDI window style, 21
setting
  concurrency, 64
  test execution values, 64
  time factor, 65
setting JavaTest preferences, 17
site administration, 48
specify tests to run, 53
specifying a test environment, 62
standard values
  Tests pane, 53
starting a test run, 75

stopping a test run, 83
Summary information, 97
  number of test
    filtered out, 98
    not run, 97
    selected to run, 98
    total, 98
  number of tests
    errors, 97
    failed, 97
    passed, 97
Summary panel
  Test Results Auditor window, 131

T

Tabbed window style, 20
Test Description tab, 102
Test Environment browser, 70
  defined in environment, 72
  Defined in file, 72
  Entry Name, 71
  Value, 72
test icons, 90
test information
  Test view, 100
Test Manager
  Configuration tab, 102
  Error test results, 99
  Failed test results, 99
  Filtered Out tests, 99
  folder view, 96
  Not Run tests, 99
  Passed test results, 99
  Summary information, 97, 98
    number of failed tests, 97
    number of passed tests, 97
    number of tests not run, 97
    number of tests with errors, 97
    subtotal, 98
    test filtered out, 98
    total, 98
  Test Run Messages, 104
    message area, 106
    message list, 104
    Output Summary message, 105
    Result message, 105
    summary message, 105
    Test Result message, 106
  Test view, 100
test manager
  properties, 115
    configuration, 117
    plug-ins, 117
    test suite, 116
    work directory, 116
Test Manager window, 31
  browsing test information, 86
  progress indicator, 78
    elapsed time, 78
    progress bar, 78
  running tests, 73
  status message, 33
  Test Description tab, 102
  Test Run Details tab, 103
    property names, 103
    values, 104
  test tree, 87
test reports, 117
test result status icon, 91
test results
  auditing, 125
Test Results Auditor window
  Bad Checksum panel, 132
  Bad Result File panel, 131
  Bad Test Cases panel, 132
  Bad Test Description panel, 132
  Summary panel, 131
test run
  monitoring, 76
  start, 75
  stopping, 83
Test Run Messages, 104
  message area, 106
  message list, 104
  Output Summary message, 105
  Result message, 105
  summary message, 105
  Test Result message, 106
test run status icon, 91
test status
  Error, 99
  Failed, 99
  Filtered Out, 99
  Not Run, 99
  Passed, 99
  Summary information, 98
test suite
  test manager properties, 116
Test Suite Errors
  viewing, 115

test tree, 87
Test view, 100
tests
    running, 73
tool bar, 37
tool tip options, 21
troubleshooting, 132
    output overflow
        , 136
    reports
        moving, 137
        viewing, 136
        writing, 137
    running tests, 133
        tests that fail, 135
        tests with errors, 134
    running the JavaTest harness, 133
    viewing test results, 135
troubleshooting a test run, 84

V

view filter
    All Tests, 108
    Current Configuration, 109
    Custom, 110
    Last Run, 108
view filters, 107
viewing source file list, 102
viewing Test Suite Errors, 115

W

window
    Audit Test Results, 129
    MDI style, 20
    SDI style, 21
    Tabbed style, 20
    Test Manager, 31
        status message, 33
window, Agent Monitor, 81
Windows menu, 16
work directory
    test manager properties, 116