

Adding Projects, .tst files, and Test Suites

This topic provides a general guide on how to add projects, .tst files and test suites using SOAtest's various test creation wizards.

Sections include:

- [Projects, .tst files, and Test Suites](#)
- [Test Suites and Scenarios](#)
- [Creating an Empty Project](#)
- [Adding a New .tst File to an Existing Project](#)
- [Adding a New Test Suite](#)
- [Using Eclipse Java Projects in SOAtest](#)

Wizard Descriptions

For a description of the various wizards available for adding projects, .tst files, and test suites, see [Wizards for Creating Projects, .tst Files, and Tests](#).

Projects, .tst files, and Test Suites

A project (an entity created by Eclipse) can contain any number of SOAtest-specific .tst files. They can also contain source files you want to analyze with SOAtest, and any other resources that make sense for your environment.

Each .tst file can include any number of test suites/scenarios, tools, and inputs. The organization and structure is up to you. To keep file size down and to improve maintainability, we recommend using one .tst file for each distinct testing requirement.

For best practices related to projects, test files, and workspaces, see [Workspaces, Projects, and Test Files](#).

Test Suites and Scenarios

A test suite is any collection of tests that are individually runnable, and has the following setting in the test suite configuration panel:

▼ Test Relationship

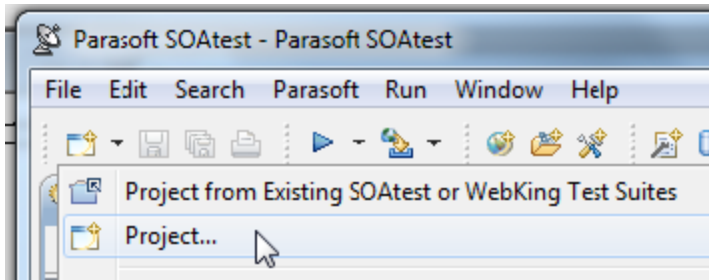
- Tests are individually runnable
Data source is iterated over once per test
- Tests run as group
Data source is iterated over once per group
- Tests run all sub-groups as part of this group
Data source is iterated over sub-groups as part of this group

A scenario is any collection of tests that are not individually runnable because they have dependencies. One example of a scenario is when a series of API tests extracts a value from one test's response and uses it as part of subsequent test message. Another example is a sequence of web scenarios recorded from a browser.

Creating an Empty Project

You can create an empty project as follows:

1. Open the pull-down menu for the **New** toolbar button (top left) then choose **Project**.

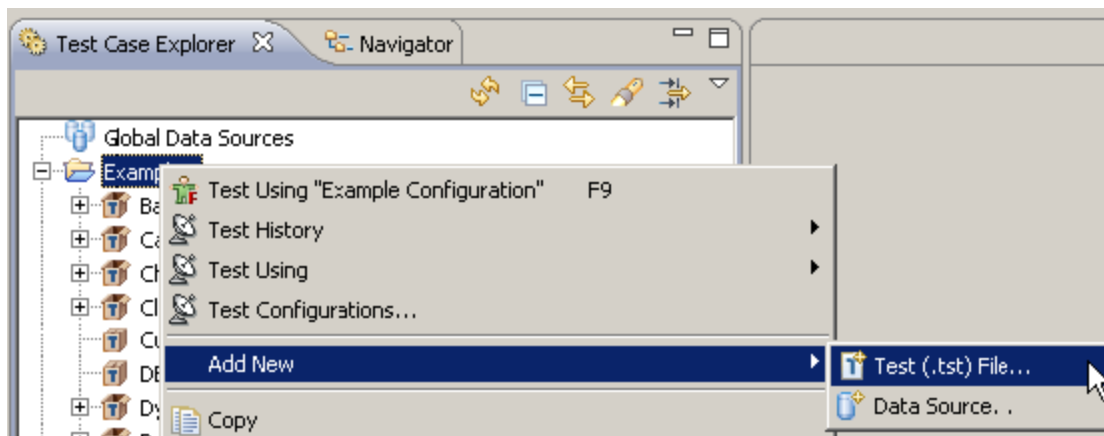


2. Choose **SOAtest > Empty Project**, then click **Next**.
3. Enter a name for the project, change the destination if needed, then click **Finish**.

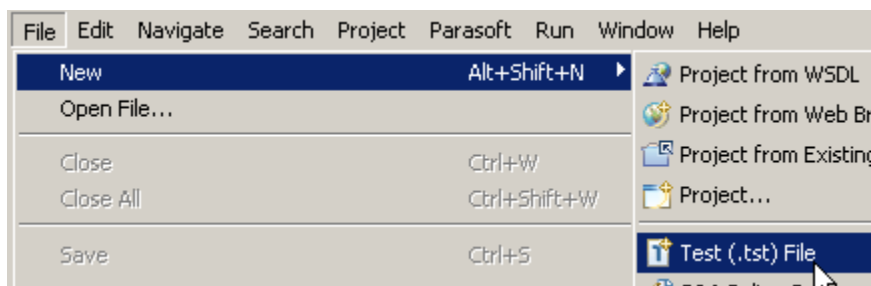
Adding a New .tst File to an Existing Project

We recommend that you create a separate test (.tst file) for each distinct testing requirement. To add a new Test Suite (.tst) file to an existing project.

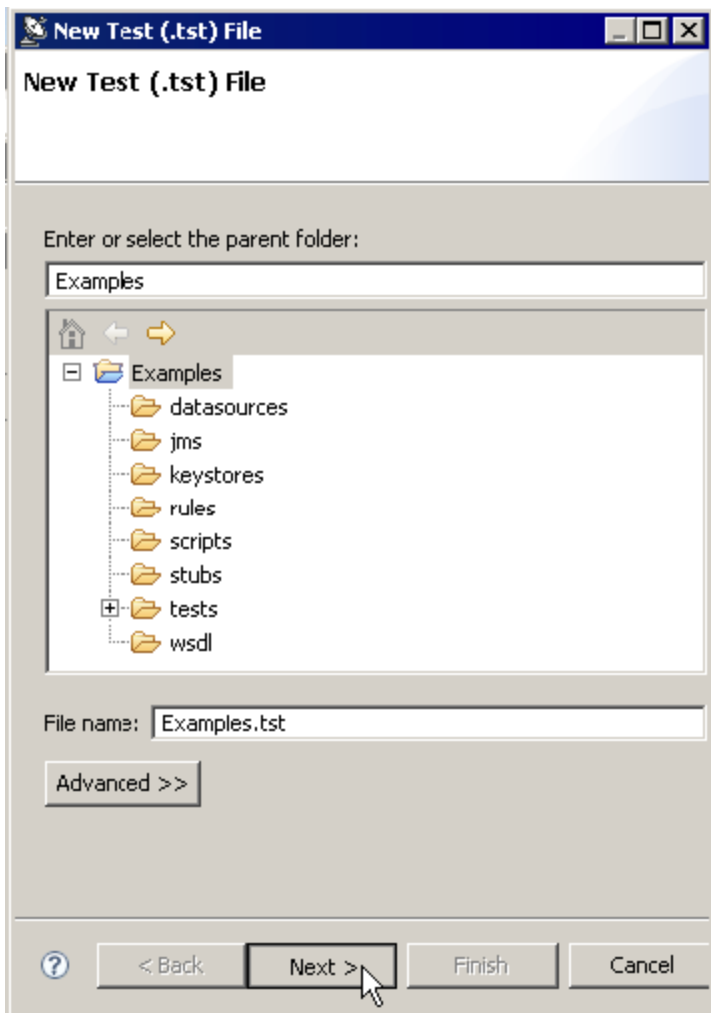
1. Do one of the following:
 - Right click the project node, and select **Add New > Test (.tst) File** from the shortcut menu.



- Choose **File > New > Test (.tst) File**.



2. In the New Test (.tst) File wizard that opens, select the project that you want to contain the .tst file, enter a name for the .tst file, then click **Next**.



You can then complete the wizard to specify what type of tests you want to create and how you want them created. For help selecting and completing the available test creation wizards, see [Wizards for Creating Projects, .tst Files, and Tests](#).

Adding a New Test Suite

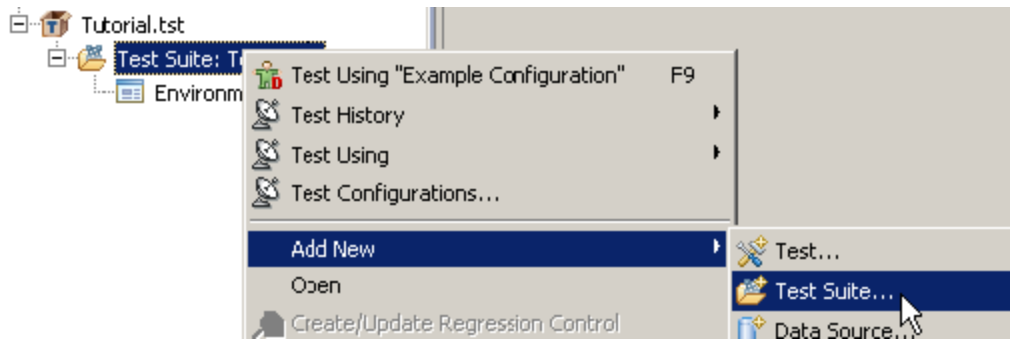
To create a new test suite:

1. Do one of the following:

- Select the Test Case Explorer node for the existing test suite into which you want to add a new test suite, then click the **Add Test Suite** button:



- Right-click the Test Case Explorer node for the existing test suite into which you want to add a new test suite, then choose **Add New> Test Suite** from the shortcut menu.



For help selecting and completing the available test creation wizards, see [Wizards for Creating Projects, .tst Files, and Tests](#).

Using Eclipse Java Projects in SOAtest

Creating a New SOAtest Java Project

SOAtest allows you to create a new Eclipse Java project that has access to SOAtest's Extensibility API, then configure SOAtest scripts and Extension tools to invoke classes from the new Java project.

To create a new SOAtest Java project:

1. Choose **File> New> Project**.
2. Select **SOAtest> Custom Development> SOAtest Java Project**, then click **Next**.
3. Complete this wizard, which has the same options as Eclipse's Java Project wizard.
4. Click **Finish**.

Your new Java project will be shown in the Package Explorer view in the Eclipse Java development perspective. The project's build path will automatically have the jar files needed in order to use SOAtest's Extensibility API. Any Java classes added to your project can be accessed by Extension tools in your SOAtest test suite. For an example of how to do this, see "Java Example" in [Extensibility and Scripting Basics](#).

Using an Existing Java Project

To use an existing Java project from your workspace, you must first add that Java project to SOAtest's classpath as follows:

1. Choose **Parasoft> Preferences**.
2. Open the **Parasoft> System Properties** page.
3. Click the **Add Java Project** button and choose the appropriate project.

The selected Java Project's build output folder and build path entries will be added to the classpath table.

If the **Automatically reload classes** option is enabled, then SOAtest will attempt to reload classes from your Eclipse project after being modified or recompiled. The **Reload** button can also be used to force SOAtest to reload classes from the classpath entries.