

Web Functional Testing

The following exercises demonstrate how to use SOAtest to perform functional testing on the Web interface. It covers:

- Introduction
- Cross-browser Testing and Validation Video Overview
- Recording in a Browser
- Adding a Data Source
- Parameterizing a Form Input
- Configuring Validation on a Page Element
- Configuring a Browser Stub
- Playing a Recorded Scenario in a Browser
- Performing Static Analysis During Web Scenario Execution

Introduction

Web interface testing is difficult to automate. Teams often abandon automated testing in favor of manual testing due to too many false positives or too much effort required to maintain the test suites.

SOAtest facilitates the creation of automated test suites that are reliable and dependable. Its ability to isolate testing to specific elements of the Web interface eliminates noise and provides accurate results.

SOAtest isolates and tests individual application components for correct functionality across multiple browsers without requiring scripts. Dynamic data can be stubbed out with constant data to reduce test case noise. Validations can be performed at the page object level as well as the HTTP message level. SOAtest also verifies the client-side JavaScript engine under expected and unexpected conditions through asynchronous HTTP message stubbing.

Cross-browser Testing and Validation Video Overview

Recording in a Browser

Before performing the following exercises, ensure that you have started the ParaBank Server (as described in [Setting Up ParaBank](#)) and that you can navigate successfully to the default address (`localhost:8080/parabank`). Close the browser window after you verify this.

To record in a browser:

- Right-click the project from the previous exercises, then choose **Add New** > **Test (.tst) File** from the shortcut menu.

Enter a name for the file, then click **Next**.

Select **Web**. Record web scenario and click **Next**.

In the first Record Web Scenario wizard page, ensure that **Record new web scenario** is selected, then click **Next**.

Complete the next Record Web Scenario wizard page as follows:

- Enter `Web Functional Testing` in the **Test Suite Name** field.
- Enter `localhost:8080/parabank` in the **Start Recording From** field.

If you will be using Internet Explorer to record/playback this test, use the name of the local machine instead of `localhost`—here and throughout this tutorial lesson.

Ensure that the following options are checked, and the others are not:

- Generate Functional Test**
- Generate Asynchronous Request Tests**

Click the **Finish** button. The test will begin, and a browser window will open.

Within the browser window that opens,

perform the following actions:

- Type `john` in the **Username** field.
- Type `demo` in the **Password** field.
- Click **Log In**.

Click the first account number listed in the Accounts Overview page (12345).

Click **Log Out**.

Click **Logout**.

Close the browser to end recording. In the Test Case Explorer view, SOAtest creates a new .tst file and a test suite that contains the scenario that you just recorded.

Expand the new test suite node in the Test Case Explorer to view the tests created for each user action taken during the recording.

Click **Test**.

Adding a Data Source

To add a data source:

- Right-click the **Scenario: Web Functional Testing** node, then choose **Add New** > **Data Source** from the shortcut menu.

Click **Add New** > **Data Source**.

To add a data source:

- Right-click the **Scenario: Web Functional Testing** node, then choose **Add New** > **Data Source** from the shortcut menu.

Click **Add New** > **Data Source**.

In the New Project Data Source wizard, select **Table** and click **Finish**.

Click **TableCreate**.

A new **Data Sources** node is added to the **Scenario: Web Functional Testing** branch.

In the New Datasource table configuration panel that is opened in the right side of the GUI, make sure the **First row specifies column** names check box is checked.

Enter `Account Number` in the top cell in column A. In the same column, enter `12345` in row 1 and `13344` in row 2.

Click **Save** to save the changes.

Click **Save** to save the changes.

Click **Test 3: Click '12345'** node to open the test configuration panel.

Note that the **Pre-Action Browser Contents** tab shows what the page looked like before the test action (clicking '12345') was performed. It uses a blue border to highlight the user action for this test.

Click **Save** to save the changes.

Click **Save** to save the changes.

Run the test. It will now click on the **Account Number** element that has been parameterized with the account number entered in the data source.

Configure a validation on a page element.

Fully expand the **Scenario: Web Functional Testing** branch.

Double-click the **Browser Contents Viewer** node under **Scenario: Form login** > **Test 3: Click 'Log In'**.

Click **BrowserContents**.

data-base-url="https://docs.parasoft.com" data-linked-resource-content-type="image/png" data-linked-resource-container-id="27522267" data-linked-resource-container-version="1">
Click **Save** to save the changes.<h2 id="WebFunctionalTesting-PlayingRecordedScenarioinaBrowser">Playing a Recorded Scenario in a Browser</h2><p>To playback the recorded scenario in a browser:</p>In the Test Case Explorer, select the **Scenario: Web Functional Testing** node.Click the **Test** toolbar button.<p>The recorded scenario will now be played back in your browser, once for each search value that we parameterized in a previous section of this example. Please wait for each action to be played out in your browser.</p><p>Note if you have been following the complete tutorial, errors will be reported due to the Browser Stub tool that was previously added. This is expected.</p><h2 id="WebFunctionalTesting-PerformingStaticAnalysisDuringWebScenarioExecution">Performing Static Analysis During Web Scenario Execution</h2><p>To configure SOAtest to perform static analysis on the Web pages that the browser downloads as web scenarios execute:</p>In the Test Case Explorer, select the **Scenario: Web Functional Testing** node.Open the **Test** toolbar button's pull-down menu, then choose one of the available static analysis configurations.

<p>When SOAtest is finished performing static analysis, static analysis violations are shown in the Quality Tasks view. To facilitate review of these results:</p>Open the pull-down menu on the top right of the Quality Tasks view.

Choose **Show**; SOAtest Static Analysis for Functional Tests Layout.</div></div></body></html>