Performing Static Analysis

This topic explains how you can perform static analysis to identify code that does not comply with a preconfigured or customized set of static analysis rules. Sections include:

About Static Analysis

Static analysis is one of many technologies used to throughout the SDLC to help team members deliver secure, reliable, compliant SOA. Static analysis helps the team ensure that development activities satisfy the organization's expectations, ensuring interoperability and consistency across all SOA layers.

SOAtest can perform static analysis on both SOA artifacts and the Web interface.

For SOA, static analysis can be performed on individual artifacts (e.g., WSDL or XML files). It is also used as one of several components in a comprehensive SOA policy enforcement framework, which is discussed in SOA Quality Governance and Policy Enforcement.

For Web interfaces, SOAtest's static analysis can perform an automated audit of Web interface content and structure, automatically scanning and analyzing a Web asset for accessibility, branding, intranet standards compliance, and consistency. It inspects and exposes issues that present potential risk to the proper functionality, usability, and accessibility of your Web-based applications.

It can cover an entire web UI or an individual component or module. Scan results are presented as actionable reports that identify erroneous objects—providing direct linkage to exposed issues for quick analysis and remediation.

The assessment analysis covers the following areas:

- Accessibility: Support for Section 508, WAI, and WCAG 2.0 guidelines.
- Branding: Automatically enforce policies related to site layout and "look and feel."
- Intranet Standards: Identifies use of sensitive corporate data.
- Consistency: Prevents broken links, spelling errors, browser compatibility issues.

More specifically, to facilitate Web accessibility validation (Section 508, WAI, WCAG), SOAtest automatically identifies code that positively or possibly violates Section 508, WAI, and WCAG 2.0 Web accessibility guidelines. During its automated audit. the solution checks whether Web interfaces comply with core accessibility guidelines and helps you identify code and page elements that require further inspection and/or modification.

Moreover, Parasoft's pattern-based code analysis monitors whether Web language code follows industry-standard or customized rules for ensuring that code meets uniform expectations around security, reliability, performance, and maintainability. We provide an extensive rule library with hundreds of configurable rules for Web languages (JavaScript, VBScript/ASP, HTML, CSS, XML, and so on), as well as a graphical RuleWizard module that makes it very simple to construct and maintain customized rules.

Analyzing Resources From Your Project Workspace

You can use the following procedure to perform static analysis on:

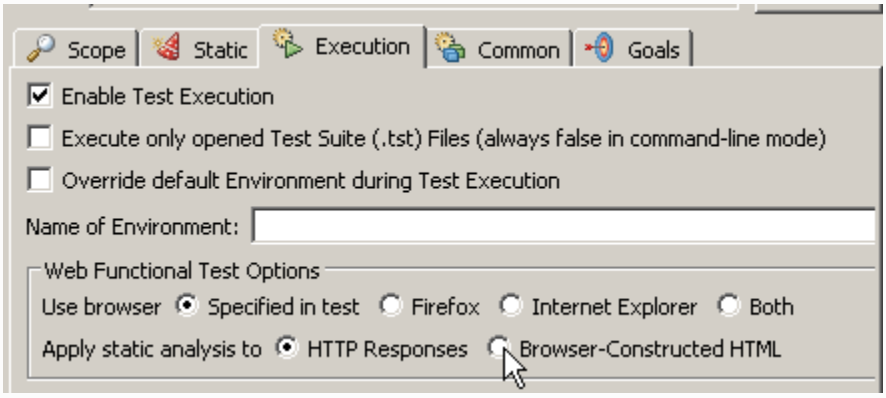
- Any source files that are available in your project workspace (e.g., HTML, XML, WSDL, and other source files added to your project workspace). For details on linking your source files to a project created by SOAtest, see the Eclipse Workbench User Guide (choose **Help> Help Contents**).
- Any Web pages that are represented in SOAtest test suites within your project workspace (e.g., the Web pages that are accessed as SOAtest crawls your web UI or the Web pages that the browser downloads as web scenarios execute).

The general procedure for performing static analysis on one or more files in your project workspace is as follows:

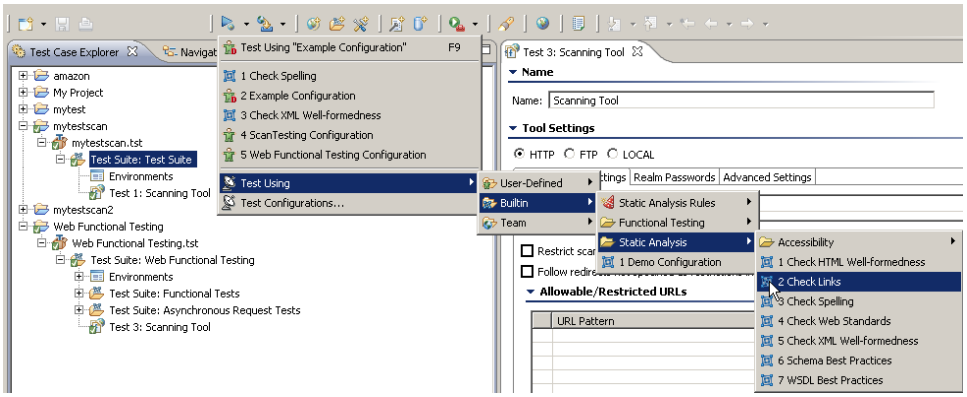
1. Ensure that the files you want to analyze are available within SOAtest. The files must be available as a project in your workspace.
    - If the files are not actually part of the project or downloaded as functional Web tests execute, you can have SOAtest "crawl" your web UI and then analyze the accessed pages; for details, see Configuring SOAtest to Scan a Web UI.
2. Select or create a Test Configuration with your preferred static analysis settings.
    - For a description of preconfigured Test Configurations, see Built-in Test Configurations.
    - For details on how to create a custom Test Configuration, see Configuring Test Configurations and Rules for Policies.

Configuring SOAtest to Run Static Analysis on Web Scenarios

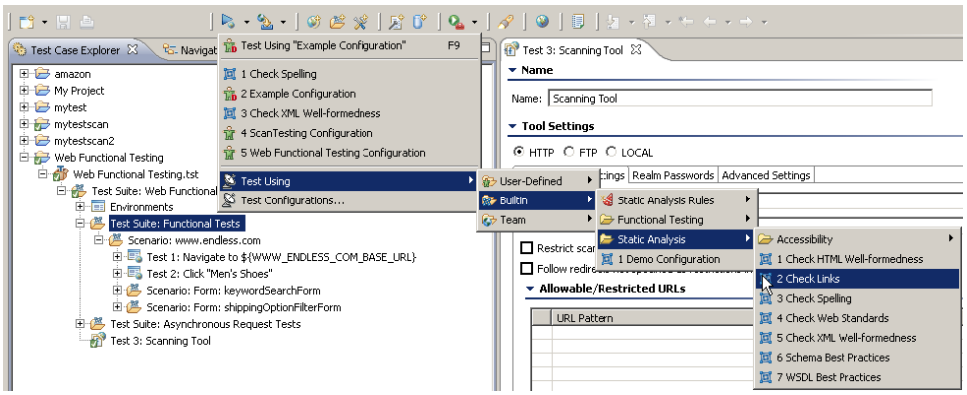
- The selected Test Configuration must have Test Execution enabled (this is the default setting for all built-in Test Configurations).
- By default, SOAtest will statically analyze the individual HTTP messages that the browser made in order to construct its data model—the content returned by the server as is (before any browser processing). If you prefer to analyze the browser-constructed HTML (the real-time data model that the browser constructed from all of the HTML, JS, CSS, and other files it loaded) you can change this by modifying the Test Configuration's Execution settings.



3. Select the resource you want to analyze, then run the appropriate Test Configuration.
   - To test the Web pages that are accessed as SOAtest crawls your web UI, select the Test Suite that contains the Scanning Tool, then run the desired Test Configuration.



   - To test the Web pages that the browser downloads as web scenarios execute, select the Test Suite that contains the scenarios, then run the desired Test Configuration.



   - To perform static analysis from the command line, use the procedure described in Testing from the Command Line Interface - soatestcli.
4. Review and respond to the results using the appropriate static analysis results layout option.
   - For details, see Reviewing Static Analysis Results.
5. (Optional) Fine-tune static analysis settings as needed.
   - For details, see Customizing Static Analysis: Overview.

Reviewing and Retesting Scanned Resources

The Scanning Perspective is designed to facilitate reviewing and retesting of resources scanned during static analysis.

Opening the Scanning Perspective

To open the Scanning Perspective:

- Choose **Window> Perspective> Open Perspective> Parasoft Scanning**.

This perspective is similar to the SOAtest perspective, but has two additional features:

- The Quick Test tool bar button for testing a single URL or file (as described Reviewing and Retesting Scanned Resources).This button can be added to any perspective by choosing **Window > Customize Perspective> Commands** and clicking the check box next to SOAtest Scanning.
- The Scanned Resources view. The view can be added to any perspective by choosing **Window > Show View> Parasoft> Scanned Resources Listing**.

Reviewing Scanned Resources

The Scanned Resources view will show the resources scanned by a Scanning tool. After a Scanning Tool has been run, you can select it in the Test Case Explorer, and the Scanned Resources Listing will show all items that have been scanned by that Scanning Tool.

You can right-click single files in that view, and choose to open them in an editor or a web browser.

Analyzing Files Outside of Your Project Workspace

To quickly access and scan a resource that is NOT available in your workspace:

1. Open the Scanning perspective by choosing **Window> Perspective> Open Perspective> Parasoft Scanning**.
2. Start the test with the Quick Test tool bar button: 
    - To run a test with the "Favorite" (default) Test Configuration, simply click that toolbar button.
    - To run a test with another Test Configuration, use that button's pull-down menu.
    - To check only a specific category of rules, choose **Built-in> Static Analysis Rules> [category]**.
3. Specify the how to access the resource you want to scan (you can specify a URL or a file path).
    - The file path can be absolute, or it can be an Eclipse workspace path).
    - The specified file can also be a .urls file that contains a list of URLs (see Using .urls Files for details).

Tutorial

For a step-by-step tutorial of how to perform static analysis on a web UI, see Web Static Analysis.