Configuring Test Configurations

In this section:

Overview

Test configurations define how your code is analyzed and tested, including which static analysis rules are enabled, which tests to run, and other analysis parameters.dotTEST ships with built-in test configurations, but users can create and store their own test configurations in the DTP server (see the DTP documentation for details).

User-defined test configurations that are stored in DTP can be downloaded from the DTP server and stored in the [INSTALL_DIR]/configs/user directory as *.properties files.

Running a Test Configuration

You can specify which configuration will be run in one of the following ways:

- Run `dottestcli`with the `-config` switch and specify a built-in, user-defined or DTP-hosted test configuration:

  ```
  -config "builtin://Recommended Rules"
  -config "user://Foo Configuration"
  -config "dtp://Foo Team Configuration"
  -config "dtp://FooTeamConfig.properties"
  ```

  You can also provide a path or URL to the test configuration .properties file:

  ```
  -config "C:\Devel\Configs\FooConfig.properties"
  -config "http://foo.bar.com/configs/FoodConfig.properties"
  ```

  For example, your command line may resemble the following:

  ```
  dottestcli.exe -solution "C:\Devel\MyFooSolution\MySolution.sln"
  -config "builtin://Demo" -report "C:\Report"
  ```

- In the .properties file, specify the default configuration that will be run when the `-config` option is not used:

  ```
  dottest.configuration=user://Configuration Name
  ```

Viewing Available Test Configurations

Use the `-listconfigs` switch to print the available test configurations.

Built-in Test Configurations

The following tables include the test configurations shipped in the [INSTALL]/configs/builtin directory.

Static Analysis

This group includes universal static analysis test configurations. See Security Compliance Pack for test configurations that enforce security coding standards.

Built-in Test Configuration	Description
Recommended Rules	The default configuration of recommended rules. Covers most Severity 1 and Severity 2 rules. Includes rules in the Flow Analysis Fast configuration.
Recommended .NET Core Rules	Includes rules that identify high-severity defects in .NET Core projects.
Find Duplicated Code	Applies static code analysis rules that report duplicate code. Duplicate code may indicate poor application design and lead to maintainability issues.
Metrics	Computes values for several code metrics.
Flow Analysis	Detects complex runtime errors without requiring test cases or application execution. Defects detected include using uninitialized or invalid memory, null pointer dereferencing, array and buffer overflows, division by zero, memory and resource leaks, and dead code. This requires a special Flow Analysis license option.
Flow Analysis Aggressive	Includes rules for deep flow analysis of code. A significant amount of time may be required to run this configuration.
Flow Analysis Fast	Includes rules for shallow depth of flow analysis, which limits the number of potentially acceptable defects from being reported.
Critical Rules	Includes most Severity 1 rules, as well as rules in the Flow Analysis Fast configuration.
Demo	Includes rules for demonstrating various techniques of code analysis. May not be suitable for large code bases.
Find Memory Issues	Includes rules for finding memory management issues in the code.
Find Unimplemented Scenarios	Includes rules for finding unimplemented scenarios in the code.
Find Unused Code	Includes rules for identifying unused/dead code.
Check Code Compatibility against .NET [2.0, 3.0, 3.5, 4.0 Client Profile, 4.0 Full, 4.5, 4.5.1, 4.5.2, 4.6.0, 4.6.1, 4.6.2, 4.7]	Includes a set of test configurations that validates the code's compatibility with the specified version of .NET framework.
IEC 62304 (Template)	A template test configuration for applying the IEC 62304 Medical standard.
Microsoft Managed Recommended Rules	Applies the Microsoft Managed Recommended Rules that identify the most critical issues in your managed code

Security Compliance Pack

This compliance pack includes test configurations that help you enforce security coding standards and practices.

Displaying compliance results on DTP

Some test configurations in this category have a corresponding "Compliance" extension on DTP, which allows you to view your security compliance status, generate compliance reports, and monitor the progress towards your security compliance goals. These test configurations require dedicated license features to be activated. Contact Parasoft Support for more details on Compliance Packs licensing.

See the "Extensions for DTP" section in the DTP documentation for the list of available extensions, requirements, and usage.

Built-in Test Configuration	Description
CWE SANS Top 25 2011	Includes rules that find issues classified as Top 25 Most Dangerous Programming Errors of the CWE-SANS standard. This test configuration is part of Parasoft Compliance Pack solution that allows you to monitor compliance with industry standards using the "Compliance" extensions on DTP. It requires dedicated license features to be activated. Contact your Parasoft representative for details.
CWE 3.1	Includes rules that find issues identified in the CWE standard v3.1. This test configuration is part of Parasoft Compliance Pack solution that allows you to monitor compliance with industry standards using the "Compliance" extensions on DTP. It requires dedicated license features to be activated. Contact your Parasoft representative for details.

OWASP Top 10 2017	Includes rules that find issues identified in OWASP's Top 10 standard. This test configuration is part of Parasoft Compliance Pack solution that allows you to monitor compliance with industry standards using the "Compliance" extensions on DTP. It requires dedicated license features to be activated. Contact your Parasoft representative for details.
PCI Data Security Standard	Includes rules that find issues identified in PCI Data Security Standard .
PCI v3.1 Data Security Standard (Server Configuration)	Includes rules that find issues identified in PCI Data Security Standard v3.1.
Security Assessment	General test configuration that finds security issues.
UL 2900	Includes rules that find issues identified in the UL-2900 standard..
Microsoft Secure Coding Guidelines	Includes rules that enforce Microsoft Secure Coding Guidelines.

Unit Testing and Collecting Coverage

This group includes test configurations that allow you to run and collect coverage data for unit tests.

Built-in Test Configuration	Description
Run VSTest Tests	Runs NUnit, MSTest, and xUnit tests that are found in the scope of analysis.
Run VSTest Tests with Coverage	Runs NUnit, MSTest, and xUnit tests that are found in the scope of analysis and monitors coverage.
Run NUnit Tests	Runs NUnit tests that are found in the scope of analysis.
Run NUnit Tests with Coverage	Runs NUnit tests that are found in the scope of analysis and monitors coverage.
Execute MSTests	Executes MSTest tests. See Unit Testing.
Execute MSTests with Coverage	Executes MSTest tests and collects coverage. See Unit Testing.
Calculate Application Coverage	Processes the application coverage data to generate a coverage.xml file. See Application Coverage for Web Applications.
Collect Static Coverage	Generates the static coverage data necessary for application coverage. See Application Coverage for Web Applications.

Creating Custom Rules

Use RuleWizard to create custom rules. To use the rule, it needs to be enabled in a test configuration and the custom rule file must be located in one of the following directories:

- [INSTALL_DIR]\rules\user\
- [DOCUMENTS DIR]\Parasoft\[engine]\rules where [DOCUMENTS DIR] refers to the "My Documents" directory in Windows.