Reducing Noise for Static Analysis and Automated Unit Testing

This topic explains why and how to run an initial static analysis and/or unit testing test and reduce what you consider to be "noise."

If you review/respond to results after the initial test, the results for subsequent tests will be focused on real problems, such as code that throws unexpected exceptions, previously-verified code functionality that changes unexpectedly, or code that violates your team's designated coding policies. Subsequent tests will not report results for expected exceptions and static analysis violations that were reported in the initial run, then handled during the baselining process described below.

To run the initial test and reduce "noise":.

1. Ensure that the code you want to test is available in an appropriate project
    - For instructions on creating projects, see the related product's user's guide (Parasoft [product_name] User's Guide> Setup and Testing Fundamentals> Initial Setup).
2. If you have not already done so, develop and share custom Test Configurations that tailor testing to your project and environment.
    - See Configuring Test Configurations and Rules for Policies for details.
3. Develop and run command line interface commands for running your preferred Test Configuration(s).
4. Verify that the test executed successfully.
5. Import results into the GUI in any of the following ways:
    - Choose **Parasoft> Import> All Tasks**.
    - In the Quality Tasks view, open the pull down menu in the top of the view, then choose **Import> Import All Tasks**.
    - See Importing Results into the UI for other import options.
6. Review the reported static analysis violations reported.
    - For tips on reviewing static analysis violations, see the appropriate product's user guide (Jtest, C++test, dotTEST)
7. Discuss the violations with the team and modify settings as needed to focus results on static analysis violations that your team cares about.
    - If you decide that you never want to see a violation of a specific rule, disable that rule in the team Test Configuration.
    - If a particular static analysis violation is allowed in a specific instance, suppress the violation message. Suppressions will automatically be shared across all Parasoft Test installations connected to Team Server. If all team members' Parasoft Test installations are connected to Team Server, the suppressions that one team member adds will automatically be applied whenever any team member uses the product to test the resource for which that suppression was added. For example, if one team member suppresses a static analysis violation reported for a given file, that same static analysis violation will be suppressed whenever another team member tests that same file.
8. Review the reported unit testing tasks.
    - For tips on reviewing unit testing tasks, see the appropriate product's user guide (Jtest, C++test, dotTEST).
9. Discuss the reported tasks with the team and respond to the findings to prepare for regression testing.
    - For tips on using automatically generated tests for regression testing, see the appropriate product's user guide (Jtest, C++test, dotTEST).
10. Ensure that the project and test files are added to the team's source control system.
    - See Sharing Projects and Test Assets Through Source Control for details on adding resources to source control.