C++test Configuration Overview

This topic explains how to configure C/C++test Professional.

Sections include:

Workflow Overview

If your organization is using Parasoft DTP, we strongly recommend that you adopt the following workflow to simplify configuration and updating of preference settings across your team:

1. Configure team-wide preferences.
    a. Configure team-wide preferences settings in the C/C++test GUI.
    b. Export the settings you configured in the GUI to a localsettings file.
    c. Add the exported settings to your team's DTP.
2. Configure C/C++test preferences on each desktop and server installation.
    a. Configure C/C++test to automatically detect the settings stored on DTP.
    b. Extend or override those settings as needed.
3. Update team-wide preferences on DTP as needed. Changes will automatically be propagated to the connected C/C++test installations.

Configuring Team-wide Preferences

To configure team-wide settings:

1. Choose **Parasoft> Preferences** in the IDE menu bar to open the Parasoft Preferences panel.
2. Configure the following settings:
    - E-mail (see Configuring Email Settings).
    - License (see Licensing).
    - DTP (see Connecting to DTP)
    - Team Server (see Connecting to Team Server).
    - Source control (see Connecting to Source Control).
    - Authors (see Specifying Author-to-Author and Author-to-Email Mappings).
    - Any additional settings you want to share (see Preference Configuration Basics).
3. Click the **Share** link in the top-level Parasoft Preferences page, specify which settings you want to export, and specify where you want to store the localsettings file that contains the exported settings.
4. Add the exported settings to Parasoft DTP (see the 'DTP Projects' section in the DTP User Guide).

Configuring Preferences on Each Installation

Configuring a C/C++test to Use DTP Preferences

To configure each desktop and server installation to use the preferences stored on DTP:

1. Choose **Parasoft> Preferences** in the IDE menu bar to open the Parasoft Preferences panel.
2. Open the **DTP** page.
3. Configure connection with your DTP server (see Connecting to DTP).
4. Click **Apply**.

The settings stored on DTP are automatically refreshed for your C/C++test istallation every time C/C++test is launched. To refresth the settins without restarting C/C++test:

1. Choose **Parasoft> Preferences> DTP**.
2. Click the **Configure** button in the Project field.
3. Select your DTP project in the Configure Project dialog that opens.
4. Click **Finish**.
5. Ensure the **Use DTP settings** option is enabled if available for a given preferences category (see Preferences Categories).

If you're working with multiple DTP projects, you can easily switch from one project's settings to another by choosing a different project in the Configure Project dialog.

Configuring Machine-specific Preferences

You can override team-wide preferences stored on DTP by configuring machine-specific preferences.

1. Open the Preferences category you want to configure (see Configuring Preferences in the GUI).
2. Ensure the **Use DTP settings** option is disabled (if available for the category you want to configure). If enabled, your machine-specific settings will be overridden by team-wide settings configured on DTP (see Updating Team-wide Preferences).
3. Configure machine-specific preferences as needed.
4. Click **Apply**.

Updating Team-wide Preferences

If you are using this recommended process, you can updated team-wide settings in Parasoft DTP, then those modifications will automatically be propagated to all the connected machines.

To prevent this automated updating (e.g., because you have updated settings locally and do not want them overridden), disable **Use DTP settings** on the Preferences page(s) you do not want to be updated from DTP.

Configuring Preferences in the GUI

Preference Configuration Overview

To customize preferences:

1. Choose **Parasoft> Preferences** in the IDE menu bar to open the Parasoft Preferences panel.
2. In the left pane, select the category that represents the settings you want to configure (see Preferences Categories).
3. Configure C/C++test preferences as needed.
4. Click **Apply** to save the settings.

Preferences Categories

Parasoft (Root-Level)

Specifies general preference and allows you to export preferences to a localsettings file.

- **User> User name:** Allows you to set a different user name than the one specified in the operating system.
- **Configure settings> Share/import:** See Exporting GUI Preferences to a localsettings File.
- **C/C++test advanced settings> Settings file**: Allows you to specify the path to a settings file that includes advanced settings for configuring analysis with C/C++test. See Configuring an Advanced Settings File.

Authors

Maps a team member's automatically-detected username to a different username and/or email address. See Configuring Task Assignment and Code Authorship Settings.

Configurations

Specifies the number of Test Configurations available in the **Parasoft> Test History** menu, the location where custom static analysis rules (user rules) are saved and searched for, and the location where user-defined Test Configurations and rules are saved and searched for.

- **Size of recently run test configurations:** Determines the number of Test Configurations available in the **Parasoft> Test History** menu.
- **Custom directories:** Indicates where user-defined Test Configurations and custom directories (e.g., for user rules, embedded cross-compilers, etc.) are saved.

Console

Specifies settings for the Console view.

- **Low:** Configures the Console view to show errors and basic information about the current step's name and status (done, failed, up-to-date).
- **Normal:** Adds command lines and issues reported during test and analysis.
- **High:** Uses full-format violation listings and also reports warnings.
- **Show console on any change:** Determines whether the console is brought to the front any time its content changes.

DTP

Specifies setting for the DTP server. See Connecting to DTP.

When expanded, it allows you to configure settings for Team Server, which is deprecated (see Connecting to Team Server).

E-mail

Specifies email settings used for report notifications and for sending files to Parasoft Technical Support. See Configuring Email Settings.

Issue Tracking Tags

Specifies custom tags that the team uses to associate a test case with an issue from an issue/feature/defect tracking system (such as Jira). See Using Custom Defect/Issue Tracking Tags.

JDBC Drivers

Specifies JDBC drivers (for example, drivers required to connect to a database used for parameterizing tests). See Configuring JDBC Drivers.

License

Specifies license settings. See Licensing.

Parallel Processing

Specifies parallel processing settings. See Configuring Parallel Processing.

Quality Tasks

Specifies general options related to how tasks are displayed in the Quality Tasks view. See Configuring Task Reporting Preferences.

Reports

Specifies what reports include and how they are formatted. See Configuring Reporting Settings.

Scope and Authorship

Specifies how C/C++test computes code authorship and assigns tasks to different team members. See Configuring Task Assignment and Code Authorship Settings.

Source Control

Specifies how C/C++test connects to your source control repositories. See Connecting Source Control.

Technical Support

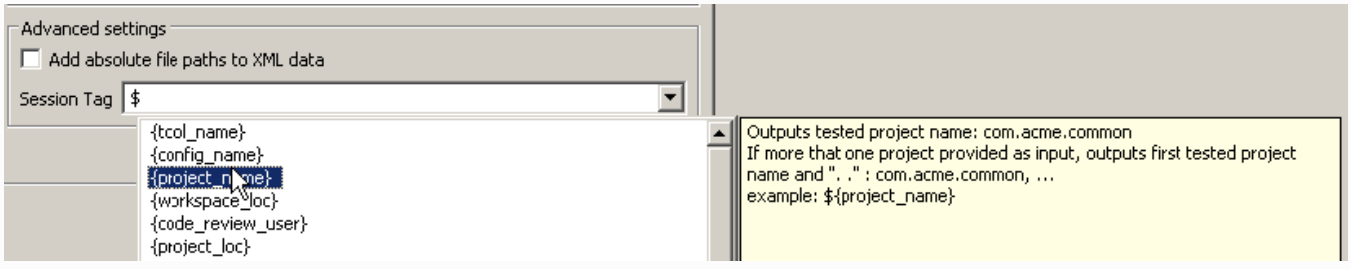
Specifies options for preparing support archives and sending them to Parasoft Support. See Preparing a Support Archive.

Using Variables in Preference Settings

The following variables can be used in reports, e-mail, Team Server, and license settings. Note that the session tag value can't contain any ':' characters.

Using Variable Assist

For help specifying variables, you can use the variable assist feature, which automatically proposes possible variables when you type $. For example:



env_var
example: ${env_var:HOME}
Outputs the value of the environmental variable specified after the colon.

project_name
example: ${project_name}

Outputs the name of the tested project. If more than one project is provided as an input, it first outputs the tested project name, then "...".**general_project**
example: ${general_project}

Outputs the name of the DTP general project that results are linked to.

workspace_name
example: ${workspace_name}

Outputs the workspace name or Visual Studio solution name. For example, `report.mail.subject=Scanner Results for ${workspace_name}`may evaluate to "Scanner Results for solutionAccoun1.sln".

solution_loc (Visual Studio)
example: ${solution_loc}

Outputs the Visual Studio solution location. For example, `report.mail.subject=Scanner Results for ${solution_loc}`may evaluate to "Scanner Results for c:/nightly/folder/.../solution.sln".

config_name
example: ${config_name}

Outputs the name of executed test configuration. Applies only to Reports and Email settings.

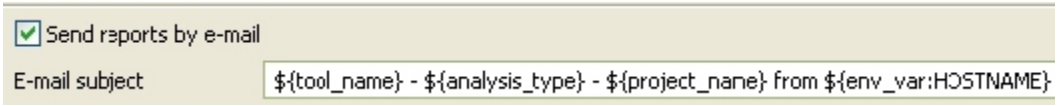
analysis_type
example: ${analysis_type}

Outputs a comma separated list of enabled analysis types. Applies only to Reports and Email settings.

tool_name
$ example: ${tool_name}

Outputs the name of the Parasoft product. For example:

Example:



Exporting GUI Preferences to a localsettings File

You can export C/C++test preferences from the GUI to a localsettings text file. This is useful if you want to:

- Copy settings into DTP for team-wide sharing (see Workflow Overview).
- Quickly create a settings file that you can modify for command-line testing preferences.
- Share the preferences via source control (rather than via DTP).
- Edit preferences in a text file rather than in the GUI.

To export preferences to a localsettings file:

1. Choose **Parasoft> Preferences** in the IDE menu bar to open the Parasoft Preferences panel.
2. Choose **Parasoft** (the root element in the left tree) in the Preferences dialog.
3. Click the **Share** link.
4. Specify the path to the localsettings file where you want to save the preferences in the **File to export** field or navigate to that file using the **Browse** button. If you select an existing file, the source control settings will be appended to that file. Otherwise, a new file will be created.
5. Specify the preferences you want to export to export. Exported passwords will be encrypted.
6. Click **OK.**

To preferences form a localsettings file:

1. Choose **Parasoft> Preferences** in the IDE menu bar to open the Parasoft Preferences panel.
2. Select **Parasoft** (the root element in the left tree) in the Preferences dialog.
3. Click the **Import** link.
4. Specify the path to the localsettings file and the settings you want to import.
5. Click **OK**.

Overriding the System User Name Outside of the GUI

You can override the system user name, or example, if you are integrating the product into an automated process and do not want the resulting tasks assigned to the default system name. To override the system user name, provide a new user name with the `-Duser.name=<username>` switch to the Java Virtual Machine

Note that this configuration is the equivalent to modifying the **User name** setting at the top level of the Preferences UI.

Using Preference Settings for Command-Line Execution

Using an Existing Locally-Stored Localsettings File

If you already have a locally-stored localsettings file that represents the settings you want to use for command-line execution, pass the location of the file using the `-localsettings` command line option, for example:

```
cpptestcli -localsettings my_localsettings_file
```

See Configuring Localsettings and Testing from the Command Line Interface for details.

Using the Settings Stored on DTP

To configure your C/C++test instance to use settings stored on DTP, you can do one of the following:

- Pass the name and port of DTP, as well as the project name, using the `-dtp.autoconfig` command line option, for example:

  ```
  cpptestcli -dtp.autoconfig project1@mydtp.company.com:443
  ```

- Enable the `dtp.autoconfig` option in a localsettings file, for example:

  ```
  dtp.enabled=true
  dtp.server=mydtp
  dtp.port=443
  dtp.project=project1
  dtp.autoconfig=true
  ```

  See Configuring Localsettings for details.

Specifying Multiple Groups of Settings

To facilitate the testing process, you can specify different subsets of setting and pass them on the command line in a certain hierarchy. This allows you to simultaneously apply, for example:

- key project settings stored on DTP
- settings configured for all tests performed on your your particular machine
- settings tailored to perform specific subset set of tests on your machine

Be sure to list the most general settings first and the most specific settings last. Settings are processed in the order in which they are listed and any settings that are duplicated across groups will be overridden each time a duplicate is found. Your command line may resemble the following:

```
cpptestcli -dtp.autoconfig project1@concerto.company.com:443 -localsettings machine_properties -localsettings
project_properties
```