Using a Debugger During Test Execution

This topic explains how to step through C++test test cases with a debugger to better examine the code's internal state during a given test. For example, you might want to debug test cases to learn more about how C++test obtained an unexpected outcome, or to determine why a test case failed. You do not need to manually add breakpoints to the code. C++test will automatically set the breakpoints at the beginning of each test case that you select for execution.You can execute tests with your compiler’s debugger, or debug directly in the Eclipse IDE (according to an Eclipse Debug Configuration you have configured).

Sections include:

About Support for Test Case Debugging

C++test supports test cases debugging in two modes using:

- Eclipse or Visual studio internal debuggers
- External debuggers

Your development environment dictates which mode for test cases debugging should be selected. If you use Visual Studio IDE, then you should be able to select Visual Studio internal debugger (For details please refer to C++test plugin for Visual Studio documentation). If your development environment supports debugging directly in the Eclipse IDE then you should be able to select Eclipse internal debugger mode in your test configuration. You can still configure external debugger for test cases debugging if your development environment does not support debugging inside Eclipse.

For native development C++test supports:

- GNU GCC compilers with gdb debuggers and ddd debuggers front-end
- Sun CC compilers with gdx debuggers
- Other debuggers that can be started and configured via command line interface (see Configuring Debugger Settings)
- Eclipse internal debugger (CDT)
- Visual Studio internal debugger

For embedded development C++test supports:

- Texas Instruments Code Composer Studio (versions 5.x)
- QNX Momentics IDE (SDP 6.4 & 6.5)
- MDK-ARM (versions starting from 4.12 and above)
- Lauterbach Trace32 debugger (starting from release Feb/2013)
- Spansion SOFTUNE (FR) Development Tools for Eclipse (V01L02)

Configuring Debugger Settings

C++test can be configured for two different modes of test cases debugging:

- Debugging internally in the IDE (Eclipse/VisualStudio)
- Debugging with external debugger

To enable debugging inside Eclipse IDE

1. Open test configuration you use for running test cases
2. Choose the **Execution> Runtime** tabs
3. In the Test cases debugging section, enable the **Run tests in debugger(*)** option.
4. Make sure the **Use Eclipse internal debugger with configuration** option is selected and choose an Eclipse Debug Configuration from the drop-down menu.

In this mode C++test will use selected Eclipse Debug configuration to automate test exectable execution and entering the debugging mode inside IDE. For more details see Debugging in Eclipse Internal Debugger Mode.

To enable debugging with external debugger:

1. Open test configuration you use for running test cases
2. Choose the **Execution> Runtime** tabs.
3. In the Test cases debugging section, enable the **Run tests in debugger(*)** option.
4. Enable the **Use External debugger** option.

In this mode some further configuration may be required depending on development environment. See Debugging Using External Debugger, mode for additional details.

Debugging in Eclipse Internal Debugger Mode

Preparing Eclipse Debug Configuration

You can use any of your existing Eclipse Debug configurations for test cases debugging. If you don't have a working Eclipse Debug configuration you will need to first prepare a configuration. For details on configuring Eclipse Debug configuration please refer to your development environment documentation.

It is recommended to verify that your Eclipse Debug configuration can be used for debugging your original project code before specifying it in C++test test configuration.

> **Use the "Standard Create Process Launcher" (standard CDT/Native development)**
>
> For the debug configurations, we recommend using the Standard Create Process Launcher because it allows you to select a debugger. The default Eclipse launcher (GDB DSF) Create Process Launcher does allow you to select a debuggger. Choose **Windows> Preferences> Run /Debug> Default Launchers** to set the select the Standard Create Process Launcher.

Using Selected Eclipse Debug Configuration

1. Open test configuration you use for running test cases
2. Choose the **Execution> Runtime** tabs
3. Make sure the **Use Eclipse internal debugger with configuration** option is selected and choose an Eclipse Debug Configuration from the drop-down menu. You may need to click **Refresh** to reload the names of existing Eclipse Debug configuration.

Debugging Using External Debugger

Test Cases Debugging for Native Development

- The default debugger for GCC compilers is gdb.
- The default debugger for Sun CC compilers is dbx.
- The xxgdb and ddd debuggers are also supported.

C++test uses the following command-lines when launching the default debugger:

- GNU GCC/Windows: `cmd.exe /C gdb -x %s`
- GNU GCC/Linux: `LC_ALL=C /usr/X11R6/bin/xterm -e gdb -x %s &`

Changing the Debugger or the Debugger Command Line

1. Right-click the project tree node for your project and choose **Properties** from the shortcut menu. The Properties dialog will open.
2. Choose the **Parasoft> C++test> Other Settings** category.
3. In the Advanced Options table, click **Add** to add a new entry to the table
4. Add your preferred command line to the Options cell using the following format: `testrunner.debuggerCommandLine <COMMAND_LINE>`.

Examples:

- Use `testrunner.debuggerCommandLine /usr/X11R6/bin/xterm -e xxgdb -x %s &` to use the xxgdb debugger.
- Usete `strunner.debuggerCommandLine ddd -x %s &` to use the ddd debugger.

Test Cases Debugging for Embedded Development

C++test support for debugging tests in embedded development environments is based on specific features of the environment. For more details see Debugging Test Cases.

Executing Tests with a Debugger

This section covers debugging tests in native development environments. For information about debugging tests in embedded environments see Debugging Test Cases.

To execute a test withyour compiler'sdebugger:

1. Prepare a "Debug Unit Tests" Test Configuration.
    - Either use the built-in "Debug Unit Tests" Test Configuration or develop a custom Test Configuration by duplicating the built-in configuration and customizing it as described in the Configuring Test Configurations and Rules for Policies.
2. Run your preferred "Debug Unit Tests" Test Configuration as follows:

a. Select test case(s) that you want to debug in one of the following ways:
   - Select the test case in the Test Case Explorer.
   - Select the test case function name in the code editor.
   - Select the test case function in the Project Explorer (not available for CDT 4.x+ Managed C/C++ projects).

b. Launch your preferred "Debug Unit Tests" Test Configuration. For example, right-click the selection, then use the **Parasoft** shortcut menu to run the preferred Test Configuration. C++test will then launch appropriate debugger and automatically set the break-points at the beginning of each selected test case function.

3. Use standard debugger features to step through the test case.

- Before you use your debugger with C++test, ensure that your debugger executable is included in the $PATH environmental variable.
- The debugger will be activated only if the debugging Test Configuration is run on a selection of one or more test cases. Otherwise, the following warning will appear in the console output and the execution will continue without debugging:
  ```
  Warning: debugger will not be activated - no valid breakpoints found. Select (test case) function definitions to set breakpoints.
  ```

Use the Standard Create Process Launcher

For the debug configurations, we recommend using the Standard Create Process Launcher (modified via **Windows> Preferences> Run/Debug> Default Launchers**) because this allows you to select a debugger. The default Eclipse launcher (GDB DSF) Create Process Launcher does not provide this option.