Connecting to Source Control

This topic explains how to connect C/C++test to your source control repository.

Sections include:

About C/C++test's Source Control Support

Any source control system that plugs into your C/C++test environment can be used to manage your source and test files.

If your team is using a one of the specified supported source control system listed below and performs necessary configuration as described later in this topic, C/C++test can:

- Use file revision data from source control to determine authorship (for automatically assigning test failures and policy violations to the responsible team member as well as for restricting test scope by author and/or modification time). See Configuring Task Assignment and Code Authorship Settings for details.
- Update projects from source control before testing (if the Test **Configuration's Common> Source Control> Update projects** setting is enabled).

C/C++test ships with support for the following source control systems:

Brand	Tested version
Git	1.7, 1.8, 1.9, 2.x
Microsoft Team Foundation Server	2012, 2015, 2017, 2018, 2019
Perforce	2006.2, 2015
Subversion (SVN)	1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 1.10, 1.11, 1.12, 1.13

You can use the source control API to integrate with other source control systems. See Adding a Custom Source Control Integration for details.

Subclipse Support Notes

- Each Subclipse plugin version is compatible with specific Subversion versions. Ensure that your Subclipse plugin is compatible with a version of Subversion that Parasoft supports. For example, you should not install Subversion 1.3 and Subclipse plugin 1.2, which uses Subversion 1.4. Subclipse 1.4.x requires Subversion 1.5.0 version of JavaHL/SVNKit. Subclipse 1.4.5 already has subversion client adapter 1.5.2.
- Due to changes introduced in Subversion 1.4, Subversion clients earlier than 1.4 cannot work with working copies produced by Subversion 1.4. If you are using Subclipse plugin 1.2 (which includes Subversion 1.4), you might receive the following error message: `svn: This client is too old to work with working copy '.'; please get a newer Subversion client` This means that Parasoft is using a command-line client that is version 1.3 or older. The solution is to update your command-line SVN client to version 1.4. The client version can be verified by executing `svn --version`

Why do dialogs open when I try to modify files?

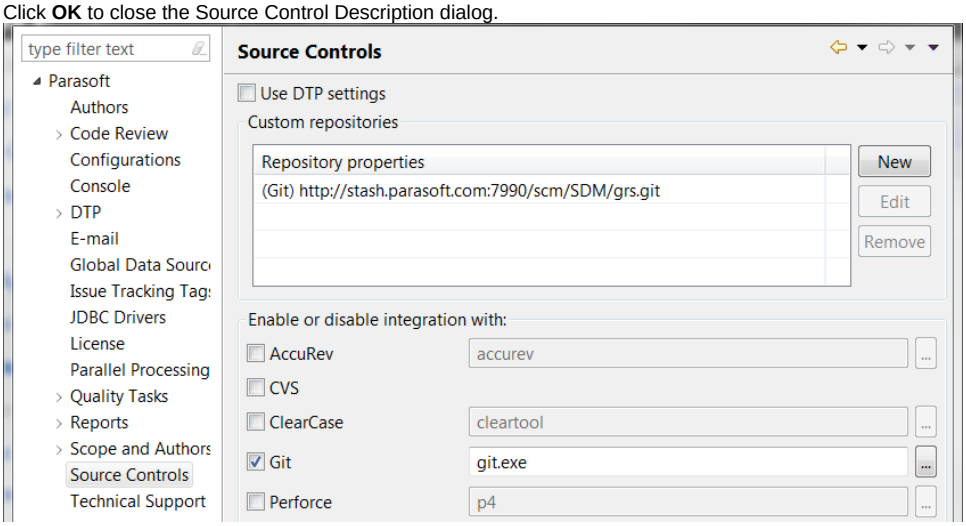
Some source controls (for example, Perforce) require users to mark (lock) sources before editing them. If you are using one of these source control systems and you prompt C/C++test to perform an operation that involves editing a "read-only" file in source control, it will first open a dialog asking you whether you want to make the file writable and lock it. Click **OK**, then provide your source control username and password in the next dialog that opens; this allows it to access the source control system and set the lock.

Enabling Source Control Support

To enable support for any of the supported source control systems:

1. Make sure that the command line client for the given source control is on the system%PATH%/$PATH and is available when C/C++test is launched. For example, if you use Subversion, it is not sufficient (or even required) to install the Subclipse plugin to Eclipse (SVN Eclipse plugin). Instead, you should have the plain command line svn.exe Subversion client.
2. Choose **Parasoft> Preferences** to open the Preferences page.
3. Select **Parasoft> Scope and Authorship** in the left pane of the Preferences page.

4. Enable the **Use source control (modification author) to compute scope** option.
5. Select **Parasoft> Source Controls** in the left pane of the Preferences page.
6. If the appropriate repositories ARE already set (from the auto-configuration process described in C/C++test Configuration Overview), enter your user and password, and specify the path to your source control client executable (if it is not already on your system path).
   If the appropriate repositories ARE NOT already set (refer to the auto-configuration process described in C/C++test Configuration Overview), specify them as follows:
   a. Enable the source control system you want to use
   b. Specify the path to the SCM client executable. You do not need to specify the executable path if it is already on your system path.
   c. Click **New** in Repository properties field and enter the source control properties required for the selected type of source control system.
   d. Click **OK** to close the Source Control Description dialog.



7. Apply your changes and click **OK**.

To test the integration:

1. In the C/C++test environment, open a project that is checked out from the repository.
2. Open a file in the editor.
3. Right-click the source code, and choose **Parasoft> Show Author at Line**. If the correct author is shown, the integration was successful.

Debugging Tip

To troubleshooting problems with source control integration, run `-consolelog -J-Dcom.parasoft.xtest.logging.config.jar.file=/com/parasoft/xtest/logging/log4j/config/logging.on.xml`. This should result in detailed log information being printed to the console.

To include messages from the source control system that may contain fragments of user source code, use an additional flag: `-Dscontrol.log=true`

Git Configuration

The Git repository you connect to must allow anonymous pulls.

1. Open a command prompt and clone your repository:
   `git clone <repository>`
2. If cloning from a git URL, then the `git push` URL must also be configured to automatically push test cases checked in with Git.
   `git config remote.origin.pushurl git@<your repository URL>`
3. Set `git config` on the newly created repository `user.name`.
   `git config user.name "<your username>"`
   This step can be skipped if a global git user.name is already set. The user.name must match the user's DTP user name.

When you are enabling source control support, specify the following repository properties in the Create Source Control Description dialog:

- **URL:** Specify the remote repository URL to pull/push to. If pulling and pushing is disabled, this field can be left blank. You can use all git urls (e.g., git://host/repositoryssh://user@host user@host). Note that since the Git command line doesn't support setting credentials with flags, protocols which normally require authentication (such as ssh) must have identity files set to enable login without credentials or they must accept anonymous connections.
- **Branch:** Enter the name of the branch in the local workspace that the source control module will use. If this is left blank, the currently checked out branch will be used.
- **Working folder:** Enter the root of the local git repository.

Shallow Clones

If you are going to report authorship information from Git to DTP, the cloned repository should not be shallow. A Git repository is considered shallow if the file `.git/shallow` exists. Git may not accurately produce authorship data when checking out shallow clones from a repository. You should check out full clones to retrieve accurate authorship information.

Perforce Configuration

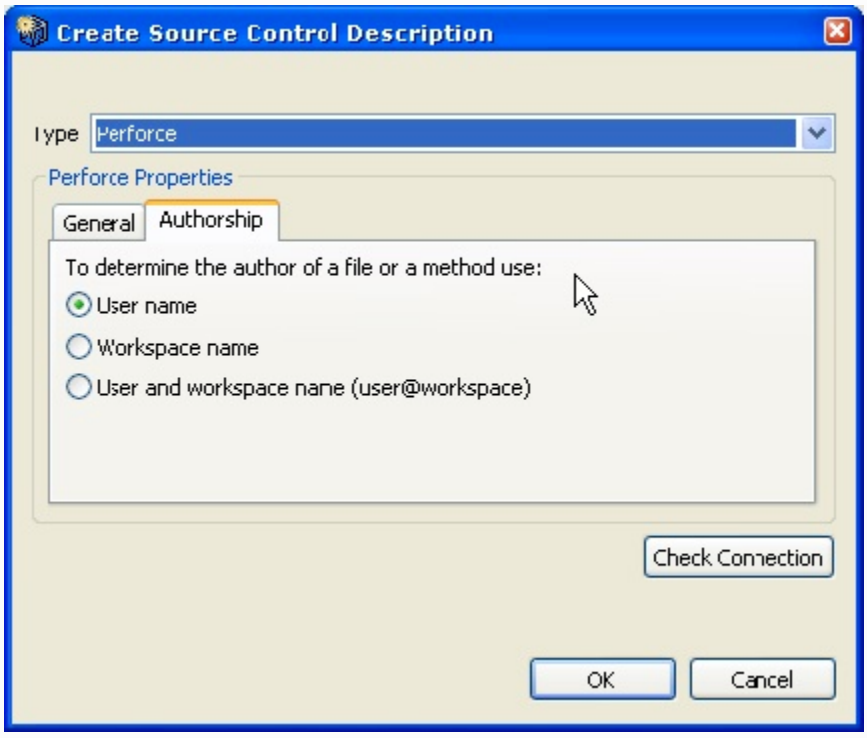
When you are enabling source control support, specify the following repository properties in the Create Source Control Description dialog:

- **Server:** Enter the Perforce server's machine name or IP address.
- **Port:** Enter the Perforce server's port.
- **User:** Enter the user name under which you want to connect to the repository.
- **Password:** Enter the password for the above user name.
- **Client:** Enter the client workspace name, as specified in the P4CLIENT environment variable or its equivalent.

Using Workspaces to Determine Authorship

By default, username is used to determine file/method authorship. However, some teams access Perforce with a shared user name and a unique workspace for each developer.

If you want to use workspace name (or user name and workspace name) to determine authorship, open the **Authorship** tab and modify the setting as needed.



Subversion Configuration

Parasoft's Subversion support is based on the command line client 'svn'. To use Subversion with C/C++test, ensure that:

- A supported version of Subversion client is installed.
- The client certificate is stored in the Subversion configuration area. The Subversion client has a built-in system for caching authentication credentials on disk. By default, whenever the command-line client successfully authenticates itself to a server, it saves the credentials in the user's private runtime configuration area—in ~/.subversion/auth/ on Unix-like systems or %APPDATA%/Subversion/auth/ on Windows.

When you are enabling source control support, specify the following repository properties in the Create Source Control Description dialog:

- **URL:** Enter the URL for the SVN server. The URL should specify the protocol, server name, port and starting repository path (for example, svn://buildmachine.foobar.com/home/svn).
- **User:** Enter the user name under which you want to connect to the repository.

- **Password:** Enter the password (not encoded) for the above user name.

Team Foundation Server Configuration

When you are enabling source control support, specify the following repository properties in the Create Source Control Description dialog:

- **URL:** Enter the URL for the Team Foundation Server repository (for example, http://localhost:8080/tfs).
- **Use custom credentials:** If you want to provide custom TFS credentials, select this option then provide those credentials. When this option is disabled, system credentials will be used.
- **User:** Enter the username. Ensure you provide the same username that you used to configure the TFS repository on your machine.
- **Password:** Enter the password.

By default, C/C++test uses the cached credentials for accessing TFS (this could be your user login or some previously logged in information). You can provide custom credentials if you want to use them instead of the cached ones.

Specifying Source Control Definitions via localsettings

Source control definitions can be specified in localsettings (e.g. for sharing team-wide settings via DTP or specifying options at the command line). See Co nfiguring Localsettings for details.