# Scope Tab Settings: Defining What Code is Tested

During a test, C++test will perform the specified action(s) on all code in the selected resource that satisfies the scope criteria for the selected Test Configuration. By default, C++test checks all code in the resource selected when the Test Configuration is run. However, you can use the Scope tab to configure restrictions such as:

- Test only files or lines added or modified after a given date.
- Test only files or lines added or modified on the local machine.
- Test only files modified by a specific user.
- Test only files that match certain filter criteria
- Exclude autogenerated code from testing

Note that some file filters and line filters are only applicable if you are working with projects that are under supported source control systems. The Scope tab has the following settings:

## File Filters

- **Time options:** Restricts C++test from testing files that do not meet the specified timestamp criteria. Available time options include:
  - **No time filters:** Does not filter out any files based on their last modification date.
  - **Test only files added or modified since the cutoff date:** Filters out files that were not added or modified since the cutoff date.
  - **Test only files added or modified in the last *n* days:** Filters out files that were not added or modified in the specified time period.
  - **Test only files added or modified locally:** Filters out files that were not added or modified on the local machine. This feature only applies to files that are under supported source control systems.
- **Author options:** Restricts C++test from testing files that do not meet the specified author criteria. Available author filter options include:
  - **No author filters:** Does not filter out any files based on their author.
  - **Test only files authored by users:** Filters out any files that were not authored by the specified users. For example, you can use this to focus on files that you—or a selected group of teammates—worked on. To specify multiple users, use a comma-separated list (for example: matt, tom, joe).
- **Path options:** Configures C++test to filter in or out files that match the specified filter criteria. Use the "accept" filter to specify the types of files you want to include. Use the "reject" filter to specify the types of files you want to exclude. For code review test configurations, these filters will be pre-populated with common filtering options.
- **Content options:** Configures C++test to skip any files that include a specified marker (e.g., auto-generated files). To use this, enable **Skip files with content that matches any of the following regular expressions**, then ensure that the **File content regular expression** list contains the tag that is used in the files that you want to skip.
  - If you want to test some code within a file that includes this tag, use the line filter's code blocks options to tell C++test how to find the sections that you want tested.

**Filter Tips and Examples**

*Tips*

- Perl-style expressions can be used.
- The following wildcards are supported:
  - * matches 0 or more characters except '/'.
  - ? matches any single character except '/'.
  - ** matches 0 or more characters, including '/'. This allows you to include path elements.
- The following sample elements are added by default to the Code Review configuration:
  - **/bin/**/*.properties is added to the sample list of rejected wildcards.
  - (.*?/(bin|obj)(/x86|/x64){0,1}/(Debug|Release)/.*?\\.(dll|exe|pdb))$ is added to the sample list of rejected regexps.
- Regular expressions can be used to identify specific differences. For instance, if you want to flag only source code changes that add, remove, or modify TODO tags, you would use the Differences regular expression `.*TODO.*`

*Examples*

A basic file mask might be:

- *.java, *.xml, *.properties
- *.c, *.cpp, *.h, *.cc, *.hpp, makefile, .project, .classpath
- *.c, *.cpp, *.h, *.cc, *.hpp, makefile, *.sln, *.prj, *.res
- *.cs, *.vb, *.sln, *.prj, *.resx

To include every file whose path has a folder named "bank" or "customer", use:

- **/bank/**, **/customer/**

To include every file whose path has a folder with a name that either starts with "bank", includes "customer", or ends with "invoice", use:

- **/bank*/**, **/*customer*/**, **/*invoice/**

To include every .java file that 1) has name that starts with "Test", and 2) is located in a folder named "security" (which is within the src/test directory of any project), use:

**/src/test/**/security/Test*.java

To include every .cs file that 1) is in the ATM solution, 2) is in the ATMLib project, 3) is within the CompanyTests subfolder, 4) has a name that starts with "Test", use:

ATM/ATMLib/CompanyTests/**/Test*.cs

# Line Filters

Restricts the lines of code that C++test operates on. The file filter is applied first, so code that reaches the line filter must have already passed the file filter. Available line filter options include:

- **Time options:** Restricts C++test from testing lines of code that do not meet the specified timestamp criteria. Available time options include:
  - **No time filters:** Does not filter out any lines of code based on their last modification date.
  - **Test only lines added or modified since the cutoff date:** Filters out lines of code that were not added or modified since the cutoff date. This feature only applies to files that are under supported source control systems.
  - **Test only lines added or modified in the last *n* days:** Filters out lines of code that were not added or modified in the specified time period.
  - **Test only lines added or modified locally:** Filters out lines of code that were not added or modified on the local machine. This feature only applies to files that are under supported source control systems.
  - **Test only files modified between working and ____:** Filters out files that were not modified between the working developer branch (in the workspace) and the specified branch (or the default integration stream detected, if that option is enabled). The stream name can be any stream from the parent's hierarchy of developer working streams. The default integration stream is the parent stream of the developer working stream. For example, if you have a stream hierarchy of `[Main] --- [Integration] --- [Developer]`, then `Integration` is the default integration stream for the `Developer` stream. This is currently supported for SVN, AccuRev, and Clear Case.
- **Author options:** Restricts C++test from testing lines of code that do not meet the specified author criteria. Available author filter options include:
  - **No author filters:** Does not filter out any lines of code based on their author.
  - **Test only lines authored by preferred user:** Filters out any lines of code that were not authored by the specified users (i.e., filters out any lines of code that were authored by another user). Use a comma-separated list to specify multiple users.
- **Code blocks options:** Configures C++test to analyze only code that occurs between the specified begin marker and end markers—and (optionally) to skip files that do not include the specified markers. To use this, enable **Use custom begin/end comments (regular expressions) to mark the code to be analyzed**, then specify the markers that indicate the start and end of the code block you want tested. If you want to skip any files that lack these begin and end markers, also enable **Skip files without these markers**.

**Note**

- Code authorship information and last modified date is determined in the manner set in the Scope and Authorship preferences page; for details about available settings, see Configuring Task Assignment and Code Authorship Settings.