File Dependency Action

[FinalBuilder Professional Edition]

The file dependency action allows you to control the flow of your build based on whether or not a file (or set of files) has changed relative to a time stamp.

For example, you can choose to run a set of compiler actions only if the source files are newer than the compiled application.
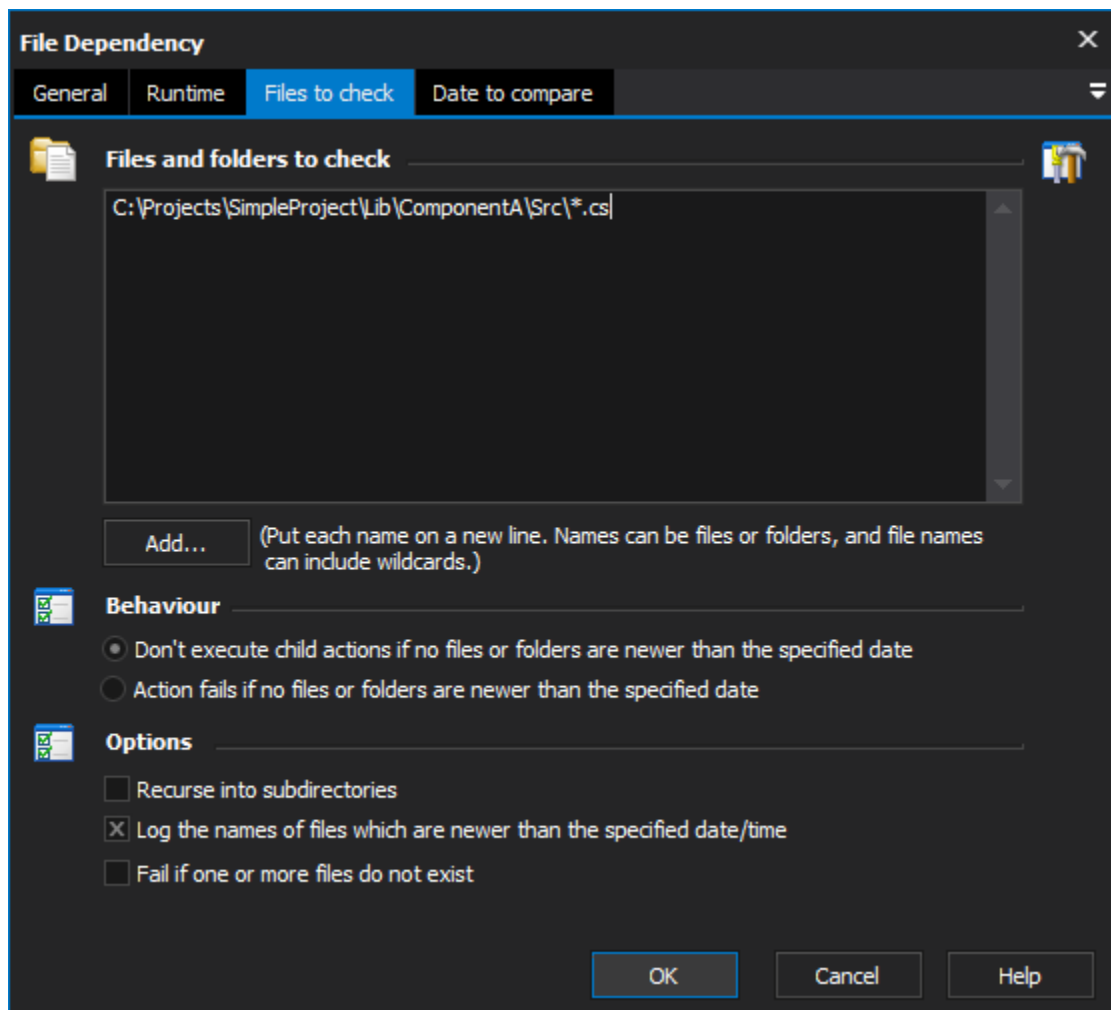
In order to use a file dependency action, you have to choose

(a) the tiles and folders to check (the "source files") and.

(b) the date timestamp to compare them to.

On this page:

Files To Check

Files and Folders to Check

These are the files which will be compared against the time stamp (the source files, in the example above.) The action only executes its children if at least one of the files in this list is newer than the time stamp.

Files can be specified in the following ways:

Format Example

Full Pathname D:\ComponentA\MainClass.cs

Wildcard Filespec D:\ComponentA*.cs

Directory Pathname D:\ComponentA\

Behaviour

Don't execute child actions if no files or folders are newer than the specified date

The child actions of the File Dependency actions will only be executed if one or more files is newer than the timestamp. Otherwise, FinalBuilder will skip to the next sibling action of the File Dependency.

Action fails if no files or folders are newer than the specified date.

Then the action will fail outright if none of the files is newer than the timestamp. This can lead to the build failing completely, or you can use a Try...Catch block to catch the error.

Options

Recurse into subdirectories

If you choose this option, then FinalBuilder will expand subdirectories of directory & wildcard pathnames that you specify. In the example from the screenshot, "Recurse into subdirectories" would return all ".cs" files from all subdirectories of the "ComponentA" directory.

Recurse into subdirectories does not recursively search for plain pathname file names.

Log the names of files which are newer than the specified date/time

If you choose this option, then FinalBuilder will log the names of files which are newer than the timestamp.

Fail if one or more files does not exist

This option will cause the action to fail if any of the full pathnames specified do not exist.

The action will not fail if a wildcard filespec fails to match any files, or if a directory is empty. However, the action will fail (regardless of how this option is set) if none of the specified files can be found.

Date to compare

Date to Compare

Specify the kind of timestamp you wish to use. This is the date of the "built files".

Use Specific Date

Enter a specific (static) date and time combination.

Use date and time that action was last executed

FinalBuilder keeps track of the last time that each File Dependency action has run.

Click the "Reset" button to set the last run time back to January 1, 2005.

Click the "Set to current date and time" to bring the last run time value up to the present moment.

Use date from file, filespec, or directory

As indicated, you can use a file (or group of files) to determine the timestamp.

Enter the full pathname to a file, a wildcard pathname (as in the screenshot), or a directory name. You can choose to have either the oldest or the newest file used for the timestamp.

Use date to oldest file from selection

Specify this option to use the oldest date of the files listed for the date comparison.

Use date to newest file from selection

Specify this option to use the newest date of the files listed for the date comparison.

Recurse into subdirectories

Specify this option to move down the sub-directories when performing the compare.

Always execute if files not found

Specify this option to always execute the action even if files are not located.

Options

Ignore files/folders with dates less than X minutes after the specified date

If you select this option then the timestamp will be brought forward the specified number of minutes to account for "jitter" in the timestamps of object files, etc.

Override and always execute if variable is true

If this option is selected, then the chosen FinalBuilder variable (which needs to be a boolean variable) is evaluated before file dependencies are calculated. If the variable evaluates to true then the dependency evaluation is skipped and the child actions are run automatically.

(This option is useful if you want to sometimes force your build to go ahead, regardless of timestamps.)