

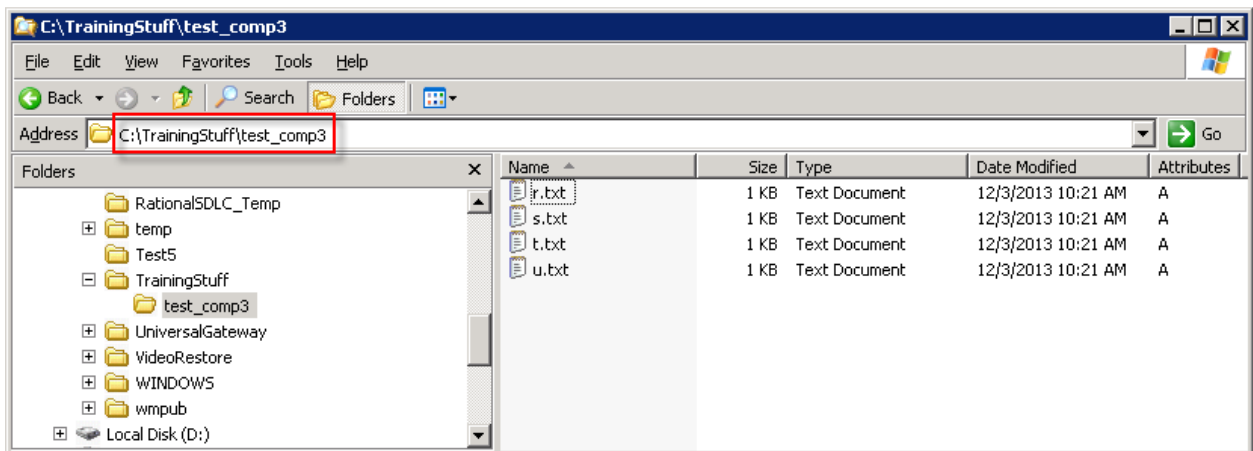
## Pitfalls and Tips on Using ClearCase clearfsimport

Ronald Ross  
[ross@spkaa.com](mailto:ross@spkaa.com)

When using the ClearCase source control management (SCM) system, we often start by importing some new code base. ClearCase has tools for converting other SCM systems such as PVCS, SCCS, RCS, SourceSafe, and CVS. There is also a tool which will import code into ClearCase from a flat directory structure. This is the tool we want to have a look at today – **clearfsimport**.

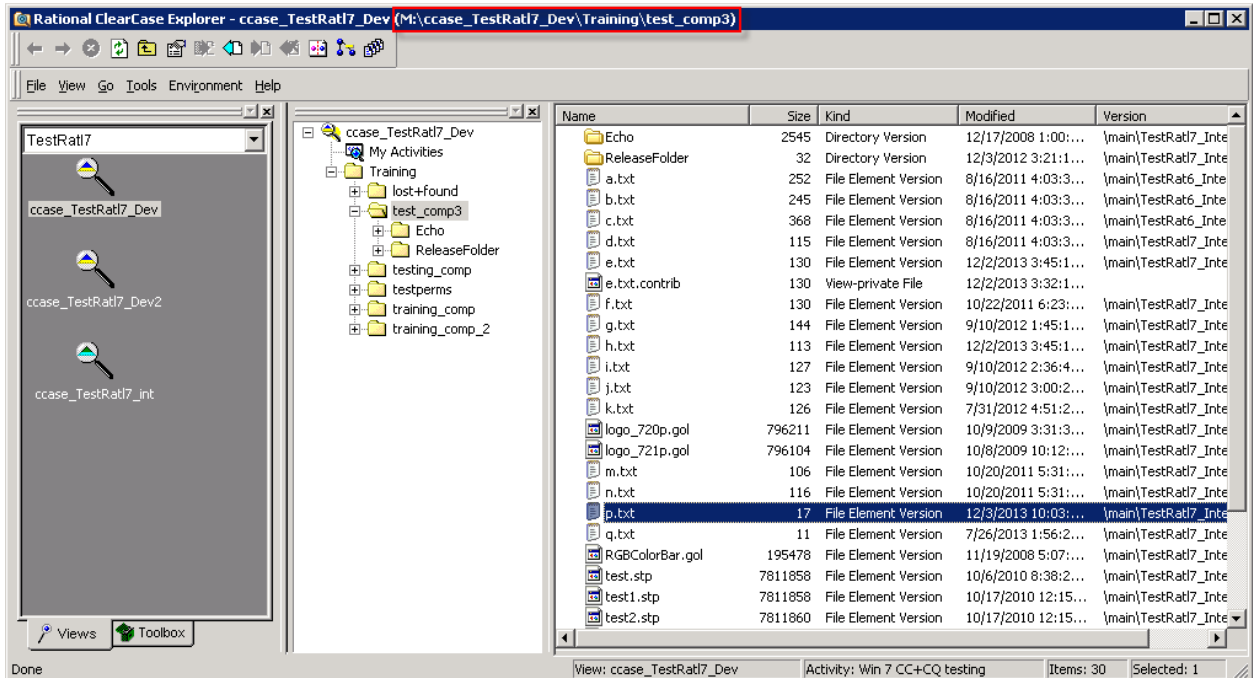
Our purpose today will be to give three examples of importing from a Windows directory structure in to ClearCase, using clearfsimport. But when doing such an import, unexpected results can result if one is not careful. What follows are examples of one successful import, and also two additional imports that have unexpected results.

We will attempt to import a set of four files into an already existing code directory under ClearCase. The files to be imported are shown in the screenshot below.



We want to take the above four files and add them to the directory under ClearCase source control shown in the next screenshot below.

This screenshot shows ClearCase Explorer with a view context displaying a set of “training files”. The first four files are to be added to the directory in the panel on the right side of the window.



Here is the line command that we will use to conduct the code import. The “\” characters simply indicate a line continuation. The four lines constitute a single command as will be seen when we launch the command shown below in a command window. Note that a log file is created containing both the standard and error output by using the redirection primitives “2>&1” and “1>”, respectively.

```
clearfsimport -preview -recurse -rmname -nsetevent -unco -c "Import Training files" \
"C:\TrainingStuff\test_comp3\*" \
M:\ccase_TestRat17_Dev\Training\test_comp3 1> \
C:\Logs\Training_import_pre.txt 2>&1
```

We will first make a few comments about the parameters being used in the command. When initially using clearfsimport, it is usually very useful to include the parameter –preview. This allows the command to be run without actually changing anything, and only displaying messages. The output generated informs the user what would have happened if the command had been run “live”, without the –preview option.

The `-recurse` option descends recursively into any subdirectories and acts on files found in each subdirectory.

The `-rmname` acts on directories to remove them if they are present in the target ClearCase repository, but not in the source directory.

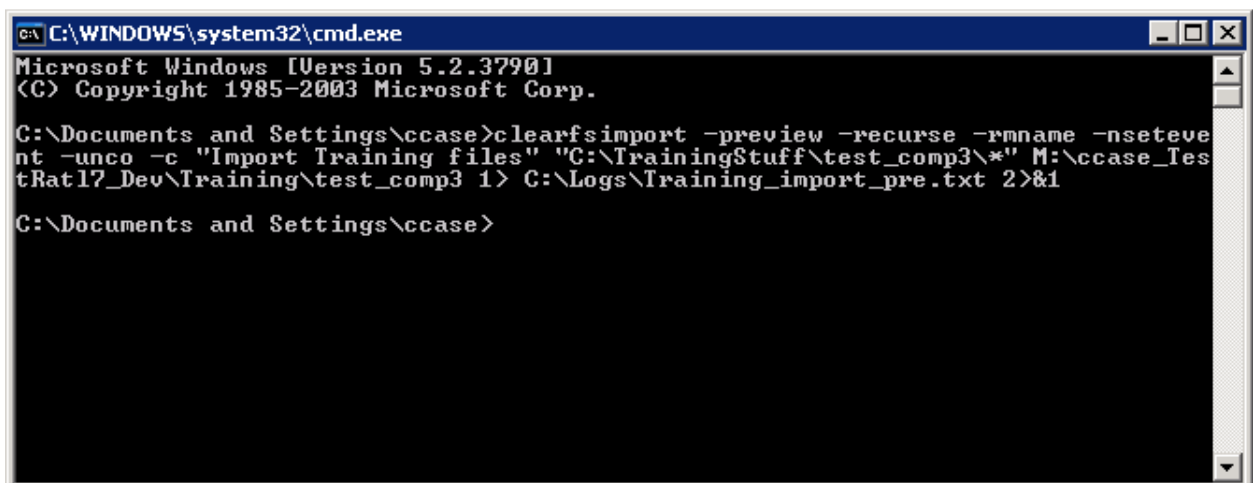
The `-nsetevent` option associates the username who is conducting the import with the files created in source control under ClearCase.

The `-unco` option cancels the checkout of an existing file in the target ClearCase repository if it is checked out, and then checks it out again to do the file import.

The above description is not intended to be an exhaustive description of the options used, but only to provide a brief description and give some context to our use of the command. The full set of `clearfsimport` command options and their descriptions are available at the following IBM webpage.

[http://publib.boulder.ibm.com/infocenter/cchelp/v7r0m1/index.jsp?topic=/com.ibm.rational.clearcase.cc\\_ref.doc/topics/clearfsimport.htm](http://publib.boulder.ibm.com/infocenter/cchelp/v7r0m1/index.jsp?topic=/com.ibm.rational.clearcase.cc_ref.doc/topics/clearfsimport.htm)

We now run the command in the Windows command window shown below. Note that a log is created, which we will subsequently examine.

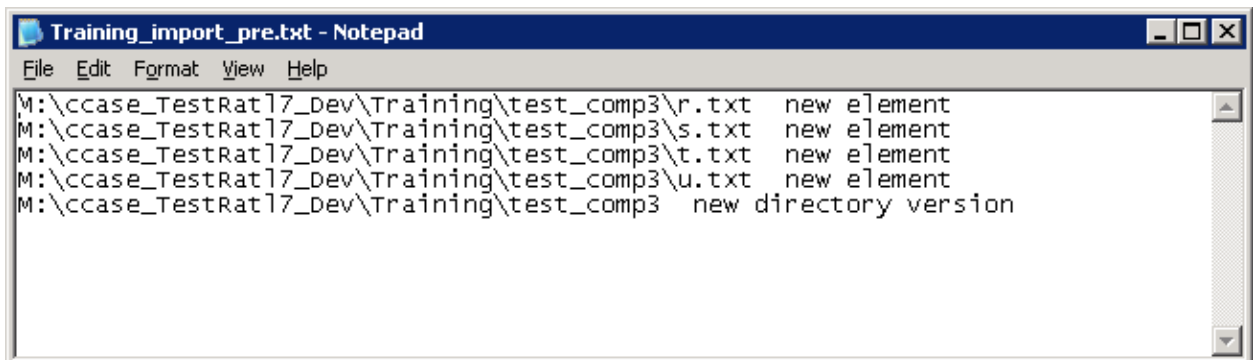


```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\ccase>clearfsimport -preview -recurse -rmname -nsetevent -unco -c "Import Training files" "C:\TrainingStuff\test_comp3\*" M:\ccase_TestRat17_Dev\Training\test_comp3 1> C:\Logs\Training_import_pre.txt 2>&1

C:\Documents and Settings\ccase>
```

Shown below is the log resulting from the clearfsimport command. Note that all four files were successfully imported into the correct directory.

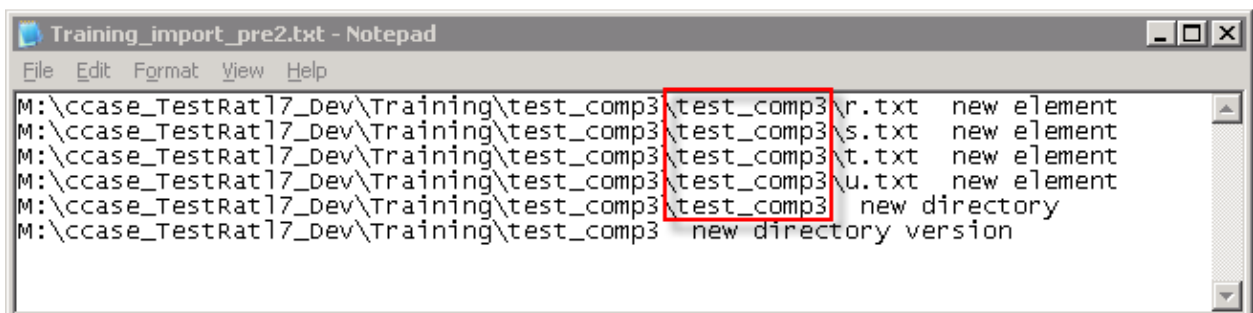


```
Training_import_pre.txt - Notepad
File Edit Format View Help
M:\ccase_TestRat17_Dev\Training\test_comp3\r.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\s.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\t.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\u.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3 new directory version
```

We now turn to a second and slightly different version of the import command. The command clause that is different from the first import has been given a bold typeface. We are now using “test\_comp3” instead of “test\_comp3/\*”. See the command below

```
clearfsimport -preview -recurse -rmname -nsetevent -unco -c "Import Training files" \
"C:\TrainingStufftest_comp3" \
M:\ccase_TestRat17_Dev\Training\test_comp3 1> \
C:\Logs\Training_import_pre2.txt 2>&1
```

After running the command, we examine the resulting log file. Note that the result is rather different from the first time we ran the import command. This time we have imported the directory itself, along with the four files, but we have added another directory level. See the additional directory level indicated in red below.



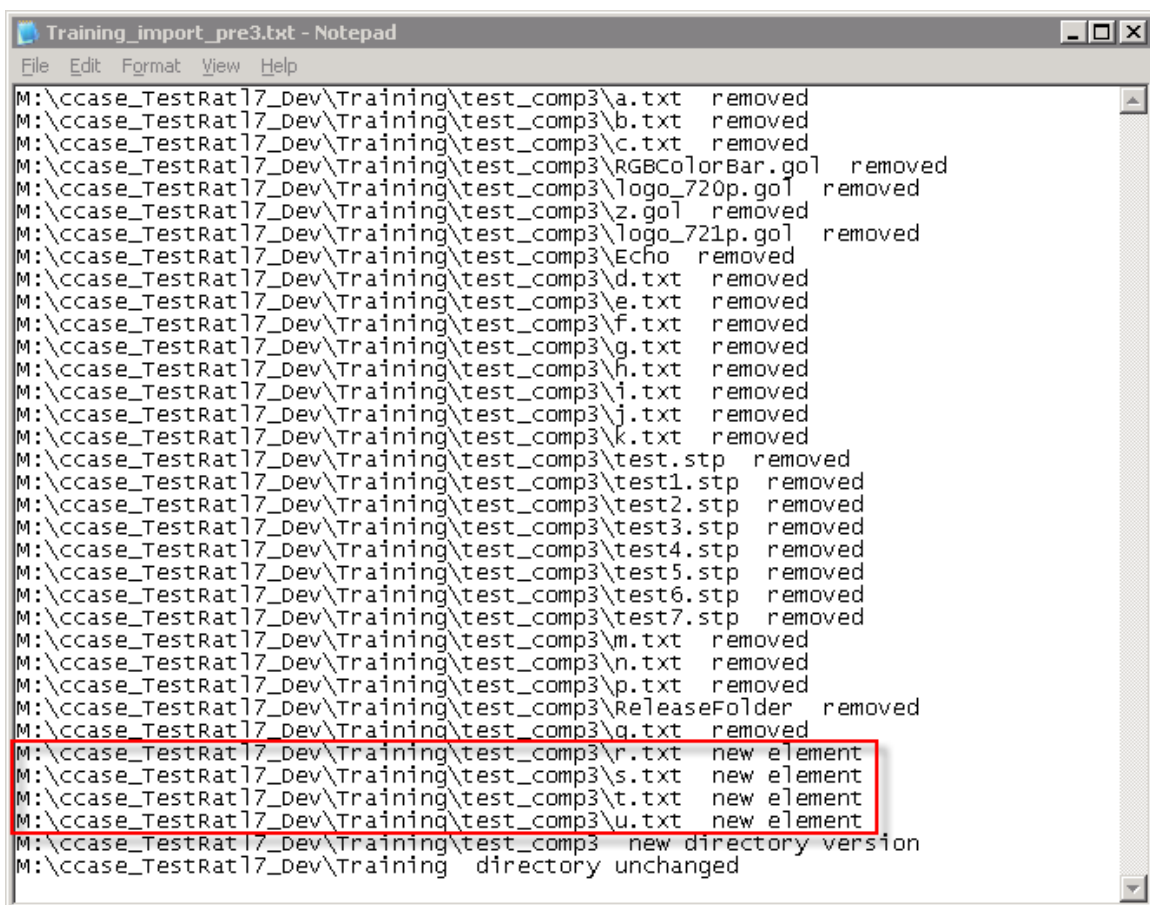
```
Training_import_pre2.txt - Notepad
File Edit Format View Help
M:\ccase_TestRat17_Dev\Training\test_comp3\test_comp3\r.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\test_comp3\s.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\test_comp3\t.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\test_comp3\u.txt new element
M:\ccase_TestRat17_Dev\Training\test_comp3\test_comp3 new directory
M:\ccase_TestRat17_Dev\Training\test_comp3 new directory version
```

Why is this? The explanation is that the first import command specified the files in the test\_comp3 directory (“test\_comp3/\*”) while the second import command specified the directory itself. This caused the directory, along with the four files, to be imported into the targeted area in ClearCase.

Let's now look at a third use of the import command. We have taken the second import command and modified it slightly. The command clause in bold typeface is what was modified. The intent is to again import the four files into the existing ClearCase directory and avoid the doubled directory level that we observed in import number two. The command is below.

```
clearfsimport -preview -recurse -rmname -nsetevent -unco -c "Import Training files" \  
"C:\TrainingStuff\test_comp3" \  
M:\ccase_TestRat17_Dev\Training 1> \  
C:\Logs\Training_import_pre3.txt 2>&1
```

After running the command, we examine the resulting log file, in the screenshot below.



```
Training_import_pre3.txt - Notepad  
File Edit Format View Help  
M:\ccase_TestRat17_Dev\Training\test_comp3\a.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\b.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\c.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\RGBColorBar.gif removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\logo_720p.gif removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\z.gif removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\logo_721p.gif removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\Echo removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\d.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\e.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\f.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\g.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\h.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\i.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\j.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\k.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test1.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test2.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test3.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test4.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test5.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test6.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\test7.stp removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\m.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\n.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\p.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\ReleaseFolder removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\q.txt removed  
M:\ccase_TestRat17_Dev\Training\test_comp3\r.txt new element  
M:\ccase_TestRat17_Dev\Training\test_comp3\s.txt new element  
M:\ccase_TestRat17_Dev\Training\test_comp3\t.txt new element  
M:\ccase_TestRat17_Dev\Training\test_comp3\u.txt new element  
M:\ccase_TestRat17_Dev\Training\test_comp3 new directory version  
M:\ccase_TestRat17_Dev\Training directory unchanged
```

Again we have something different from what we might have expected. We have indeed imported the four new files into the correct directory (outlined in red), but what else has

happened? All the other elements in the target ClearCase directory have been removed! This is quite likely not what was wanted or anticipated.

What has happened to cause this outcome? If we take a close look at the syntax of the third command, we can see that it is coded in such a way as to REPLACE the directory contents to which we initially wanted to add the four files. So, when the directory contents are replaced, we should expect what we observed in the above log file. That is, all the existing files and subdirectories were removed, and the four new files were added.

In some cases, the command behavior of replacing a directory may indeed be what we want to happen. But of course we also always want any commands that we use to do what we expect, and not surprise us with a different outcome.

Here are some takeaways from what we observed above.

1. Always be alert to how the clearfsimport command is coded. As we saw, depending on how the source and target parameters are expressed in the command, the results may be rather different than what is expected.
2. Always use the `-preview` option to determine if the action of the command is what was intended. This can help avoid mistakes which might be time consuming to correct. The old adage “measure twice, cut once” can still apply in the world of software.
3. Always create a log file to record the results of any import activity. If the output just rolls off the screen, it’s not very useful. Having a log file enables us to go back and carefully examine the results of our activities. In some cases, even weeks or months later, questions may arise that can be answered by referring to the original log file.

This concludes our investigation into the use of the ClearCase clearfsimport command. Hopefully this brief introduction to some of the capabilities and pitfalls of the tool is helpful to you. As mentioned above, the IBM website has more complete information on all the options for the command. What we have attempted here is to show in a practical way how the command may be used, and how to insure that the results are what you want.