


# October 3, 2011

- Download the SVNTest project from the class web site.
- Un-tar (tar xzf )
- Import the Project into Eclipse
- Build, test, make sure the project works.

# Subversion

- What is source code version control?
  - <http://svnbook.red-bean.com/>
  - allow multiple people to modify the same source code
  -  – allow one person to manage multiple versions of their source code
    - move from computer to computer to develop
    - track all changes

Repository



zeus.cs.pacificu.edu  
/home/chadd/SVNROOT/

Store your source code on zeus  
check it out and edit it on any  
other machine and upload your  
changes back to zeus.

Client



moe.cs.pacificu.edu  
/home/chadd/workspace/HelloWorld

Client

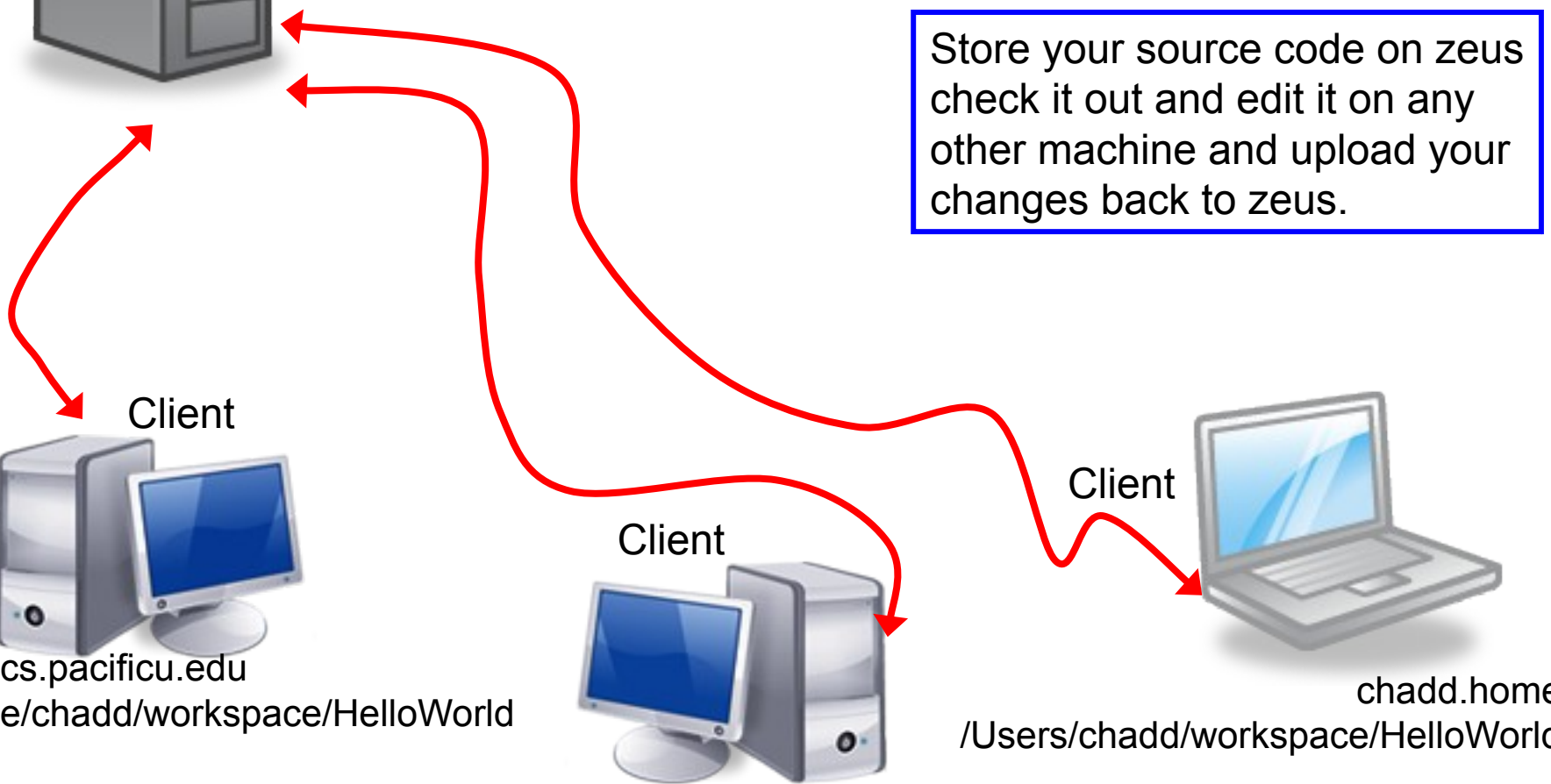


lisa.cs.pacificu.edu  
/home/chadd/workspace/HelloWorld

Client



chadd.home  
/Users/chadd/workspace/HelloWorld



# Topics

- Subversion
  - source control
  - Check in
  - Check out
  - Update
  - Commit
  - Merge Conflict
  - Revert a file

# Version Control

- Each change you make to the source code is a *revision* stored in the repository
  - can annotate your change with a note
    - why did I do that?
  - you can browse back through the repository to find old revisions of file
    - changed a data structure and it did not work
    - rewrote an algorithm and it got slower!
  - check out the old (working) revision from the repository

# Hmmm....

- How often should I **update** and **commit**?
  - every major change
  - once every 15 minutes
  - right before you do something you think may be a bad idea
  - be sure to update and commit before you log off of a lab machine!
    - Or before you leave the lab
    - Someone may reboot your machine!

# How to get this to work

- Create a repository on zeus
  - do this exactly once
  - use this one repository for all your projects
- login to zeus

```
zeus$ svnadmin create /home/chadd/SVNROOT
```

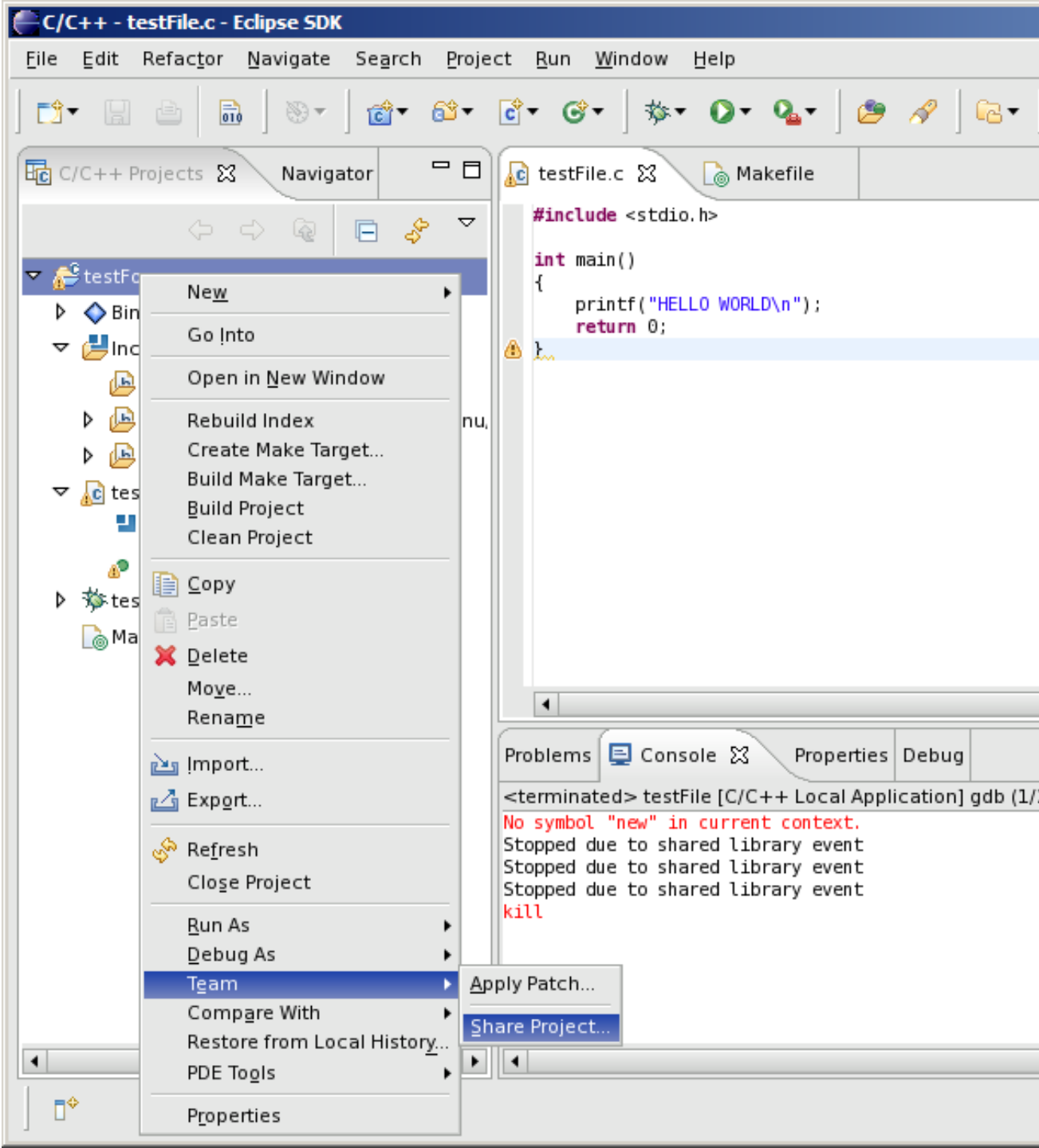
# Check in SVNTest

Problem: Let's check this project into the repository.

How?

Right click on the SVNTest project, then Team, then Share Project.





## Share Project

### Share Project

Select the repository plug-in that will be used to share the selected project.



Select a repository type:

 CVS

 SVN

## Share Project



### Enter Repository Location Information

Define the location and protocol required to connect with an existing SVN repository.



Location

Url:



< Back

Next >

Finish

Cancel

## Share Project



### Enter Folder Name

Select the name of the folder in the SVN repository.



Use project name as folder name

Use specified folder name:

Select...

URL:

svn+ssh://zeus.cs.pacificu.edu/home/ryand/SVNROOT/HelloWorld



< Back

Next >

Finish

Cancel

**Enter SSH Credentials**

Repository:

Username:

Authentication

Use password authentication

Use private key authentication

Password:

Key file:

Passphrase:

Port number:

Save information

**Enter SVN Author Name**

Repository:

Author Name

Save author name

## Share Project



### Ready to Share Project

Select Finish to import the project into the SVN repository.



The wizard has all the information necessary to share your project with the SVN repository. When you click "Finish", the wizard will import your project into the repository and open the Commit dialog to allow you to commit your resources.

Edit the commit comment:

Initial import.|

Choose a previously entered comment:



< Back

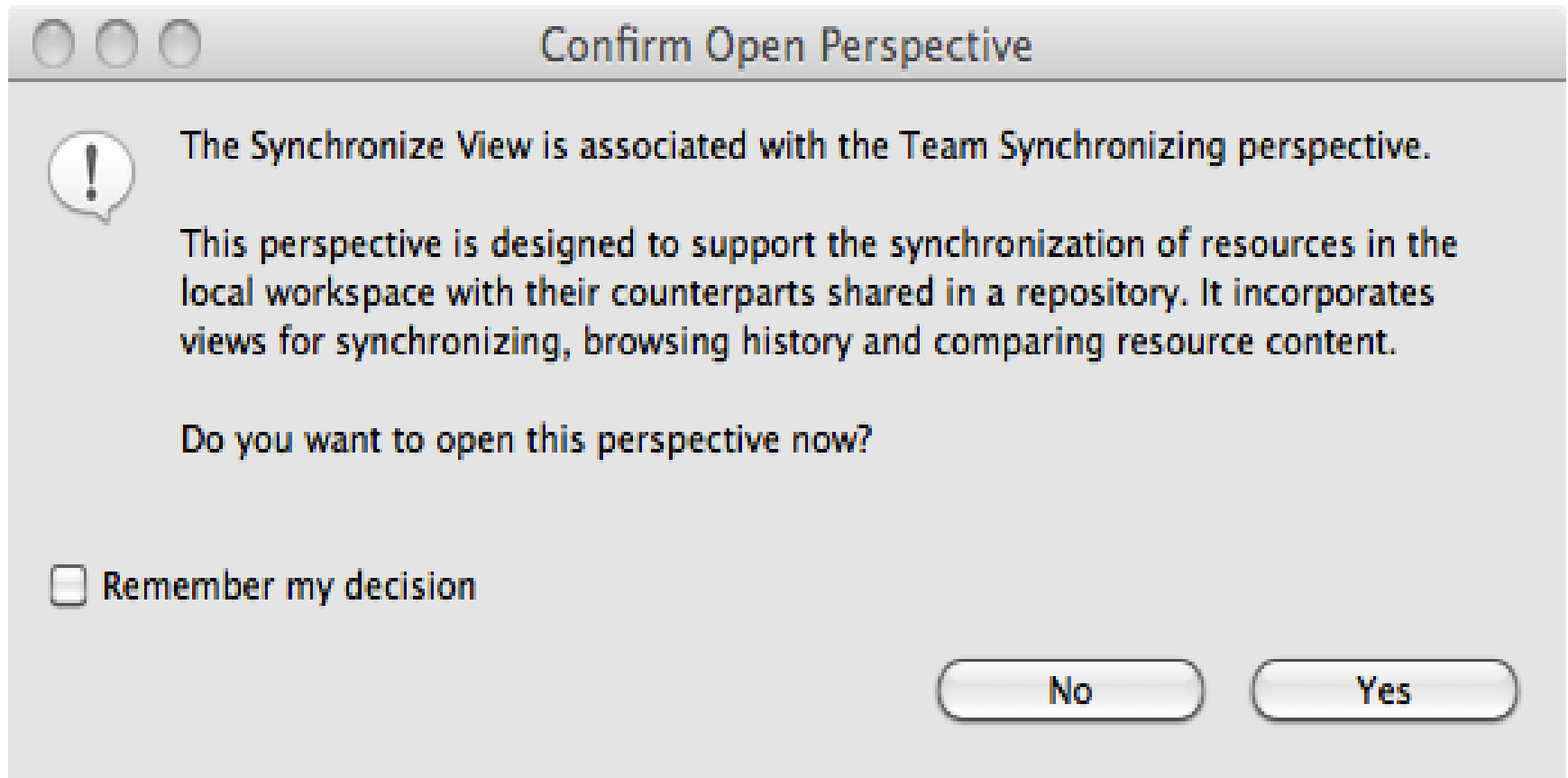
Next >

Finish

Cancel

# Important!!!

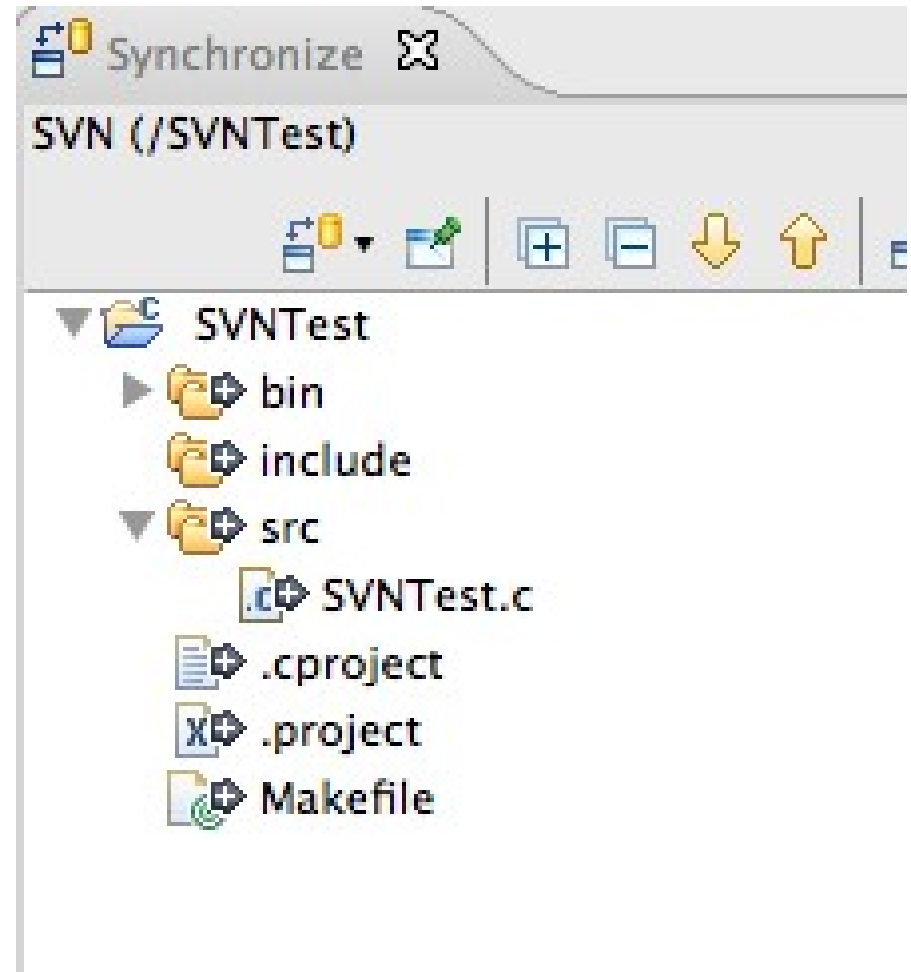
The Initial Import does not commit any code



# Synchronize Perspective

Shows files that need to be committed or updated

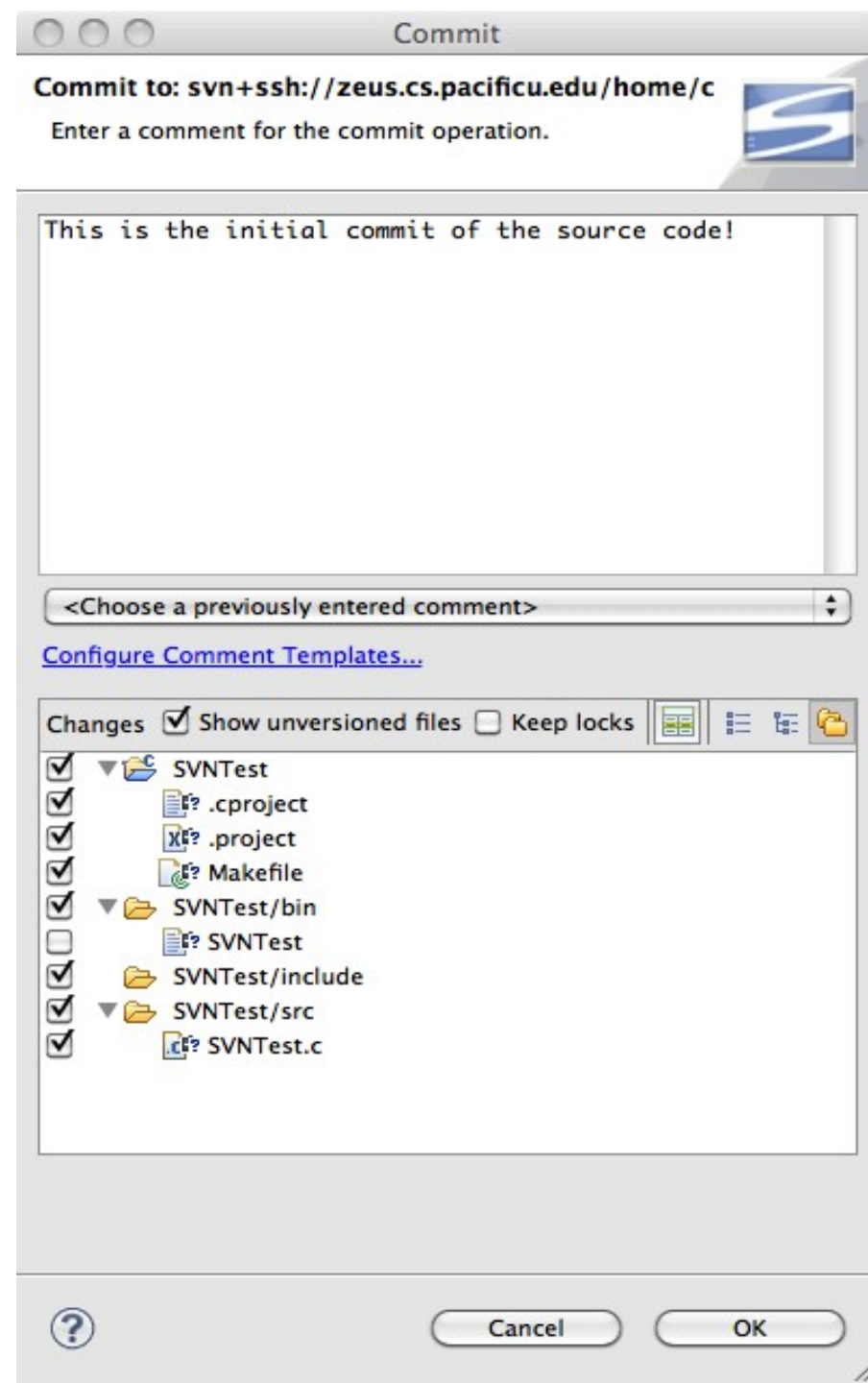
Right Click SVNTest  
Commit





# Commit

- Select files to commit
- Uncheck \*.o or executable files
- Leave a good message!
- Change back to the C/C++ Perspective



# How to do a code commit

To commit a project, right click on the project folder → Team → Commit

Add very descriptive comments for EACH code commit. You will not be sorry.

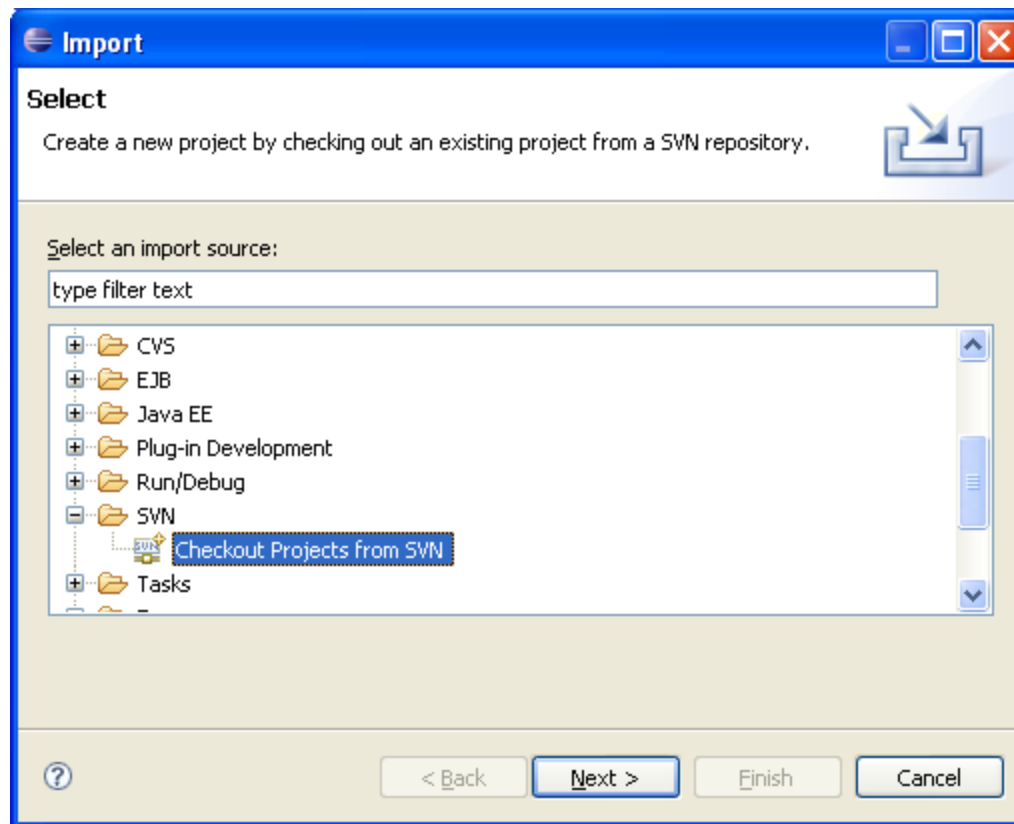
# Let's Delete SVNTest

Right Click the SVNTest Project | Delete

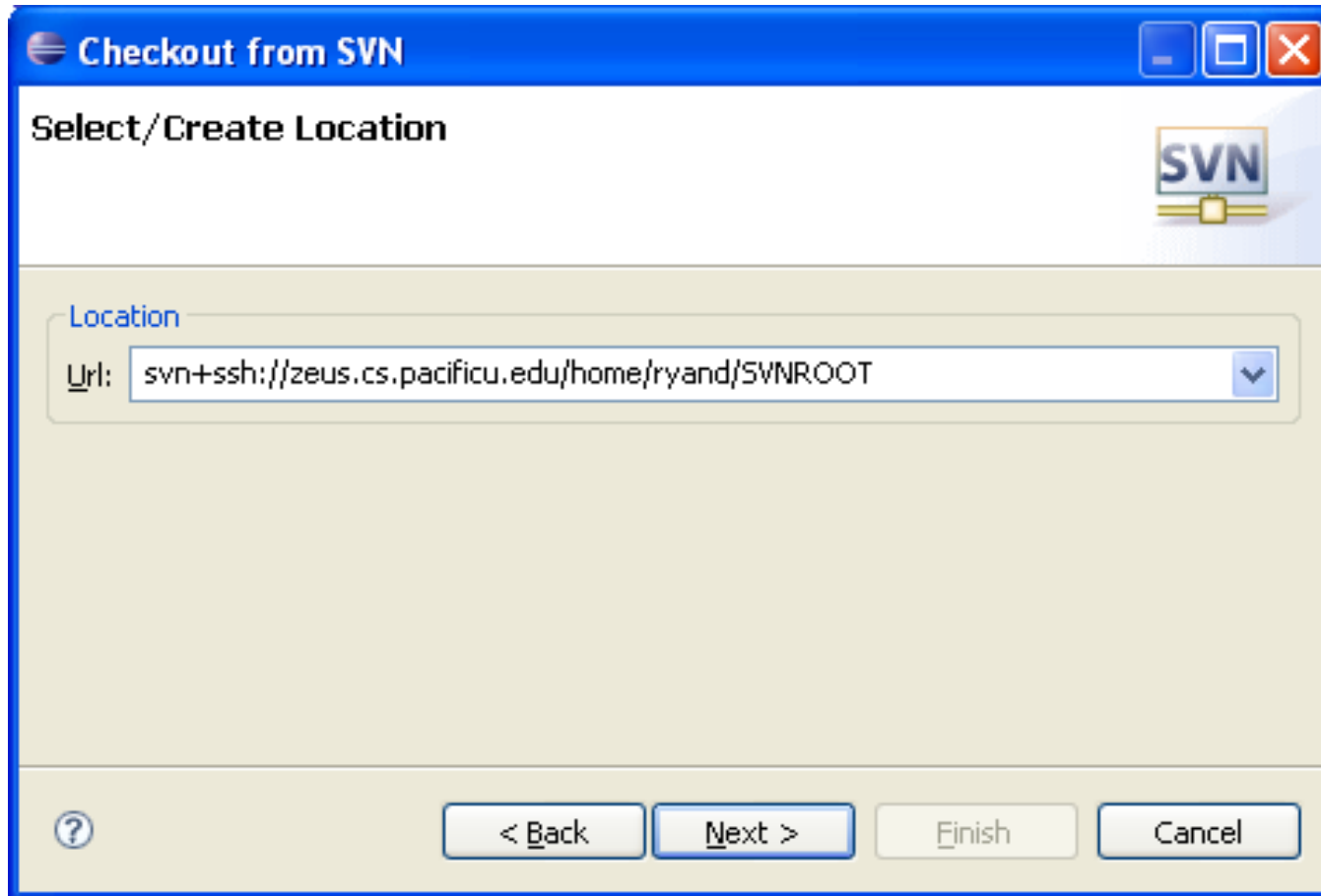
CHECK → Delete project contents on disk

# How to checkout

To checkout a project, File → Import, then



# Finish checking out SVNTest



### Select Folder

Select the folder to be checked out from SVN.



- [-] svn+ssh://zeus.cs.pacificu.edu/home/ryand/SVNROOT
  - [+] ActivityLifeCycleDemo
  - [+] ColorMind
  - [+] DeviceRotation
  - [+] EggDrop
  - [+] GolfCheckPosture
  - [+] HelloAndroid
  - [+] ImageCapture
  - [+] JavaCodingStandards
  - [+] LunarLander
  - [+] MobileUtilities
  - [+] MortgageCalculator
  - [+] Notepadv1
  - [+] Stimpmeter
  - [+] TicTacToe
  - [+] a1.sockets
  - [+] a2.threads
  - [+] a3.tcp
  - [+] blackjack
  - [+] compiler07
  - [+] ds1.introC
  - [+] interpreter

Select the Project  
Press Next



< Back

Next >

Finish

Cancel

# Using Subversion by hand

- Open a single shell prompt
- Create a folder called Junk and change into it
- Check out SVNTest project in Junk directory
- Type

```
$ svn checkout svn+ssh://zeus/home/chadd/SVNROOT/SVNTest
```
- Using Geany, add a printf to main().

# Using Subversion by hand

- From a command line, find the Makefile and re-make the project and run it.
- Now commit the changes to the repository by hand.

```
$ svn commit -m "add second printf"
```

- In Eclipse and do an **update** on SVNTest. Your changes should show up.  
Right Click SVNTest | Team | Update to HEAD



# Make a change in Eclipse

- Add `printf("I love CS 300!\n");` to `main()`
- Build and run (just to be sure)
- Commit to SVN:  
Right Click SVNTest | Team | Commit  
  
Do NOT commit .o or executable files!

# Check out on Zeus

ssh to zeus.

```
zeus~> mkdir cs300
```

```
zeus~> cd cs300
```

```
ZEUS~> svn checkout svn+ssh://zeus/home/chadd/SVNROOT/SVNTest
```

```
zeus~> cd SVNTest
```

```
zeus~> make clean
```

```
zeus~> make
```

This is how you should test on zeus from now on.

# Show History

- In Eclipse
- Right Click a File
  - Team | Show History

# When things go bad...

- Let's revert ONE FILE back to before the last change.
- Right Click the file to revert
- Replace With | Revision
  - Revisions listed with comments
  - Double-click a revision
- Copy All Non-Conflicting Changes from Right to Left
- Right click on Workspace File Pane | Save
- Next commit will save the changes to a new revision in the repository



# Merge Conflict

- Changes from the repository and the local disk conflict!
- Edit SVNTest.c in Geany again, change “Testing Subversion....\n” to “Let's Test Subversion!”
- From the command line:  
`svn commit -m “change printf”`

# In Eclipse

- Edit SVNTest.c in Eclipse again, change “Testing Subversion....\n” to “I will test Subversion”
- Team | Update to HEAD
- Merge conflict!

# Merge Conflict!

- Changes in the Repository conflict with changes in your local directory

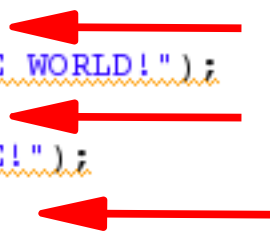
```
Problems Tasks Console Properties
SVN
A /home/chadd/data/workspace/HowDoIRevert2/src/main.c
A /home/chadd/data/workspace/HowDoIRevert2/bin
A /home/chadd/data/workspace/HowDoIRevert2/bin/main
A /home/chadd/data/workspace/HowDoIRevert2/Makefile
Checked out revision 165.
commit -m "changed second message!" /home/chadd/data/workspace/HowDoIRevert2/src/main.c
Sending /home/chadd/data/workspace/HowDoIRevert2/src/main.c
Transmitting file data ...
Committed revision 166.
commit -m "Changed second message to "GOODBYE WORLD!"" /home/chadd/data/workspace/HowDoIRevert/src/main.c
Sending /home/chadd/data/workspace/HowDoIRevert/src/main.c
Transmitting file data ...
svn: Commit failed (details follow):
svn: Commit failed (details follow):
svn: Out of date: '/HowDoIRevert/src/main.c' in transaction '166-1'
```

# Update!

```
#include <stdio.h>

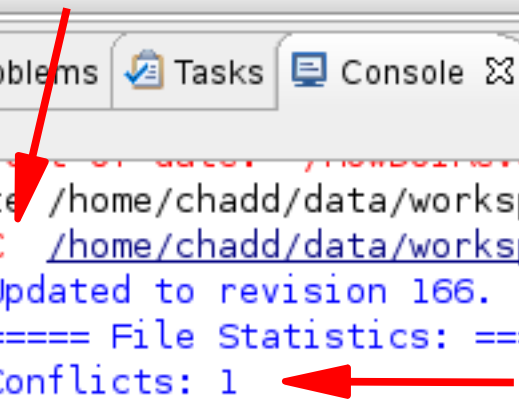
void printer(char *str)
{
    printf("%s\n", str);
}

int main()
{
    printer("HELLO WORLD!");
    <<<<<<< .mine
    printer("GOODBYE WORLD!");
    =====
    printer("GOODBYE!");
    >>>>>>> .r166
}
```



Problems Tasks Console Properties

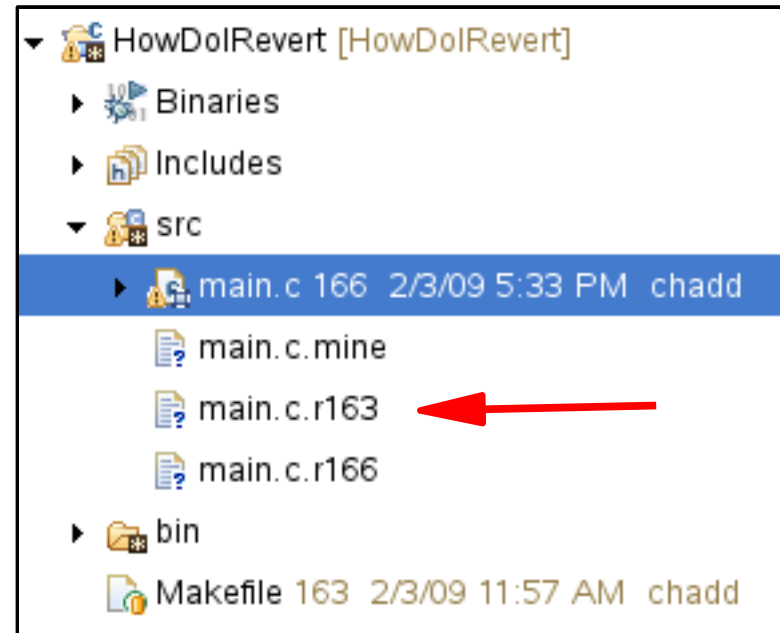
```
SVN
update /home/chadd/data/workspace/HowDoIRevert/src/main.c -r HEAD --force
C /home/chadd/data/workspace/HowDoIRevert/src/main.c
Updated to revision 166.
==== File Statistics: ====
Conflicts: 1
```





# Resolve!

- Edit the source file
  - Remove <<<< >>>>
- Right Click File | Team | Mark Resolved
- Commit



# SVN Keywords

- \$Author\$
- \$Id\$
- Have SVN automatically put the author name or revision information into the file
- <http://blog.gorges.us/2009/03/how-to-enable-keywords-in-eclipse-and-subversion-svn/>
- Linked on the class schedule

# Advanced Subversion

For more use cases see:

Current SVN Notes

Script (svn use cases)

on the following web page

<http://zeus.cs.pacificu.edu/chadd/cs480s11/schedule.html>

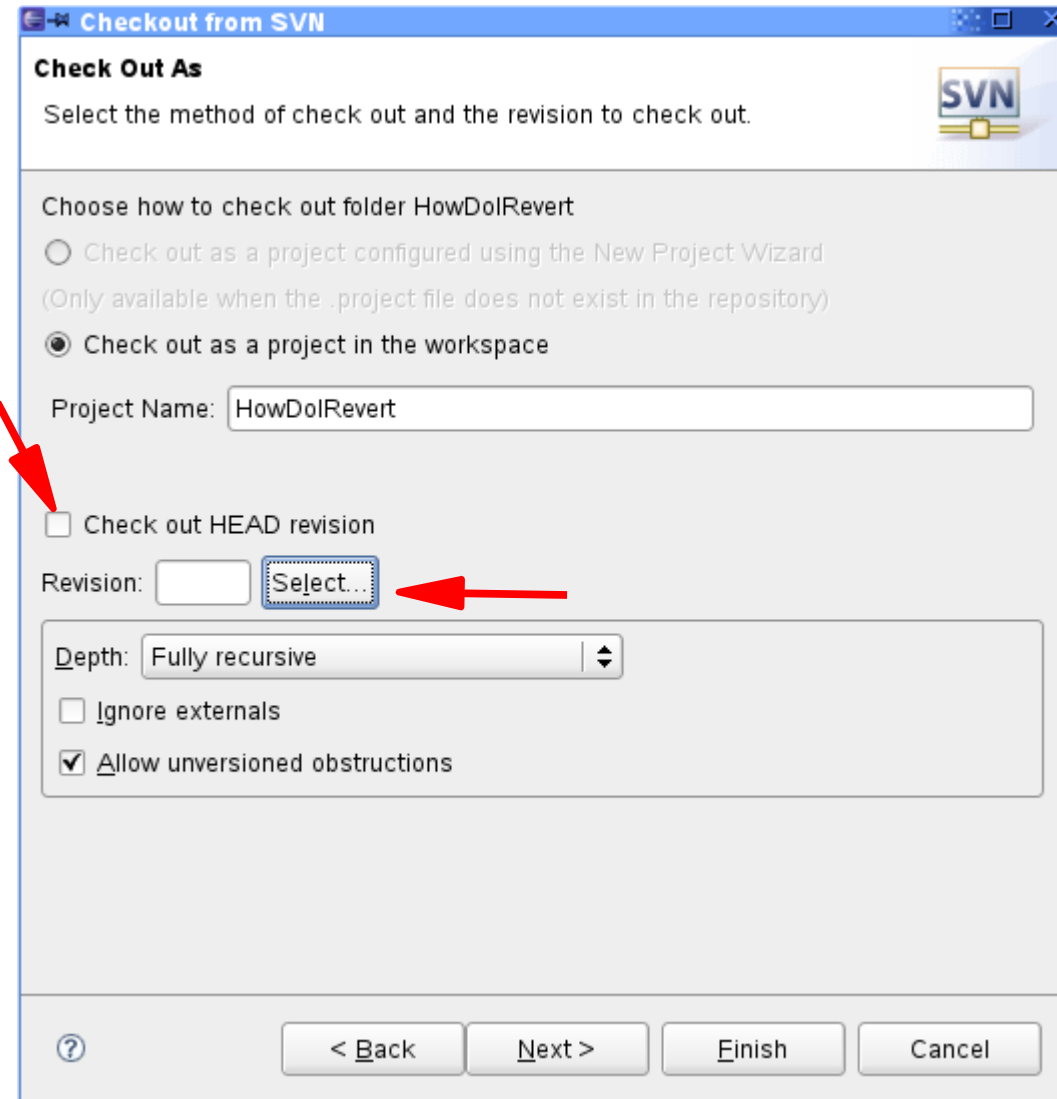


# Revert the **Entire Project**

- Rename existing Project
- Right click Project Name | Rename
- Don't worry, from Eclipse, you cannot destroy your SVN Repository
  - Unless you really, really try

# Revert the Entire Project

- Check out New Project from SVN
- Don't check out the HEAD
- Select Revision



Resource History - HowDoIRevert

R#	Date	Author	Comment
<b>*161</b>	<b>2/3/09 10:36 AM</b>	<b>chadd</b>	<b>changed func name again! I can't decide!</b>
160	2/3/09 10:36 AM	chadd	change func name
159	2/3/09 10:34 AM	chadd	reverted a bunch of stuff!
158	2/3/09 10:26 AM	chadd	added funct prototype
157	2/3/09 10:25 AM	chadd	re-added !
156	2/3/09 10:25 AM	chadd	added funct
155	2/3/09 10:24 AM	chadd	added !
154	2/3/09 10:24 AM	chadd	initial source commit
153	2/3/09 10:23 AM	chadd	Initial import.



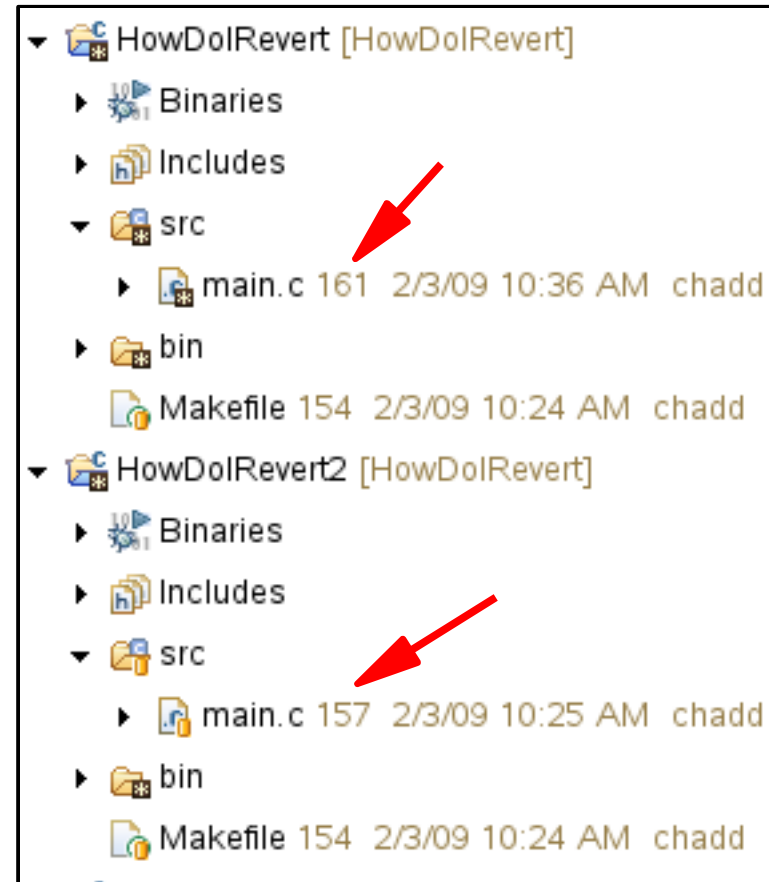
Stop on Copy/Rename



Get All    Next 25    **OK**    Cancel

# Both Projects

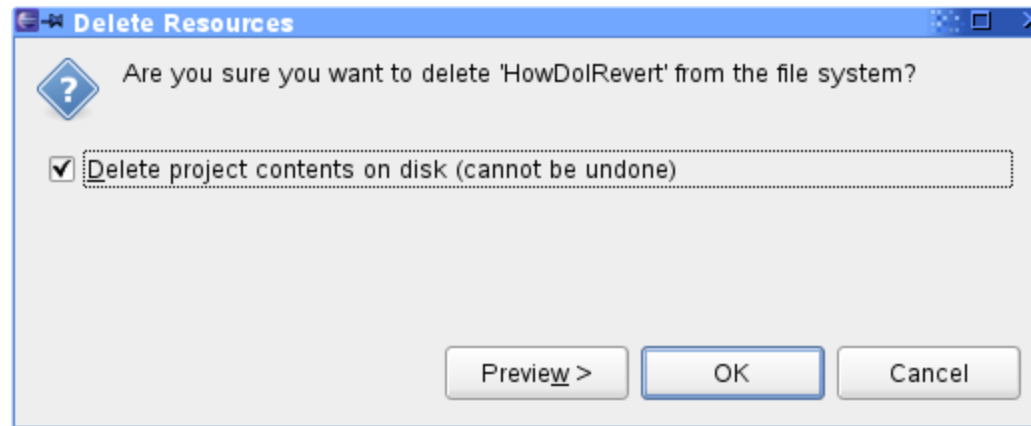
- Project checked out twice
- Different revisions in each project





# Delete Old Project

- Not strictly necessary
- Right click on (Old) Project Name | Delete



- Make sure you delete Project from disk
  - Does not affect Subversion repository
- Close Eclipse and restart
  - To clean up the workspace

# Reverted Project: Commit new Changes

- Update code in project
- Right click Project Name | Team | Synchronize with Repository
- Right click Project Name | Mark as merged
- Go back to C/C++ Perspective
- Right click {File,Project} | Team | Commit