CS270 Recitation 13
"Subversion Source Control"

Goals: Help you to learn subversion (http://subversion.apache.org/), one of the most widely used open source tools for controlling source files in a software project. Here are the specific goals:

- Learn how to create a subversion repository for your source code.
- Practice subversion commands to manage versions of your source files.
- Get ready for PA9, which will require you to submit from a subversion repository.

The Assignment

This assignment will walk you through the process of creating and using a subversion repository. The subversion commands are loaded on department systems, so no installation is necessary.

Step One

Create a repository in your home directory.

linux prompt> cd ~
linux prompt> mkdir Repository
linux prompt> cd Repository
linux prompt> svnadmin create cs270.svn

Note: please use the exact names and paths specified throughout this assignment.

The commands shown above create a repository, which is a database that manages multiple versions of source files. After creating the repository, you don't need to do anything else in the ~/Repository directory. Modifying or removing anything in this directory can compromise your repository and result in the loss of your source code!

Step Two

Make a PA9 directory and copy your PA7 code into it, renaming files as shown below. Next import the directory into the repository. This command puts the PA9 directory under source control.

linux prompt> mkdir ~/PA9
linux prompt> cd ~/PA9
linux prompt> cp -R ~/PA7/* .
linux prompt> make clean
linux prompt> mv lc3parse.c lc3assemble.c
linux prompt> mv lc3parse.h lc3assemble.h
linux prompt> svn import . file:///$HOME/Repository/cs270.svn -m "Initial import"

Step Three

Remove the old contents of your PA9 directory. This seems dangerous, but don't worry, the repository has your source code stored safely.

linux prompt> rm -rf ~/PA9/*
linux prompt> svn co file:///$HOME/Repository/cs270.svn .

Step Four

Your files are now source controlled. Modify the lc3assemble.c and lc3assemble.h module headers by changing the date, name of the assignment, or adding documentation. Modify the Makefile to build the new files. Now you can see how your files have changed by using the **svn diff** command to ask subversion for the difference between the repository version of the file and the one in your working directory. This is one of the advantages of source control, it lets you compare different versions of files. Try an individual file and the whole directory:

linux prompt> svn diff lc3assemble.h
linux prompt> svn diff lc3assemble.c
linux prompt> svn diff Makefile
linux prompt> svn diff .

Step Five

Imagine at this point that you are content with the changes in lc3assemble.c and lc3assemble.h. This doesn't mean you are done working on the files, just that the changes you have made have been checked and seem correct. The subversion command **svn commit** updates the repository with your changes. Usually you would do this when you have finished some code, and the program compiles and the new functionality is working. This is another feature of source control; it allows you to decide when to commit changes to a source file. Note that after the commit, asking for the difference between the working directory and repository gives you nothing, since there are no longer any differences:

linux prompt> svn commit . -m "Changed comments"
linux prompt> svn diff .

Step Six

Try making some random changes to a couple files. Do something that is obviously wrong and keeps the program from compiling. Look at the differences and then throw away the changes you have made by using the **svn revert** command to revert the files you have changed. This show another feature of source control, which is to allow you to back up to the last version of the files that have been checked in. This is why you want to check in a file whenever you get something new working. Once again there is no difference between the working directory and the files in the repository:

linux prompt> svn diff .
linux prompt> svn revert lc3assemble.c
linux prompt> svn revert lc3assemble.h
linux prompt> svn diff .

Step Seven

Try a few more of the commands available in subversion. The **svn log** command tells you the history of revisions to a file, meaning all the versions that have been checked in. Of course you can also ask for a difference between any two of these versions. The **svn update** fetches the latest version of the code and tells you the version number, this is mostly useful when sharing a repository with someone who is also making changes to the source.

linux prompt> svn log lc3assemble.c
linux prompt> svn update .

Step Eight

In addition, subversion allows you to add new files to the repository with **svn add**, and delete files you no longer need with **svn delete**. These commands do not add or delete the files from the repository until you commit the changes. A single **svn commit** command processes pending adds, pending deletes, and checks in modified files. The **svn status** commands tells which files are scheduled for addition or deletion. Try the following series of commands to add a new file and commit the change, then delete the same file and commit the change:

linux prompt> cp utilities.c temp.c
linux prompt> cp utilities.h temp.h
linux prompt> svn add temp.*
linux prompt> svn status
linux prompt> svn commit . -m "Added files"
linux prompt> svn status
linux prompt> svn delete temp.*
linux prompt> svn status
linux prompt> svn commit . -m "Deleted files"
linux prompt> svn status

Step Nine

To get checked off, show the teaching assistant your repository, then go to your working directory and shown him a log message for a file that has multiple versions. Also show him how you modify a file, display the difference with the repository, and commit the changes. Going forward you must checkin your files each time you work on the project, and the files lc3assemble.c and lc3assemble.h must show at least 8 revisions, the output of the svn log command will be part of the submission for PA9.