Handout 3: Version Control Basics

Introduction

SVN (Subversion) provides the following functionality:

- It allows multiple users to edit the same files independently, working on their own copies and synchronizing their work.
- It provides backups, permitting you to recreate the state of your files at any moment in the past. This is valuable if your home computer crashes (or is stolen). It is also valuable if you make some edits, then decide they were a bad idea and you want to recover a previous version.
- We use SVN to manage homework assignments. When you submit through the Homework Server, the Homework Server collects the current version of your files from your repository.

SVN works as follows. There is a “repository” containing the master version of the files, including a history of all previous versions. Each user “checks out” a working copy of the files.

Each user can edit his or her copy of the files arbitrarily, without affecting other users or the master version.

- To store your version of the files in the repository, you “check in” (a.k.a. “commit”) your files.
- To incorporate others’ changes into your working copy, you “update” your copy.
- You can perform many other operations, such as comparing versions of the files or reverting to a previous version if a change introduces a bug.

Setup: Checking out the csci2600 project from SVN

You always edit your own personal copy of files that are under SVN control. Before you can make such edits, you must “check out” your own copies of the repository’s master files. You should only need to do this step ONCE at the beginning of the term. If you are tempted to use this command at other times, you most likely want SVN's “update” command.

The following instructions assume you wish to check out your repository which is located at URL https://csci2600svn.cs.rpi.edu/YourRCSID.

The following instructions should show how to use Eclipse and Subclipse to check out SVN projects. Before doing so, please check to make sure that you have properly set up the Eclipse environment.

1. Goto File » Import
2. Select SVN » Checkout Projects from SVN
   - If that option does not exist, then you may not have installed Subclipse properly!
3. Select “Create a new repository location” and click “Next”
4. Enter the URL https://csci2600svn.cs.rpi.edu/YourRCSID.
   - YourRCSID is your RPI email id, e.g., thompw4, not your RIN.
5. Click Next.
6. Enter your RCSID and password (this is your RPI id, e.g., thompw4, and password) in the “Enter SSH credentials” dialog and click OK.
7. In the “Select Folder” dialog, select the https://csci2600svn.cs.rpi.edu/YourRCSID
directory and click Finish.

Important: You must select the parent folder, https://csci2600svn.cs.rpi.edu/YourRCSID, NOT the src subfolder!

8. In the “New Project” dialog that appears, select the Java->Java Project wizard, click Next (this will start the creation of a new Java project)

9. In Project Name, type "csci2600", then click Finish. In Project Explorer you should see a project named csci2600 [https://csci2600svn.cs.rpi.edu/YourRCSID] with subdirectory src as well as other directories showing the libraries you have included in your project. (Typically, you will see just the JRE Library System. Later, you will include the JUnit library as well, as described here.) In directory src you should have a directory hw0, the first homework assignment. Open the description of the homework: .../hw0/docs/hw0.html.

IMPORTANT: Make sure that you have created the correct directory structure. If you break the structure, compilation on the Homework Server will fail resulting in a grade of 0. At this point, you must have project csci2600 with subdirectory src. Directory src must have subdirectory hw0 and hw0 must have subdirectories answers, docs and test. These show as hw0, hw0.answers, hw0.docs and hw0.test subfolders of src in Project Explorer.

Homework assignments are structured into packages (package hw0, hw3, etc.) Code must be in the main package directory (e.g., in Homework 0, all *.java code is under hw0), JUnit tests must be in the test subdirectory (all test cases are under hw0.test) and your text answers to questions must be in answers (all text files should be created under hw0.answers).

Adding and committing files in Eclipse

After making changes to, adding, or removing files, you must "commit" your changes to SVN. This step will cause SVN to record your changes to the repository, so that your changes are backed-up and available to other people working on the repository, or to you when working on a different computer system.

Working with SVN in Eclipse is generally quite easy. (Subclipse simplifies many of the steps.) To commit a directory (for example, hw0 after you have edited the files), right-click on that directory, then select Team » Commit. With Eclipse you do not need to worry about adding new files. Eclipse will automatically ask you if you want to add or ignore specific files when you run commit on the directory that contains the new files.

Note: if you create a file or folder in an Eclipse project but via the command line (or any mechanism outside Eclipse), then you will need to “refresh” the package explorer for Eclipse to recognize the change. To do this, right-click on the project name in the package explorer and select the Refresh item.

It is a good idea to commit your changes frequently. It backs up your work, thus enabling you to revert to an earlier version of your code if you find yourself going down a wrong path. Also, when you are working with others, it minimizes conflicts.

Updating files

SVN’s “update” command updates your local copy of files to reflect changes made to the repository by other people (or by you when working on a different computer system). The only changes made by people other than you will be the changes made by the Principles of Software staff when adding new homeworks to your repositories.

In the Package Explorer window, right-click on a file or directory then select Team » Update. If the item is
not a directory, just that file will be updated; otherwise, everything inside the directory will be updated.

If you check out to a single local directory in Eclipse, you should not worry about merges and conflicts. But if you check out multiple local versions of your repository and you make changes to these versions, then you may need to merge and resolve conflicts.

To minimize the possibility of conflicting changes being made simultaneously, you should update frequently and commit frequently.

For more information on SVN, including how to work from the command line, see the following reference.

Parts of this handout are copied from University of Wanshington's Software Design and Implementation course.