Version Control with SVN

For IMAT 5314 MSc Project Students

*Apache Subversion* (often abbreviated SVN, after the command name svn) is a software versioning and revision control system distributed as free software under the Apache License. Developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation. DMU uses the TortoiseSVN client interface.

Students doing projects in the Faculty of Technology may, if they wish, use SVN for storing and managing their code and/or their project documents including their final reports or dissertations.

If you want to use the Faculty of Technology TortoiseSVN repository, you should ask Andrew Curry (whom you should contact by email at [ajc@dmu.ac.uk](mailto:ajc@dmu.ac.uk)) for an account.

The DMU repository server is at ***svn.dmu.ac.uk***. (Some old documentation may mention svn.tech.dmu.ac.uk, but this is out of date.)

**The Manual**

The online SVN manual is at <http://svnbook.red-bean.com/>

It is hard to overstate the importance of this manual when using SVN. Chapter 2 (Basic usage) and chapter 9 (Complete reference -> The Subversion Command-Line Client: svn -> svn Subcommands) are probably the most useful.

**Your Subversion repository**

Each project student (who asks) will have his or her own Subversion repository. This will allow project development work (code) and documentation (especially the report but also other documents such as the terms of reference) to be stored under version control. This will allow you to "go back" to older versions if you need to.   
  
It will also permit the supervisors to see what actual progress is being made. In particular, it gives a robust story of the development of your project when it comes to write up. It also gives a strong sense of the provenance of your work, and thus an easy defence of the work being your own.  
  
If we assume your p-number is p99999999, then your repository will be at the url  
https://svn.dmu.ac.uk:3690/p99999999/  
  
You will be able to read and write to your own repository. The project supervisors and MSc project module coordinator will have read only access.  
  
More detail on how the repositories should be used and organised is in "Using Subversion" in this folder.

**Using SVN**

Concepts (see chapter 1 of the manual for a fuller description):

On your client machine, there are two types of file / directory:

* ***unversioned*** (everything that has no relationship with a repository – typically all existing files and folders)
* ***working copy*** (directories and their recursive content which have been checked out of a repository – a working copy directory contains meta-data, hidden within a .svn directory)

On the server, there may be many different repositories. Each repository is effectively a tree of files and directories, together with their history. Whenever you change anything in the repository, you must supply a message which explains (to you at some point in the future) why the change was made.  
  
Typical workflow between client and server:

1. import unversioned directory (and contents) from client to repository. Using the command-line client this is:  
     
   **svn** **import** *<src-unversioned-dir-on-client> <dest-dir-in-repository>*  
     
   where

*<src-unversioned-dir-on-client>* is typically . (dot i.e. the current directory) and

*<dest-dir-in-repository>* would be something like https://svn.dmu.ac.uk:3690/p09876543/imat3451/trunk/  
  
Make sure that the only things in the source directory are things that you want in the repository; once in the repository, they can never be fully deleted (that is the point of the repository after all).  
  
The directory on the client machine is still unversioned but there is now also a copy in the repository under version control.  
  
2) Checkout a brand new working copy of a directory (and its subdirectories) from the repository to the client machine. These will be under version control (look in the directory listings that you check out and you find .svn directories in each one - it is these that make it a working copy). Go to a different directory from the one where you did the import in 1 above. To checkout the command would look something like  
  
**svn checkout** *<src-dir-in-repository> <working-copy-dir-on-client>*  
  
where  
*<src-dir-in-repository>* is something like  
https://svn.dmu.ac.uk:3690/p09876543/imat3451/trunk/  
and the optional  
*<working-copy-dir-on-client>*  
is the name that you want to give this directory on your client. If omitted, it keeps the same name ("trunk" in this case) as it had in the repository.  
  
3) Edit the files in the working copy, just as you would if they were not under version control, with the following three exceptions:

3.1) If you create a new file or directory that you want to go in the repository, you will need to explicitly add it to your working copy  
  
**svn add** *<some-new-file-or-dir>*  
  
3.2) If you want to remove a directory or file from the repository (remember, it doesn't really go, it will just removed from the HEAD or most up to date version when you next commit your changes), then  
  
**svn del** *<some-file-or-dir-that-you-don't-want>*  
  
3.3) If you want to move a directory or file within the repository, then  
  
**svn mv** *<old-name>* *<new-name>*  
  
4) When you have completed a logical chunk of editing on your working copy, then commit it to the repository   
  
**svn commit**   
  
At this point it is worth deleting the original unversioned copy (the one that you first imported) so you don't accidentally work on it.  
  
5) When you return to carry on editing the working copy, the first thing you usually want to do is to update this working copy with any changes that you may have made in the repository whilst editing a working copy on a different machine. (You can have working copies of the same repository all over the place - laptop, desktop, faculty PC, faculty linux box, H: drive, C: drive etc etc).  
  
**svn update**  
6) Go back to 4 and do the edit / add / del-> commit -> update -> edit / add / del -> commit -> update as long as you wish.   
  
7) If you finally want to get an unversioned copy (without all the .svn dirs), then go to an otherwise empty directory and use:  
  
**svn** **export** *<src-dir-in-repository> <dest-unversioned-dir-on-client>*  
  
8) Useful odds and ends.

8.1) To see the status (added, changed, unknown, deleted are the main ones) of the files and dirs in the working copy:  
**svn status**  
  
8.2) To see the line by line changes between the working copy and the last update:  
**svn diff**  
  
8.3) To undo local edits (because you messed up - you could of course just use the operating system to delete the working copy, then checkout a fresh one)  
**svn revert**  
  
8.4) At some point you will forget to update before you start work. You will then have to resolve conflict. Read the red-bean SVN manual!!!  
  
8.5) You can see the HEAD (most up to date) version of your repository by simply typing the URL of your repository into a browser. You will be challenged for your faculty user-id and password before the content is revealed. You cannot put material into your repository via the browser.