

Using the **Subversion** Version Control System

(these instructions courtesy of Dr. Bryan Jones, Ryan Irwin (SECON '07), David Warren (SECON 07)).

Subversion is a very flexible version control system for practically any type of electronic data. It offers more than just version control; it provides a great way for senior design teams to share electronic documents with each other by creating a common *repository* that all team members have access to. The learning curve on Subversion is not steep; it is **strongly** suggested that you take the time to learn how to take advantage of this extremely useful tool (I am requiring all Senior Design 1 teams to use Subversion; you do not have to use it in Senior Design 2 if you feel like it was not helpful).

Client, Server, Repository:

Subversion consists of three parts:

- The *repository* is the location on the Subversion server where the shared electronic files are stored. In ECE, any file space accessible from yavin.ece.msstate.edu can serve as a repository. All team members must have read/write access to this file space in order to share documents. Documents in a repository are versioned controlled; you can have multiple repositories.
- The subversion *server* is the code performs operations on the repository items on the server. The subversion server in ECE is yavin.ece.msstate.edu.
- The subversion *client* runs on a remote machine and checks data out of the repository to create a local copy of the data; the user can edit the data locally then check the data back into the subversion server. There are different clients available; the Tortoise SVN is the client recommended for the Windows OS.

To create a repository:

It is recommended that you use the file space under your team's WWW directory on yavin. You must use the '*svnadmin*' command to create the repository; this is only done one time:

1. Login into Yavin using your ECE account
2. Navigate to your team's WWW directory and create a repository somewhere under that directory. The following code makes a repository space called "repos" in the *data/* subdirectory:

```
cd /data/www/htdocs/courses/design/2006/fall/drum_pad
mkdir data
cd data
svnadmin create /data/www/htdocs/courses/design/2006/fall/drum_pad/data/repos --fs-type
fsfs
```

The `--fs-type fsfs` argument is important; this specifies that the repository type be a file-system type instead of a Berkeley database format (the database format has been found to be problematic).

Installing the Tortoise SVN client

1. Download and install the Tortoise SVN client from:

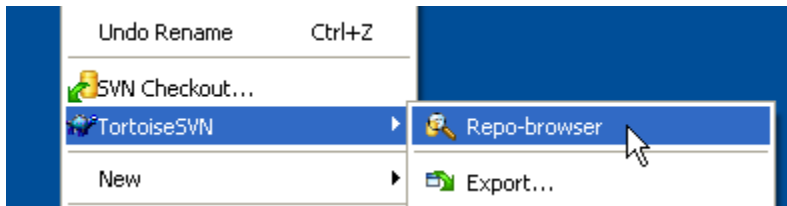
The following instructions will install a subversion shell on your computer and then access your repository.

<http://prdownloads.sourceforge.net/tortoisesvn/TortoiseSVN-1.3.5.6804-svn-1.3.2.msi?download>
<http://tortoisesvn.net/downloads>

On a windows machine, download the file by clicking on the top link. If the link is broken, download the TortoiseSVN-xxxx.msi 32-bit Installer.

Follow the installer instructions.

2. Once the client is installed, you can right click, and select the repo-browser from the TortoiseSVN:

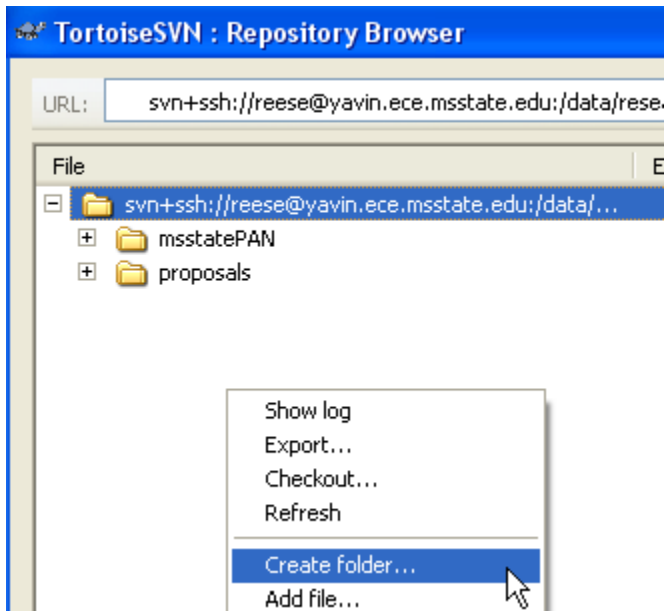


Type in the name of your repository as:

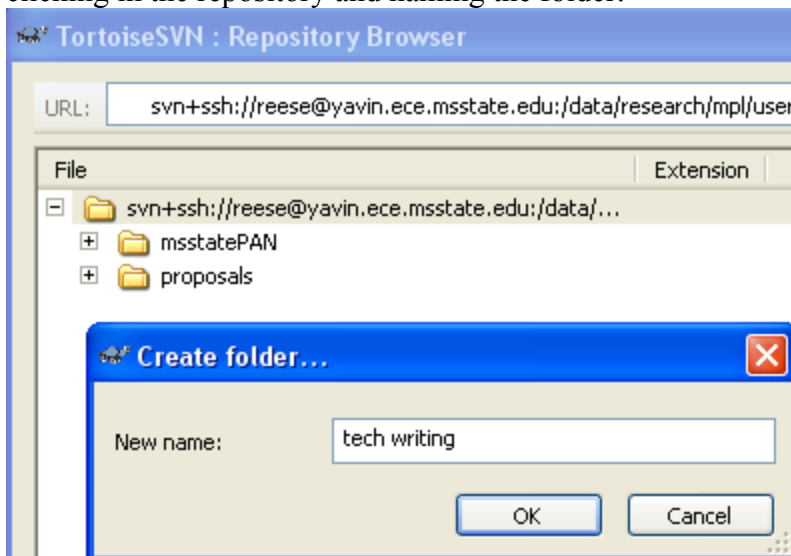
```
svn+ssh://NETID@yavin.ece.msstate.edu:/data/www/htdocs/courses/design/2006/fall  
/drum_pad/data/repos
```

where NETID is your yavin login name.

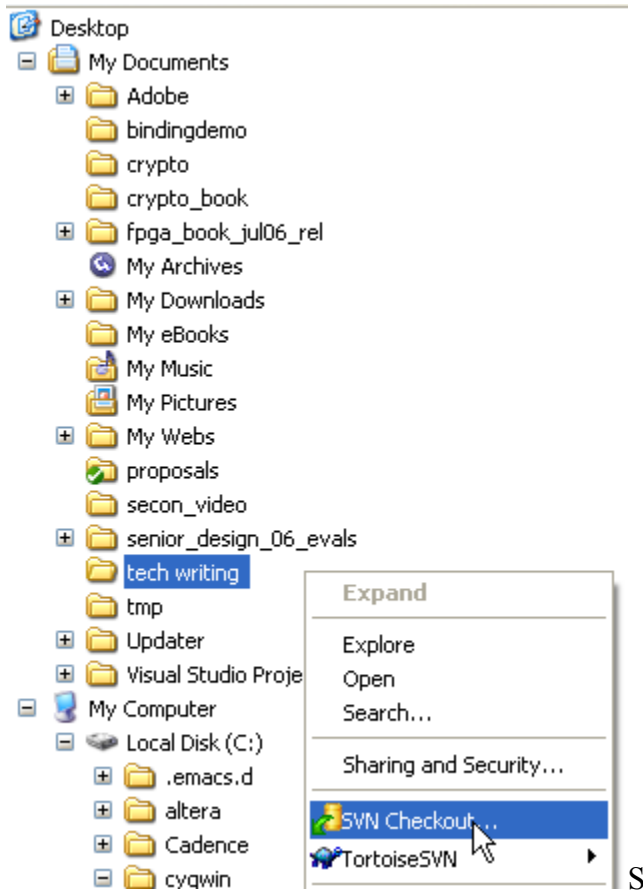
You should see the repository browser pop-up:



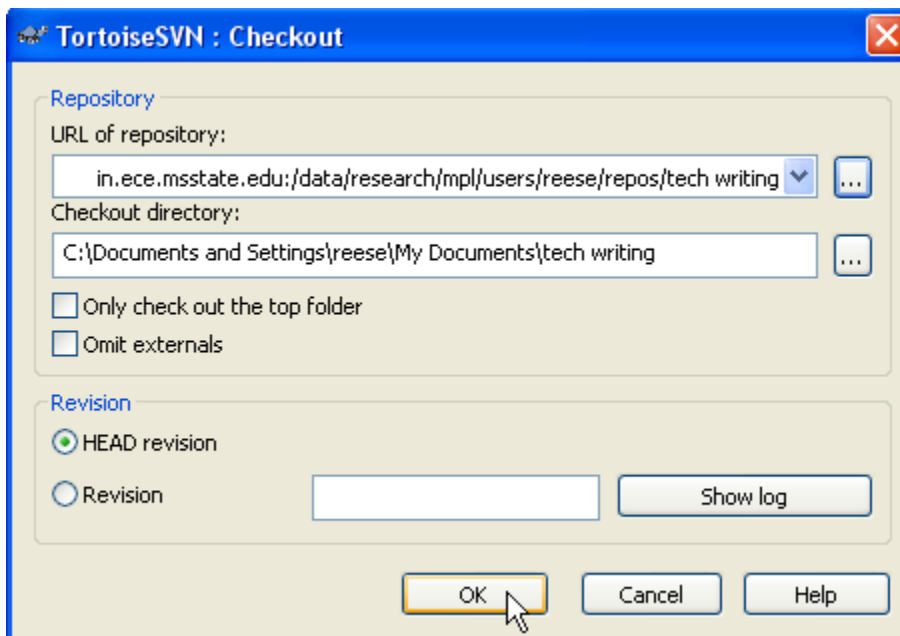
If a team member has not already created a folder to work in, you can create one by right clicking in the repository and naming the folder:



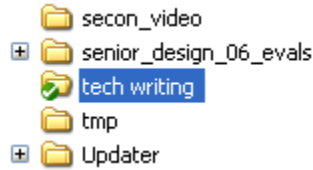
Create the same folder on a local directory, perhaps in My Documents; then select the folder, right click and select *SVN Checkout* so that you can check out a local copy of this to your machine:



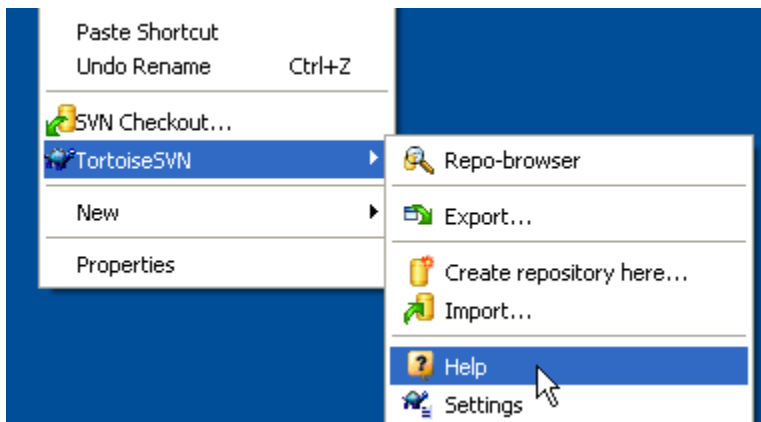
Once the checkout window appears, in the “URL of repository” line browse to the folder in the repository so that the same folder appears in both “URL of repository” and “Checkout Directory”. Then click on OK:



After you do this, you will see a *Green Check* (an *icon overlay*) appear on the directory, this means that this is a freshly checked out version without any changes:



See the Subversion HELP, section “Daily Use Guide →Getting Status Information →Icon Overlays” for the meaning of other icon overlays (sometimes the overlay icons do not update unless you close down the explorer window and reopen it).



When you do a checkout of data from the repository, the checkout should be to an empty directory because it will not overwrite pre-existing files.

Using the Tortoise SVN Client

The HELP for the Tortoise SVN is very readable; look in the “Daily Use Guide” for explanations on the most commonly used commands in Subversion. The basic cycle for subversion is *Update* (which refreshes your local copy with changes made to the repository), followed by *Commit*, which sends your changes to the repository. You must perform an Update before you do a Commit so that you have an up-to-date version of the data.

If more than one team member is modifying a file, then subversion has a visual merge mechanism that allows you to merge your changes with the other person’s changes (See the documentation for more details). However, the visual merge mechanism only works for text files. For binary files (such as Microsoft *.doc* files), team members should use the lock mechanism in Subversion to lock a file before editing it; that prevents other team members from changing the file while it is locked.

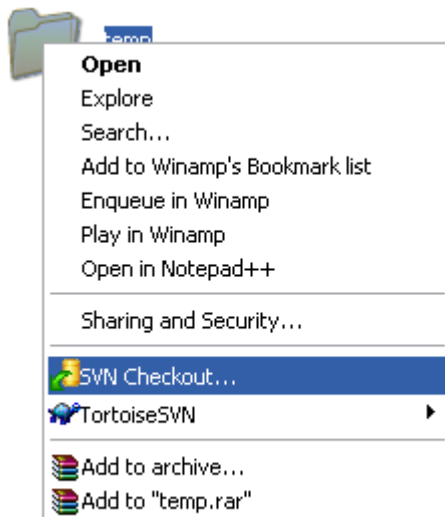
You can use the log and statistics menus to see how often particular team members modify files.

Also, there are ways to prevent Subversion from versioning tool-generated files such as object files, list files etc, since you generally only want version control over the source files, and not the tool-generated files (see the help documentation on “Daily Use Guide → Ignoring Files and directories”).

SVN – Setup and Password Caching

This procedure allows frees you from having to type in your password every time you access the repository.

1. You need to download
 1. [Tortoise SVN](#)
 2. [pageant.exe](#)
 3. [puttygen.exe](#)
2. Install SVN
3. Open windows explorer, right click on a folder and click on SVN Checkout to download material from the selected repository.

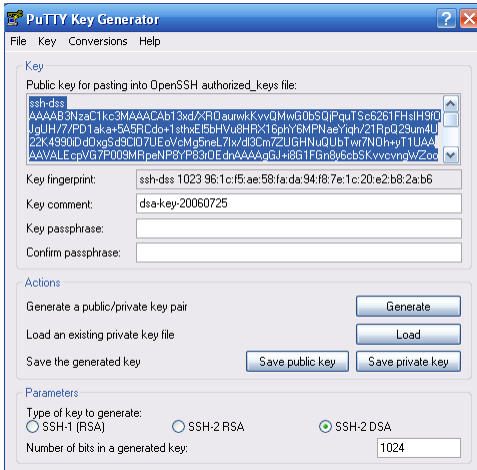


4. Open puttygen.exe (PuTTY Key Generator) to generate a key pair.
4. Select SSH-2 DSA radio button in parameters section.



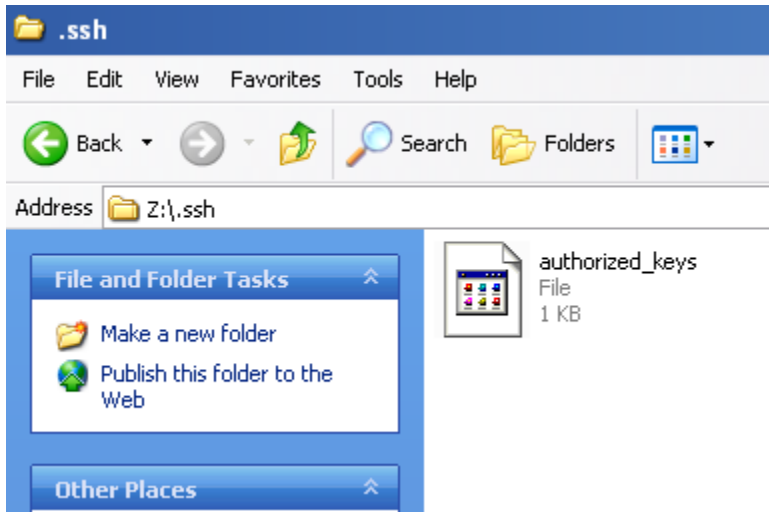
6. Click Generate to generate a public and private key pair. Keep this window open.

7. Open blank text document in notepad and paste all the content in the section “public key for pasting into OpenSSH authorized_keys file” in to the text file.

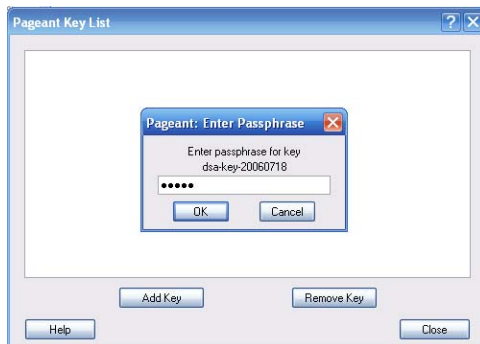


8. Look for the ~/.ssh directory (it is hidden) in your home directory of yavin. If you can't find it create one (the filename is “dot”ssh, “.ssh”).

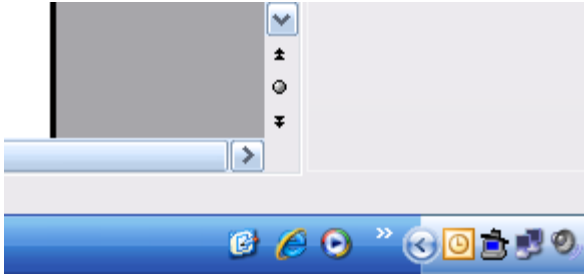
9. Save the text file with the name “authorized_keys” (without any extension no .txt) in the .ssh directory.



10. Now go back to PuTTYgen window and type some key passphrase. This serves as a password for your key.
11. Click on “save private key”... save the key to some location on local hard drive. (It will be saved with an extension .ppk)....now, close the window.
12. Open Pageant.exe
13. Click on addkey, add the private key saved in step 11.type in the key passphrase you entered in step 10.



14. You can see the Pageant running on the taskbar. Every time you connect to SVN, Pageant will provide the private key that matches the public key in the server and logs you in without asking for a password.



15. Don't forget to initialize Pageant (steps 12&13) every time you restart your computer.

The procedure has been tested with:

Microsoft Windows XP Professional (build 5.1.2600)

PuTTY utilities v0.58

TortoiseSVN v.1.3.5 (build 6804)

Courtesy: http://www.cleversafe.org/wiki/Caching_passwords_with_tortoisesvn