Code development using Subversion

- Subversion is a version control system that can be used to keep track of different versions while developing code.

- We have used it so far to checkout pieces of code from the OpenFOAM-extend project at SourceForge

- In the following slides you learn the basics of setting up and using your own Subversion *repository*.

- For detailed information, see:
  `http://subversion.tigris.org/`
  `http://svnbook.red-bean.com/`

- For a similar system, named Git, see:
  `http://git.or.cz/`

- For a translation between the two systems, see:
  `http://git.or.cz/course/svn.html`

Why use a version control system?

- Many users developing the same code have access to the same files.

- A single user working on the same files on different computers.

- Keeps previous versions for future needs.

- Tracks originator of files, including commit comments.

Create a Subversion *repository* and check out a *working copy*

- We will now create a new Subversion *repository*, check out a *working copy* of it and add files to it

- Create the Subversion *repository*:

```
mkdir $HOME/SVN_repositories
svnadmin create $HOME/SVN_repositories/myDevelopments
```

- Checkout a *working copy* of the *repository*:

```
mkdir $HOME/SVN_workingCopies
cd $HOME/SVN_workingCopies
svn checkout file://$HOME/SVN_repositories/myDevelopments
```

What do we have in our SVN *repository*?

- Investigate the *working copy*:
  ```
  cd myDevelopments
  ls -a
  ```
  The folder is empty except for the `.svn` directory. The files in that directory contain svn info. In particular the `.svn/entries` file.

- `svn list`
  This gives no output, so the *repository* is empty.

- `svn status`
  This gives no output, so the *working copy* is also empty.

Let's add the icoFoam solver to the *working copy*

- ```
  cd $WM_PROJECT_DIR
  cp -r --parent applications/solvers/incompressible/icoFoam \
        $HOME/SVN_workingCopies/myDevelopments
  cd $HOME/SVN_workingCopies/myDevelopments/applications/solvers/\
  incompressible/icoFoam/
  wclean
  rm -rf Make/linux*
  cd $HOME/SVN_workingCopies/myDevelopments
  svn update
  svn list
  svn status
  ```

- We see that the *repository* is still empty, but the *working copy* has a new directory named
  `applications`. The question mark tells us that it is not in the *repository*.

Add a directory to the *repository*, including all its contents

- We must *add* the `applications` directory to the *repository*:

```
svn add applications
svn update
svn list
svn status
```

- The *repository* is still empty but the 'A' tells us that the directory and all its contents has been added. We must commit the changes in the *working copy* to the *repository*:

```
svn commit -m "Added icoFoam"
svn update
svn list
svn status
```

- We see that the *working copy* is at the same revision as we just committed.

- We see that the *repository* contains the `applications` directory.

- We see that the *working copy* has no differences compared with the *repository*.

Make a modification and commit to the repository

- Do some modification to the `icoFoam.C` file:

```
echo " " >> applications/solvers/incompressible/icoFoam/icoFoam.C
svn update
svn list
svn status
```

- The 'M' tells us that the `icoFoam.C` file has been modified compared with the repository.

- Commit the change to the repository:

```
svn commit -m "Added space at end of icoFoam.C"
svn update
svn list
svn status
```

- We see that the *working copy* is at the same revision as we just committed.

- We see that the *repository* contains the `applications` directory.

- We see that the *working copy* has no differences compared with the *repository*.

Check out another *working copy* and delete the first one

- You can have as many *working copies* as you like:

```
cd $HOME/SVN_workingCopies/
svn checkout file://$HOME/SVN_repositories/myDevelopments \
    anotherWorkingCopyOfMyDevelopments
```

- Both *working copies* are now identical (except for some details in the `.svn` directories)

- If you modify something in one of the *working copies*, you must `update` the other one.

- When you don't need a *working copy* anymore, you can simply delete it without affecting the *repository*:
```
cd $HOME/SVN_workingCopies
rm -rf myDevelopments
```

- You can continue working with the other *working copy*

- When you don't need the repository anymore you can simply delete it:
```
cd $HOME/SVN_repositories
rm -rf myDevelopments
```

- Note then that any remaining *working copies* can of course no longer update or commit.

Further svn commands

- `svn help <subcommand>`

- Note that when you want to add, remove or move files in the svn *repository* you must use svn commands:

  ```
  svn add <file>
  svn remove <file>
  svn move <file> <newfile>
  ```

- Remove all the `.svn` directories in a *working copy* by *standing in the root of the working copy* and typing:
  ```
  find . -name .svn -exec rm -rf {} \;
  ```
  Be EXTREMELY careful when you use this RECURSIVE command!!!

- If you are not sure how a specific svn command should be used, create a temporary *repository* and *working copy* for testing purposes.

Remote access to a *repository*

- Create a repository at `/chalmers/groups/am-kurs-os2010`:

  ```
  ssh remote1.student.chalmers.se
  svnadmin create \
      /chalmers/groups/am-kurs-os2010/SVN_repositories/${USER}Repository
  exit
  ```

- Remote access to the *repository* (you might have to type your password twice!):

  ```
  svn ls svn+ssh://<CID>@remote1.student.chalmers.se\
  /chalmers/groups/am-kurs-os2010/SVN_repositories/<CID>Repository
  svn co svn+ssh://<CID>@remote1.student.chalmers.se\
  /chalmers/groups/am-kurs-os2010/SVN_repositories/<CID>Repository
  ```

- Note that if you go inside the checked-out `<CID>Repository` you can now do:

  ```
  svn list
  ```
  (i.e. without the rest of the above line - 'ls' = 'list')
  The file `.svn/entries` keeps track of where the original *repository* is located.

- Keep this repository for exchange of files during supervision of your project!

- Make sure to only add the necessary files.

- Don't add large binary files!!! We share this 1GB disk space.