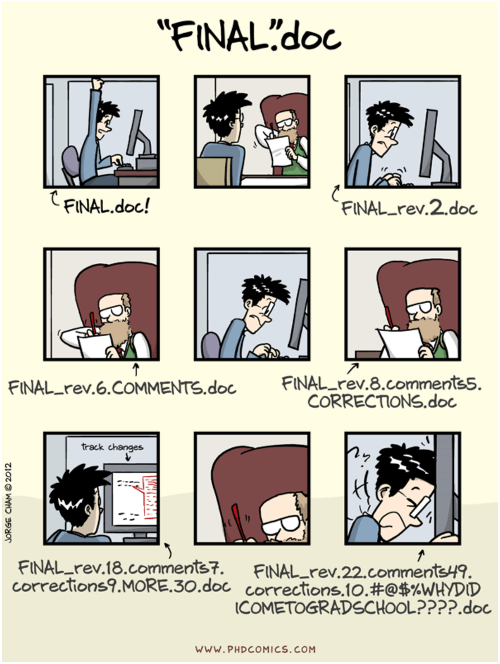
Mercurial (hg)

Introduction

Learning Goal

1. Explain when and why you should use version control



"Piled Higher and Deeper" by Jorge Cham, http://www.phdcomics.com

A Better Kind of Backup - Part 1

1. Explain which initialization and configuration steps are required once per machine, and which are required once per repository.
2. Add files to Mercurial's collection of tracked files.
3. Go through the modify-commit cycle for single and multiple files and explain where information is stored before and after the commit.
4. Identify and use Mercurial revision numbers and changeset identifiers.
5. Compare files with previous version of themselves.

- Mercurial.ini (windows)
- /.hgrc (Linux/Mac)
- mkdir planets
- cd planets
- hg init
- ls -a
- hg verify

- nano mars.txt
- hg status
- hg add mars.txt
- hg commit -m “Starting...”
- hg log
- hg diff

Mercurial.ini for Windows

Create a new file called %USERPROFILE%\Mercurial.ini (that's spelled $USERPROFILE/Mercurial.ini if you are in gitbash)

```
[ui]
username = Vlad Dracula <vlad@tran.sylvan.ia>
editor = nano

[extensions]
color =

[color]
mode = win32
```

A Better Kind of Backup - Part 1

1. Explain which initialization and configuration steps are required once per machine, and which are required once per repository.
2. Add files to Mercurial's collection of tracked files.
3. Go through the modify-commit cycle for single and multiple files and explain where information is stored before and after the commit.
4. Identify and use Mercurial revision numbers and changeset identifiers.
5. Compare files with previous version of themselves.

- Mercurial.ini (windows)
- /.hgrc (Linux/Mac)
- mkdir planets
- cd planets
- hg init
- ls -a
- hg verify

- nano mars.txt
- hg status
- hg add mars.txt
- hg commit -m "Starting..."
- hg log
- hg diff

A Better Kind of Backup - Part 2

5. Compare files with old versions of themselves.
6. Restore old versions of files.
7. Configure Mercurial to ignore specific files, and explain why it is sometimes useful to do so.

- hg diff --rev 1:2 mars.txt
- hg diff -r 0:2 mars.txt
- hg diff --change 1
- hg revert mars.txt
- hg revert --rev 0 mars.txt
- hg status

- mkdir results
- touch a.dat b.dat c.dat results/a.out results/b.out
- hg status
- nano .hgignore
- hg status --ignored

.hgignore

```
syntax: glob
*.dat
results/
```

A Better Kind of Backup - Part 2

5. Compare files with old versions of themselves.
6. Restore old versions of files.
7. Configure Mercurial to ignore specific files, and explain why it is sometimes useful to do so.

- hg diff --rev 1:2 mars.txt
- hg diff -r 0:2 mars.txt
- hg diff --change 1
- hg revert mars.txt
- hg revert --rev 0 mars.txt
- hg status

- mkdir results
- touch a.dat b.dat c.dat results/a.out results/b.out
- hg status
- nano .hgignore
- hg status --ignored

Exercise

Create a new Mercurial repository on your computer called bio. Write a three-line biography for yourself in a file called me.txt, commit your changes, then modify one line and add a fourth and display the differences between its updated state and its original state.

Collaborating

1. Explain what remote repositories are and why they are useful.
2. Explain what happens when a remote repository is cloned.
3. Explain what happens when changes are pushed to or pulled from a remote repository.

- hg paths
- hg push
- hg pull
- hg clone
- hg log --graph
- hg update

We're going to explore collaborating via a remote repository clone on Bitbucket by pretending that we are going back and forth between our home and work computers. We'll simulate that by creating a directory for each location and moving our `planets/` repository into the work computer directory.

```
$ cd
$ cd Desktop/swc/
$ mkdir home-pc work-pc
$ mv planets/ work-pc/
```

These could just as easily be directories on our own and our supervisor's computer, or on the computers of a group of collaborators spread around the world.

Collaborating

1. Explain what remote repositories are and why they are useful.
2. Explain what happens when a remote repository is cloned.
3. Explain what happens when changes are pushed to or pulled from a remote repository.

- hg paths
- hg push
- hg pull
- hg clone
- hg log --graph
- hg update

Conflicts and Merging

1. Explain what conflicts are and when they can occur.
2. Resolve conflicts resulting from a merge.

- hg heads
- hg log -G
- hg merge --tool=kdiff3
- hg summary

Open Science

1. Explain how the GNU Public License (GPL) differs from most other open licenses.
2. Explain the four kinds of restrictions that can be combined in a Creative Commons license.
3. Correctly add licensing and citation information to a project repository.
4. Outline options for hosting code and data and the pros and cons of each.