How to Collaborate

CSC230 Database Technologies for Analytics

31 October 2019

Take some time to learn how to use some new tools before you begin your project. The software that I am showing you is hard to use in the beginning. However, people who take the time to learn how to use these tools can be very productive with them.

Our tools include the Emacs text editor, Linux commands, and the Mercurial distributed version control system.

This document contains more instructions for the use of these tools. Look for more help on the Web.

Let's suppose that Gwen Bell and Clifford Pickover are collaborating on a project. Both have accounts on the server at *cs1.cornellcollege.edu*. Gwen's home directory is /home/gbell. Clifford's home directory is /home/cpickover.

(Gwen Bell and Clifford Pickover do not really have accounts on our server, but they are real people—you might like learning more about them by searching on the Web.)

Gwen can begin a project by creating folder for the project ('mkdir' is the 'make directory' command), moving into the new folder ('cd' is the 'change directory' command), and initializing a repository ('hg' is the prefix for all of the commands in the Mercurial distributed version control system).

```
mkdir db-project
cd db-project
hg init
```

Then she can use Emacs to create a file named *winners.sql*.

```
emacs winners.sql
```

After adding some content, she saves the file by type c-x c-s ('control x' followed by 'control s') and exit from the editor by typing c-x c-c.

She then adds the file to the repository.

```
hg add winners.sql
hg commit -m "begin project"
```

Clifford logs in and clones Gwen's repository.

```
hg clone ../gbell/db-project
```

He edits his copy of *winners.sql*.

```
cd db-project
emacs winners.sql
hg commit -m "added more names to table"
```

Clifford informs Gwen that he has made a change. She incorporates Clifford's change into her copy of the repository.

```
hg pull ../../cpickover/db-project
hg update
```

Gwen edits *winners.sql*.

```
emacs winners.sql
hg commit -m "added names of still more winners"
```

She adds another file to the repository.

```
emacs universities.sql
hg add universities.sql
hg commit -m "added info about universities"
```

Gwen might have more collaborators than just Clifford. They also clone her repository. She pulls changes from each of their copies of the repository.

All members of the team periodically check the status of their own copy of the repository.

```
hg status
```

Mercurial will list files with prefixes that indicate their status.

- An "A" before the name of a file means that the file has been added to the repository but that addition has not yet been made a permanent part of the project's history (it has not yet been committed).
- An "M" before the name of a file means that the file has been modifed but that change has not yet been made a permanent part of the project's history (it has not yet been committed).
- A "?" before the name of a file means that the Mercurial knows nothing about the file—it is not tracking changes in the file and clones of the repository will not include this file.

Teammates might also examine the history of the project.

```
hg log
```

If two or more teammates each make changes to their own copies of a file, Gwen might see conflicts when she pulls from their copies of the repository. Mercurial will prompt her to resolve the conflicts by merging the contributions of her teammates.

Gwen can also copy Clifford's files directly, without using Mercurial.

```
cp ../../cpickover/db-project/employers.sql .
```

Recall that two periods means "go up one level in the directory tree" and a single period means "where I am right now." Therefore, with the command shown above Gwen goes up two levels in the tree to find Clifford's folder and copies his file to the folder in which she is currently located. If she wanted to copy the file to a different location or give her copy of the file a different name, she would replace the period at the end of the line with a specification of the different location and/or name.

Clifford can do the same.

Clifford and Gwen are familiar with Linux commands:

cd change directory

cp copy file

ls list files

ls -l list files together with information about the file's ownership, permissions, size, and age.

mkdir make a directory

mv move (or rename) a file

rm remove (delete) a file

Let's suppose that Gwen compiles a list of people who won the Turing Award in the 1980s in a file named *winners90.sql*. Suppose that Clifford compiles a list of people who won the Turing Award in the 1990s in a file named *winners80.sql*.

Gwen is still in her *db-project* folder. Gwen can copy Clifford's file to her folder.

```
cp ../../cpickover/db-project/winners90.sql .
```

Maybe Gwen has a file named *winners.sql* in which she wants to put records of all winners of the Turing Award. She concatenate *winners80.sql* and *winners90.sql* to *winners.sql*.

```
cat winners80.sql >> winners.sql
cat winners90.sql >> winners.sql
```

Alternatively, she open *winners.sql* with Emacs and insert the other files with Emacs' c-x i command.