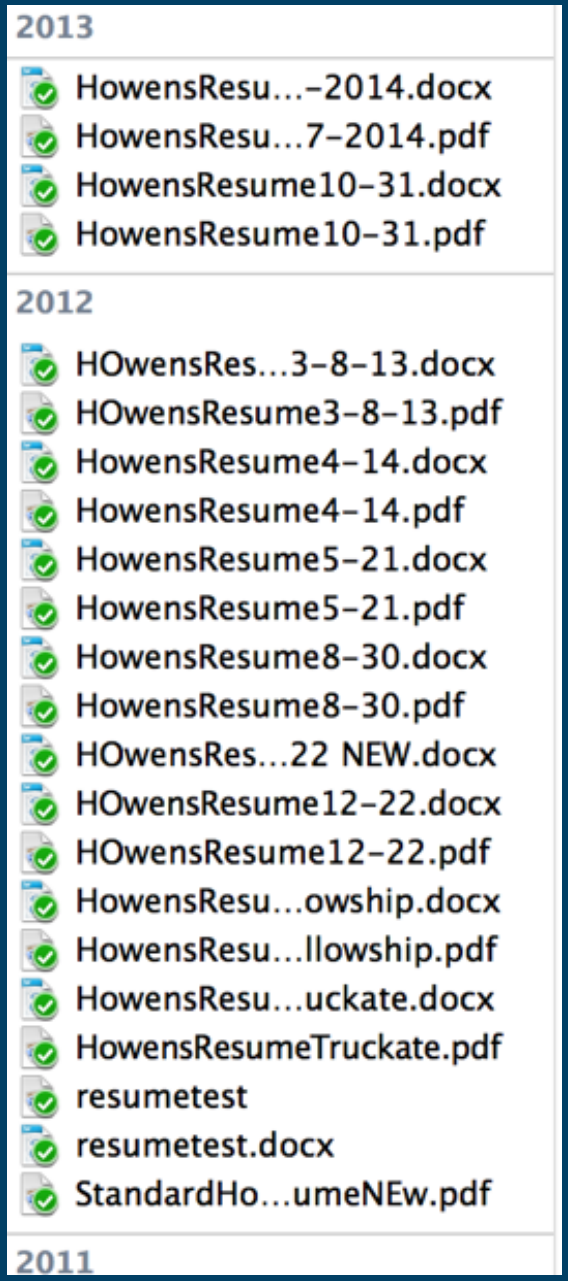
Intro to Version Control

- Questions we will answer
  - What is Version Control?
  - What is `git`?
  - What is `github`?
  - Why should I use them

The Problem

- Same file, many names
- Update, Update
- -1, -final, etc.
- What if you want to share your work with others?
- Or have them edit it?



The Solution

- Version Control Systems (VCS)
- Git (written by Linus Torvalds, same person who wrote Linux) is what is called a Decentralized Version Control System
- Doesn't require constant `client-server` interactions, rather, `peer-to-peer`

What Git Does

- Stores atomic transactions of changes to a file
- "Semantic Saving"
  - Each save of a file should have some sort of meaning
- Merging files
- Reverting to prior state
- And much, much more

Git vs Github

- Git: Program that you run on your computer
- Github: A git server + website that allows you to share code, do other things
- Work in tandem with each other

Using Git

- Key Commands:
  - git clone {uri}
  - git add {filename, list of files}
  - git commit -m “{commit message”
  - git push {remote} {branch}
  - git pull {remote} {branch}
  - git status
  - git log
  - git revert

Git Clone

- ie, https://github.com/computingForSocialScience/test-repo.git
- Your personal repo:
  - https://github.com/computingForSocialScience/cfss-homework-{your-github-id}.git
- Copies an entire repo and it's history into a new directory with name {repo}.

Git add

- Git add adds a file, list of file to a staging zone.
- This staging zone is what is going to be included on the next commit.
- Usage:
  - git add {filename}
  - git add . #adds all changes in directory and subdirectories since last commit
  - git add --all #add all changes, include removal of files

git commit

- Make a semantic save
- Usage
  - After git add (n times, where n >= 1),

```
Changes to be committed:
        modified:   howens.txt
```

  - git commit -m “some message describing what you’ve done”

git push

- Send you code somewhere.
  - ie, $*git push origin master*
  - push = send code
  - origin = a git remote, in this case, github
  - master = branch you are working on (we aren't going to cover branches in this class, we will just always use master).

git pull

- Pulling down changes, works like git push in reverse.
  - $*git pull origin master*
  - Will initiate a merge of your code (in commits) with other code.
  - Can't push to remote without having latest version, so often have to run before git push.
    - This occurs when collaborating with others most often.

git status

- Check what is going on with out with the repo.

```
~/D/c/g/test-repo git:master ››› git status
On branch master
Changes to be committed:
        modified:   howens.txt

Changes not staged for commit:
        modified:   README.md
```

- What is going part of git commit if I ran it?
- What isn't going to be part of git commit right now?

git log

- Display the history of repo

```
~/D/c/g/test-repo git:master ››› git log
Wed Jan 28 10:35:06 2015 -0600 1e6be9b (HEAD, origin/master, master) adding howens.txt  [hunterowens]
Wed Jan 28 10:34:56 2015 -0600 b27cb90 adding readme  [hunterowens]
~/D/c/g/test-repo git:master ›››
```

- Note the commit ‘hashes’ 6 character combinations

git revert

- Undo the changes in a particular commit hash.
- $*git revert {some hash}*
  - ie, $*git revert b27cb90*
- This itself is a commit. So yes, you can run revert reversions.

```
~/D/c/g/test-repo git:master ››› git log
Wed Jan 28 10:57:18 2015 -0600 d379447 (HEAD, master) Revert "readme details"  [hunterowens]
Wed Jan 28 10:57:00 2015 -0600 febda05 readme details  [hunterowens]
Wed Jan 28 10:56:49 2015 -0600 bca2838 adding text to howens  [hunterowens]
Wed Jan 28 10:35:06 2015 -0600 1e6be9b (origin/master) adding howens.txt  [hunterowens]
Wed Jan 28 10:34:56 2015 -0600 b27cb90 adding readme  [hunterowens]
~/D/c/g/test-repo git:master ›››
```