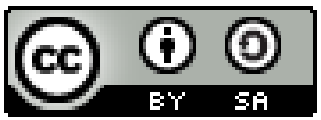
Introduction to git and GitHub

Spring 2016 BZAN 583

Drew Schmidt

February 01, 2016



Homework 2

- Live on BlackBoard now
- Due Friday before midnight.
- USE THE NOTES
- Questions?

Today's Learning Objectives

- Learn about the how's and why's of version control.
- Learn some basic `git` workflows.
- Understand the difference between git and GitHub.

Background

Data Management

- "They're going to have to know it when they get to work, so sure. Good idea."
- CODE ***IS*** DATA
- For many, code *most important* data they have!
- Learning to manage code well a *critical skill*.

File Management Without Version Control

- file.txt
- file1.txt
- file2.txt
- file2a.txt
- file2a1.txt
- file2a1_final.txt
- file2a1_final1.txt
- file2a1_final1_FINALFINALFORREAL.txt
- file2a1_final1_FINALFINALFORREAL_a.txt
- ...

Story Time!

- Preparing for lecture 1!
- A consultation...

tl;dr

If you're collaborating on code, ***USE VERSION CONTROL***.

Version Control

Version Control

- From `https://en.wikipedia.org/wiki/Version_control`:

  [Version control is] the management of changes to documents, computer programs, large web sites, and other collections of information

- Also sometimes called "revision control" and "source control".

VC Systems

- cvs
- svn
- git
- Hg (mercurial)
- Some commercial ones (perforce, . . .)
- Lots more `https://en.wikipedia.org/wiki/Comparison_of_version_control_software`

git vs GitHub

- git: Version control system
- GitHub: Web interface for git + cloud backup
- We'll return to this topic later.

File Management With Version Control

1. Create repo
2. Create/edit files
3. Commit changes, possibly push to a remote at some point.
4. Search history/revert as needed.
5. go to 2.

Philosophy

- How often to commit?
- How "big" should each commit be?
- Kind of irrelevant; just do what works for you/your team.

Terminology

- **repository**: (repo): where all the files and changes live
- **staging**: preparing to add changes to a repo
- **commit**: add changes to a repo
- **fork**: a copy of a repository
- **diff**: differences between files/commits
- **branch**: a bit advanced. All of ours are called "master"
- **remote**: a repository stored somewhere else (like a cloud backup)
- **origin**: common name for a remote in git world
- More: `https://en.wikipedia.org/wiki/Version_control#Common_vocabulary`

git

What is git?

- A VCS
- VERY FAST!
- Emphasizes data integrity.
- *distributed*
- Ludicrously, unbelievably useful.
- My 5 Most Used Commands (descending order)
  1. cd
  2. ls
  3. git
  4. vim
  5. grep

Distributed VCS

- git is *distributed*
- Each developer has a local copy, merging changes into a remote repo.
- commit is not the same as commit/push

99% of Commands You Need

- `git init`: create new git repo
- `git status`: see what files have changed/been added
- `git add`: "stage" files
- `git commit`: commit changes to repo
- `git push`: send changes to remote
- `git pull`: grab changes from remote

99.9% of Commands You Need

- The aforementioned
- `git log`: show commit log
- `git diff`: show what has changed
- `git show`: look at old diffs

.gitignore

- Some files (often binary) you *never* want to track:
  - .pdf
  - .exe
  - .o
  - knitr cache files
  - etc.
- Avoid them showing up in your search path with `.gitignore`.

Example .gitignore File

```
*~
*.pdf
*.html
*.swp

src/*.o
src/*.so
```

Live Demo

Setting up

- Install git. . .
- You can work from a Linux VM, Newton, or your laptop:
  - **Windows**: Open "git shell".
  - **Mac**: Open Terminal.App.
  - **Linux**: Open a terminal.

GitHub

What is GitHub?

- Private company
- Hosts git repositories
- Open repos free; private cost (a lot of!) money.
- Also a gui for the painful parts of git.
- Student developer pack: `https://education.github.com/`

~Social Coding~

- You can favorite repos.
- Easily fork.
- Open "issues", submit "pull requests" (patches).
- Also gives free web hosting!
  - Live: `http://knoxrug.github.io/`
  - Source: `https://github.com/KnoxRUG/KnoxRUG.github.io`

GitHub

- "Facebook for programmers."
- Dropbox for programmers.

Do Me a Favor?

- Everyone go star this repo:
  `https://github.com/heckendorfc/harp`
- Trust me, it'll be hilarious.

Adding a Remote

- You can use `https` or `git` protocols (latter if you set up ssh keys)
- Example:

```
git remote add NameOfOrigin NameOfBranch
```

Remote Naming

- General convention to call it "origin"
- If you have more than one, give it a useful name.
- You probably shouldn't have more than one any time soon...

Using GitHub (for actual work)

Let's move our example repo from earlier to GitHub.

Wrapup

Advanced Topics

- Branching
- Stashing
- Rewriting history
- Fixing broken things (bad merges, . . .)
- Some useful links:
  - Advanced git: `https://help.github.com/categories/advanced-git/`
  - git Flight Rules: `https://github.com/k88hudson/git-flight-rules`

Other Stuff

- Other gui's (don't know don't care)
- RStudio integrates with git (see above)

Really Learning git

- Just use it.
- Make mistakes and break stuff.
- Keep at it.

Wrapup

- It's been fun!
- git homework coming soon.
- Don't forget about HW2.
- Thanks!