Lecture #2: GitHub

Nicholas J. Gotelli

19 January 2023

Learning git and GitHub

- multiple copies of files
- access to past copies
- originally developed for teams of coders interacting on a single project
- learn how to use for your own work
- use for building portfolio and maintaining webpage
- use in tandem with a project in r studio, but can be used with any files

Basic Vocabulary

- `Version Control` A system for maintaining current and past copies of all of your files and a record of all the changes you have made;
- `git` Version control software for tracking your project on your local computer
- `GitHub` A free, public remote website where your project is permanently hosted
- `Repository` Any collection of files that belong to the same project, plus a few special `git` files that set up the version control. Also called a `repo` by hipster programmers.
- `Clone` Create a local copy of a repository from the GitHub website to your own website. You can clone from your own repository or anyone else's that is posted on GitHub. Once the files are on your local computer you can edit or use them as you like. However, you can only change files on GitHub for the repositories that you yourself created or for projects that you have been invited to collaborate on with others.
- `Commit (the noun)` A snapshot of your file system at a particular time. Since your last commit, it keeps track of the files you currently have, the ones you have deleted or added, and any changes you have made to files.
- `Commit (the verb)` When you commit, you are taking a snapshot of your files at the current time. You must add a summary of the commit that briefly describes what changes you have made (such as "add new homework file"). Note that all commits affect only the local copy of your repository. They do not affect the remote copy that is stored on GitHub
- `Push` Pushing a set of changes means copying them from your local repository up to GitHub.
- `Pull(=fetch)` Pulling means copying the repository in its current state on GitHub down to your local repository. This would incorporate any changes that others have made (and pushed) in a collaborative project.

Creating a github repository for an r project

Set up the repository on GitHub

- go to your github site (https://github.com/MyGitHub)

- log in to your github account (username, password)
- create a new repository by clicking the "+" button
- Name it (write down this name exactly)
- use buttons to include a read me statement
- initialize with a readme
- do NOT add a gitignore
- add a license (MIT is fine)

Build the local r Project on your machine

- open Rstudio
- upper right use pulldown menu to create new project
- select "Version Control"
- select "Git"
- add '*.pdf' to gitignore or other files to not track

Link this new local project to the remote repo on GitHub

- the RStudio set up is asking for your Repository URL
- go back to your GitHub repo
- click "clone or download" button
- click on the clipboard copy icon to copy the URL for this repo
- go back to RStudio
- paste with CTRL-V into the URL box
- give the name of the repo to be the same thing you called it on GitHub
- use browse button to find a location on your computer for this folder, or take the default on your desktop
- click "Create Project" to finish
- you are in your new project, which has .gitignore, LICENSE, and README.md files already

Build a webpage portal in your project that will be hosted by GitHub

- from the pulldown menu in Rstudio, select new markdown file
- change title to "Initial webpage set up" (or some such thing)
- save this file with file name "index" (RStudio will provide the .Rmd suffix)
- run CTRL-SHFT-K to "knit" the file

Commit these changes from the terminal

- git status
- git add -A (stages changed files)
- git status
- git commit -am 'first commit' (makes local commit)
- git status
- git push (push the changes up to the GitHub repo)

Host the "index" webpage

- Return to GitHub
- Go to "Settings"

- Scroll down and under “Pages” and “Branch” change from “None” to “main”
- Wait a minute and click the link at “Your site is live at…”
- New webpage should display
- Note the address https://UserName.github.io/RepoName/
- Example https://ngotelli.github.io/Trial/

Basic workflow for git from the terminal

Check things out before starting your work

- git status (make sure your local repo is not ahead of the remote)
- git pull (just in case there was a change made from another location)
- git status (ready to start working)

Make changes

- modify files, including knitting new versions of html pages or pdfs
- delete files
- add files

Stage changes

- git status
- git add -A
- git status

commit changes

- git commit -am ‘write a brief informative commit message’ (no caps or punctuation, present tense)
- git status

push changes up to GitHub

- git push
- git status
- git pull

check and refresh website to see changes

Set up simple local project in RStudio

- New Project
- Bio381ClassCode
- open repository with *.Rproj

Review of basics (example demo)

- check `status` of the local repo (uncommitted changes? local commits ahead of remote?)
- `pull` any changes from the repo (incorporating changes pushed by others
- do some work (edit, create, delete files)
- `stage changes` (local)
- `commit changes` (local)
- `push` changes from local repo to GitHub