Git + GitHub 101: Version Control, Workflow, and Branching

Wha is git? What is GitHub?:

Git: a version control software

1. command line tool that keeps track of changes we make to code
2. synchronizes code between different people
   - make changes on a local repo, push it to a remote repo
   - pull changes from the remote repo
3. Lets you test changes you want to make without losing access to the original
   - make changes on a separate branch
4. lets you revert back to old versions of the code

GitHub: a website that hosts a git repository and provides and interface to interact with it

GitHub New: where you go to create a new repository

(1) name + description (2) public or private?

Downloading a remote repo: how do we take stuff from GitHub and get it on our local computer

[] git clone {repo url} : take a repository from the web, and download it on your computer

ex: git clone https://github.com/rickyriled/test-dumby-repo.git

Making/ changing a file:

[] vi/ touch : create a file and edit them

Saving locally: how do we get git to record the current state of our files?

[] git add {filename/repo}: tells git you'd like to add a certain file to the current 'snapshot' pile

adds a file to the 'to save' pile

ex: git add hello.py

[] git add . : adds the whole content of the current directory to the current 'snapshot' pile

[] git commit -m "{message}" : saves a snapshot of your current 'snapshot' pile

ex: git commit -m "added hello.py file to the repo"

output: tells you how many files you've changed,
and how many insertions/ deletions occurred

[] git commit -am "{message}" : combines git add step and git commit step

'Pushing' to GitHub: all are changes/ saves have only been done locally — how do we send it to the remote repo?

[] git status: will tell you

(1) how ahead you are of the "origin repo" (the one on GitHub)
(2) what files are/ aren't in your 'snapshot' pile

[] git push: pushes your committed 'save-pile' to GitHub

[] git pull: downloads the latest version of the file on GitHub

also tells you the insertions/ deletions

you can edit files on GitHub too! just click the 'pencil' button in the right-hand corner when you're on a file

Merge conflicts: what if our locally repo isn't in-sync with our remote repo?

usually happens if you're trying to pull in changes that are out-on-sync with your current version

- If you make a pull without checking things, git will try to merge files automatically the best it can
- sometimes it cant though, and we need to edit things manually
- usually after a merge failure, git will show a few modifications on the files that caused the conflict

your local version: top half
GitHubs remote version: bottom half

after making the changes, commit the results again, and try pushing them back to GitHub

[] git log : shows a list of all the previous commits you have made i.e. changes to your code that you recorded

[] git reset: takes the current state of your repository, and reverts it back to and older version of it

[] git reset −−hard {commit hash} : resets your repo to a version associated with a previous hash
[] git reset −−hard origin/master : resets your repo to the version currently on github

Making Changes: how do we work with branches/ how do we work on our code in parallel and non-linearly

Often we change code in a non-linear way; e.g. you're working on fixing on a bug, but then you notice theres a new feature you want to add. Working on the new feature and fixing the bug can happen in parallel and independently. One approach to this is branching.

Head: the current branch you're focused on

- Later you can merge branches back together

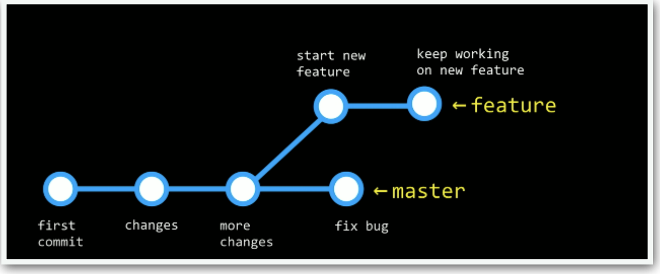
[] git branch : tells you what branch you're on (* symbol) and which branches exist
[] git branch {branch name} : makes a new branch

[] git checkout -b {new branch name} : sends you to a different branch, or creates a new branch for you
[] git checkout {existing branch name} : sends you to a pre-existing branch

Make sure to commit between changes

merge in changes:
[] git merge {branch name} : merges a sub branch into whatever branch you're currently working on

- often this results in a merge conflict



*Forking : making your own copy of some repository

- go on GitHub, click on a repo, and fork it
- clones the repo - you can then make a pull request

Command Summary:

```
[] git clone {repo url}
[] git add {filename/repo}:
[] git commit -m “{message}”
[] git status
[] git push
[] git pull

[] git log
[] git reset
    [] git reset −−hard {commit hash}
    [] git reset −−hard

[] git branch
[] git checkout -b {new branch name}
[] git merge {branch name}
```