General Assembly

STUDY GUIDE

GITHUB BASICS

Key Terms and Definitions

» **Version Control:** The concept of being able to save new versions of files as well as revert to older ones.

» **Git:** A distributed version control system run as a program from the command line.

» **GitHub:** A public platform for housing version-controlled repositories, which includes a web interface for exploring repositories and serves as a social network for programmers. This is *separate* from Git itself.

» **GitHub Enterprise:** A private platform for housing version-controlled repositories; the professional application of GitHub.

» **Repository:** A location in which data is stored and managed. Think of a repository just as you would a directory, folder, or file system. Referred to as "repo" for short.

- Repositories can be **local** or **remote**.
  - **Local:** Refers to the copy of the data and/or files stored directly on your machine.
  - **Remote:** Refers to the copy of the data and/or files stored in a location other than your machine, such as on GitHub.
- Repositories are made up of several pieces:
- **Working directory:** Holds the actual files (the current state).
- **Index:** Acts as a staging area (the "unofficial" most recent save).
- **HEAD:** Points to the last commit you've made (the "official" most recent save).

» **Commit:** A "save point" in your repository, which is added as an event in the changelog and can be referred to or restored.

» **Branch:** "Versions" of a repository that can be modified independently without affecting each other. Once the point of origination is established, they are blind to any changes made after that point on the master OR any other branches. They can be merged back into the master branch when ready.

» **Clone:** A downloaded copy of a remote repository. Cloning preserves a link to the remote repo.

» **Fork:** An individual version of the master repo stored in your individual repository. This is*not* the same as a branch. If you clone your fork, the fork will be the "master" and the original master will be the "origin." Unlike a branch, the fork will still be able to "pull in" changes from the origin, and your changes would merge directly into the origin's master branch if you pushed your commits.

» **Push:** Updating the *remote* repository with commits you made in your*local* one.

» **Pull:** Updating your *local* repository with commits that have been made to the*remote* one.

» **Homebrew:** A programming package manager for macOS that allows users to easily install new packages and update many packages simultaneously.

» **Anaconda:** Another package manager for Python-related packages, such as NumPy.

Guiding Questions

1. What are the benefits of using version control instead of saving multiple versions of your files?
2. When might you choose to use a fork of a repository? A branch?
3. When is a good time to make a commit? What would be the problem with committing too frequently or not frequently enough?

Additional Resources

» Git Foundations
- A helpful resource for syntax practice.

» GitHub Tutorial
- There are many similar tutorials out there, but this one is efficiently explained.

» GitHub Demo
- Watch our sample tutorial for a brief walkthrough.