About Git

- A version control system
- Allows you track and control changes to a project
- You can view the history of a project
- You can revert to older changes if needed
- Stores data in a database called a “repository” or “repo”
- Also creates a subfolder called .git to store the data

About GitHub

- A hosting service for Git repositories
- Popular in the development community
- Developers use it to contribute to Open Source projects or to showcase their work

Command Line (CLI)

`cd [directory]` Change directory or folder
`cd ..` Move up a level
`cd ../` Move back a level
`ls` List all files and sub directories in the working directory or folder
`mkdir [newname]` Make new folder (aka directory)
`rm [filename]` Remove file. BEWARE: You can’t undo this, so be careful.
`mv myfile.txt hello/` Move a file to another folder (in this example, a folder named “hello”)
`pwd` Displays or prints the name of the working directory

More: https://www.codecademy.com/articles/command-line-commands

Git

Commands

Open **Git Bash** (Windows) or **Terminal** (Mac)

`git --version` Check if git is installed
`git init` Initialize a git repo
`git status` Check what's been changed or staged, if you're in a git repo
`git log` View commit history (`q` to exit)
`git diff [file]` See all changes to a file that hasn't been added

NOTE: To get out of a git prompt window, hit the escape key `esc` and then type `:wq` and hit `return` to get out of that

Downloading a repo

`git clone [git-repo-url] [optional: name of folder to clone into]` Download a git repo. Using this will automatically set up the remotes

Adding files

You want to move through this workflow when staging and committing changes in git:

Working files	Staging (`git add *`)	Commit (`git commit -m "message"`)	Send to Github (`git push`)

`git add [relative path to files separated by spaces]` Add a new or modified file to be commited. Use a `*` to add all files (`git add *`)

`git commit -m "Commit message"` Commit files that have been added with a message

Shortcut: `git commit -am "Commit message"` Add and commit all modified files

`git commit --amend -m "Commit message"` Edit last commit message

Sync changes to Github

First time: `git push -u origin master` Will allow you to connect to the correct branch

`git push` Push commits on your computer to Github

`git pull` Pull commits from Github to your computer

Branches

`git branch` Will list current branch (likely *master)
`git branch -a` will show all branches (remote/local)
`git branch -r` will show all remote branches

Create new: `git checkout -b [branchname]`

Switch to Master: `git checkout master`

Merge (go to master to merge in another branch): `git merge [branchname]`

Delete: git branch -d [branchname]

Log

`git log --oneline --decorate` To see your log with the shortened hash, one line

Three ways to revert changes

You have yet to add or stage a change.
`git checkout index.html` Simply checkout the last staged version

You added (or staged) a change.
`git reset HEAD index.html`
`git checkout index.html`

You committed a change. Whoops!
`git revert d391fb4 --no-edit` To revert a committed change

Resources

Get Git if you don't have it
https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Great resource: http://product.hubspot.com/blog/git-and-github-tutorial-for-beginners

How to "undo" in Git
https://github.com/blog/2019-how-to-undo-almost-anything-with-git

Restore repo to previous revision
https://www.git-tower.com/learn/git/faq/restore-repo-to-previous-revision

Using Terminal
http://mac.appstorm.net/how-to/utilities-how-to/how-to-use-terminal-the-basics/