Git Basics

- Clone
  - Copy the repo onto your local machine
- Status
  - Show the difference between local and remote repo
- Add
  - Add a new file to be tracked
  - Use to resolve untracked files
- Commit
  - Sending a change up to the repo, but NOT synced
- Push
  - Pushing a change to GitHub.com
- Pull
  - pull from GitHub.com
  - Updates files, not copy

The Command Line

- Clone
  - git clone https://GitHub.com/repoInformation
    - Get the link from GitHub homepage
- Status
  - git status
- Add
  - git add file.txt
  - git add -A
    - Adds all
- Commit
  - git commit -m “message here”
    - Trying to commit without a message will bring up vim to write a multi-line message
    - Esc, “:wq” quits and saves
- Push
  - git push

Workflow

- Start with a Git Pull
- Add, Commit, and Push your changes
  - Commit often and write descriptive messages

Merge Conflicts

- When two or more people make and push changes to a line simultaneously
- To resolve, pull the repo and fix
  - Format of a conflict:

```
<<<<<<<<< HEAD
        My changes
=========
        Others’ changes
>>>>>>>>> ID
```

  - Fix the conflict, and commit the result

Creating a New Repository - GitHub Desktop

- File > New Repository
  - Name it and write Description
  - Check README initialize
  - Leave Git Ignore at None
  - Choose MIT license for Open Source
- Create Repository

Adding A File to The Repo - GitHub Desktop

- Create a new file in the repo folder and edit it
- On GitHub Desktop, it should be seen under the Changes tab
- Write a commit message in the box on the bottom left side
- Clicking the Commit button creates the commit

Push To Origin - GitHub Desktop

- Push local commits to the GitHub repo
- Click “Push Origin” button on the top bar of the app

Ignoring A File - GitHub Desktop

- Uncheck the box next to a file under the changes sidebar
  - The file then won’t be included in any commits
- Right click the file and click Ignore
  - This puts the file name in a new .gitignore record
- In the .gitignore, you can use wildcards

Deleting a File - GitHub Desktop

- Delete it from the local repo
- Commit and Push to Origin

Reverting a Commit - GitHub Desktop

- Go to the History tab in the left sidebar
- Find the commit you want to revert
- Right click, and select “Revert changes”
- A new commit is automatically created rolling back the old changes

Branching

- Each branch is a unique version of the code base
- Usually it’s good to separate developers and tasks into different branches
  - Home-page-bug, careers-page, etc.
  - Created off main branch
- When work is complete, side branches are merged back into the main branch

Using a Branch - GitHub Desktop

- In File Bar, select Branch, then Create Branch
  - Name it, then you’ll see your new branch in the app under “Current Branch”
- Make local code changes
  - Commit those changes to your branch (not main)
- Once we are ready to deploy, merge the branches
  - Ensure you are on the main branch in the desktop app
  - Then in File Bar, select Branch, then Merge into Current Branch
  - Select your branch from the pop up menu
  - A new merge commit is automatically created
- For teams, branch merges are usually done with pull requests