Introduction to git and GitHub

What is git?

- A version control system
    - Multiple "versions" of your project files are maintained
    - You can go back to any previous snapshot
    - Effect: you can experiment safely
- A *collaborative* version control system
    - Lots of people can work on a project
    - Changes from different people can be merged together

git concepts

- **Repository**: A directory that contains your project files and the entire revision history.
- **Commit**: A snapshot of your files at a specific point in time.
- **Branch**: A parallel version of your repository.
  May contain some commits in common with the main branch and some different ones.
- **Merge**: The process of combining changes from different branches.

Initially, we will just use one branch, called `main`, so we won't need merges. But pretty soon we will need to merge changes from different branches. For example, if you update this course resources repository using the sync-updates script, it will grab the latest changes from the instructor's branch and merge them into yours.

What is GitHub?

- A website for hosting git repositories online.
- Makes it easy to share code and collaborate.
- Also provides a backup in the cloud for all of your snapshots!
- Provides a web interface for viewing snapshots and the differences, file by file, between snapshots.
- Supports workflows for proposing changes and reviewing them before they are merged. (*pull requests*)

git vs. GitHub

- **git**: A version control system for tracking changes in your code.
    - You maintain a local copy (clone) of a repository on your machine
        - in our case, in a codespace, a virtual machine that runs in the cloud
    - You can push your changes to GitHub and pull changes from others.
- **GitHub**: A platform for hosting and collaborating on git repositories.

File States

When you work with git, each file in your project can be in one of several states:

- **Untracked**: The file is new and not yet being tracked by git.
- **Unmodified**: The file is tracked by git and unchanged since the last commit.
- **Modified**: Changed, but those changes are not yet staged for commit.
- **Staged**: Changes have been added to the staging area and will be committed when the commit command is run.

When you commit, files return to the "unmodified" state until you change them again. Files move between these states as you work, add, and commit changes.

Other important terms

- **HEAD**: A pointer that refers to the latest commit in your current branch. It represents your current working state.
- **Remote**: A version of your repository that is hosted on the internet or another network. GitHub is a popular remote hosting service.

Common git commands

vscode, our IDE (Integrated Development Environment) has some visual tools for interacting with git. But its helpful to understand the git commands they invoke.

- `git status` — see what files are modified or staged
- `git add <file>` — move the file to the staging area
- `git commit -m "<message>"` — save a snapshot with all the staged files and describe it with the message
- `git log` - view this history of commits, each with their messages describing the changes that were made since the previous commit
- `git push` — upload changes to GitHub
- `git pull` — download changes from GitHub

Lab Exercise 1: Use git commands

- use `git status` to see the current state of your files
    - You should see at `week1/madlibs/sample_story.txt` which you changed in session 1 lab.
- use `git add <file>` to stage changes
- use `git commit -m "<message>"` to commit your changes
- use `git log` to view your commit history
- use `git push` to upload your changes to GitHub

Not sure what to do? Ask CoPilot chat for assistance!

Lab Exercise 2: Explore your repository on GitHub

- on GitHub (the website), navigate to your repository
- Click on the "Code" tab to view your files
- Click on the "Commits" tab to view your commit history
    - You should see a list of all your commits, along with their messages and timestamps. Just like you saw with `git log`
    - Click on a commit to view its details, including the changes made in that commit. Notice the green and red diff view that shows what was added and removed in each file.