Git, GitHub, & RStudio

Git

Git is a version control system that helps you keep track of changes in your code or any other files. It's like a time machine for your work. Git allows you to:

- Track Changes: You can record every change you make to your files. This includes adding new files, editing existing ones, and deleting files.
- Collaborate with Others: Git is designed for collaboration. Multiple people can work on the same project simultaneously, and Git helps to manage changes and merge them seamlessly (ish).
- Create Backups: Git acts as a backup system. If something goes wrong, you can easily restore your project to a previous state.
- Work Offline: You don't need an internet connection to use Git. You can commit changes to your local repository, and when you're online, you can sync those changes with a remote repository (like GitHub).

GitHub

GitHub is a web-based platform that uses Git for version control. It's like a social network for programmers and a hosting platform for Git repositories. Here's what you can do with GitHub:

- Store Code in the Cloud: You can store your Git repositories online. This is particularly useful because it acts as a backup, and you can access your code from anywhere.
- Collaborate with Others: GitHub is an excellent platform for team collaboration. You can invite team members to your project, and everyone can work on the same codebase. This is what we will be using for our class to access notes and labs.
- Share Your Work: GitHub allows you to share your code with the world. You can make your repositories public so that anyone can see your code and contribute to it, or you can keep them private for restricted access.
- Issue Tracking: GitHub provides tools for issue tracking and project management. You can use it to keep track of tasks, bugs, and feature requests.

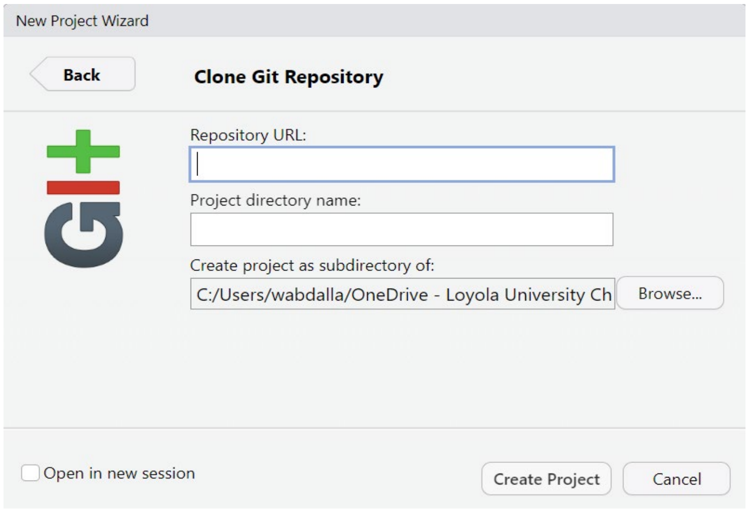
Installing Git

1. To install Git on your computer, go to the official website.
2. Under "downloads" select Windows or MacOS.
3. For Windows, click on "click here to download".
4. For Mac, go to "Binary installer" and click on "installer".
5. Follow all the steps to install git. Once git is downloaded successfully, you will need to close RStudio.
6. If you have trouble downloading/installing git, you can refer to the following video: https://www.youtube.com/watch?v=F02LEVYEmQw

RStudio and GitHub

- Cloning a repository into RStudio:

1. Go to https://github.com/wabdalla
2. Click on the repository named DSCI_101
3. Go to "<> Code" and copy the directory of the repository.
4. Open RStudio: File > New Project…
5. Click on "Version Control", then click on "Git". You will get a window that looks like this:



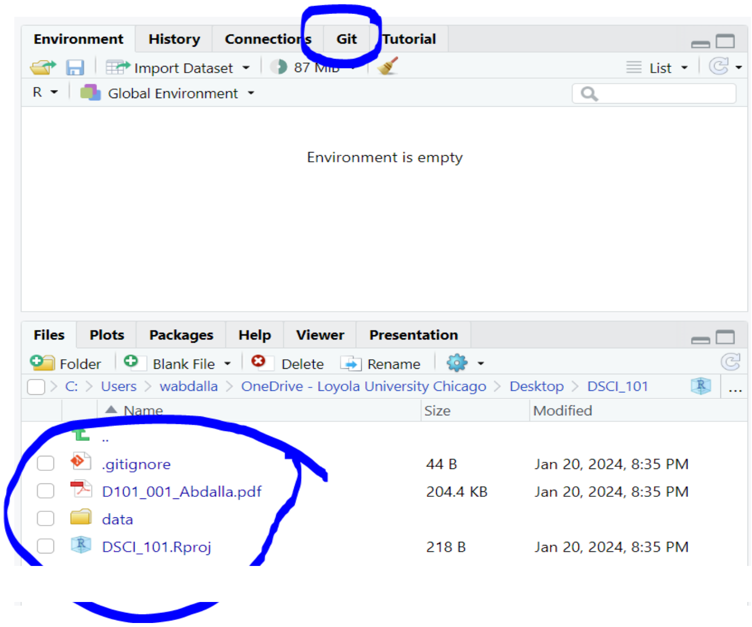
Repository URL:

Project directory name:

Create project as subdirectory of: C:/Users/wabdalla/OneDrive - Loyola University Ch

□ Open in new session

6. Paste the URL that you copied in Step 3 from github where it says "Repository URL".
7. The "Project directory name" will automatically be named DSCI_101.
8. Click on "Browse" and choose the location where you'd like to save this folder. I recommend choosing the desktop since it's easier to access.
9. Click on "Create Project" and wait for it to load.
10. Your RStudio should now look like this:



Environment is empty

Name	Size	Modified
..		
.gitignore	44 B	Jan 20, 2024, 8:35 PM
D101_001_Abdalla.pdf	204.4 KB	Jan 20, 2024, 8:35 PM
data		
DSCI_101.Rproj	218 B	Jan 20, 2024, 8:35 PM

- Pull the Latest Changes: Before working on any labs/HW, pull in the git window to get the latest version of the repository. The pull is the green arrow under the Git tab. You will not be committing or pushing on this class. Just *pull*.
- If you were working with others in a collaborative manner, you could be committing and pushing changes after you pull.
    - Commits:
        - In Git, a "commit" is like taking a snapshot of your project at a specific moment. It records all the changes you've made since the last snapshot.
        - Think of commits as checkpoints in your project's history, showing what was changed and who made the changes.
        - Every commit has a unique ID.
        - You write a brief message to explain what each commit does.
    - Pushing:
        - "Pushing" means sharing your local project changes with others on GitHub.
        - It's like uploading your work online so that your team members or collaborators can see and use it.
        - When you push, you're keeping the shared project up to date.
    - Typical Workflow:
        - Pull Down Latest Work
        - Make Changes
        - Record Changes (Commit)
        - Share Changes (Push)
        - Stay Updated (Pull)

This cycle helps everyone stay coordinated when working on the same project.