Day 3, Lecture One: Version Control with GitHub

Andie Creel

January, 2023

1 Why use GitHub?

GitHub helps us implement our best practices for file structure. Remember, having a clear file structure acts as an outline of your project so that your collaborators can easily follow your work, and so that you can pick up a project quickly after taking a break.

GitHub will keep track of all the edits you make to your files. This way you won't have to have a million files labeled some version of `3_analysis_final_final_final.R`. Instead, will have one file and then a log of all the changes you've made to that file. And example of all my log for this workshop: https://github.com/a5creel/intro_to_programming/commits.

GitHub enables collaboration with others. Everyone can be a part of one project on GitHub, allowing you to easily see the log what others have done and share what you've changed.

Your code is stored in the "cloud" so if you break you're computer it's okay. I'm on my third laptop of grad school so this point is really important for me.

1.1 Some Language

- Repository (repo): repositories are *kind of* to GitHub as R Projects are to your local computer
- push: you push changes from your computer (*your R Project*) to the cloud (*your GitHub repo*)
- pull: you pull changes (*your GitHub repo*) from the cloud to your computer (*your R Project*)
- commit: a message you write about the changes
- clone: GitHub word for copy
- stage: select the files you have changed on your local computer and want to push to github

2 Create a GitHub Account

- Go to https://github.com/
- Sing up for an account with email and password
- Do NOT use your Yale email, use an email you'll have access to ~forever.

3 Install Git to your computer

- Git is the operating system that GitHub is built on
- Download here: https://git-scm.com/
- Follow installation instructions

4 Set Up Git

- Configure your Git installation with your name and email. This information will be attached to your commits.
- Open your **terminal** (Command Prompt or Git Bash on Windows) and enter the following commands. *Remember that terminal is different from console.*:

```
git config --global user.name "Your Name"
git config --global user.email "your_email@example.com"
```

5 Integrate Git with RStudio

- Open RStudio.
- Go to Tools (far right on tool bar) > Global Options > Git/SVN.
- Ensure that the path to the Git executable is correctly set.
- Figures crossed this just goes as planned! Hopefully nothing to actually do here.

6 Create a New Repository on GitHub

- Log in to your GitHub account.
- Click on the "+" icon in the top right corner and select "New repository."
- Give your repository a name, a description, choose public or private, and initialize it with a README.

7 Clone the Repository in RStudio

This step is almost identical to what we did yesterday, we're just adding the step of syncing our GitHub with our project.

- In RStudio, go to File > New Project. (similar to yesterday)
- Choose Version Control, then Git. *This is where it's slightly different.*
- Enter the repository URL from GitHub and decide where to store it locally.
- Click Create Project. This will clone the GitHub repository to your local machine and open it as a project in RStudio.

8 Make Changes and Commit

- Within the RStudio project, you can now create or edit files.
- You can create a file, or you *should* have a README.md file that you can edit.
- To commit changes, go to the Git tab in RStudio (usually in the upper-right pane).
- Stage the changes you want to commit (*click the files*), add a commit message (*that log we saw earlier? This is the message*), and then click Commit.

9 Push Changes to GitHub

- To share your local commits with the GitHub repository, click the Push button in the Git tab in RStudio (*Green arrow, pushing it to the cloud*)

On this step we're going to need to enter our "password." GitHub doesn't let us use normal passwords, we have to use "personal access tokens" instead. You'll need to do this as often as you set your access token to expire.

9.1 Generate a Personal Access Token (PAT) on GitHub

- Go to GitHub
- Access Settings: Click on your profile picture in the top right corner, then select "Settings."
- Developer Settings: Scroll down to the bottom of the sidebar and click on "Developer settings."
- Personal Access Tokens: In the left sidebar, click on "Personal access tokens."
- Generate New Token: Click on "Generate new token," give your token a descriptive name, and set the expiration according to your preference. *However ever often you feel like doing this.*
- Select Scopes: Choose the scopes or permissions your token should have. For typical Git operations, select repo, which includes access to private repositories.
- Generate Token: Click "Generate token" at the bottom. **Make sure to copy the token and keep it somewhere safe, as you won't be able to see it again.**
- Push your changes again
- Enter the PAT when prompted

10 Pull Changes from GitHub

Sometimes a collaborator will have changed things and you'll need to pull the changes from the cloud to your local computer. To show and example of this:

- Go to GitHub
- Go to your repository's homepage, which is the README file
- Click the edit button (little pencil)
- Make some changes
- Commit those changes
- Go back to R Studio
- Pull the changes to your local machine (*blue arrow, pull to local machine*)

11 Fork a repository

Forking creates a copy of someone's GitHub repository that you can edit and work with. You can fork any public GitHub repository. Let's fork this workshops repository.

- Navigate to this courses webpage (repo): https://github.com/a5creel/intro_to_programming
- Fork the repo: button is usually located in the top-right corner of the repository page, near the "Star" and "Watch" buttons. GitHub will create a copy of the repository in your account.
- Go to your GitHub profile. Look at your repositories. You should have one called "intro_to_programming". When you click on the link, is should be "https://github.com/YOUR_USER_NAME/intro_to_programming".
- Clone to your computer so you can make changes to it. (follow same steps as before.)
    - make new r project (somewhere you can find!)
    - paste in your URL link
    - Should populate your project with all my files