# Intro to Git and GitHub

Tara Henechowicz

April 25, 2022

# Session Outline

1. Download and Install Git help, creating github account help (first 15 minutes)
2. Introductions (15 minutes)
3. Git Slides (see slides from U of T coders) (20 minutes)
4. Working with Git locally: edit a text file and git it
5. Pushing to GitHub
6. GitHub features for Collaboration

# Step 1. Configure git

```
#This step you only do it once (first time using git on yo
git config --global user.name "tarahenechowicz"
#your github username, check your github profile
git config --global user.email "tara.henechowicz@mail.utor
#also recommend using your github account email
```

# Step 2. Create a Repository on your computer with gitGit

```
#First step is we need to know where we are in our computer
#To figure out where you are (if you're lost!):
pwd

#On mac you might want to change to a place that finder can
  #windows it is called "documents"
  #to change the director use the command cd
cd Documents #name of directory
# to go backwards use cd .. to go back one level, have to d
cd ..
cd ..
#We are going to make a directory (or folder) for our proje
mkdir GitPractice2
# now let's go into the directory that we just made
cd GitPractice2
```

```
## /Users/tarahenechowicz/Documents/RTeaching/IntroGitGitHu
## bash: line 7: cd: Documents: No such file or directory
```

# Step 3. Git Init (Initialize the Repo)

```
git init ## Create the repository (init = initialize)

#change the default branch name to main
git config --global init.defaultBranch main
```

```
## Reinitialized existing Git repository in /Users/tarahene
```

# Adding files to the Repo

- Let's create a text file to store some data to work on

1. Create a text file named mydata.txt

```
#use touch command and create a file
touch mydata.txt
```

2. Open the file and add:

- your name
- your program
- your research area

3. Save the file.

# Making our first commit

```
git status mydata.txt ## Check the activity
#we need to add it to the stage so it can be commited
git add mydata.txt
#you can also do git add all to add all the files in the r
  #git add -A
```

```
## On branch master
## Changes not staged for commit:
##   (use "git add <file>..." to update what will be commit
##   (use "git restore <file>..." to discard changes in wor
##  modified:   mydata.txt
##
## no changes added to commit (use "git add" and/or "git co
```

you can also use `git add -A` to add all the files in your repository (remember we only have one right now)

# Making our first commit

Let's save our first set of changes using a commit. You need to put a message with a commit.

```
#git commit to save file to the history with a message (-m
git commit -m "Commit for the first time"
```

```
## [master 010d8d2] Commit for the first time
##  1 file changed, 1 insertion(+), 1 deletion(-)
```

# Making our second commit

- Now we are going to back to the text file and make a change
- for example, I'm going to add my affiliation in the first line
- you can add your programming languages and experiences
- or add a sentence about why you want to use git

# Check activity and changes with git status or git diff

1. Check the activity: `git status`
2. Shows the difference between changes in our repo and last commit: `git diff`
3. Add all of our changes in the repo use: `git add -A` or select what files you want to commit
4. `git status` - should return green to show that our text file is on the stage ready to be commited
5. `git commit -m "message describing changes"`

Let's check our log of changes: `git log`

# Pushing repos from your computer to GitHub

1. Create repo on GitHub
2. Set up connection between our computer and GitHub
3. "Push" repo from computer to GitHub

# Create a repository on GitHub

## Create a repository on github that matches the name of your repository on your computer

## Why would we want to do this?

1. storage (privately or publicly)
2. Public sharing of data (open science)
3. now that its on github your team members or even general public can collaborate with you on your work and you can log the changes of multiple users!

# Set up the connection between our computer and github

- To push our repository to github, we need to first set up the connection between the repository on our computer with the repository on github
- go to the repository that we just made and there is the green button called "code" and copy the code from the https: `git remote add origin <link for github repo>`
- Now the computer will prompt to ask for your github username and password
- GitHub no longer takes your account password, you need to set up an authentication token

# Set up the Git authentication

Git authetication token start up : https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token

- verify email address
- go to settings
  - developer settings, personal access tokens
  - "generate a personal access token"
  - recommended you change expiration, depends on the security of your data/code.
- Personal access token lets git control your github account
- Don't lose it its like a password.
- If you forget it you have to do this process again.

# “Push” the repo from your computer to GitHub

In your terminal use this code: `git push --set-upstream origin <name of branch>` would be main or master depending on how your git is configured

- next enter your github username
- paste your access token when it asks for a password

# GitHub features for Collaboration

# Issues

- Issues are like little emails that you can send to yourself on GitHub or tag and assign others
- You can leave notes to let people know about errors in datasets or bugs in code.
- You can make tasks or to-do lists.

# Issues practice

- Let's try to make an issue called "Set up github profile" on GitHub with the following to-do list:

1. Make a repository that is named the same as our github username
2. Create a README.md file
3. Inside the README.md file put in your name, bio, contact information

# Cloning

For this lesson we are going to clone your profile repo onto your device.

1. Move back to your documents folder

- review: how do you find out which folder you are currently in?
- how do we move to the documents folder?

2. On GitHub, go to the repo that you want to clone and hit the code button
3. Select https and copy that link to: ‘git clone

# Branches

- Allow you to split off onto a new version of work without affecting the main branch
- Helpful for version control and collaboration if you want multiple versions of the same work
- This is an introduction to a complicated git topic - the more branches, the trickier it gets.
- Use with caution with starting, better to focus on just using the main branch

# Make a new branch

1. Use `git chekcout` to split between branches:

`git checkout -b <new branch name>`

`-b` option creates a new branch, you don't need it just to switch

If you already have a branch created you can use 'git checkout

`git branch` is code to check branches and which branch you are working on

# Make changes on the new branch and git commit

2. Now open one of the files in the repo and make changes in the file.
3. Follow the workflow to make a commit on the branch:

- review what is the workflow for making a commit?

# Push the branch to GitHub:

4. Once you have made a commit, you can push that branch to GitHub: `git push --set-upstream origin <branch name>`

# Pull Request

- Demo how to use Pull Request to merge changes from branch and the main
- get's more complicated with more complicated changes, code, or with multiple branches/users

# Forking

- go to the repo that you want to form, press “fork” button in top corner
- try forking one of my repos from: https://github.com/tarahenechowicz?tab=repositories

# Collaboration with Github Further Reading:

- Social aspects of GitHub:
- starring
- following
- Further reading for how to collaborate on GitHub: https://github.com/tarahenechowicz/studyGroup/blob/gh-pages/lessons/git/collaboration/lesson.md
- Intermediate and Advanced GitHub topics: https://github.com/tarahenechowicz/studyGroup/blob/gh-pages/lessons/git/