Essentials of Git for Nucastro

McKenzie Myers

Basics of Git

What is Git/GitHub?

- Git: most popular version control system
- Version control: tracking/managing different versions of code
- GitHub: cloud-based code storage and retrieval

Why should we use Git?

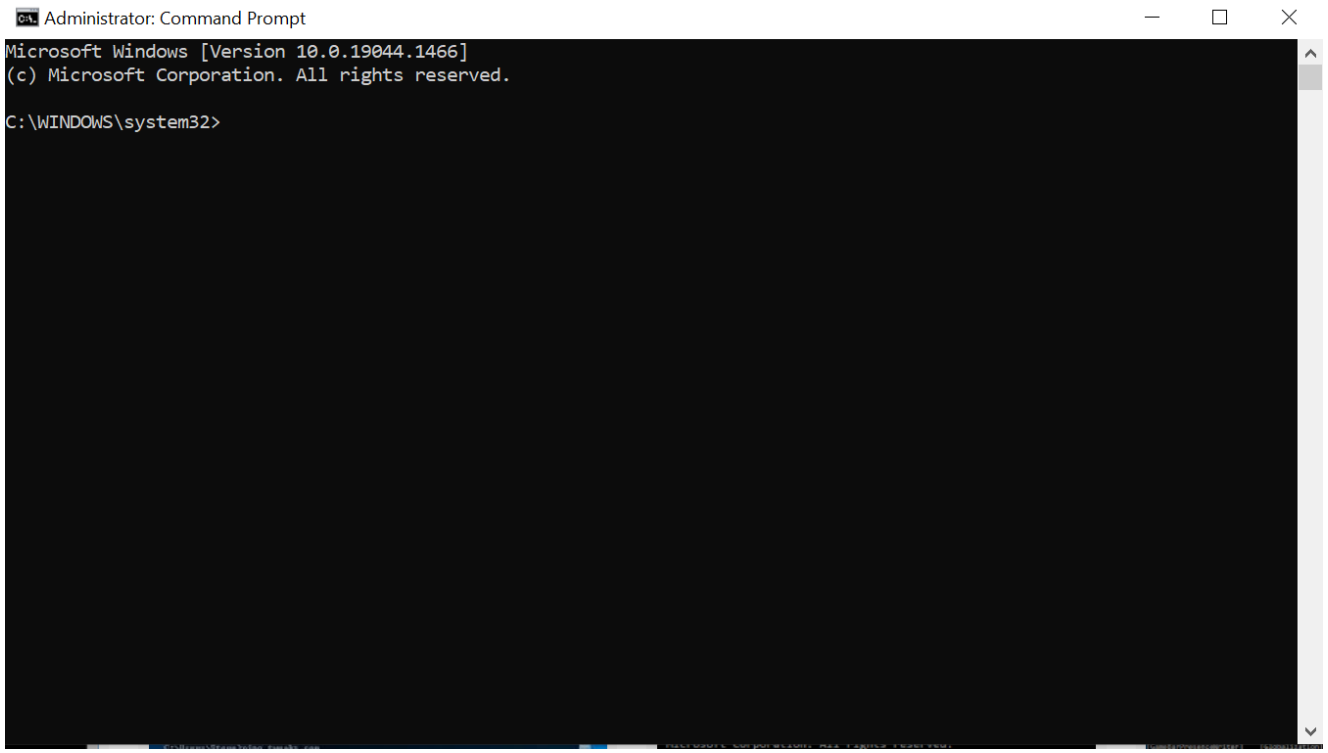
- Helps us organize our code
- View revision history and restore previous versions of code
- Cloud-based backup for our code/data
- Easy way to share and collaborate on code
- View/edit your code on any computer

What you need

- Git works best in a Linux/Mac environment
- If you're using Windows 10/11, install Windows Subsystem for Linux (WSL) – I use this daily with no issues
- You need permission to install things on the computer you're using

WSL for Windows users

- Type `cmd` into search bar
- Right click Command Prompt → Run as Administrator
- Type `wsl --install` when this window comes up
- Ubuntu is the default distro



Installing Git

- On Mac and CentOS (our desktops), Git is installed by default
- On Ubuntu: `sudo apt install git-all`
- Verify your installation by running `git version`

Setting up Git

- Tell Git who you are so it knows which account to authenticate when you push code

- Your email:

- Your name:

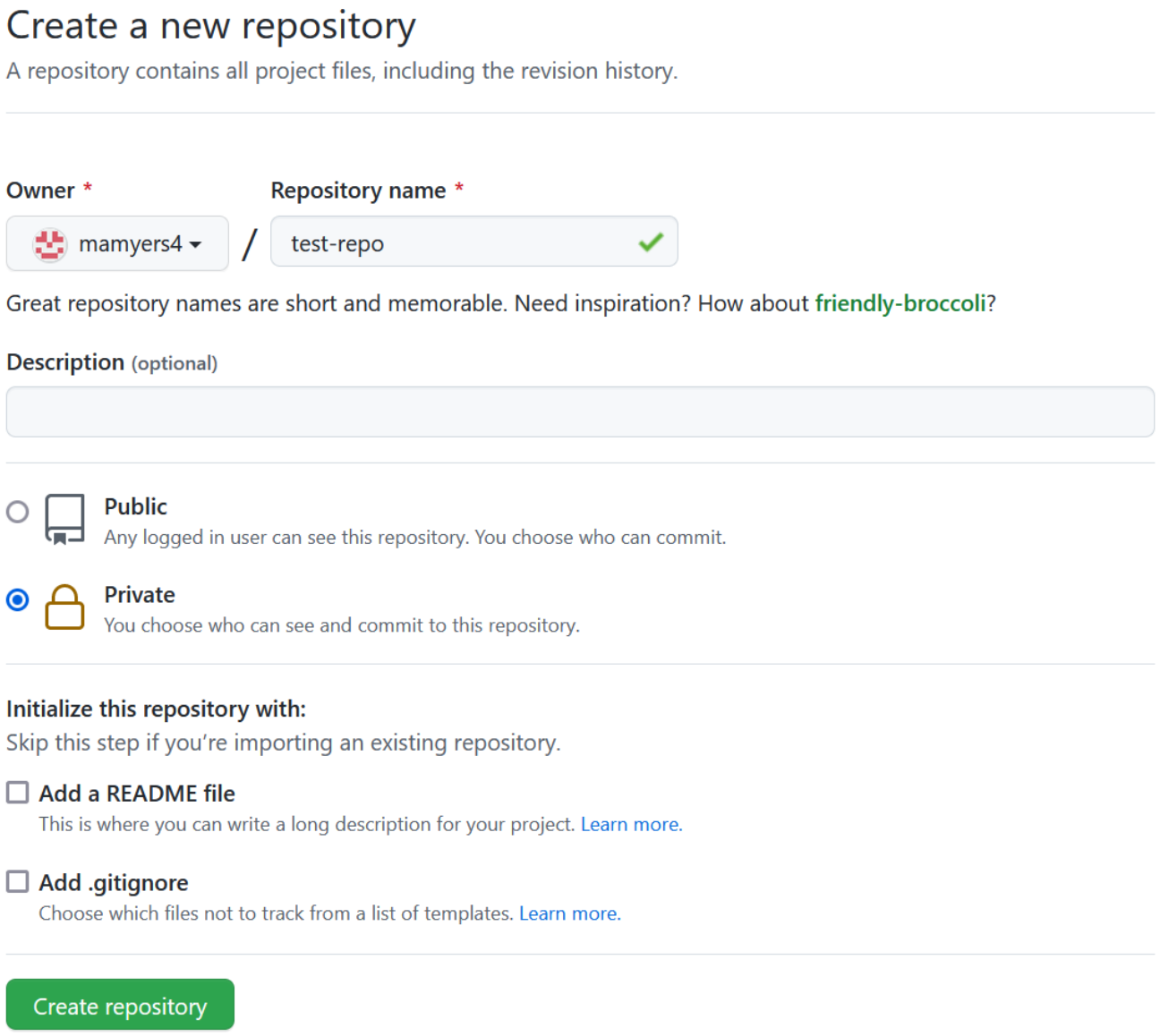
GitHub 101

- You have an NC State GitHub Enterprise account. Log in at https://github.ncsu.edu/login
- Your code is under Repositories
- You should make a different repo for each code base

Making and Managing Repos

Making a new repo

- To make a new repo, Repositories → New
- Give the repo a suitable name
- Make sure your repo is PRIVATE
- You can skip the README and .gitignore for now



Setting up the repo

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH https://github.ncsu.edu/mamyers4/git-talk.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# git-talk" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.ncsu.edu/mamyers4/git-talk.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.ncsu.edu/mamyers4/git-talk.git
git branch -M main
git push -u origin main
```

README.md & .gitignore

README.md

- Extended documentation for your code
- A place for instructions about the code's dependencies, how to compile it, recent updates, etc.
- Very helpful to future you and collaborators!

.gitignore

- Tell Git not to track certain file types
- Use it to ignore data files, etc.
- Keeps your remote repo from getting junked up
- Ex.: to ignore all .txt files, add the line '*.txt' to .gitignore

Essential commands: commit, push, & pull

- Commit: save the state of your code with a short message

```
git add changed_file.py
git commit -m “fixed infinite loop”
```

- Push: send your commit(s) to the remote repo

```
git push
```

- Pull: download what is in the remote repo

```
git pull
```

Local vs remote repo

Local repo

- Just on your computer
- The copy of the code you are running/making changes to
- Commits are only visible here until they are pushed
- What collaborators can't see

Remote repo

- Stored on GitHub
- Where you send changes to be backed up
- Doesn't know about your commits until you push them
- What you get when you pull
- What your collaborators can see

Branches

- Useful if you need to maintain multiple versions of a code, for ex., the current version of a simulation code and a new version you are working on that adds magnetic fields

- You can switch between branches and commit to both

Make new branch: `git branch magfld`

Switch to main branch: `git checkout main`

Switch to new branch: `git checkout magnetic`

Merge branches: `git checkout main && git merge magfld`

Viewing revision history

main | 1 branch | 0 tags | Go to file | Add file | Code

mamyers4 make bins according to event rate, not mass 3181819 4 days ago 17 commits

__pycache__	one histogram for each star	2 months ago
data	before changing to 1 probability curve for each star	10 days ago
explode_or_fail	one histogram for each star	2 months ago
mass_files	one histogram for each star	2 months ago
plots	work on 3 bins per decade for events	4 days ago
warren_plots	one histogram for each star	2 months ago
README.md	Initial commit	7 months ago
draw_snoopy.py	draw_snoopy	4 months ago
explode_or_fail.py	one histogram for each star	2 months ago
nearby-stars-test.py	make bins according to event rate, not mass	4 days ago
stars.txt	one histogram for each star	2 months ago
warren_models.py	one histogram for each star	2 months ago

Clicking the brackets lets me view the main branch at commit 3181819

main

Commits on Jan 27, 2022

make bins according to event rate, not mass mamyers4 committed 4 days ago	3181819
fixed bins mamyers4 committed 4 days ago	0f28c70
work on 3 bins per decade for events mamyers4 committed 4 days ago	d4bd95e

Other useful Git commands

- Reset your local repo to a past commit – copy commit id by clicking squares next to commit message in previous slide

```
git reset --hard [commit id]
```

- Show changes staged for commit

```
git status
```

- Remove your changes to ex.py from the commit

```
git rm ex.py
```

Disaster Mitigation Strategies

Merge conflicts

- This happens when the commit you're trying to push isn't compatible with the remote repo and Git asks you what to do
- Common issue when you haven't kept your local repo up to date with the remote (not pulling)
- This situation can be difficult to resolve and is best avoided

Best practices

- Commit and push often to save your code
- Pull often if you have collaborators or are using multiple computers – but PUSH BEFORE YOU PULL
- Write informative commit messages!!!
- Keep your README.md up to date

What if you ruined everything?!?!

- Example: you haven’t pulled in 6 months and have a lot of merge conflicts you don’t know how to deal with
- The easiest thing to do is just clone the repo again in a new directory (any changes not pushed before the disaster will not be there)
- Compare the new repo with the mistake repo and implement any changes in the new repo
- Use the new repo going forward and delete the mistake repo

Resources

Cheat sheets

- GitHub's Git cheat sheet
- Atlassian's Git cheat sheet
- Pick the one you like best, print it out, tape it above your desk

Free Git/GitHub courses/videos

- You have free access to LinkedIn Learning through NC State
- Learning Git and GitHub - LinkedIn Learning
- Git and GitHub for Beginners - YouTube

FIN