Lecture 2: Data in Linguistics, Git/GitHub, Jupyter Notebook

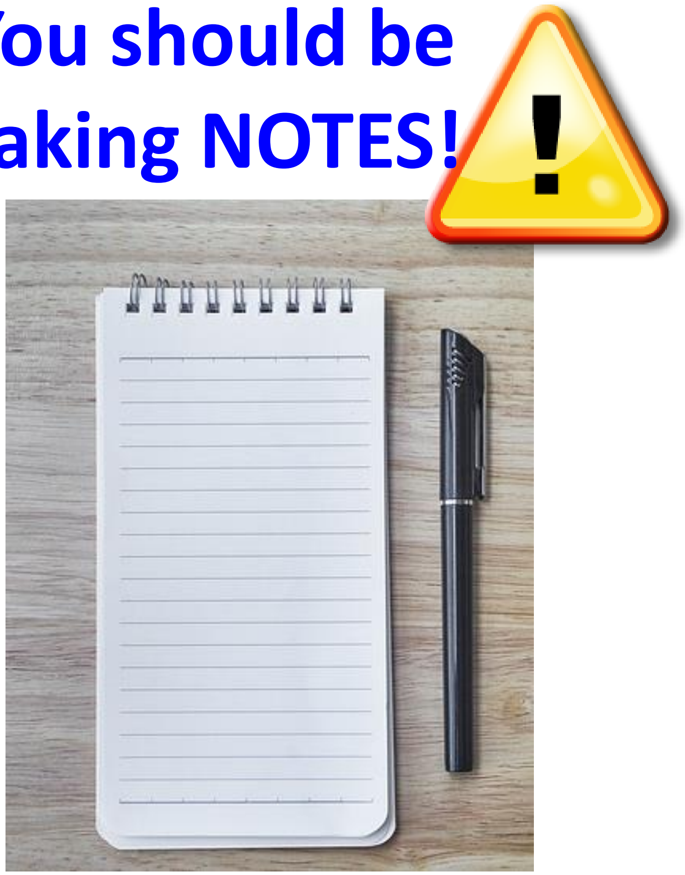
LING 1340/2340: Data Science for Linguists

Na-Rae Han

Objectives

- Tools:
  - Git and GitHub
  - Jupyter Notebook
- To-do #1
  - What do linguistic data look like?
- If time:
  - Review Term Project requirement
  - Student introduction

You should be taking NOTES!



First thing to do every class

1. Open up a Terminal/Git Bash window ("shell" window).
2. Move into your Data_Science directory.

   `cd Documents/Data_Science`

   Hit TAB for auto-completion.

3. Make sure you are in the right directory.

   `pwd`

   "Print Working Directory"

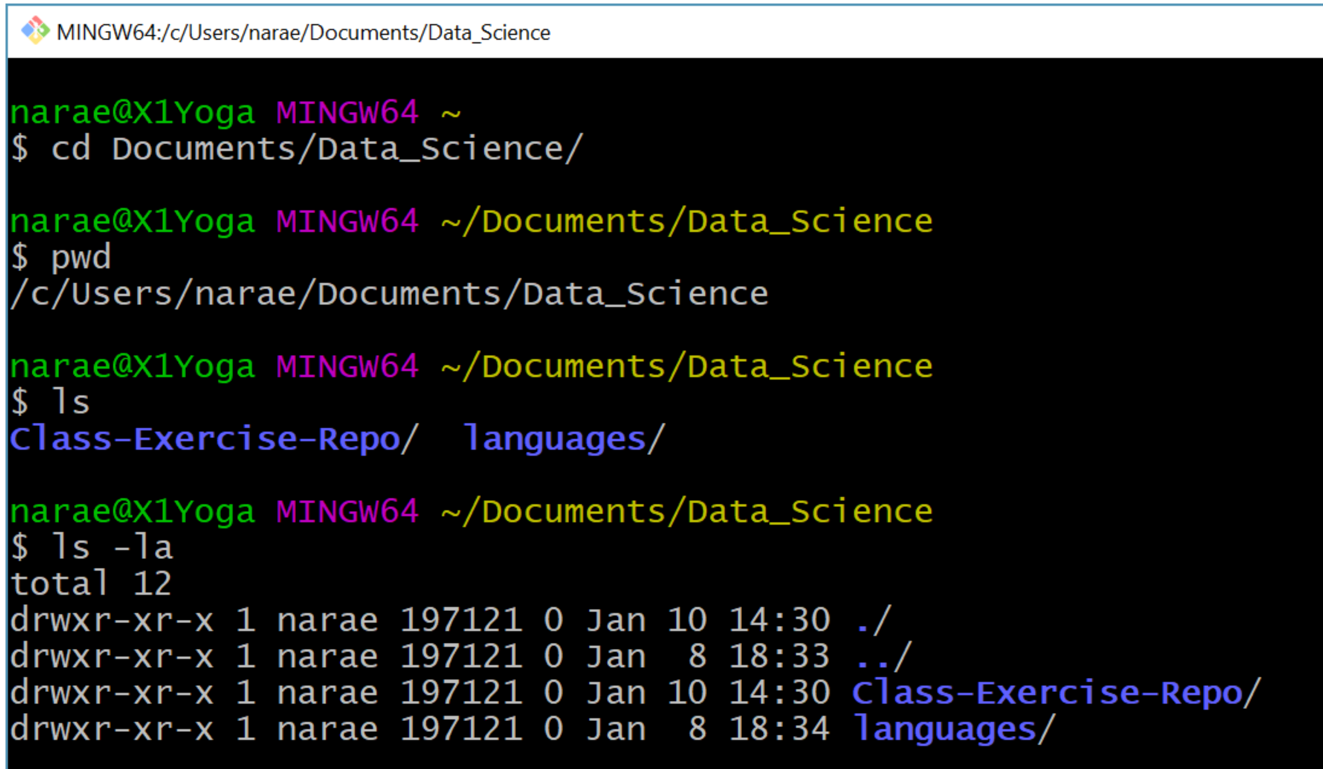
4. Look at what's inside the directory.

   `ls`

   or

   `ls -la`

   `ls` for "list directory".
   `-la` for "long/all". Shows all hidden files in long output.



MINGW64:/c/Users/narae/Documents/Data_Science

```
narae@X1Yoga MINGW64 ~
$ cd Documents/Data_Science/

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ pwd
/c/Users/narae/Documents/Data_Science

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ ls
Class-Exercise-Repo/  languages/

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ ls -la
total 12
drwxr-xr-x 1 narae 197121 0 Jan 10 14:30 ./
drwxr-xr-x 1 narae 197121 0 Jan  8 18:33 ../
drwxr-xr-x 1 narae 197121 0 Jan 10 14:30 Class-Exercise-Repo/
drwxr-xr-x 1 narae 197121 0 Jan  8 18:34 languages/
```

Your first local repository: getting started

Follow steps in Tutorial Part 1, Creating a Repository

1. Create a directory called `languages`
2. Initiate it as a Git repository:
   `git init`
3. Create a new text file 'zulu.txt', add lines to it
4. Add files to staging area:
   `git add zulu.txt`
5. Commit the change:
   `git commit -m "started zulu"`
6. Edit the text file again
7. Add files to be committed:
   `git add zulu.txt`
8. Commit the change:
   `git commit -m "details on…"`

Check status between steps:
`git status`

Your first local repository: tracking, history

Follow steps in Tutorial Part 1: Tracking Changes, A Commit Workflow, and Exploring History.

- To view entire version history:

  ```
  git log
  ```

- To view changes:

  ```
  git diff
  git diff HEAD~1 file.txt
  git diff --staged
  ```

- To view what changed in a particular version:

  ```
  git show HEAD~1
  ```

- To scrap new changes since the last commit:

  ```
  git checkout HEAD file.txt
  ```

- To restore an earlier version:

  ```
  git checkout VERSION file.txt
  ```

  🡐 commit to make this the new HEAD

If thrown into pagination, use SPACE to page down, q to quit.

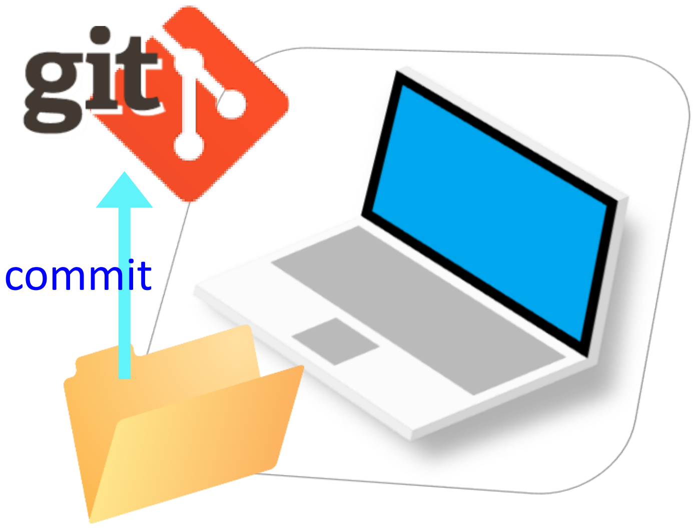
HEAD: the last committed version
HEAD~1: one before that

Your first local repository

- Your directory `languages` was set up with a **Git repository**.

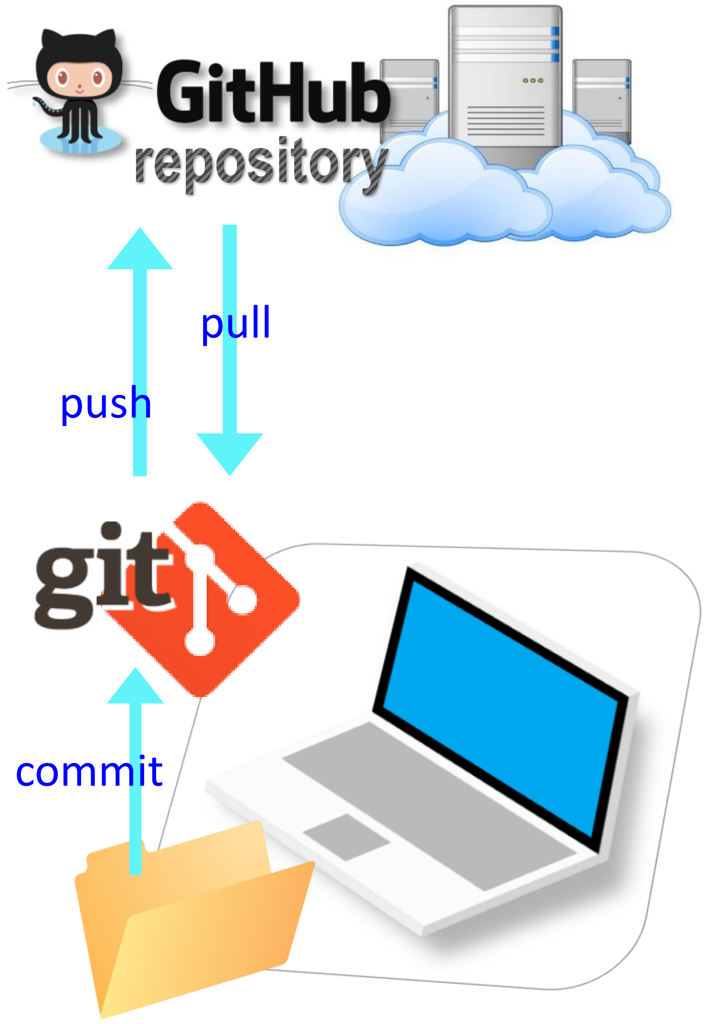
- `languages` is now:
  - tracked by Git
  - all changes will be documented
  - able to revert back to earlier version, if needs be

- But is this all?
  - How about backup? collaboration? social?



GitHub: a *remote* repository

- This is where **GitHub** comes in.
- GitHub is a **repository hosting service**.
  - A website where you can keep a copy of your Git repository.
  - REMOTE repository on GitHub, LOCAL repository on your laptop.
  - Great way to backup, and also showcase your work



Setting up a remote ("GitHub") repo

- There are TWO main methods of setting up a remote GitHub repo.

Scenario 1: Your laptop already has an **existing LOCAL Git repo**. You configure it to link it up to a new, empty repo on GitHub, then push up the content.

- We can set up our `languages` repo with a GitHub repo this way.
- Part 2 Linking Git with GitHub goes this route.

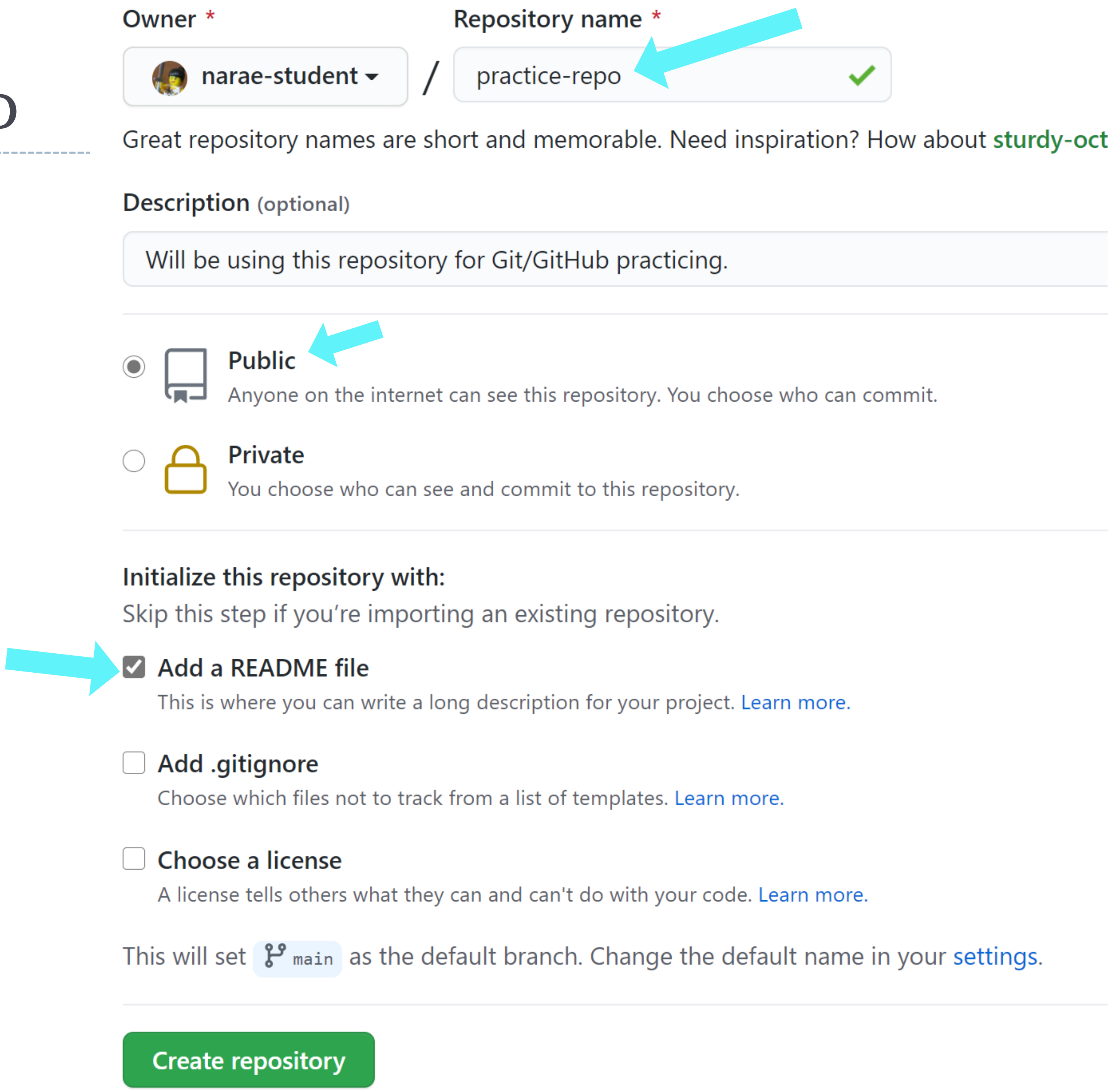
Scenario 2: Start from scratch. Create a **new repository on GitHub**, and then **clone it onto your laptop** as a brand-new local repository.

- This YouTube tutorial shows you how.

🡐 Let's have you try this.

Your first GitHub repo

- On GitHub, create a new repository called "practice-repo".
  - Provide a short description.
  - Keep it public.
  - Initialize it with a README.



Owner * narae-student / Repository name * practice-repo ✓

Great repository names are short and memorable. Need inspiration? How about **sturdy-oct**

Description (optional)

Will be using this repository for Git/GitHub practicing.

● **Public**
Anyone on the internet can see this repository. You choose who can commit.

○ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☑ **Add a README file**
This is where you can write a long description for your project. Learn more.

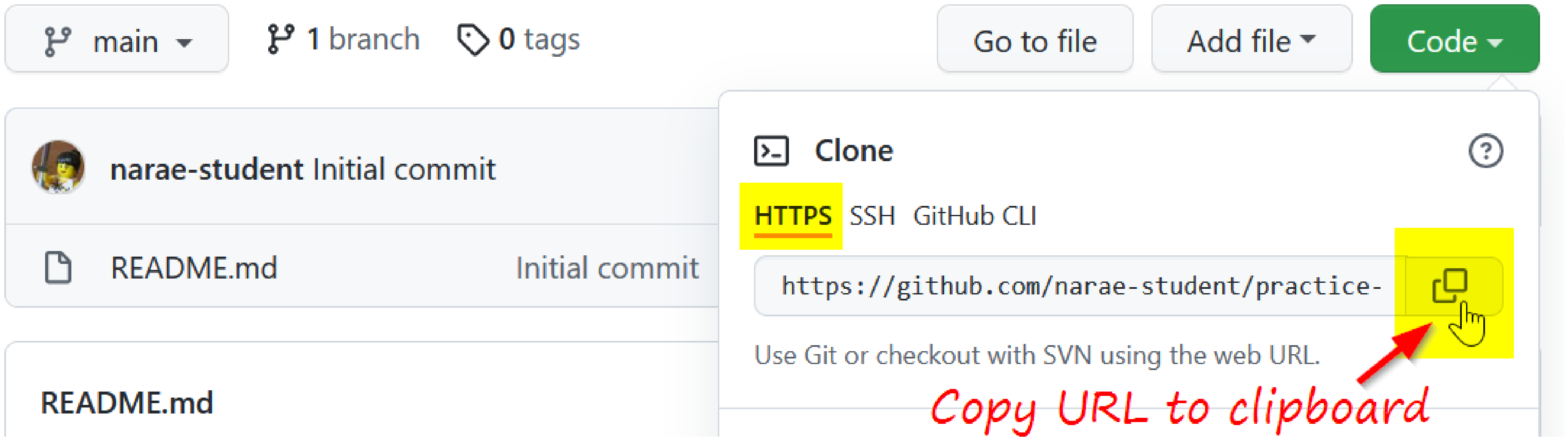
☐ **Add .gitignore**
Choose which files not to track from a list of templates. Learn more.

☐ **Choose a license**
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

Cloning first GitHub repo

- GitHub shows a URL to use in cloning. Copy to clipboard.



https://github.com/narae-student/practice-

Use Git or checkout with SVN using the web URL.

- In Terminal/Git Bash, move into your Data_Science/ directory (use `cd` command,) then execute (paste copied URL):

`git clone https://github.com/`*`yourid`*`/practice-repo.git`

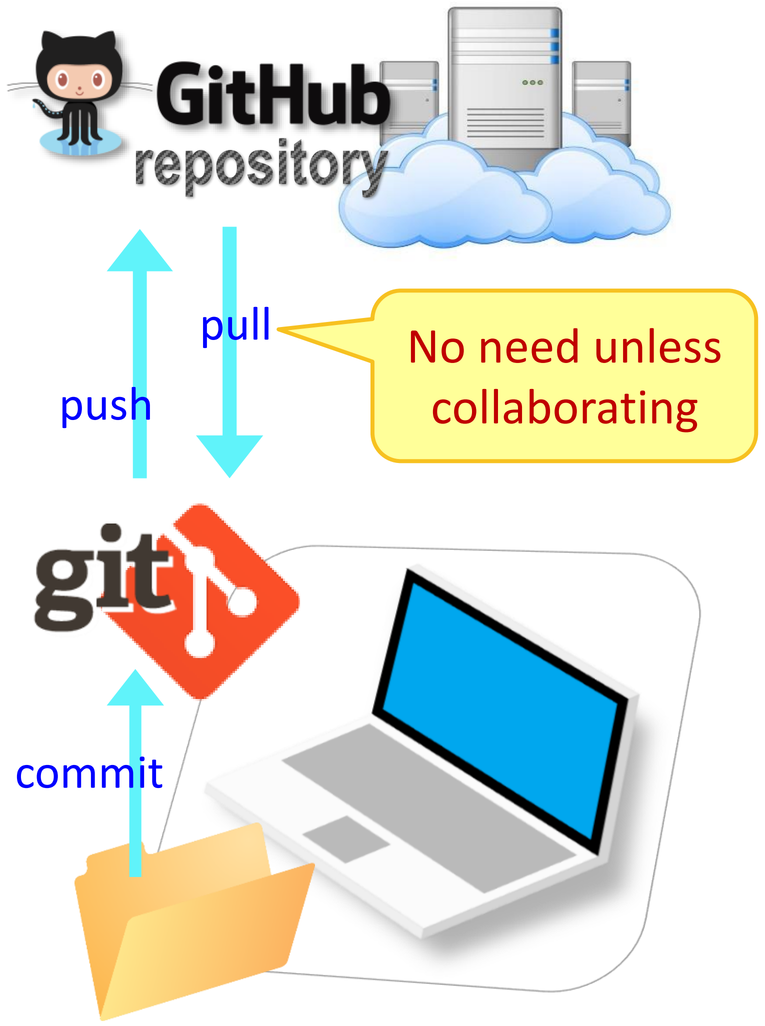
← practice-repo directory is cloned as a local repository.

Local repository ↔ remote repository

- After committing, you now need to *push* to remote repo.
  1. Create a new text file 'notes.txt'
  2. Add files to be committed:
     `git add notes.txt`
  3. Commit:
     `git commit -m "first commit"`
  4. **Push change to GitHub: `git push`**
  5. Edit the text file
  6. Add files to be committed:
     `git add notes.txt`
  7. Commit:
     `git commit -m "changed x, y, z"`
  8. **Push change to GitHub: `git push`**

Check frequently:

```
git status
git diff
git log
```



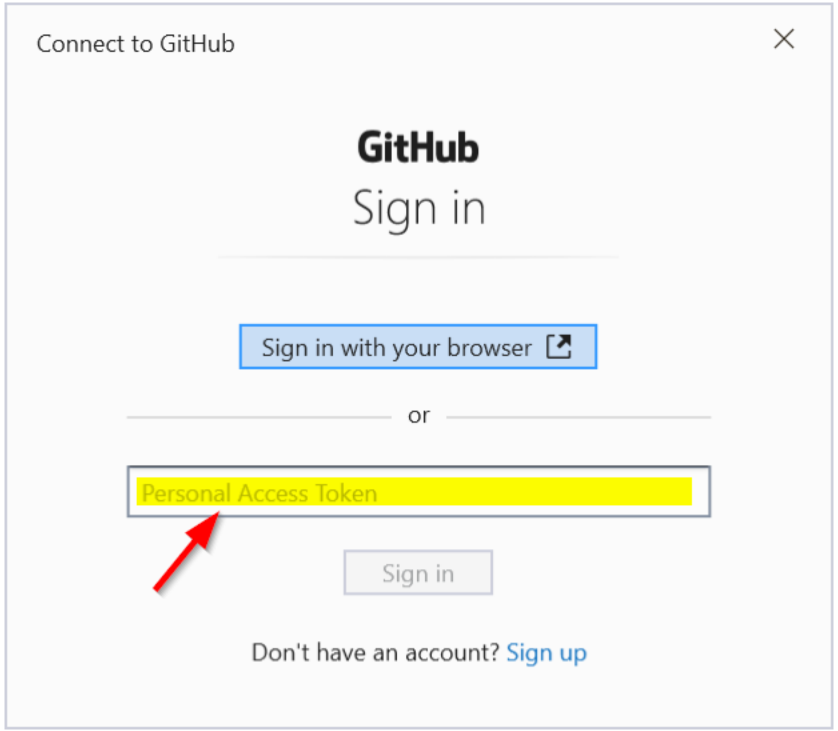
GitHub's Personal Access Token

- Looks something like: `ghp_uTzu7a11BpYoog3luLicbCAIOhqCDhpL5qUi`
- This replaces your password. But it's not for remembering and typing in (obviously…)

➔ Solution: store in your OS's credential manager

- Windows: "Manage Windows Credentials"
- Mac OS: "Keychain Access"

- For details, see
  - https://www.cloudsavvyit.com/14870/how-to-set-up-https-personal-access-tokens-for-github-authentication/



Try it out: Your first git/GitHub repo

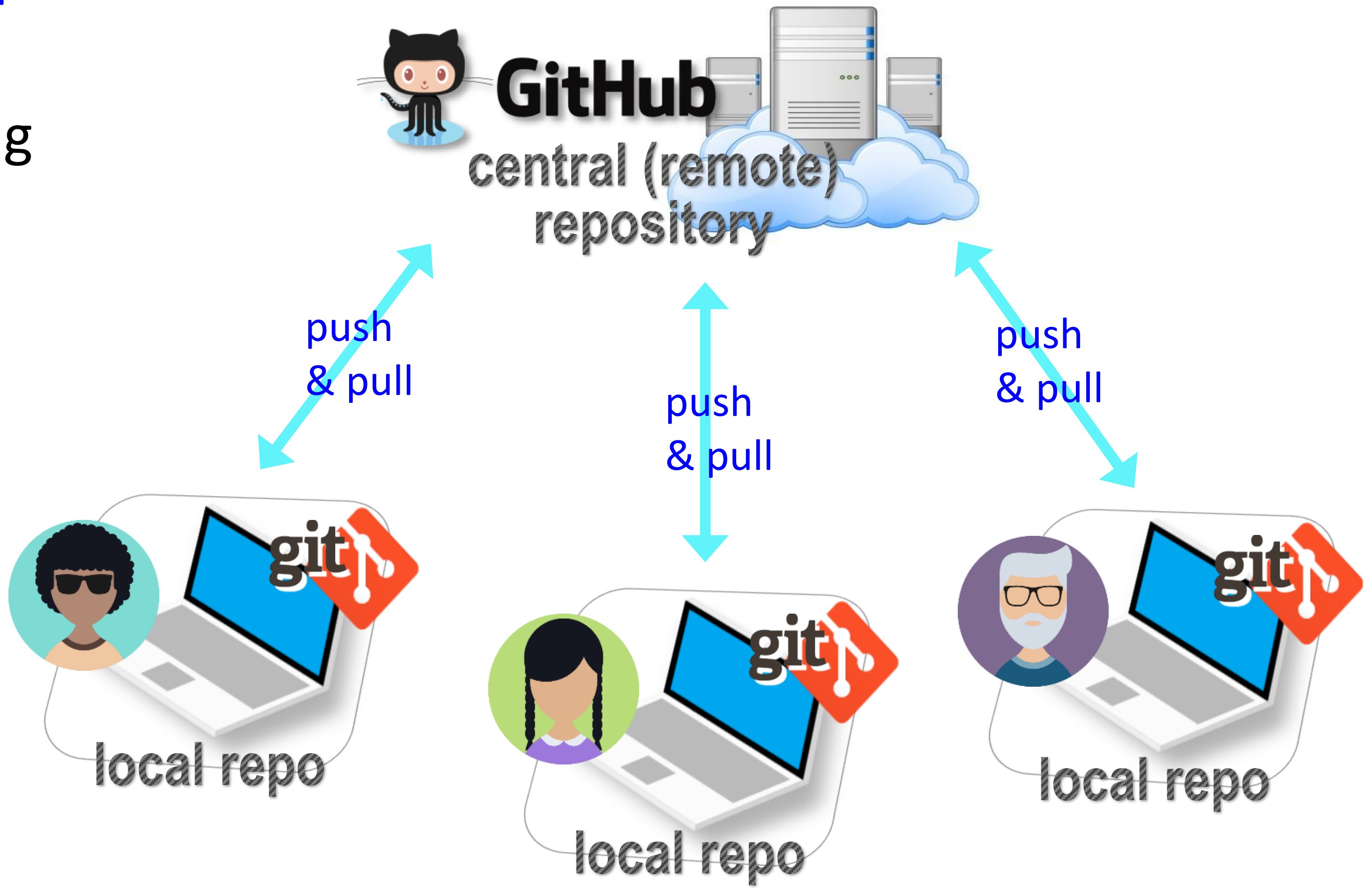
5 minutes

- Create "Practice-Repo" on GitHub
- Clone it to your laptop
- Make changes to your local repo
- Git operations: git status, git add, git commit

*Avatar icons by FLATICON

GitHub: a *social*, remote repository

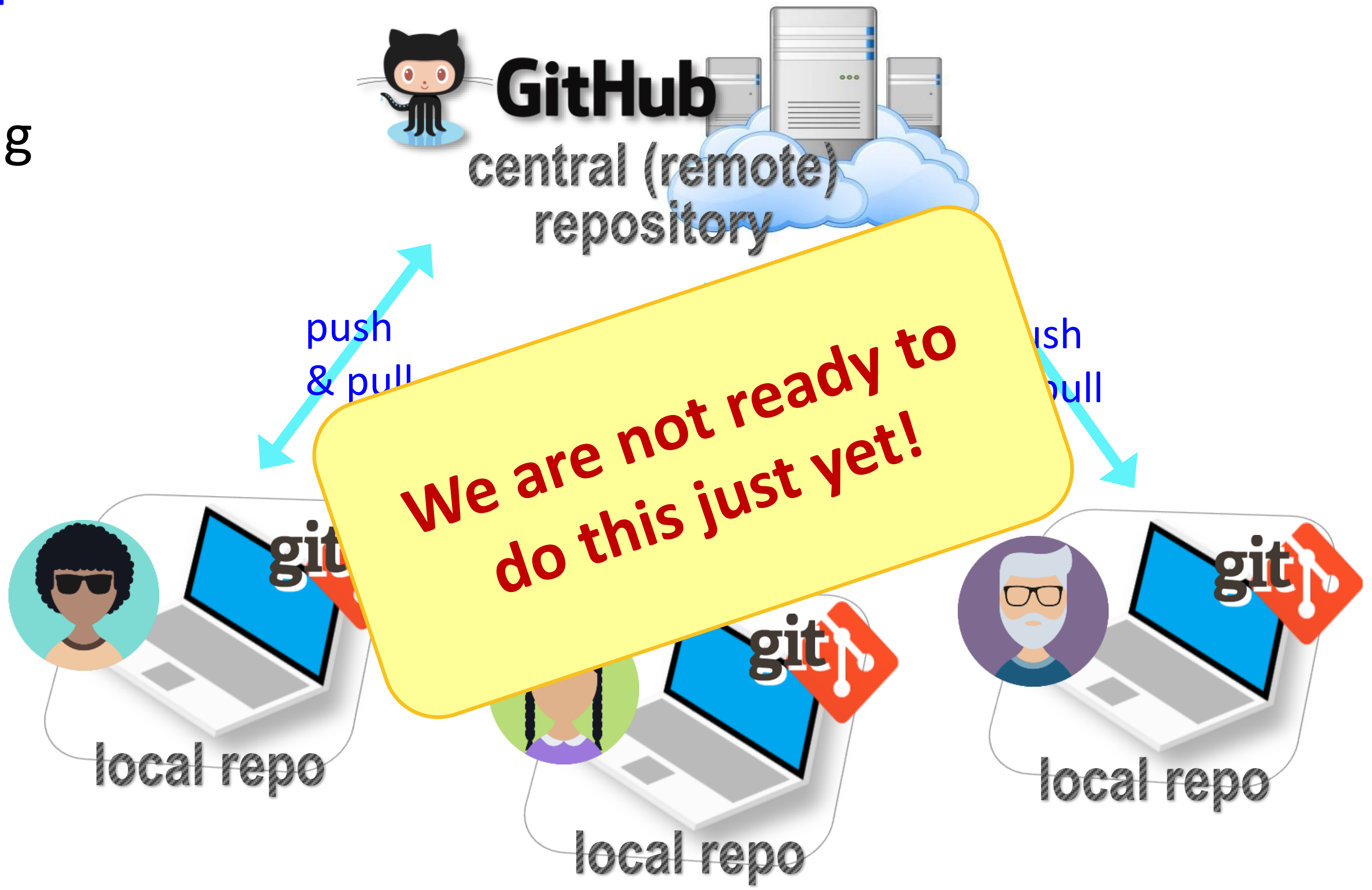
- GitHub also works as a central remote repository among a group of **collaborators** working on a shared project.
  - Everyone works on their own *local* copy of the repository, making changes.
  - Git is able to keep track and merge changes submitted by everyone.



*Avatar icons by FLATICON

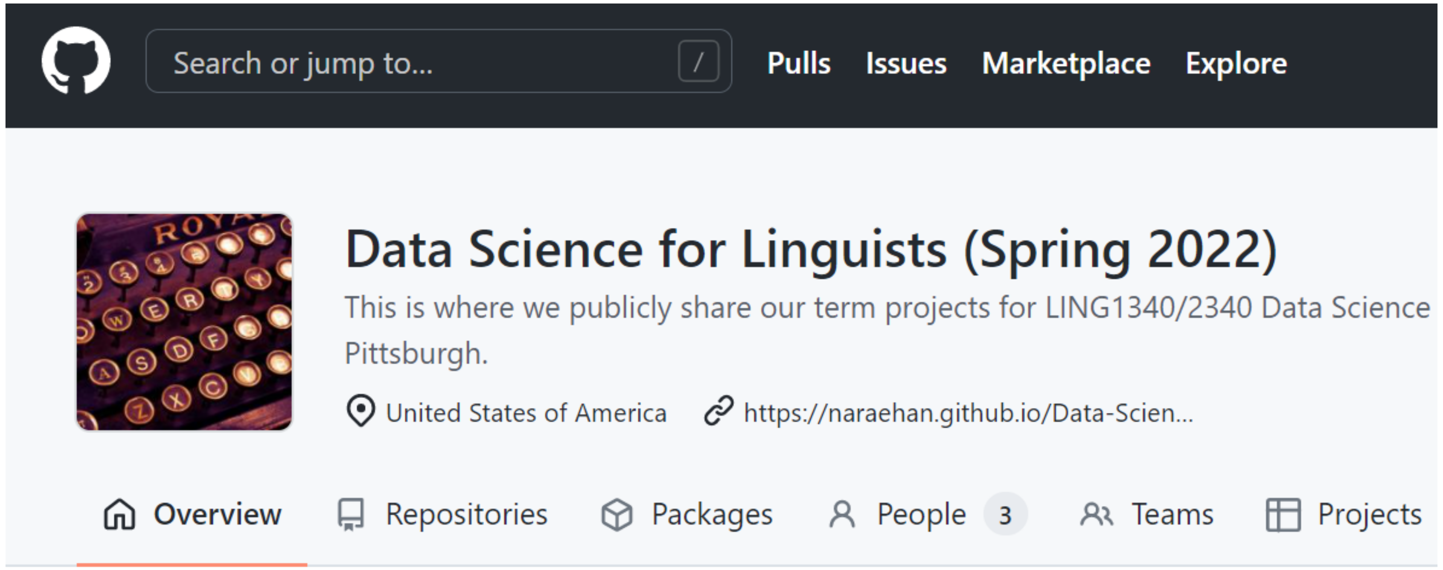
GitHub: a *social*, remote repository

- GitHub also works as a central remote repository among a group of **collaborators** working on a shared project.
  - Everyone works on their own *local* copy of the repository, making changes.
  - Git is able to keep track and merge changes submitted by everyone.
  - Everyone is an **equal collaborator** with push (=write) access.


GitHub
central (remote) repository
push
& pull
ush
ull
git
local repo
local repo
local repo
We are not ready to do this just yet!

Introducing... GitHub Class Organization

- https://github.com/Data-Science-for-Linguists-2022



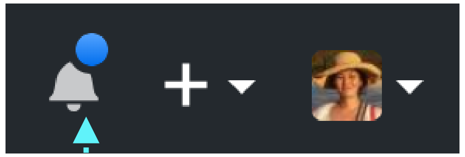
Search or jump to... Pulls Issues Marketplace Explore

Data Science for Linguists (Spring 2022)

This is where we publicly share our term projects for LING1340/2340 Data Science Pittsburgh.

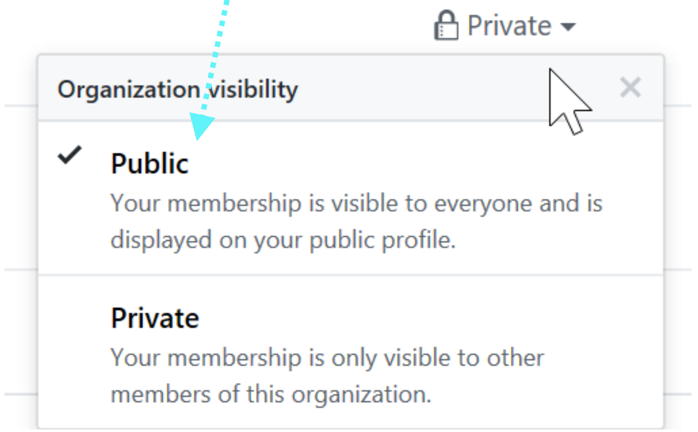
United States of America https://naraehan.github.io/Data-Scien...

Overview Repositories Packages People 3 Teams Projects



Accept invitation from "notifications" page

Click "People", and make your membership public



Private

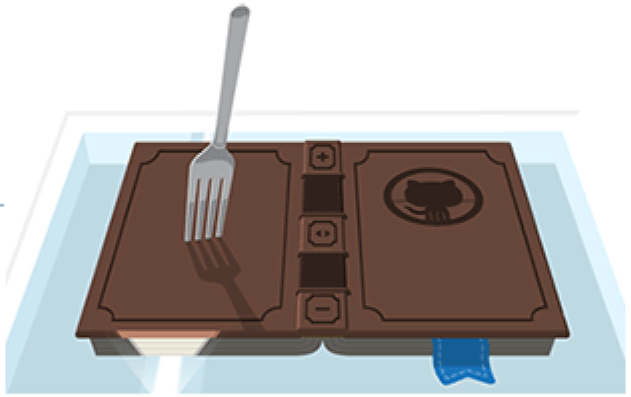
Organization visibility

✓ Public
Your membership is visible to everyone and is displayed on your public profile.

Private
Your membership is only visible to other members of this organization.

- So we can:
  - have everyone in one spot.
  - have all class materials in one spot.
  - have everyone's term project in one spot.
  - share *private* repos as a group.

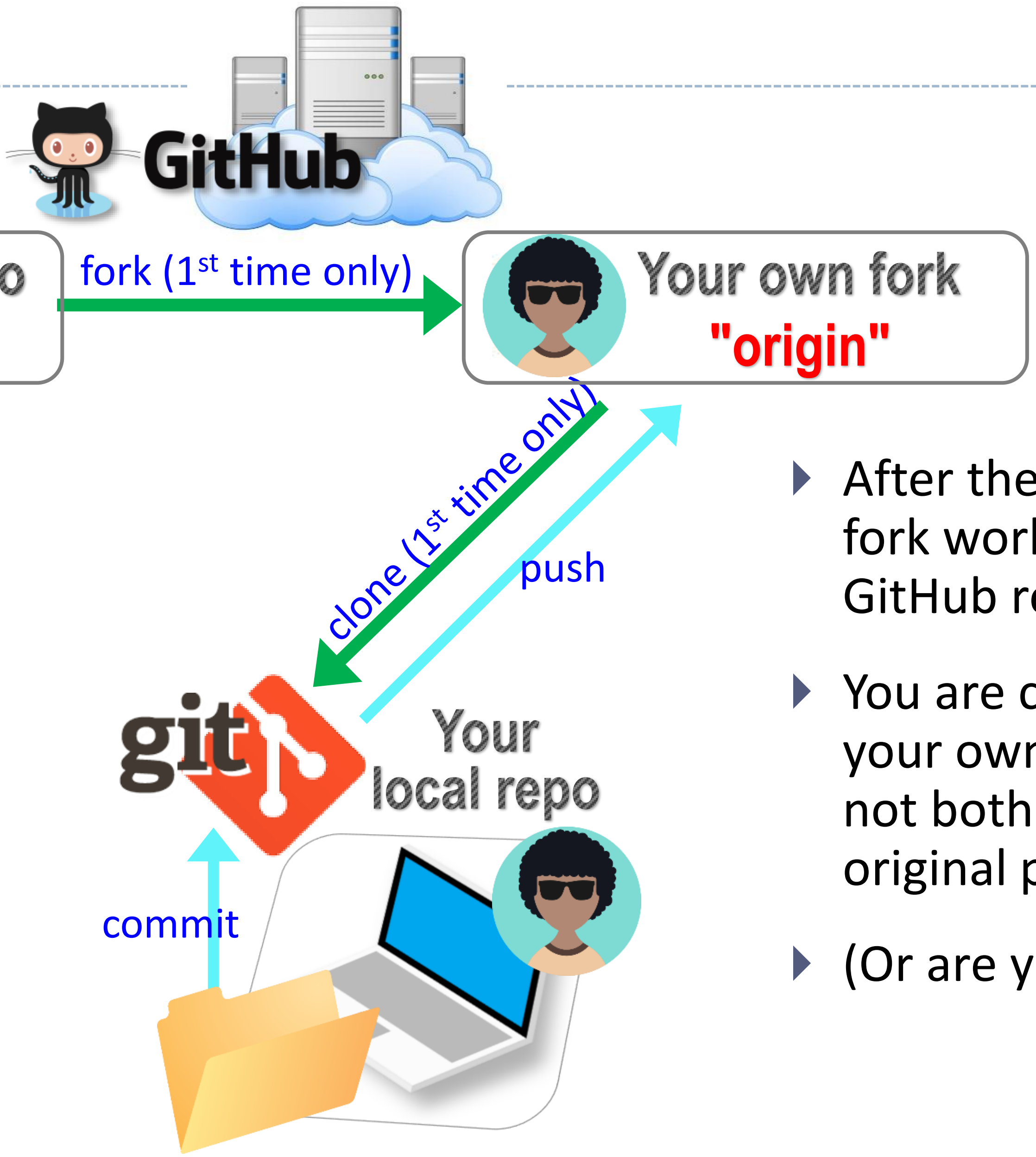
But first, forking



- When you **start with someone else's project**.
  - You are *not* a collaborator in their repo. (No push access)
- https://help.github.com/articles/fork-a-repo/
- You **fork** the original repo into your own GitHub account, creating your own "fork".
- You make changes in your own fork. The original repo is not affected!

*Avatar icons by FLATICON

Forking, one-way



- After the spin-off, your fork works as if your own GitHub repo.
- You are content to do your own development, not bothering the original project owner…
- (Or are you??)

Your first fork

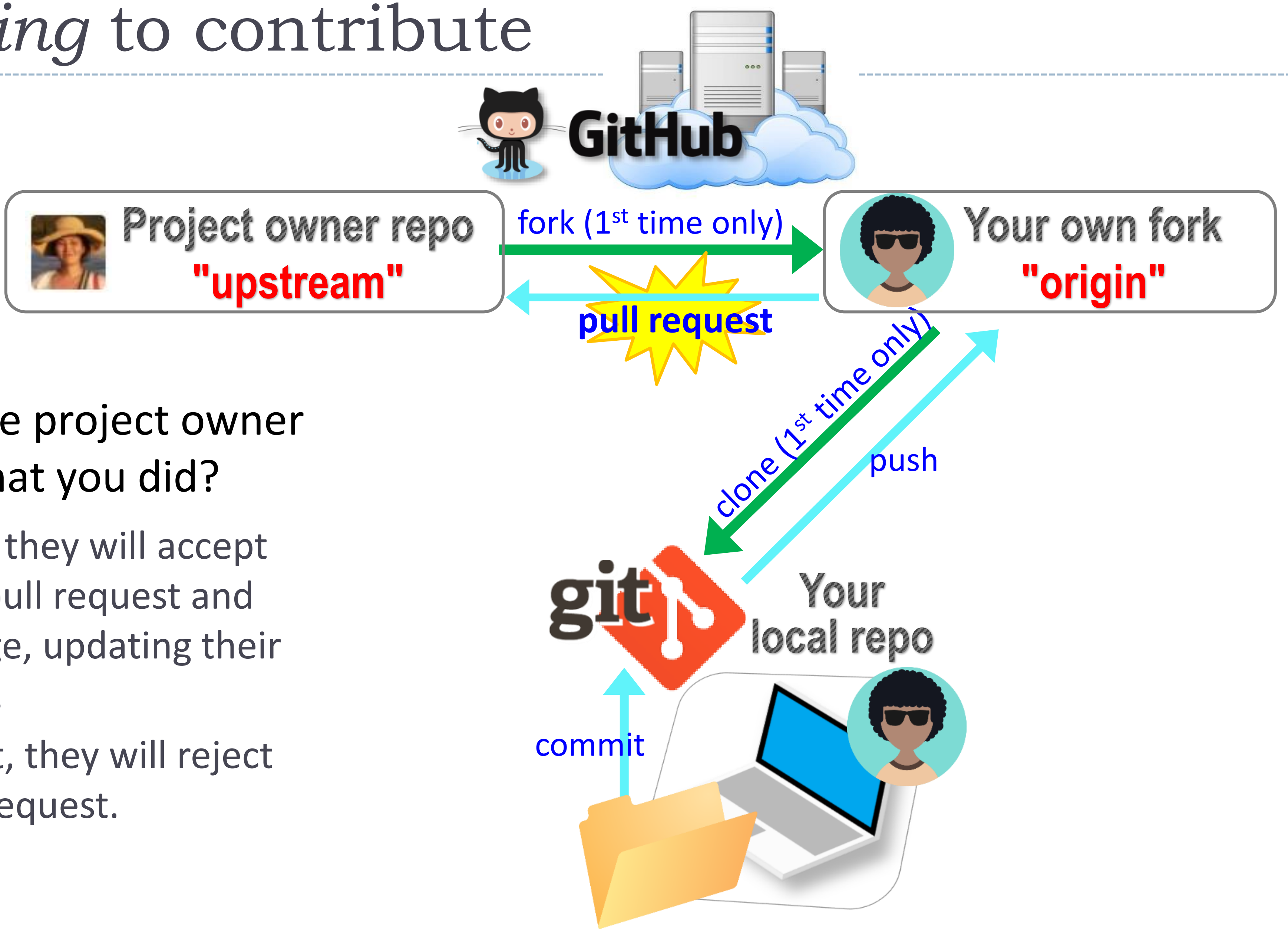
- On **GitHub**:
  1. Go to class GitHub org.
  2. Fork "Class-Exercise-Repo". You will now have the exact same content in your own GitHub account.

- On your **laptop**:
  1. Move into your `Data_Science/` directory. Clone your fork there via git clone URL.
  2. Copy over your To-Do1 submission file into `todo1/` directory. Make sure the file name has your name in it: lingdata_narae.txt etc.
  3. Add, commit, and then push to your fork.
- Back on **GitHub**:
  1. Confirm your GitHub fork now has your submission file.

But the original repo doesn't have this file...

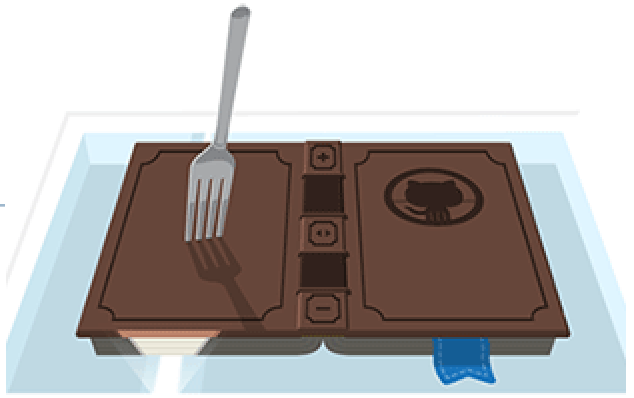
*Avatar icons by FLATICON

Offering to contribute



- Will the project owner like what you did?
  - If so, they will accept the pull request and merge, updating their repo.
  - If not, they will reject the request.

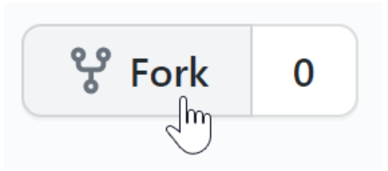
Forking: contributing back



- When you **start with someone else's project**.
  - You are *not* a collaborator in their repo. (No push access)
- https://help.github.com/articles/fork-a-repo/
- You **fork** the original repo into your own GitHub account, creating your own "fork".
- You make changes in your own fork. The original repo is not affected!
- **pull request**: When you think the original project could benefit from your new work, you ask the owner to "pull" from your fork.
  - Owner of original ("upstream") will review your contribution, and then either merge it or reject it.

Your first fork

- On **GitHub**:
  1. Go to class GitHub org.
  2. Fork "Class-Exercise-Repo". You will now have the exact same content in your own GitHub account.



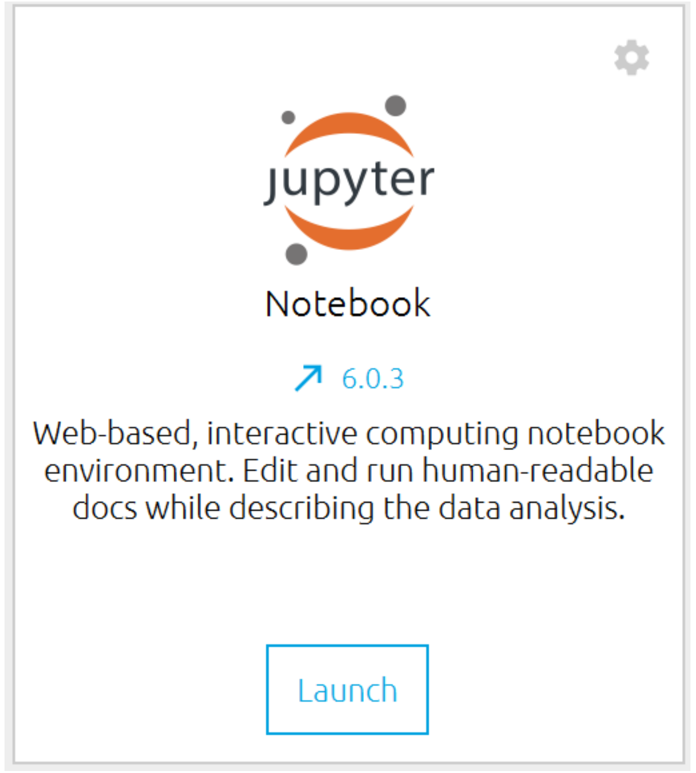
- On your **laptop**:
  1. Move into your `Data_Science/` directory. Clone your fork there via git clone.
  2. Copy over your To-Do1 submission file into `todo1/` directory. Make sure the file name has your name in it: lingdata_narae.txt etc.
  3. Add, commit, and then push to your fork.
- Back on **GitHub**:
  1. Confirm your GitHub fork now has your submission file.
  2. Create a **pull request** for Na-Rae.

Jupyter Notebook

- Allows you to create and share documents that contain live code, equations, visualizations and explanatory text.
- Learn how to use it. Your Python code should be in the Jupyter Notebook format:
  - xxxx.ipynb
- You can launch it from the command line.
  - Move into the desired directory, and then execute

    ```
    jupyter notebook &
    ```

  ← '&' is not necessary, but it lets you keep using the terminal



Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

To-do #1

- What linguistic data sets did you look at?
  - Corpus data?
  - Non-corpus data?

- What makes a dataset a corpus?

Wrapping up

- Homework #1 is out: due on Tuesday.
  - Don't be too ambitious! This HW is more about tools (git, GitHub, JNB).
- Office hours
  - Posted on Course home page.
  - We are all happy to meet by appointment.
  - Need help with Git and GitHub set up? Come to our office hours on Friday
- Start learning:
  - Git, GitHub
  - Jupyter Notebook