

COGS138: Neural Data Science

Lecture 6

C. Alex Simpkins, PhD

UCSD Dept. of Cognitive Science, Spring 2023

RDPRobotics, LLC

http://casimpkinsjr.radiantdolphinpress.com/pages/cogs138_sp23

rdprobotics@gmail.com | csimpkinsjr@ucsd.edu

(Based on a course created by Prof. Bradley Voytek)

Plan for today

- Announcements
- Assignment 1 overview
- Review - Last time
- NWB data and BIDS data - definition, accessing, usage and relevance
- DANDI - putting datasets together and making it all available, reusable and documented
- Version control, git, github review

Announcements

- Final reminder to check on your FinAID status
- A1 - due **a week from release**, which will be tonight or tomorrow
- Reading 1 - Released on canvas and in web site password protected area soon, lecture quiz due **a week from release**, released tonight
- **Group formation** - time to start choosing who you want to work with for your project group

Last time

Course links

Website	http://casimpkinsjr.radiantdolphinpress.com/pages/cogs138_sp23	Main face of the course and everything will be linked from here. Lectures, Readings, Handouts, Files, links
GitHub	https://github.com/drsimpkins-teaching	files/data, additional materials & final projects
datahub	https://datahub.ucsd.edu	assignment submission
Piazza	https://piazza.com/ucsd/spring2023/cogs138_sp23_a00/home (course code on canvas home page)	questions, discussion, and regrade requests
Canvas	https://canvas.ucsd.edu/courses/44897	grades, lecture videos
Anonymous Feedback	Will be able to submit via google form	If I ever offend you, use an example you are uncomfortable with, or to provide general feedback. Please remain constructive and polite

So far we have discussed

- Neural Data science
- Programming
- Tools for data exploration, modeling, visualization (Python, Jupyter, Matlab, others)
- NLP
- EEG, MEG, associated analysis and tools (at a high level), other imaging
- MOCAP
- Eye tracking
- Other behavioral observations

That's a lot of data!

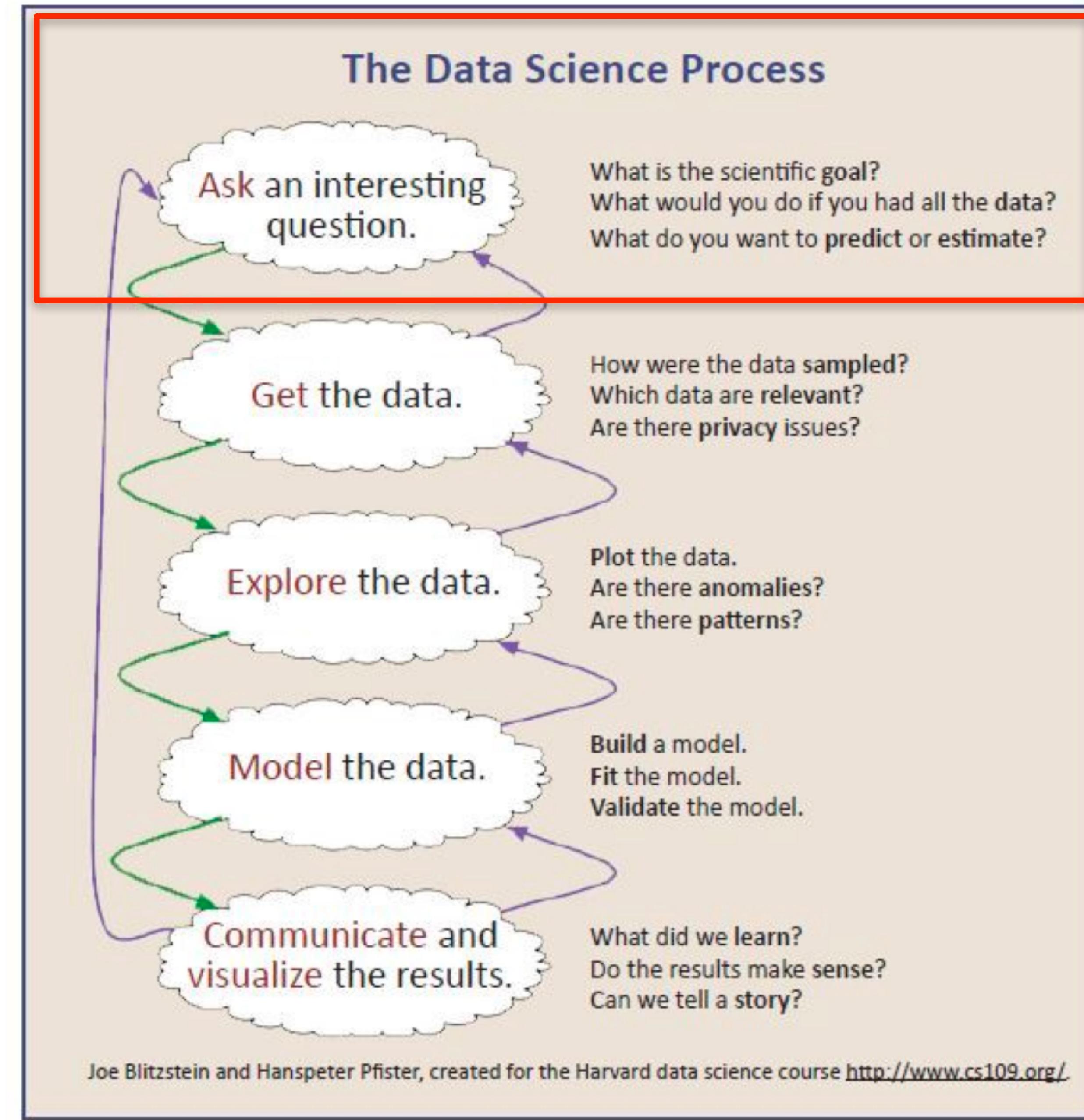
- How do you deal with it all, standardize, organize, communicate it?
- How can you talk across disciplines?
- How do you collaborate and work in teams with this?
- How can you ask questions with all that data and the results generated?

Data science questions, hypothesis generation
(automated), Genes/gene expression, animal
models, FAIR, Neurodata Without Borders (NWB),
Brain Imaging Data Structure (BIDS), DANDI

Formulating Data Science Questions

When you and your group sit down to figure out what you're going to do for your final project in this class, you'll have to formulate a strong question. It should be:

1. **Specific,**
2. Can be answered with **data,**
3. And makes **clear what** exactly **is** being **measured.**



adapted from Chris Keown

Hypothesis testing

- Cannot prove hypothesis
- Can only reject or fail to reject null hypothesis
- Why?

Data Science questions should...

- Be specific
- Be answerable with data
- Specify what's being measured



What makes a
question a good
question?

Specifying what you're going to measure is important

Examples of poor questions that leave wiggle room for useless answers:

- What can my data tell me about the brain?
- What should I do about the brain?
- How can I increase my neuroscience?

Examples of good questions where the answer is impossible to avoid:

- Does a subject's reaching trajectory change when put under a static force field? Is this change static or dynamic?
- What is the average/maximum grip strength required to manipulate a pen during writing tasks (pen and paper)?
- What is the minimum light intensity perceptible by the average subject of age range 18-24yrs in pitch black darkness for a point light at a distance of 2m?

Working toward a strong data
science question

Working toward a strong data science question

Vague: How does the brain change when you have a brain injury?

Better: What neurological changes are there after a stroke?

Even better: What neurological and behavioral changes can be measured with EEG and motion capture between an average normal subject and a stroke patient who had a recent stroke that impaired motor function?

Best?

Practicing asking questions . . .

- Could reflex be measured with brain activity

Previous questions asked during this class's
projects...

Genes and text, LISC

()

- Leveraging LISC and NLTK for research like gene expression studies
- Creating gene dictionaries
- Looking through literature to collect information about topics of interest, data and results using python (LISC)

LISC project

- Open source python module “Literature Scanner”
 - <https://github.com/lisc-tools/lisc>
- Donoghue, Thomas. (2019). LISC: A Python Package for Scientific Literature Collection and Analysis. *Journal of Open Source Software*. 4. 1674. 10.21105/joss.01674.
- https://www.researchgate.net/publication/336082537_LISC_A_Python_Package_for_Scientific_Literature_Collection_and_Analysis
- LISC is based on BRAIN-SCANR by Voytek (2012)

LISC- Automated methods for digesting vast information

()

- Scientific literature is vast, expanding and beyond a single researcher's ability to digest completely
- By the time an article is read, more are published
- >30M published articles as of 2019 in biomedical sciences alone!
- Automated methods for curation and digestion of literature has been explored to enhance a researcher's abilities to absorb information
- "Knowledge discovery, literature-based discovery, hypothesis generation"

LISC- Automated methods for digesting vast information

()

- Easily accessible
- Connects to several external resources through APIs
- e.g. PubMed, OpenCitations database
- Supports utilities to analyze collected data

LISC- types of data collection

()

- **Counts:** tools to collect and analyze data on the co-occurrence of specified search terms
- **Words:** tools to collect and analyze text and meta-data from scientific articles
- **Citations:** tools to collect and analyze citation and reference data

LISC- includes for supporting use cases

()

- URL management and requesting for interacting with integrated APIs
- Custom data objects for managing collected data
- A database structure, as well as save and load utilities for storing collected data
- Functions and utilities to analyze collected data
- Data visualization for plotting collected data and analysis outputs

LISC vs. Moliere

- LISC takes a lightweight, fast and efficient approach to hypothesis generation
- A complement for other tools like Moliere or Meta (www.meta.org)
- More customizable (LISC), tools included for efficient analysis on the results
- Connective interface to Natural Language Processing (NLP) tools such as NLTK
- Moliere/Meta better for more complex analyses

Caveats

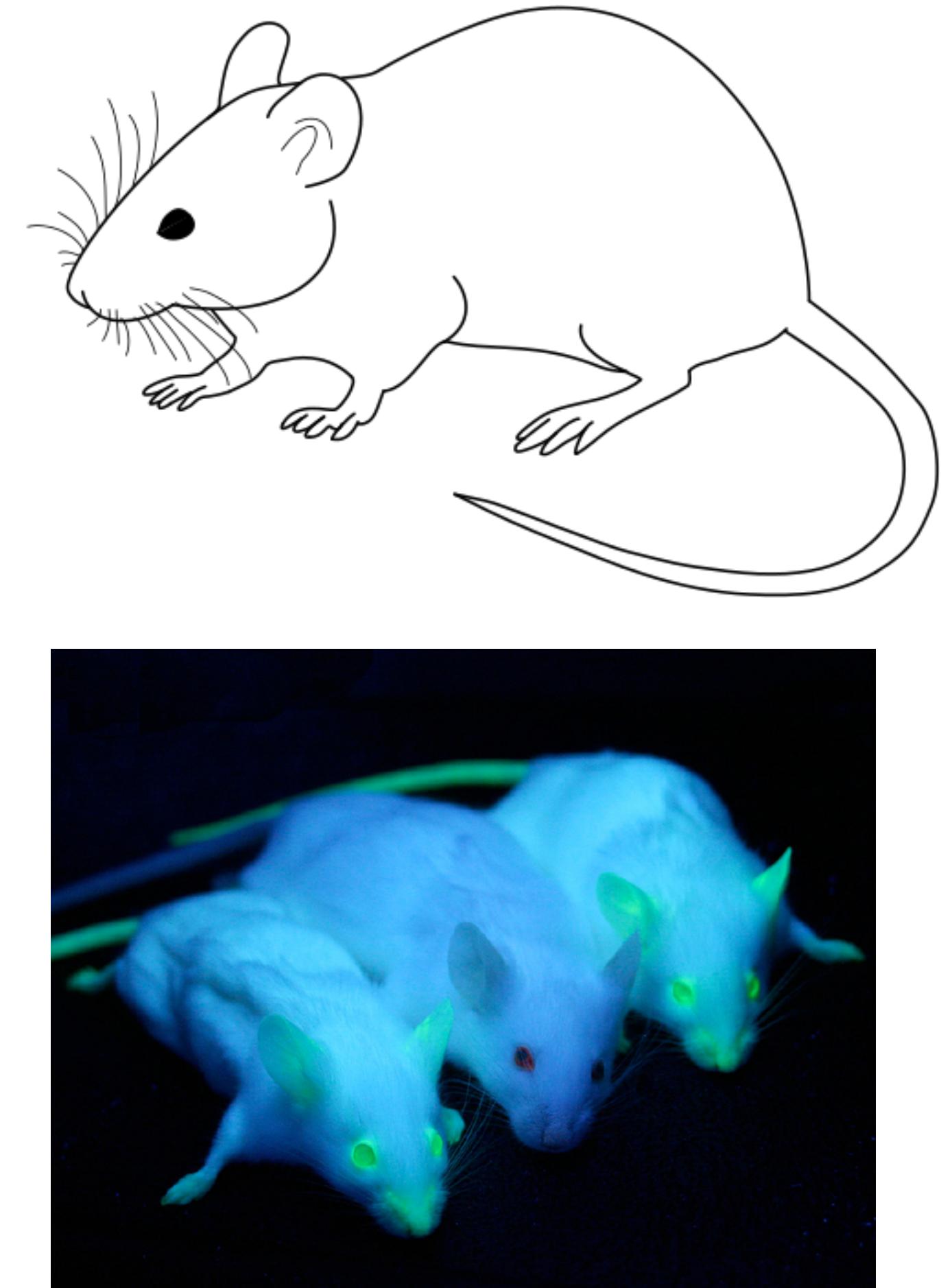
- Take care using automated systems since they don't "understand" the literature as a human does
- Programming biases are inevitable
 - Chatbot knowledge biases
 - Programmer biases
- Statistics can be biased
- Use with a grain of salt - it's a tool
 - ***"The hammer does not make the building" [Simpkins 2023]***

Gene expression studies

- **Gene expression definition** - *the process by which the information encoded in a gene is turned into a function. This mostly occurs via the transcription of RNA molecules that code for proteins or non-coding RNA molecules that serve other functions.*

Why Animal Models?

- We use ***animal models*** for gene expression because, unless a human is undergoing brain surgery where tissue can be sampled, ***we cannot currently measure*** gene expression in the brain otherwise
 - So to avoid harming a human (ethics are complicated!)
- Animals are found that have certain genomic similarities and assumptions are made about mapping behaviors, diseases and gene patterns into insights about humans
- Often an animal is bred for the study with specific genes or “knockouts” are created with certain genes removed in order to understand effects



(Source: https://en.wikipedia.org/wiki/Laboratory_mouse)

Why **Not** Animal Models?

- Ethical considerations
- Differences between animals and humans
- Time
- Cost
- Space, resources, pollution, energy use

Alternatives to animal models

- Simulation/computational modeling
- Artificial hardware systems/embodied systems
- Organoids
- Others?

F.A.I.R.

Findable **A**ccessible **I**nteroperable **R**Reusable
Data

Science and reproducibility

- Understanding the brain requires broad, diverse and complex sets of data taken from many species of creatures, simulation, models and worldwide contributors
- The data must be findable, accessible, interoperable and reusable (FAIR)

The FAIR Data Principles

- <https://force11.org/info/the-fair-data-principles/>
- “One of the grand challenges of data-intensive science is to facilitate knowledge discovery by assisting humans and machines in their discovery of, access to, integration and analysis of, task-appropriate scientific data and their associated algorithms and workflows. Here, we describe FAIR – a set of guiding principles to make data Findable, Accessible, Interoperable, and Reusable. The term FAIR was launched at a Lorentz workshop in 2014, the resulting FAIR principles were published in 2016.”

To be Findable

- F1. (meta)data are assigned a globally unique and eternally persistent identifier.
- F2. data are described with rich metadata.
- F3. (meta)data are registered or indexed in a searchable resource.
- F4. metadata specify the data identifier.

To be Accessible

- A1 (meta)data are retrievable by their identifier using a standardized communications protocol.
- A1.1 the protocol is open, free, and universally implementable.
- A1.2 the protocol allows for an authentication and authorization procedure, where necessary.
- A2 metadata are accessible, even when the data are no longer available.

To be Interoperable

- I1. (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
- I2. (meta)data use vocabularies that follow FAIR principles.
- I3. (meta)data include qualified references to other (meta)data.

To be Re-usable

- R1. (meta)data have a plurality of accurate and relevant attributes.
- R1.1. (meta)data are released with a clear and accessible data usage license.
- R1.2. (meta)data are associated with their provenance.
- R1.3. (meta)data meet domain-relevant community standards.

FAIR Principles Working Detailed Document

- <https://force11.org/guiding-principles-for-findable-accessible-interoperable-and-re-usable-data-publishing-version-b1-0/>

On to today . . .

Neurodata Without Borders

(N.W.B.)

Introduction, tools, definitions and relevance

Use NWB for

- Use this for cellular neurophysiology, such as electrophysiology and optical physiology

NWB Definition

- <https://www.nwb.org/>
- “**Neurodata Without Borders (NWB)** is a ***data standard*** for neurophysiology, providing neuroscientists with a common standard to share, archive, use, and build analysis tools for neurophysiology data. NWB is designed to store a variety of neurophysiology data, including data from intracellular and extracellular electrophysiology experiments, data from optical physiology experiments, and tracking and stimulus data.” [www.nwb.org]

NWB Introduction

- <https://www.nwb.org/>
- <https://nwb-overview.readthedocs.io/en/latest/>
- So essentially
 - A data format for sharing/archiving
 - Standardized (set of rules and best practices)
 - Packages Data and Metadata together so human- and machine-readable

NWB Introduction

- Take advantage of established techniques for processing, analysis, visualization tools
- Makes data easier to reuse - additional scientific insights
- Essential step to getting data into the DANDI archive (<https://dandiarchive.org/>)

Brain Imaging Data Structure

(B.I.D.S.)

Introduction, tools, definitions and relevance

Use **BIDS** for

- Use for neuroimaging data such as MRI

Brain Imaging Data Structure

- <https://bids.neuroimaging.io/>
- A second data standard

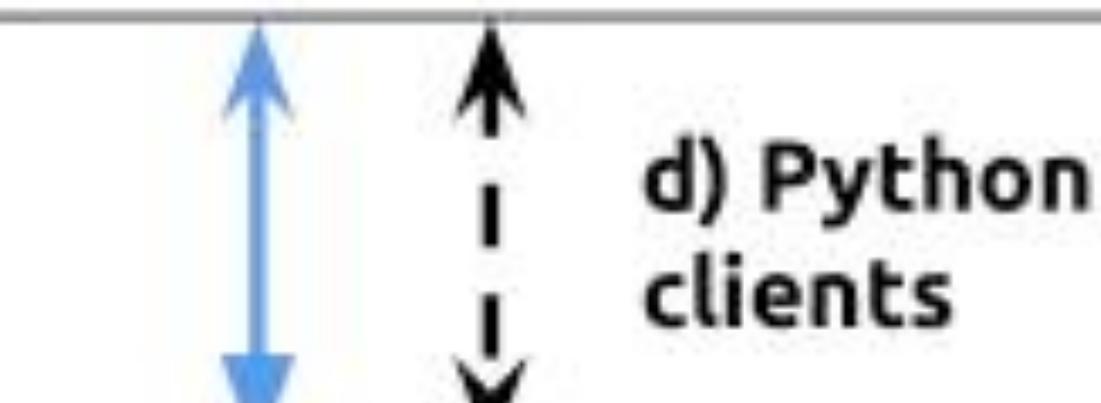
**Distributed Archives for
Neurophysiology Data Integration
(D.A.N.D.)**

What is DANDI?

- The BRAIN Initiative archive for publishing and sharing neurophysiology data including
 - Electrophysiology, Optophysiology, Behavioral time-series, Images from immunostaining experiments.
- A persistent, versioned, and growing collection of standardized datasets
- A place to house data to collaborate across research sites
- Supported by the BRAIN Initiative and the AWS Public dataset programs

a) Web application

The screenshot shows the homepage of The DANDI Archive. At the top, there is a navigation bar with links: DANDI (with a brain icon), WELCOME, PUBLIC DANDISETS, MY DANDISETS, ABOUT, DOCUMENTATION, HELP, NEW DANDISET, and a user profile icon. Below the navigation bar, the title "The DANDI Archive" is displayed in large blue text, followed by a subtitle: "The BRAIN Initiative archive for publishing and sharing neurophysiology data including electrophysiology, optophysiology, and behavioral time-series, and images from immunostaining experiments." A search bar with the placeholder "Search Dandisets by name, description, identifier or contributor name" is present. Below the search bar, three dark grey boxes provide summary statistics: "138 datasets", "311 users", and "157 TB total data size".



Collaborator(s)

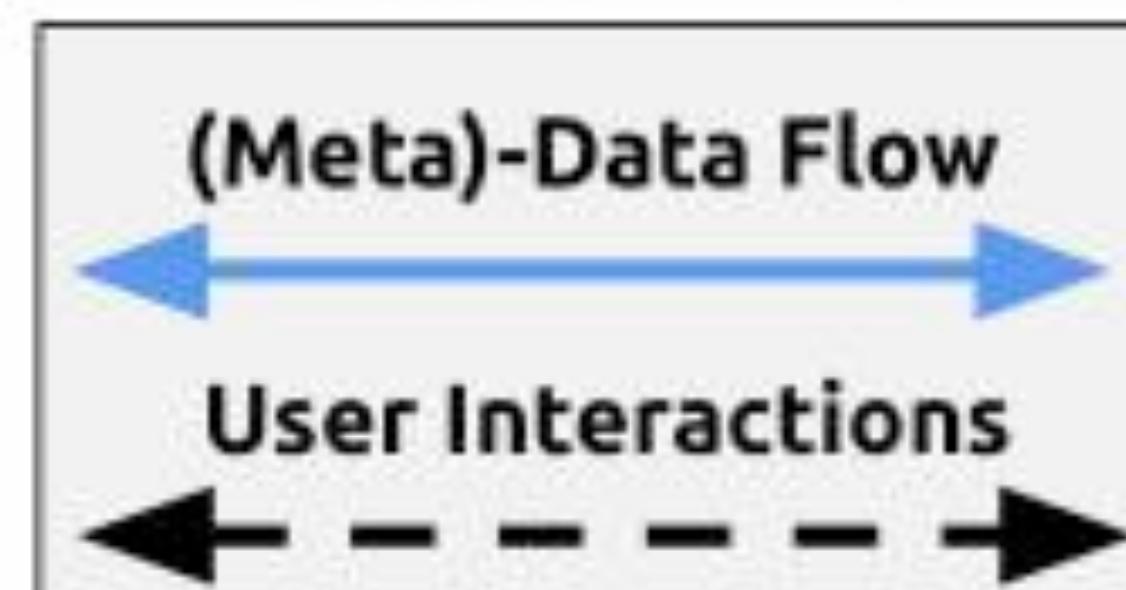


Lab Member(s)

b) Supported standards



c) Analysis platform



JSON/JSON-LD, NWB, NIfTI, TIFF

Web Browser, Shell, API

Benefits of DANDI

- A FAIR (Findable, Accessible, Interoperable, Reusable) data archive to house standardized neurophysiology and associated data
- Rich metadata to support search across data
- Consistent and transparent data standards to simplify data reuse and software development.
 - Uses NWB, BIDS, Neuroimaging Data Model (NIDM), and other BRAIN Initiative standards to organize and search the data.
 - The data can be accessed programmatically allowing for software to work directly with data in the cloud
- The infrastructure is built on a software stack of open source products, thus enriching the ecosystem

DANDI compatibility

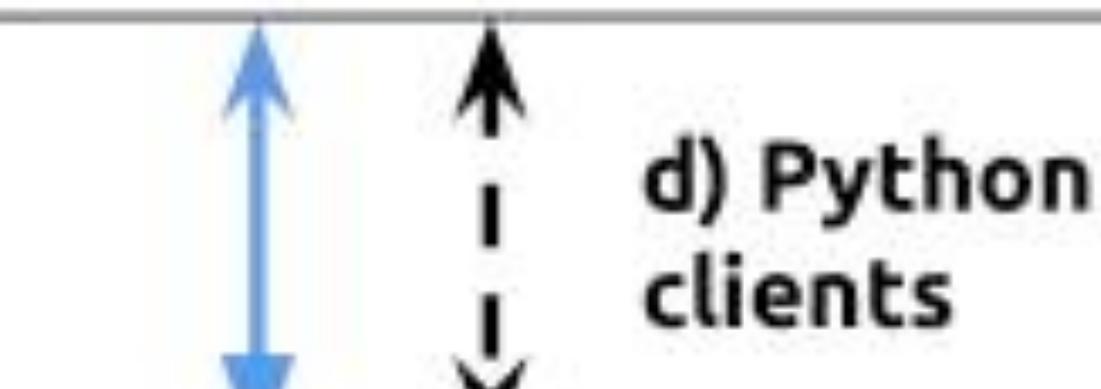
- Uses NWB for core data language
- “Dandisets” - DANDI datasets - collection of NWB files recorded over multiple sessions, organized together
- Viewable from a web browser
- Can interact through Jupyterhub interface for exploring, visualizing and analyzing the data stored in the archive

DANDI python client

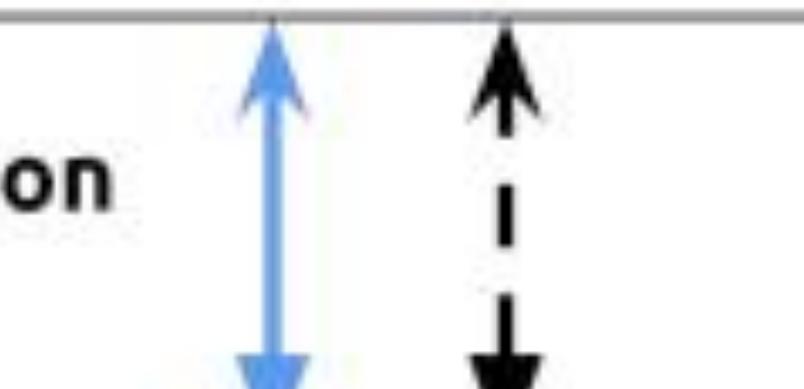
- Organize data locally into the required structure
- Download/upload data from/to the DANDI archive

a) Web application

The screenshot shows the homepage of The DANDI Archive. At the top, there is a navigation bar with links: DANDI (with a brain icon), WELCOME, PUBLIC DANDISETS, MY DANDISETS, ABOUT, DOCUMENTATION, HELP, NEW DANDISET, and a user profile icon. Below the navigation bar, the title "The DANDI Archive" is displayed in large blue text, followed by a subtitle: "The BRAIN Initiative archive for publishing and sharing neurophysiology data including electrophysiology, optophysiology, and behavioral time-series, and images from immunostaining experiments." A search bar is present with the placeholder "Search Dandisets by name, description, identifier or contributor name". Below the search bar, there are three dark grey boxes showing statistics: "138 datasets", "311 users", and "157 TB total data size".



Collaborator(s)



Lab Member(s)

b) Supported standards



c) Analysis platform



DANDI archive

- **Public DANDI sets:** <https://dandiarchive.org/dandiset>
- **Documentation:** https://www.dandiarchive.org/handbook/10_using_dandi/

DANDI Properties

- **Data identifiers:** The archive provides persistent identifiers for versioned datasets and assets, thus improving reproducibility of neurophysiology research
- **Data storage:** Cloud-based platform on AWS. Data are available from a public S3 bucket. Data from embargoed datasets are available from a private bucket to owners only
- **Type of data:** The archive accepts cellular neurophysiology data including electrophysiology, optophysiology, and behavioral time-series, and images from immunostaining experiments and other associated data (e.g. participant information, MRI or other modalities)
- **Accepted Standards** and Data File Formats: NWB (HDF5), BIDS (NIfTI, JSON, PNG, TIF, OME.TIF, OME.BTF, OME.ZARR) (see Data Standards for more details)

Neurophysiology Informatics Challenges and DANDI Solutions

Challenges	Solutions
Most raw data stays in laboratories.	DANDI provides a public archive for dissemination of raw and derived data.
Non-standardized datasets lead to significant resource needs to understand and adapt code to these datasets.	DANDI standardizes all data using NWB and BIDS standards.
The multitude of different hardware platforms and custom binary formats requires significant effort to consolidate into reusable datasets.	The DANDI ecosystem provides tools for converting data from different instruments into NWB and BIDS.
There are many domain general places to house data (e.g. Open Science Framework, G-Node, Dropbox, Google drive), but it is difficult to find relevant scientific metadata.	DANDI is focused on neurophysiology data and related metadata.
Datasets are growing larger, requiring compute services to be closer to data.	DANDI provides Dandihub, a JupyterHub instance close to the data.
Neurotechnology is evolving and requires changes to metadata and data storage.	DANDI works with community members to improve data standards and formats.
Consolidating and creating robust algorithms (e.g. spike sorting) requires varied data sources.	DANDI provides access to many different datasets.

DANDI archive

- <https://elifesciences.org/articles/78362>
- **Oliver Rübel, Andrew Tritt, Ryan Ly, Benjamin K Dichter, Satrajit Ghosh, Lawrence Niu, Pamela Baker, Ivan Soltesz, Lydia Ng, Karel Svoboda, Loren Frank, Kristofer E Bouchard (2022) The Neurodata Without Borders ecosystem for neurophysiological data science eLife 11:e78362**
- <https://doi.org/10.7554/eLife.78362>

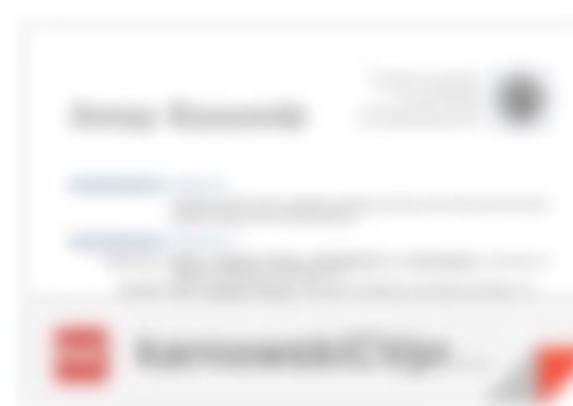
This sucks

 main_simple_bak9-pretty-good.c	Aug 1, 2008, 1:01 AM	33 KB	C Source
 main_simple_bak9-pretty-good.o	Aug 1, 2008, 1:00 AM	303 KB	object code
 main_simple_bak9-pretty-goodv2.c	Aug 2, 2008, 1:16 AM	33 KB	C Source
 main_simple_bak10.c	Sep 28, 2008, 1:16 PM	33 KB	C Source
 main_simple_bak11-workingUART_correctspeed.c	Aug 30, 2008, 2:49 AM	27 KB	C Source
 main_simple_bak11-workingUART_correctspeed.o	Aug 2, 2008, 1:17 AM	303 KB	object code
 main_simple_bak12_willspin.c	Aug 2, 2008, 1:30 AM	28 KB	C Source
 main_simple_bak12_willspin.o	Aug 2, 2008, 2:35 AM	301 KB	object code
 main_simple_bak13-worksA-D-nonoise-spins.c	Aug 7, 2008, 12:57 PM	26 KB	C Source
 main_simple_bak14-widersinefunctionsworkingrotation.c	Aug 8, 2008, 5:02 PM	26 KB	C Source
 main_simple_bak15-spins-stillneedsquadrantfixed.c	Aug 15, 2008, 7:32 PM	30 KB	C Source
 main_simple_bak16-15backup-spins-needs-improvement.c	Oct 15, 2008, 8:54 PM	31 KB	C Source
 main_simple_bak17-smoother-stillnostandingstart.c	Aug 16, 2008, 6:50 PM	30 KB	C Source
 main_simple_bak17-smoother-stillnostandingstart.o	Aug 18, 2008, 9:41 PM	305 KB	object code
 main_simple_bak18-notgood.c	Aug 18, 2008, 9:42 PM	31 KB	C Source
 main_simple_bak20SIMPLE-DCnotbrushless.c	Sep 17, 2009, 11:02 PM	27 KB	C Source
 main_simple_bak20WORKS_PWM_COMMAND_CONTROL.c	Aug 19, 2008, 12:54 AM	29 KB	C Source
 main_simple_timer_intrpt_bak.c	Aug 12, 2008, 12:16 AM	13 KB	C Source
 main_simple_timer_intrpt_bak2.c	Aug 12, 2008, 2:00 PM	13 KB	C Source
 main_simple_timer_intrpt_bak3.c	Aug 18, 2008, 12:14 AM	13 KB	C Source
 main_simple_timer_intrpt.c	Aug 18, 2008, 12:17 AM	13 KB	C Source
 main_simple_workingHWPWM.c	Aug 18, 2008, 7:19 PM	15 KB	C Source
 main_simple.c	Sep 17, 2009, 11:02 PM	29 KB	C Source

Thanks for chatting with me earlier today. I added the link to the visualization project into my resume and attached the resume. Thanks for any connections you can make for me. I'd love to know where you send it, so I can keep track of that. Thanks again!

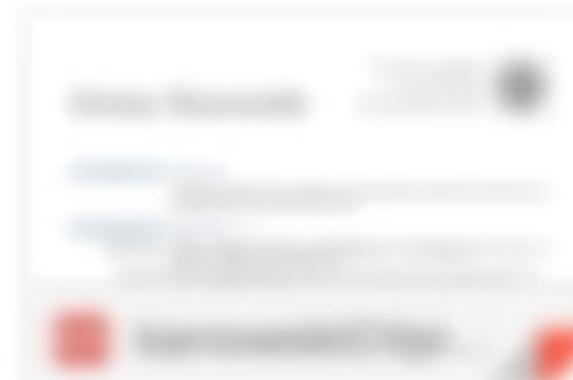
Best,

Yup, this sucks too.



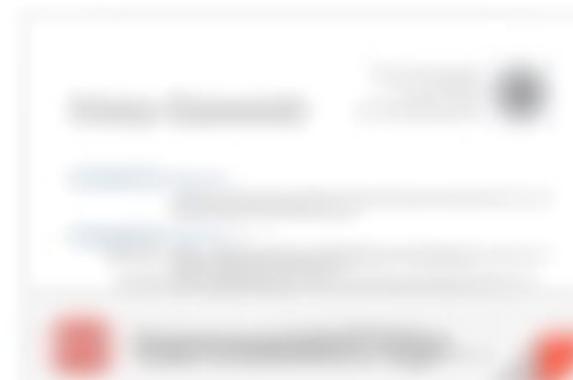
✉ May 11 ★ ⌂ ⌂

Actually, please use this one. I fixed a typo that was previously missed. Thanks!



✉ May 11 ★ ⌂ ⌂

Final copy, I swear. Thanks for helping out.



✉ May 11 ★ ⌂ ⌂

adapted from Brad Voytek

This is a step in the right direction

Total: 9 edits | ^ v Only show named versions

SDSS Teacher Workshop

Considering how to incorporate data science into your high school STEM classroom?

~~The goal of this workshop is for you to leave with data science skills and applicable examples that can be used in your classroom.~~

The goal of this workshop is for you to leave with data science skills and applicable examples that can be used in your classroom.

This workshop will answer questions like—

- ~~What is data science?~~
- ~~How can high schoolers prepare for data science courses in college?~~
- ~~What does a career in data science involve? Dd~~

~~discuss answer questions like:~~

- ~~What is data science?~~
- ~~How can high schoolers prepare for data science courses in college?~~
- ~~What does a career in data science involve? what data science is, what high schoolers can do to best prepare for data science courses in college, and what a career in data science involves.~~
-

We will walk through how data scientists carry out projects using RStudio, introduce the basics of the R programming language, and work with real datasets to generate visualizations and analyze data. ~~The goal of this workshop is for you to leave with data science skills and applicable examples that can be used in your classroom.~~

Version history

MARCH

▶ March 4, 7:27 AM Current version ● Shannon Ellis

▶ March 3, 9:47 AM ● Donna LaLonde ● Shannon Ellis

FEBRUARY

▶ February 27, 6:29 AM ● Shannon Ellis

February 26, 5:44 PM ● Shannon Ellis

▶ February 26, 4:57 PM ● Shannon Ellis

▶ February 26, 3:50 PM ● Kelly McConville

▶ February 25, 3:53 PM ● Shannon Ellis

February 25, 3:33 PM ● Shannon Ellis

Show changes

Version Control

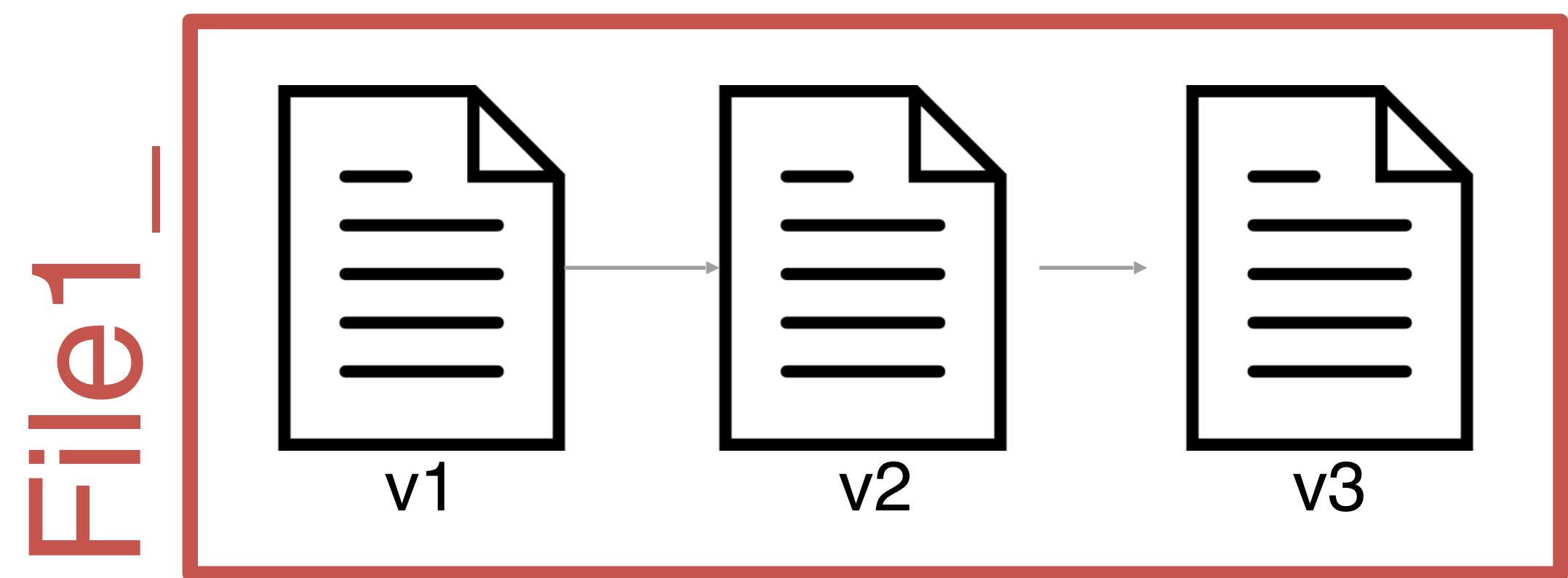
- Enables multiple people to simultaneously work on a single project.
- Each person edits their own copy of the files and chooses when to share those changes with the rest of the team.
- Thus, temporary or partial edits by one person do not interfere with another person's work

What is version control?

A way to manage the evolution of a set of files

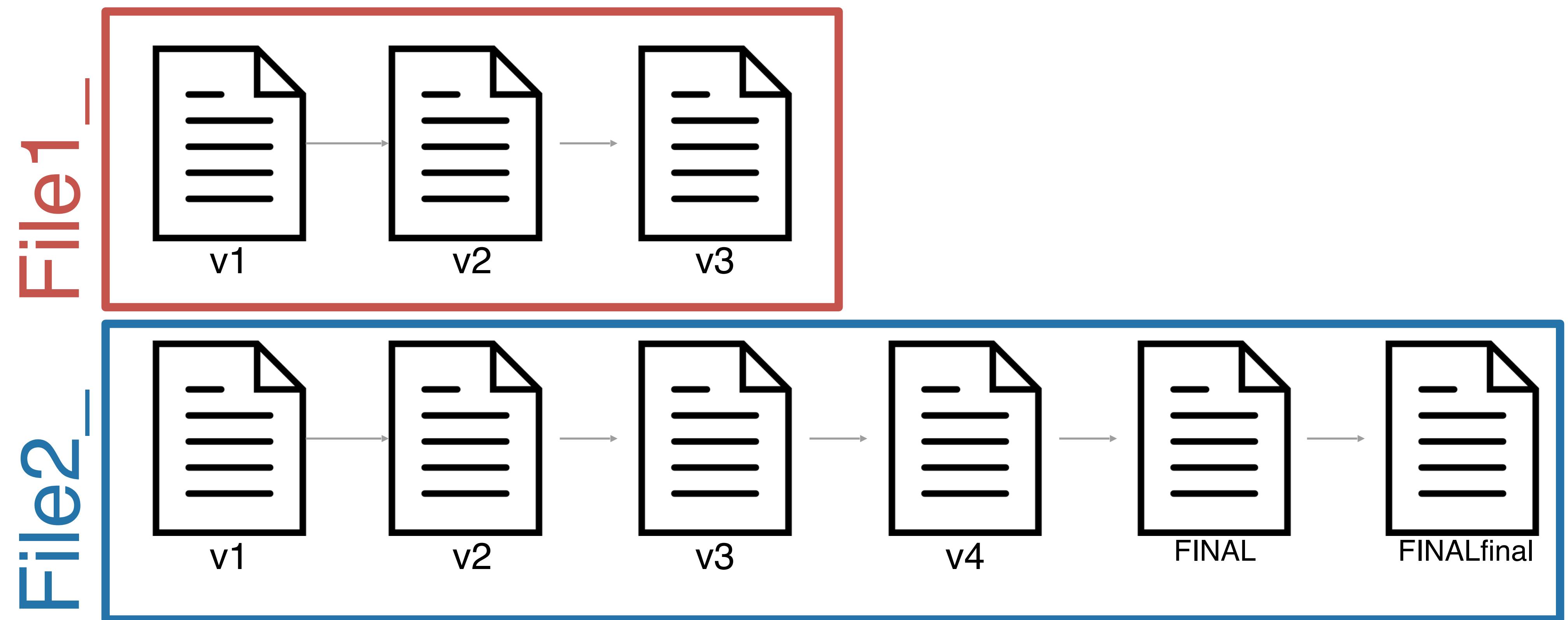
What is version control?

A way to manage the evolution of a set of files



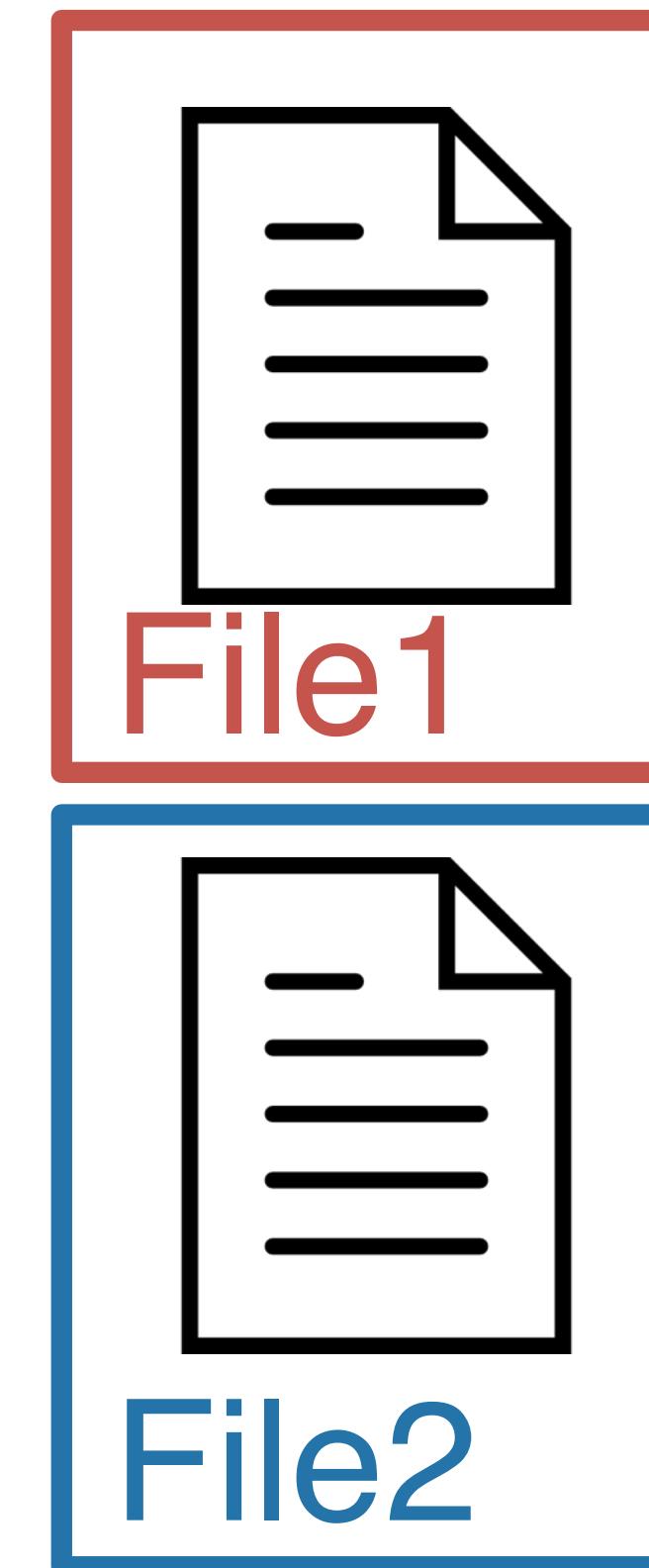
What is version control?

A way to manage the evolution of a set of files



What is version control?

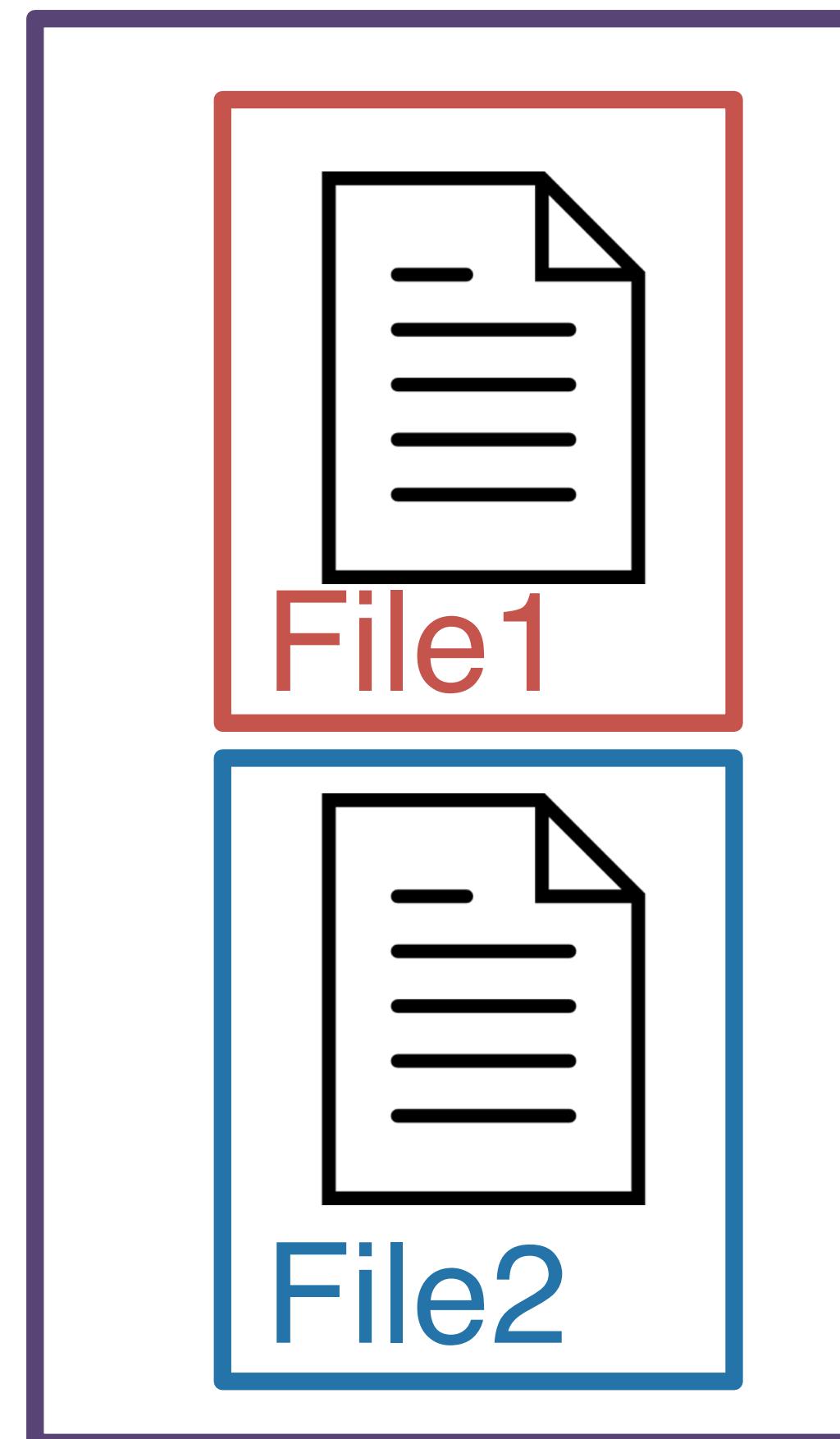
A way to manage the evolution of a set of files



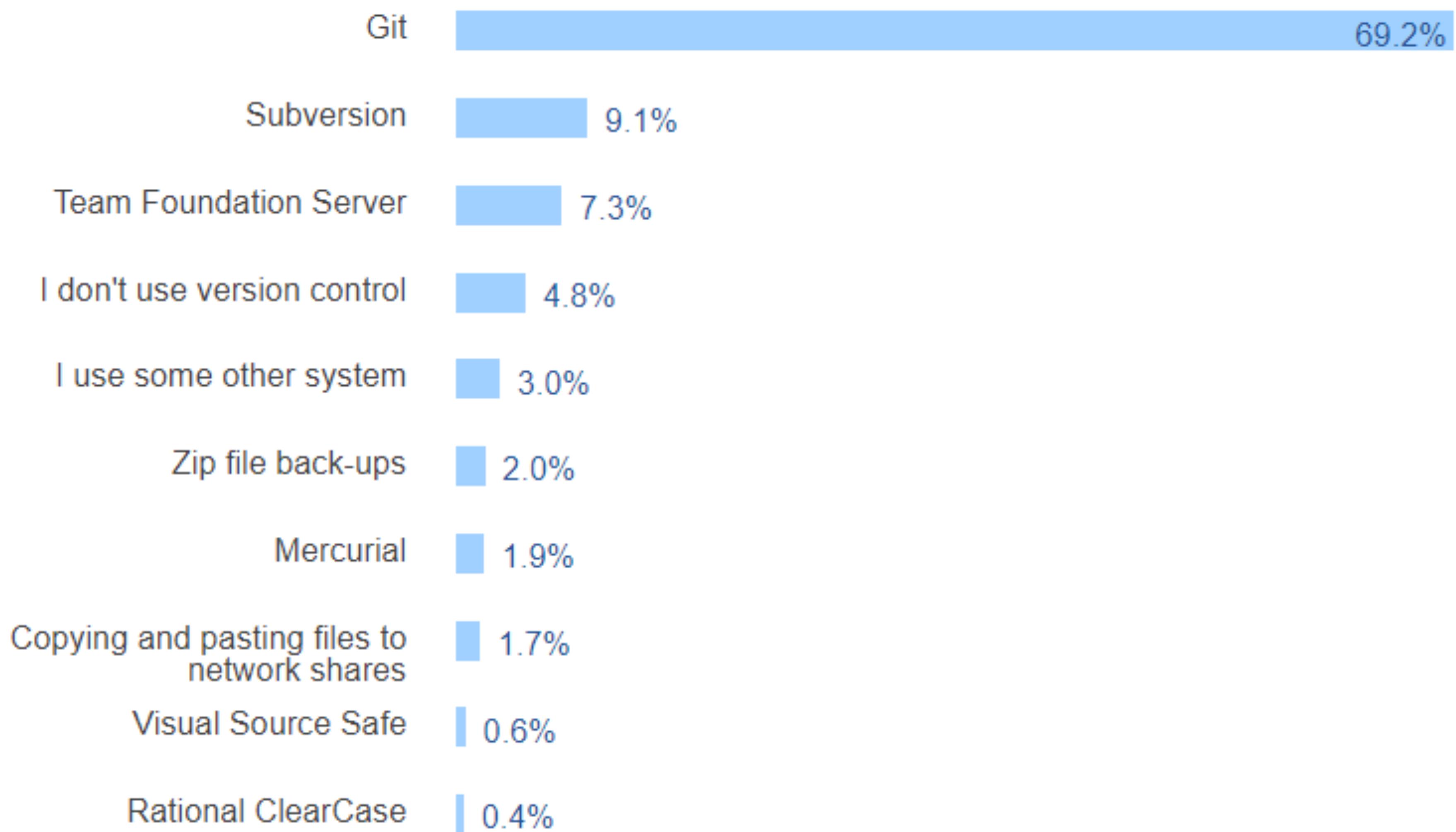
When using a version control system,
you have **one copy of each file** and
the *version control system tracks the
changes* that have occurred over time

What is version control?

A way to manage the evolution of a set of files



The set of files is referred to as a **repository** (**repo**)

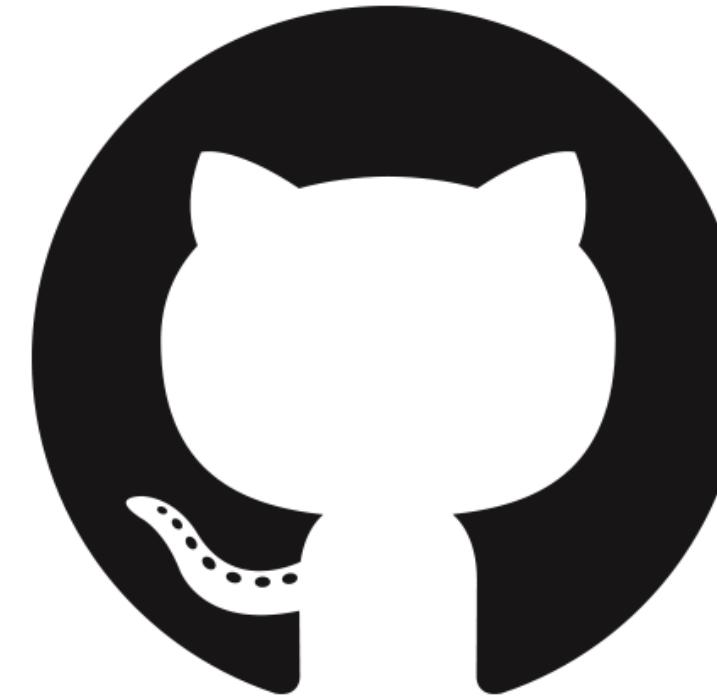


git & GitHub

git

the version control system

~ Track Changes
from Microsoft
Word....on
steroids



GitHub (or Bitbucket or
GitLab) is the home **where**
your git-based projects live

on the Internet.

~ Dropbox....but
way better

What version control looks like

```
$ git clone https://www.github.com/username/repo.git  
$ git pull  
$ git add -A  
$ git commit -m "informative commit message"  
$ git push
```

Terminal
git



COGS108 - Data Science in Practice
Course materials for Hands-On Data Science.
UC San Diego COGS108@gmail.com

Repositories 22 People 7 Teams 2 Projects 0 Settings

Pinned repositories

- Overview Overview and map of the organization, which services COGS108: Hands-On Data Science, from UCSD. ★ 38 89
- Lectures-Sp19 Slides and Notebooks used in Lecture for Sp19 COGS108 ★ 1 81
- Section_Workbooks Workbooks for practice during discussion section Jupyter Notebook 81
- Tutorials Tutorial notebooks for hands-on data science, following along with the course topics. Jupyter Notebook ★ 38 108
- Projects Final Project materials and description. Jupyter Notebook ★ 3 82
- Readings A curated list of suggested reading materials. ★ 4 81

Find a repository... Type: All Language: All New

MyFirstPullRequest To be used for the assignments in Cogs 108 ★ 1 87 Updated 7 minutes ago

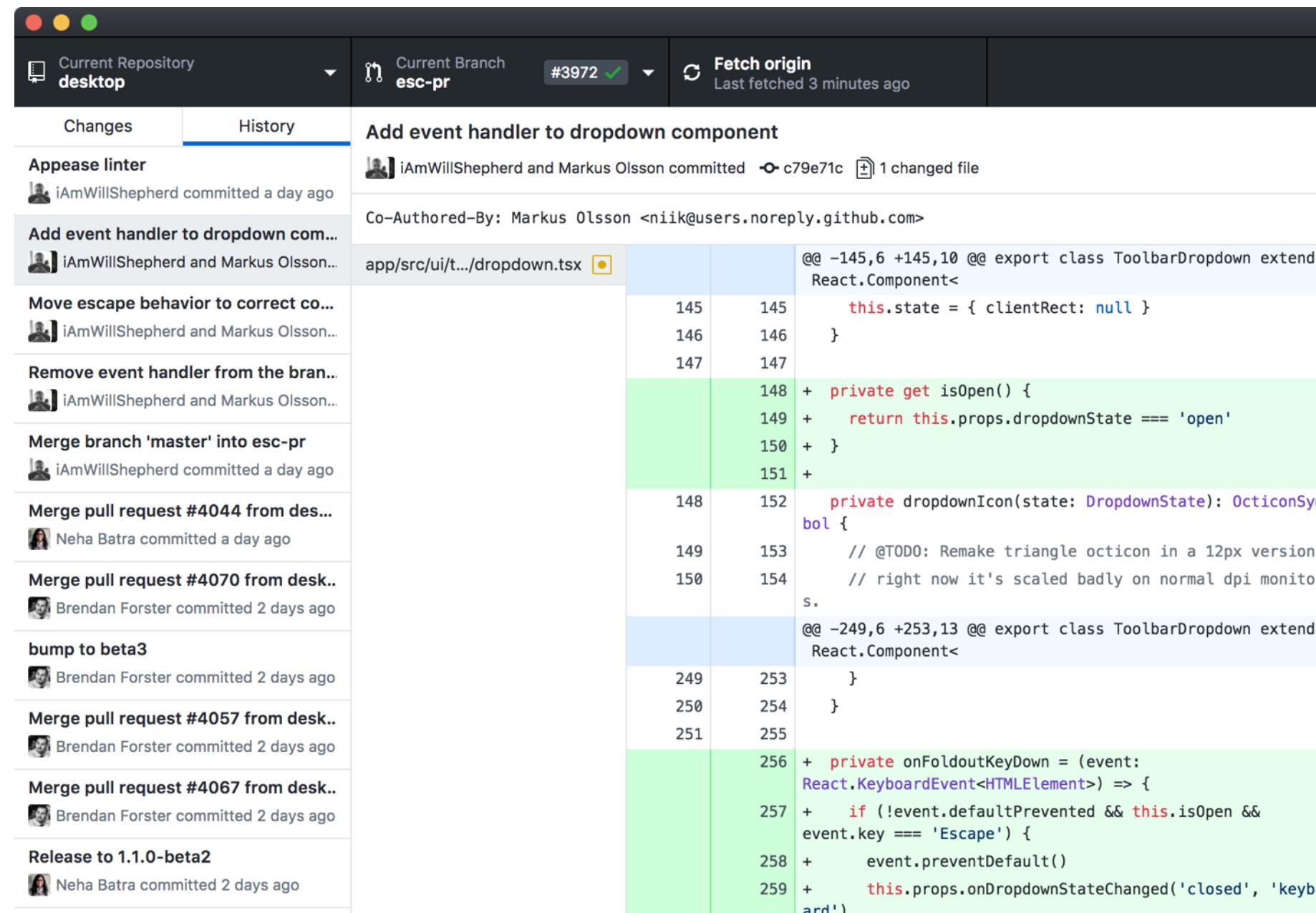
Top languages Jupyter Notebook Python

Overview Overview and map of the organization, which services COGS108: Hands-On Data Science, from UCSD. Manage data-science python tutorial material class

Most used topics Manage data-science python tutorial

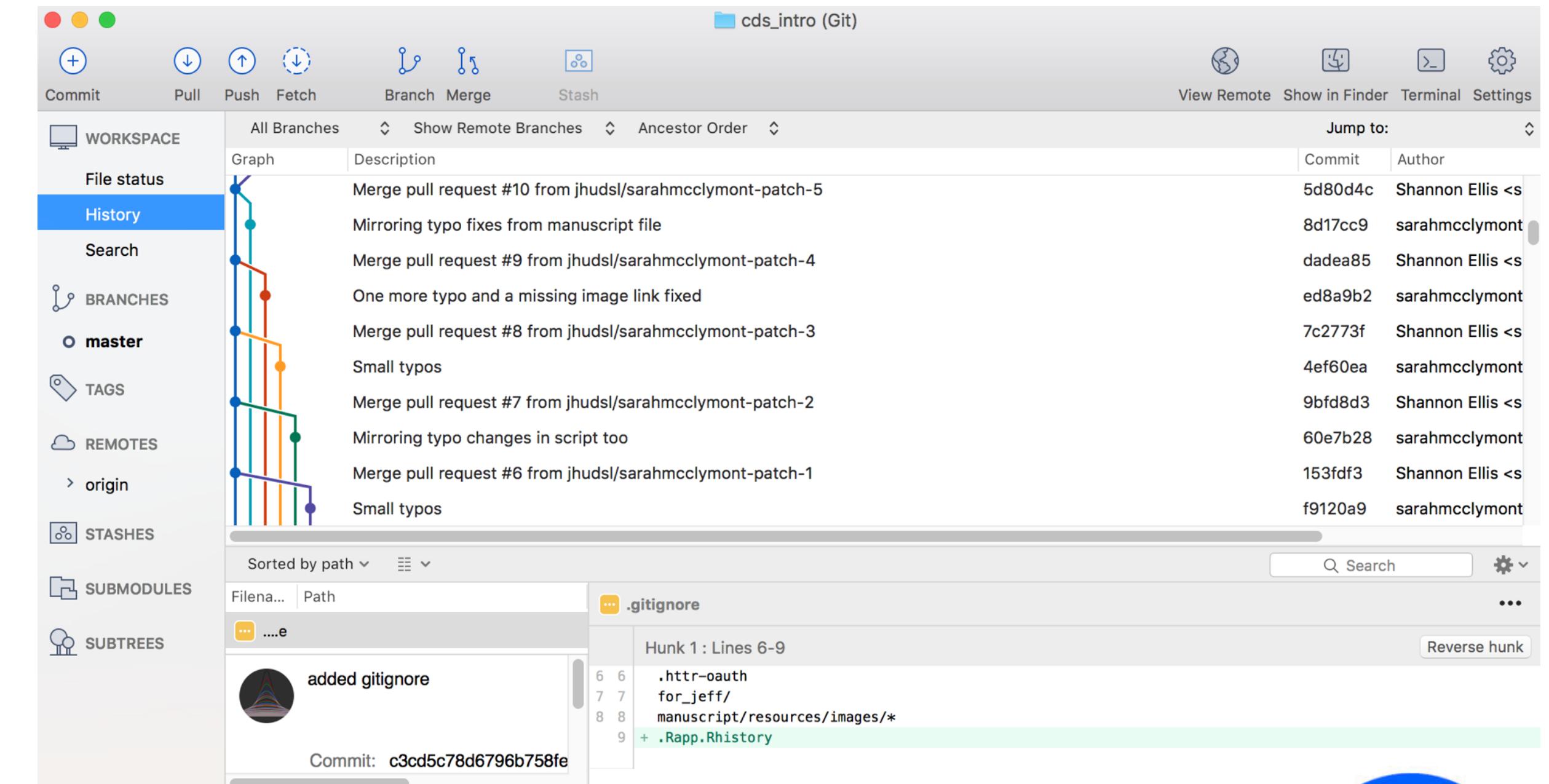


GUIs can be helpful when working with version control



The screenshot shows the GitHub Desktop application interface. At the top, it displays the current repository ("desktop") and branch ("esc-pr #3972"). A "Fetch origin" button is visible. The main area is titled "Add event handler to dropdown component" and shows a commit from "iAmWillShepherd and Markus Olsson" (c79e71c) that adds a private get isOpen() method. The code editor shows the implementation of this method. Below the commit, there's a list of other recent commits, including merges from pull requests #4044, #4057, and #4067.

GitHub Desktop



The screenshot shows the SourceTree application interface. It displays a git history graph for the repository "cds_intro (Git)". The graph shows multiple branches merging into the "master" branch. The right pane lists the commit history, starting with "Merge pull request #10 from jhudsl/sarahmcclmont-patch-5" and continuing through several merge pull requests and small typo fixes. The bottom pane shows a file diff for ".gitignore", which includes entries for ".httr-oauth", "for_jeff/", "manuscript/resources/images/*", and ".Rapp.Rhistory".

SourceTree

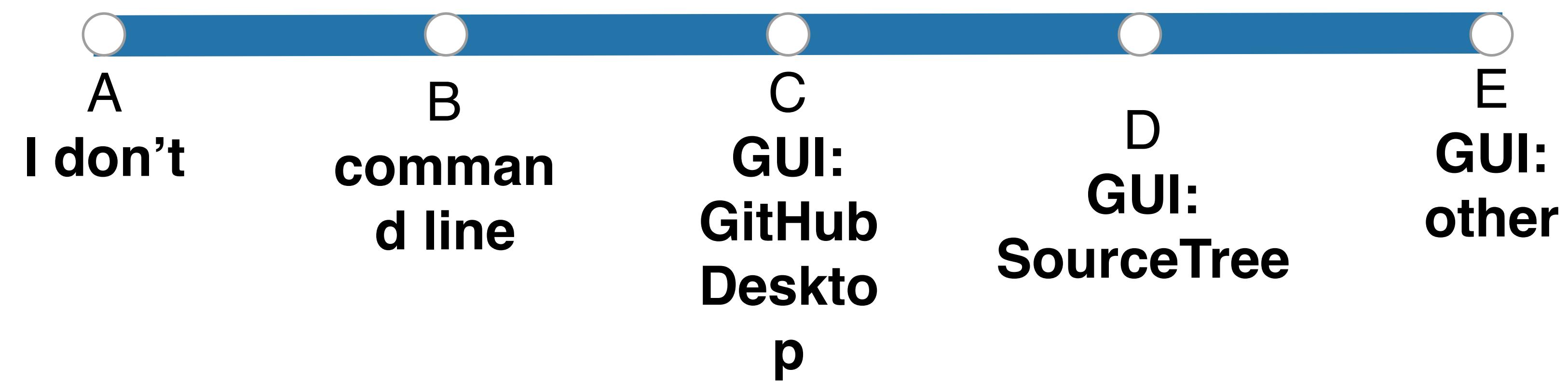




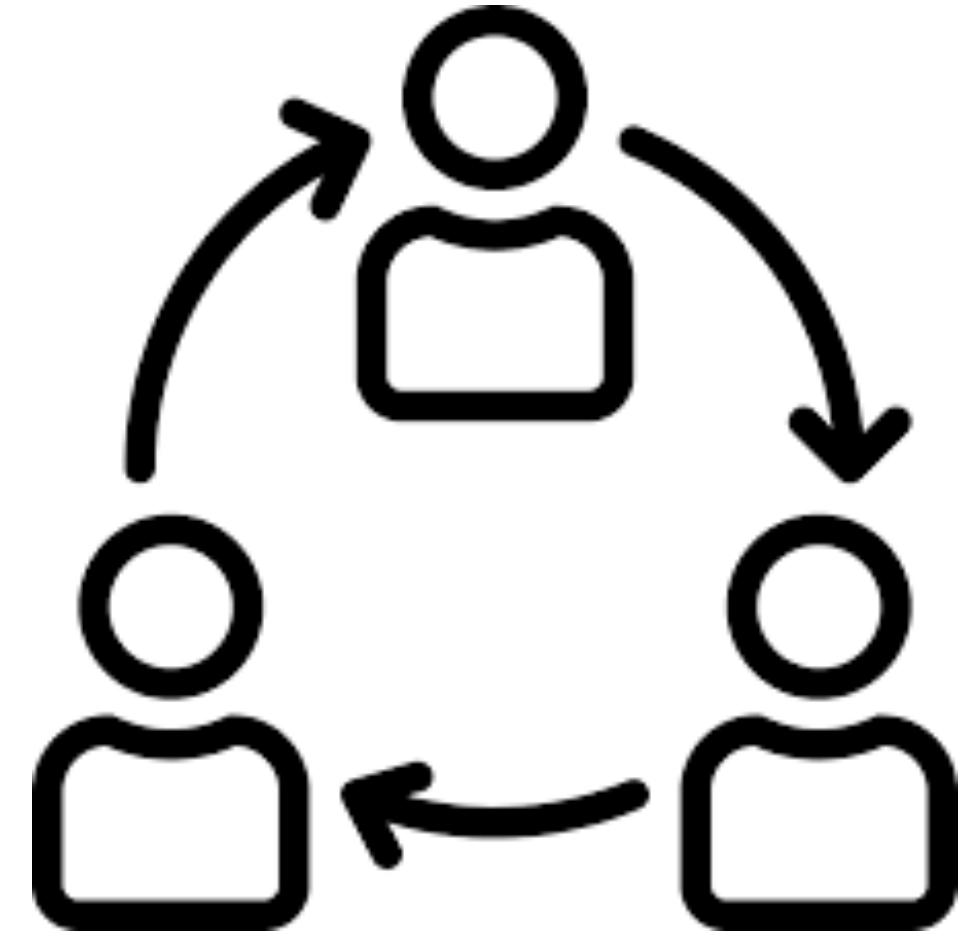
Version Controller

a

How do you typically interact with git?



Why version control with git and GitHub?



Collaboration



Returning to
a safe state

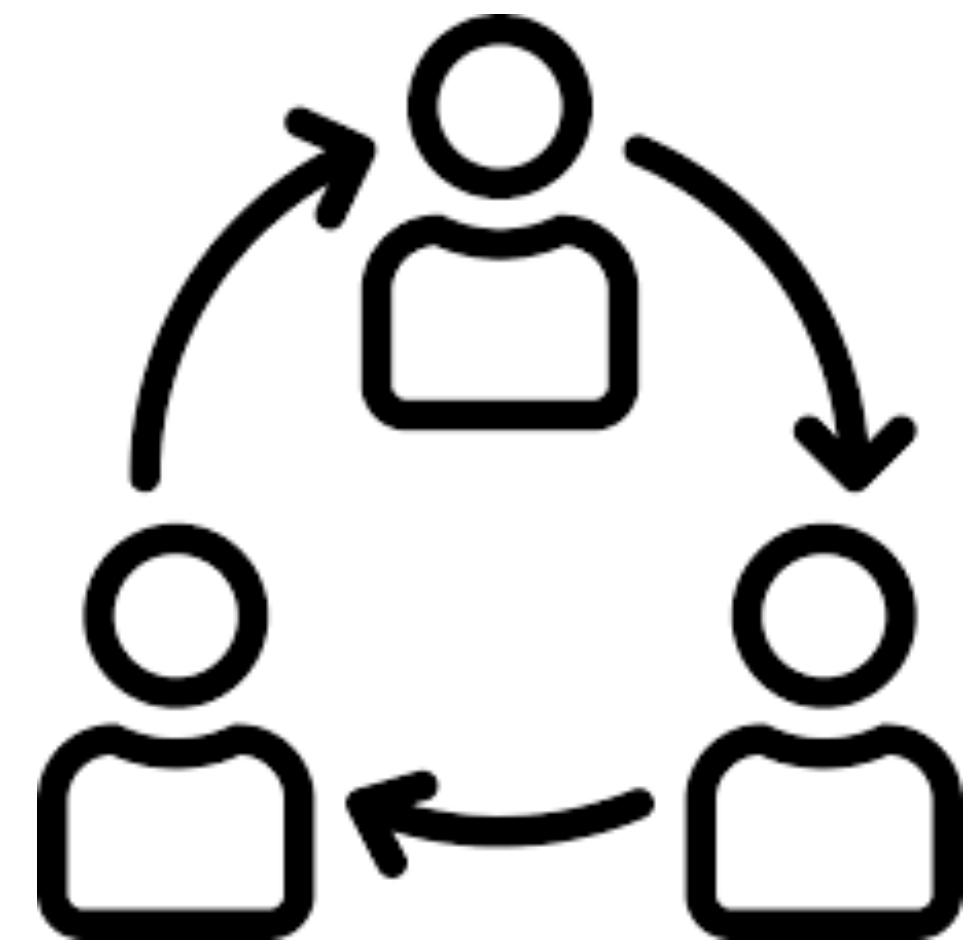


Exposure
for your
work

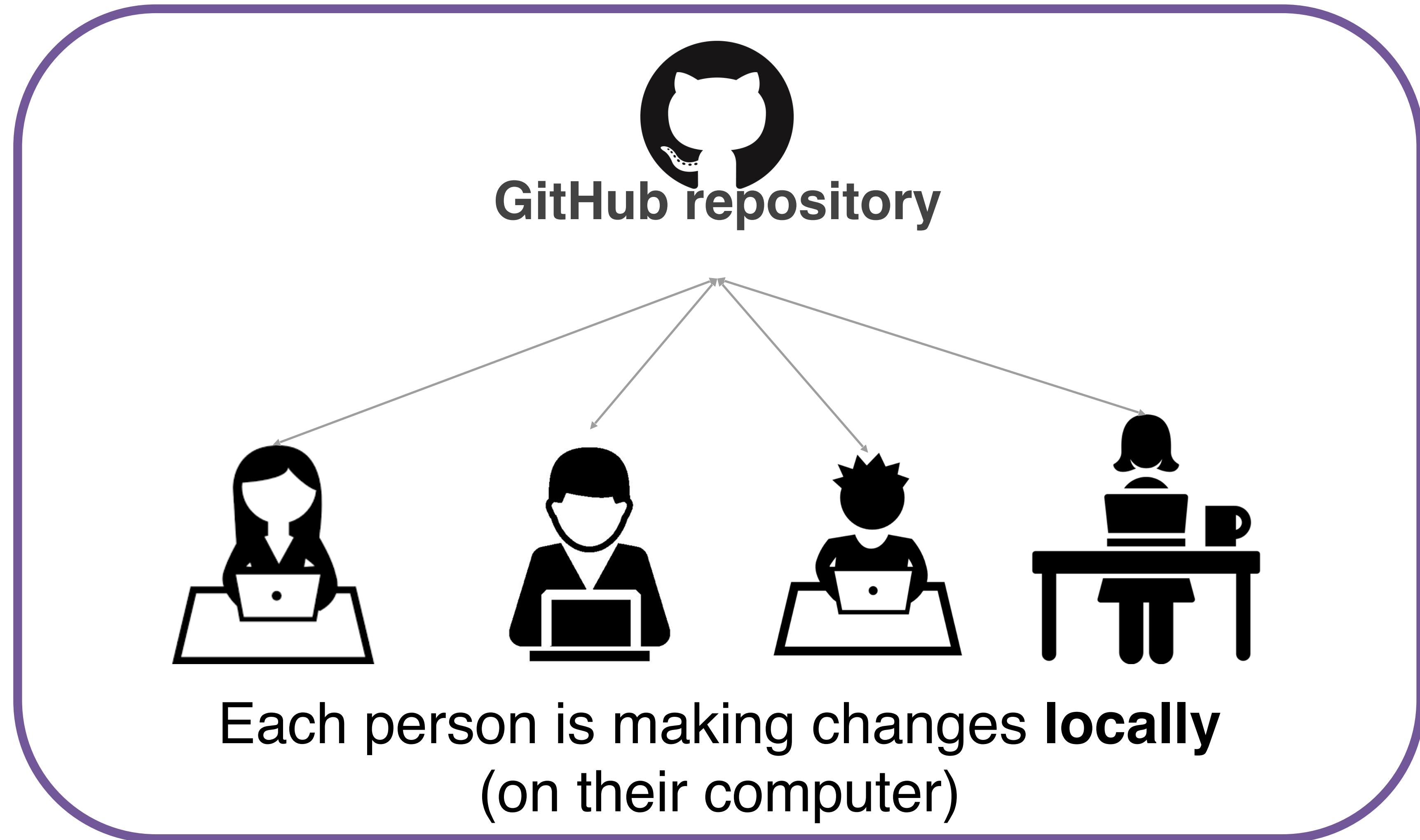


Tracking
others' work

Collaborate like you do with Google Docs



Collaboration



Each person is making changes **locally**
(on their computer)

Make changes locally, while knowing a stable copy exists



Returning to
a safe state



Your repositories will be visible to others!



Exposure
for your
work

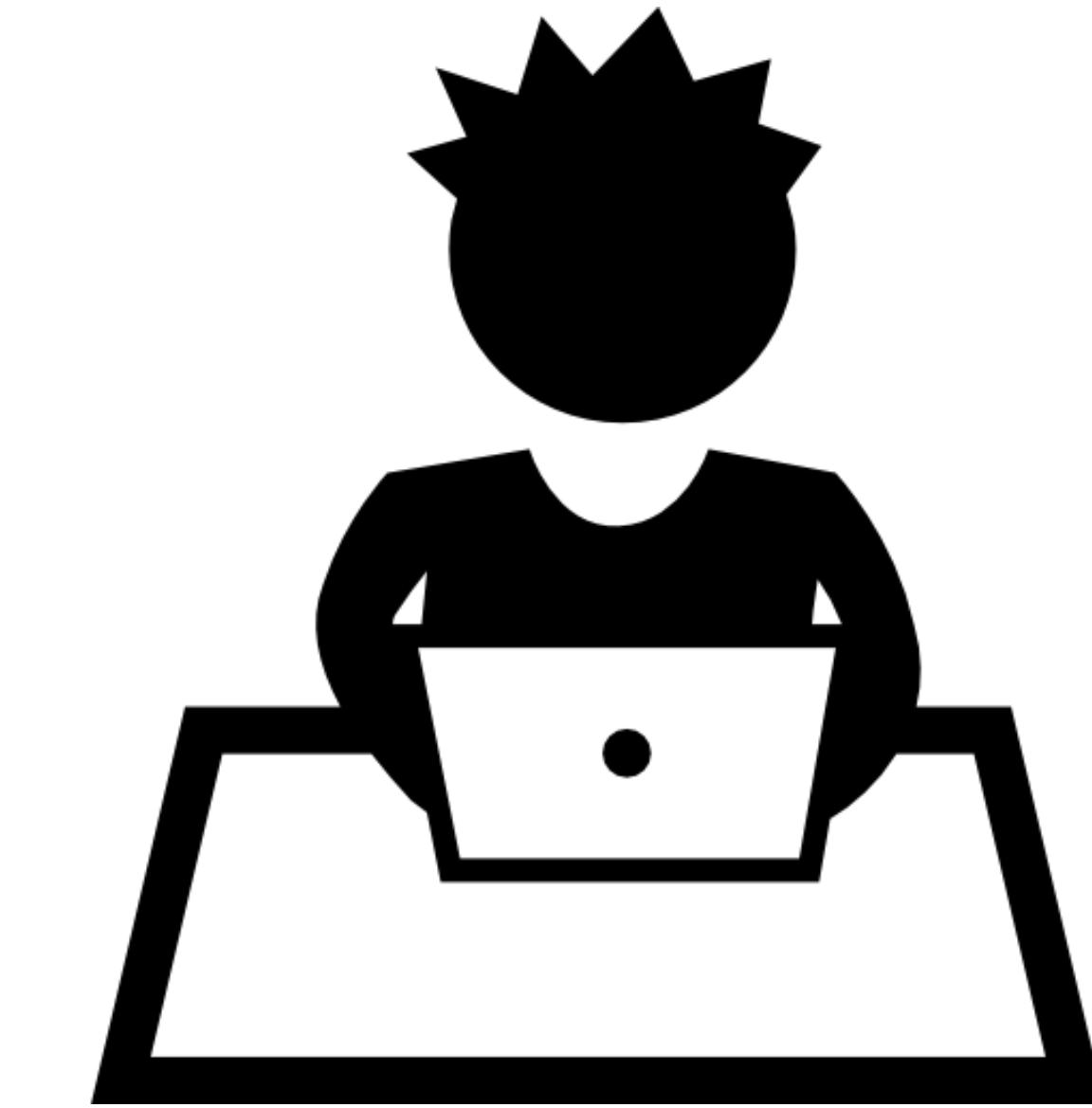


Your public GitHub repos
are your coding social
media

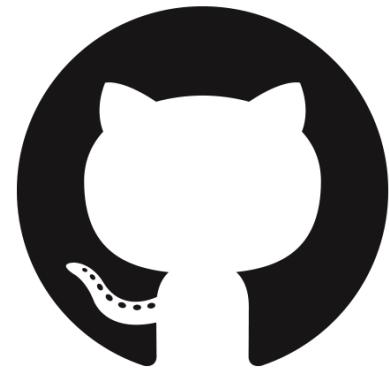
Keep up with others' work easily



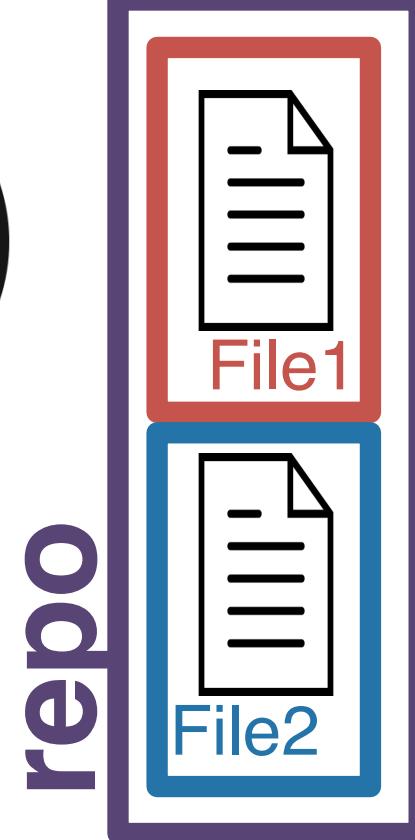
Tracking
others' work



As a social platform, you
can see others' work too!

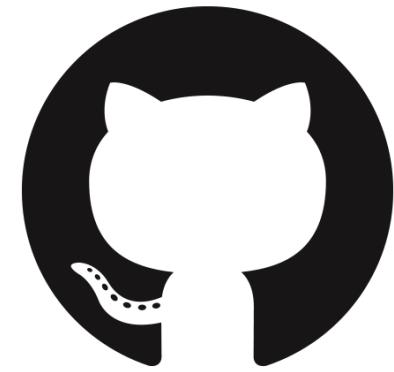


repo

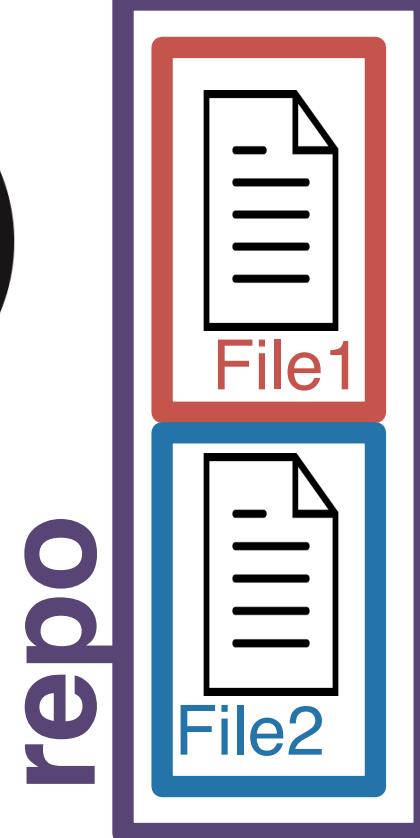


A **GitHub repo** contains all the files and folders for your project.

GitHub is a **remote host**. The files are geographically distant from any files on your computer.



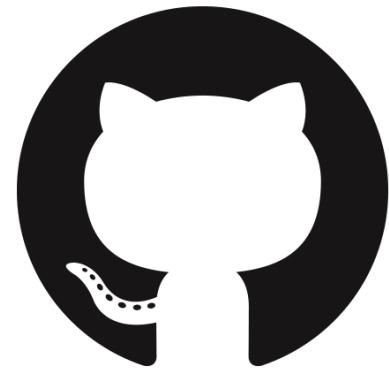
repo



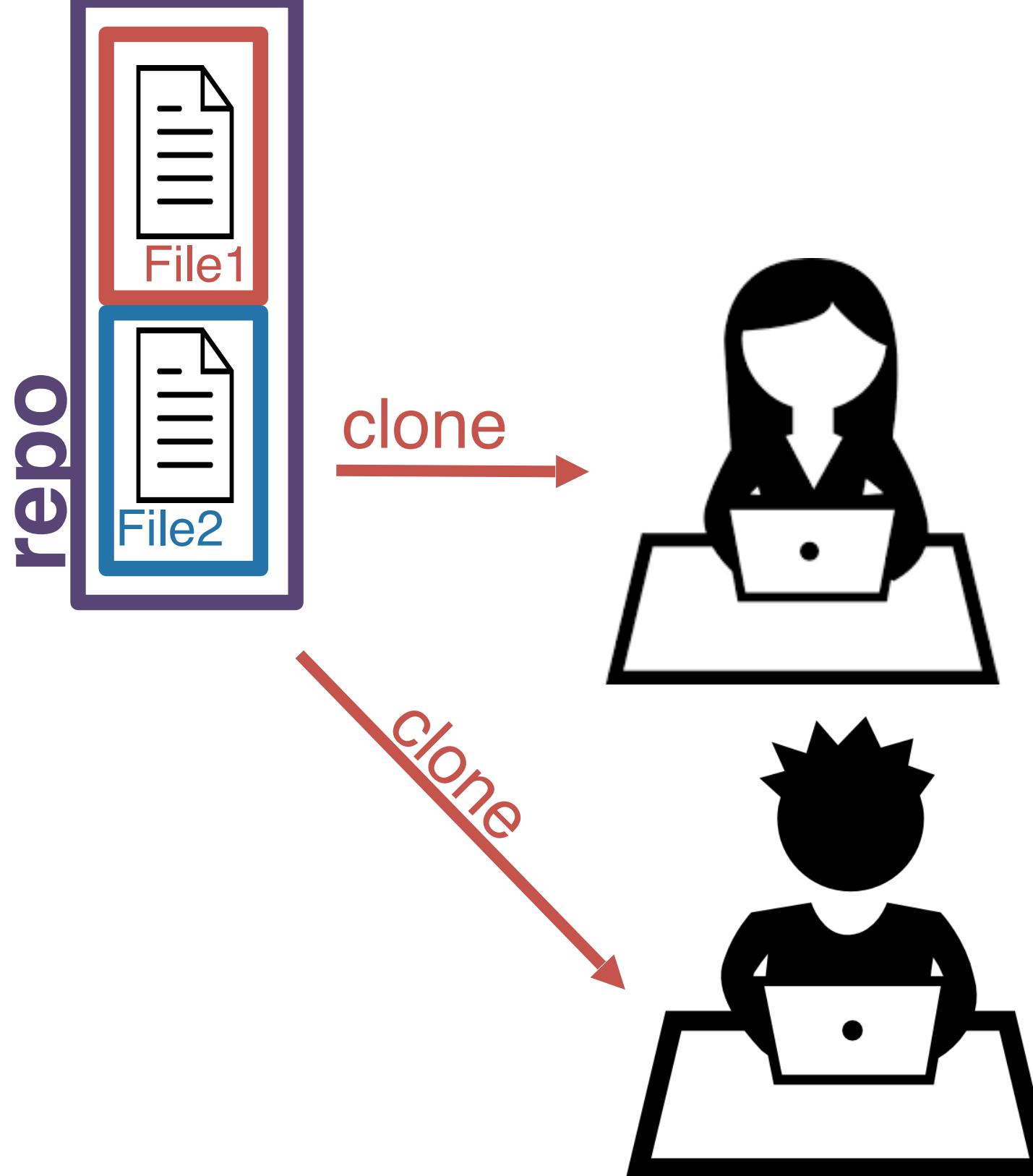
clone



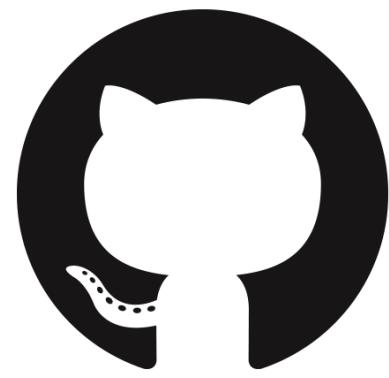
When you first make a copy onto your local computer (read: laptop), you **clone** the repository.



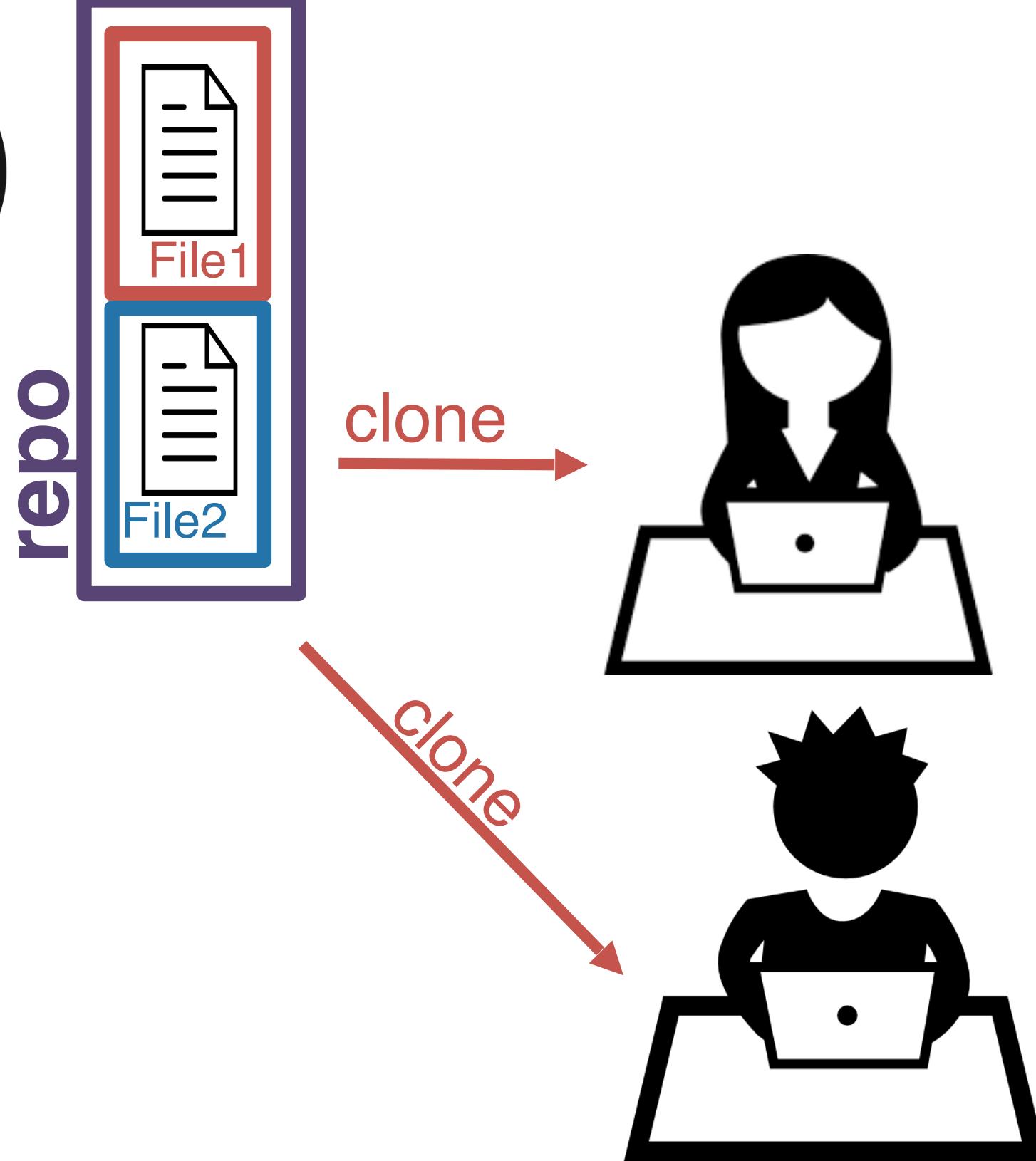
repo



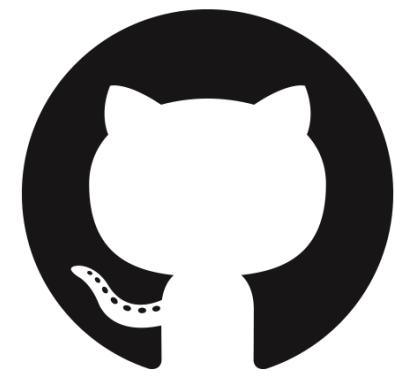
If someone else on your project cloned the repo at the same time, you would have identical copies of the project on each of your computers.



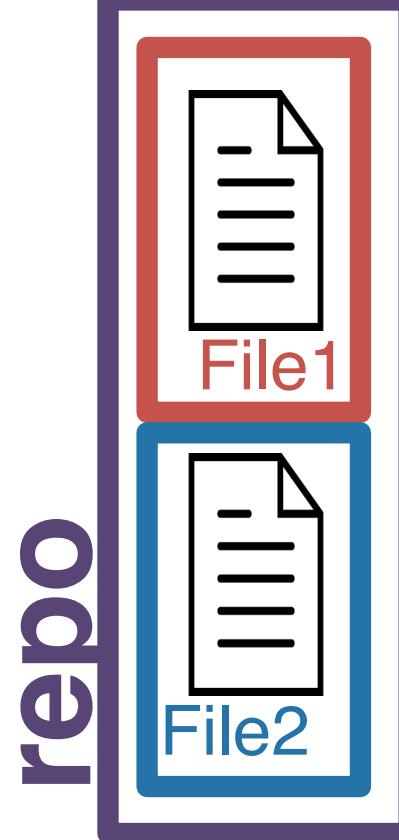
repo



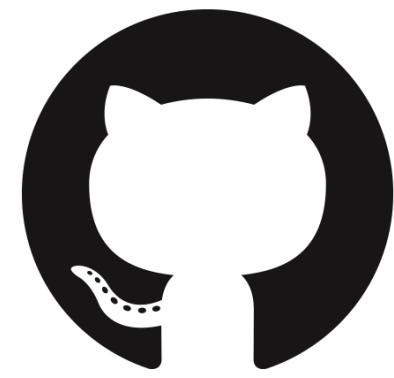
Yay! Everyone can
work on the project!



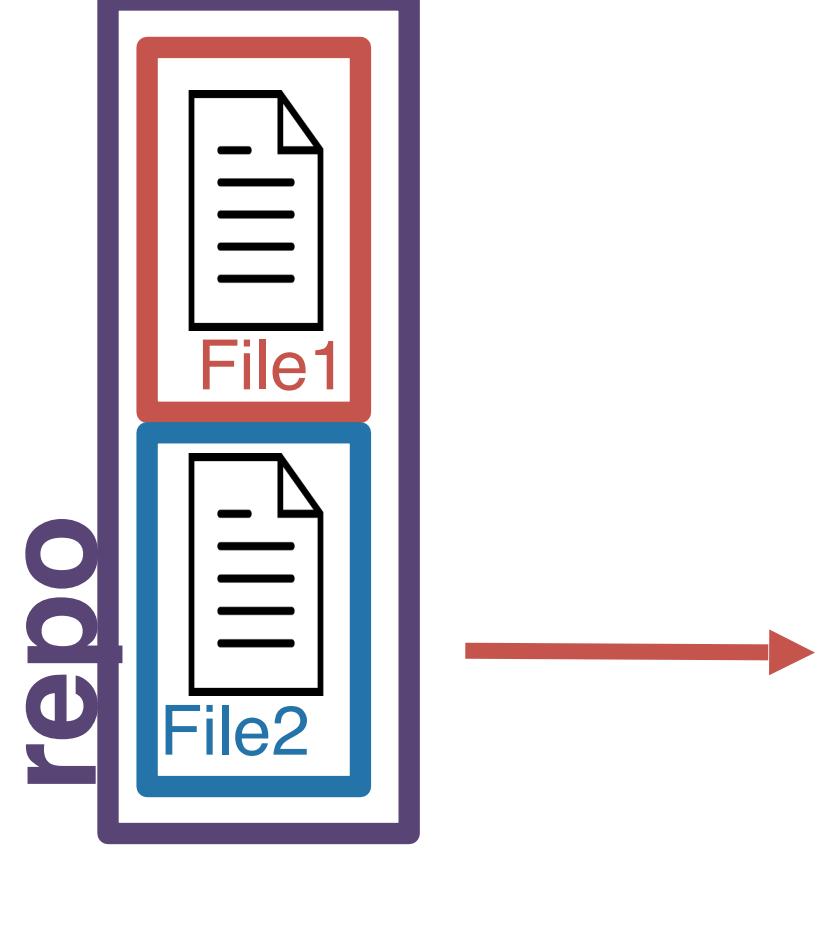
repo



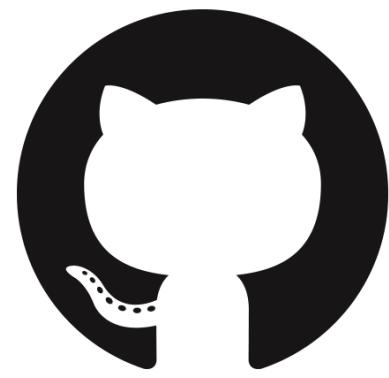
You decide you want to
change some of the text
in the project.



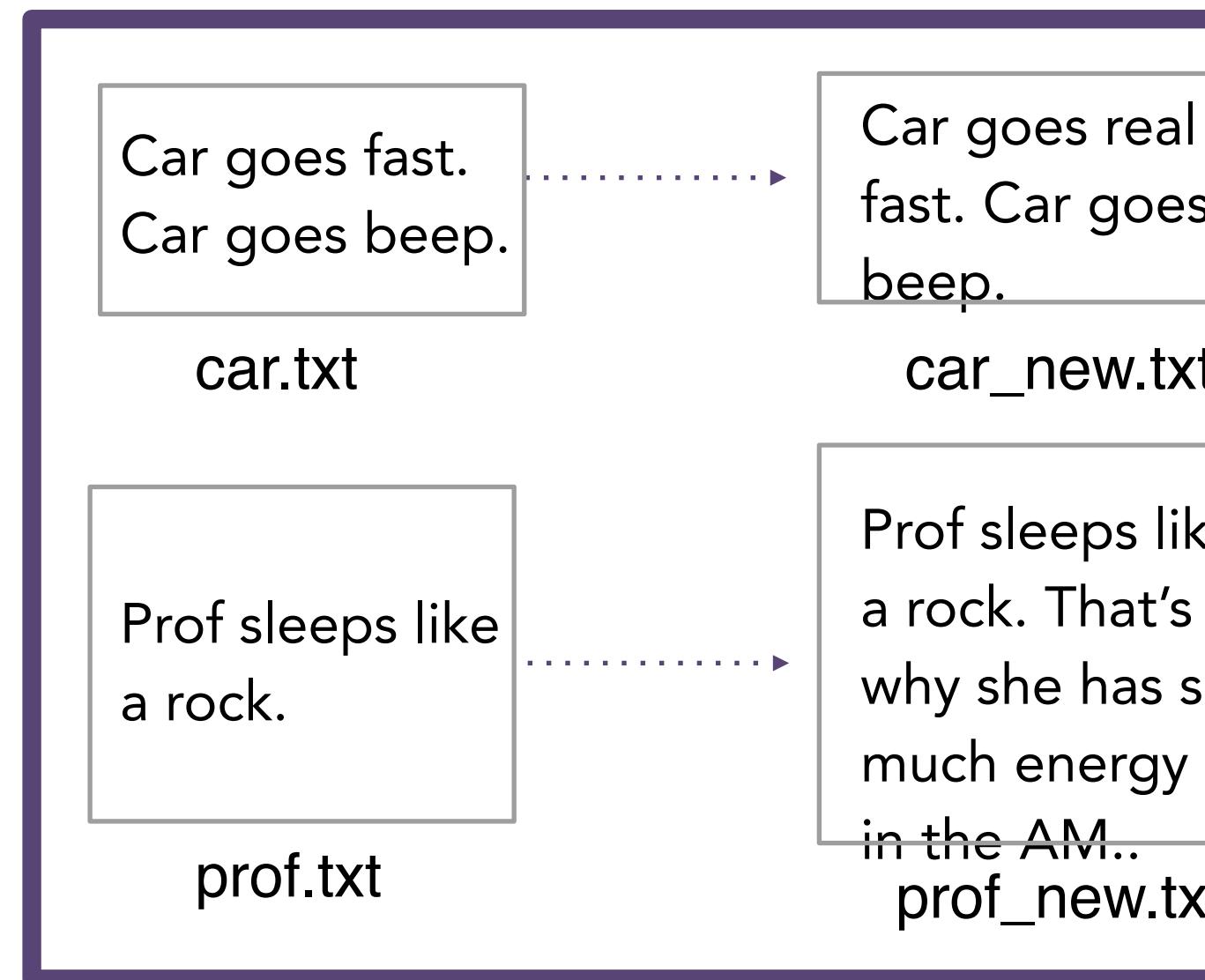
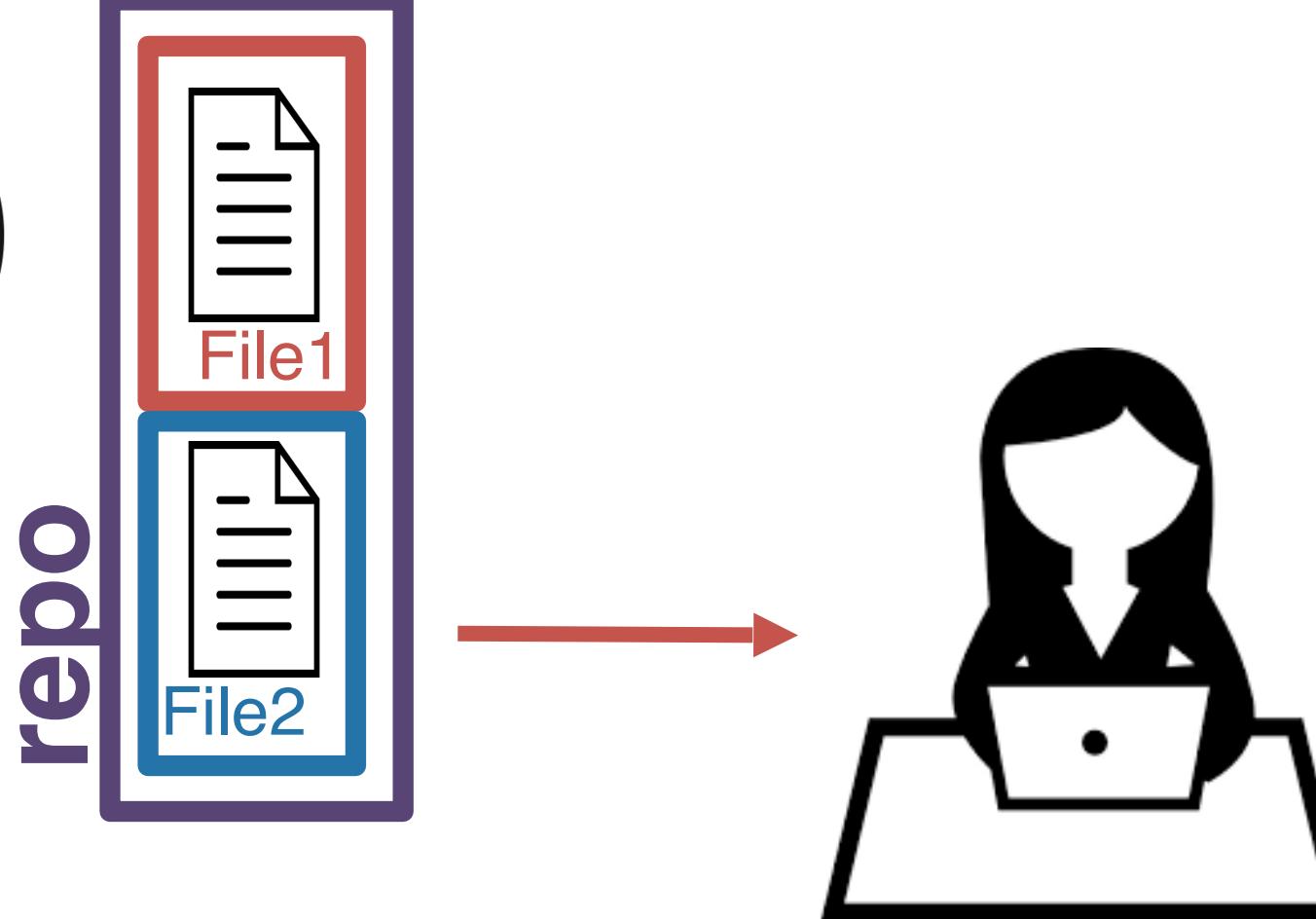
repo



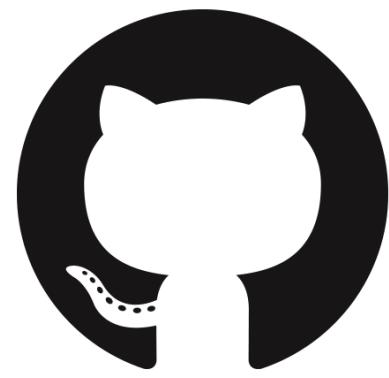
You decide you want to
change some of the text
in the project.



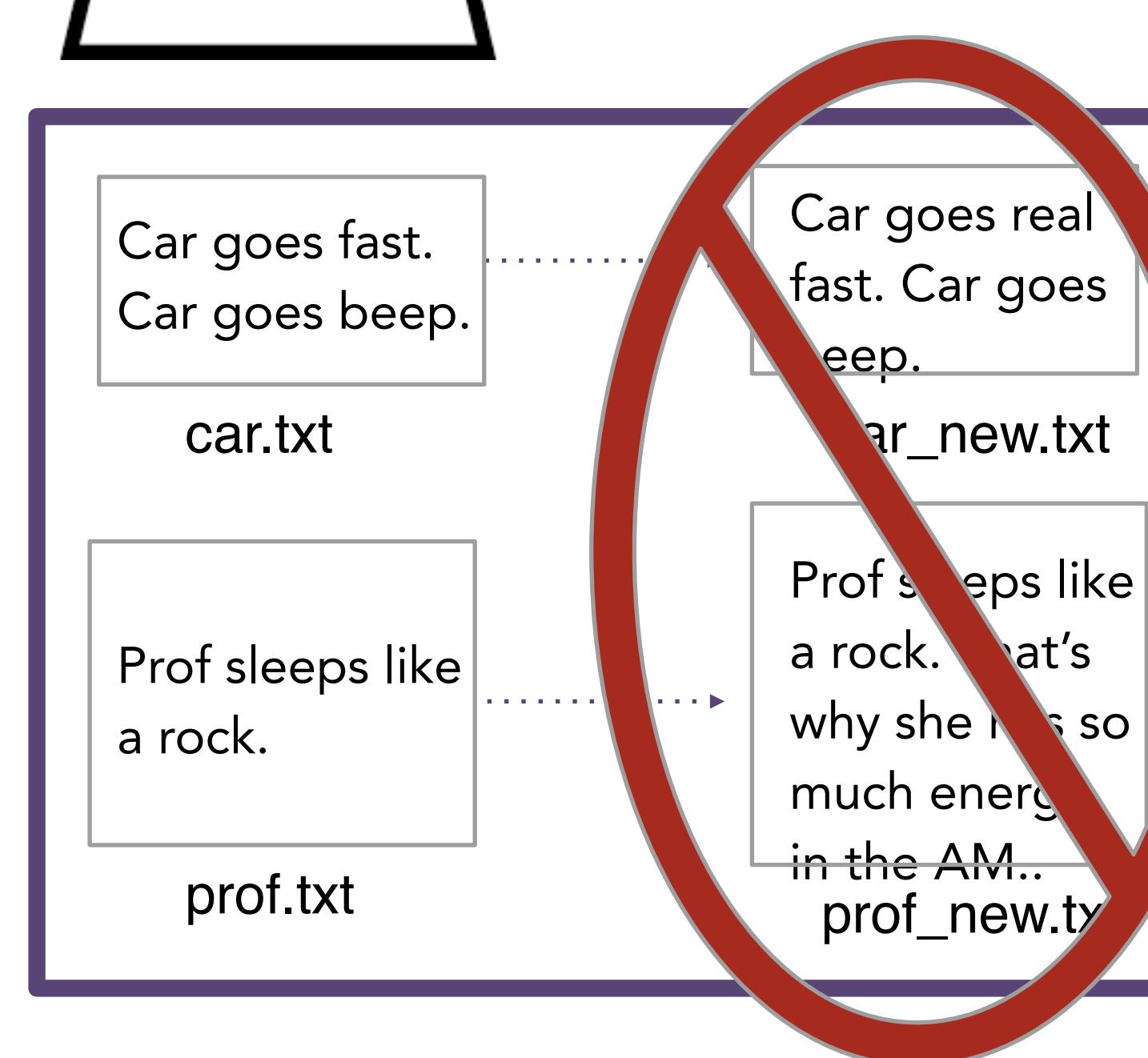
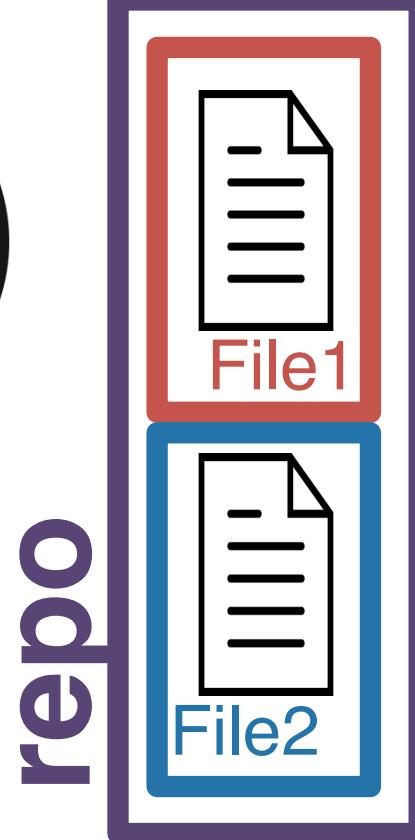
repo



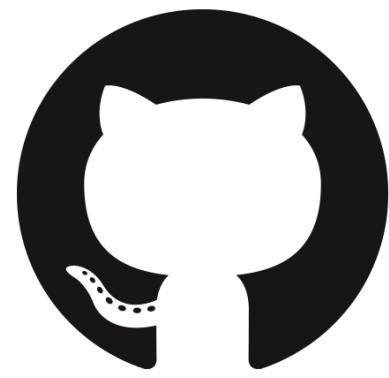
without git...you'd
likely rename
these files....



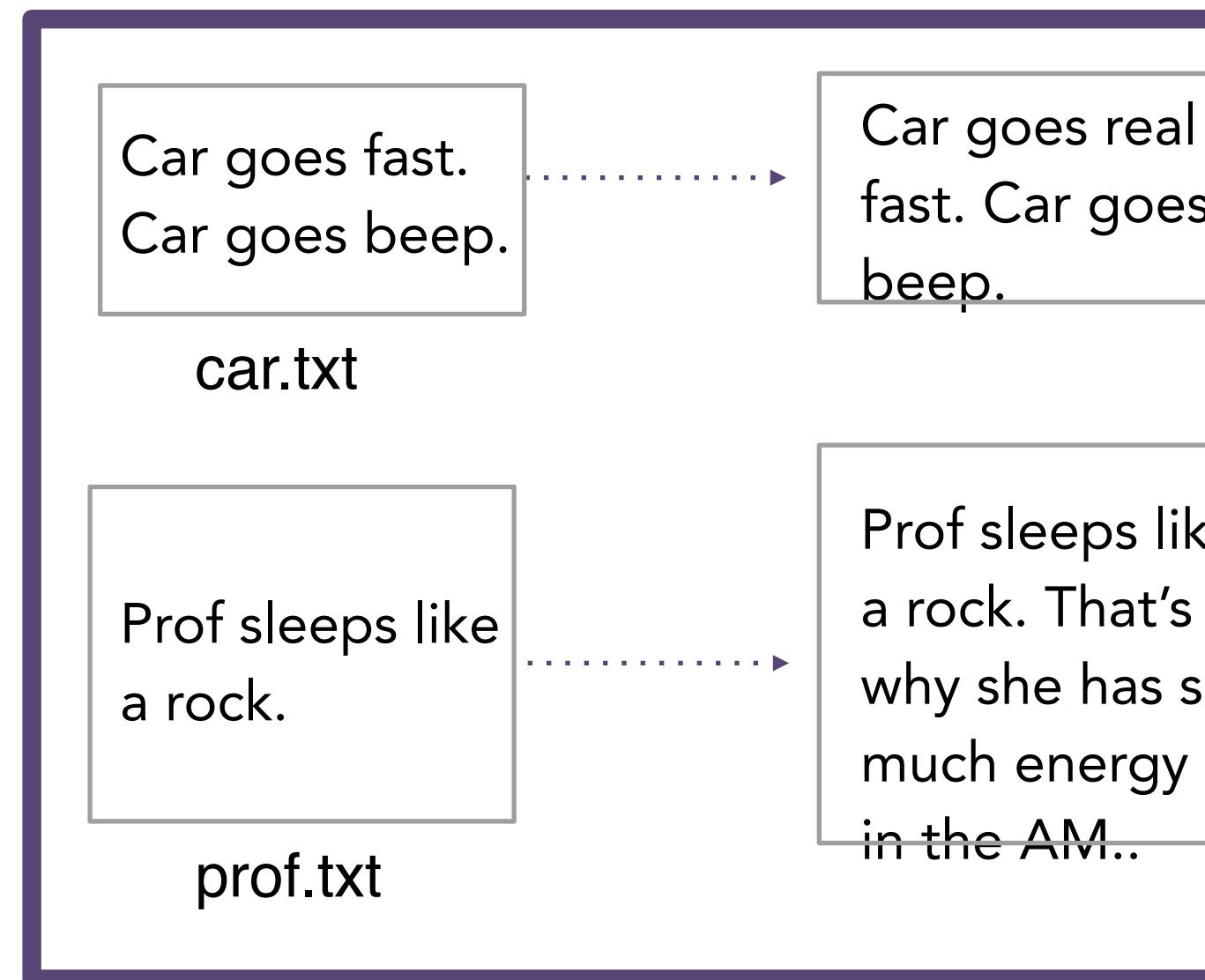
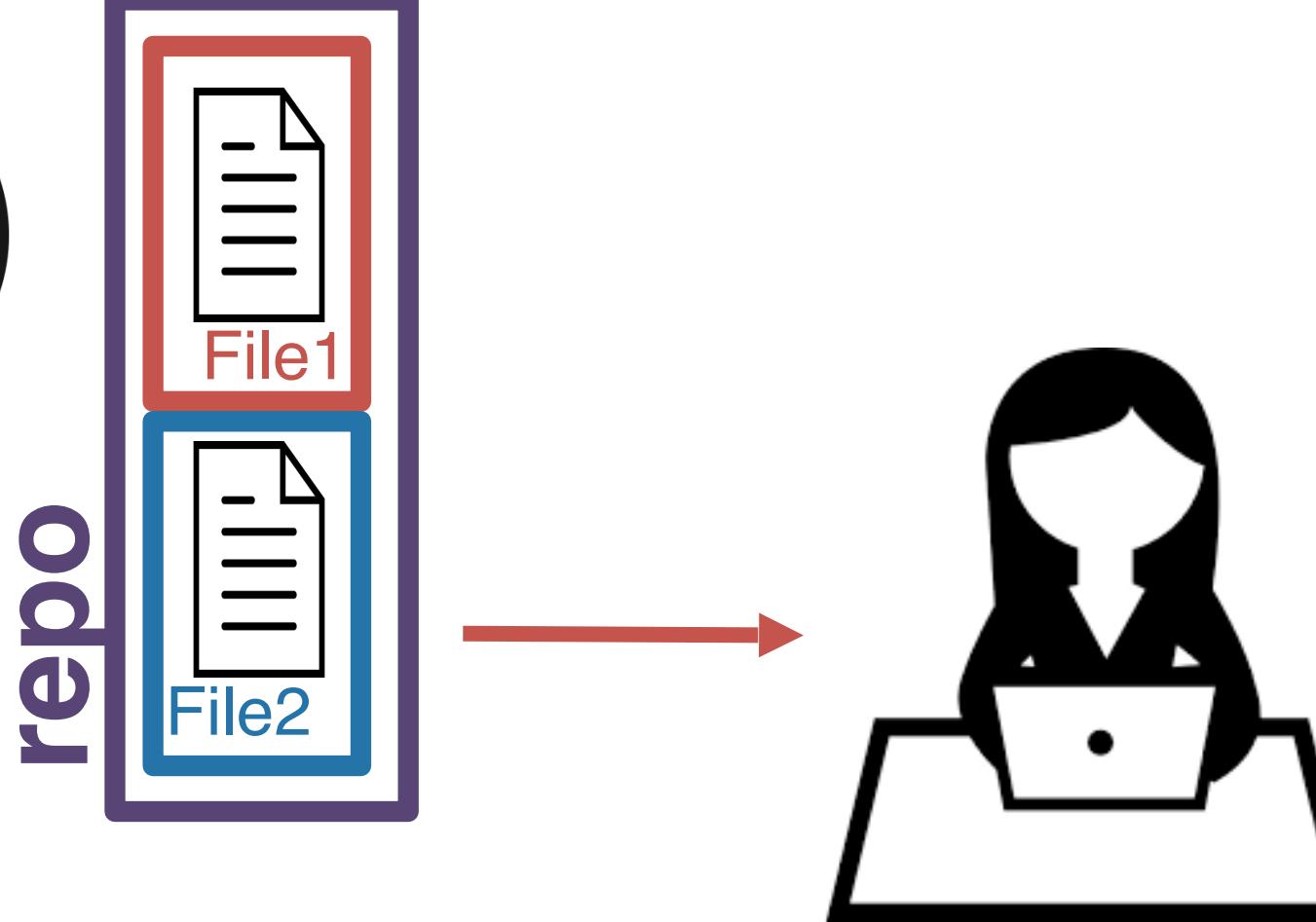
repo



Thank
goodness those
days are over!



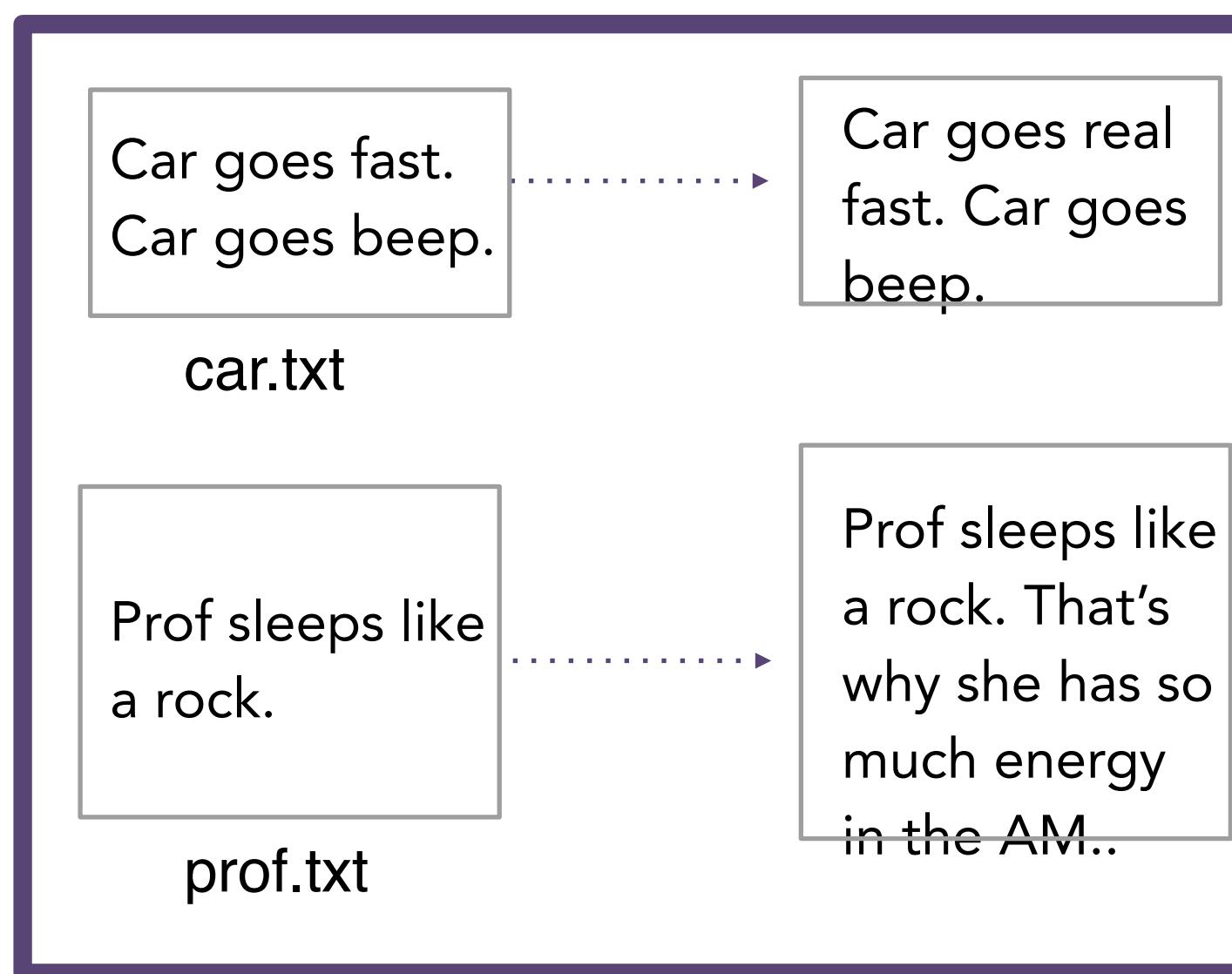
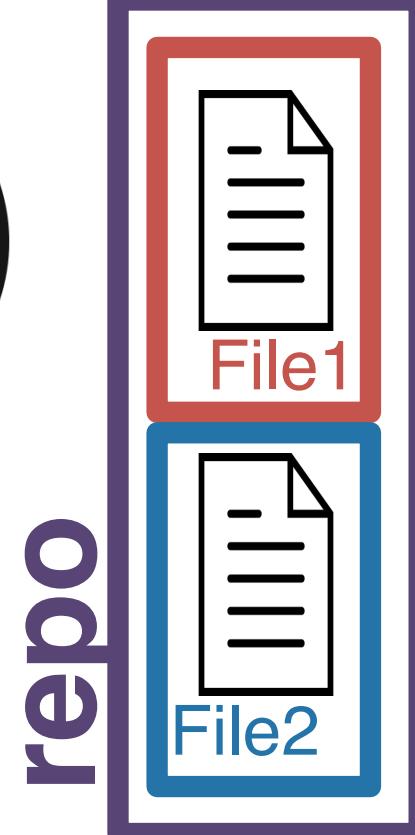
repo



Instead, you tell git which files you'd like to keep track of using **add**. This process is called *staging*.

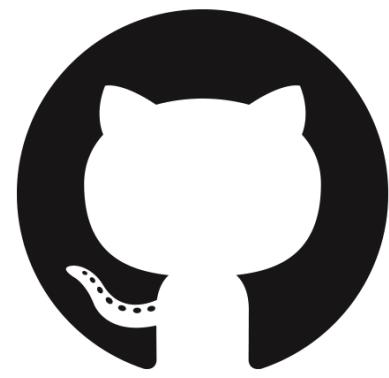


repo

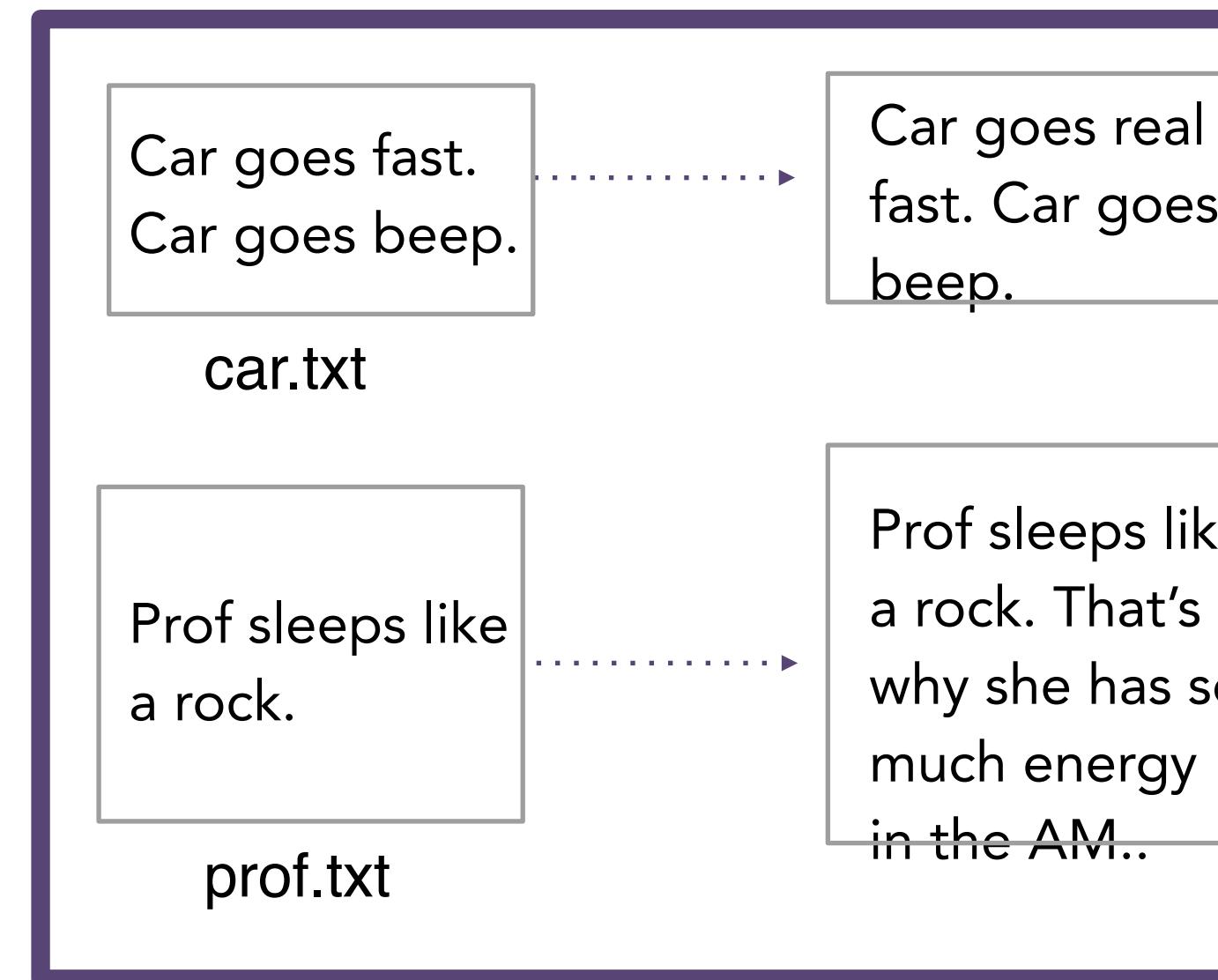
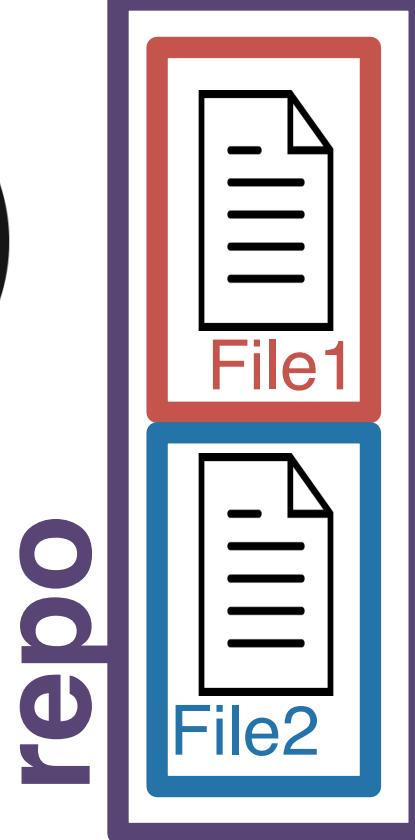


git add file	stages specified file (or folder)
git add .	stages new and modified files
git add -u	stages modified and deleted files
git add -A	stages new, modified, and deleted files
git add *.csv	Stages any files with .csv extension
git add *	Use with caution: stages everything

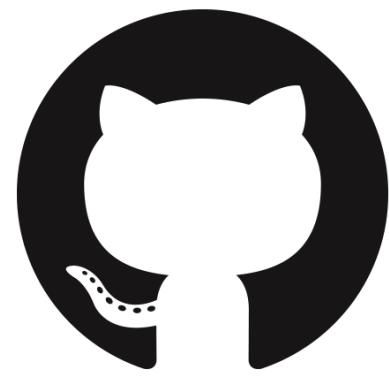
Instead, you tell git which files you'd like to keep track of using **add**. This process is called *staging*.



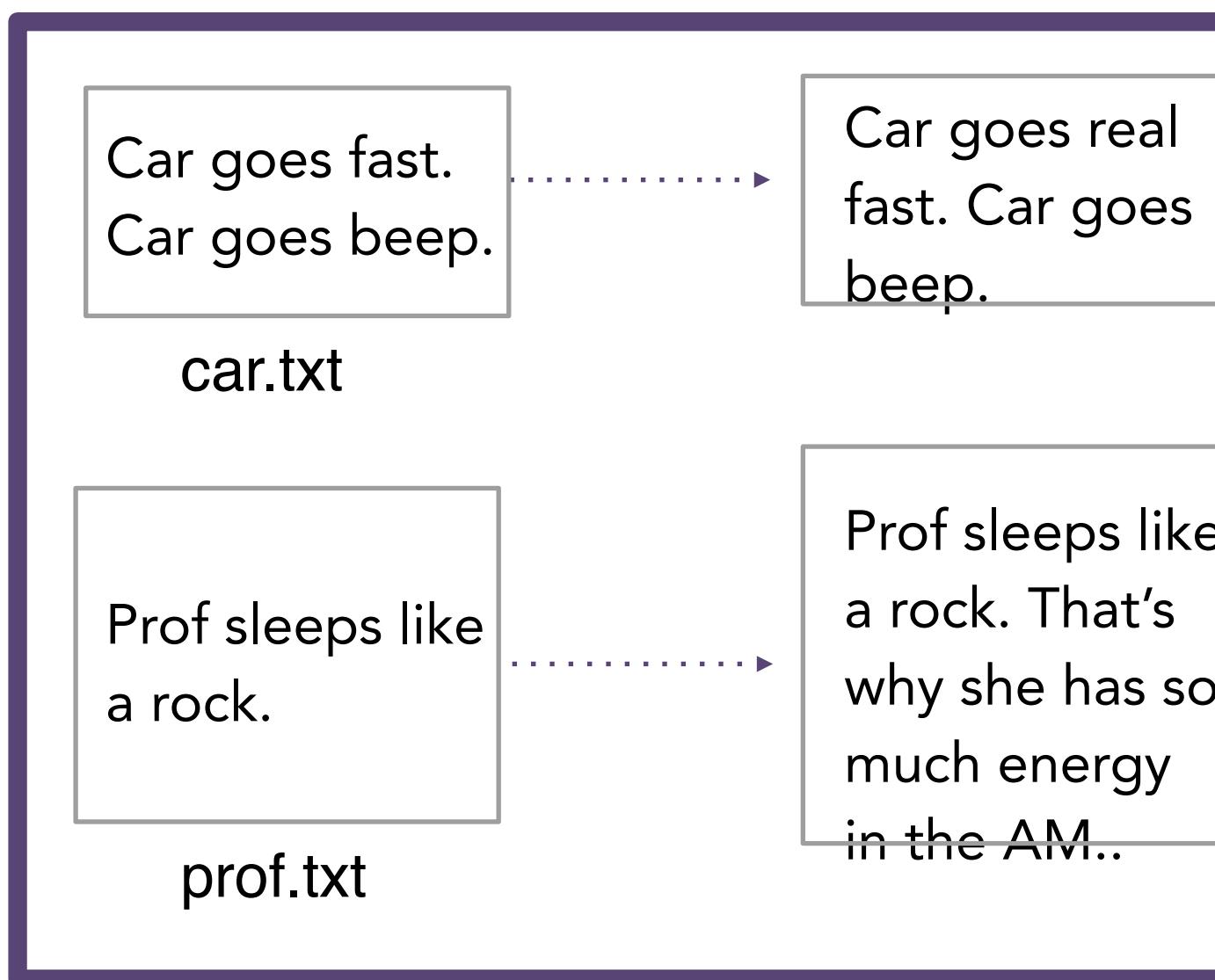
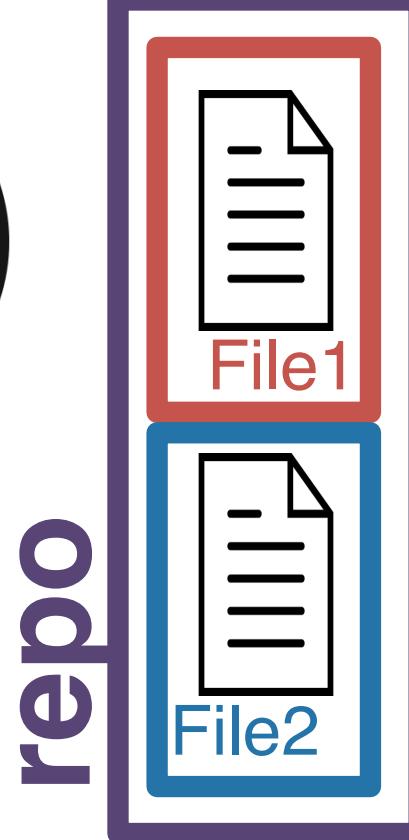
repo



Then, you create a snapshot of
your files at this point. This
snapshot is called a **commit**.



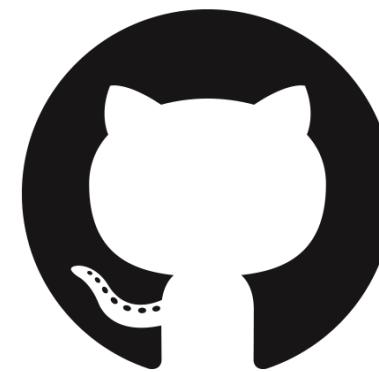
repo



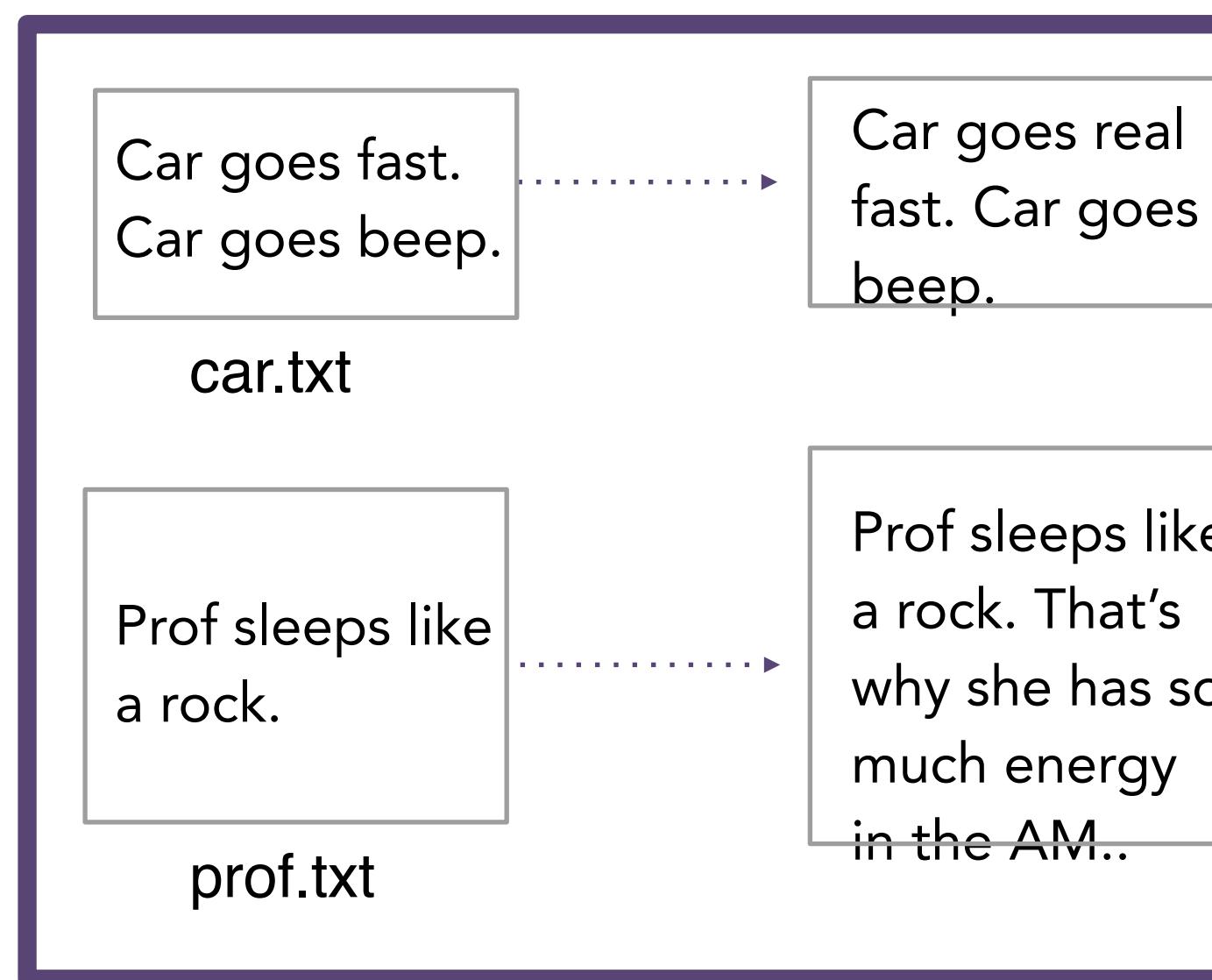
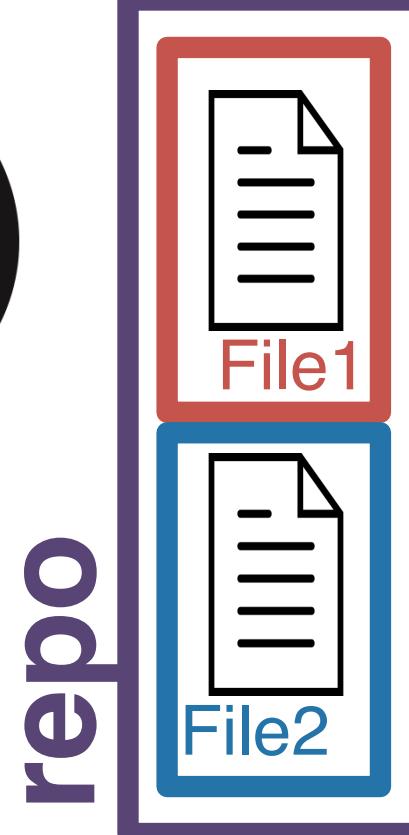
Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.



A **commit** tracks who, what, and when



repo

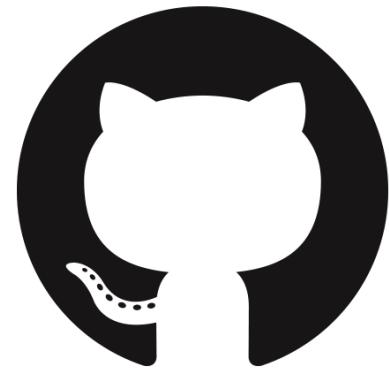


A **commit** tracks
who, what, and
when

You can make commits more informative by adding a **commit message**.

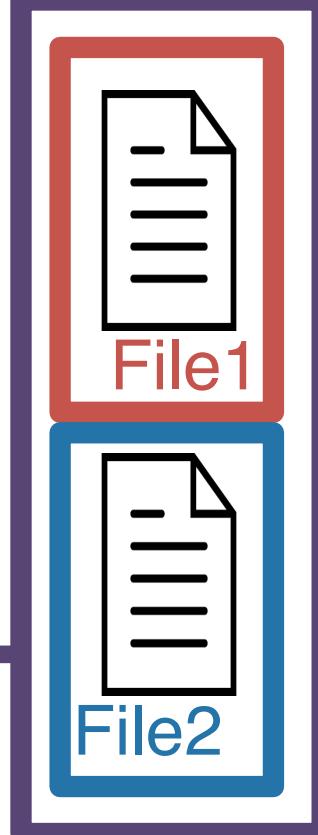
Example: `git commit -m 'fix typos in car and prof'`

Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.



repo

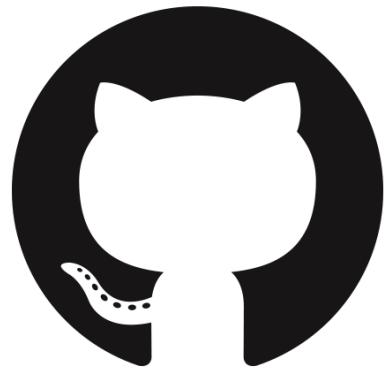
repo



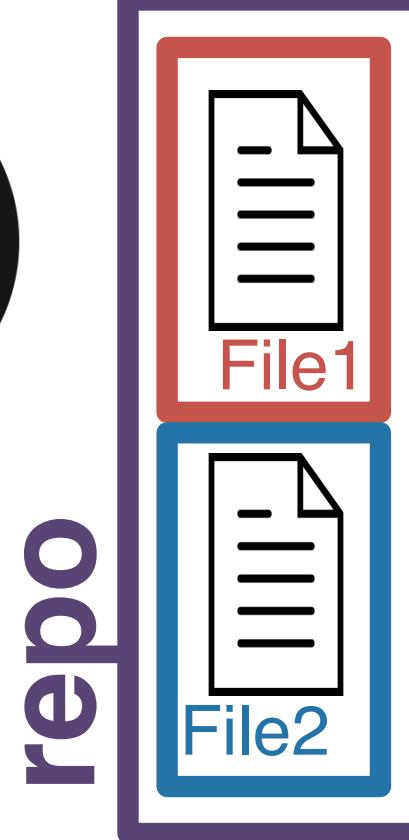
Shannon Ellis

3/28/21 3:28pm

fix typos in car and prof



repo



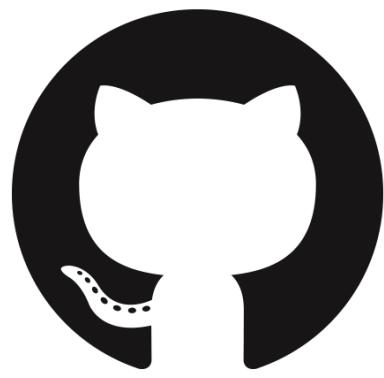
push

Remember, you're not the only one working on this project though! You want your teammates to have access to these changes! You **push** these changes back to the remote.

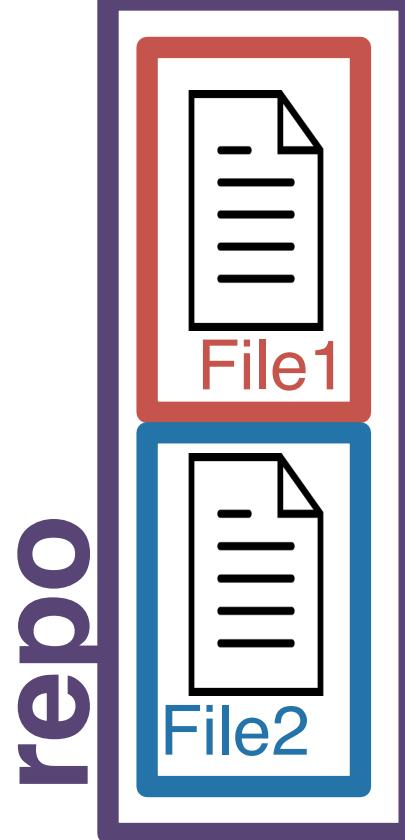


Shannon Ellis
3/28/21 3:28pm

fix typos in car and prof

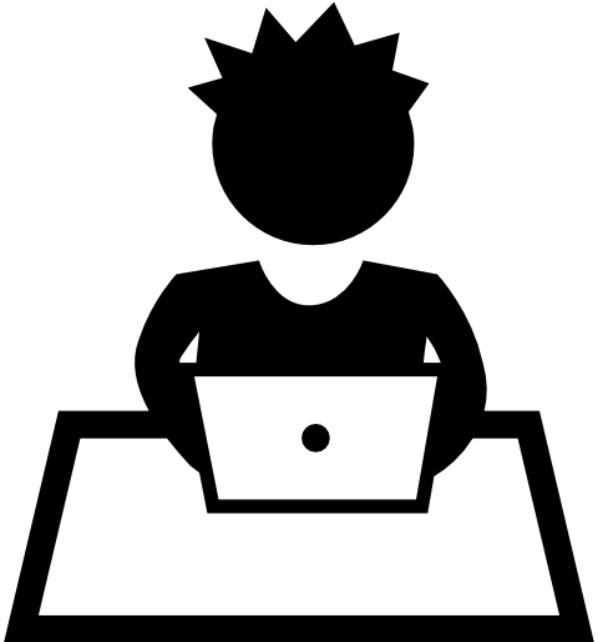


repo



Shannon Ellis
3/28/21 3:28pm

fix typos in car and prof

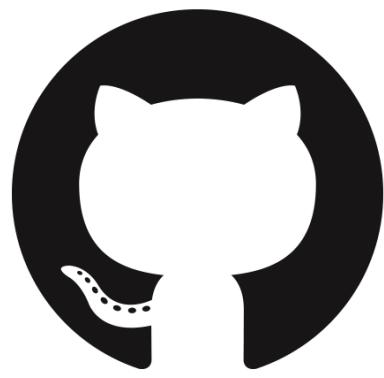


Your teammate is still
working with the (out-
of-date) copy he
cloned earlier!



Shannon Ellis
3/28/21 3:28pm

fix typos in car and prof



repo

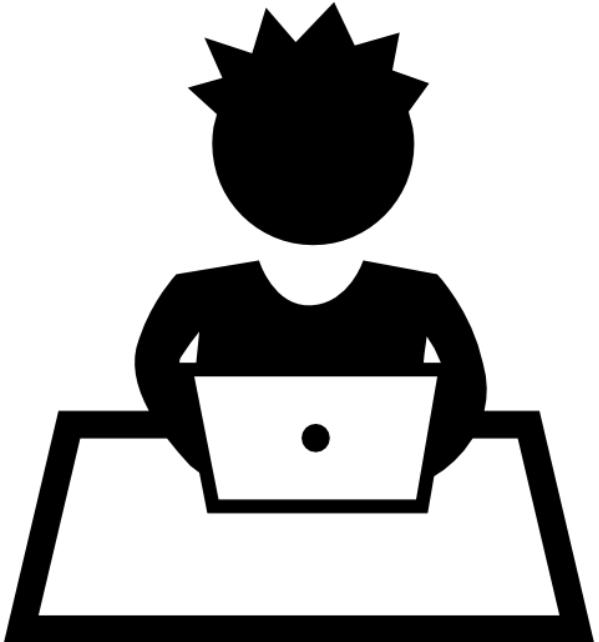


To catch up, your teammate will have to
pull the changes from GitHub (remote)



Shannon Ellis
3/28/21 3:28pm

fix typos in car and prof

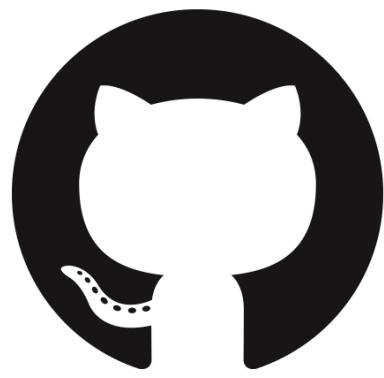


Your teammate is still
working with the (out-
of-date) copy he
cloned earlier!

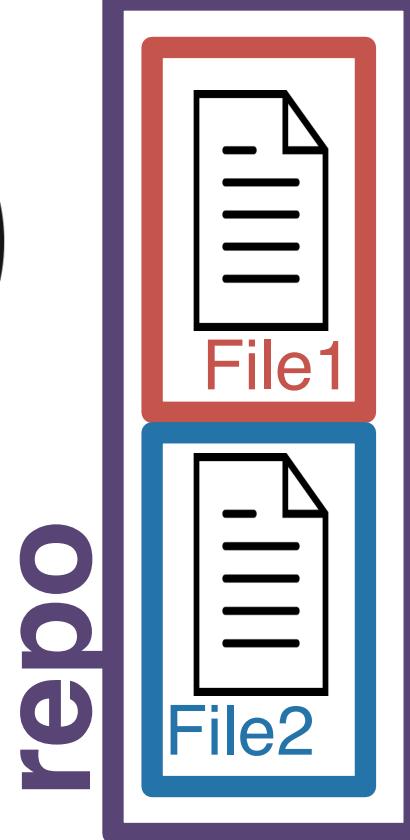


Shannon Ellis
3/28/21 3:28pm

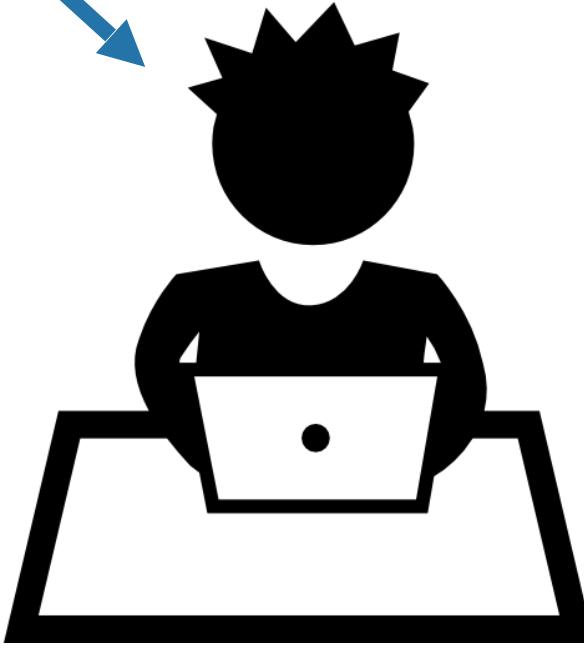
fix typos in car and prof



repo



pull



Your teammate pulls
from remote and is
now up-to-date!

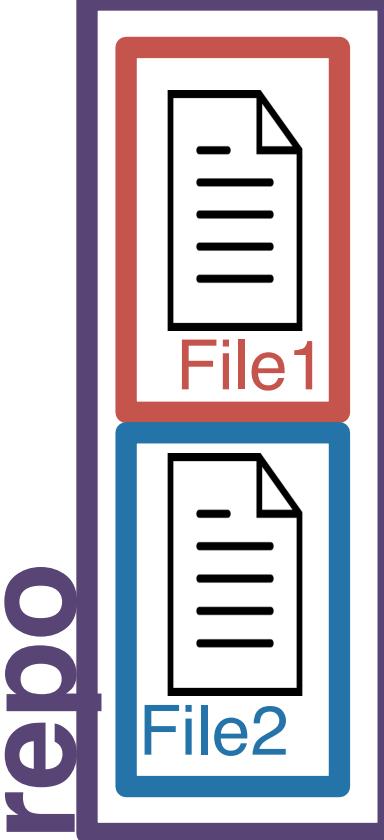


Shannon Ellis
3/28/21 3:28pm

fix typos in car and prof



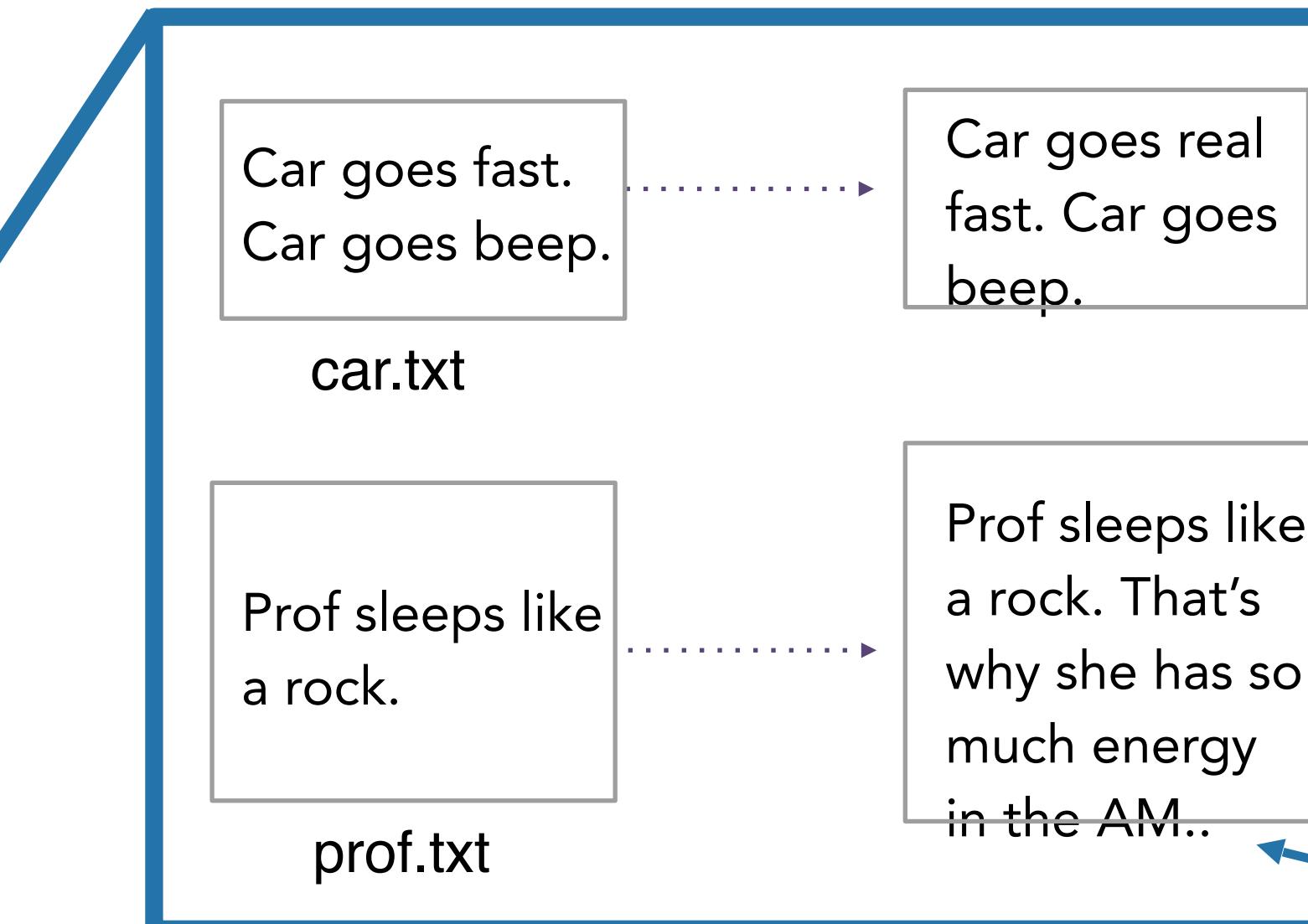
repo



pull



Your teammate pulls
from remote and is
now up-to-date!

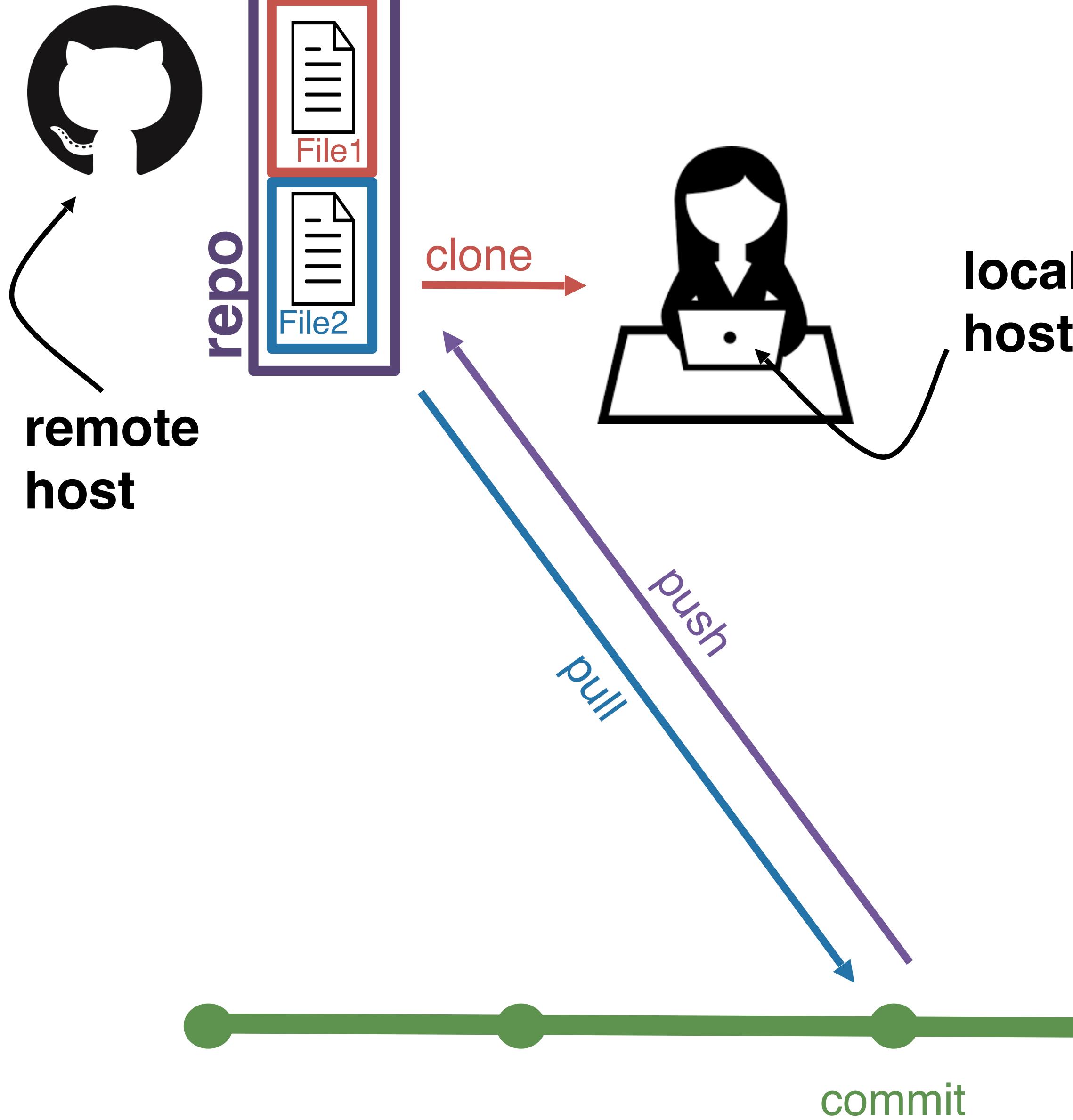


The files in his project
locally will now have
the updated files



Shannon Ellis
3/28/21 3:28pm

fix typos in car and prof



Let's recap real quick!

repo - set of files and folders for a project

remote - where the repo lives

clone - get the repo from the remote for the first time

add - specify which files you want to stage (add to repo)

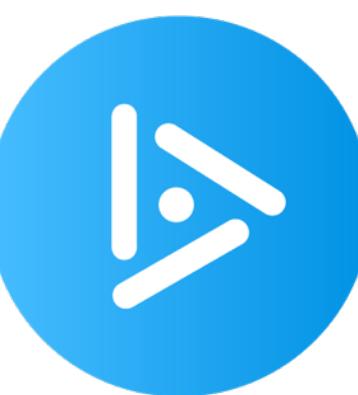
commit - snapshot of your files at a point in time

pull - get new commits to the repo from the remote

push - send your new commits to the remote

```
(base) sellis:Projects shannonellis$ git status  
On branch master  
Your branch is up to date with 'origin/master'.  
  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
  
    FinalProject_Guidelines.pdf  
  
nothing added to commit but untracked files present (use "git add" to track)  
(base) sellis:Projects shannonellis$ git add FinalProject_Guidelines.pdf  
(base) sellis:Projects shannonellis$ git commit -m "update Project Guidelines"  
[master 264e91a] update Project Guidelines  
  1 file changed, 0 insertions(+), 0 deletions(-)  
  create mode 100644 FinalProject_Guidelines.pdf  
(base) sellis:Projects shannonellis$ git push  
Counting objects: 3, done.  
Delta compression using up to 8 threads.  
Compressing objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 148.21 KiB | 29.64 MiB/s, done.  
Total 3 (delta 1), reused 0 (delta 0)  
remote: Resolving deltas: 100% (1/1), completed with 1 local object.  
To https://github.com/COGS108/Projects.git  
  6931768..264e91a  master -> master
```

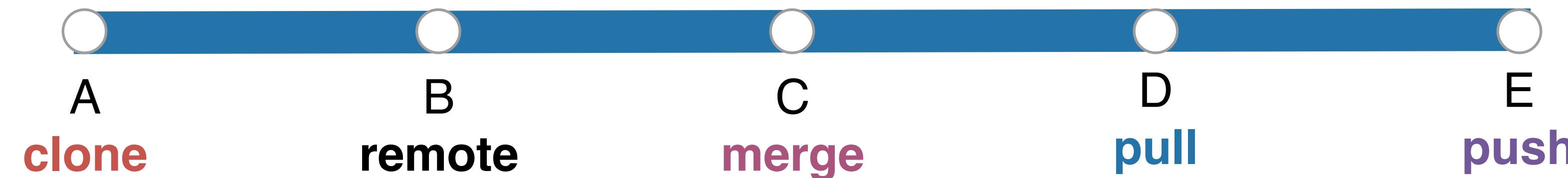
Review & Question Time

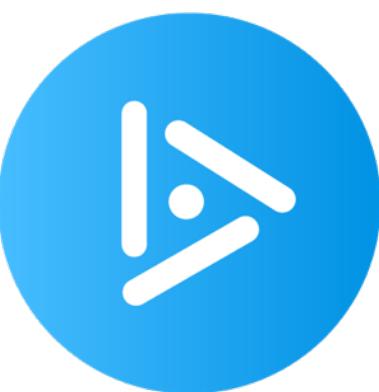


Version Controller I

You've been working with a team on a project in a repo. You've made changes locally and you want to see them on the remote.

What do you do to get them on the remote?

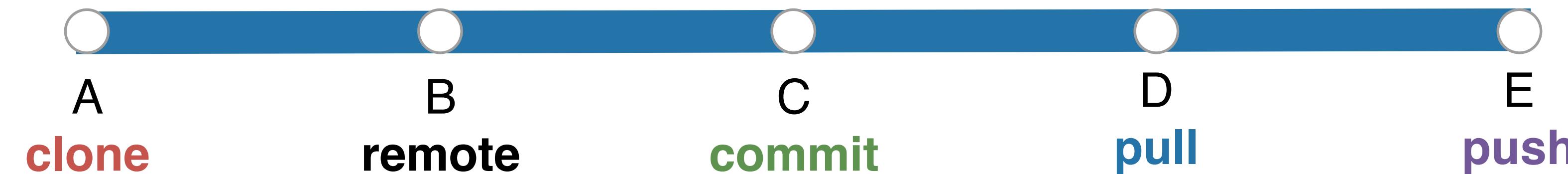


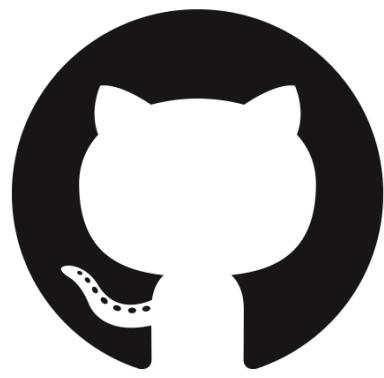


Version Controller II

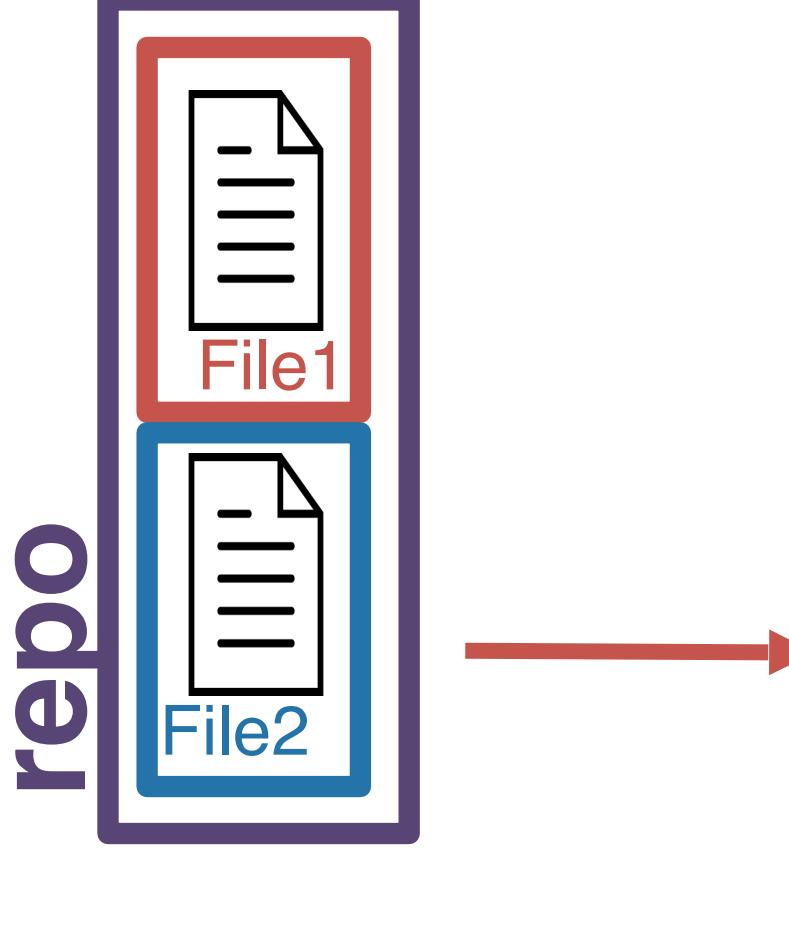
Your teammate has given you access to a GitHub repository to work on a project together. You want to get them for the first time on your computer locally.

What do you do to get the repo on your computer?



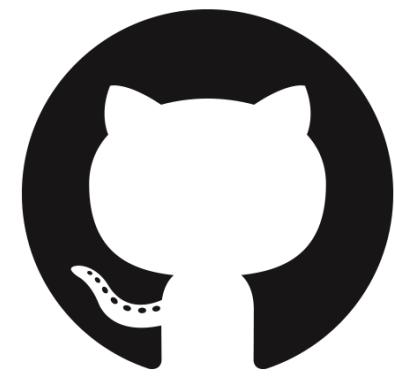


repo



Each time you create a commit, git tracks the changes made automatically.



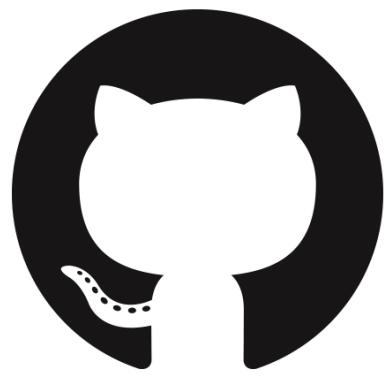


repo

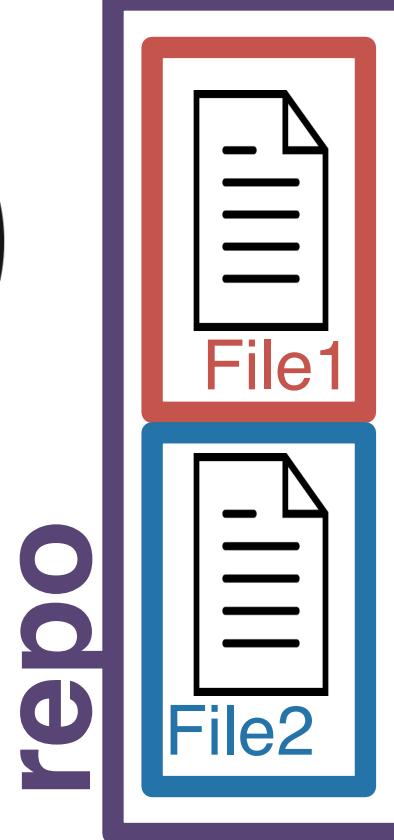


By committing each time you make changes, git allows you to time travel!





repo



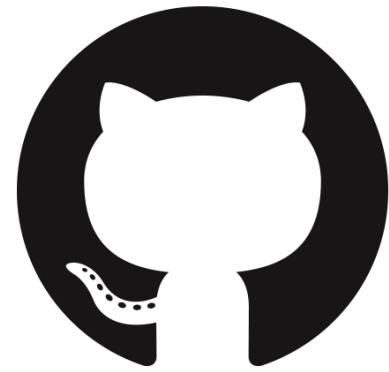
By committing each time you make changes, git allows you to time travel!

377dfcd00dd057542b112cf13be6cf1380b292
ad

439301fe69e8f875c049ad0718386516b4878
e22

456722223e9f9e0ee0a92917ba80163028d89
251

There's a unique id, known as a **hash**, associated with each commit.

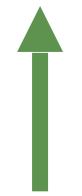


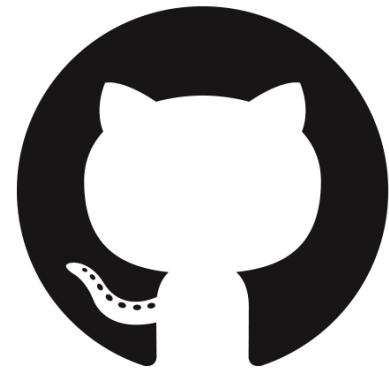
repo



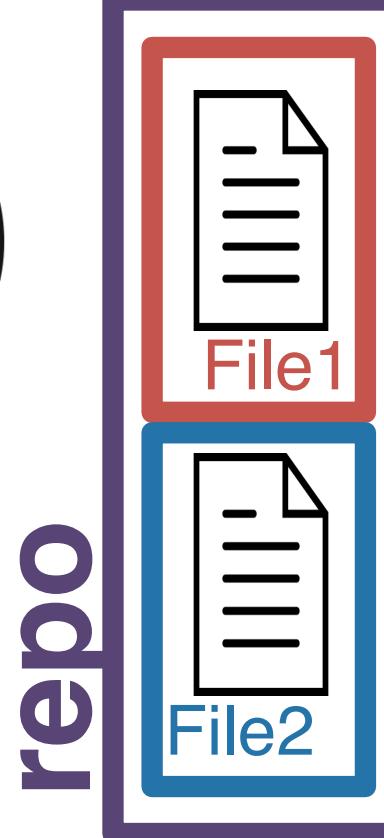
You can return to the state of the repository at any commit. Future commits don't disappear. They just aren't visible when you **check out** an older commit.

377dfcd00dd057542b112cf13be6cf1380b292
ad



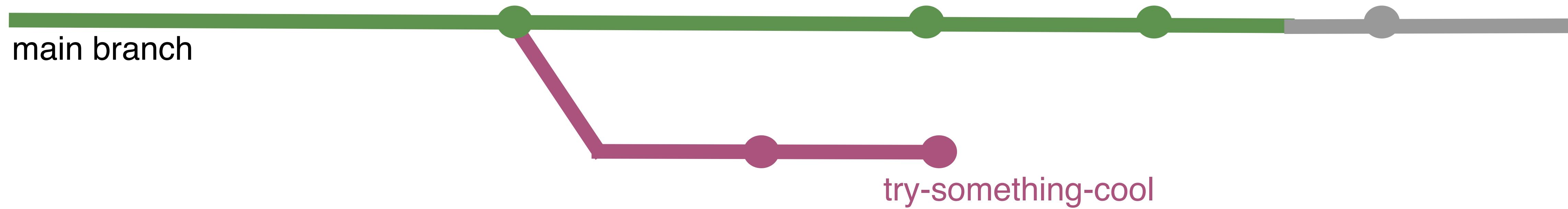


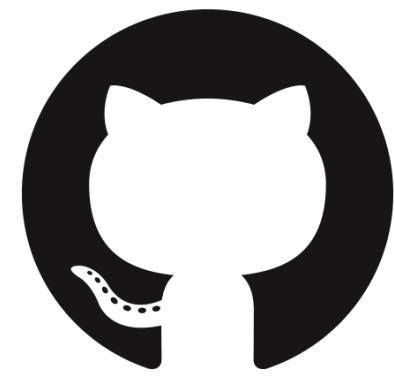
repo



main branch

But...not everything is always linear.
Sometimes you want to try something
out and you're not sure it's going to
work. This is where you'll want to use a
branch.



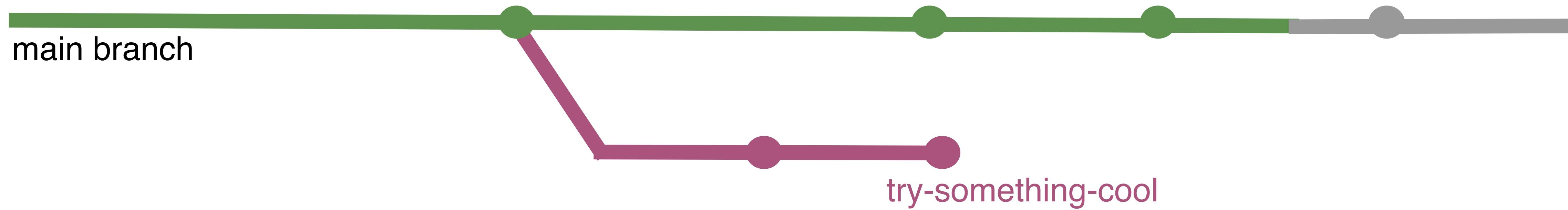


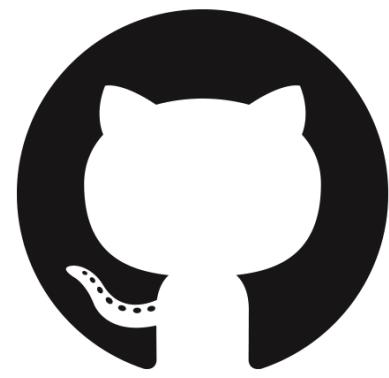
repo



main branch

It's a good way to experiment. It's pretty easy to get rid of a branch later on should you not want to include the commits on that branch.





repo



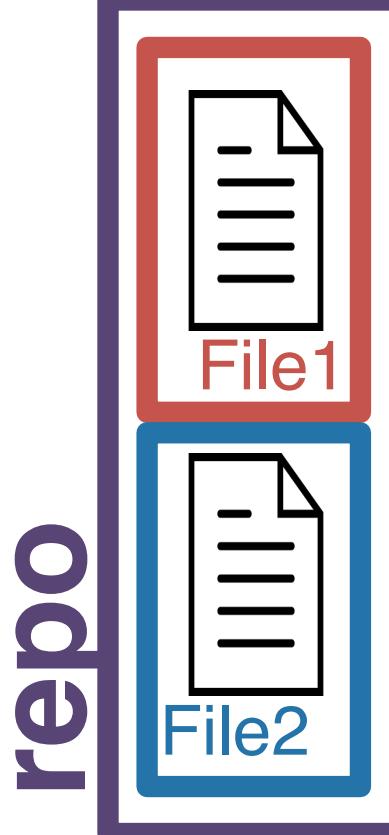
main

But...what if you DO want to include the changes you've made on your try-something-cool branch into the main branch?

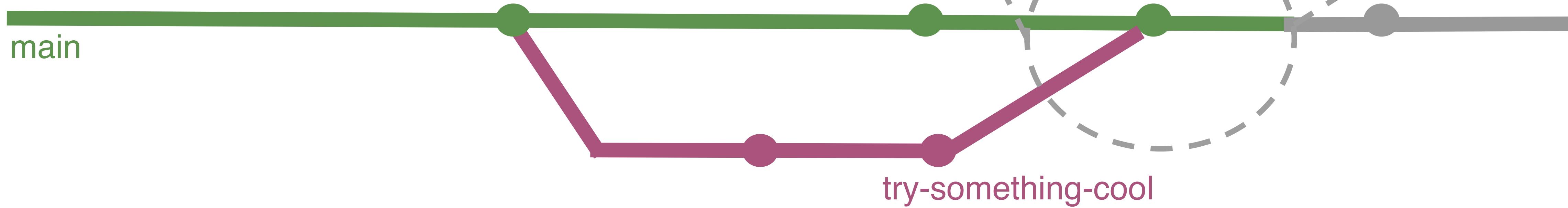
try-something-cool



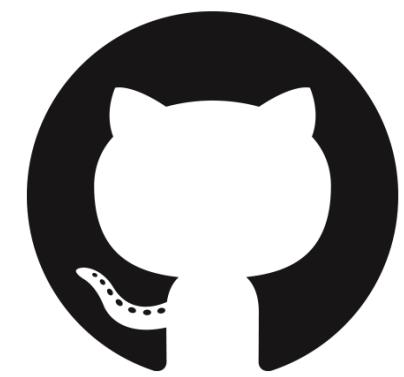
repo



main



A merge allows you to combine the commits from a branch back into the main.

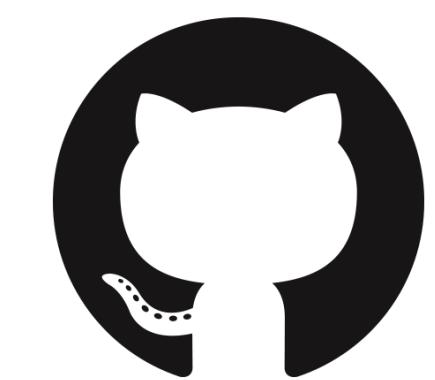


someone
else's
repo

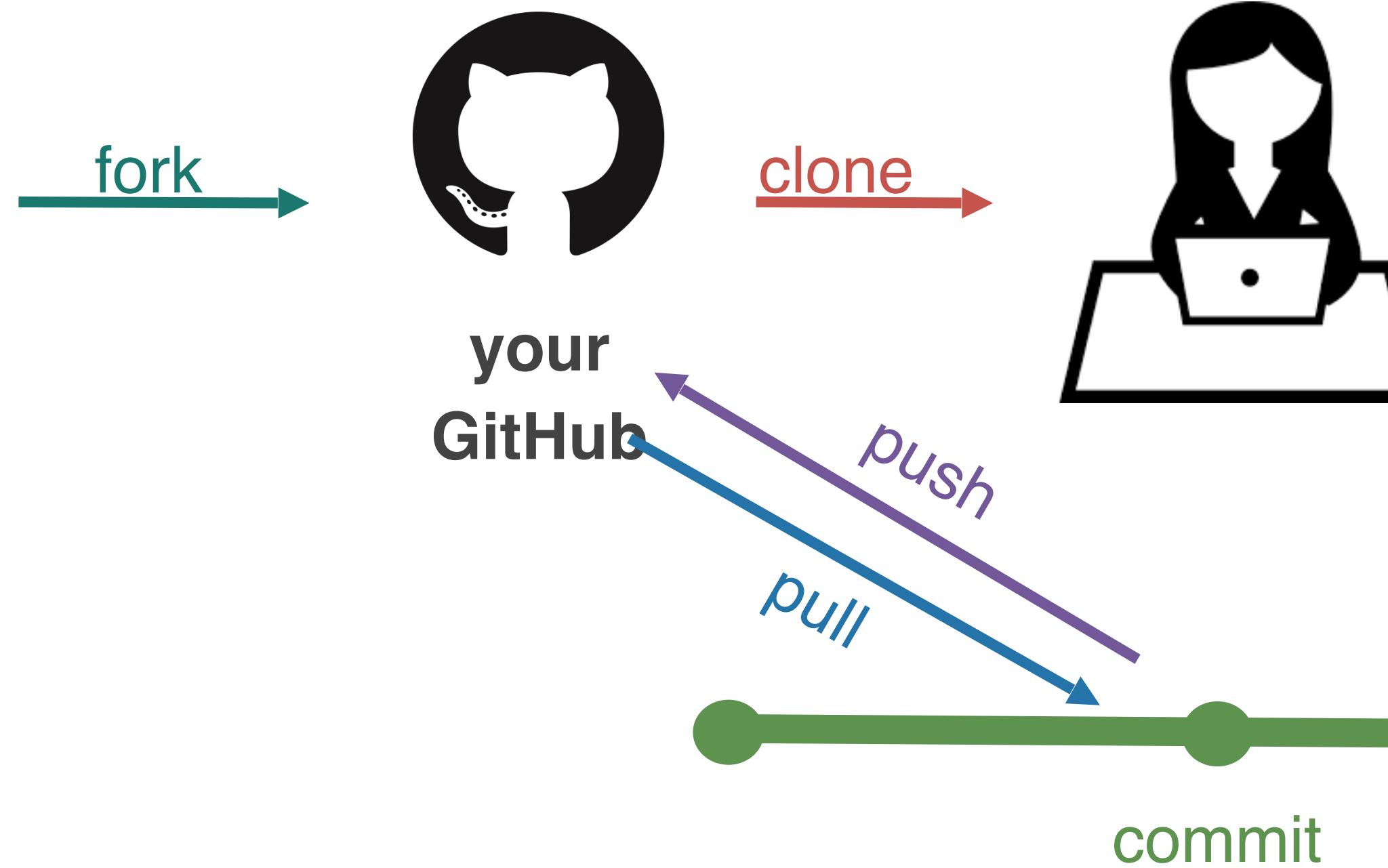


your
GitHub

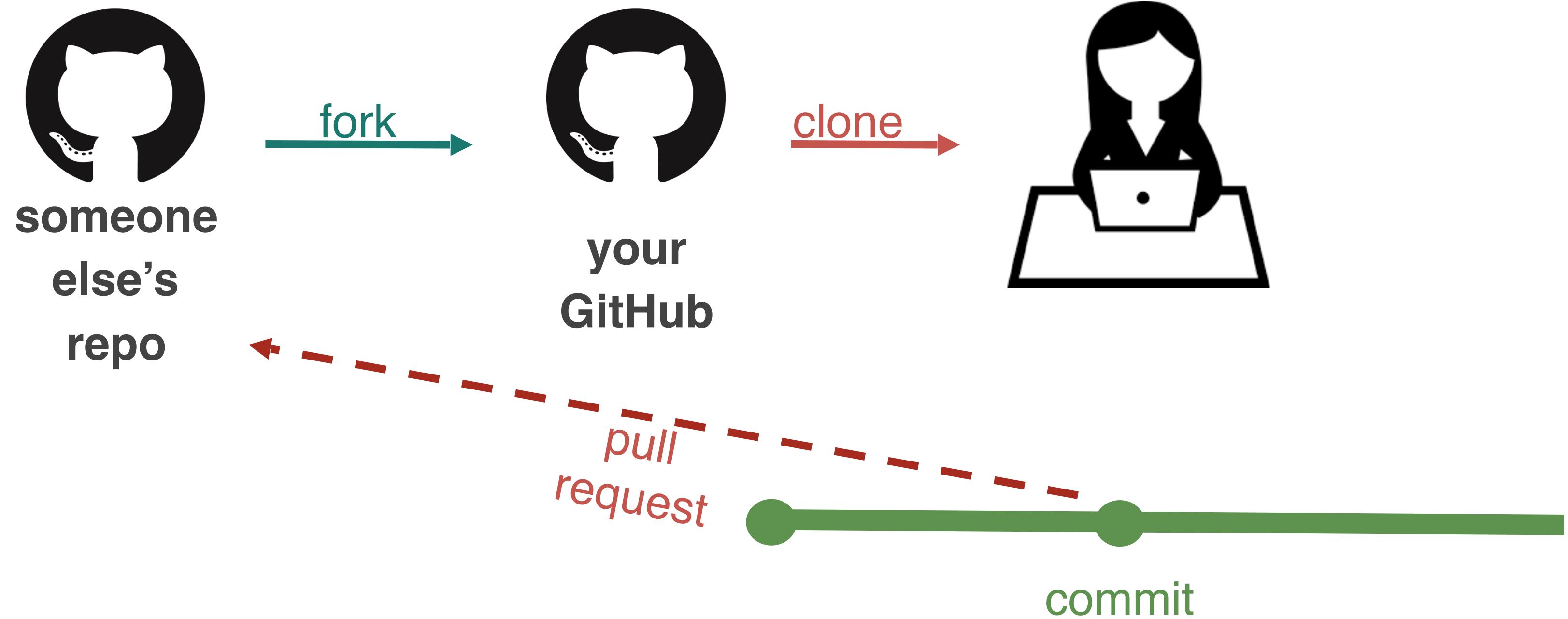
What if someone else is working on something cool and you want to play around with it? You'll have to **fork** their repo.



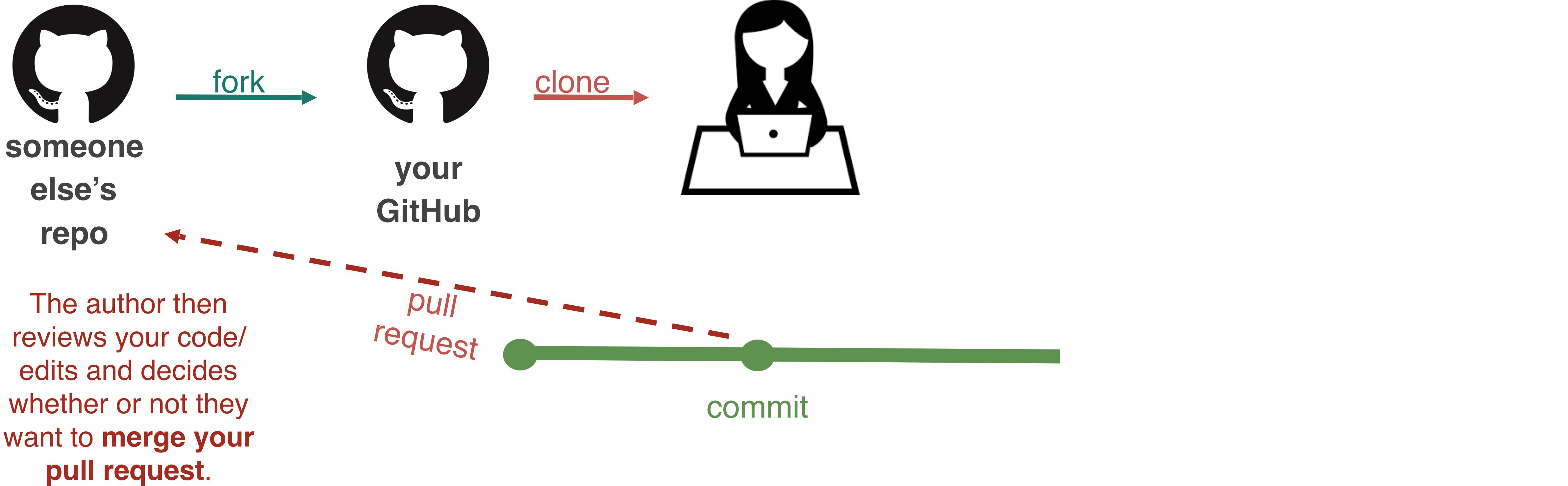
someone
else's
repo



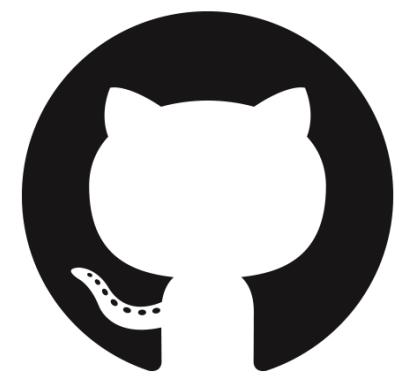
After you fork their repo, you can play around with it however you want, using the workflow we've already discussed.



But what if you think you've found a bug in their code, a typo, or want to add a new feature to their software? For this, you'll submit a **pull request** (aka **PR**).

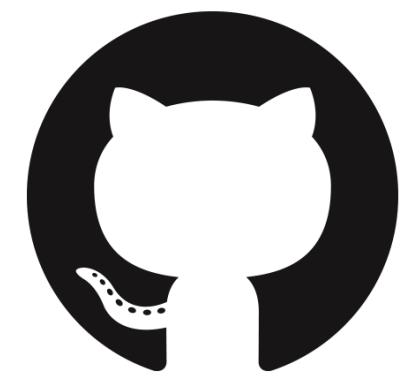


But what if you think you've found a bug in their code, a typo, or want to add a new feature to their software? For this, you'll submit a **pull request** (aka **PR**).



someone
else's
repo

Last but not least...what if you find a bug in someone else's code OR you want to make a suggestion but aren't going to submit a suggestion with a PR. For this, you can file an **issue** on GitHub.



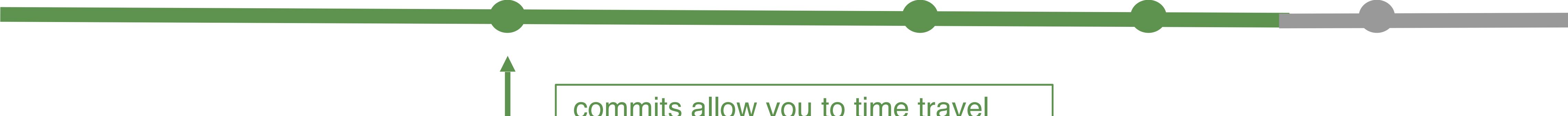
someone
else's
repo

Last but not least...what if you find a bug in someone else's code OR you want to make a suggestion but aren't going to submit a suggestion with a PR. For this, you can file an **issue** on GitHub.

Issues are *bug trackers*. While, they can include bugs, they can also include feature requests, to-dos, whatever you want, really!

They can be assigned to people.

They can be closed once addressedor if the software maintainer doesn't like the suggestion



377dfcd00dd057542b112cf13be6cf1380b292
ad

commits allow you to time travel
because each commit is assigned
a unique **hash**

One more git recap...

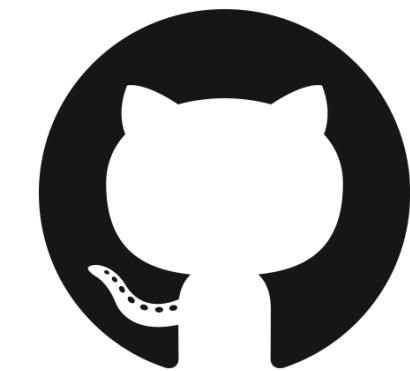


One more git recap...

377dfcd00dd057542b112cf13be6cf1380b292
ad

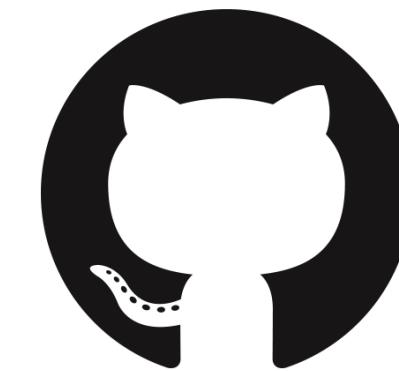
commits allow you to time travel
because each commit is assigned
a unique **hash**

main branch



someone
else's
repo

fork



your
GitHub

You can work on others'
repos by first **forking** their
repository onto your GitHub

try-something-cool

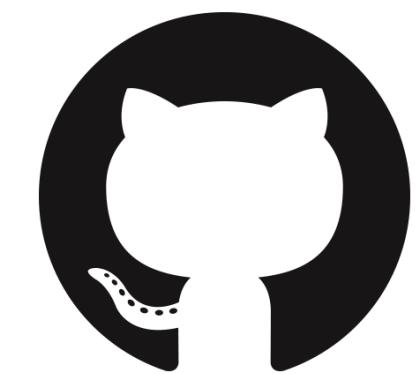
branches allow you to
experiment. branches can be
abandoned or **merged**

One more git recap...

377dfcd00dd057542b112cf13be6cf1380b292
ad

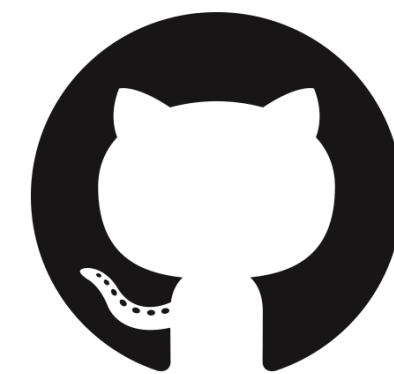
commits allow you to time travel
because each commit is assigned
a unique **hash**

main branch



someone
else's
repo

fork



your
GitHub

You can work on others'
repos by first **forking** their
repository onto your GitHub

try-something-cool

branches allow you to
experiment. branches can be
abandoned or **merged**

Pull requests allow you to make
specific edits to others' repos

Issues allow you to make general
suggestions to your/others' repos

One more git recap...

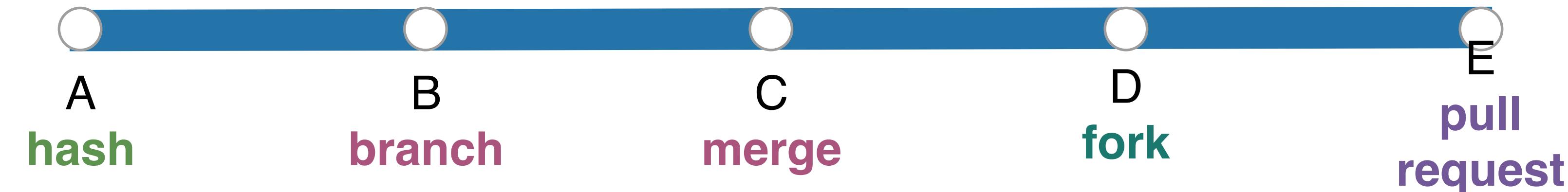
Review & Question Time



Version Controller III

To experiment within your own repo (test out a new feature, make some changes you're not sure will work)...

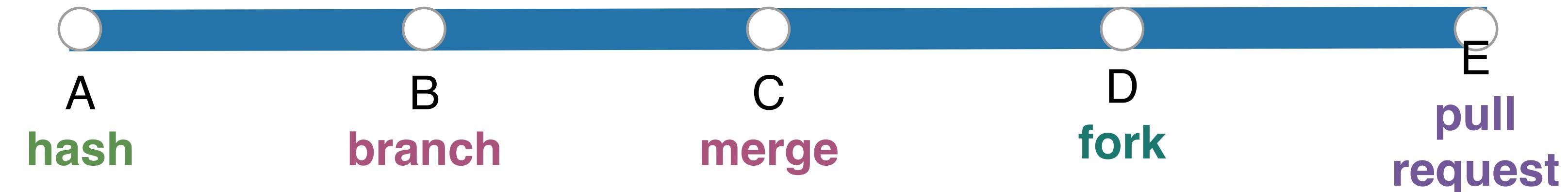
what should you do?





Version Controller IV

If you've made edits to someone else's repo that you're not a collaborator on...
what would *they* have to do to incorporate your changes?



Jupyter notebooks have problems for version control

<https://nextjournal.com/schmudde/how-to-version-control-jupyter>

Version Control: Practice

- Discussion Lab 1: Part 3
- Assignment 1: Part 1
 - This will get you practice with git & GitHub
 - Understand what you're doing in the assignment!
 - You may have to google, ask others, spend some time with this!
 - Part II is a Python review; each part of this assignment is self-contained
 - Do this part of the assignment ASAP
- git & Github == How to get the course lectures/materials
 - Assignment 1 will have you fork the Lectures and Project repos
 - You can keep the lectures up-to-date throughout the quarter
- you'll be using GitHub for your final projects

Note: You're encouraged to put projects on GitHub. Please do not put assignments on GitHub.

COGS 108 Final Projects

The COGS 108 Final Project will give you the chance to explore a topic of your choice and to expand your analytical skills. By working with real data of your choosing you can examine questions of particular interest to you.

- You are encouraged to work on a topic that matters to the world (your family, your neighborhood, a state/province, country, etc).
- Taboo Topics: Movie Predictions/Recommendation System; YouTube Data Analysis, Kickstarter success prediction/analysis, prediction of what makes a song popular on Spotify; political patterns or singling out some individual

Final Project: Objectives

- Identify the problems and goals of a *real* situation and dataset.
- Choose an appropriate approach for formalizing and testing the problems and goals, and be able to articulate the reasoning for that selection.
- Implement your analysis choices on the dataset(s).
- Interpret the results of the analyses.
- Contextualize those results within a greater scientific and social context, acknowledging and addressing any potential issues related to privacy and ethics.
- Work effectively to manage a project as part of a team.

Upcoming Project Components

Project Planning Survey (1%) - 1 submission per group (due Fri Week 2)

Project Review (5%) - Before Mon of week 3, your group will be assigned a previous COGS 108 project to review; A google Form will be released to guide your thinking/discussion about and review of what a previous COGS 108 group did for their project. (due Fri Week 3)

Project Proposal (8%) - a GitHub repo will be created for your group; ‘submit’ on GitHub (due Fri Week 4)

Project Proposal (8%)

Full project guidelines are here [https://github.com/COGS108/Projects/
blob/master/FinalProject_Guidelines.md](https://github.com/COGS108/Projects/blob/master/FinalProject_Guidelines.md)