

Course Announcements

Due dates

- Please complete pre-course survey (due *next* Sunday)
- Q1 (Canvas) due Sunday; released 3PM Friday

Notes

- Lab today/Friday is a Python review - *nothing to turn in*
- D1 & A1 will be posted Friday
- Office Hours begin next week

COGS 108 Final Project

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 you (you, your family, your neighborhood, your community, a state/province, country, community, etc).
- **Taboo Topics:** Movie Predictions/Recommendation System; YouTube Data Analysis, Kickstarter success prediction/analysis, prediction of what makes a song popular on Spotify & Whatever is MOST popular EVER and whatever is HOTTEST RN on Kaggle

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.

Finding groupmates

- Chat with people in lecture
- Dedicated time in lab this week and next
- Use Ed Megathread

...and there's probably a Discord for this class? Feel free to share on Ed.

Best to have overlap on: topic interest, schedules, how you prefer to work

Best to have diverse skillset across team

Upcoming Project Components

- **Groups fixed** (week 3) - there will be a Google Form where you can submit your group (released Friday)
- **Project Review** (5%) - your group will be assigned two previous COGS 108 projects 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. (Week 4)
- **Project Proposal** (8%) - a GitHub repo will be created for your group; 'submit' on GitHub (Week 5)

Project Proposal

Full project guidelines:

https://github.com/COGS108/Projects/blob/wi25/FinalProject_Guidelines.md#project-proposal

! Shannon: tell students how to get notes in datahub and give a quick tour of datahub before you start talking about version control!








Version Control

Shannon E. Ellis, Ph.D
UC San Diego



Department of Cognitive Science
sellis@ucsd.edu

This sucks

Documents	Kind
 K99_Ellis_SpecAims_v2_ajEdits.docx	Micros... (.docx)
 K99_Ellis_v1_FAedit.docx	Micros... (.docx)
 K99_Ellis_v2	Micros... (.docx)
 K99_Ellis_v2_ajEdits.docx	Micros... (.docx)
 K99_Ellis_v2_FAedit.docx	Micros... (.docx)
 K99_Ellis_v3	Micros... (.docx)
 K99_Ellis_v4.docx	Micros... (.docx)

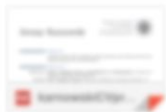
Yup, this sucks too.

May 11



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,



May 11



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

...

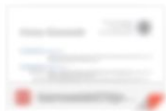


May 11



Final copy, I swear. Thanks for helping out.

...



This is a step in the right direction

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?

issues 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.

Total: 9 edits

^ v

Version history

Only show named versions ☐

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

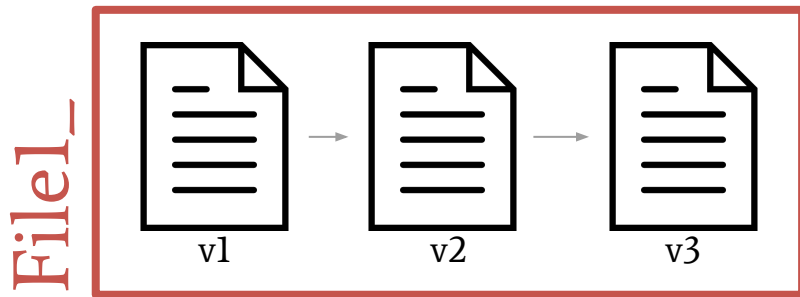
- Enables multiple people to simultaneously work on a single project.
- Each person edits their own copy of the files and chooses when to share those changes with the rest of the team.
- Thus, temporary or partial edits by one person do not interfere with another person's work

What is version control?

A way to manage the evolution of a set of files

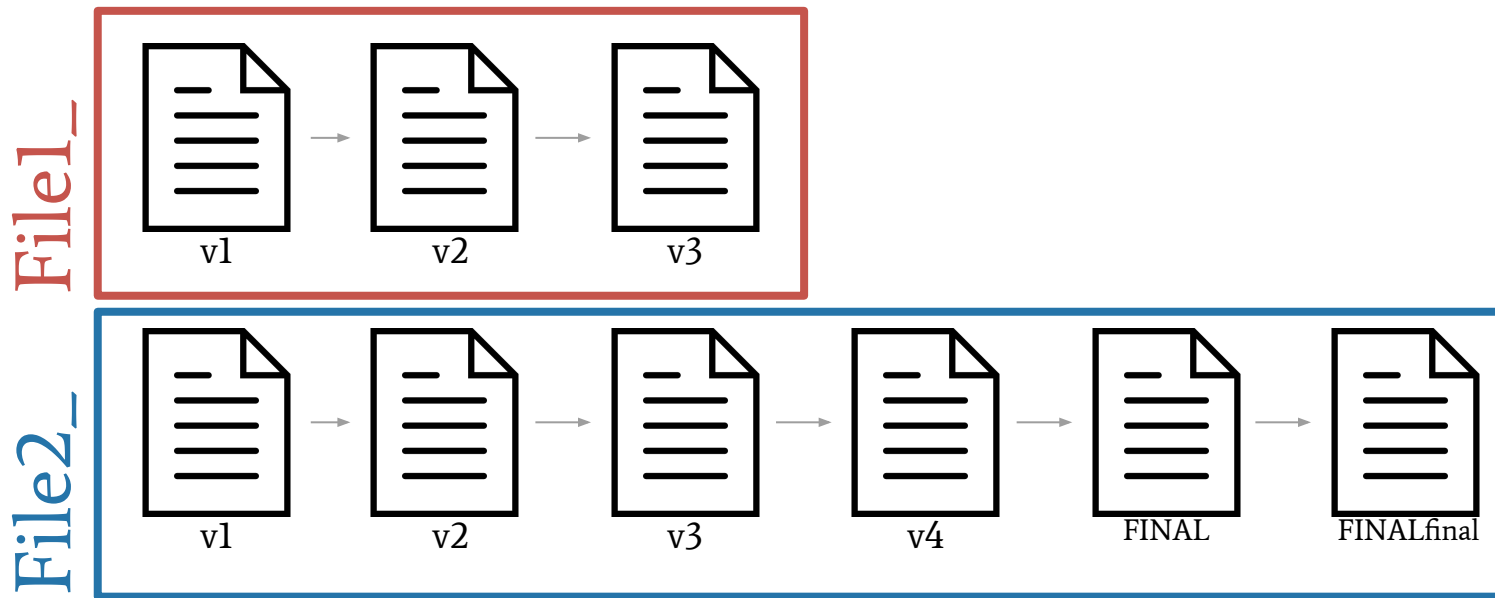
What is version control?

A way to manage the evolution of a set of files



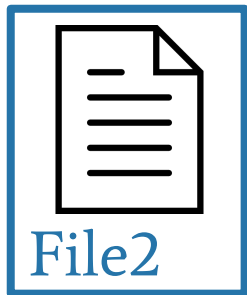
What is version control?

A way to manage the evolution of a set of files



What is version control?

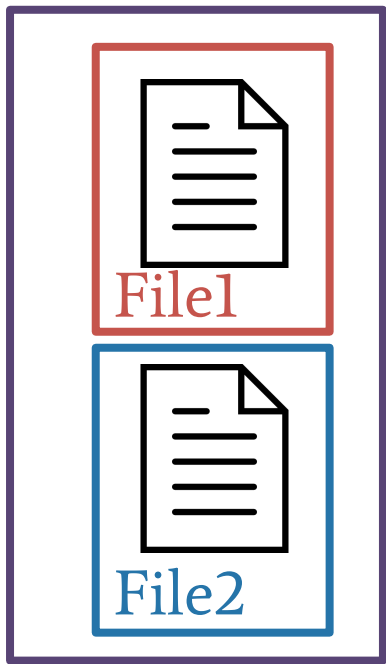
A way to manage the evolution of a set of files



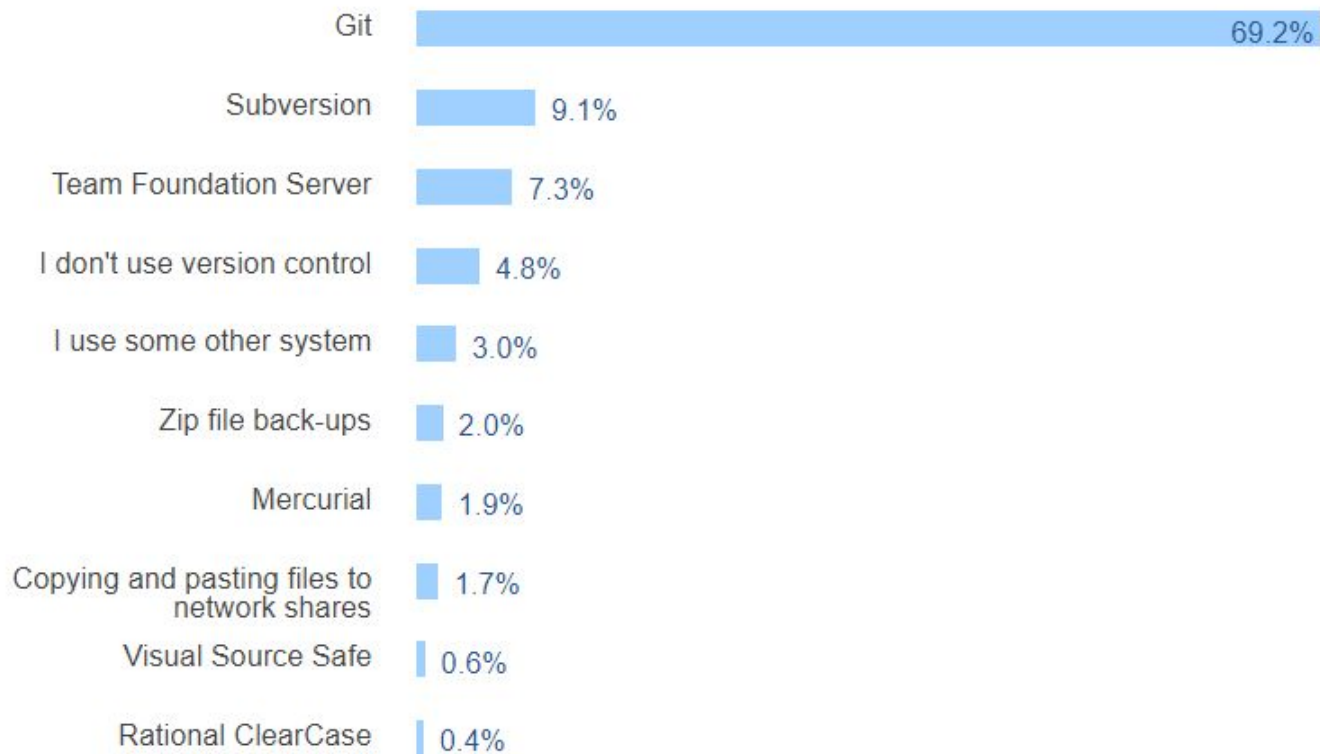
When using a version control system, you have **one copy of each file** and the *version control system tracks the changes* that have occurred over time

What is version control?

A way to manage the evolution of a set of files



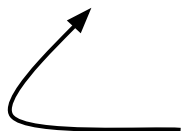
The set of files is referred to as a **repository (repo)**



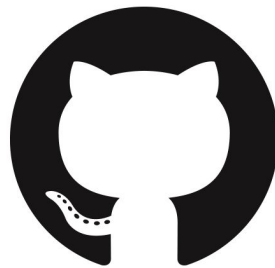
git & GitHub

git

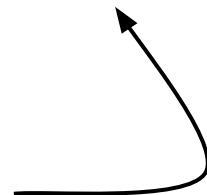
the version control system



~ Track Changes
from Google
Docs....on steroids



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

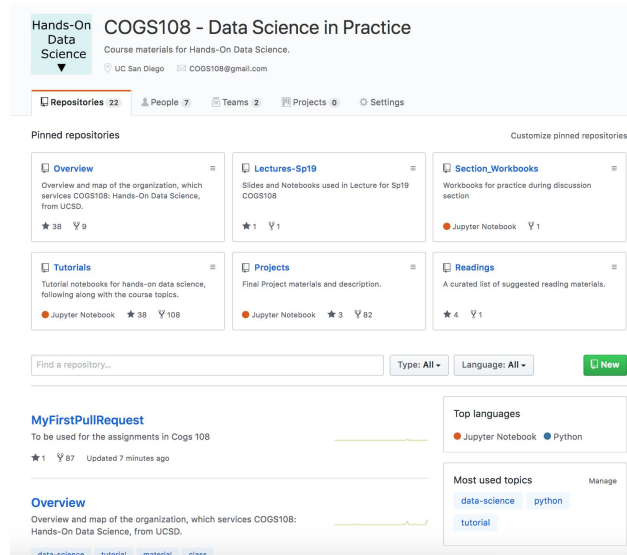


~ Google
Drive...but more

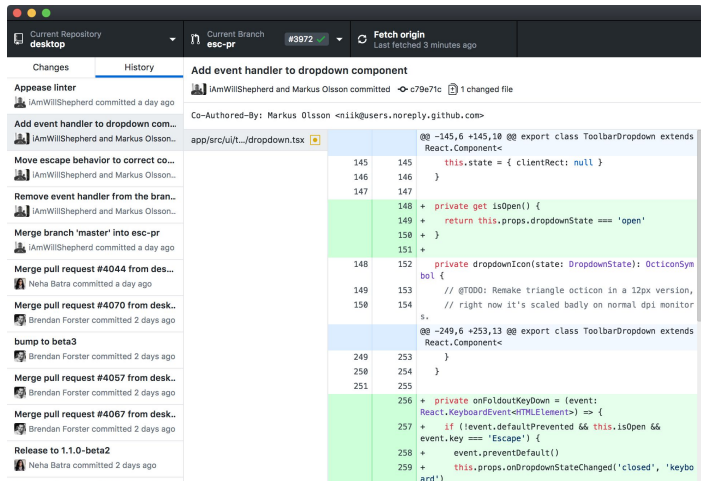
What version control looks like

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

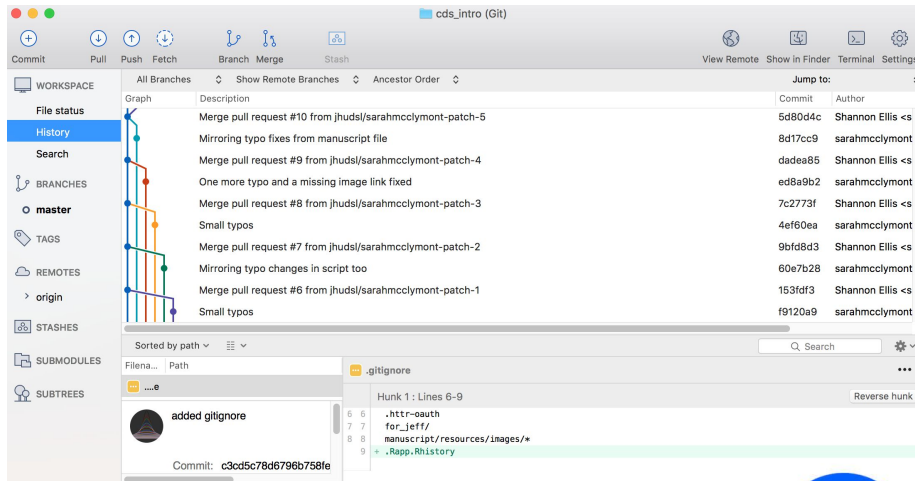
Terminal
git



GUIs can be helpful when working with version control



GitHub Desktop

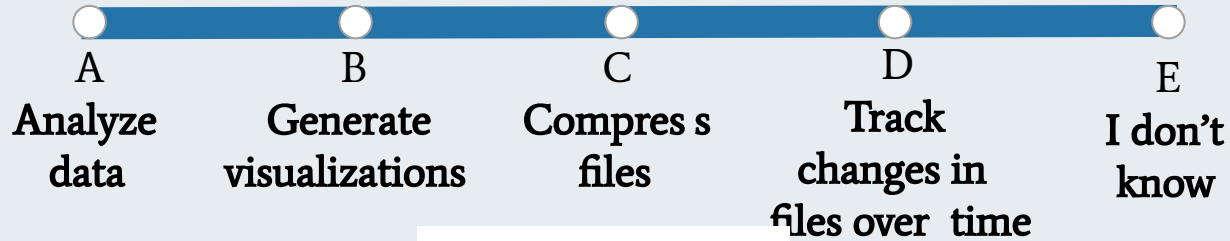


SourceTree



Version Control

git is primarily used to _____.

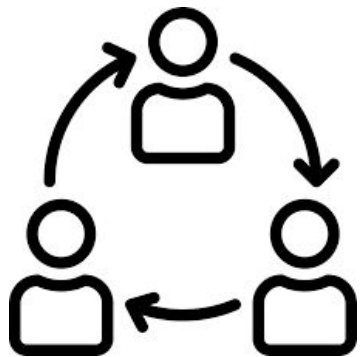


Version Controller

How do you typically interact with git?



Why version control with git and GitHub?



Collaboration



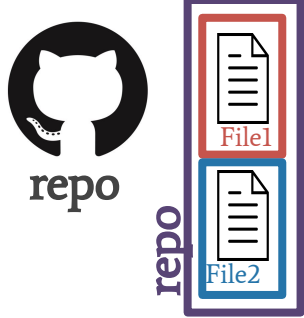
Returning to
a safe state



Exposure for
your work

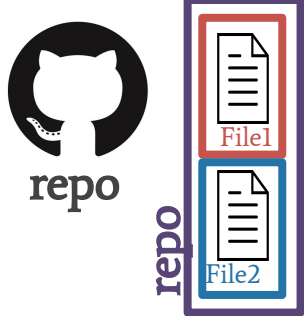


Tracking
others' work



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

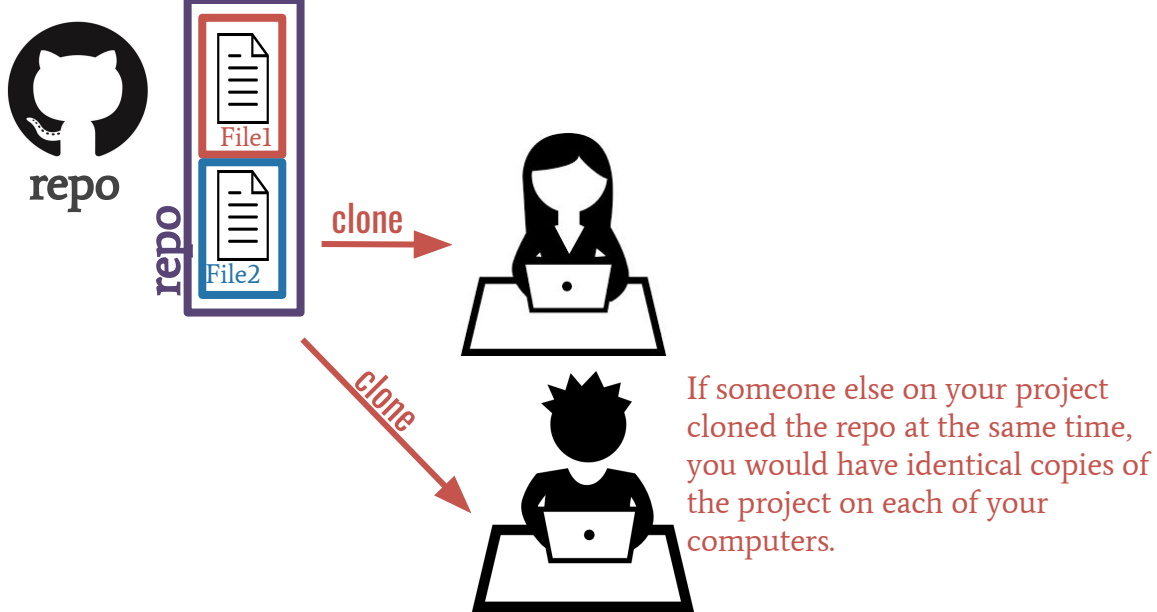
GitHub is a **remote host**. The files are geographically distant from any files on your computer.

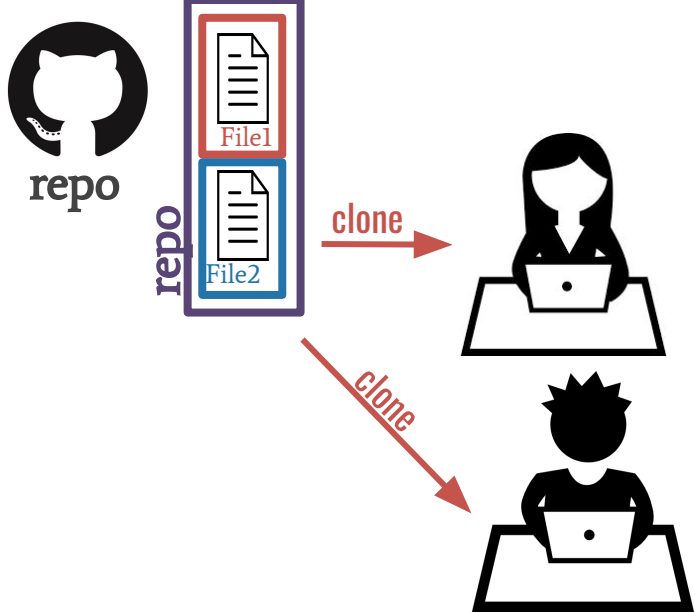


clone →

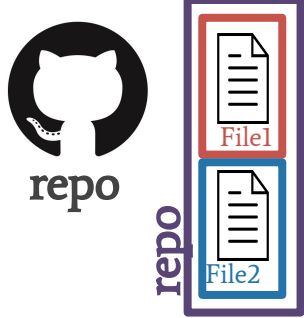


When you first make a copy onto your local computer (read: laptop), you **clone** the repository.

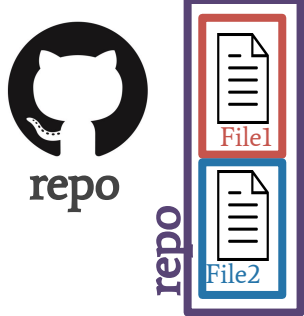




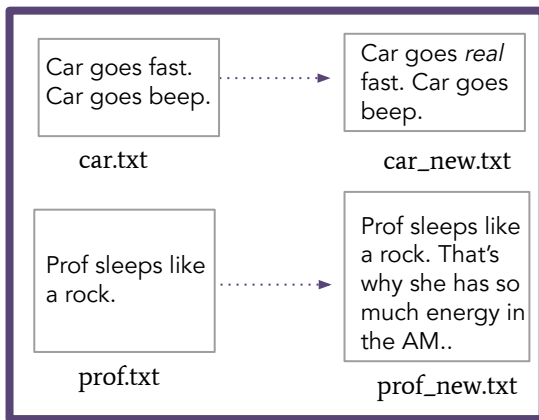
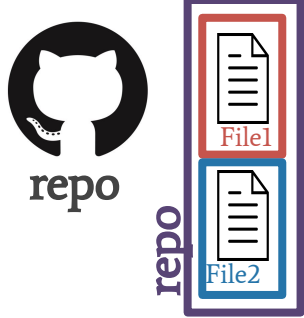
Yay! Everyone can
work on the project!



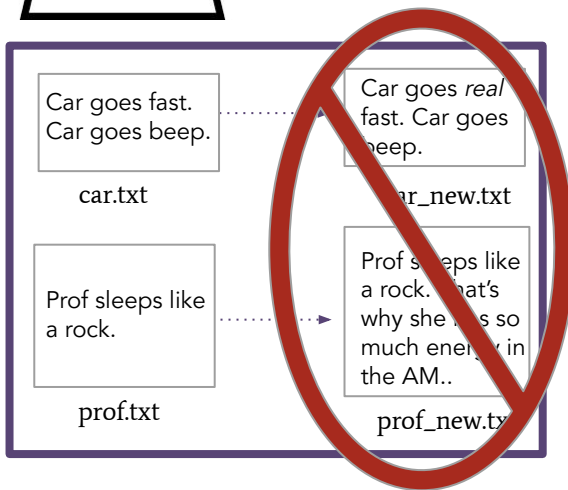
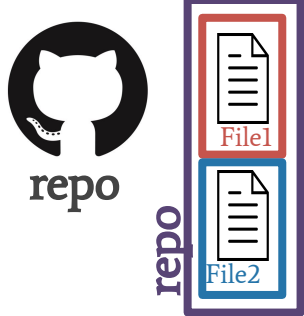
You decide you want to change some of the text in the project.



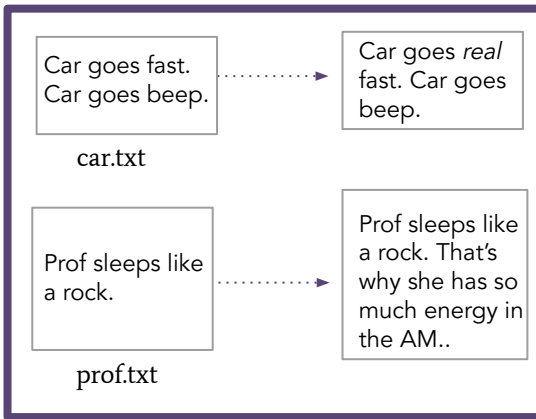
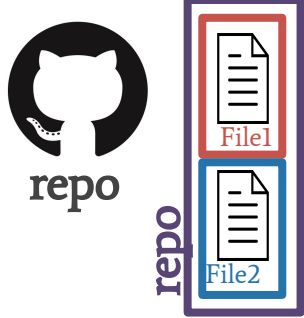
You decide you want to change some of the text in the project.



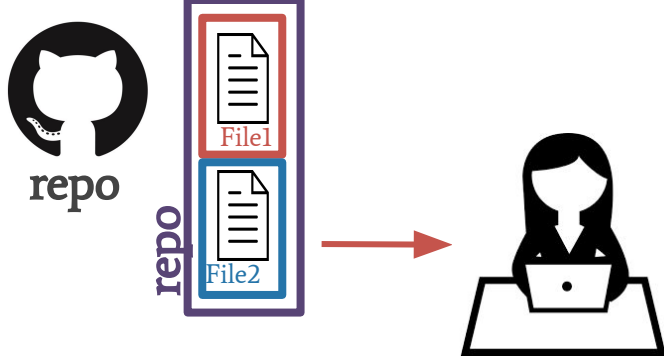
without git...you'd
likely rename
these files....



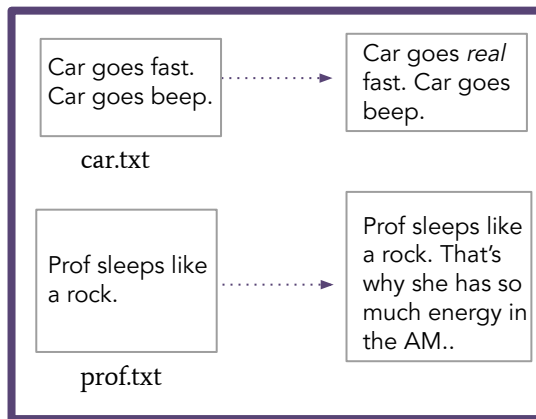
Thank goodness
those days are
over!



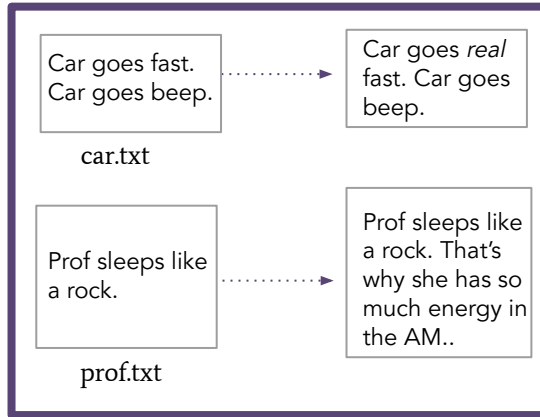
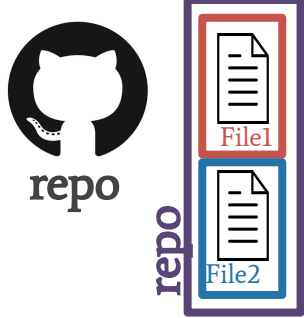
Instead, you tell git which files you'd like to keep track of using **add**. This process is called *staging*.



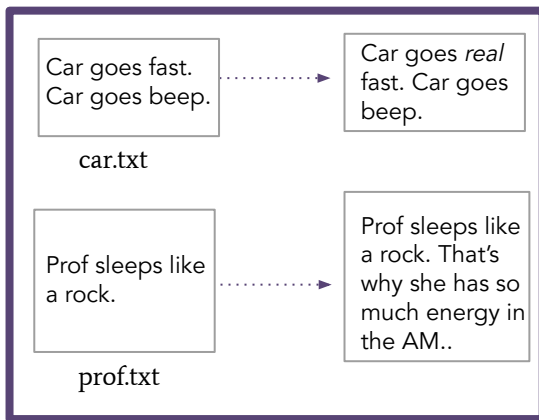
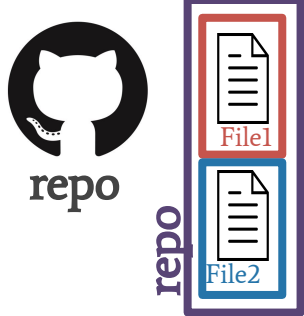
git add file	stages specified file (or folder)
git add .	stages new and modified files
git add -u	stages modified and deleted files
git add -A	stages new, modified, and deleted files
git add *.csv	Stages any files with .csv extension
git add *	Use with caution: stages everything



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



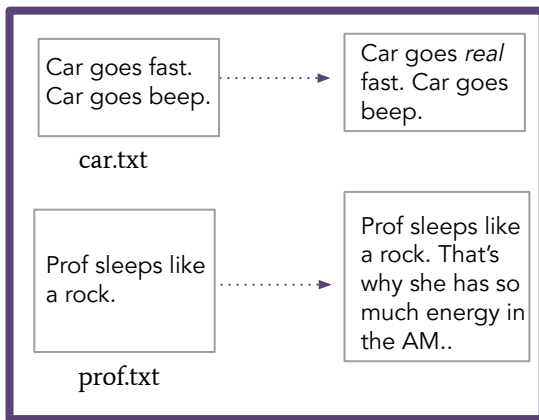
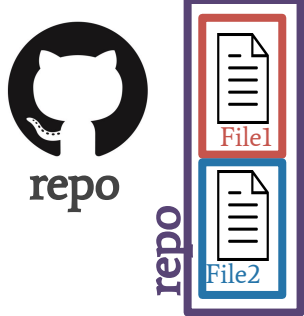
Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.



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



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

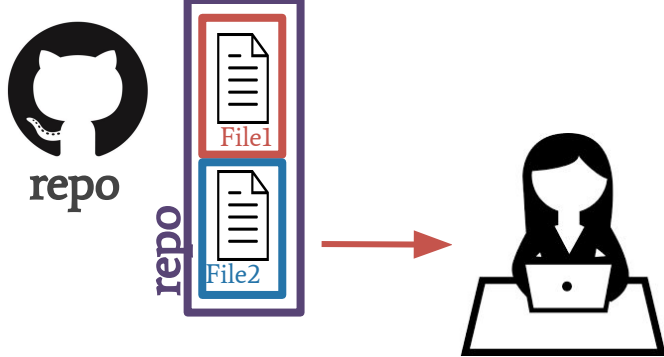


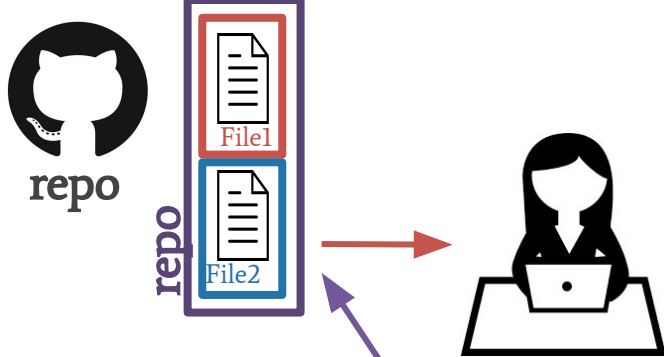
You can make commits more informative by adding a **commit message** .

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** .

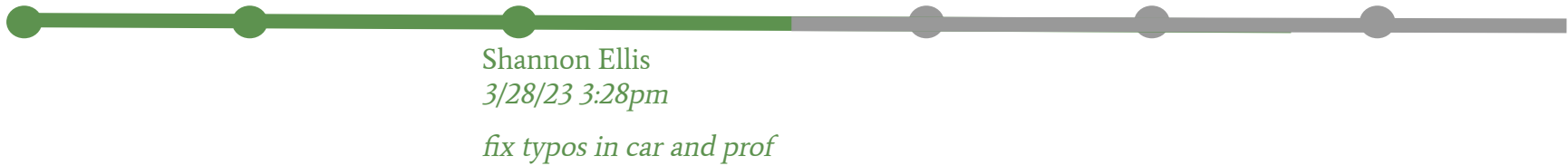
A **commit** tracks who, what, and when

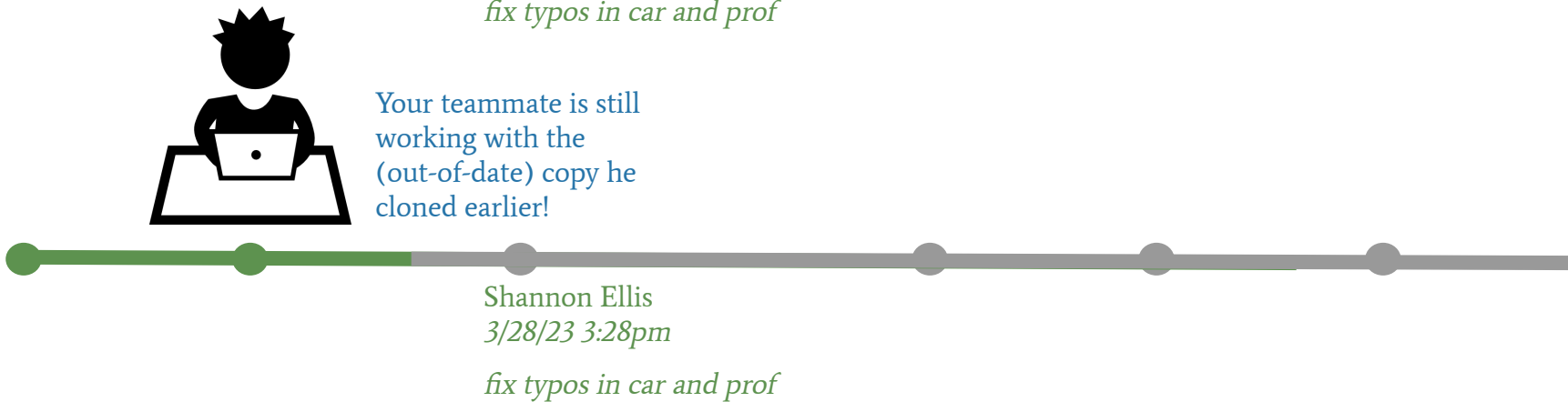
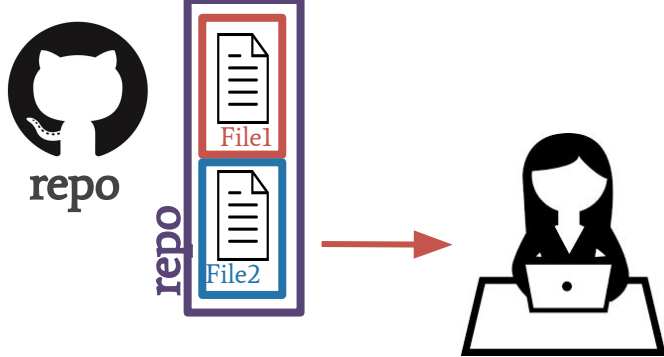


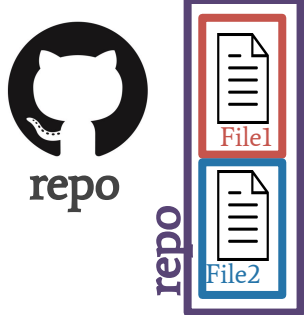


push

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.







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



Shannon Ellis
3/28/23 3:28pm

fix typos in car and prof

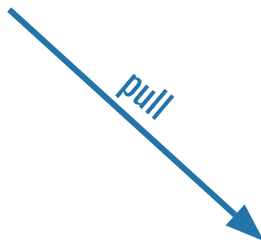
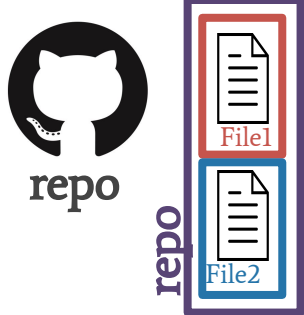


Your teammate is still
working with the
(out-of-date) copy he
cloned earlier!



Shannon Ellis
3/28/23 3:28pm

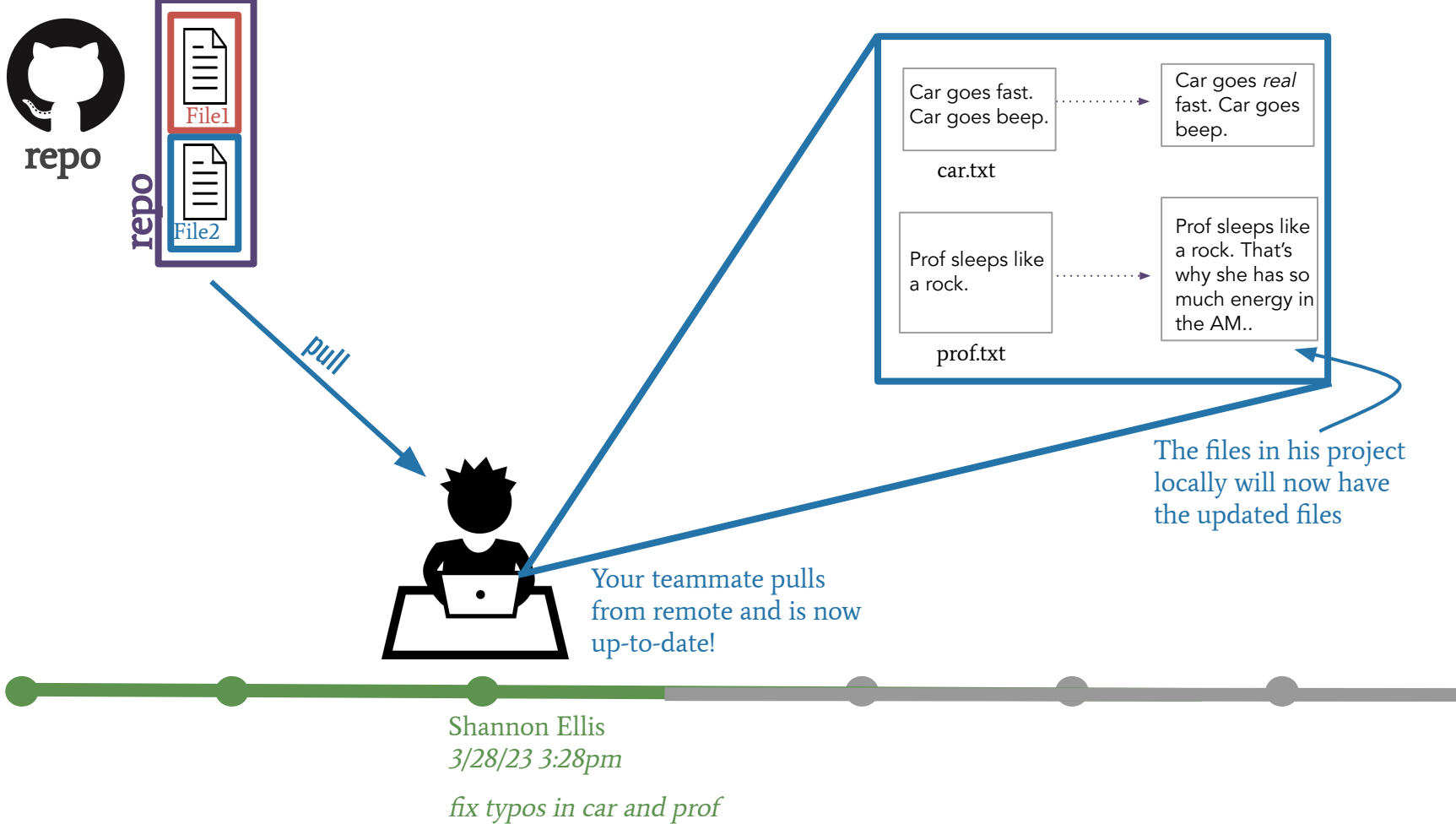
fix typos in car and prof

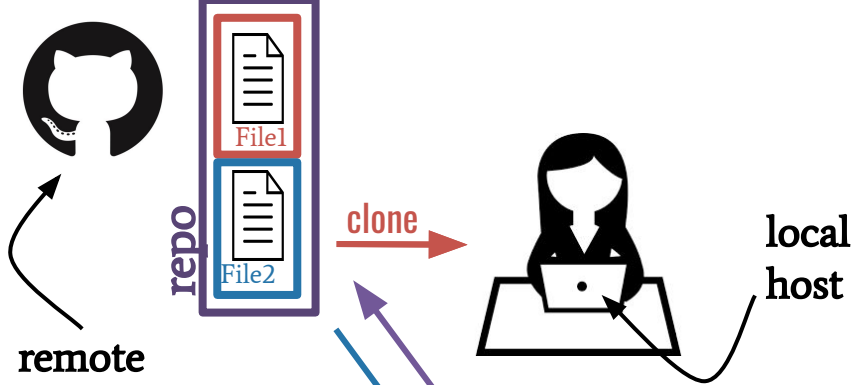


Your teammate pulls
from remote and is now
up-to-date!

Shannon Ellis
3/28/23 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
```

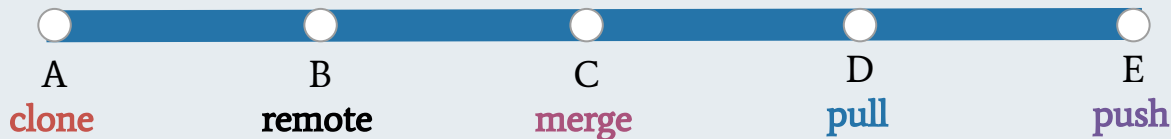
Review & Question Time

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?

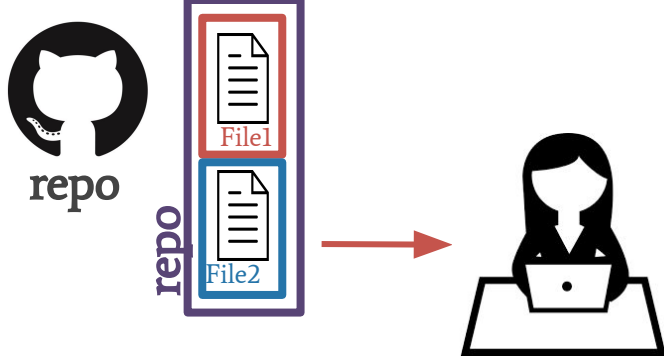


Version Controller II

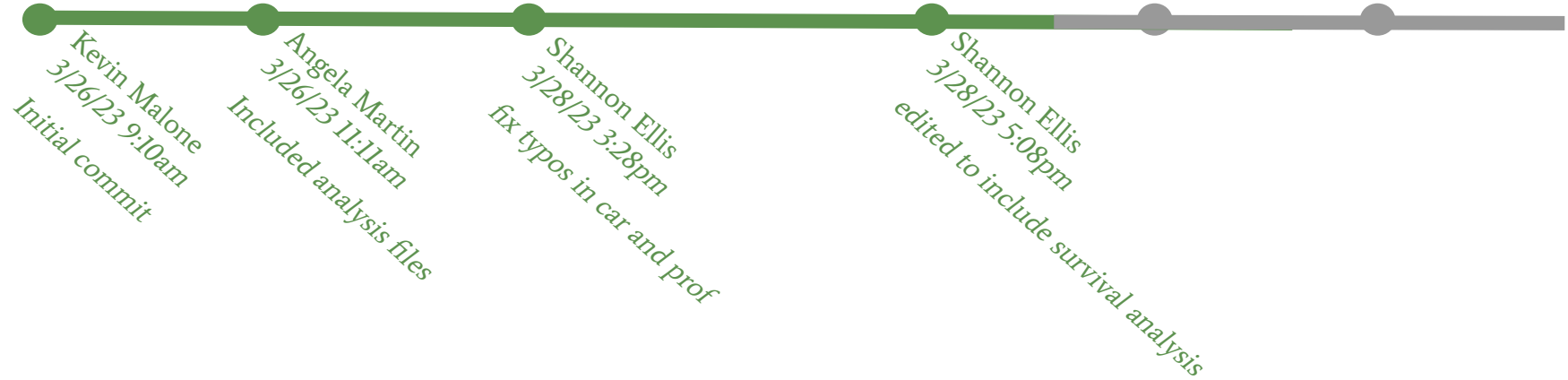
Your teammate has given you access to a GitHub repository to work on a project together. You want to get them for the first time on your computer locally.

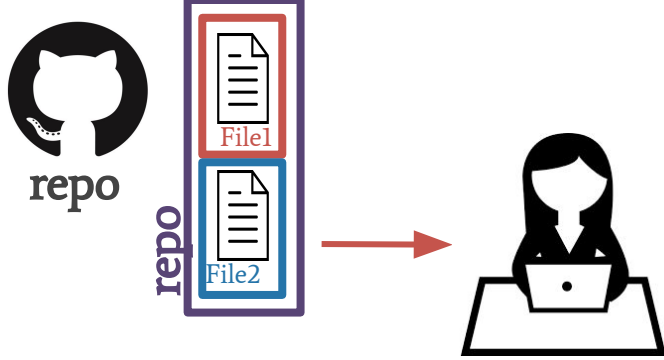
What do you do to get the repo on your computer?





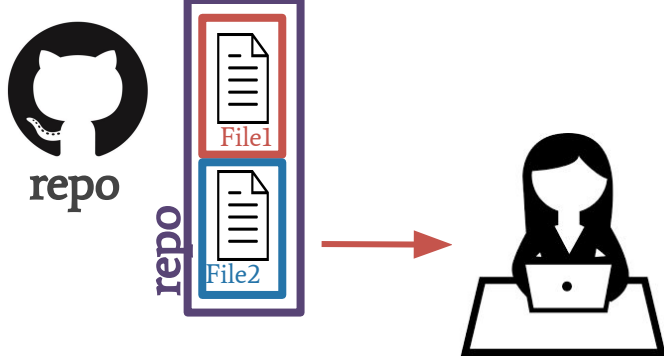
Each time you create a commit, git tracks the changes made automatically.



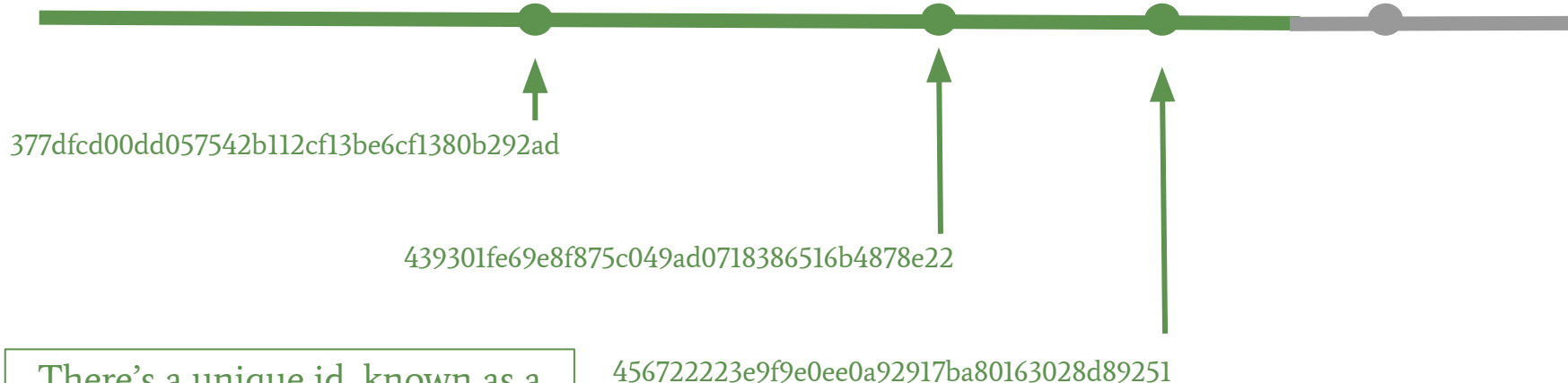


By committing each time you make changes, git allows you to time travel!

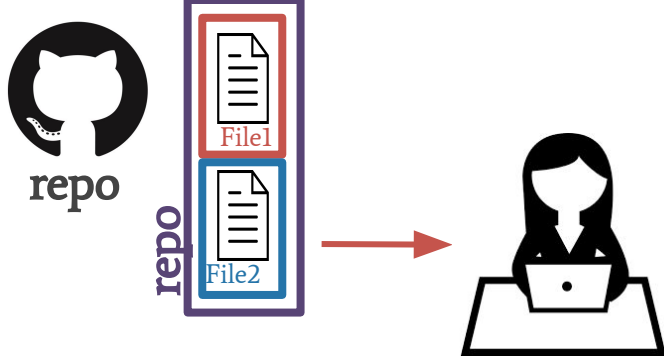




By committing each time you make changes, git allows you to time travel!

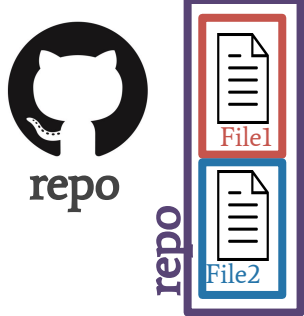


There's a unique id, known as a **hash**, associated with each commit.

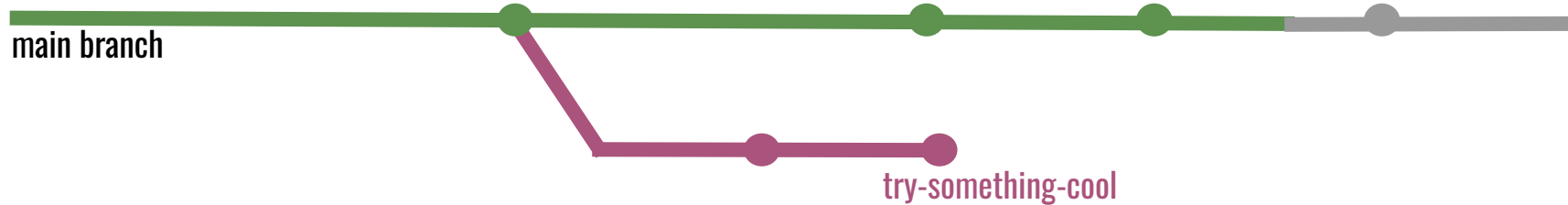


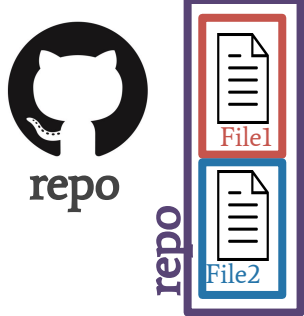
You can return to the state of the repository at any commit. Future commits don't disappear. They just aren't visible when you **check out** an older commit.



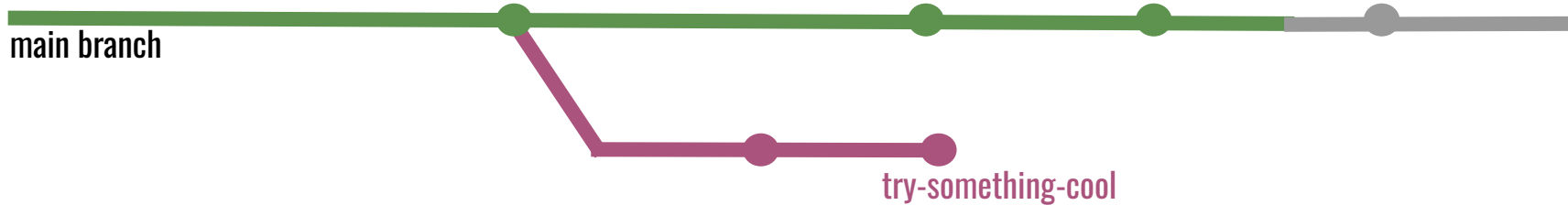


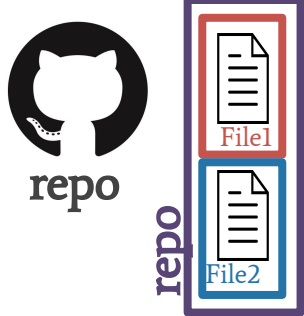
But...not everything is always linear. Sometimes you want to try something out and you're not sure it's going to work. This is where you'll want to use a **branch**.



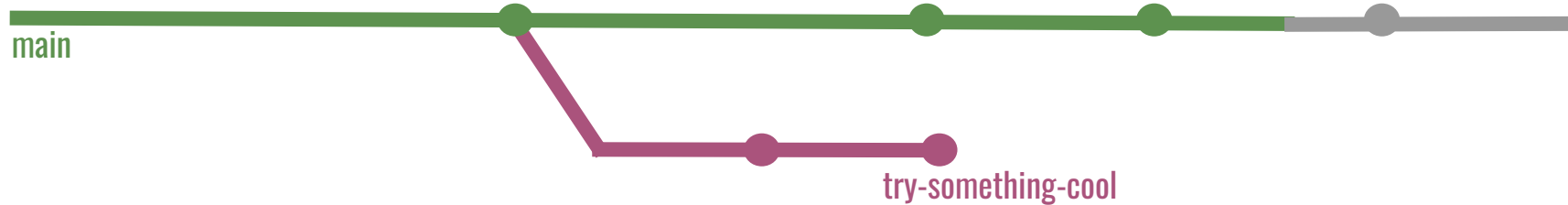


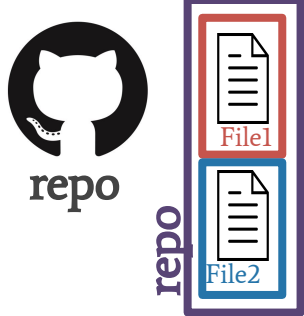
It's a good way to experiment. It's pretty easy to get rid of a branch later on should you not want to include the commits on that branch.



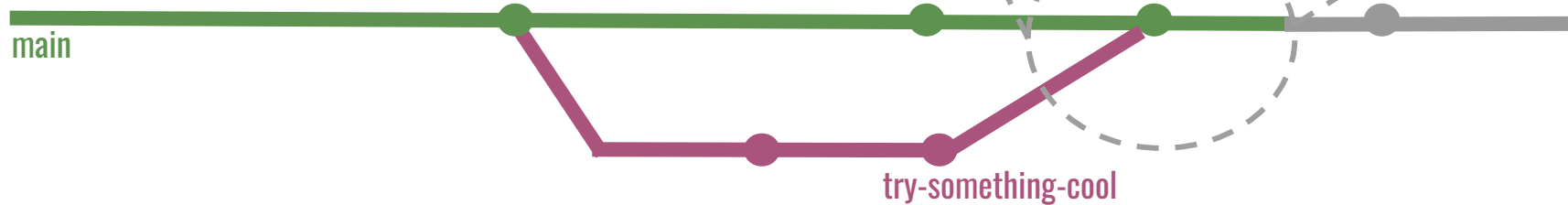


But...what if you DO want to include the changes you've made on your **try-something-cool** branch into the **main** branch?





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



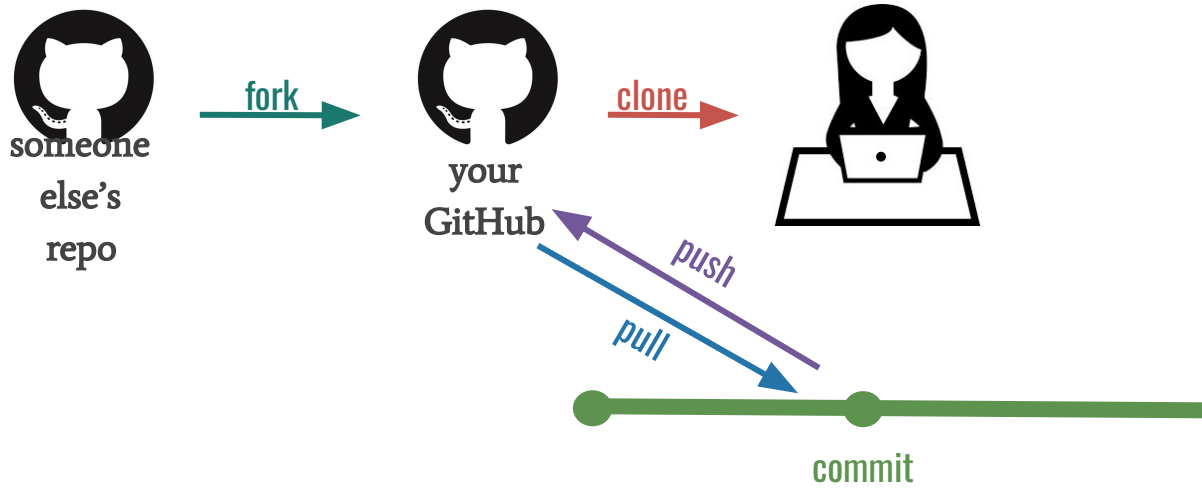


someone
else's
repo

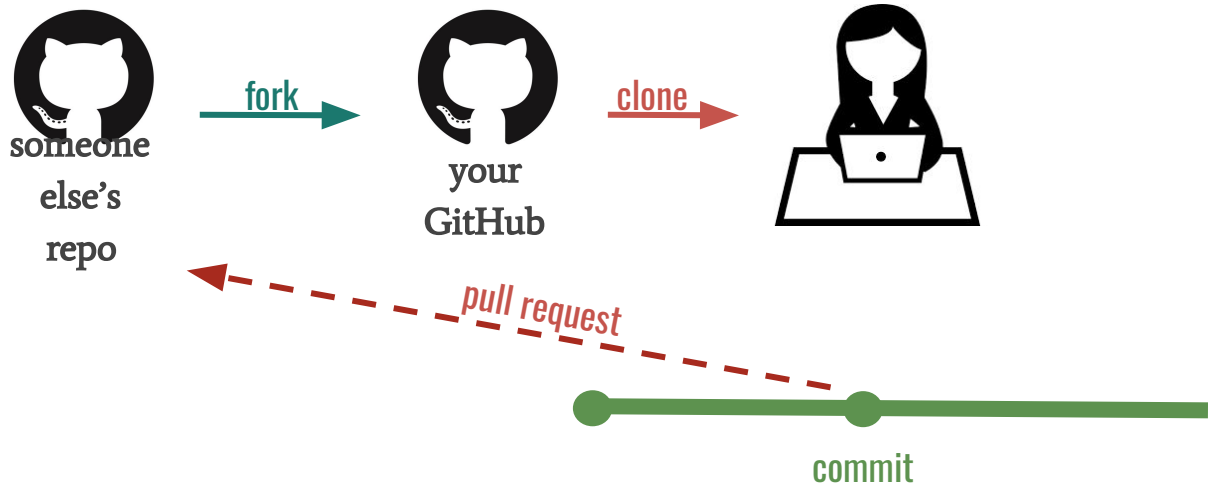


your
GitHub

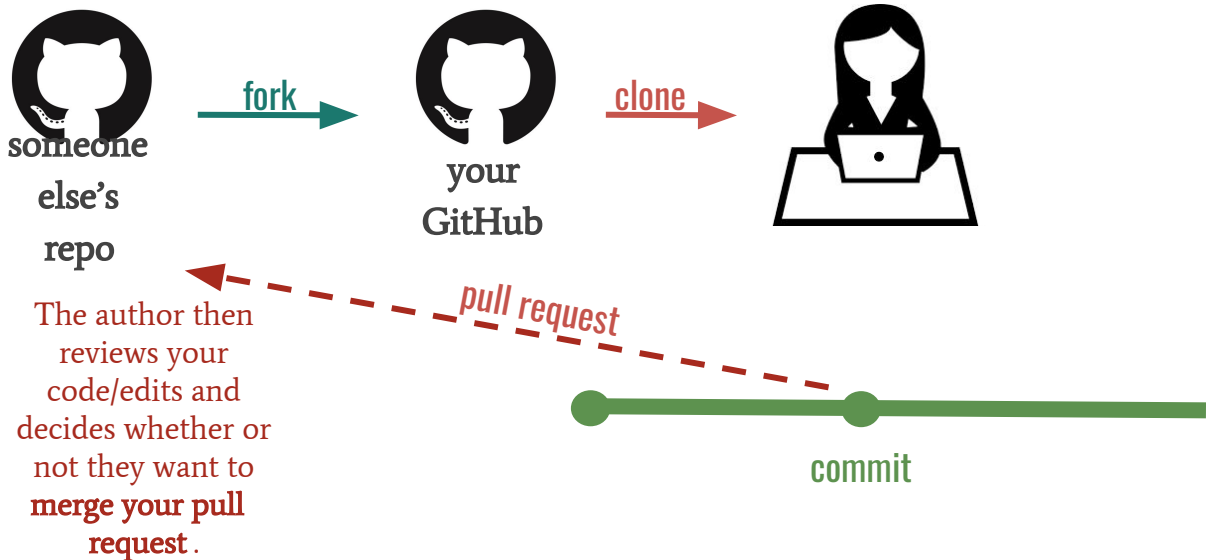
What if someone else is working on something cool and you want to play around with it? You'll have to **fork** their repo.



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 addressed ...or if the software maintainer doesn't like the suggestion



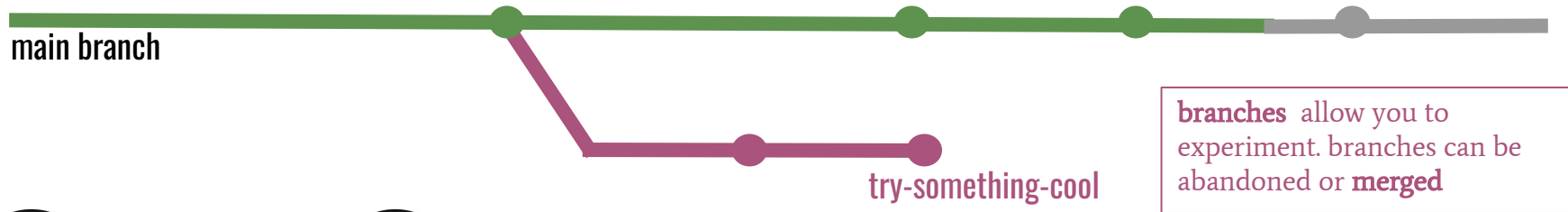
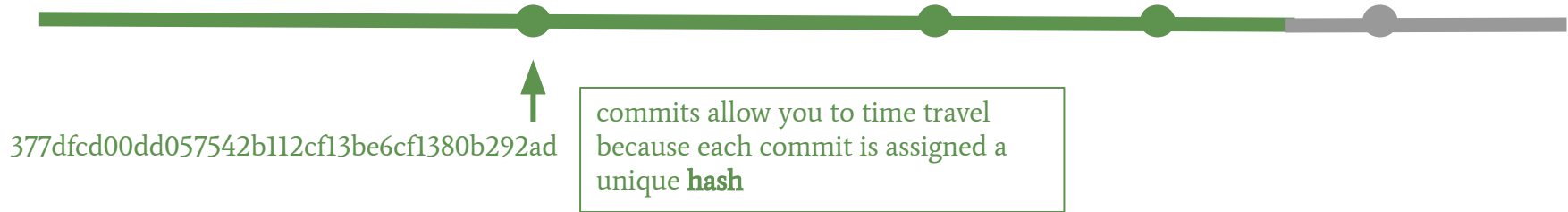
377dfcd00dd057542b112cf13be6cf1380b292ad

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

One more git recap...



One more git recap...

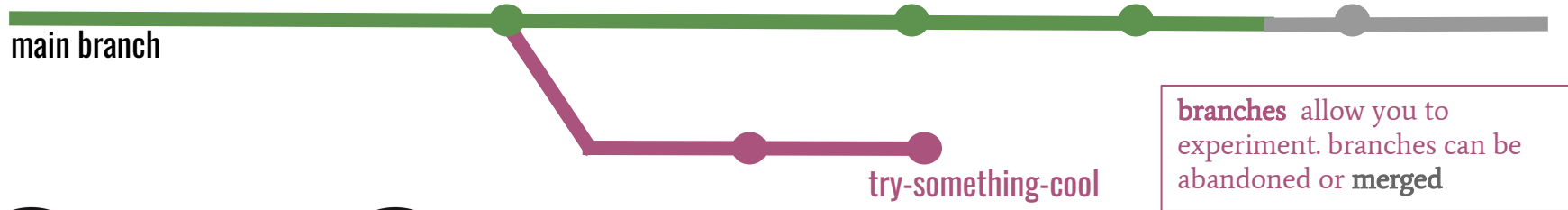
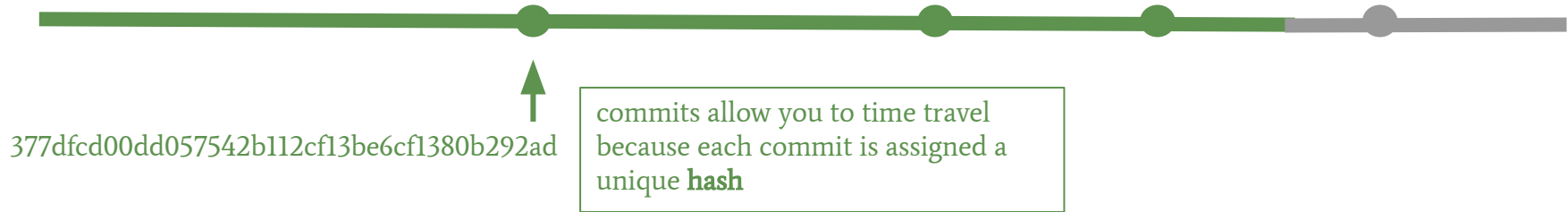


fork



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

One more git recap...



fork



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

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...

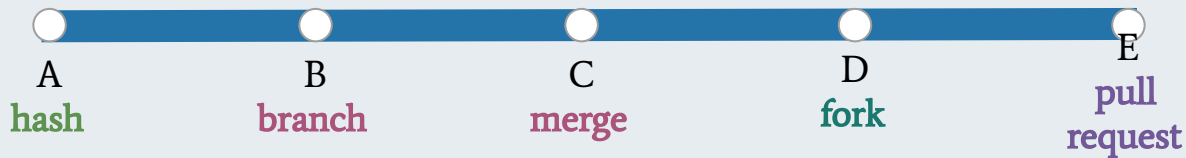
Review & Question Time

Version Controller III



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

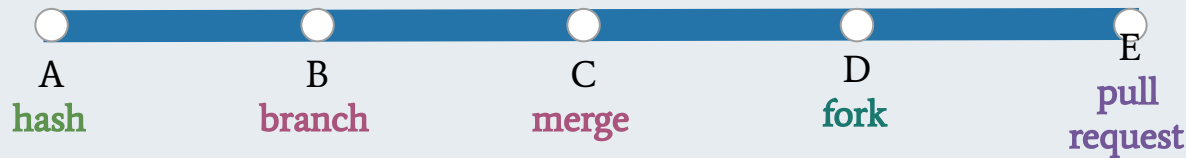
what should you do?



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?



GH Authentication

GitHub PAT (Personal Access Token)

- More secure
- Unique per person *or* per device
- You can have lots of them (different PATs for different roles in different projects)

Scott Yang (former TA) How To: [Link to Doc](#)

GitHub SSH Key (passwordless login)

Detailed instructions + video from COGS 137:

<https://cogs137-fa24.github.io/cogs137-fa24/content/labs/01-lab-intro-r.html#github-housekeeping>

Good for:

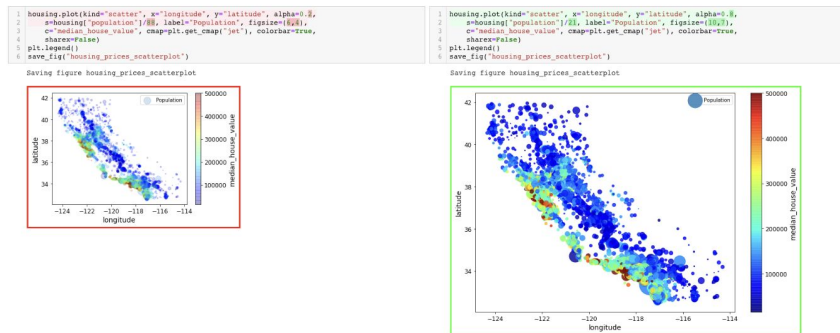
- Automated tasks
- Frequent Access

Version controlling notebooks sucks rocks

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

ReviewNB

ReviewNB is a GitHub app that also offers visual diffing with an interface that looks similar to the traditional Jupyter IDE. Because the outputs are visualized, problems associated with committing binary blobs disappear.



ReviewNB example courtesy of the [ReviewNB website](#)

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.

Learn by doing...

<https://github.com/COGS108/Resources/blob/main/README.md#version-control-git-and-github>

- If you use any of these and have feedback (good, bad, otherwise) please lmk
- If you find other resources to add, feel free to suggest (maybe a PR?!)