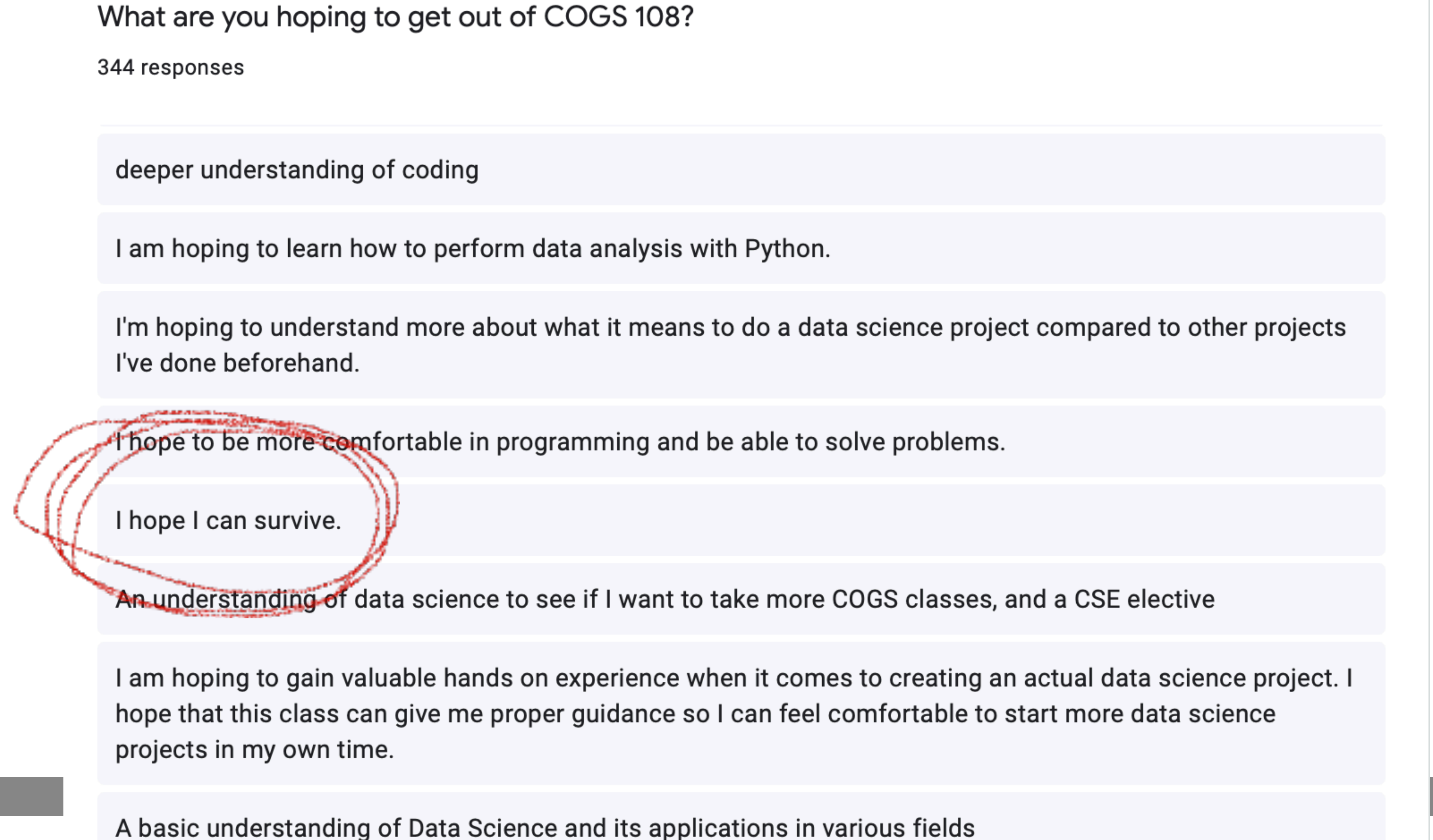
Course Announcements

- Due Today (Monday)
    - Q1 (Canvas quiz)
- Due Friday
    - D1 (discussion lab)
    - A1 (Assignment)
    - Group submission (1 form/group)

Surveys - thank you! (N=459)

- 90% have reliable internet
- 71% comfortable in Python; 9.4% not comfortable in any language
- Comfort in stats: 5.8/10
- Comfort in programming: 6.8/10
- Common hobbies: TV/movies/YouTube, video games, photography, walks/hikes, outdoor activities/hiking, music, journaling, etc.



What are you hoping to get out of COGS 108?

344 responses

deeper understanding of coding

I am hoping to learn how to perform data analysis with Python.

I'm hoping to understand more about what it means to do a data science project compared to other projects I've done beforehand.

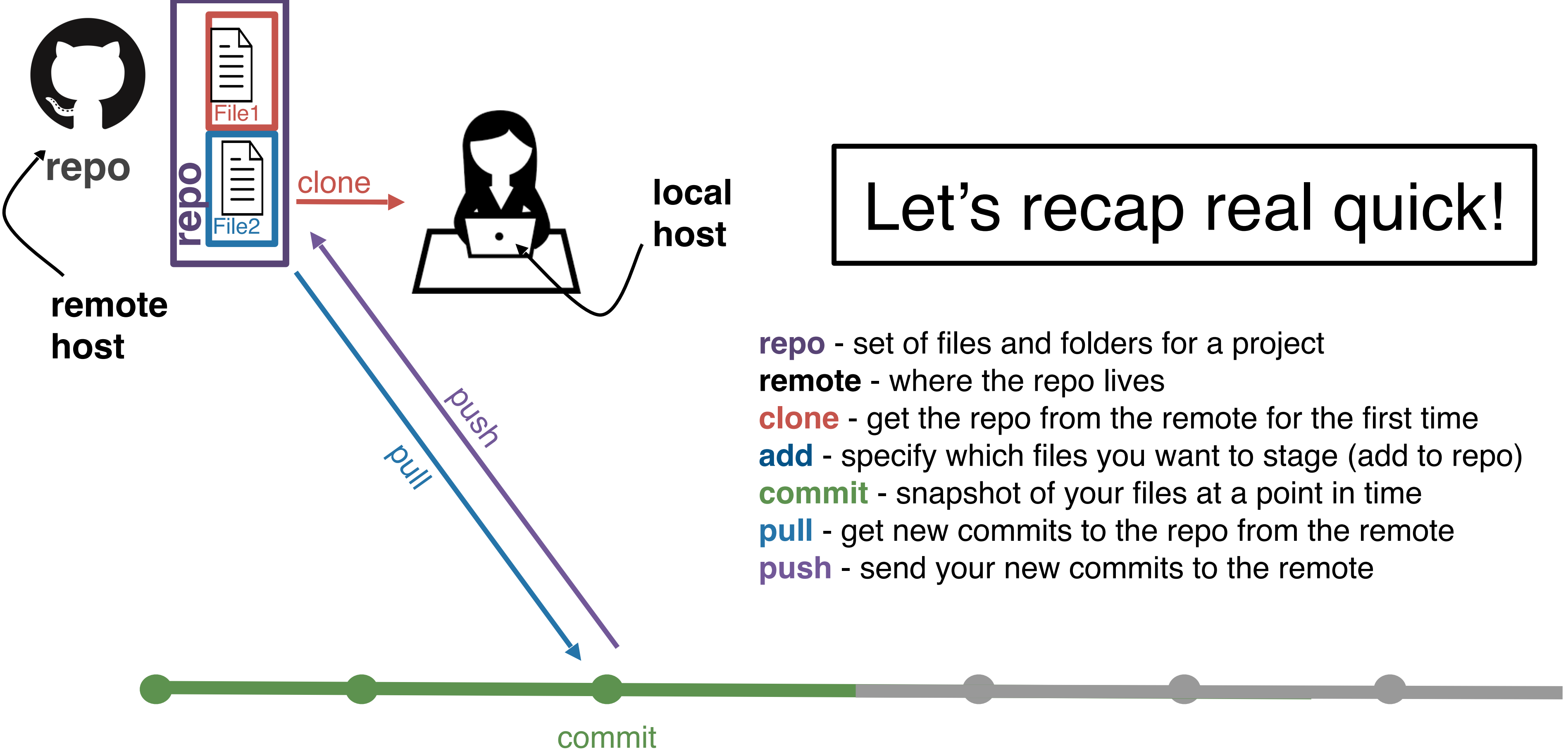
I hope to be more comfortable in programming and be able to solve problems.

I hope I can survive.

An understanding of data science to see if I want to take more COGS classes, and a CSE elective

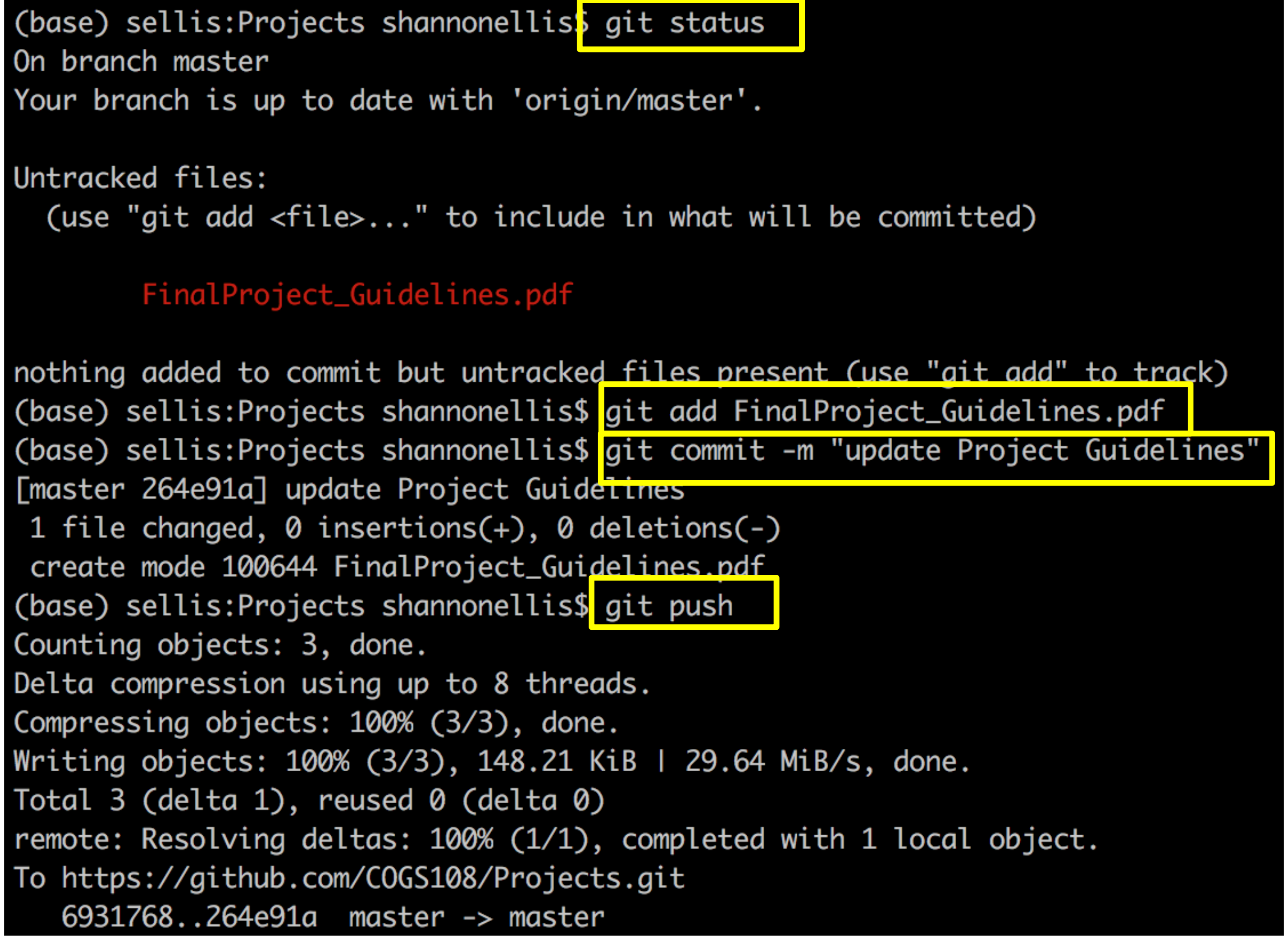
I am hoping to gain valuable hands on experience when it comes to creating an actual data science project. I hope that this class can give me proper guidance so I can feel comfortable to start more data science projects in my own time.

A basic understanding of Data Science and its applications in various fields



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

https://forms.gle/wHA2GSyuycFre5qr6

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?

A	B	C	D	E
clone	**remote**	**merge**	**pull**	**push**

Version Controller II

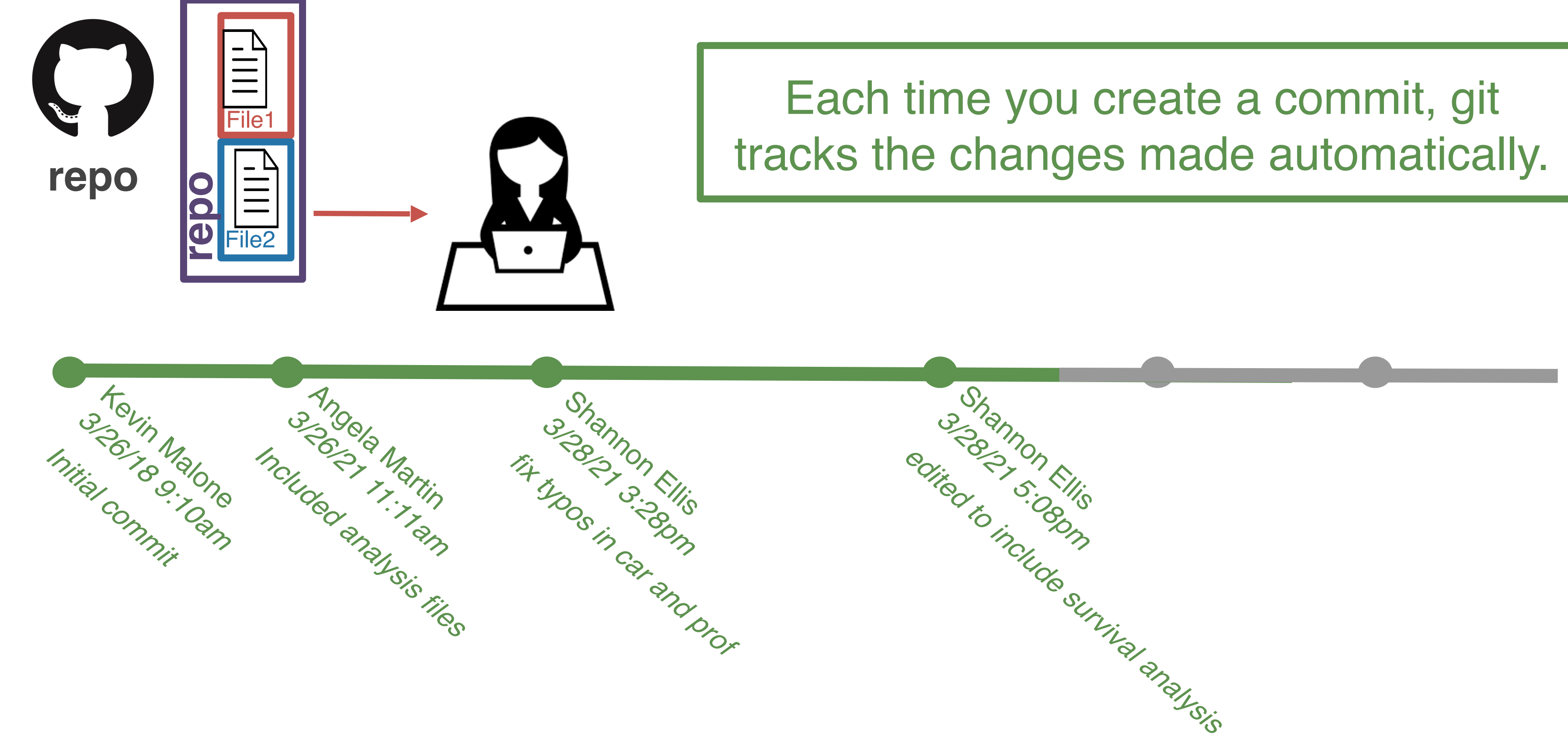
https://forms.gle/wHA2GSyuycFre5qr6

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?

A	B	C	D	E
clone	**remote**	**commit**	**pull**	**push**

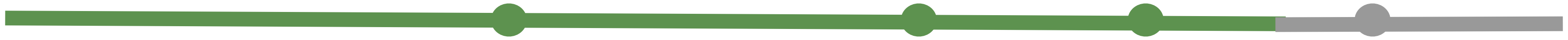
Each time you create a commit, git tracks the changes made automatically.



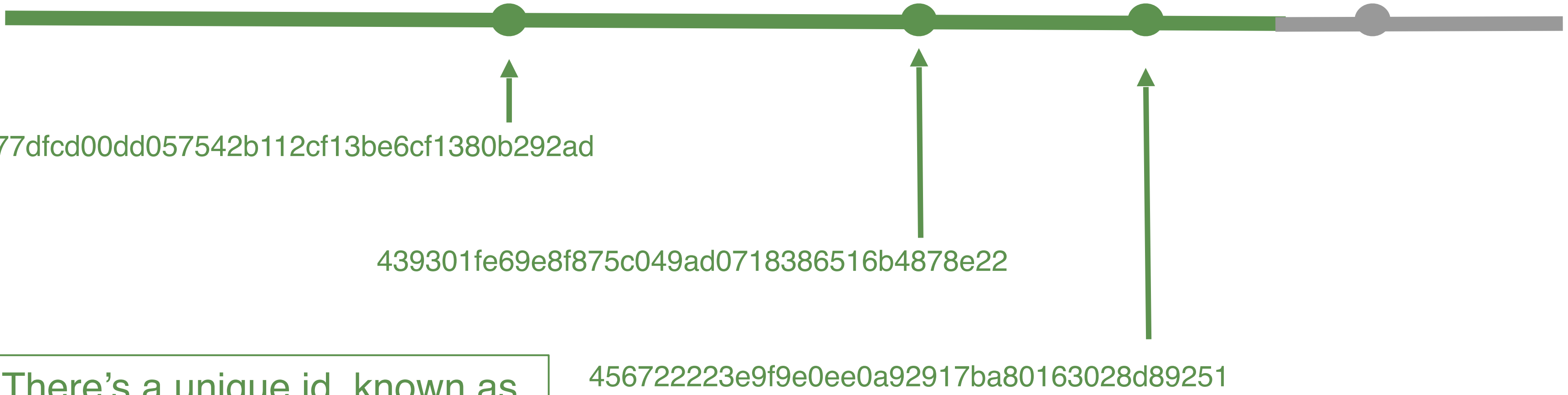
repo
File1
File2
repo
Kevin Malone 3/26/18 9:10am Initial commit
Angela Martin 3/26/21 11:11am Included analysis files
Shannon Ellis 3/28/21 3:28pm fix typos in car and prof
Shannon Ellis 3/28/21 5:08pm edited to include survival analysis

repo
repo
File1
File2

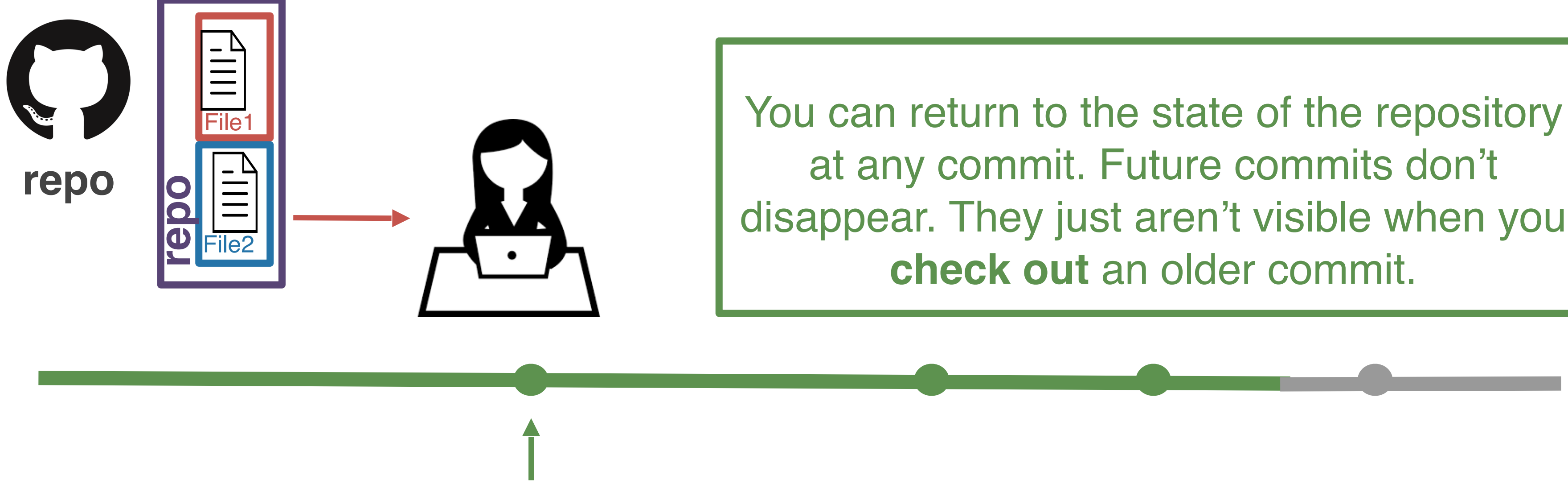
By committing each time you make changes, git allows you to time travel!



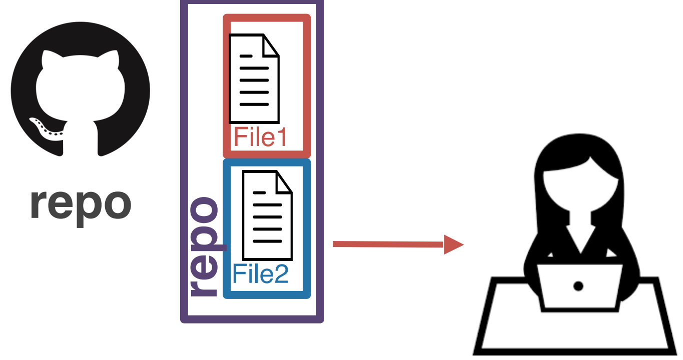
By committing each time you make changes, git allows you to time travel!


377dfcd00dd057542b112cf13be6cf1380b292ad
439301fe69e8f875c049ad0718386516b4878e22
456722223e9f9e0ee0a92917ba80163028d89251

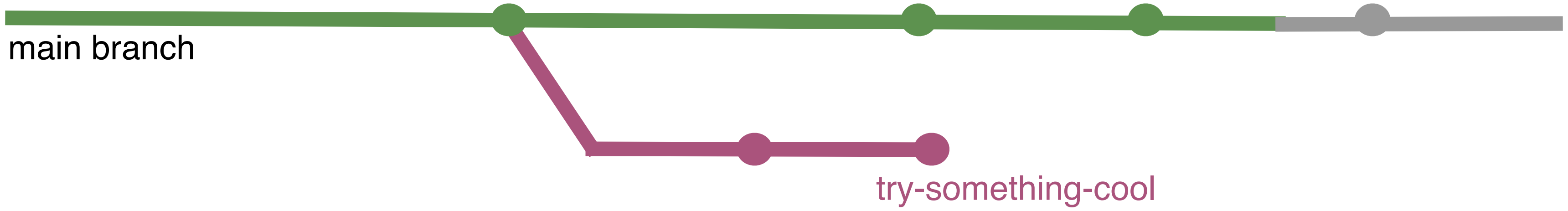
There’s a unique id, known as a **hash**, associated with each commit.

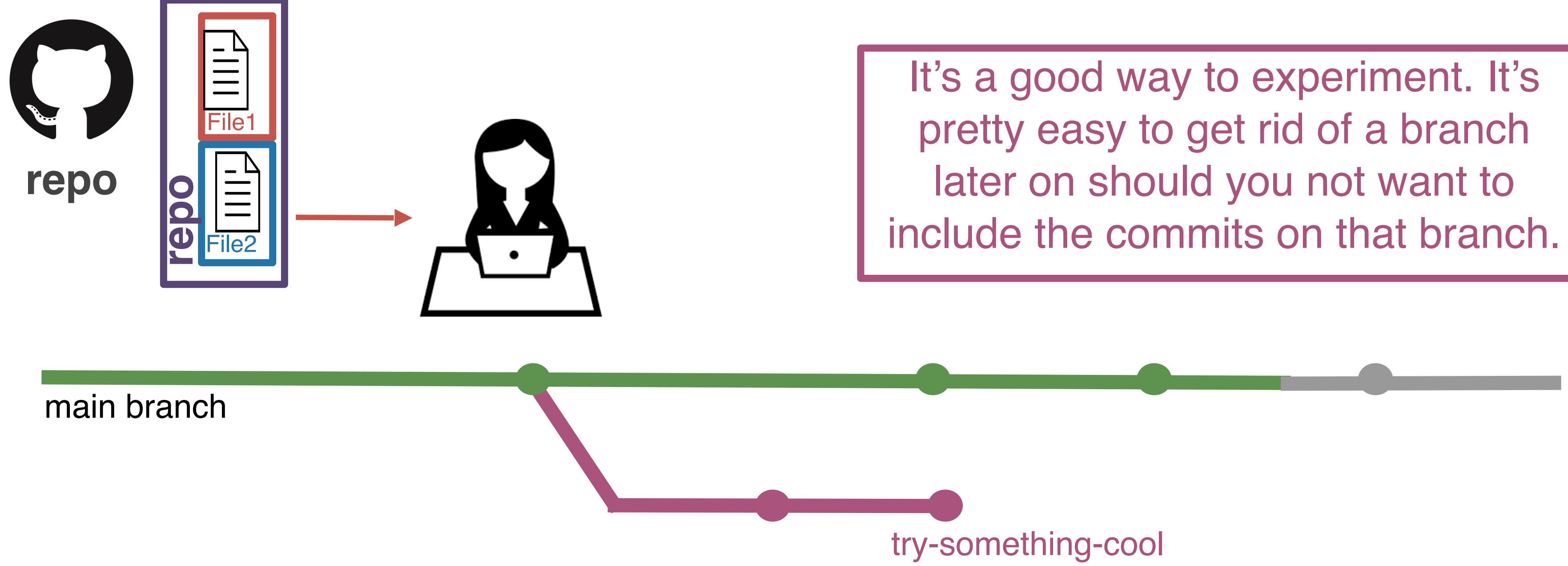


You can return to the state of the repository at any commit. Future commits don’t disappear. They just aren’t visible when you **check out** an older commit.

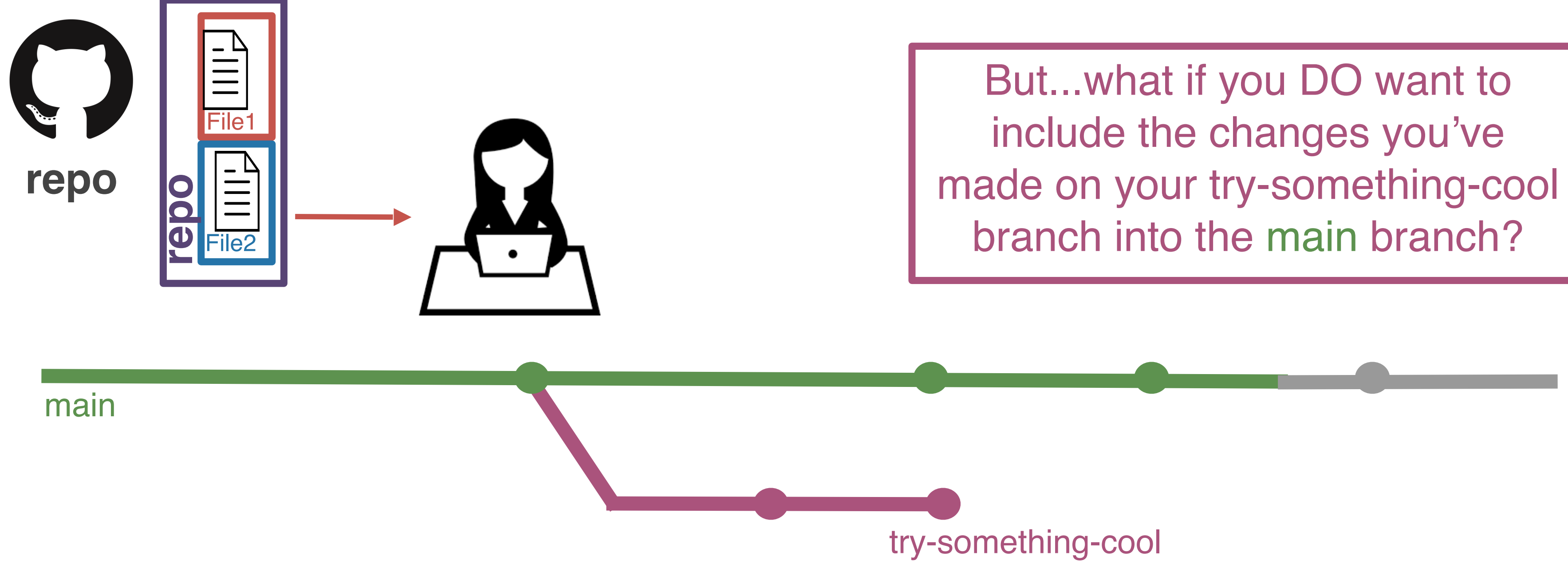


But...not everything is always linear. Sometimes you want to try something out and you're not sure it's going to work. This is where you'll want to use a **branch**.



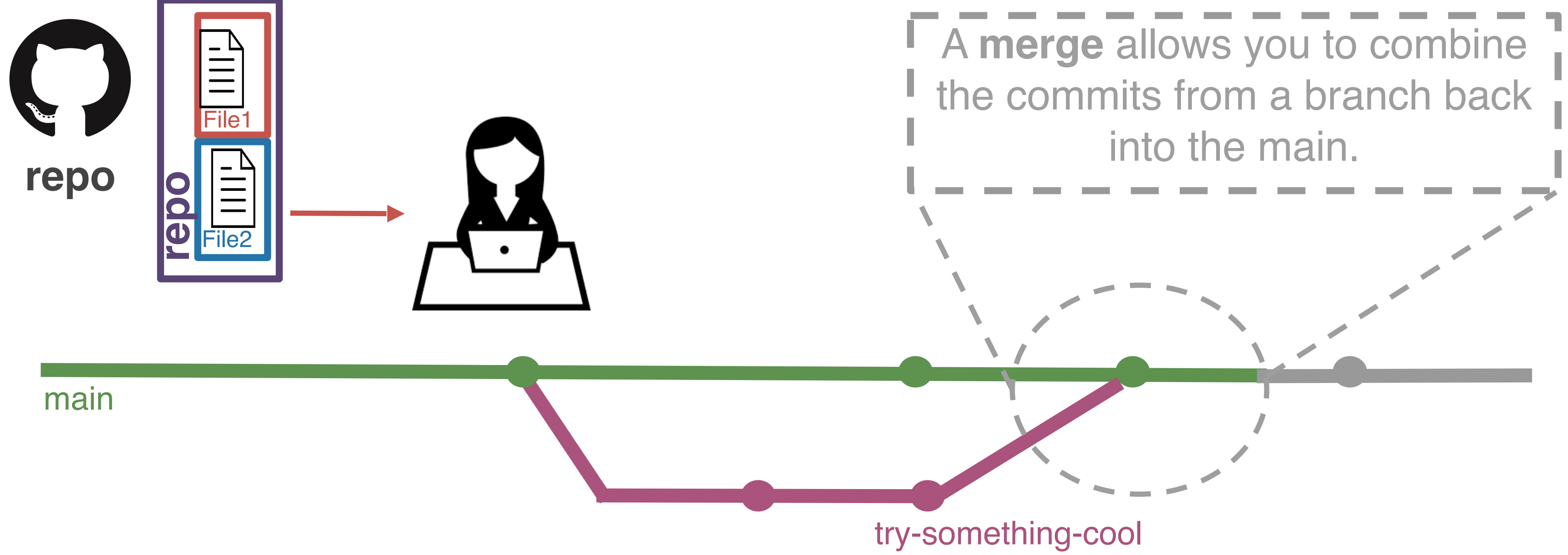


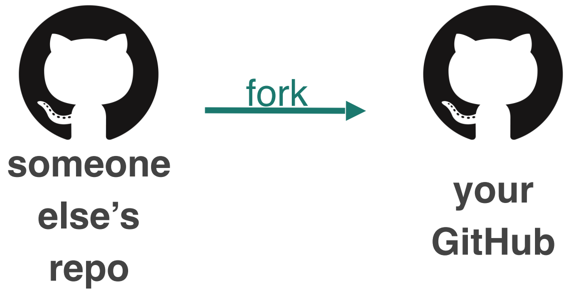
It's a good way to experiment. It's pretty easy to get rid of a branch later on should you not want to include the commits on that branch.



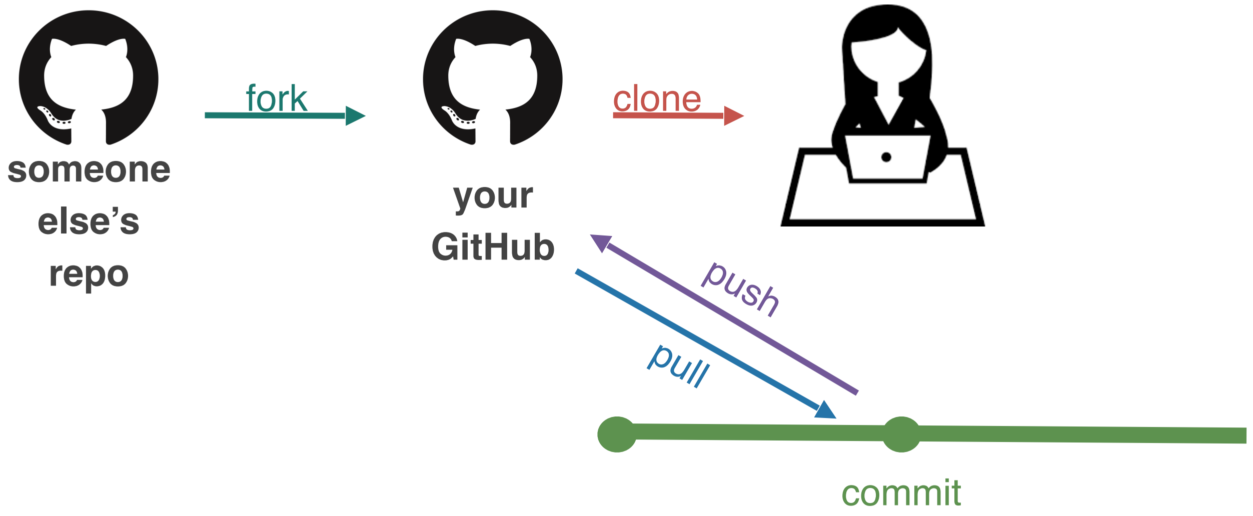
But...what if you DO want to include the changes you've made on your try-something-cool branch into the main branch?

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

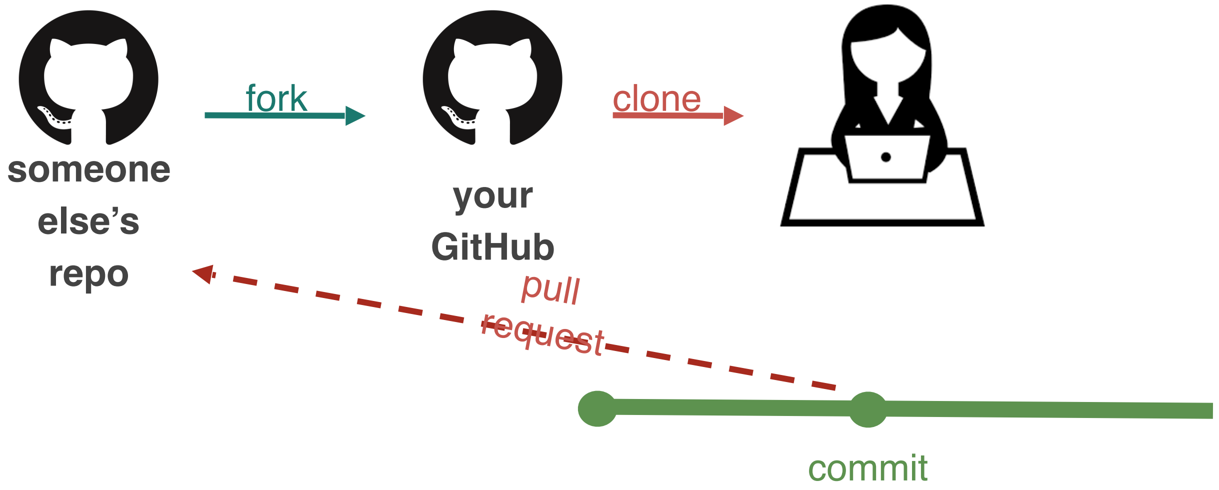




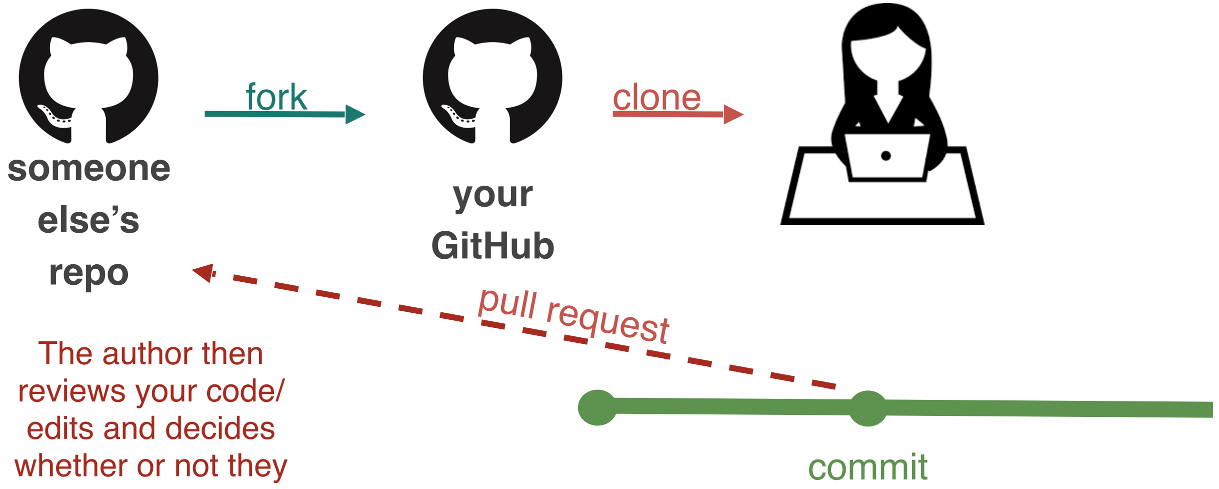
What if someone else is working on something cool and you want to play around with it? You'll have to **fork** their repo.



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**).



The author then reviews your code/ edits and decides whether or not they want to **merge your pull request**.

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.

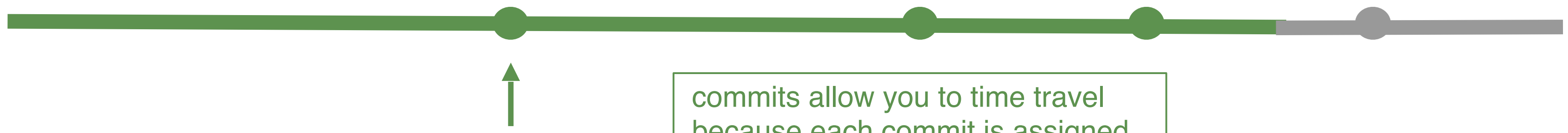
someone else's repo

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!

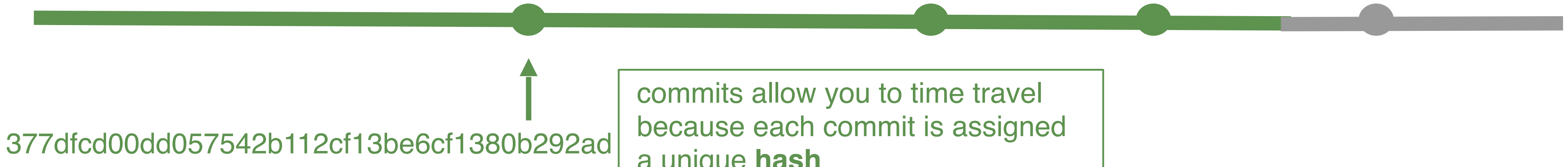
They can be assigned to people.

They can be closed once addressed ….or if the software maintainer doesn't like the suggestion

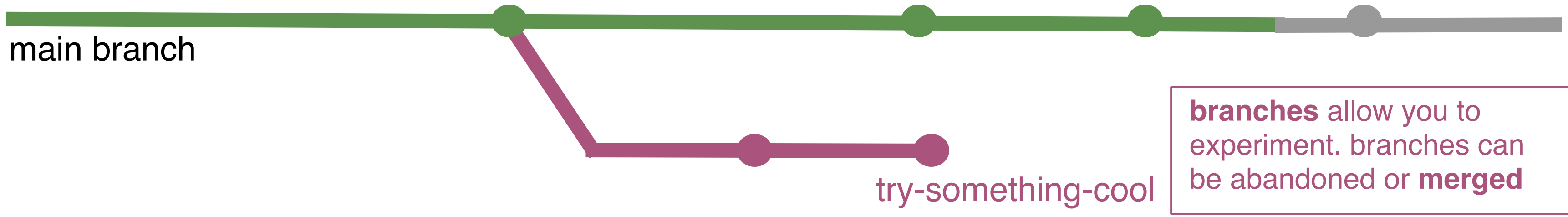


commits allow you to time travel because each commit is assigned a unique **hash**

One more git recap...

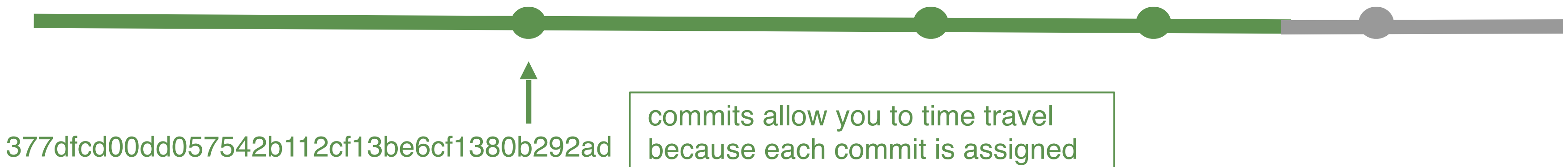


commits allow you to time travel because each commit is assigned a unique **hash**

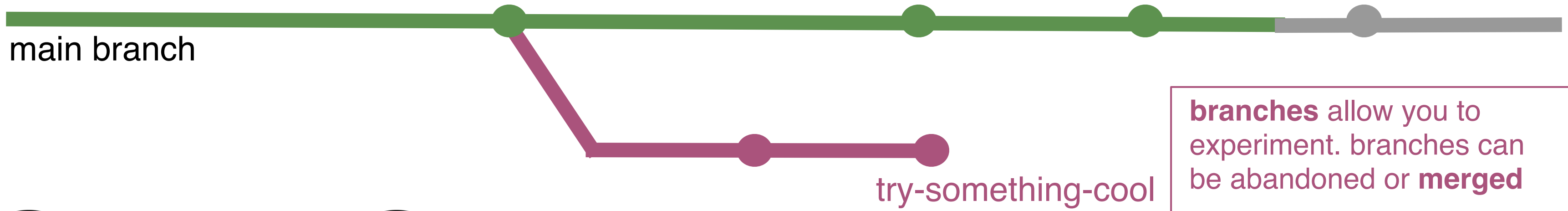


branches allow you to experiment. branches can be abandoned or **merged**

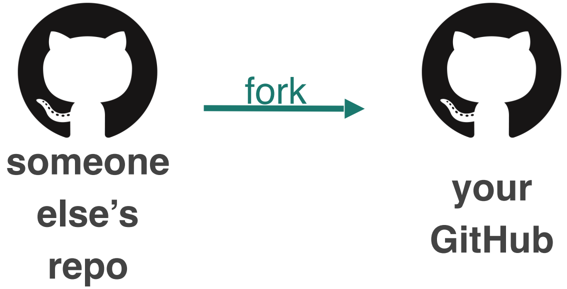
One more git recap...



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

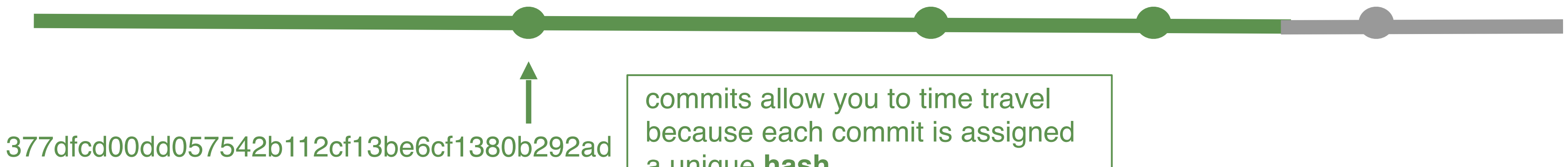


branches allow you to experiment. branches can be abandoned or **merged**

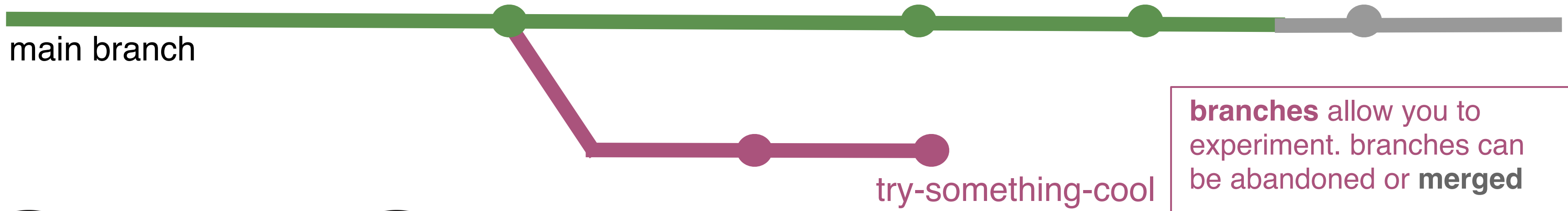


You can work on others' repos by first **forking** their repository onto your GitHub

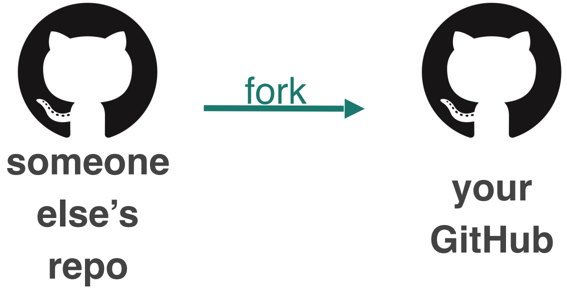
One more git recap...



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



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



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

https://forms.gle/eyxgHB3wvqmy17uR9

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

what should you do?

A **hash**

B **branch**

C **merge**

D **fork**

E **pull request**

Version Controller IV

https://forms.gle/eyxgHB3wvqmy17uR9

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?

A	B	C	D	E
hash	**branch**	**merge**	**fork**	**pull request**

Jupyter notebooks suck to version control

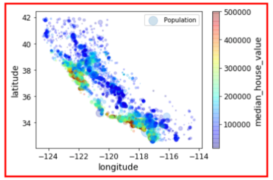
https://nextjournal.com/schmudde/how-to-version-control-jupyter

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.

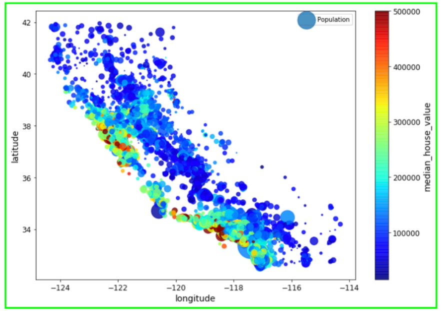
```
housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.2,
    s=housing["population"]/88, label="Population", figsize=(6,4),
    c="median_house_value", cmap=plt.get_cmap("jet"), colorbar=True,
    sharex=False)
plt.legend()
save_fig("housing_prices_scatterplot")
```

```
Saving figure housing_prices_scatterplot
```



```
housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.8,
    s=housing["population"]/21, label="Population", figsize=(10,7),
    c="median_house_value", cmap=plt.get_cmap("jet"), colorbar=True,
    sharex=False)
plt.legend()
save_fig("housing_prices_scatterplot")
```

```
Saving figure housing_prices_scatterplot
```



ReviewNB example courtesy of the ReviewNB website

Version Control: Practice

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

- 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

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

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