

Course Reminders

- Survey due *this* Fri 4/5 (11:59 PM) : http://bit.ly/cogs108_survey
- A1 - due *next* Sunday 4/14 (11:59 PM)

Note: Projects can be put on GitHub. Please do not put assignments on GitHub.

Section Info

	Day	Time	Location	TA/IA
B01	Mon	9 AM	CENTR 222	Mayank
A02	Mon	12 PM	CENTR 222	Yinhe
B02	Mon	4 PM	MANDEB-150	Ashlesha
A04	Wed	9 AM	CENTR 222	Chris
B03	Wed	10 AM	PETER 102	Qiuli
A03	Wed	1 PM	PCYNH 121	Emily
B04	Wed	2 PM	PETER 102	Devendra
B05	Wed	3 PM	PETER 102	Alkin
A01	Wed	4 PM	PCYNH 121	Alkin
B06	Fri	3 PM	MANDEB-150	Phillip
A05	Fri	3 PM	PCYNH 121	John
B07	Fri	4 PM	MANDEB-150	Akshansh
B08	Fri	5 PM	MANDEB-150	Yanyi

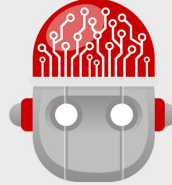
This information has been added to the syllabus.

Office Hours

Date & Time	Location	Instructional Staff
M 3-5PM & W 3-5PM	CSB 243	Professor Ellis
M 2PM-3PM	CSB 114	Akshansh Chalal
Tu 11AM-12PM	CSB 114	Mayank Rajoria
W 11AM-12PM	PC Theater Lounge	Phillip Lagoc
F 1-2PM	CSB 114	Chris Chen

This information has been added to the syllabus and Piazza. Any changes will be posted to Piazza and announced in class.

PROBOT PLAYGROUND



Poker Bot Competition + Career Fair

LOCATION: PC Ballroom East

DATE: April 28th, 2019

TIME: 10am to 5pm

Visit www.probotplayground.com
for more information!

Python, Jupyter, & Version Control

Shannon E. Ellis, Ph.D
UC San Diego



Department of Cognitive Science
sellis@ucsd.edu

GitHub Course Materials:
www.github.com/COGS108

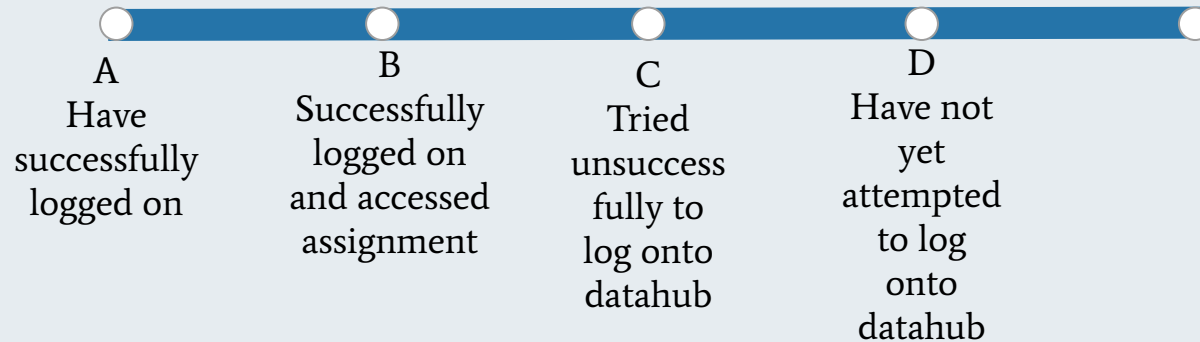
Course Assignments:

<http://datahub.ucsd.edu>










First iclicker question!

What's your current datahub status?



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)

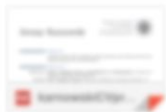
Yup, this sucks too.

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!

...

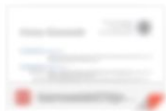


May 11



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

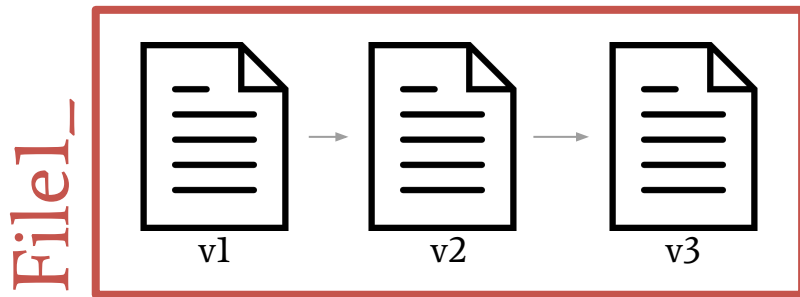
- Enables multiple people to simultaneously work on a single project.
- Each person edits their own copy of the files and chooses when to share those changes with the rest of the team.
- Thus, temporary or partial edits by one person do not interfere with another person's work

What is version control?

A way to manage the evolution of a set of files

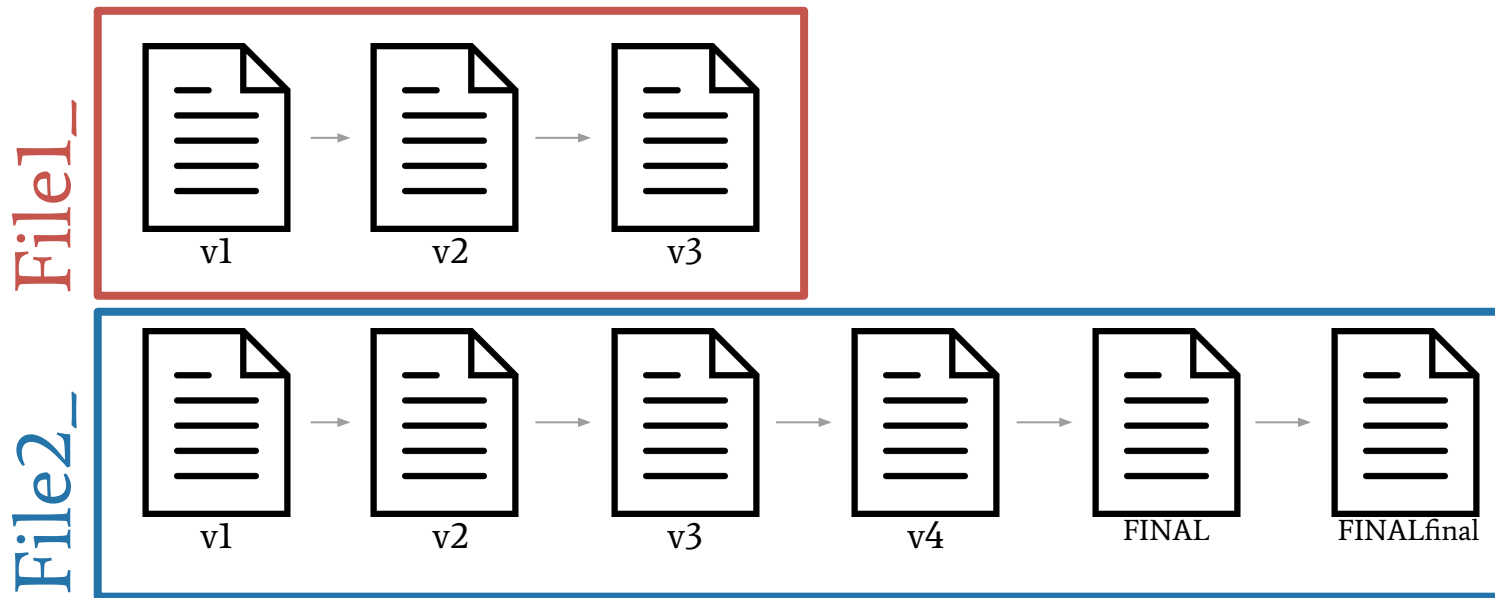
What is version control?

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?

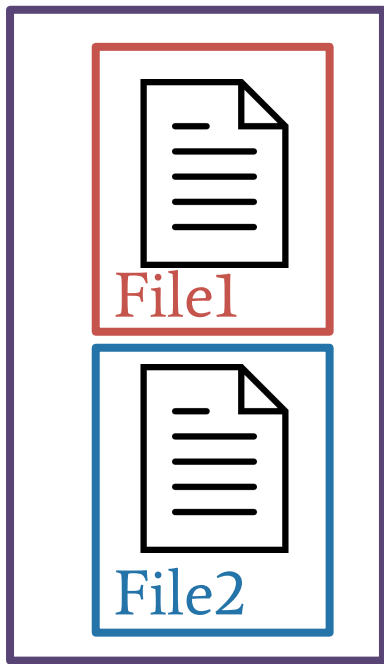
A way to manage the evolution of a set of files



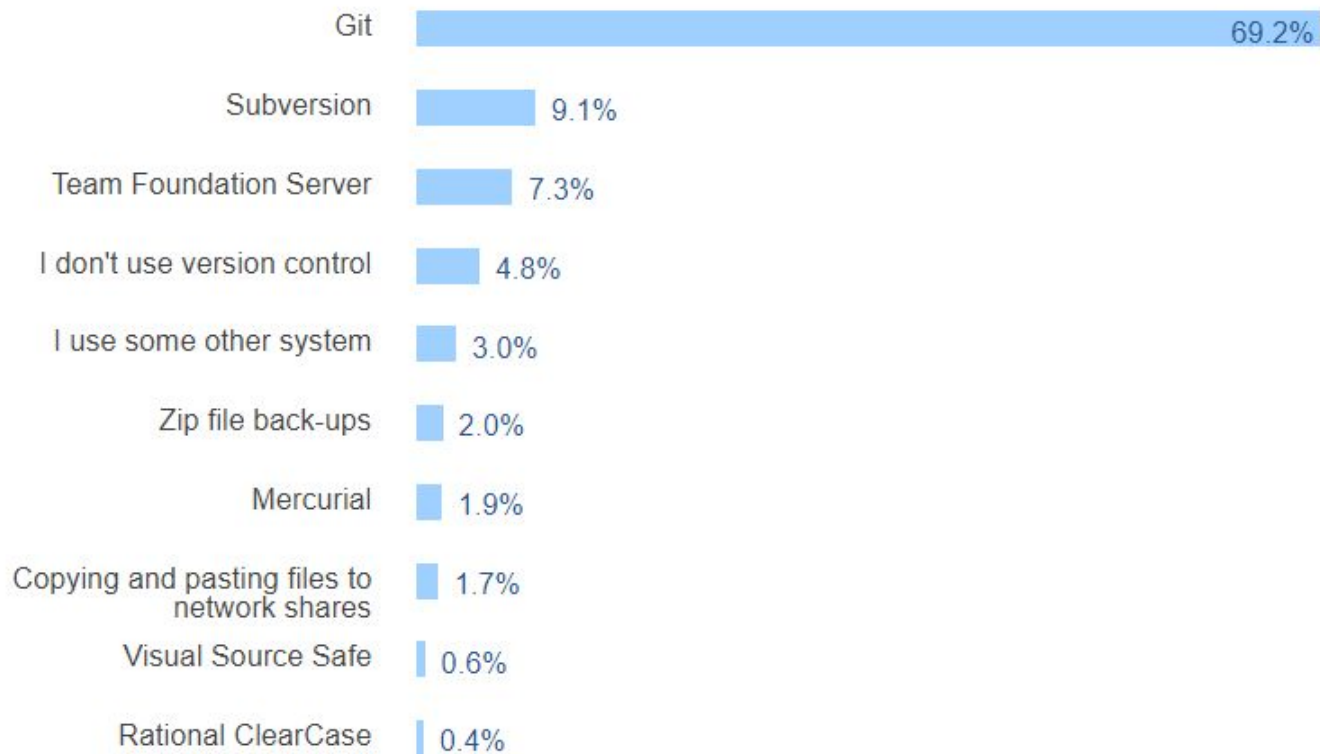
When using a version control system, you have **one copy of each file** and the *version control system tracks the changes* that have occurred over time

What is version control?

A way to manage the evolution of a set of files



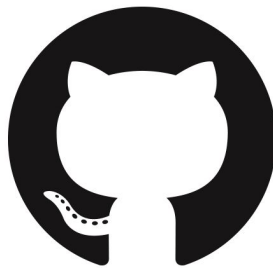
The set of files is referred to as a **repository (repo)**



git & GitHub

git

the version control system



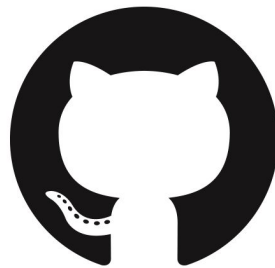
GitHub (or Bitbucket or GitLab) is the home **where your git-based projects live** on the Internet.

git & GitHub

git

the version control system

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



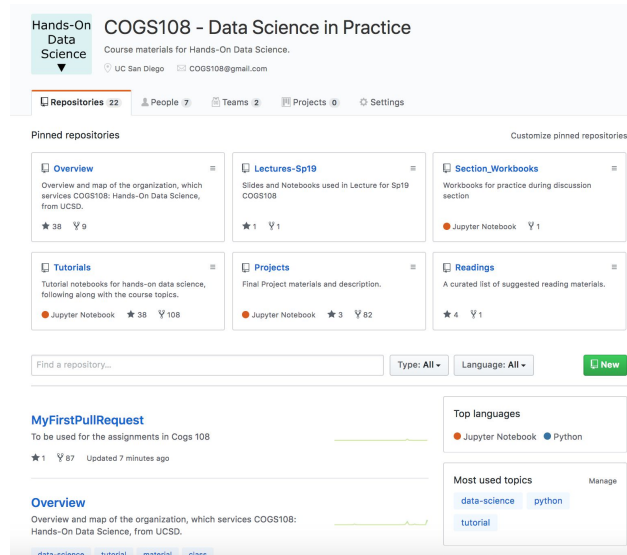
GitHub (or Bitbucket or GitLab) is the
home **where your git-based projects**
live on the Internet.

~ Dropbox....but
way better

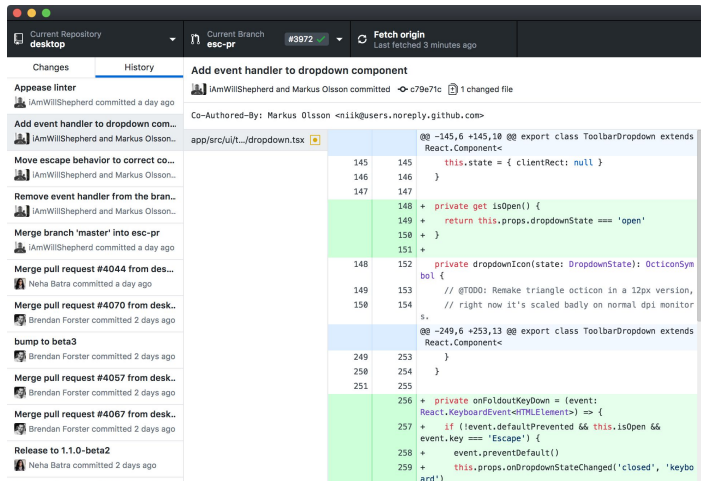
What version control looks like

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

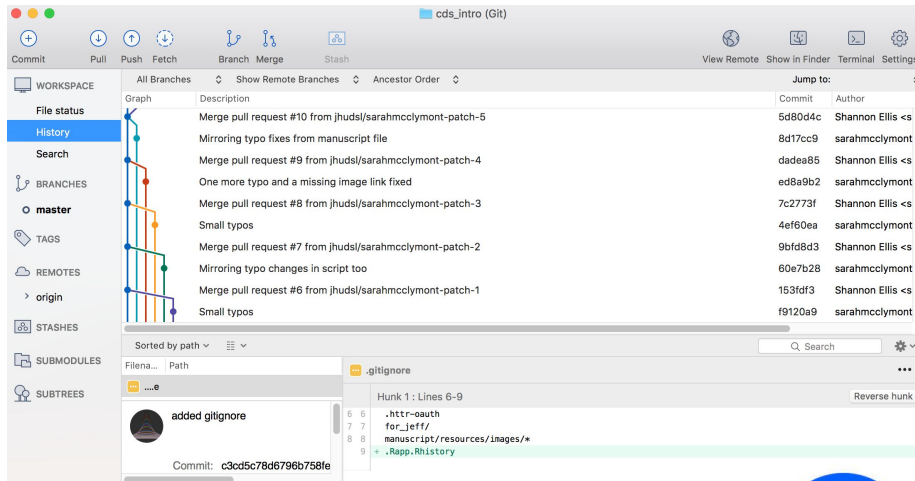
Terminal
git



GUIs can be helpful when working with version control



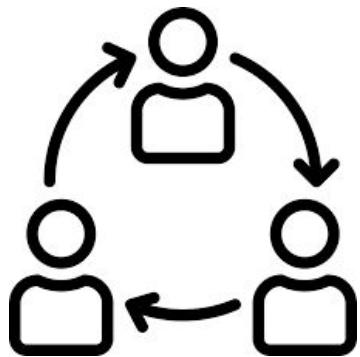
GitHub Desktop



SourceTree



Why version control with git and GitHub?



Collaboration



Returning to
a safe state

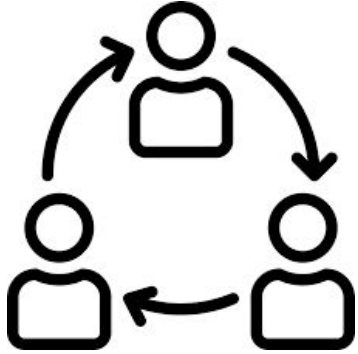


Exposure for
your work

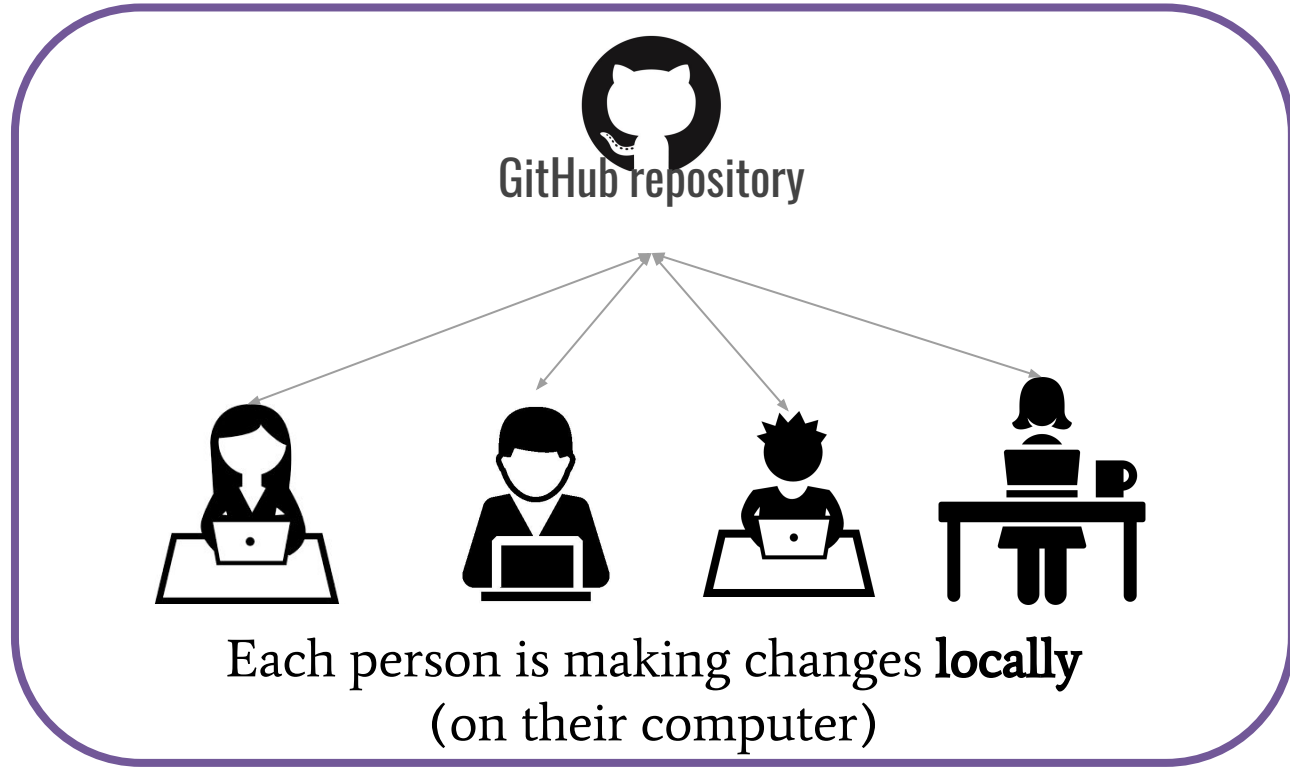


Tracking
others' work

Collaborate like you do with Google Docs



Collaboration



Make changes locally, while knowing a stable copy exists



Returning to
a safe state



You're free and safe to **try things out locally**. You'll only send changes to the repo when you're at a stable point

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!

When you'll HAVE to use GitHub in this course

- Course materials
- completing A1 (individual)
- Final Project Submission (one member of group)

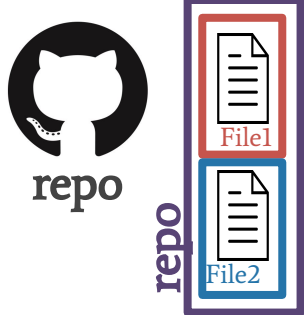
Your team SHOULD use GitHub to work on and complete your final project in this course!

When you'll HAVE to use GitHub in this course

- Course materials
- Completing A1
- Final Project Submission

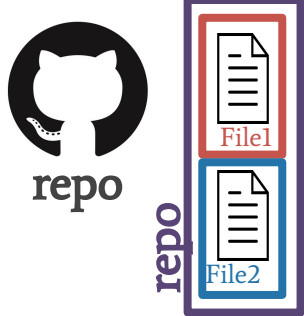
When you SHOULD use GitHub in this course

- To work on your projects with your group members!
 - version control is *perfect* for this!
- To share your project with the world!
 - GitHub is your social media platform in the coding world!
 - Remember to give credit to teammates



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

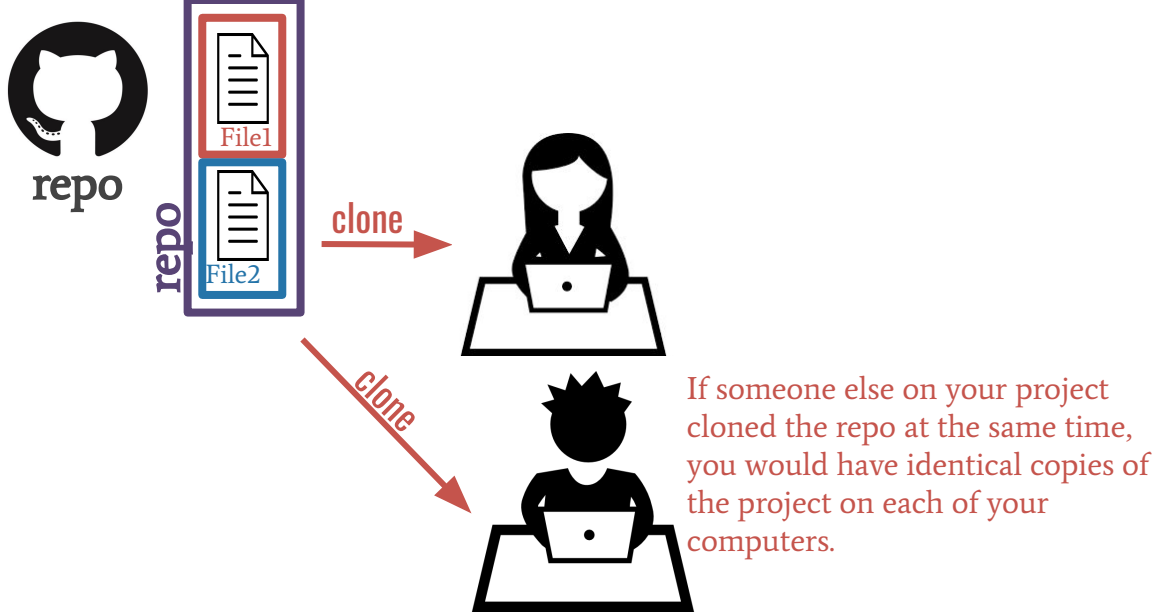
GitHub is a **remote host**. The files are geographically distant from any files on your computer.

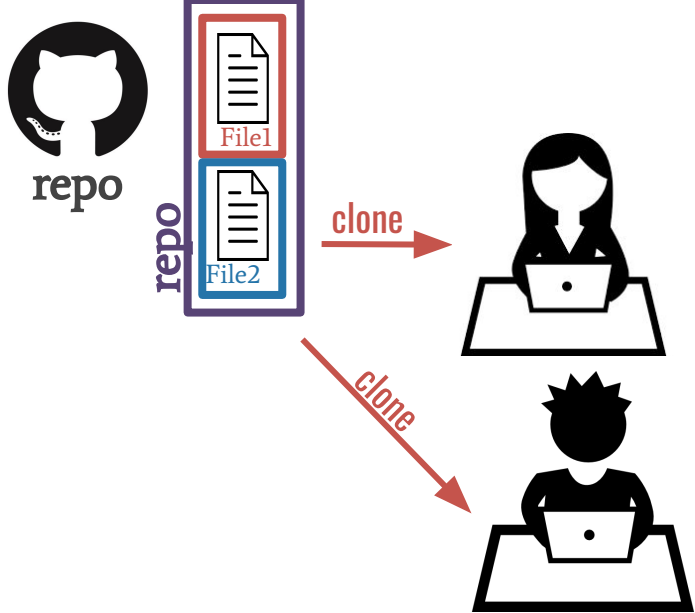


clone

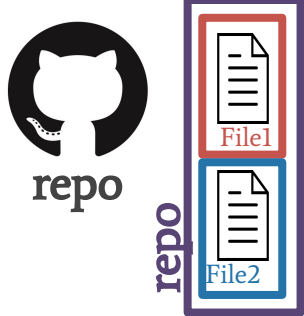


When you first make a copy onto your local computer (read: laptop), you **clone** the repository.

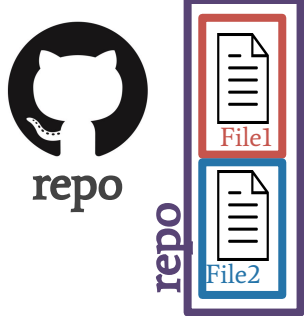




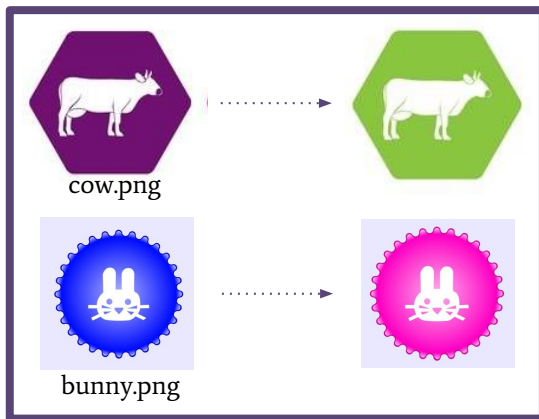
Yay! Everyone can
work on the project!

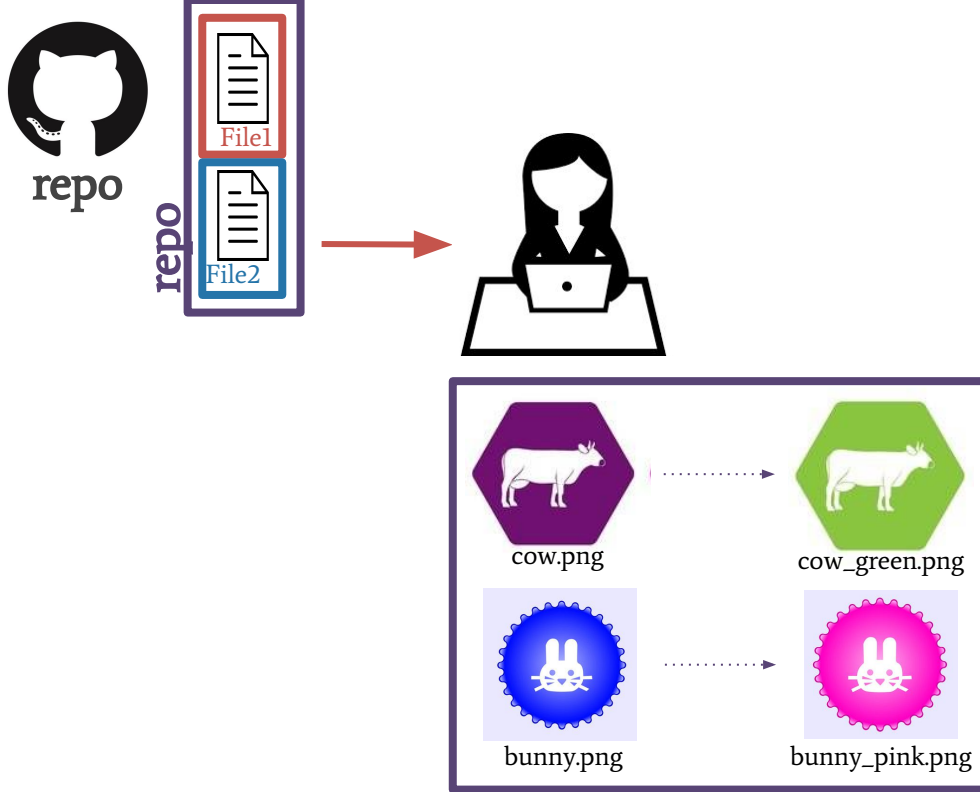


You decide you want to
change a few of the
images in the project.

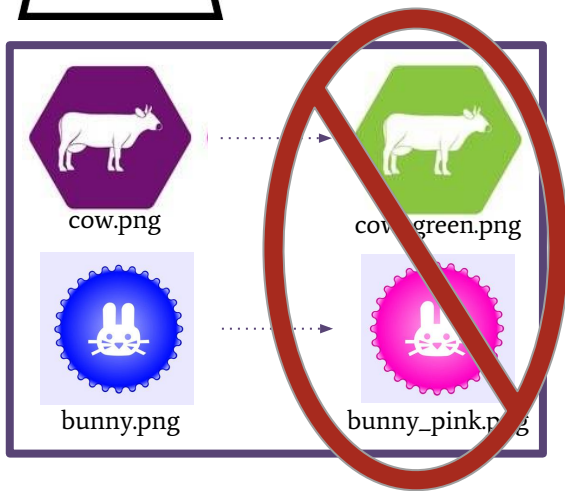
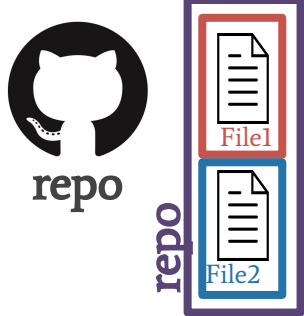


You decide you want to change a few of the images in the project.

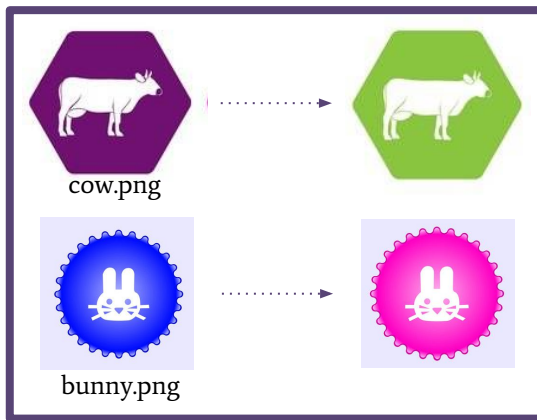
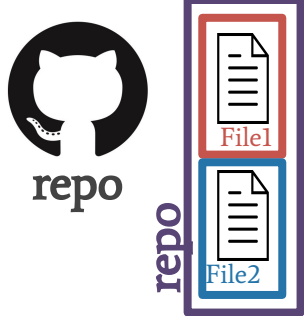




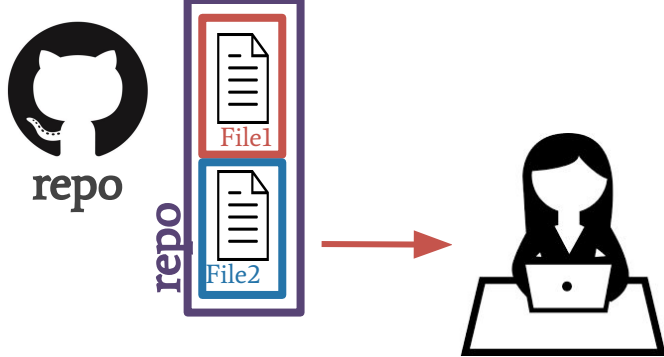
without git...you'd
likely rename
these files....



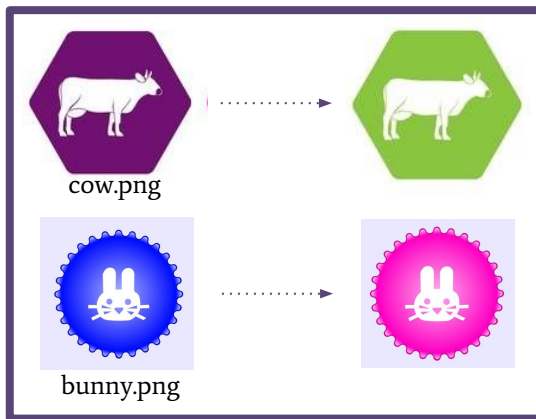
Thank goodness
those days are
over!



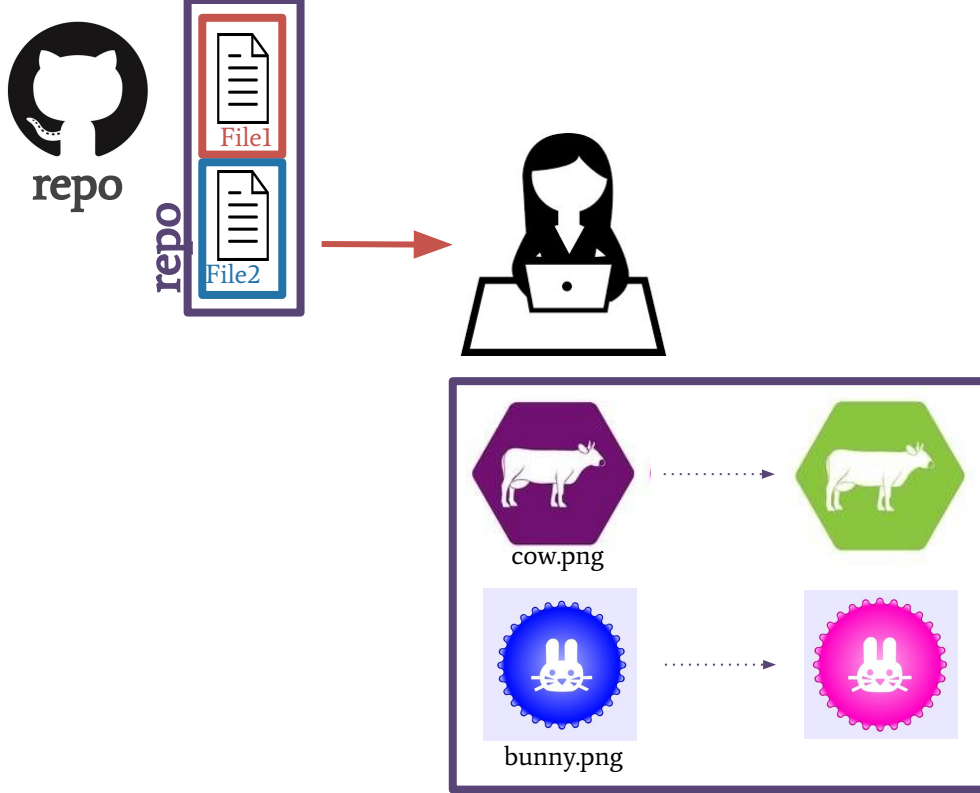
Instead, you tell git which files you'd like to keep track of using **add**. This process is called *staging*.



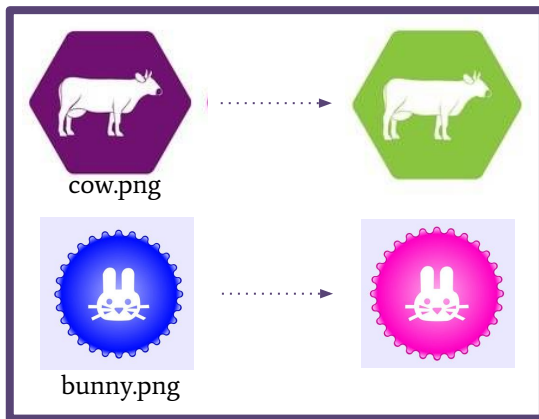
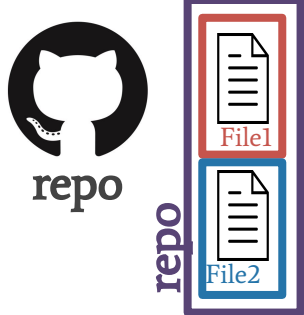
git add file	stages specified file (or folder)
git add .	stages new and modified files
git add -u	stages modified and deleted files
git add -A	stages new, modified, and deleted files
git add *.csv	Stages any files with .csv extension
git add *	Use with caution: stages everything



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



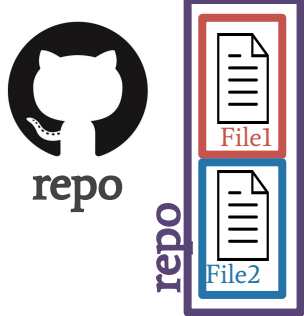
Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.



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

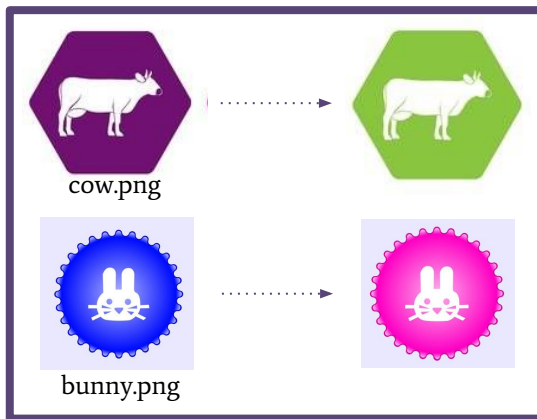
A horizontal timeline with a green segment on the left and a grey segment on the right. There are five dots along the timeline: the first three are on the green segment, and the last two are on the grey segment. The text 'A commit tracks who, what, and when' is centered below the green segment.

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



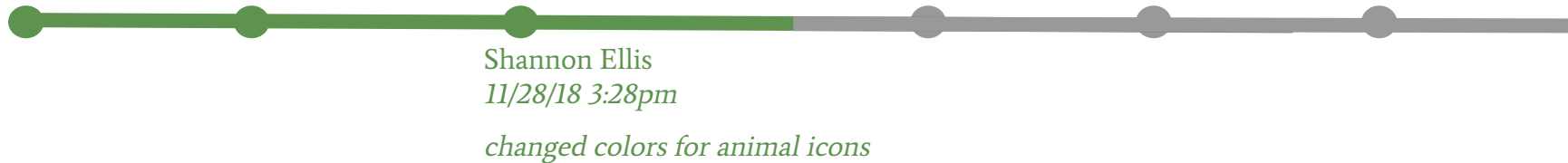
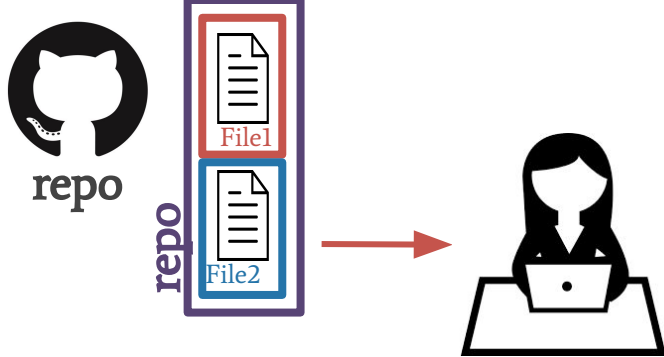
You can make commits more informative by adding a **commit message**.

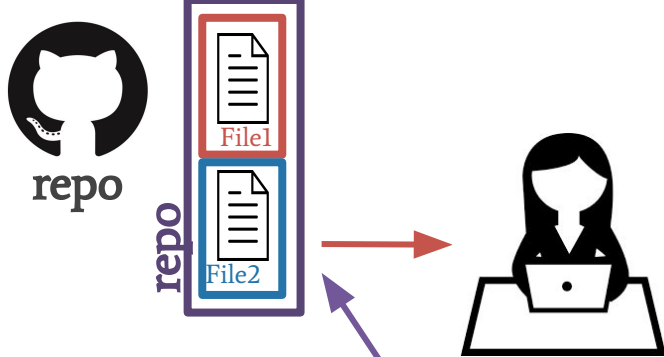
Example: `git commit -m "changed colors for animal icons"`



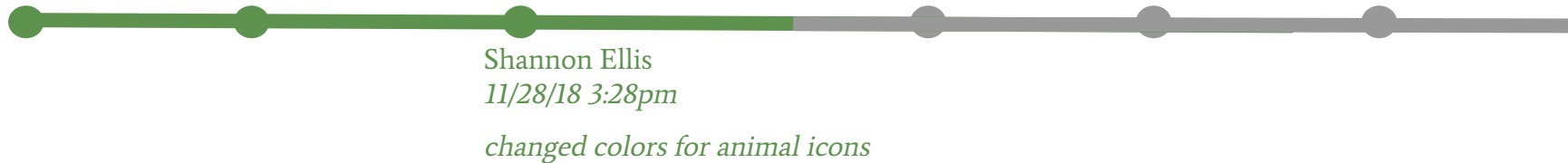
Then, you create a snapshot of your files at this point. This snapshot is called a **commit**.

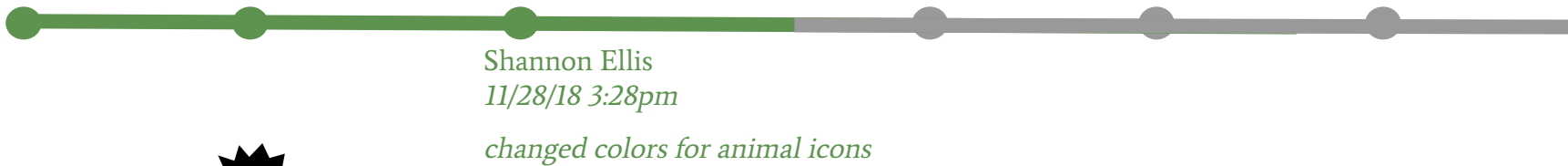
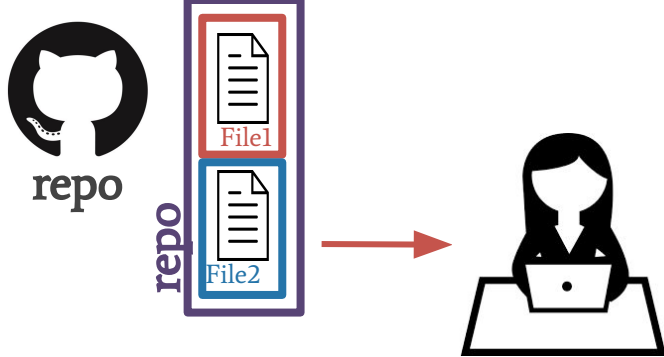
A **commit** tracks who, what, and when



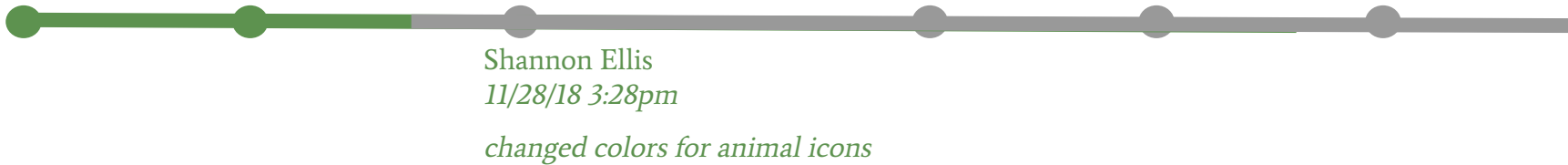


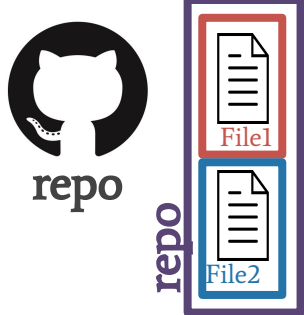
Remember, you're not the only one working on this project though! You want your teammates to have access to these changes! You **push** these changes back to the remote.





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

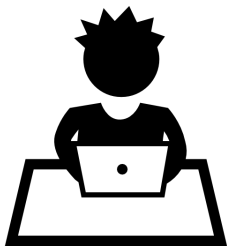




To catch up, your teammate will have to **pull** the changes from GitHub (remote)



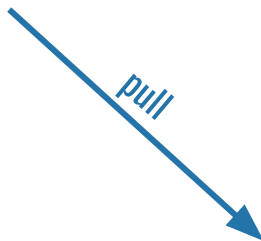
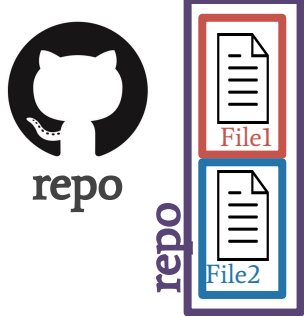
Shannon Ellis
11/28/18 3:28pm
changed colors for animal icons



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



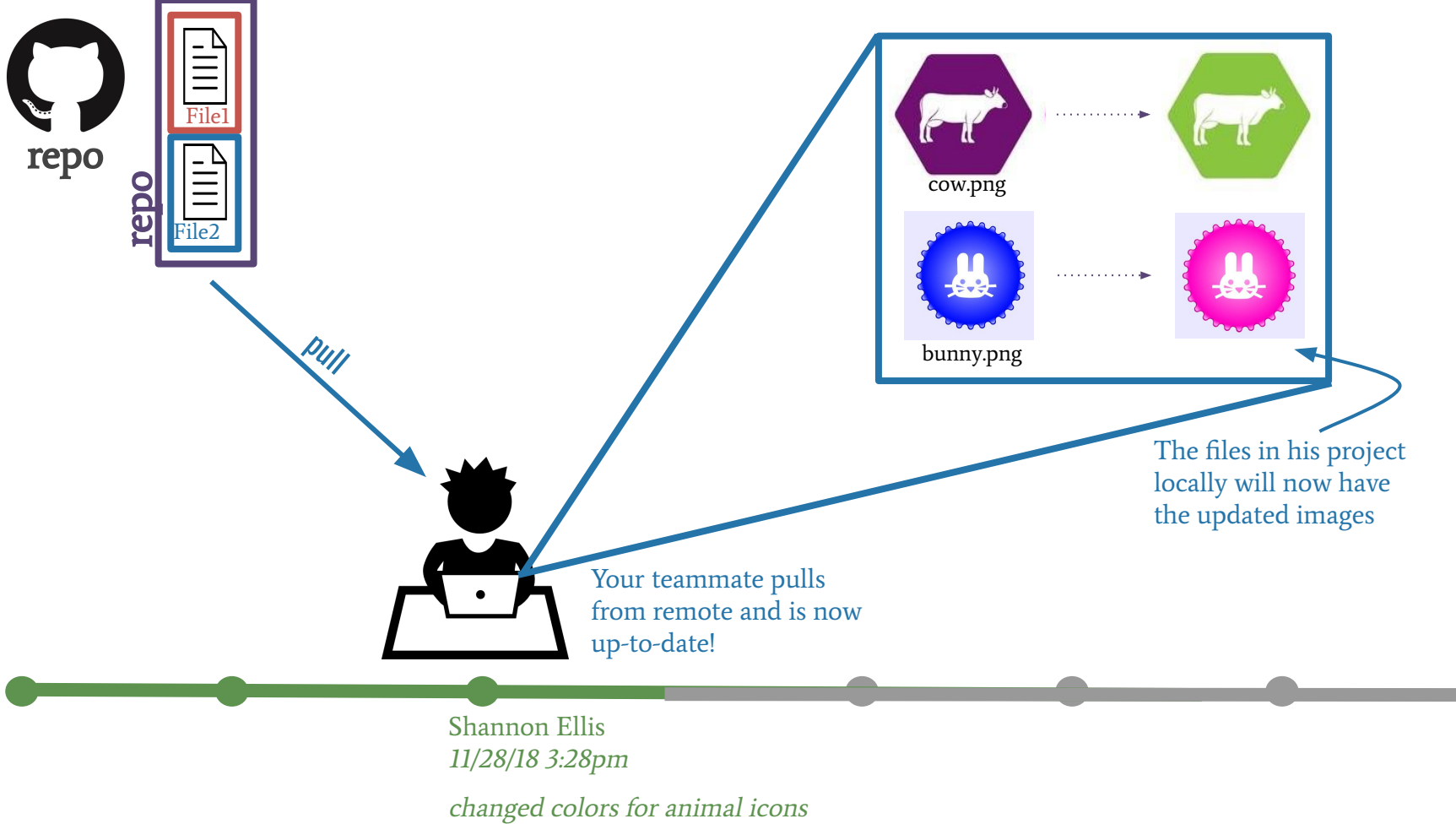
Shannon Ellis
11/28/18 3:28pm
changed colors for animal icons

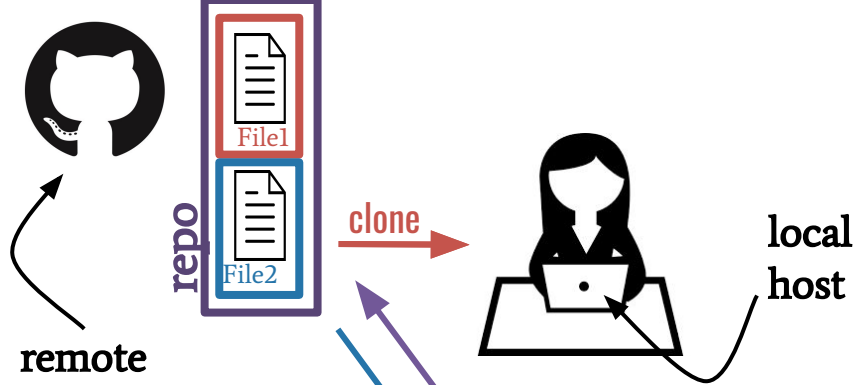


Your teammate pulls
from remote and is now
up-to-date!



Shannon Ellis
11/28/18 3:28pm
changed colors for animal icons





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
```

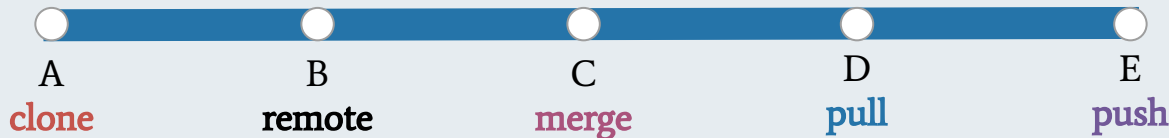
Review & Question Time



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?



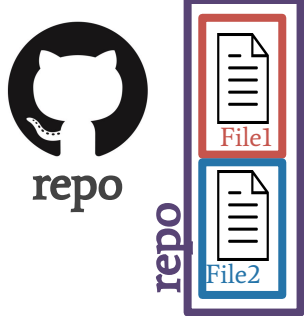


Version Controller II

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.

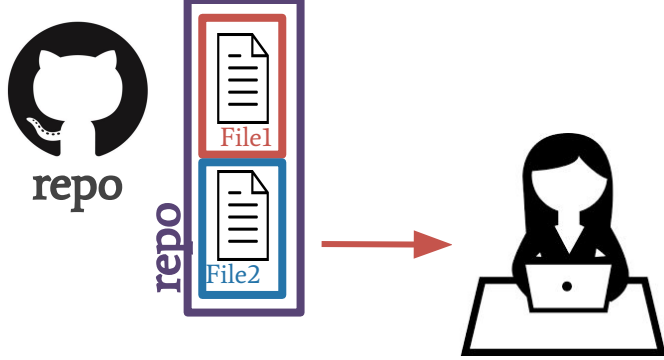
What do you do to get the repo on your computer?





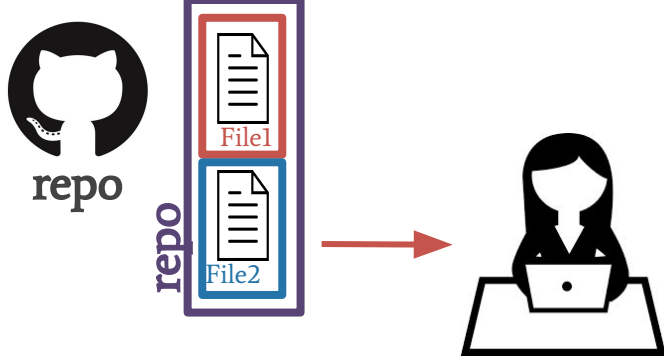
Each time you create a commit, git tracks the changes made automatically.





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

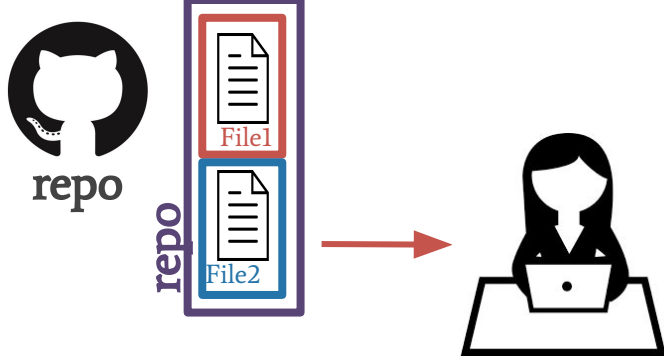




By committing each time you make changes, git allows you to time travel!

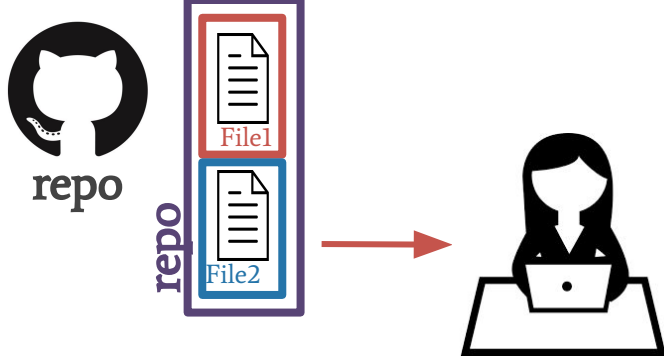


There's a unique id, known as a **hash**, associated with each commit.



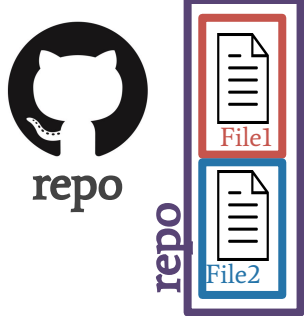
You can return to the state of the repository at any commit. Future commits don't disappear. They just aren't visible when you **check out** an older commit.



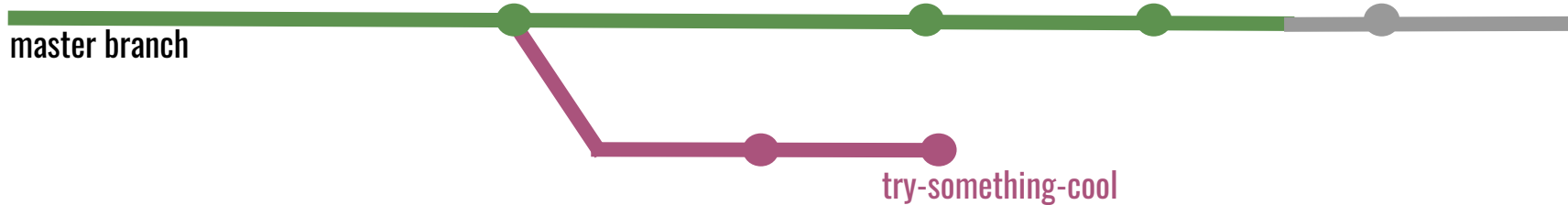


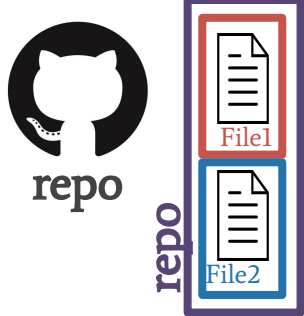
But...not everything is always linear. Sometimes you want to try something out and you're not sure it's going to work. This is where you'll want to use a **branch**.



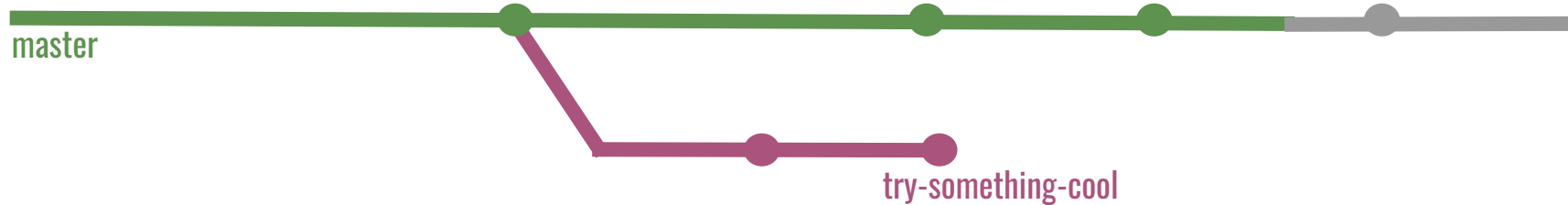


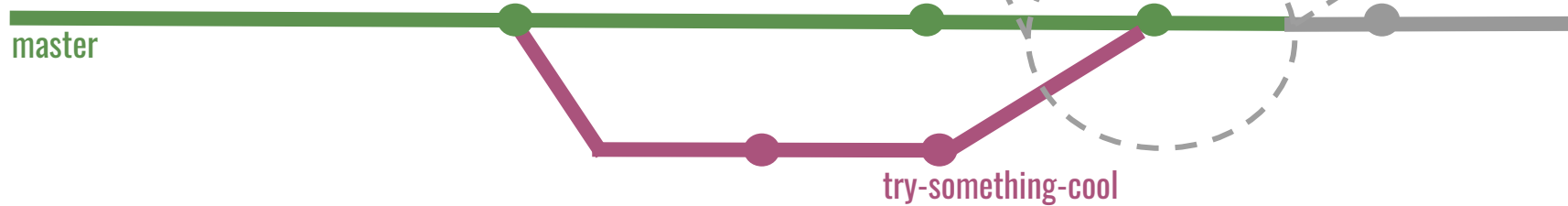
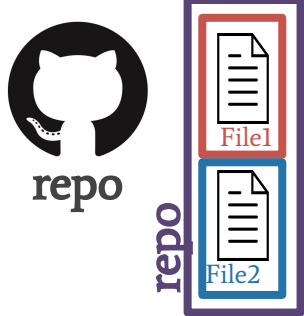
It's a good way to experiment. It's pretty easy to get rid of a branch later on should you not want to include the commits on that branch.





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

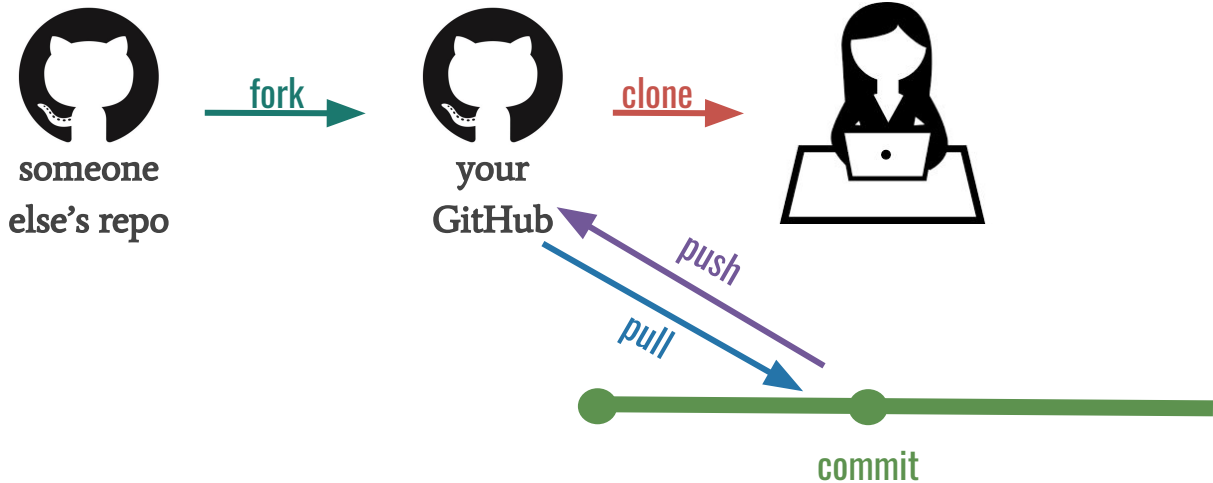




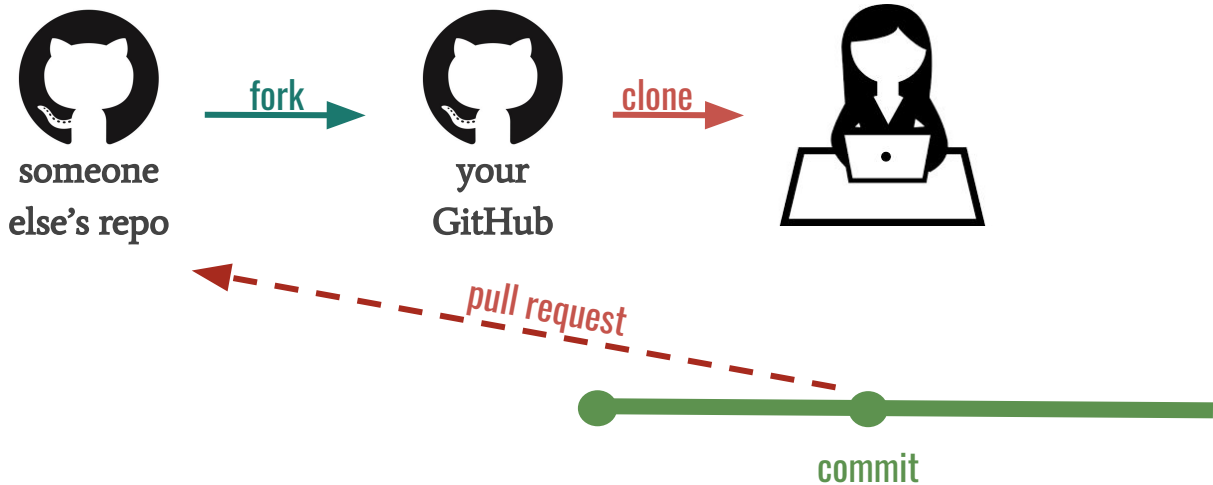
A **merge** allows you to combine the commits from a branch back into the master.



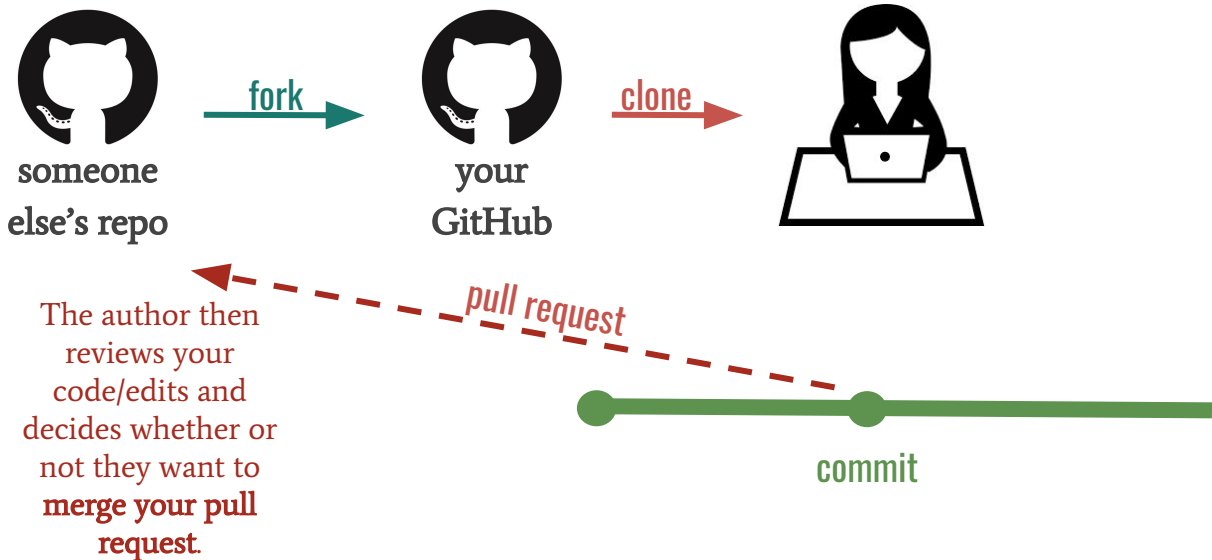
What if someone else is working on something cool and you want to play around with it? You'll have to **fork** their repo.



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

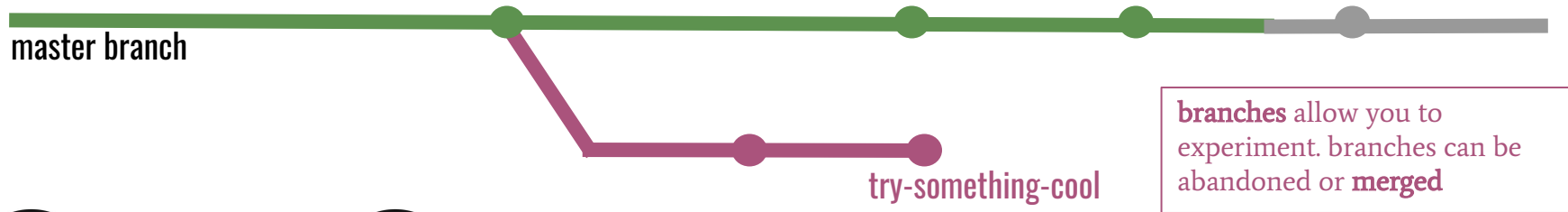
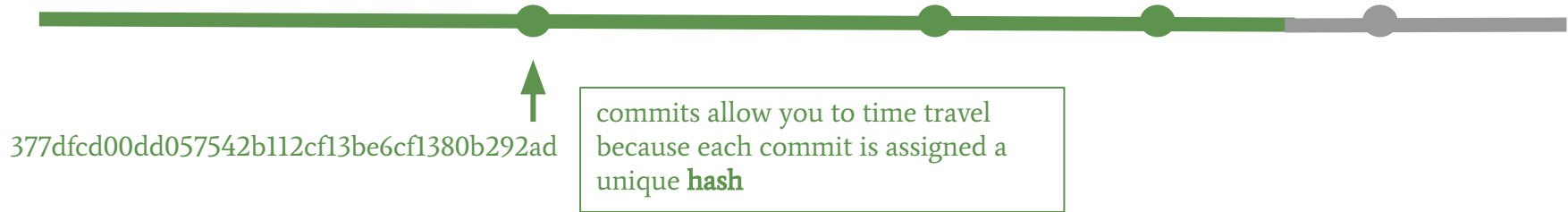


commits allow you to time travel
because each commit is assigned a
unique **hash**

One more git recap...

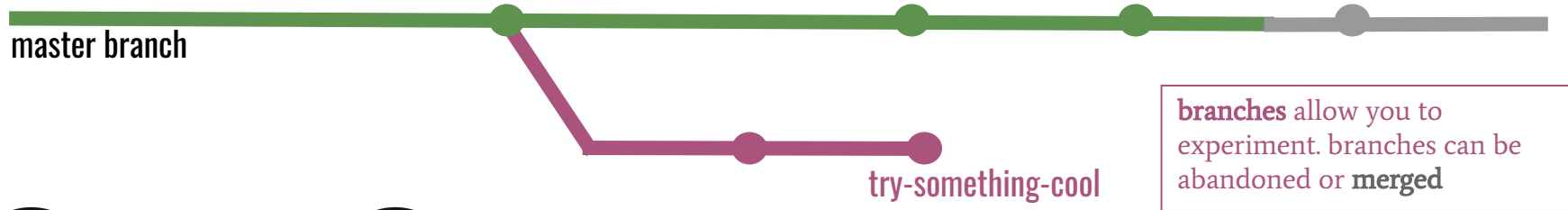
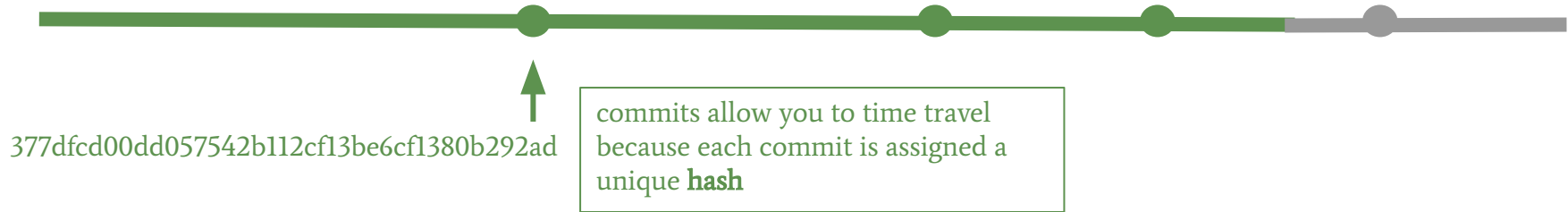


One more git recap...



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...

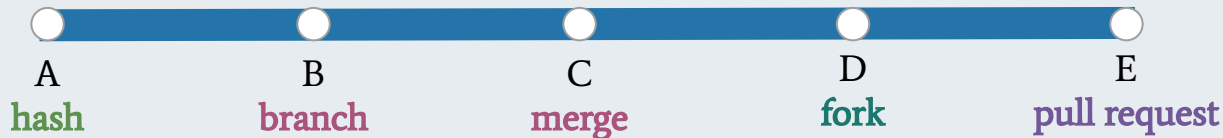
Review & Question Time



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?

