Git

Some of the stuff we don't really learn by default

What I want to cover

- Forking
- What is Git?
- Why use branching?
- Rebasing, and merging.
- Why use pull requests?
- How are these used in small projects?
- How are these used in bigger open source projects?

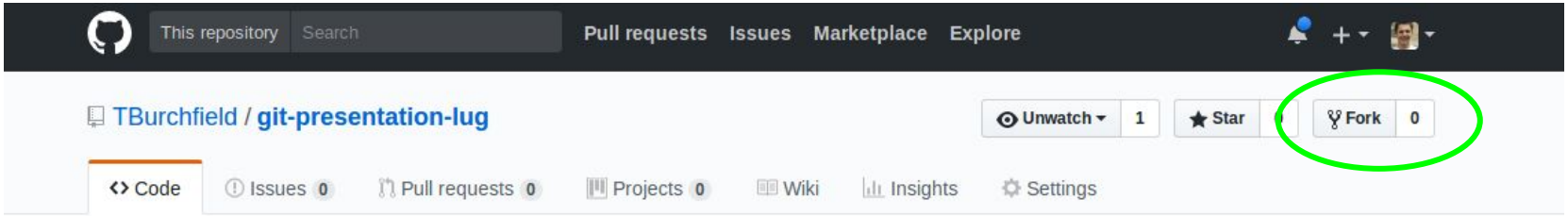
What You'll Need

- git (Preferably cli, a GUI wrapper could work, but no promises)
- web browser

What is git?

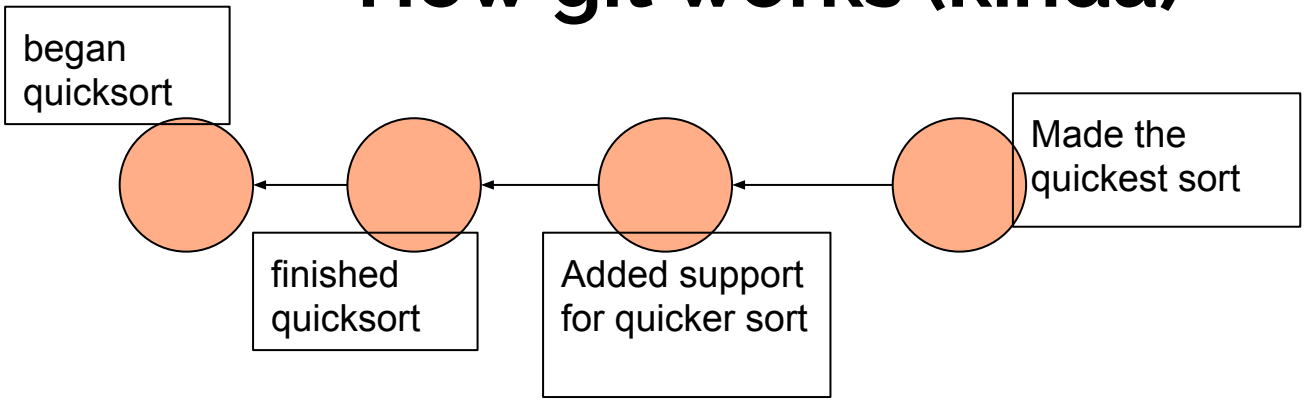
- A "Version Control System"
- Allows you to develop features independently, and keep a history of the state of a project

https://github.com/TBurchfield/git-presentation-lug



```
$ alias gitlog='git log --oneline --branches --graph --decorate'

$ git clone git@github.com:$YOURUSERNAME/git-presentation-lug.git

$ cd git-presentation-lug
```

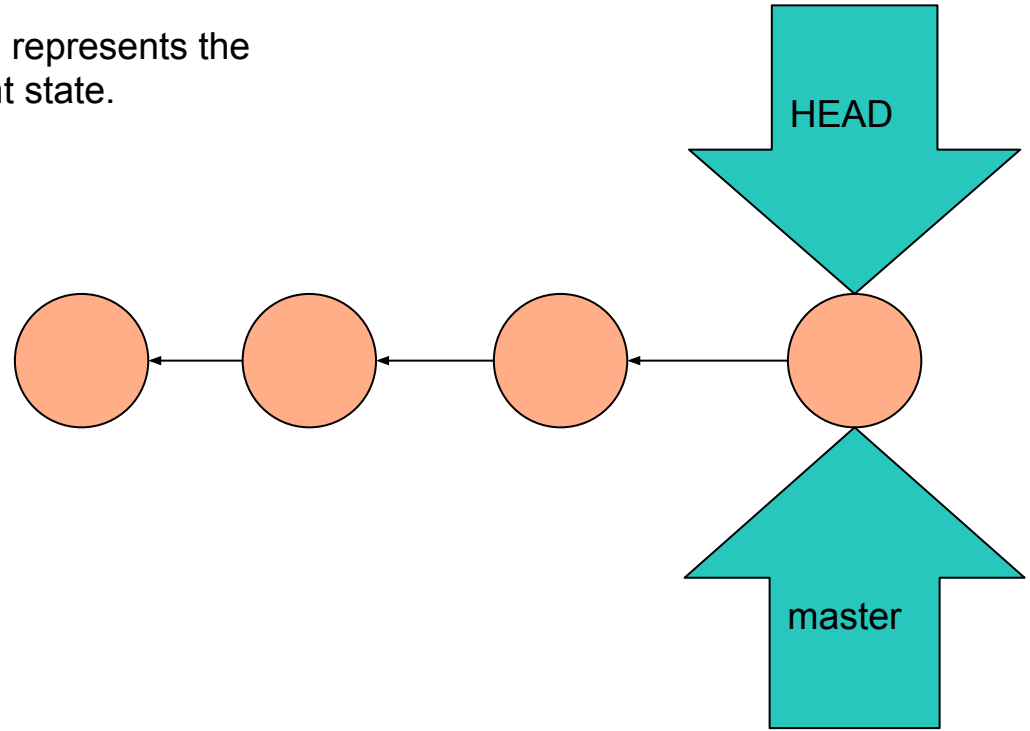
How git works (kinda)



Each node is a commit, which represent changes to the file(s). They each reference the previous one, or "parent commit"

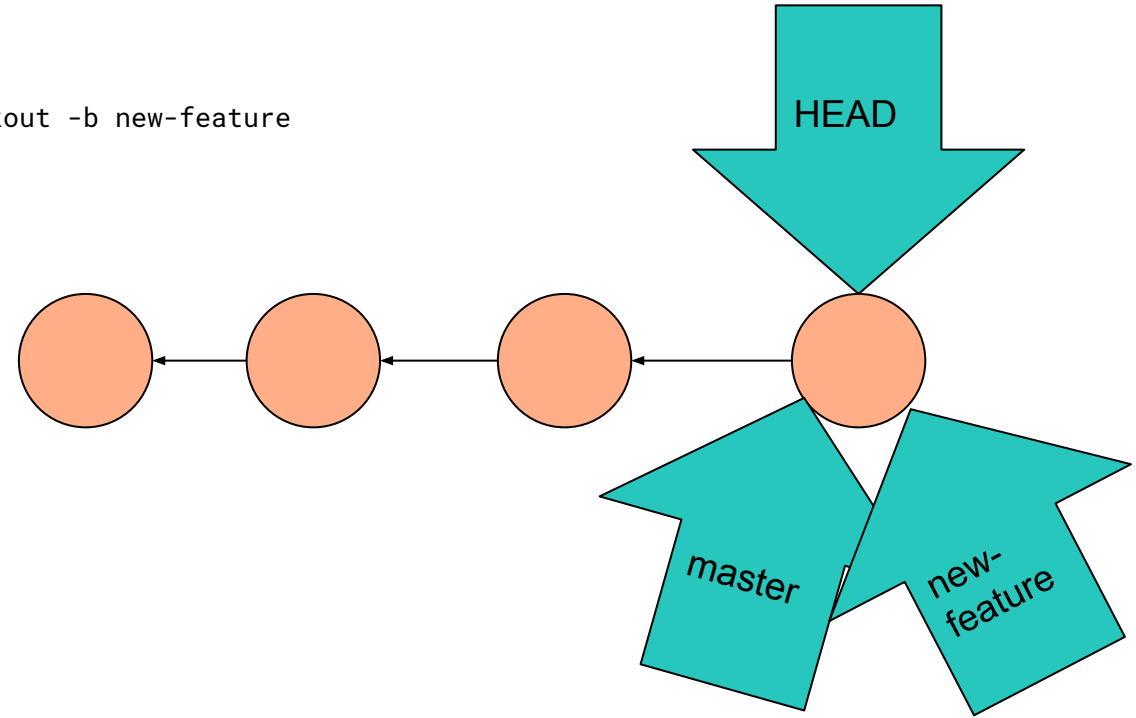
How git works (kinda)

HEAD represents the current state.

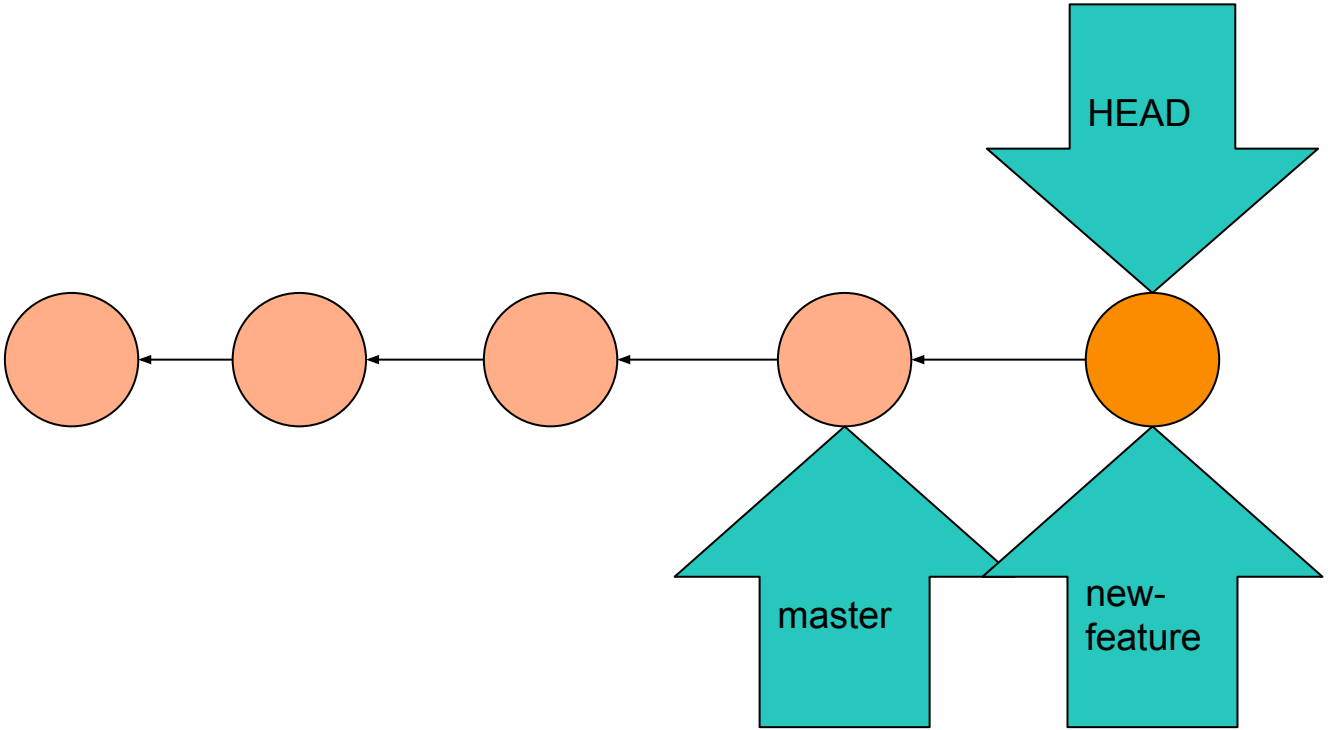


How branching (kinda)

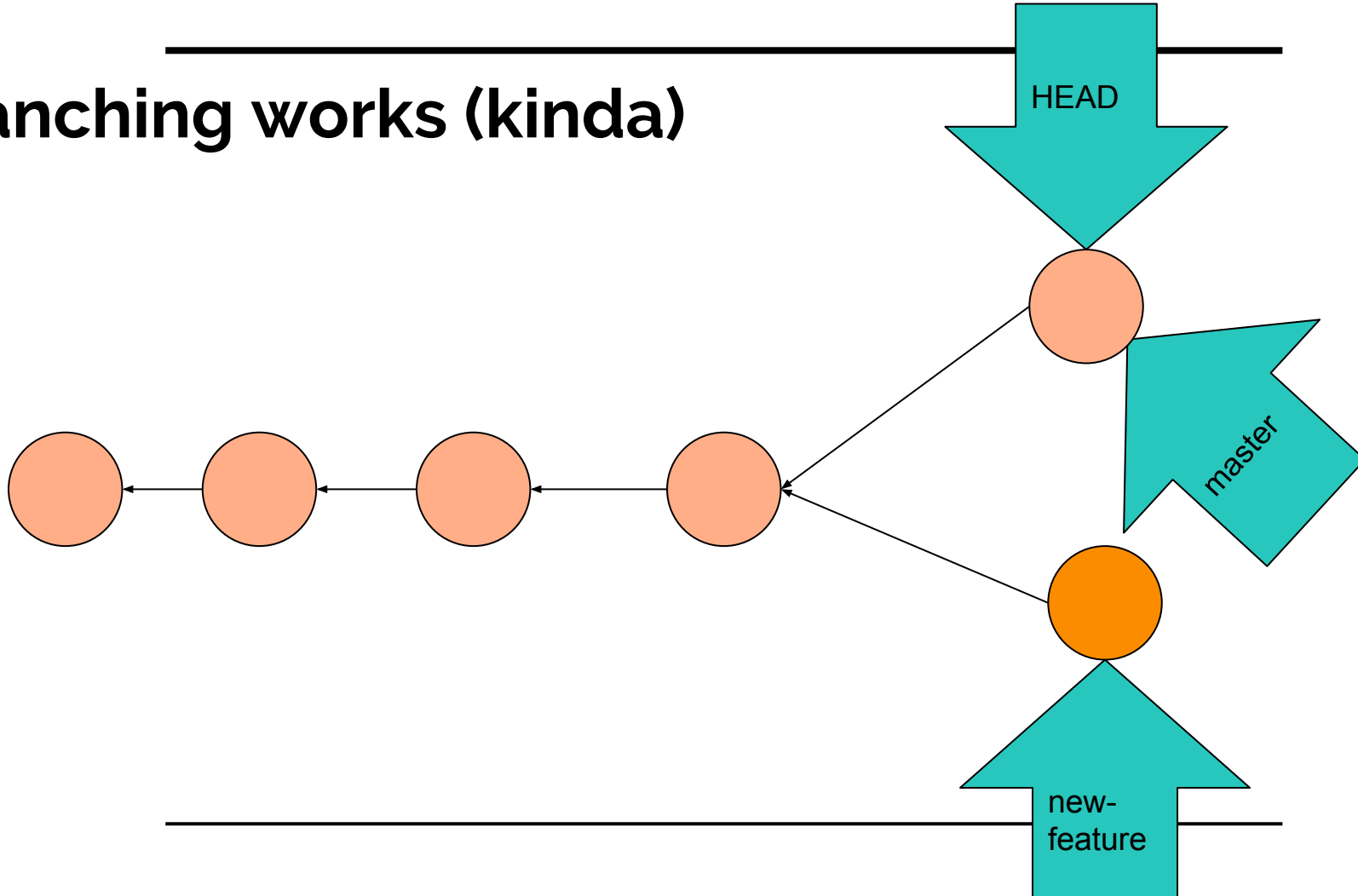
```
#done with
$ git checkout -b new-feature
```



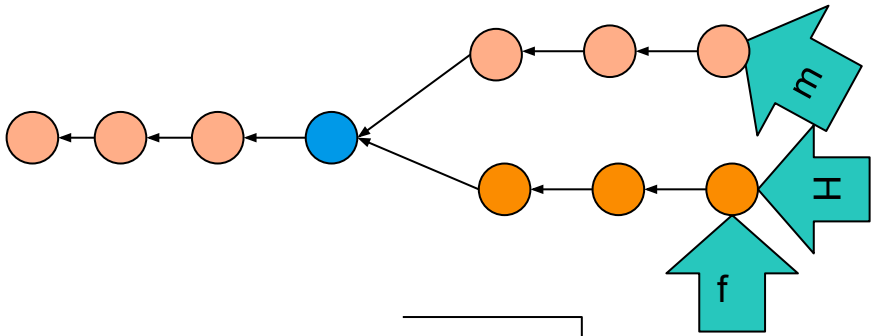
How branching works (kinda)



How branching works (kinda)



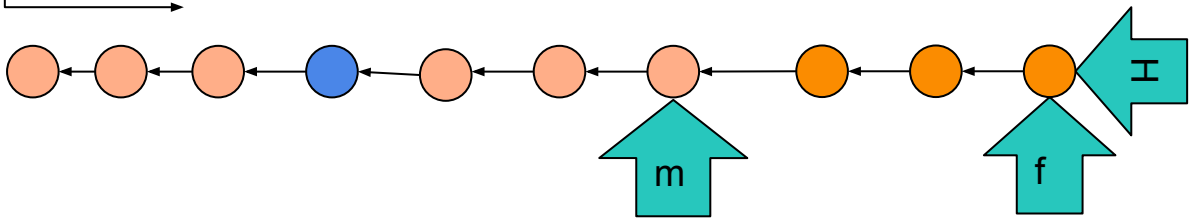
How rebasing works (kinda



Just moves a part of the graph!

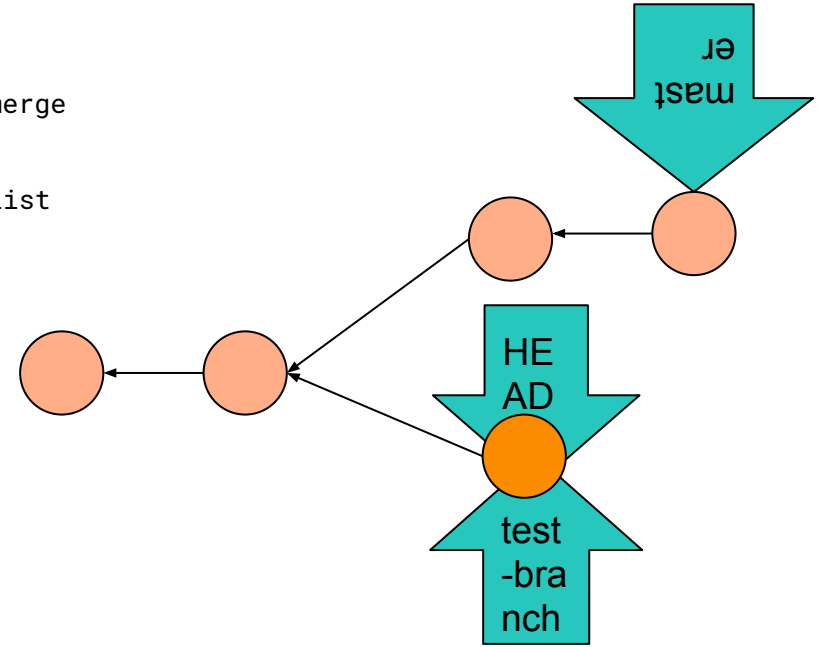
Find common ancestor, move nodes from HEAD until there onto the target.

```
$ git rebase master
```



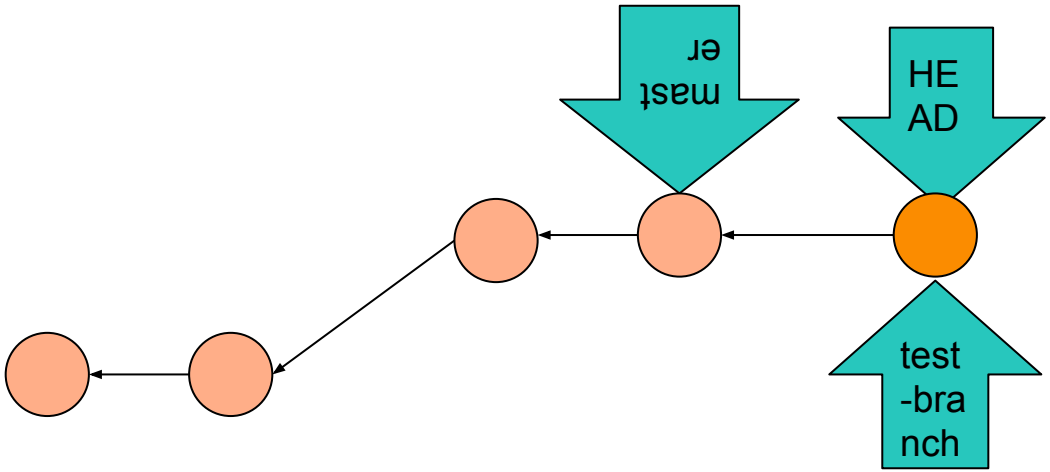
Example

```
$ git checkout test-branch

$ gitlog

* 27977b4 (origin/master, origin/HEAD, master) Add item about merge
conflicts
* 7e1247b Add example.py
| * 73d2856 (HEAD -> test-branch, origin/test-branch) Example list
|/
* 6cec5fc Add basic sentence
* 64cd004 Initial Commit
```



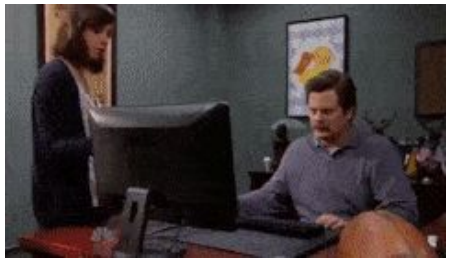
Example

```
$ git checkout test-master

$ git rebase master

#Should yield:
```



MERGE CONFLICTS

```
$ cat README.md

Today I'll show you a few basic git things that will be helpful in real
projects!

<<<<<<< 27977b4a7d83ef2ad4a806fcde8c8bd5049a14c4
 - Help!  A merge conflict!
=======
 - Branching
 - PRs
 - Forking
 - Rebasing
>>>>>>> Example list
```

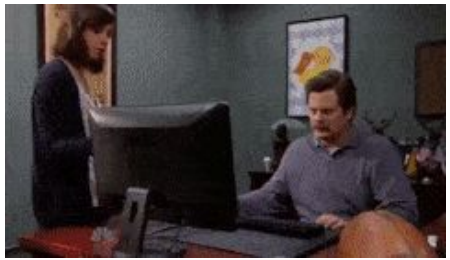


fixing merge conflicts

```
$ vim README.md

$ cat README.md
```

Today I'll show you a few basic git things that will be helpful in real projects!

- Help! A merge conflict!
- Branching
- PRs
- Forking
- Rebasing



fixing merge conflicts

```
$ git status

# Helpful stuff about a rebase being in progress,
# and merge conflicts

$ git add README.md

$ git rebase --continue

$ gitlog

* 4f5d727 (HEAD -> test-branch) Example list
* 27977b4 (origin/master, origin/HEAD, master) Add item [...]
* 7e1247b Add example.py
* 6cec5fc Add basic sentence
* 64cd004 Initial Commit
```

Pull Requests

- Used to request copying commits from one repo into another. (Or even from one branch to another in the same repo)
  - For features/bugfixes
- Can be used for big projects that tons of people use
- Can also be used in a smaller project, just to help people collaborate

Your turn! Well-

First, I'm gonna do a few things: *

```
$ git checkout master
$ vim $MOREFILES
$ git add $MOREFILES
$ git commit -m "Added more files"
$ git push
```

* reminder to self actually do this

But now your fork doesn't have this stuff!

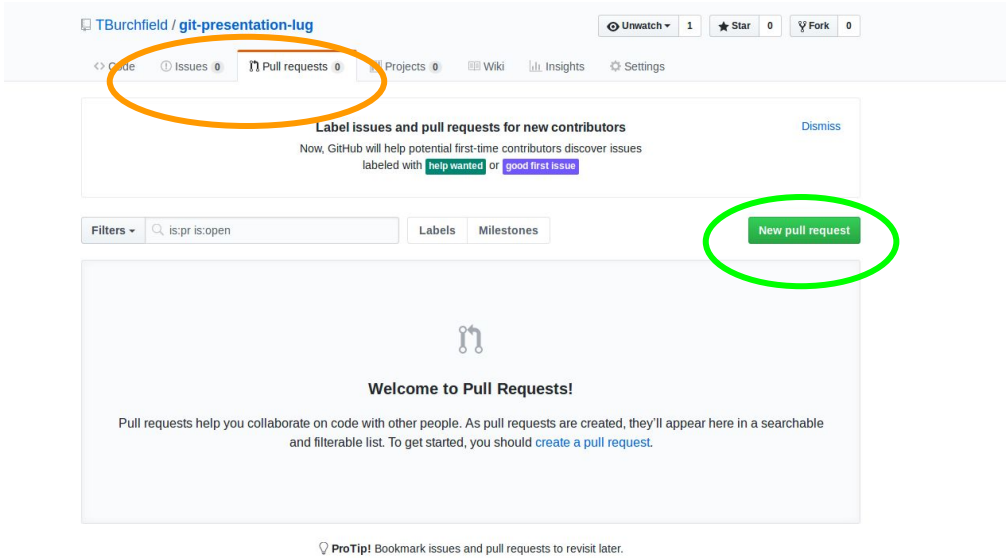
```
$ git remote add upstream git@github.com:TBurchfield/git-presentation-lug.git

$ git remote -v

#Info about remotes

#Now, pull my changes into your repo

$ git pull upstream master
```

Now actually your turn!

For practice,

1. Make a new branch
2. Add a new file with your name in it
3. Commit it, push it to your repo, to the new branch
    a. `$ git push origin <newbranchname>`
4. Then we will make a PR on the next slide

Pull Requests



Make a PR!

"compare across forks"

Choose the appropriate branch

Put a blurb in there about what you did

Submit the PR

It's that easy!

- Helpful for working on projects with a lot of moving pieces.
- Kinda needed to contribute to Open Source