Goals for today

• Clone a github repo
• Create two different local branches
• Fork the repo, rebase your changes to origin, push to your fork
• Raise a pull request, so the maintainer can merge your branch into the main repo
From now on, most of these lectures will be based on some particular textbook or article

This one is based on

Cloning a github repo

• I created a repo for the purpose of this course.
• The URL is https://github.com/uiuc-sst/minicourse/
Cloning a github repo

• Just to make things interesting, we’re going to work with the “2020” branch instead of the “master” branch.

• You can see the “2020” branch by choosing it from the “branch” menu on github. But you don’t need to. What you look at on github has no effect on what’s downloaded to your PC.
Installing git

To work with git, first you need to install it. If you haven’t done so yet, please do it now.
• On ubuntu (linux or windows):
  
  sudo apt-get install git
• On apple 10.9 or greater, just type any git command, e.g.,
  
  git --version

...and if you don’t already have it, you’ll be prompted to install it.
Cloning the git repo

- cd to a directory whose subdirectory will be your local copy of the git repo, then type:
  git clone https://github.com/uiuc-sst/minicourse
  ls minicourse

- Note, you can create as many local copies as you want; delete them, move them, etc.:
  git clone https://github.com/uiuc-sst/minicourse bar
  ls bar
git log, git status

- git log tells you what are the changes most recently checked into the current branch, and by whom.
  - There are three commits: f22ff33, cad0393, and f89207f
  - The pointers HEAD, master, origin/master, and origin/HEAD all point to f22ff33.
- git status tells you if there are any changes that need to be checked in. (There aren’t: we haven’t made any changes yet).
where’s 2020?

- Remember, we want the 2020 branch.
- If we type “more README.md” to see the readme file, we find that we’re in the wrong branch.
git checkout

• Remember that we want the 2020 branch, not the master branch.
• We can change HEAD so it points to our local copy of 2020 by typing `git checkout 2020`.
• Since our local copy has not been changed, our local copy of 2020 is still the same as origin/2020.
• Now git log shows us that origin/2020 is two commits ahead of origin/master.
There’s 2020!

• ...and indeed, now “more README.md” shows the expected content.
Goals for today

• Clone a github repo

• Create two different local branches

• Fork the repo, rebase your changes to origin, push to your fork

• Raise a pull request, so the maintainer can merge your branch into the main repo
The "three trees" of git

Working directory

• The “working directory” is your normal filesystem. To see what’s there, you just type “ls”
• In other words, these are the files you’d have even without git.
HEAD: the commit

- HEAD points to a tree of commits.
- A “commit” is a complete image of all of your files. It has a compressed image of each file, stored in your directory .git/objects/pack
- The example commit shown here, 98ca9, has three files:
  - README
  - LICENSE
  - test.rb

HEAD: the tree of commits

- All of the past commits in this repository are stored, in compressed form, in the .git subdirectory.
- In addition to a complete copy of all of the files, each commit also has a pointer to its parent.
  - f30ab (“add feature #32”) has a pointer to...
  - 34ac2 (Fixed bug #1328”), which has a pointer to...
  - 98ca9 (“Initial commit”)

**HEAD: the tree of commits**

- HEAD is a pointer, pointing to one of the commits.
- This is the commit that git thinks you’re currently working on.
- In the example from my .git directory, notice
  - **HEAD** points to **master**
  - **master** points to a particular commit, in this case, **f22ff33**

index: files that have been staged for a commit

- In between HEAD and the working directory, there is a filetree called index.
- index contains a list of all of the changes that you have “staged” or “cached,” meaning that you intend to commit them, but you haven’t done it yet.

Now you try it:

- Remember that we want the 2020 branch, not the master branch.
- We can change HEAD so it points to our local copy of 2020 by typing `git checkout 2020`
- Since our local copy has not been changed, our local copy of 2020 is still the same as origin/2020
- Now git log shows us that origin/2020 is two commits ahead of origin/master.
Now you try it: checkout

• Remember that we want the 2020 branch, not the master branch.
• We can change HEAD so it points to our local copy of 2020 by typing
  
  $ git checkout 2020$

• Since our local copy has not been changed, our local copy of 2020 is still the same as origin/2020

• Now git log shows us that origin/2020 is two commits ahead of origin/master.
Now create your own local branch, so you can edit without having to coordinate with the repo maintainer

`git branch mybranch`

- Creates a new branch called “mybranch,” pointing to the same commit that HEAD currently points to.

`git checkout mybranch`

- Moves HEAD, in your local repo, so it points to mybranch
Now modify your local copy

mkdir egs/mynname
vim egs/mynname/README.md

This example subdirectory contains files written by me.
:wq

git add egs/mynname/README.md

git commit –m “created egs/mynname”
Goals for today

• Clone a github repo
• Create two different local branches
• Fork the repo, rebase your changes to origin, push to your fork
• Raise a pull request, so the maintainer can merge your branch into the main repo
Create an account

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 50 million developers.
Now you have your own page https://github.com/<username>
Go to [https://github.com/uiuc-sst/minicourse](https://github.com/uiuc-sst/minicourse) and fork the repository.
... put the new fork in your own account...
Go to your local repo, and add your new fork as a new remote

```
$ git remote -v
origin https://github.com/uiuc-sst/minicourse (fetch)
origin https://github.com/uiuc-sst/minicourse (push)

$ git remote add myfork https://github.com/<your_username>/minicourse
...where, in case it isn’t clear, please replace <your_username> with your actual github username. Thank you!

$ git remote -v
myfork https://github.com/<your_username>/minicourse (fetch)
myfork https://github.com/<your_username>/minicourse (push)
origin https://github.com/uiuc-sst/minicourse (push)
origin https://github.com/uiuc-sst/minicourse (fetch)
```
Here’s what we want to do:

```
git push mybranch to myfork
```

Ask the owner of origin to:
1. make a copy of mybranch
2. merge it with the 2020 branch

Why this can’t be done:
While you were working, origin has already merged in some changes from somebody else!
The solution: rebase

```
git pull origin 2020 --rebase
```

- “origin” specifies the remote from which you want to pull
- “2020” specifies which branch you want to pull (needs to have the same name locally as on the origin)
- “--rebase” means you want to rewrite history, so that it looks like mybranch came from the new 2020, not the old one.
Now we can:

```
git push myfork mybranch
```

jhasegaw$ git push myfork mark
Username for 'https://github.com': jhasegaw@illinois.edu
Password for 'https://jhasegaw@illinois.edu@github.com':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 465 bytes | 465.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'mark' on GitHub by visiting:
remote: https://github.com/jhasegaw/minicourse/pull/new/mark
Goals for today

• Clone a github repo
• Create two different local branches
• Fork the repo, rebase your changes to origin, push to your fork
• Raise a pull request, so the maintainer can merge your branch into the main repo
https://github.com/jhasegaw/minicourse/pull/new/mark

Change this from master to 2020, to show that you want mybranch merged into 2020.

This text will go to the owner of uiuc-sst/minicourse. If they’re not expecting your pull request, here’s your chance to explain why you think it is a good idea.
Cleaning things up

Now that the maintainer has merged mybranch into 2020, you probably want to synchronize your own local copy with his. You can also delete “mybranch,” since you don’t need it anymore.

$ git pull origin 2020
$ git checkout 2020
$ git branch -d mybranch
Here’s a list of the commands we used today

- `git clone https://github.com/uiuc-sst/minicourse`
- `git log`
- `git status`
- `git checkout 2020`
- `git branch mybranch`
- `git checkout mybranch`
- `vim egs/mynname/README.md`
- `git add egs/mynname/README.md`
- `git commit --m "added egs/mynname/README.md"`
- `git remote add myfork https://github.com/jhasegaw/minicourse`
- `git remote -v`
- `git push myfork mybranch`
- `git pull origin 2020`
- `git checkout 2020`
- `git branch --d mybranch`