Version Control Using Git and GitHub:
One Way to Better Manage Research Workflows

Kee Hyun Park

University of Maryland

GVPT Methods Workshop

September 22, 2020

Things that can mess you up during research

While conducting research, we may...

- have 9 different versions of a paper. Which is the latest?
- need to revert changes made to a data file 3 months ago. How?
- mistakenly delete an important CSV file completely
- have been exchanging ideas via email thread. Now it's over 100..
- share R scripts with coauthors. Which part did they change?

...and lots, lots more...

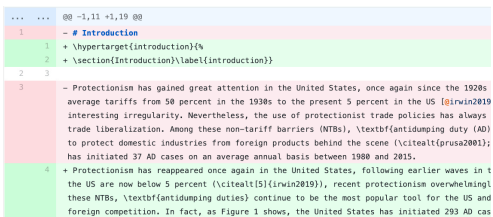
What if. . .

- we have a system that records changes to a set of **same files over time**
- so allows us to
  - revert particular file / entire project back to a previous state
  - compare changes over time
  - see who modified the contents and how
  - *without thinking much* about the file's location and contents?

$\rightsquigarrow$ exactly what version control is/does

Git: a version control system

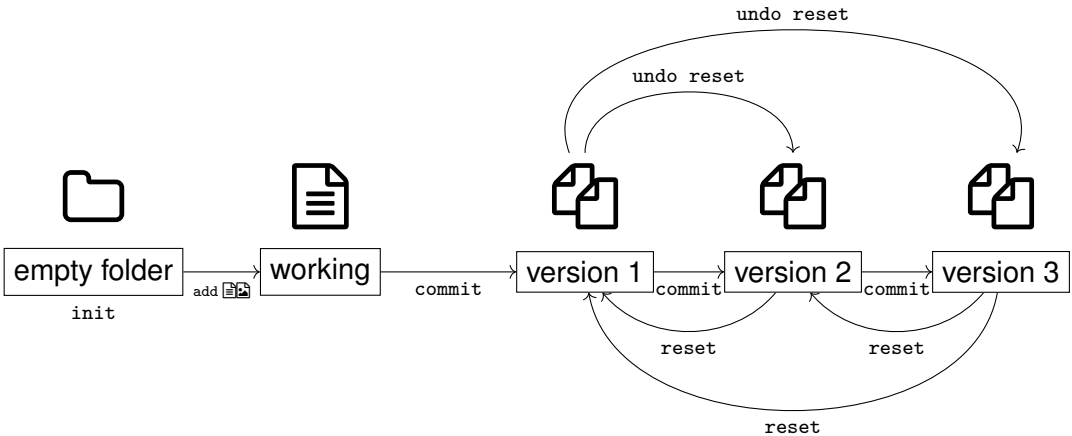
- Git is a type of software for version control (think MS Word's "Track changes")
  - takes "screenshot"s of all files $\rightsquigarrow$ come back to previous versions anytime
  - shows detailed changes in *text-based files* (.txt, .tex, .csv, .R, and much more)



Tracking changes in TEXfile

Tracking changes in R script

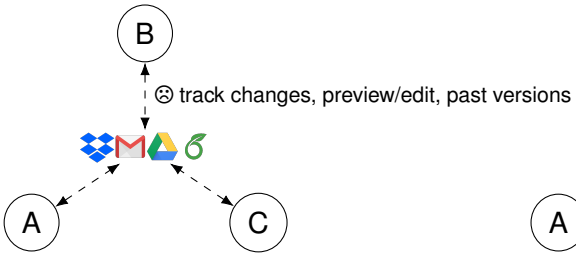
How Git works on local computer

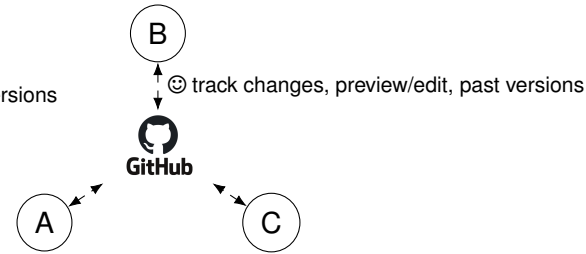


Git can take screenshots & show details of each and every step

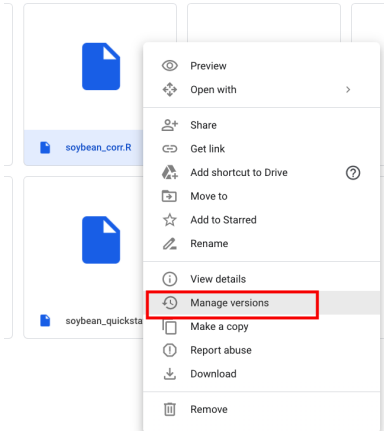
GitHub: a platform that connects git users

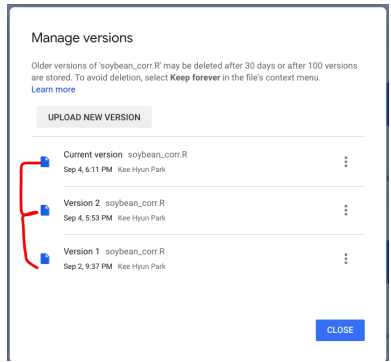
- GitHub is a git-based online platform that connects individual git users
  - think Dropbox, Google Drive, Gmail, or Overleaf





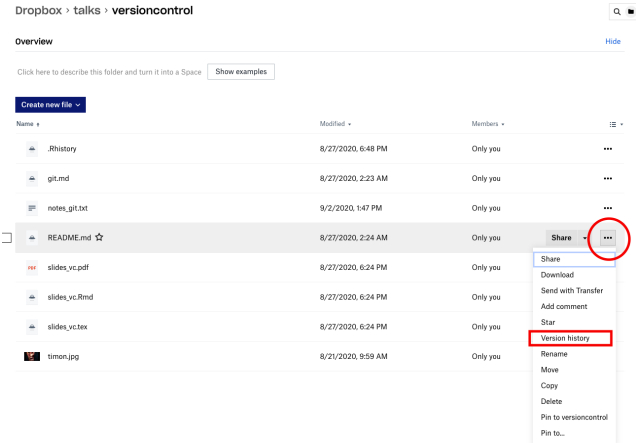
GitHub vs. Others





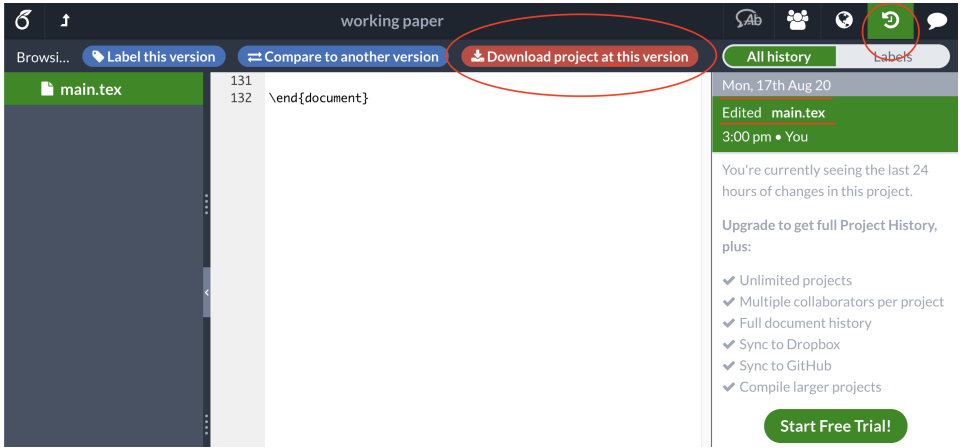
Google Drive

GitHub vs. Others



Dropbox

GitHub vs. Others



Overleaf

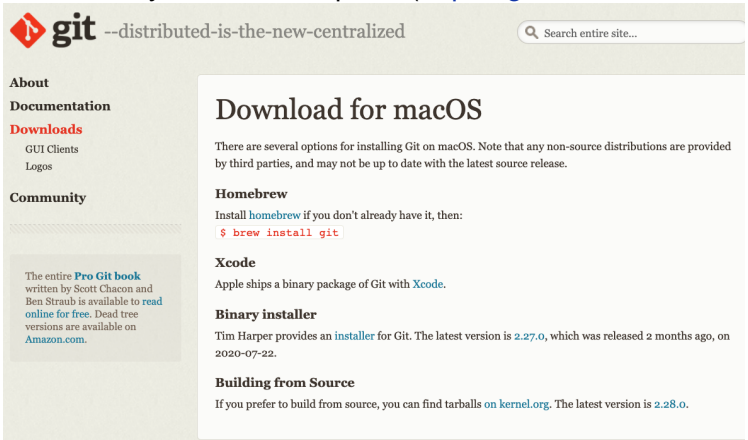
GitHub vs. Others

	GitHub	Google Drive	Dropbox	Overleaf
collaborator	∞	∞	∞	1
max file number	∞	∞	∞	2,000
max indiv. upload	100MB	50MB (text)	50GB	50MB (zip)
basic storage	1GB	15GB	2GB	∞
track changes	Yes	Google Docs	Dropbox Paper	Yes
past versions	∞	30 days	30 days	24hrs
web preview/edit	text-based files	limited	limited	TEX

Web-based version control systems (basic plan)

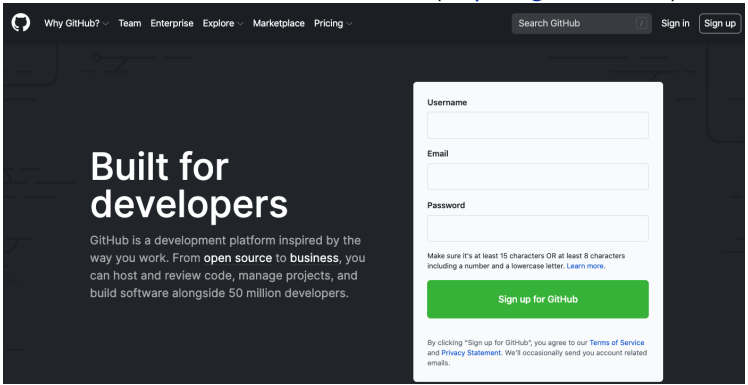
How do we use Git and GitHub?

1. Install Git on your local computer (https://git-scm.com/download)



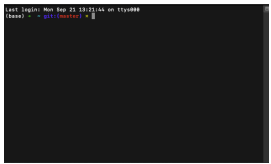
How do we use Git and GitHub?

2. Create a GitHub account (https://github.com)

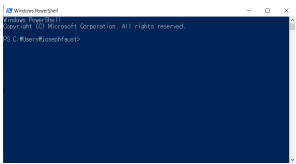


How do we use Git and GitHub?

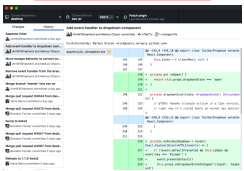
3. Choose the most comfortable mode of version control for local Git



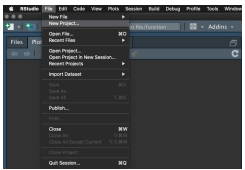
Mac Terminal



Windows PowerShell



GitHub Desktop



RStudio

How do we use Git and GitHub?

4. Initiate Git in your interested project folder
5. Write papers, codes, produce figures, but this time version control with Git

Let's give it a try

Setup Git

- Open Terminal (Mac) or PowerShell (Windows)
  - Terminal: spotlight search → type “terminal.app” + return
  - PowerShell: press “ + R” → type “powershell” + enter
- In the Terminal/Shell,

```
git config --global user.name "yourusername"
#recommend using your local computer username.

git config --global user.email "youremailaddress@someproducer"
#recommend using email that you used during GitHub sign-up.
```

Assign your project folder and initiate Git

- In the Terminal/Powershell,

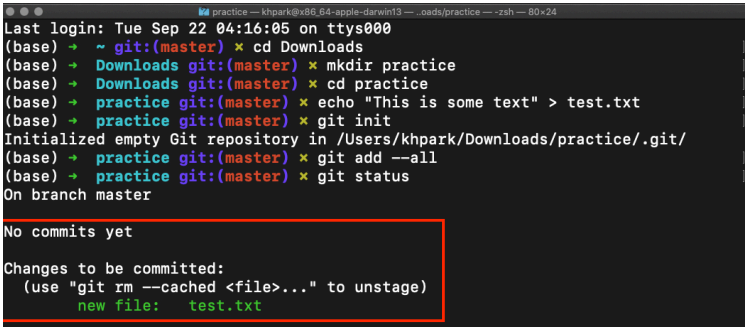
```
cd Downloads
#meaning: change directory (move) to "Downloads" folder.

mkdir practice
#meaning: make directory (folder) "practice" under Downloads.

cd practice
#meaning: move to "Downloads/practice".

echo "This is some text" > test.txt
#meaning: make a "test.txt" with a line "This is some text" written.

git init
#meaning: initate git at the current "Downloads/practice" folder.
```

- Now you should have a hidden “.git” folder created under “Downloads/practice”

Record your progress

- In the Terminal/Shell,

```
git add --all
#meaning: temporarily record all changes that I made so far.

git status
#meaning: check changes made so far, before confirming new version.
```



Confirm changes and make new version of work

Type:

```
git commit -m "first commit"
#meaning: confirm new version and add a tag "first commit".
```

- What you write for this tag is *very* important. The “future you” will look at these individual tags and figure out what each version actually contains.

Make second/third version

Type:

```
echo "another line" >> test.txt
git add --all
git commit -m "second commit"

echo "Yet another" >> test.txt
git add --all
git commit -m "third commit"
```

Check detailed changes

```
git diff HEAD^^ HEAD test.txt
#meaning: show diff between previous (HEAD^^) and current (HEAD) versions.
```

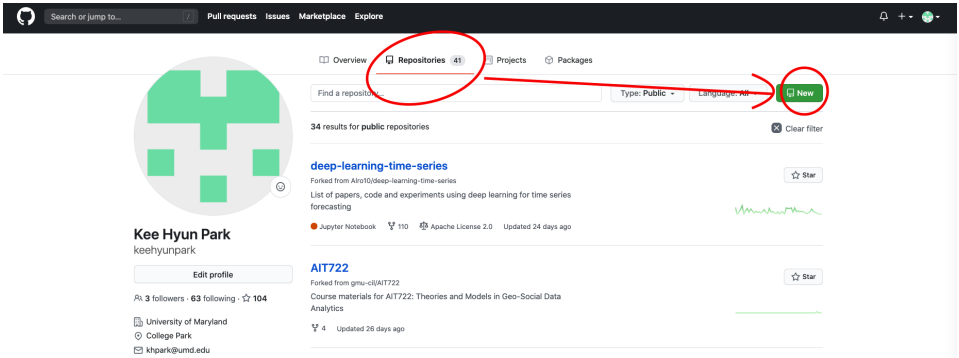
Revert changes

```
git reset HEAD~2
#meaning: Reset to the first version.

git reset 'HEAD@{1}'
#meaning: Cancel reset.

git log
#Try 'git log' to see if reset / undo reset really worked.
```

Local project folder $\Rightarrow$ GitHub

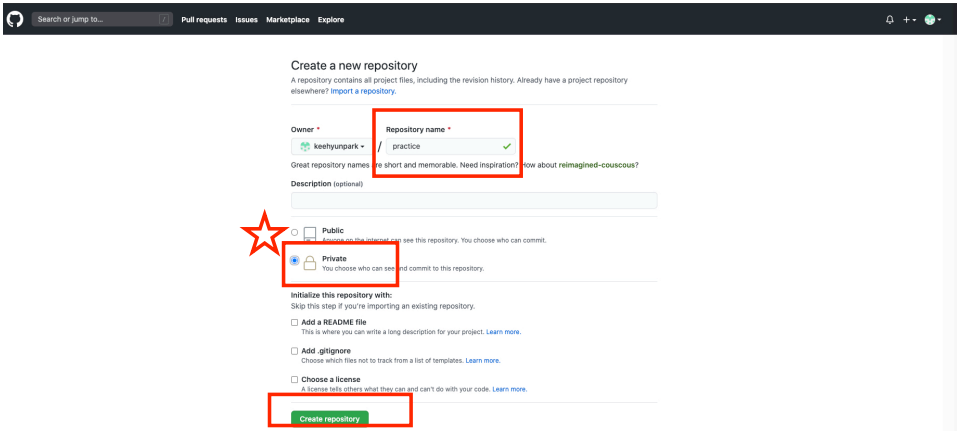
- We have "practice" folder with "third committed" test.txt file.
- Let's sync this with GitHub
- Go to https://github.com/yourusername and sign-in

Local project folder ⇒ GitHub



For example, https://github.com/keehyunpark

Local project folder ⇒ GitHub



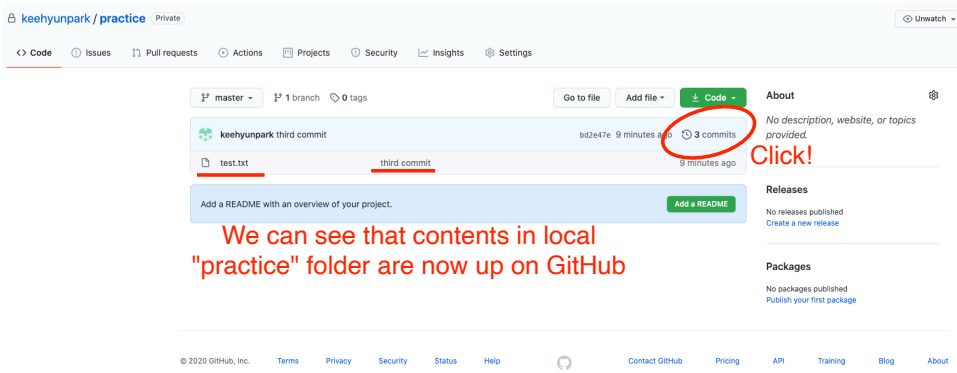
Local project folder ⇒ GitHub

- Now, back to the Terminal/Shell,

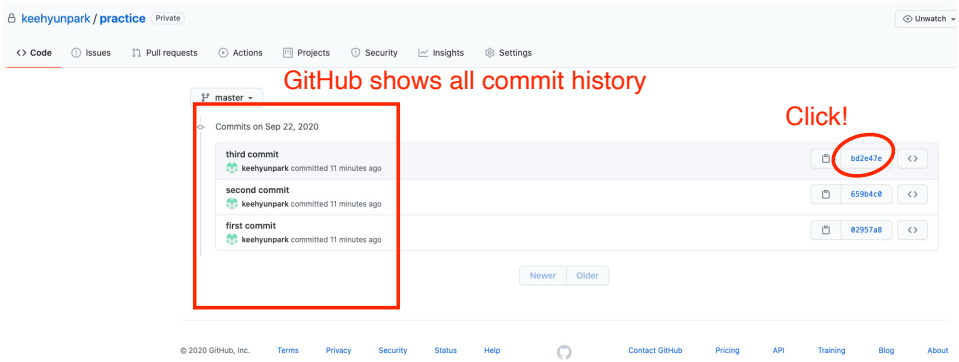
```
git remote add origin https://github.com/yourusername/practice
#meaning: connect "Downloads/practice" to GitHub repo (origin).

git push origin master
#meaning: push "Downloads/practice" (master) to GitHub (origin).
```

- Refresh browser or visit https://github.com/yourusername/practice again

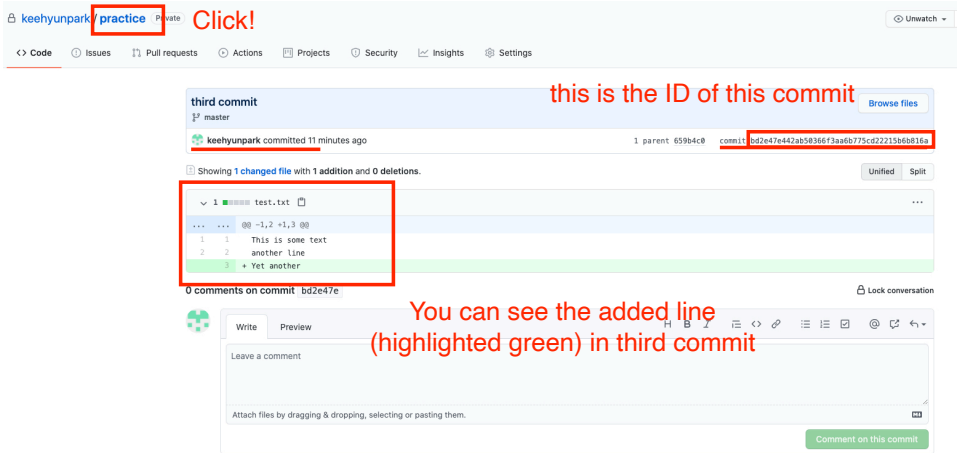
Local project folder ⇒ GitHub



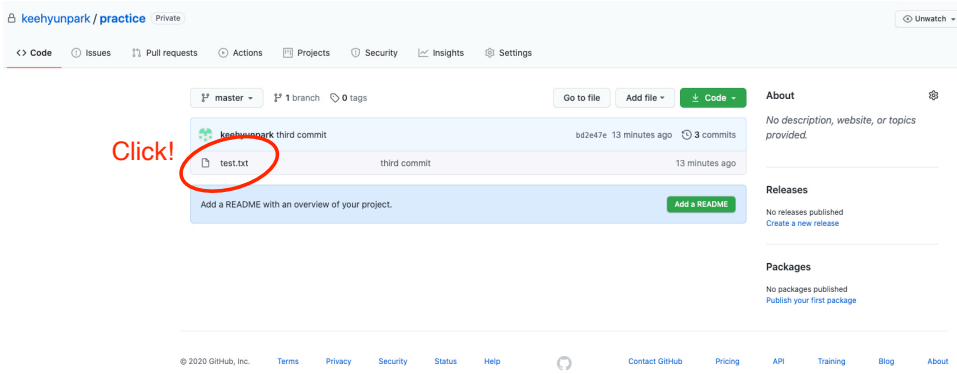
Local project folder ⇒ GitHub



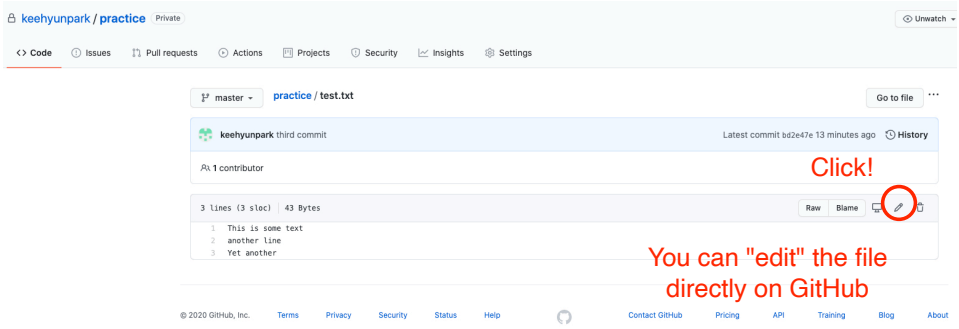
Local project folder ⇒ GitHub



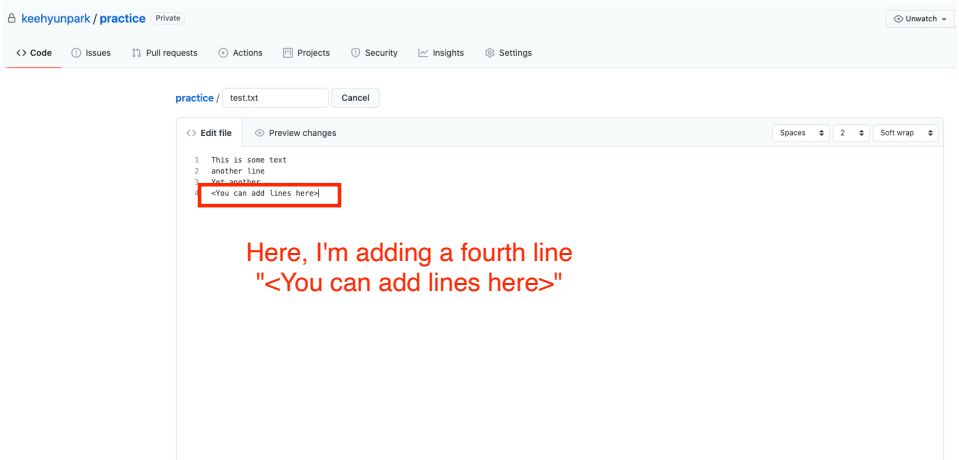
Local project folder ⇒ GitHub



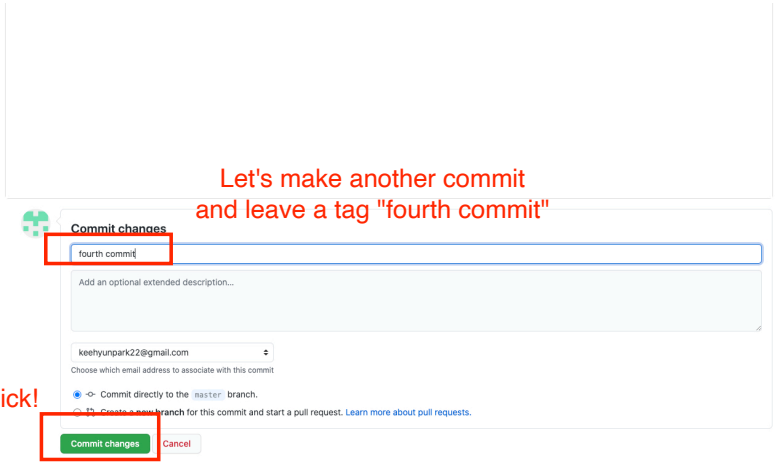
Local project folder ⇒ GitHub



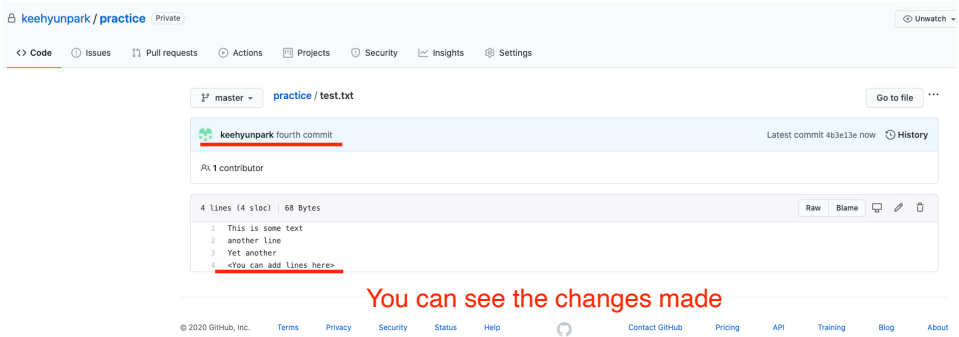
Local project folder ⇒ GitHub



Local project folder ⇒ GitHub



Local project folder ⇒ GitHub



Local project folder ⇒ GitHub

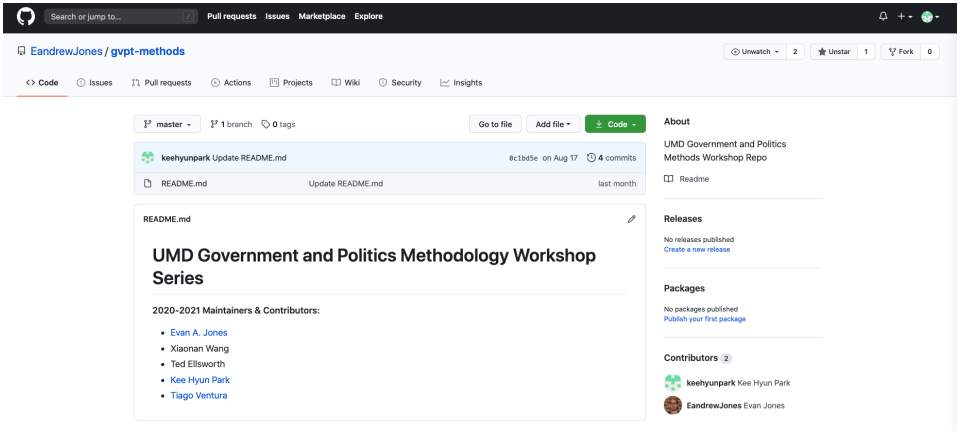
- We just made a "fourth commit" on GitHub repo. We can now "pull" this change to our local "Downloads/practice" folder.

```
git pull origin master
#meaning: Pull origin (GitHub repo) to master (local "practice")
```

Local project folder $\Leftarrow$ GitHub

- What if we we want to copy someone's GitHub repository to a local folder?
- Let's try cloning https://github.com/eandrewjones/gvpt-methods

Local project folder ⇐ GitHub



Goal: copy this folder to local Downloads folder

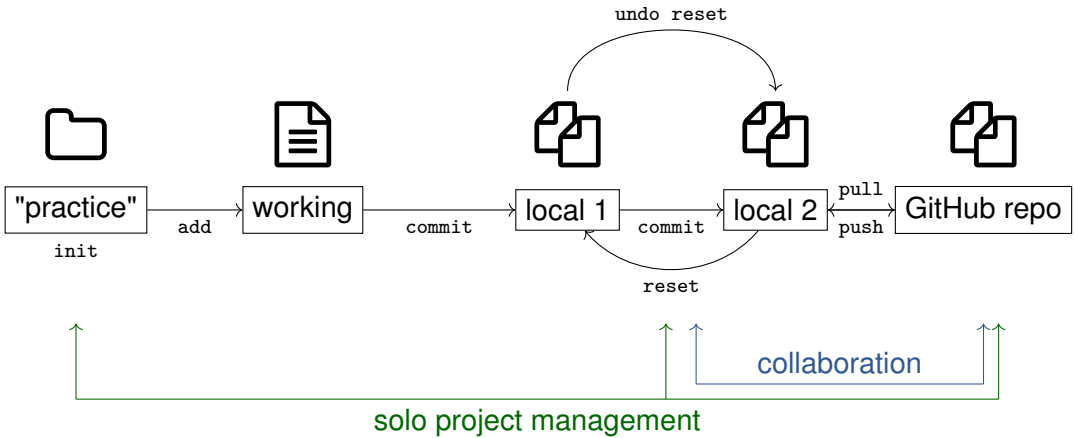
Local project folder ⇐ GitHub

- In the Terminal/Shell,

```
cd ..
#meaning: move one ladder up, to "Downloads" folder.
#Be careful! Should place a space between "cd" and ".."

git clone https://github.com/eandrewjones/gvpt-methods
#meaning: copy GitHub repo to local "gvpt-methods" folder.
```

- You'll notice that "Downloads/gvpt-methods" has been created

Summary


undo reset
"practice"
working
local 1
local 2
GitHub repo
init
add
commit
commit
pull
push
reset
collaboration
solo project management

Concluding Remarks

In this talk:

- introduce Git and GitHub as a mode of workflow management
- show that both are good candidates for solo/collaborative work
- particularly useful for tracking detailed changes, managing past versions

But:

- Git and GitHub *may* work best for empiricists who use programming languages (R, Python, julia..) to deal with massive/various data types (XML,JSON..)
- Jargons, jargons. . . barrier to entry
- Other popular modes (e.g. MS Word + emails / Google Drive + @) are as much as effective

Recommend learning! You can always come back to the materials.

- Slides and other materials available at:

  https://github.com/EandrewJones/gvpt-methods

- Follow-up questions:

  khpark@umd.edu