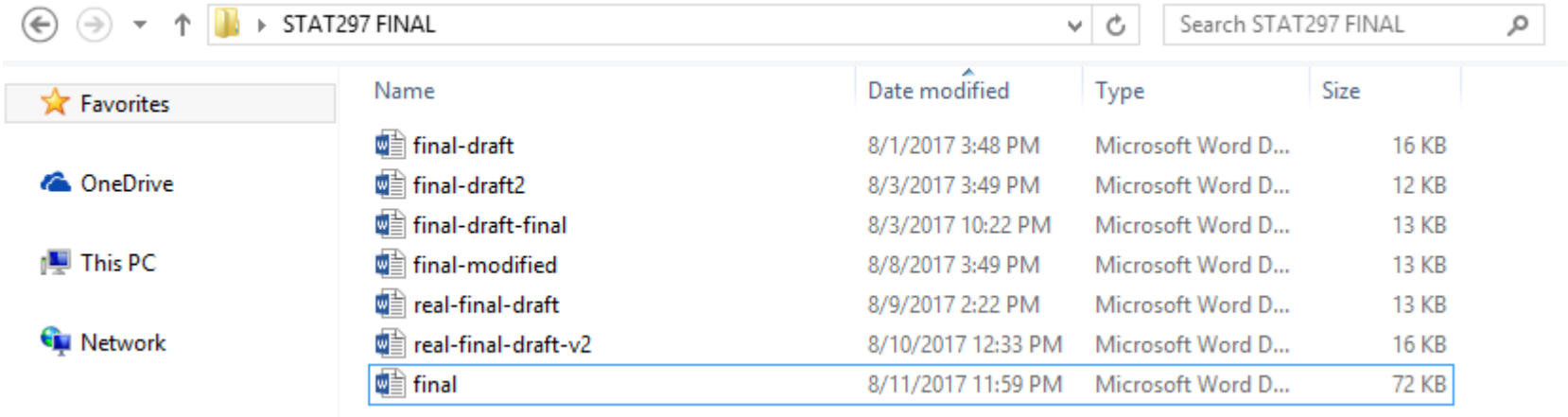
Welcome to Git

Intro to Git and Git Fundamentals

FLATIRON SCHOOL

The need for Git



Differences between all these versions?

Was real-final-draft's predecessor:

- final-draft2?
- final-modified?

Want a **history** of revisions.

The need for Git

Want a **history** of revisions.

If we make a big mistake:

- Where is the booboo?
- What file/file version did the booboo rear its ugly head?

Revert state to point before mistake was made.

The need for Git

Want a **history** of revisions.

Gets very important in collaboration:

Real world scenario

Attila, Genghis, and Odoacer: building ML predictor.

Is your kingdom ready to fall?

- Attila: wants to rebuild some data cleaning processes.
- Odoacer: still working on fine tuning predictor algorithm.
- Genghis: using current model in deployment.

Don’t wanna make Genghis mad!

The need for Git

Want a **history** of revisions.

Gets very important in collaboration:

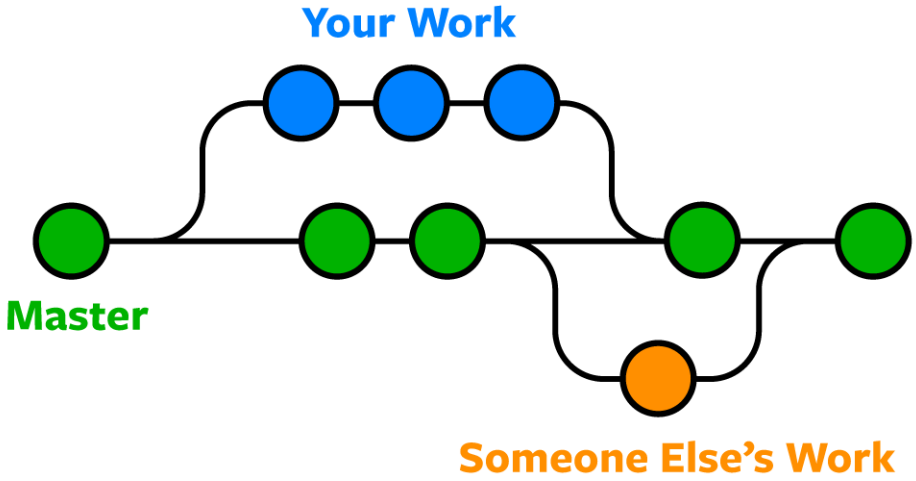
Workflow

- Atilla and Odoacer work on their contributions in their own project space.
- Combine their work: resolve any conflicts.
- Combine with original.

- Uh-oh…Atilla's work has created a problem. Genghis starts getting mad.

Undo Atilla's changes to original, keep Odoacer's. Phew! Close call.

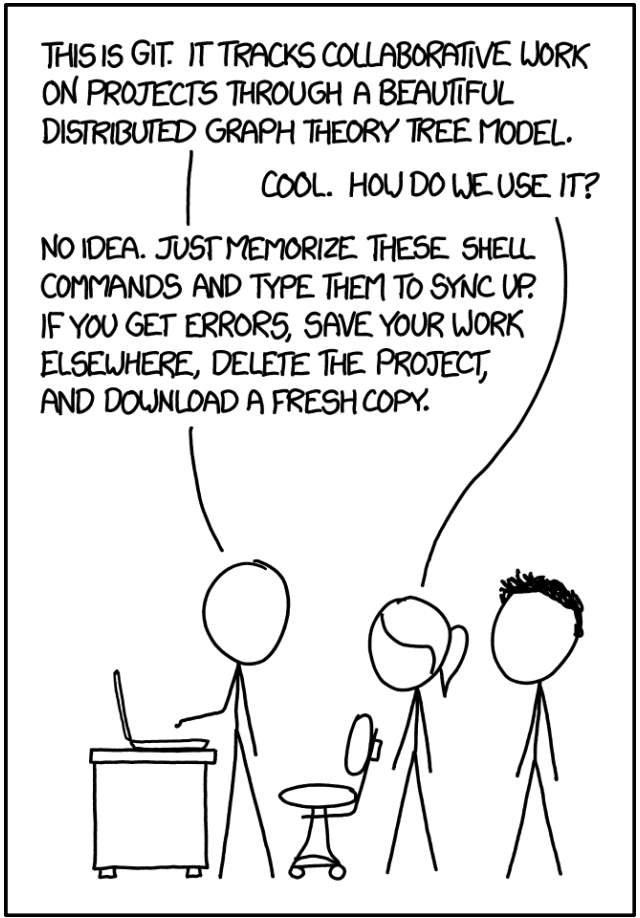
What is Git?



Each node of graph: a tracked version of a project (commit).

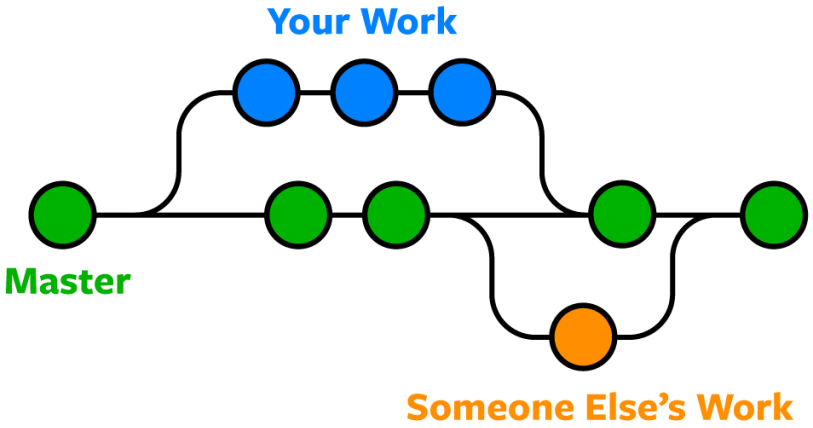
Keeps track of history: branch offs, modifications, etc.

Merging between branches and resolving conflicts.



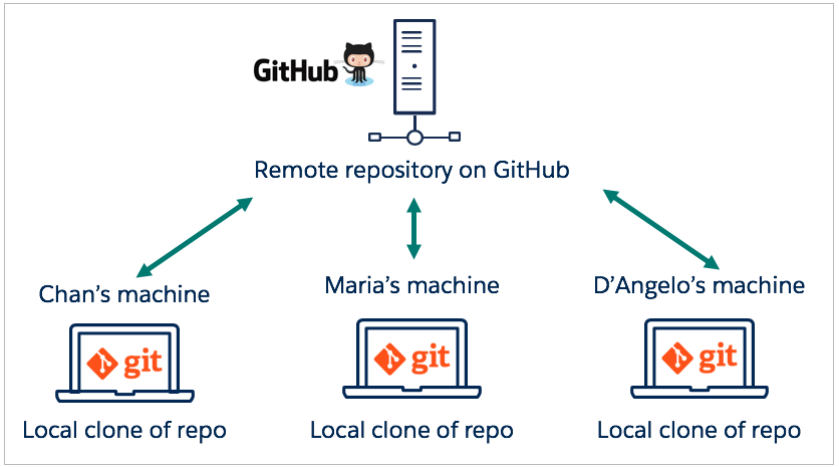
Git and Github

Git



Project (repository) history/workflow management.

Github



Cloud based repository hosting. Designed with Git workflow in mind.

Git and Github

Git

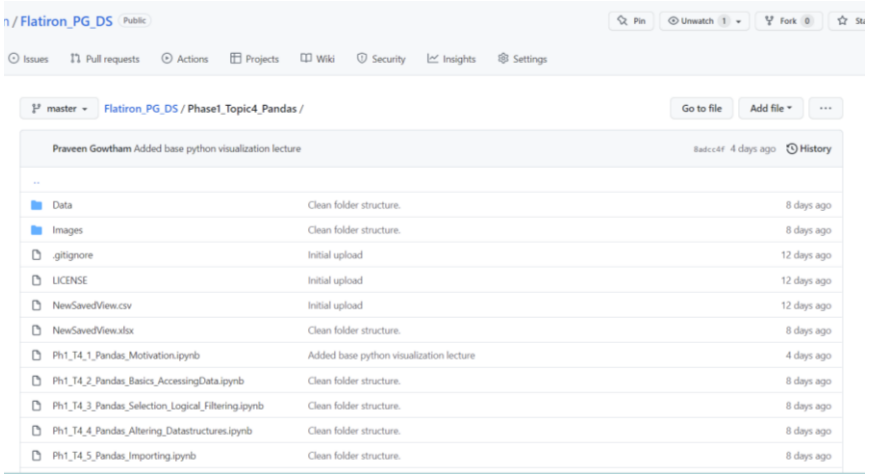
```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Flatiron_Lectures (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Phase1_Top1_Intro/MacOS_ Anaconda Installation Step-by-Step.pdf
        Phase1_Top1_Intro/Windows_ Anaconda Installation Step-by-Step.pdf
        Phase1_Top1_Intro/Windows_ Git Installation Step-by-Step.pdf
        Phase1_Topic2_TerminalGit/

nothing added to commit but untracked files present (use "git add" to track)
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Flatiron_Lectures (master)
$ git add *
```

Track and commit changes (maintaining workflow graph).

Can submit changes to files **and** workflow graph to Github.

Command line.

Github



Files/workflow stored in common remote repository.

Interact via Graphical User Interface.

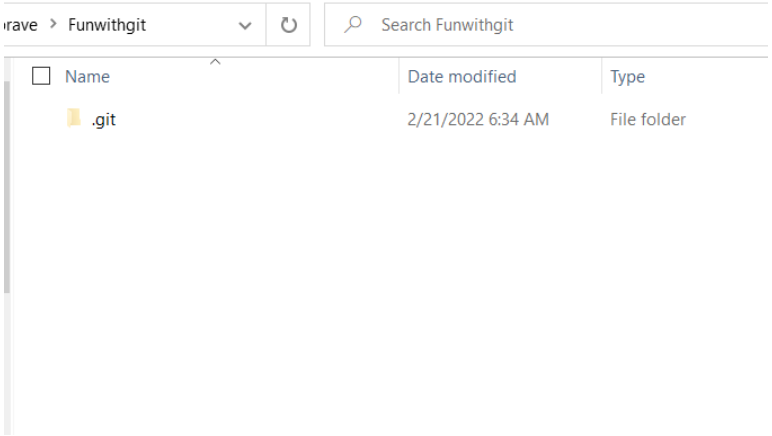
Git/Github: steep learning curve

Basically what we are going to do.

git init

Make local directory a git repository.

Hidden .git folder



Stores workflow graph info.

Github: make a remote repository

Praveen Gowtham
admveen

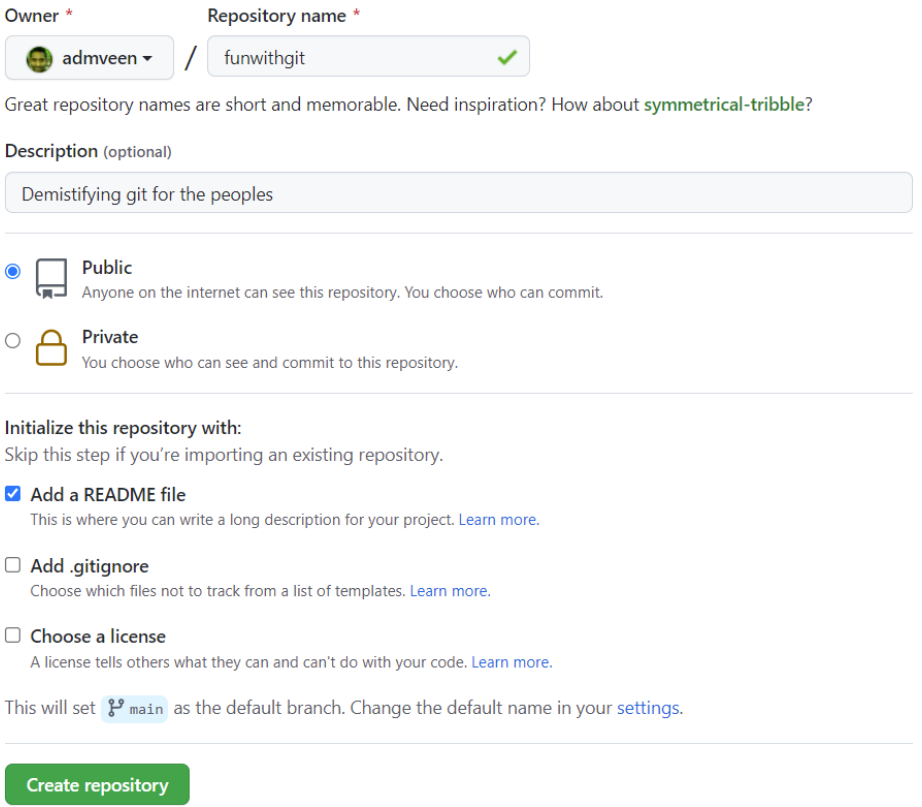
Overview | Repositories 28 | Projects | Packages | Stars

Find a repository... | Type | Language | Sort | New

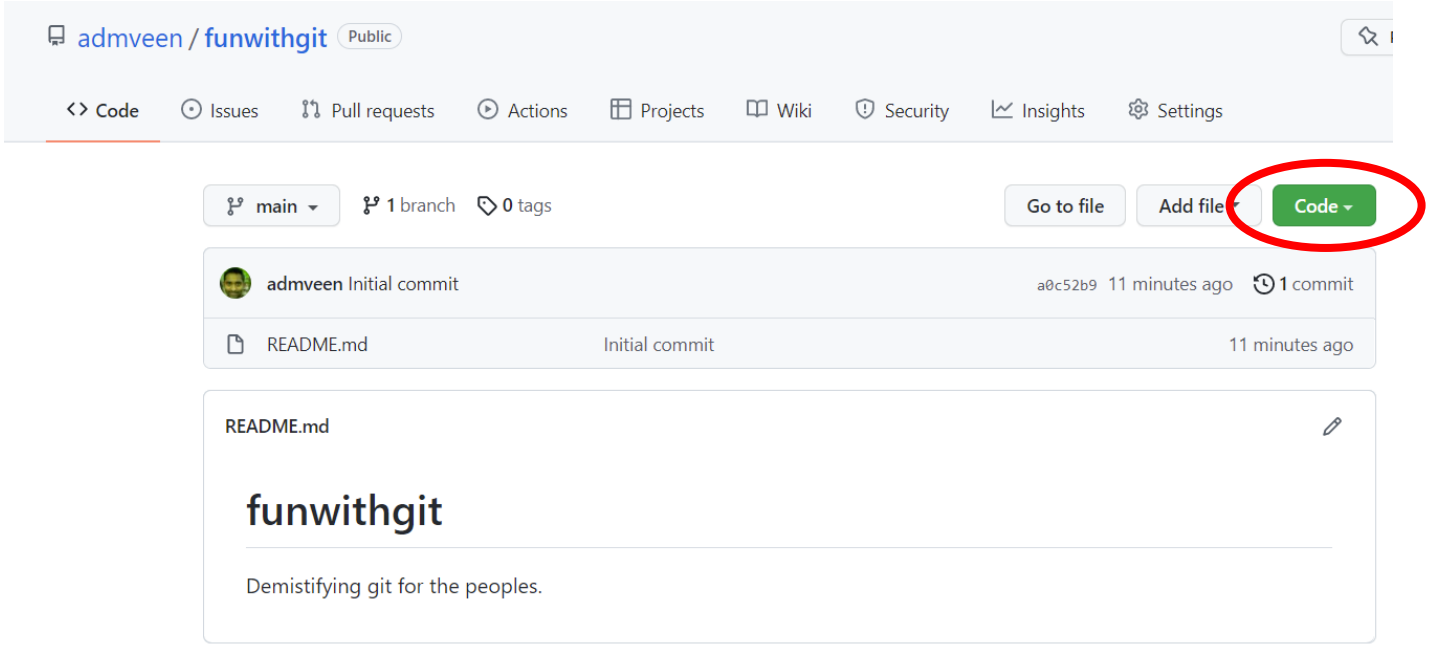
Flatiron_PG_DS Public
Lectures for Flatiron's DS LIVE NYC 2022
Jupyter Notebook 1 Updated 15 hours ago
Star

NMF_tutorial Public
Some tweet topic modeling with NMF
Jupyter Notebook 1 Updated 20 days ago
Star

Github: make a remote repository

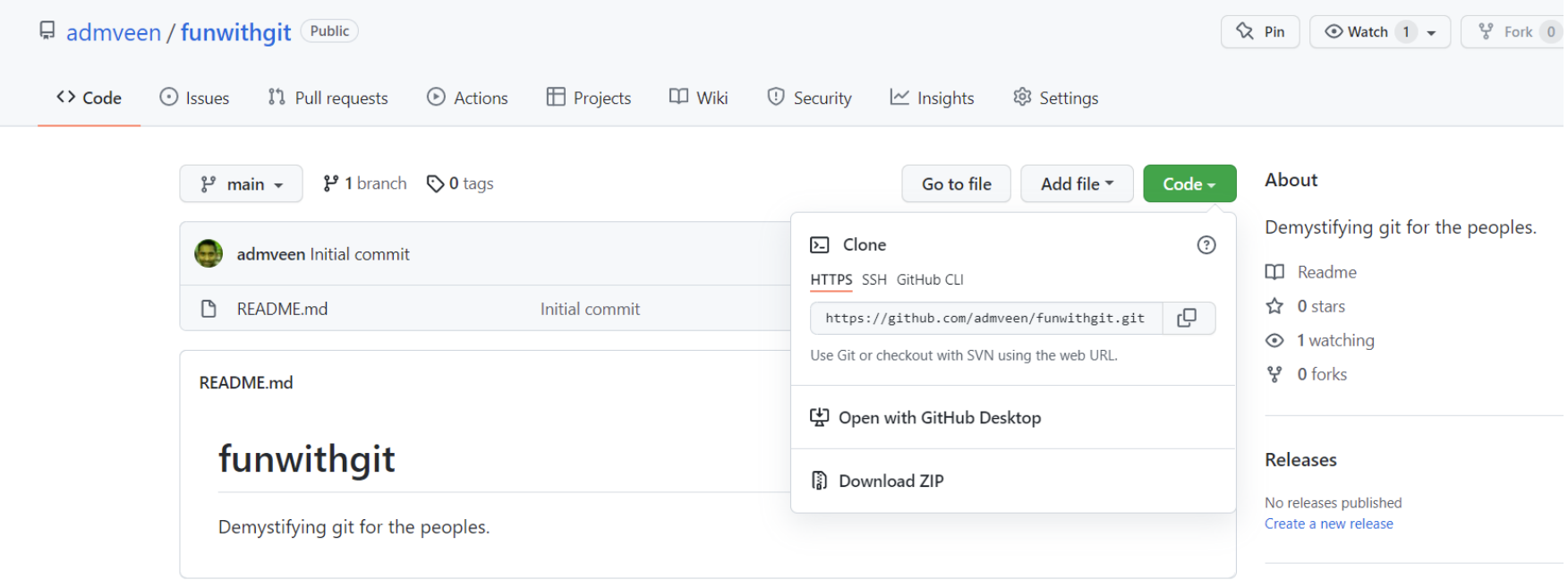


Github: make a remote repository



Contains address to remote repository

Github: make a remote repository



Link git to github: git remote

git remote add remote_name remote_address

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit
$ git init
Initialized empty Git repository in C:/Users/prave/Funwithgit/.git/
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ ls
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git remote add origin https://github.com/admveen/funwithgit.git
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$
```

Git repository (local)
and Github remote repo now linked!

git remote -v

```
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git remote -v
origin  https://github.com/admveen/funwithgit.git (fetch)
origin  https://github.com/admveen/funwithgit.git (push)
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$
```

Made some changes

I created a new Jupyter notebook:

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ ls
naivefibonacci.ipynb
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$
```

jupyter naivefibonacci Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Markdown

Fibonacci (Naive)

Implement the Fibonacci sequence with recursion naively. Element sum of previous two elements.

0,1,1,2,3,5,8,13,21,34...

In [3]:
```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)
```

In [4]:
```
print(fib(8))
```
```
21
```

In [5]:
```
calc_time = %timeit -n 500 -r 7 -o fib(8)
```
```
10.4 µs ± 1.56 µs per loop (mean ± std. dev. of 7 runs, 500 loops each)
```

In [9]:
```
print(fib(18))
```
```
2584
```

In [10]:
```
calc_time = %timeit -n 500 -r 7 -o fib(18)
```
```
1.17 ms ± 25.6 µs per loop (mean ± std. dev. of 7 runs, 500 loops each)
```

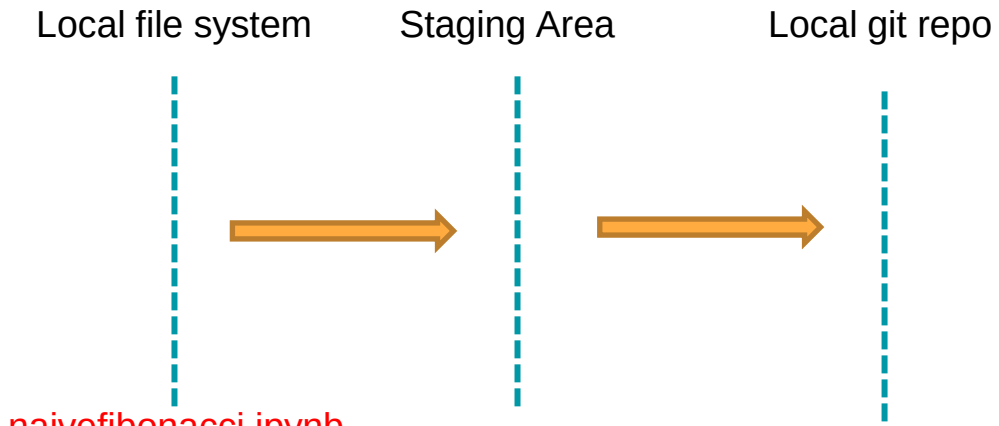
Yikes! Not scalable. I need to change this!

Make change part of history

File created (in folder)
Not yet in workflow graph (git repo)

```
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git log
fatal: your current branch 'master' does not have any commits yet
```

Want to commit changes: add to graph/make part of our history.



git status

Compares files in directory to local git repo:

- Are there changes? Which files?
- Are they in staging area or not (tracked/untracked)?

```
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .ipynb_checkpoints/
        naivefibonacci.ipynb

nothing added to commit but untracked files present (use "git add" to track)
```

git add

git add .

This adds all untracked changes to staging area

git add filename

Adds file with untracked changes to staging area

```
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git add *
warning: LF will be replaced by CRLF in naivefibonacci.ipynb.
The file will have its original line endings in your working directory
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   naivefibonacci.ipynb

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .ipynb_checkpoints/
```

git commit

When ready: add changes to the official record/history.

Transfer changes from staging area to the git graph / repo.

git commit –m “Message”

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git commit -m "Add naive implementation of Fibonacci"
[master (root-commit) 3bc1064] Add naive implementation of Fibonacci
 1 file changed, 135 insertions(+)
 create mode 100644 naivefibonacci.ipynb
```

git log (see history of graph)

```
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git log
commit 3bc106469af0662593Off9fa6ac0762f1170fee3 (HEAD -> master)
Author: Praveen Gowtham <praveen.gowtham@flatironschool.com>
Date:   Mon Feb 21 12:53:04 2022 -0500

    Add naive implementation of Fibonacci
```

commit messages

Commit messages are important:
- Tells you/others what commit is about

git commit –m “meaningful stuff”

Rule 1: One line description.
Rule 2: Should be meaningful.

Style guideline:

Start with a verb in the present tense, imperative mood.

“Implement dynamic programming solution.”

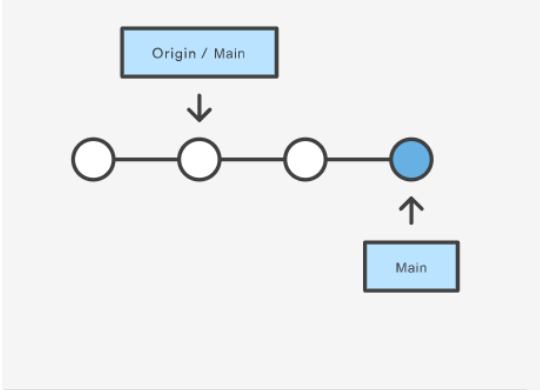
git push

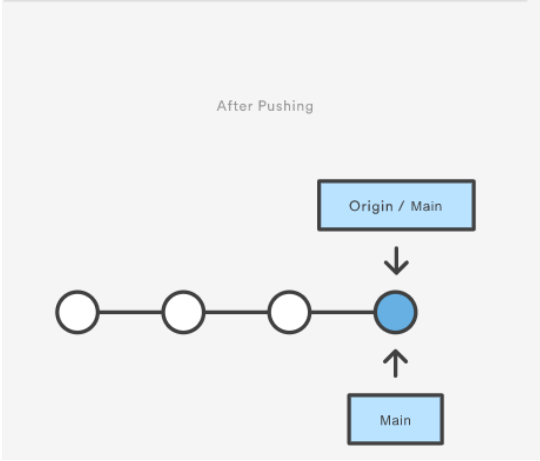
Previously: linked local git to github remote repo.

Now: send new changes + git history to remote repo/branch.

git push remote_name local_branch

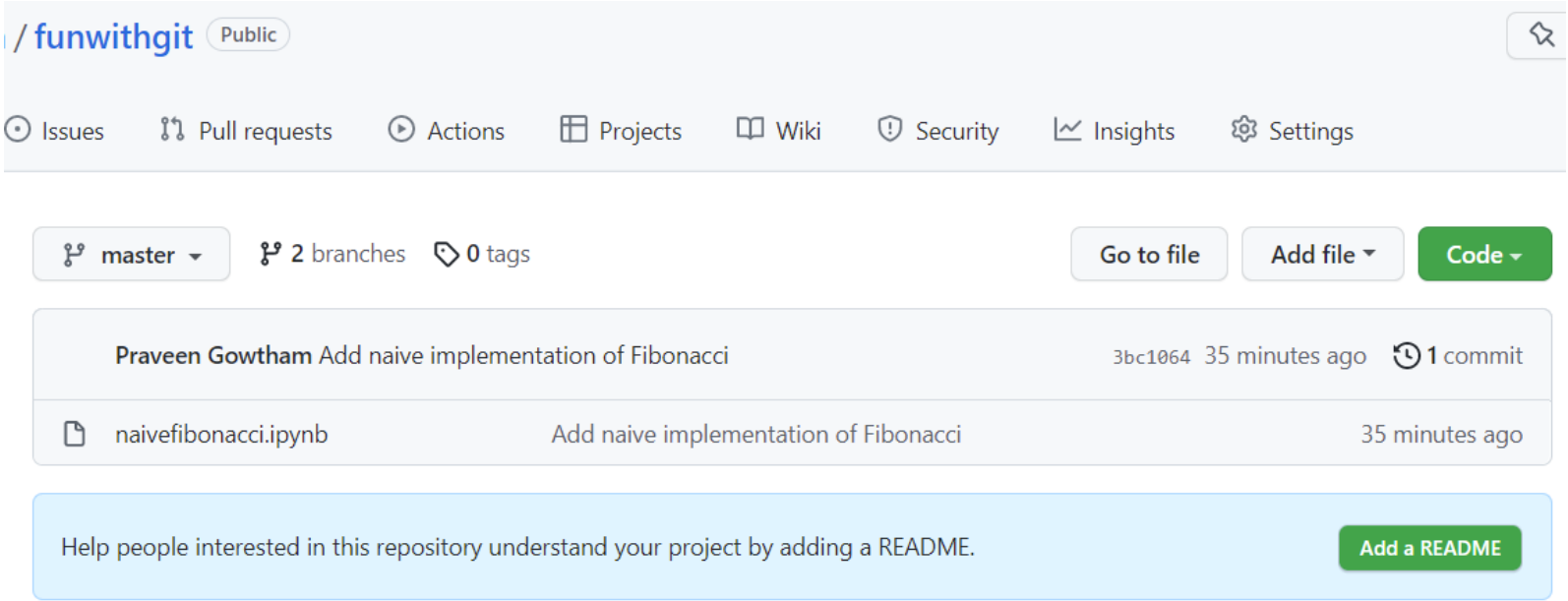
```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git remote -v
origin  https://github.com/admveen/funwithgit.git (fetch)
origin  https://github.com/admveen/funwithgit.git (push)
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$
```





git push

Bam! There it is on github.



Make better implementation.

- Created new one (good_fibonacci).
- Deleted old one (naivefibonacci).
- Also created new text file.

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ ls
good_fibonacci.ipynb  untracked_data.txt
```

Fibonacci with Dynamic Programming

(a.k.a don't repeat work)

In [2]:

```
# lookup table for results
result_dict = {0: 1, 1: 1}

def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:

        # if already computed, use value in lookup table
        if (n - 1) in result_dict.keys():
            fibn1 = result_dict[n-1]
        else:
            # otherwise, compute
            fibn1 = fib(n-1)
        # if already computed, use value in lookup table
        if (n - 2) in result_dict.keys():
            fibn2 = result_dict[n-2]
        else:
            # otherwise, compute
            fibn2 = fib(n-2)

        fibn = fibn1 + fibn2

        # store new result in lookup table
        result_dict[n] = fibn

        return result_dict[n]
```

In [3]:

```
calc_time = %timeit -n 500 -r 7 -o fib(18)
```

You can be a little fancy

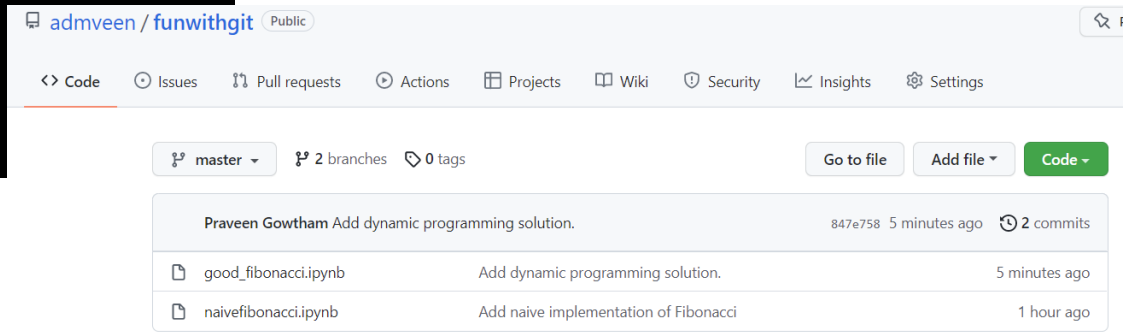
Stage/commit new one (good_fibonacci)

But:
Don’t commit delete of old implementation or textfile just yet.

```
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    naivefibonacci.ipynb

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .ipynb_checkpoints/
        good_fibonacci.ipynb
        untracked_data.txt

no changes added to commit (use "git add" and/or "git commit -a")
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git add good_fibonacci.ipynb
warning: LF will be replaced by CRLF in good_fibonacci.ipynb.
The file will have its original line endings in your working directory
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   good_fibonacci.ipynb

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    naivefibonacci.ipynb

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .ipynb_checkpoints/
        untracked_data.txt

(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
```

Push!

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git commit -m "Add dynamic programming solution."
[master 847e758] Add dynamic programming solution.
 1 file changed, 99 insertions(+)
 create mode 100644 good_fibonacci.ipynb
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.10 KiB | 1.10 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/admveen/funwithgit.git
   3bc1064..847e758  master -> master
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git log
commit 847e7582b0d7deb7bf0e3258256c2e88beb822cb (HEAD -> master, origin/master)
Author: Praveen Gowtham <praveen.gowtham@flatironschool.com>
Date:   Mon Feb 21 14:13:13 2022 -0500

    Add dynamic programming solution.

commit 3bc106469af0662593Off9fa6ac0762f1170fee3
Author: Praveen Gowtham <praveen.gowtham@flatironschool.com>
Date:   Mon Feb 21 12:53:04 2022 -0500

    Add naive implementation of Fibonacci
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$
```



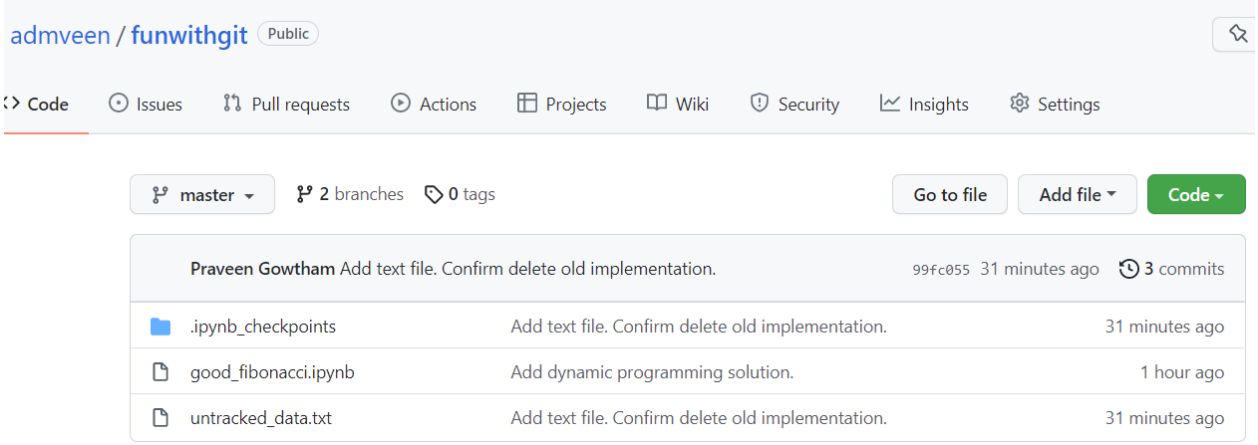
Do rest of changes

After git add . and commit:

- Commit delete naivefibonacci in repo.
- Commit add text file

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git add .
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Funwithgit (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .ipynb_checkpoints/good_fibonacci-checkpoint.ipynb
        renamed:    naivefibonacci.ipynb -> .ipynb_checkpoints/naivefibonacci-checkpoint.ipynb
        new file:   untracked_data.txt
```

Then push:



Forking/Cloning

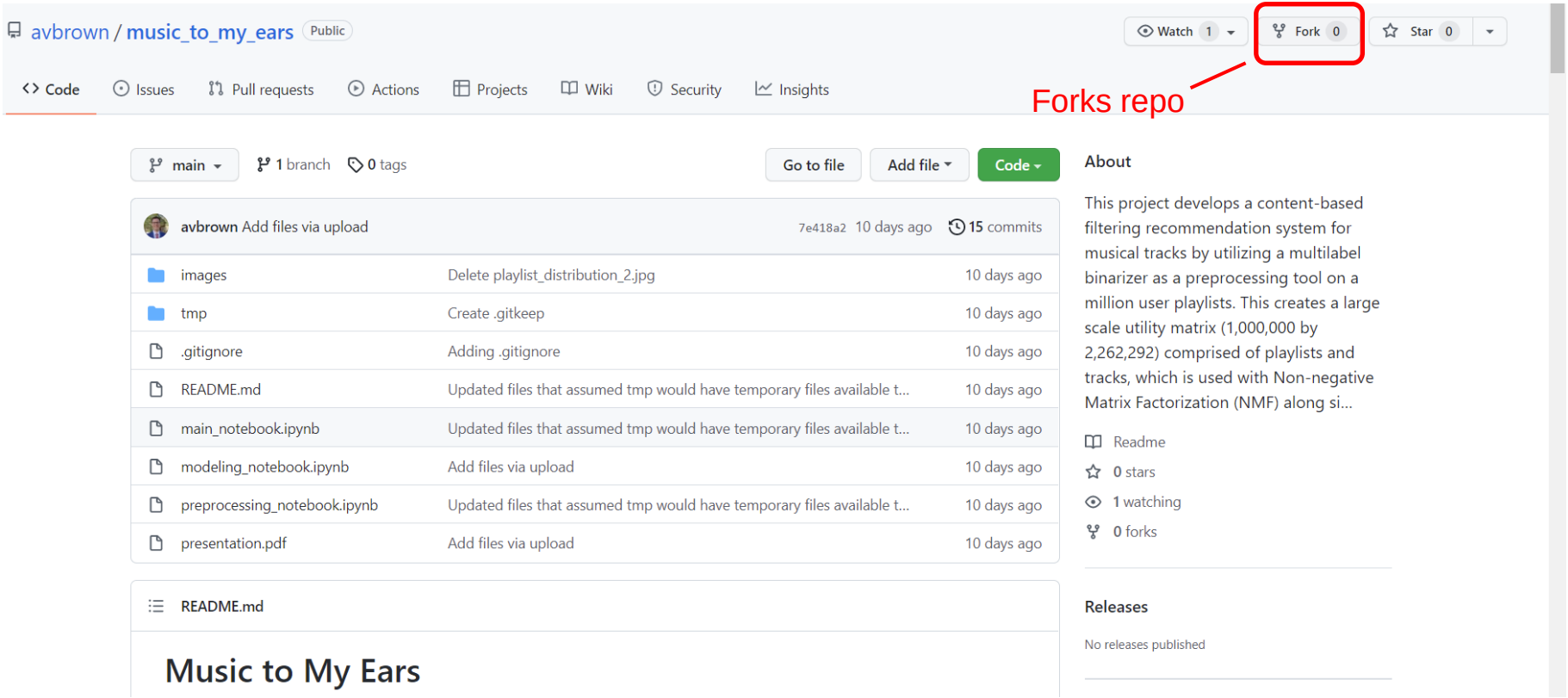
Want to work with someone else's project:

Step 1: Fork
Step 2: Clone

Forking

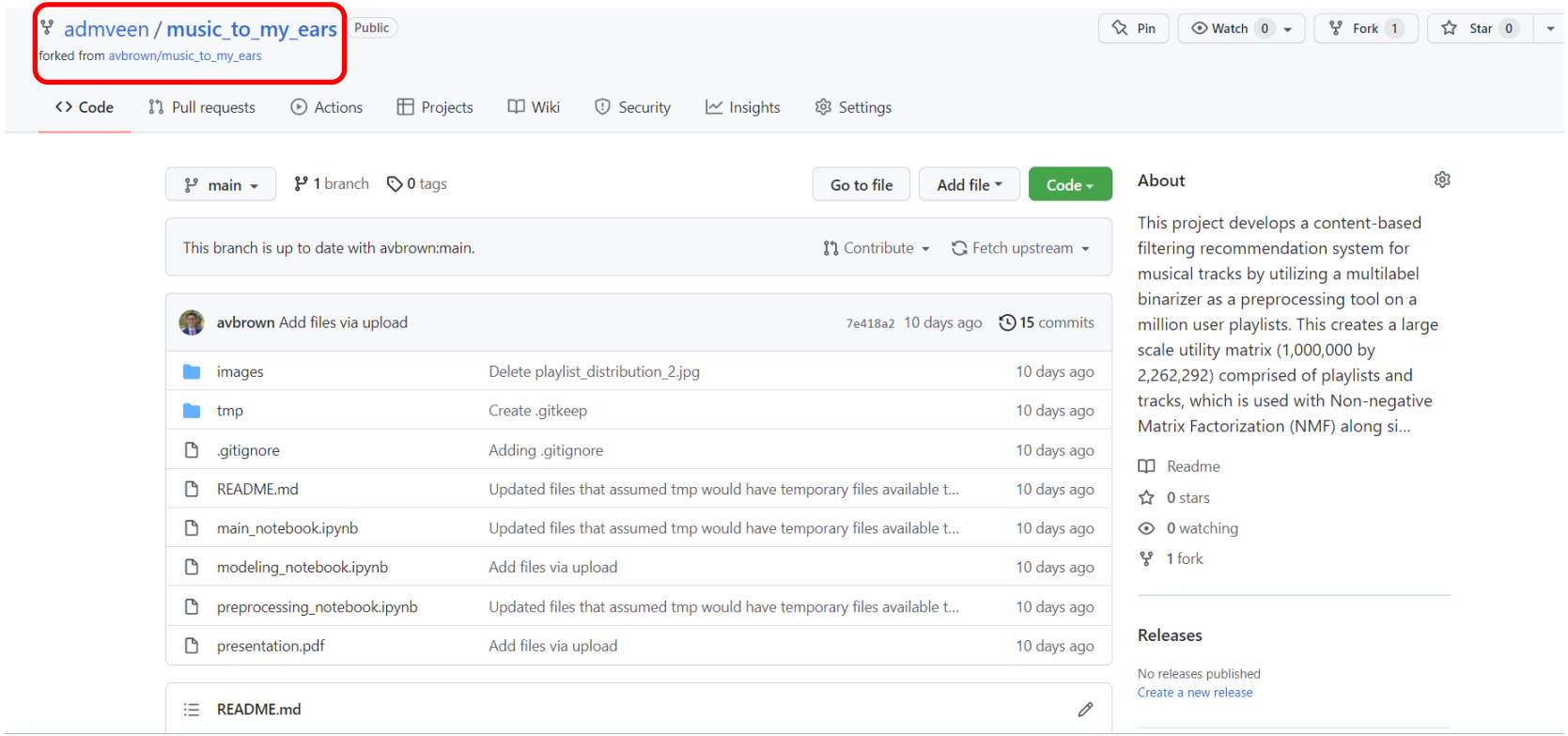
- Create copy of another's repo in your Github account.
- Any changes (commit/push) are made to ***your*** copy.
- Doesn't mess their project up!

Forking


avbrown / music_to_my_ears
Public
Watch 1
Fork 0
Star 0
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Forks repo
main
1 branch
0 tags
Go to file
Add file
Code
avbrown Add files via upload
7e418a2 10 days ago
15 commits
images Delete playlist_distribution_2.jpg 10 days ago
tmp Create .gitkeep 10 days ago
.gitignore Adding .gitignore 10 days ago
README.md Updated files that assumed tmp would have temporary files available t... 10 days ago
main_notebook.ipynb Updated files that assumed tmp would have temporary files available t... 10 days ago
modeling_notebook.ipynb Add files via upload 10 days ago
preprocessing_notebook.ipynb Updated files that assumed tmp would have temporary files available t... 10 days ago
presentation.pdf Add files via upload 10 days ago
README.md
Music to My Ears
About
This project develops a content-based filtering recommendation system for musical tracks by utilizing a multilabel binarizer as a preprocessing tool on a million user playlists. This creates a large scale utility matrix (1,000,000 by 2,262,292) comprised of playlists and tracks, which is used with Non-negative Matrix Factorization (NMF) along si...
Readme
0 stars
1 watching
0 forks
Releases
No releases published

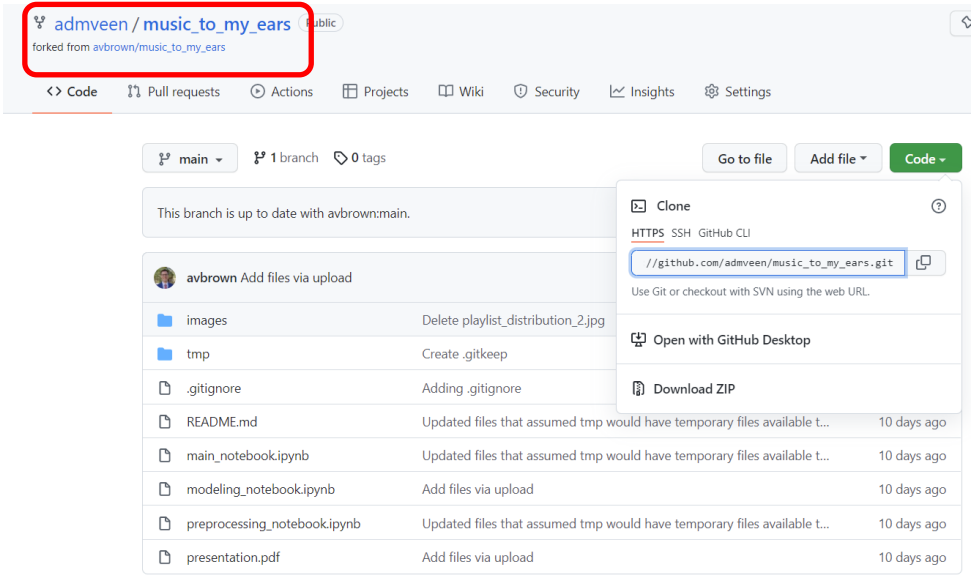
After forking

Now a copy of repo in my github



Cloning

Now a copy of repo in my github



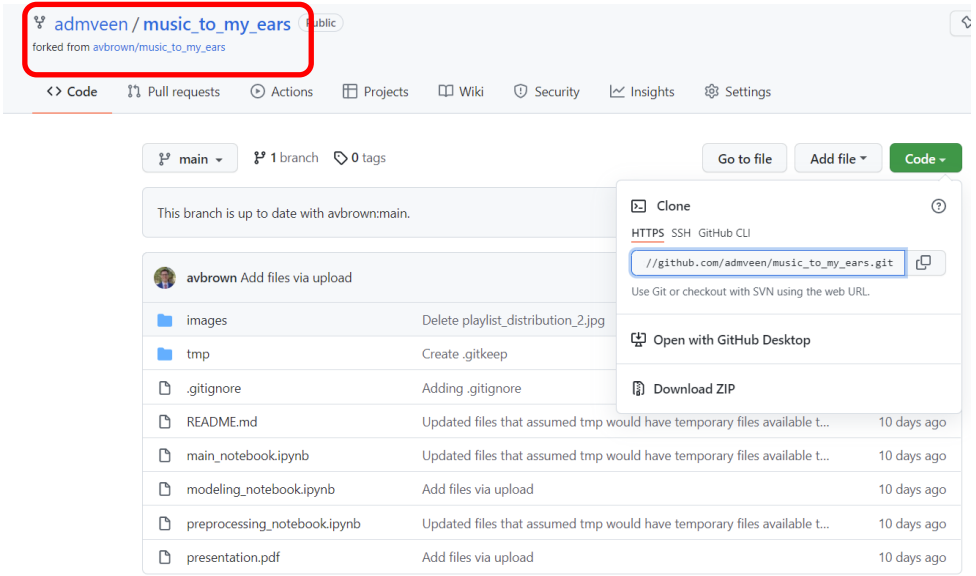
git clone address_forked_repo

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun
$ git clone https://github.com/admveen/music_to_my_ears.git
Cloning into 'music_to_my_ears'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 50 (delta 25), reused 8 (delta 5), pack-reused 0
Receiving objects: 100% (50/50), 8.78 MiB | 10.96 MiB/s, done.
Resolving deltas: 100% (25/25), done.
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun
$ ls
music_to_my_ears/
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun
$ cd music_to_my_ears
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun/music_to_my_ears (main)
$ ls
README.md  main_notebook.ipynb      preprocessing_notebook.ipynb  tmp/
images/    modeling_notebook.ipynb  presentation.pdf
```

Clones files/commit graph in local repository

git clone

Now a copy of repo in my github



git clone address_forked_repo

```
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun
$ git clone https://github.com/admveen/music_to_my_ears.git
Cloning into 'music_to_my_ears'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 50 (delta 25), reused 8 (delta 5), pack-reused 0
Receiving objects: 100% (50/50), 8.78 MiB | 10.96 MiB/s, done.
Resolving deltas: 100% (25/25), done.
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun
$ ls
music_to_my_ears/
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun
$ cd music_to_my_ears
(base)
prave@LAPTOP-JA891BKF MINGW64 ~/Cloning_Repo_Fun/music_to_my_ears (main)
$ ls
README.md  main_notebook.ipynb      preprocessing_notebook.ipynb  tmp/
images/    modeling_notebook.ipynb  presentation.pdf
```

Clones files/commit graph in local repository

Now, can work on project safely!

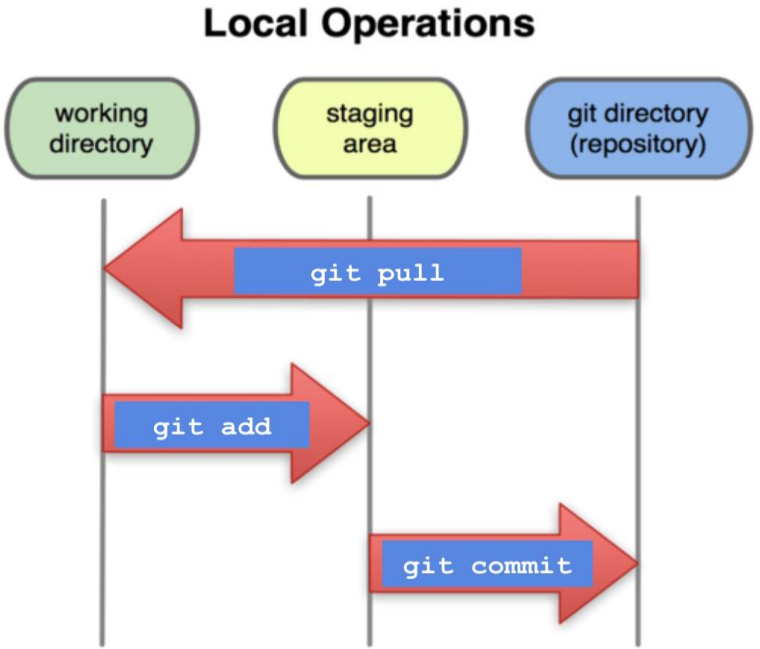
git pull

Someone pushes change to your remote repo.

You want to update your local repo.

git pull remote_name local_branch

Fetches changes in remote branch and merges in your local branch.



So…

- That was a lot.

- We’ll get some practice in subsequent labs.

- Can only learn git by doing it.

- Later: will learn collaborating with git, branch management, reverting git state.