Git & GitHub Workshop

its not only for programmers

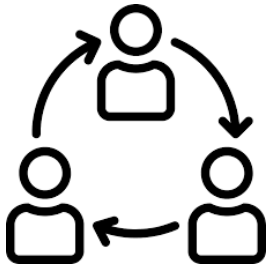
(ﾉ°□°)ﾉ ⌒ ┻━┻

Micaela Chan
11.16.2019

Why is git important for scientists?

Documentation

Sharing / Collaborating



Both are important in science !

Git facilitates both!

Why is git important for scientists?

Data collection & experiment

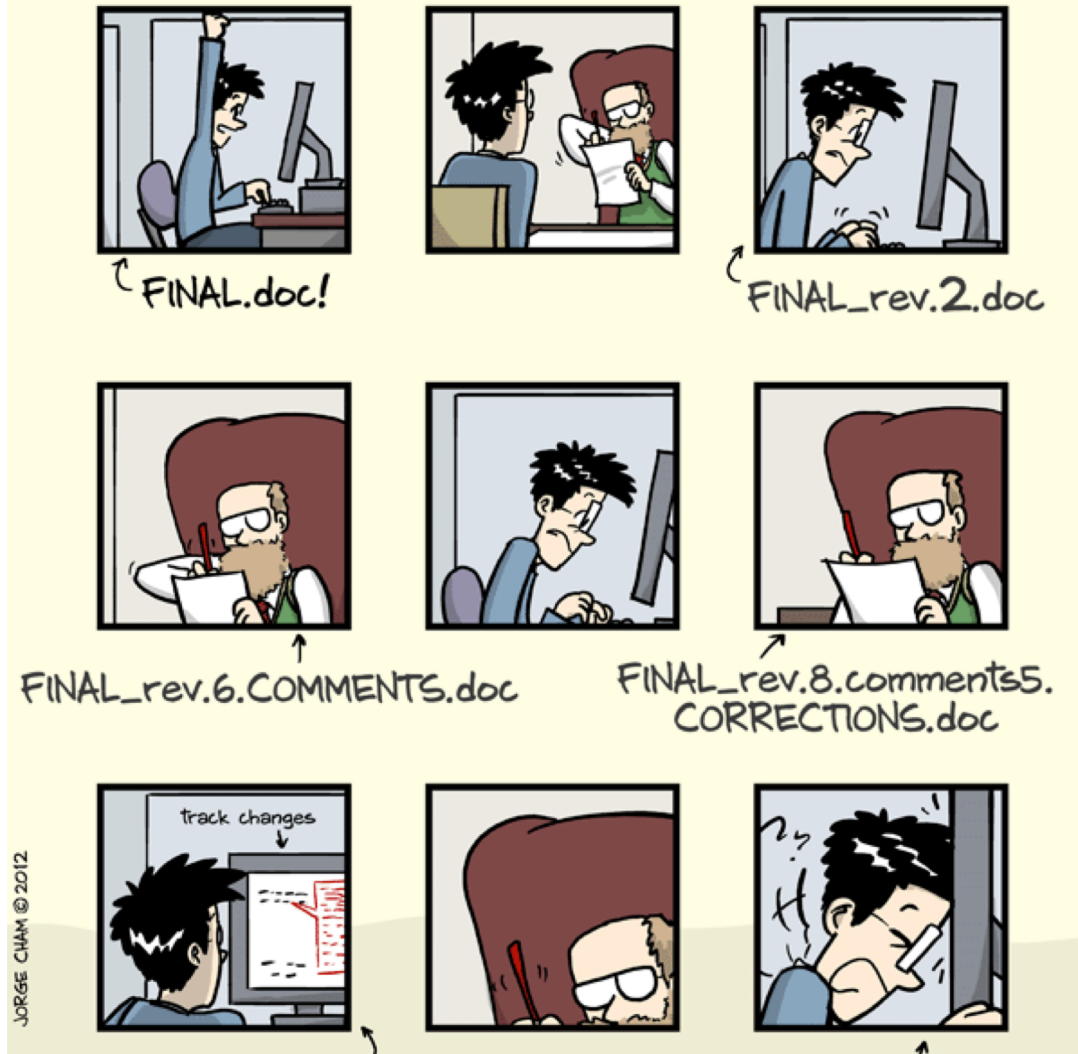
Record parameters and changes in lab notebook (paper/digital)

Data processing & analysis

?

Version control of code = lab notebook of experiment

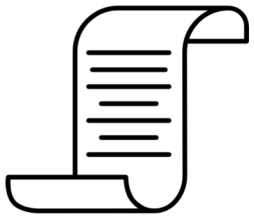
"FINAL".doc



Why is learning git important for scientists?

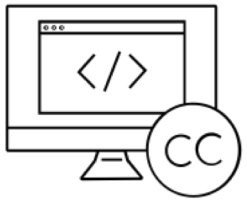
Sharing/Collaboration

- Sharing codes are difficult through traditional means



- GitHub enables code sharing, but it uses git
  - Someone can **follow** your codes
  - Incorporate or suggest changes to others (even *strangers*!)

- Encourages open source and open science
  - Open science is a huge topic **(Kendra will talk about OSF tomorrow, Nov 17th!!)**
  - Openness to your code is generally a good idea



What is the goal of this workshop?

Git for version control

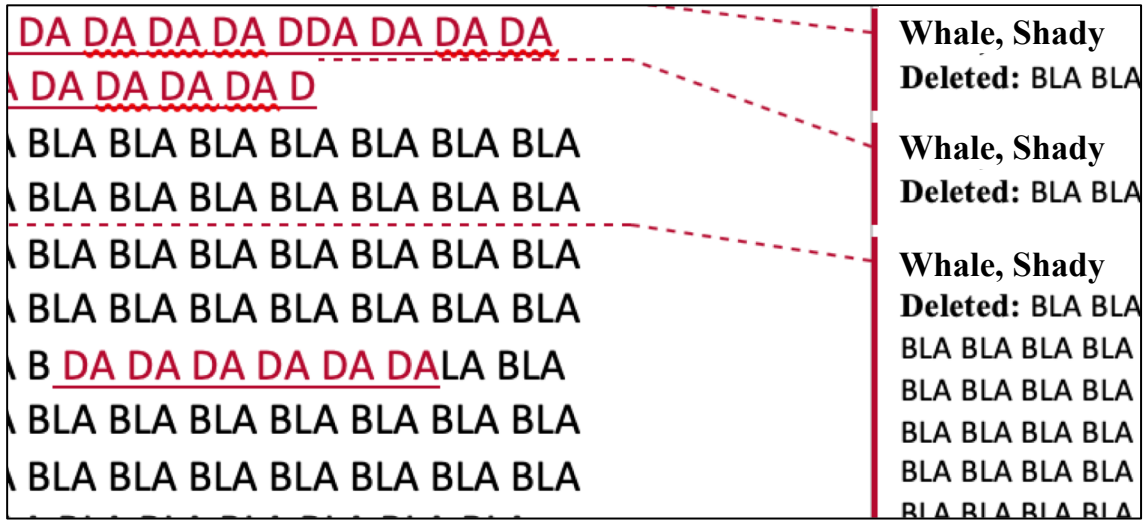
GitHub for sharing your code

Git Overview

What do we do for word documents?

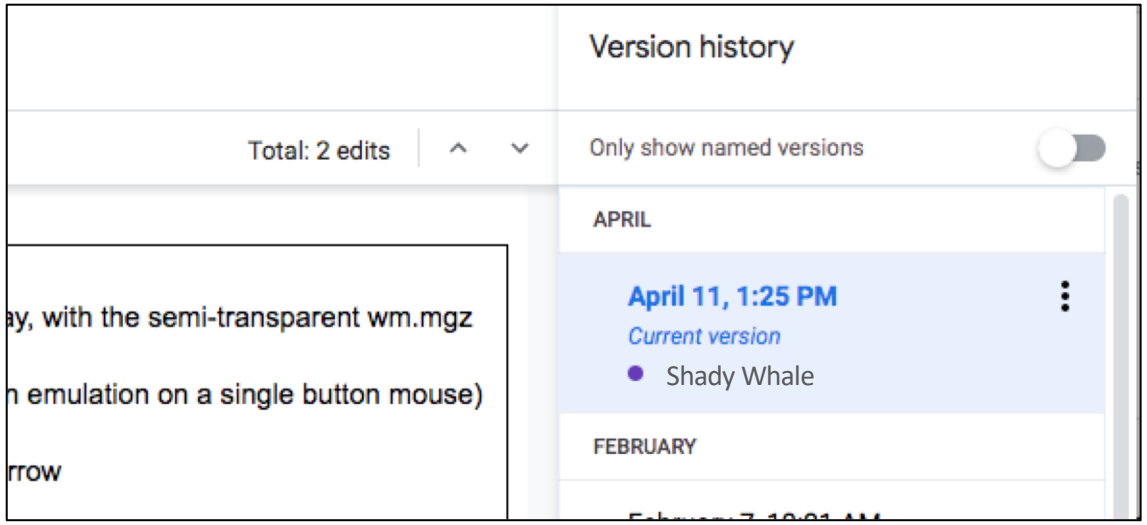
Track Changes (Microsoft Word)

- Highlighted changes



Version History (Google-doc)

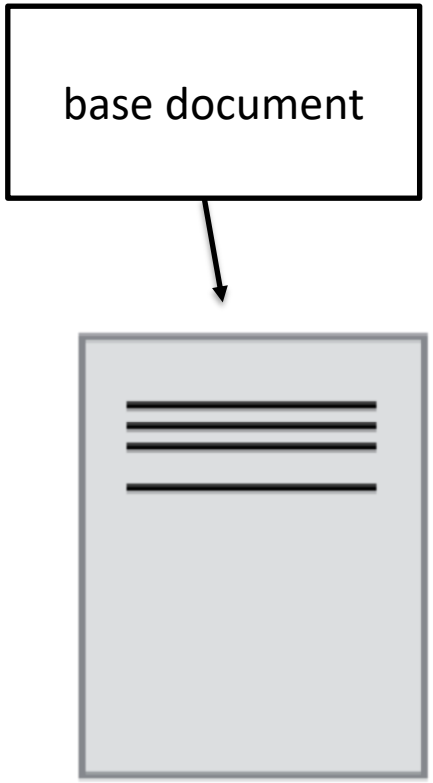
- Snapshots



Git **track changes** & gives you a **version history**!

Git Overview
What is being tracked?

Don’t think of it as multiple documents, but **base document** + **sets of changes**



Git Overview

Separate changes can be made, and recorded

Hands on portion of the workshop

Dark bubble with text are terminal commands (bash)

```
> echo hello
```

- Type into a terminal window or Git-Bash

OS icon will be next to the commands specific to an OS


Windows


Mac


Linux

Everything in this presentation is on a bookdown page:
https://gitbookdown.site

1. Setting up git

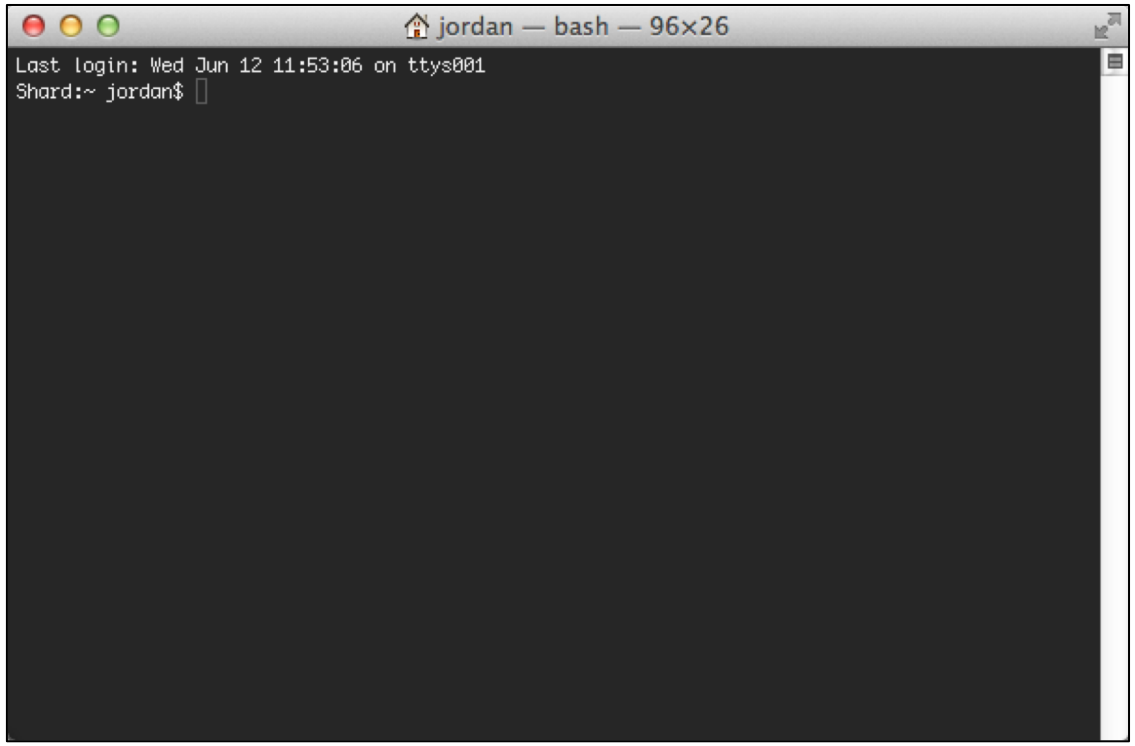
1.1 Download git

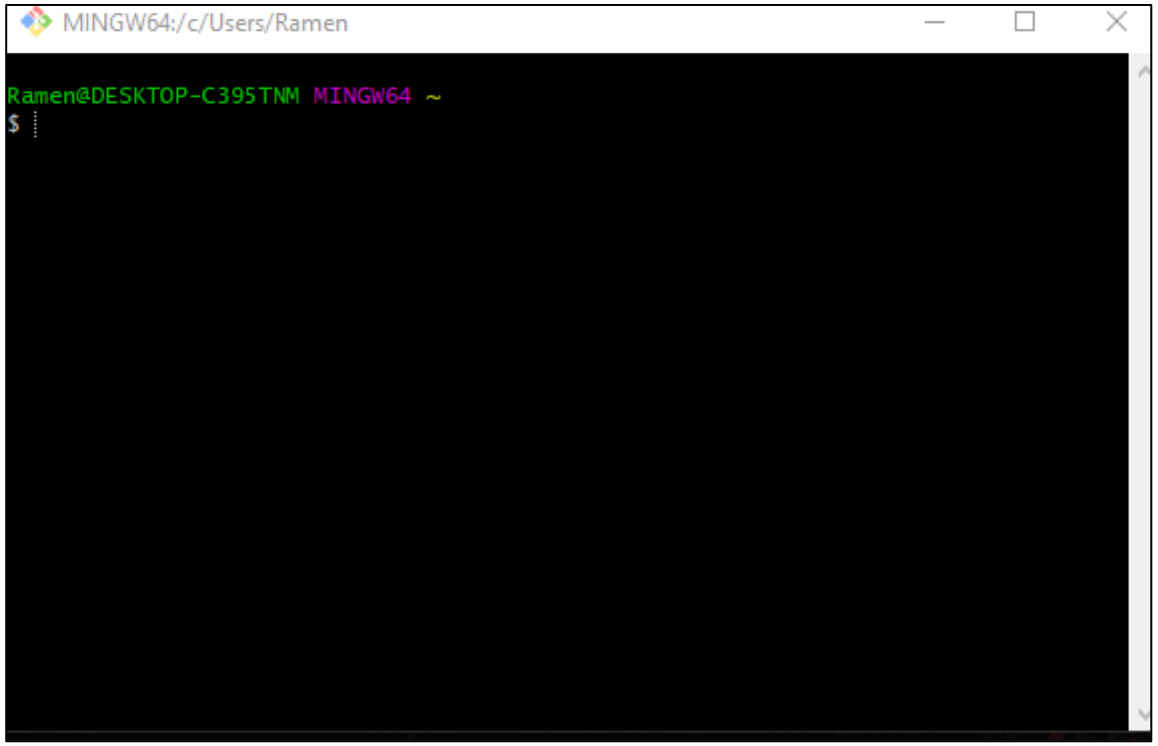
- Windows users: https://gitforwindows.org/ 
  - **Git-Bash** is a bash terminal that comes with Git for Windows.
  - In this workshop, we assume Windows User have Git-Bash, so you can use functions such as “echo” or “touch”

- Linux/Mac Users: https://git-scm.com  

1. Setting up git

Open a terminal / Git-Bash





1. Setting up git
1.2 Configure git

- Username & email

```
> git config --global user.name "Shady Whale"
> git config --global user.email "shadywhale@allthewhales.com"
```

- Setup default text editor (‘nano’ is used in this tutorial)

```
> git config --global core.editor "nano -w"
```

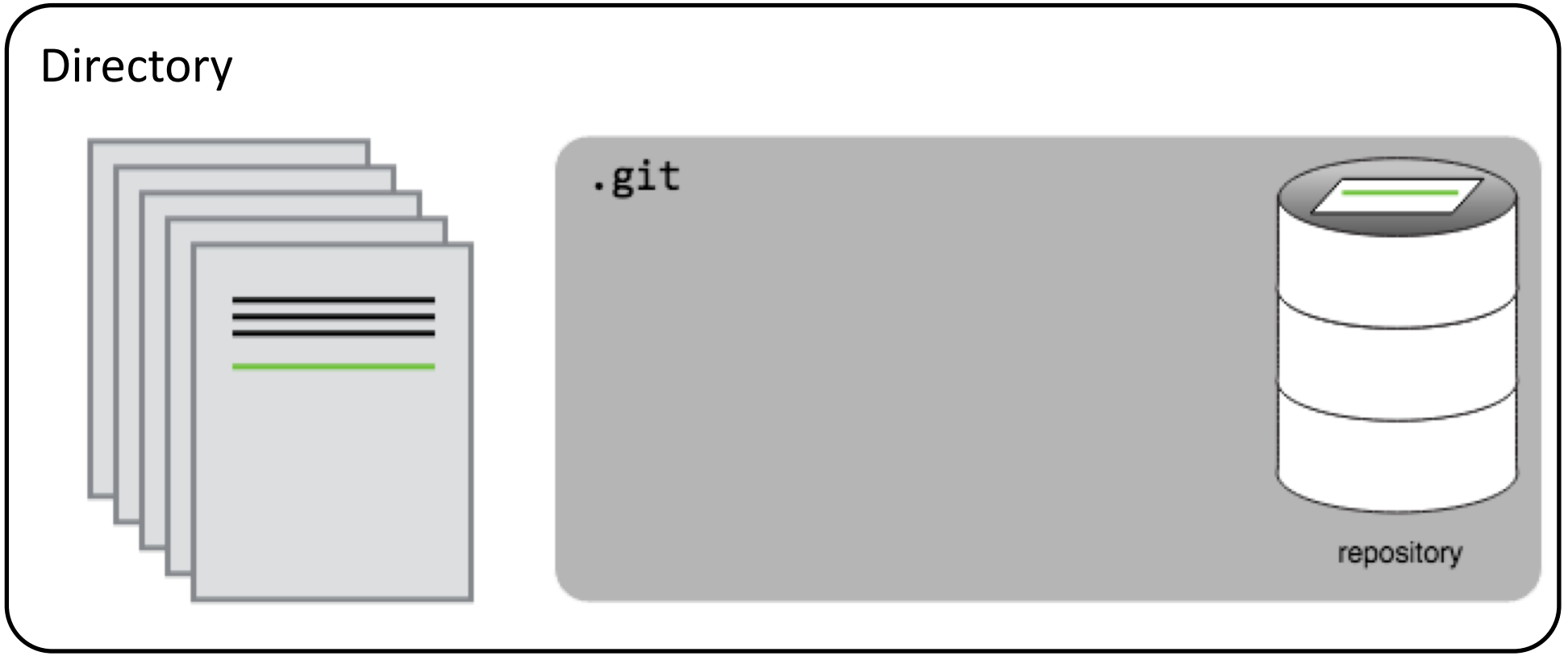
Now you should have Git installed on your personal computer (or server), with basic configurations all set and ready to go!

2. Creating a repository

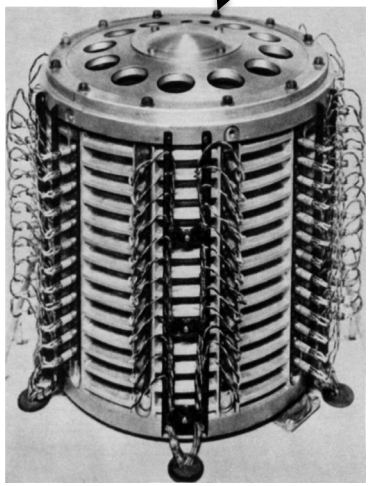
What is a repository?

Repository (repo):

- An index that tracks changes of a directory
- a `.git/` subdirectory



Old school drum memory



2. Creating a repository
`> git init`

2.1 – Create a directory in your home directory

```
> mkdir ~/workdir
> cd ~/workdir
```

2.2 – Create a **repository**

```
> git init
```

Check the status of the git repository

```
> ls -a
> git status
```

You now have a repository (i.e., a .git/ folder) within a directory that you want to track!

3. Tracking Changes

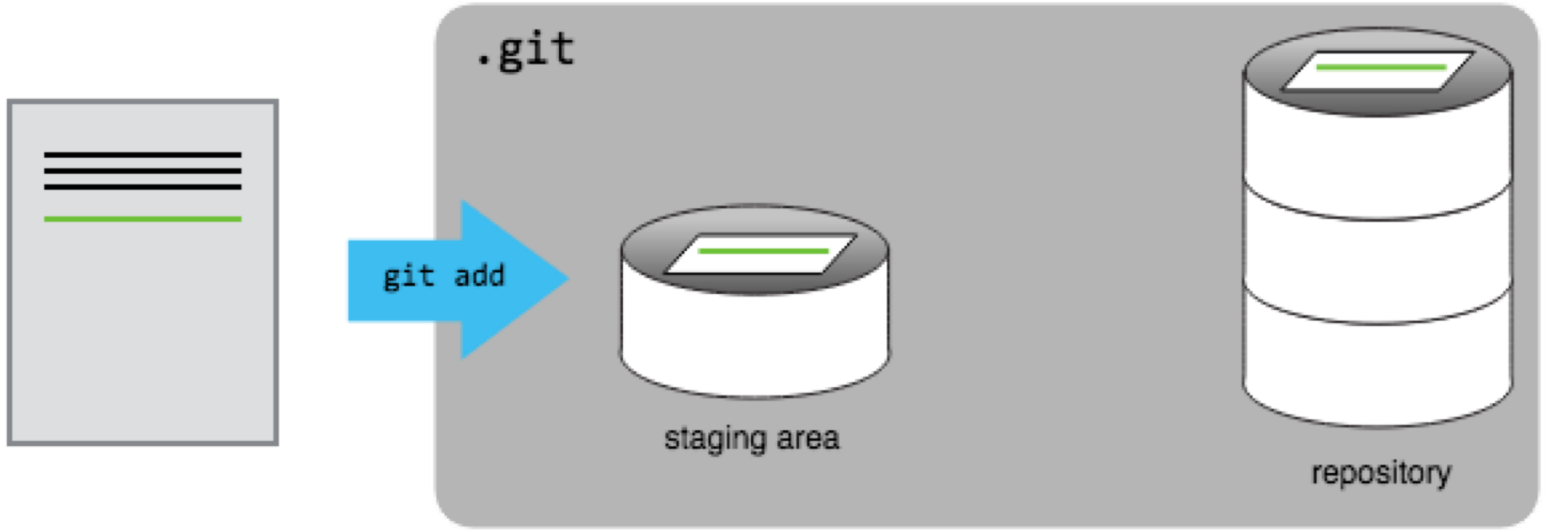
> git add, commit

git add

- Tells git which changes you want to track
- Formally, this is call adding a file to the "staging area"

git commit

- Tell git to document the changes you "added" (i.e., the files in the staging area)
- Mandatory **commit message** that you **must provide** to explain what changes were made



3. Tracking Changes
> git add, commit

3.1 – Adding new files/modification to **staging area**

- Make a new file and check status of git

```
> touch foo.txt
> git status
```

- Let's tell Git to track the file

```
> git add foo.txt
> git status
```

3. Tracking Changes
> git add, commit

3.2 – Saving these changes to the **repository**

- Commit the file with a commit message

```
> git commit –m “Add foo.txt to repo”
```

- Let’s check the log, which lists all the commits made so far

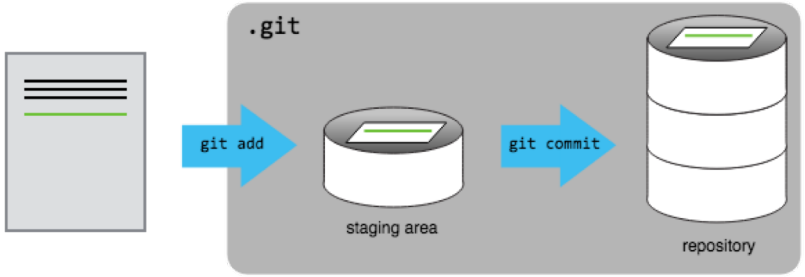
```
> git log
```

3. Tracking Changes
> git add, commit

PRACTICE:

- Let's add a line to the file and **git add & commit** it!

```
> echo hello >> foo.txt
> git add foo.txt
> git commit –m "Add hello in foo.txt"
> git log
```



You have now used `git add & commit` to track a file. Use `git log` to read what you have done so far in this repository!

4. Exploring History

`> git diff & show`

4.1 – Difference between current file and N commit ago

- Edit the file and check difference between your current foo.txt and the last commit

```
> echo world >> foo.txt
> git diff HEAD foo.txt
```

- What about compared to two commits ago?

```
> git diff HEAD~1 foo.txt
```

HEAD~1 can be replaced with the specific identifier of a commit. Use git log to check

4. Exploring History

`> git diff & show`

4.2 – What was done in ___ commit?

- Sometimes we want to check what was done during a certain commit

```
> git show HEAD~1 foo.txt
```

You have now compared different versions of a file using **`git diff`** and check changes during a specific commit using **`git show`**.

5. Telling git to ignore certain files
`.gitignore`

- Sometimes we don't want git to track a certain type of files
  - Temporary files/big data/OS files
- Adding a file to `.gitignore`

```
> touch ignore_this.txt
> echo ignore_this.txt >> .gitignore
```

- Try `git add` a file we added to .gitignore, it won't let us

```
> git add ignore_this.txt
```

There should now be a .gitignore file in your directory.

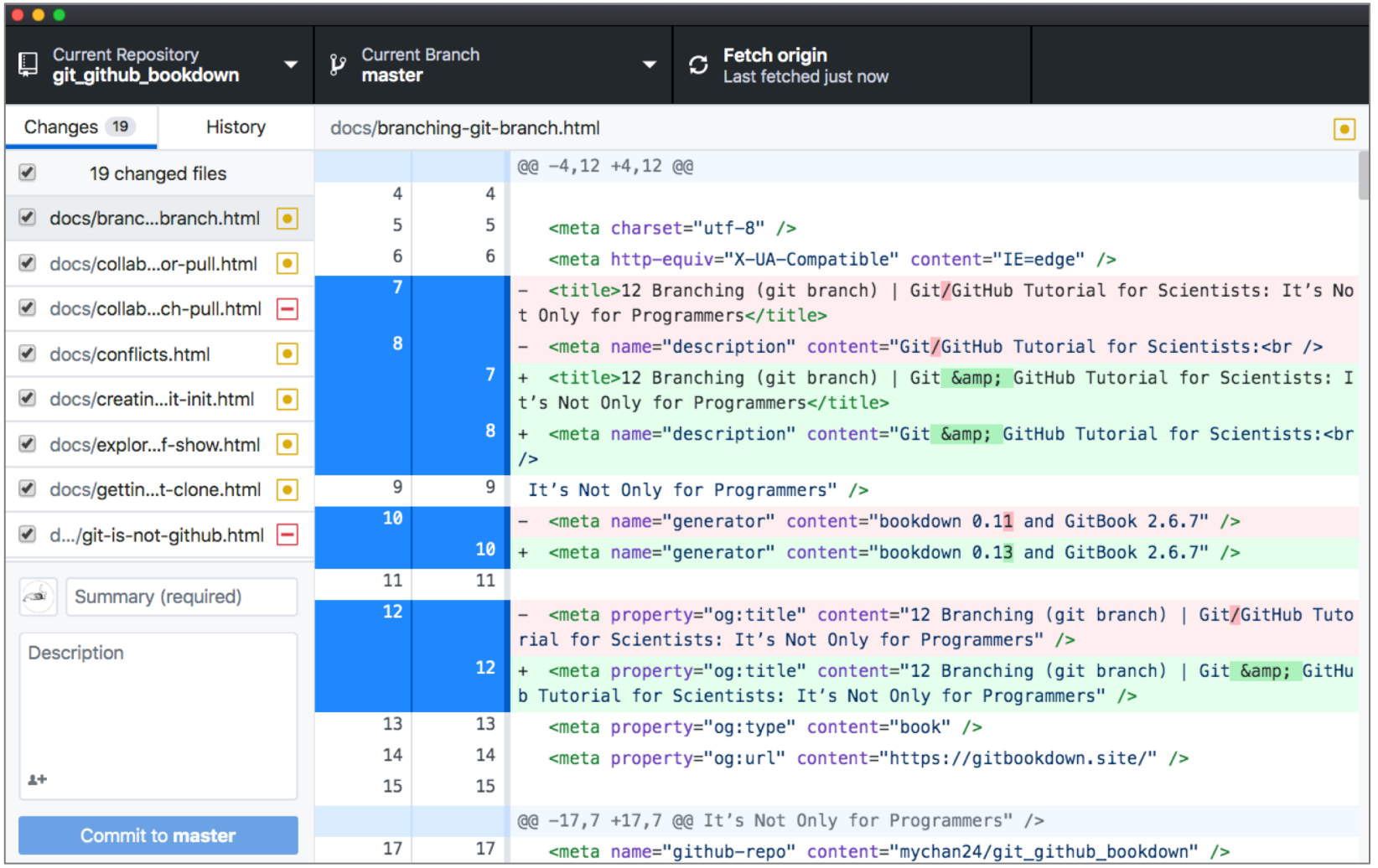
Desktop Application

GitHub Desktop FREE

Sourcetree FREE

GitKraken FREE $

Tower $



This ends the intro to the **version control** function of git.

Next up:

GitHub and its cousins to help your share your codes

BREAK

If you don’t have a GitHub account yet,

please sign-up for an account at https://github.com

GitHub

and other remote hosts

A remote server that hosts repositories for free

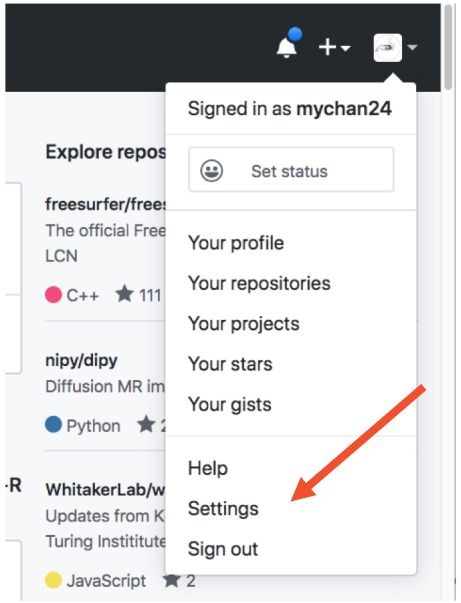
It also offers visualization of code changes and collaboration insights!

(lets take a look)

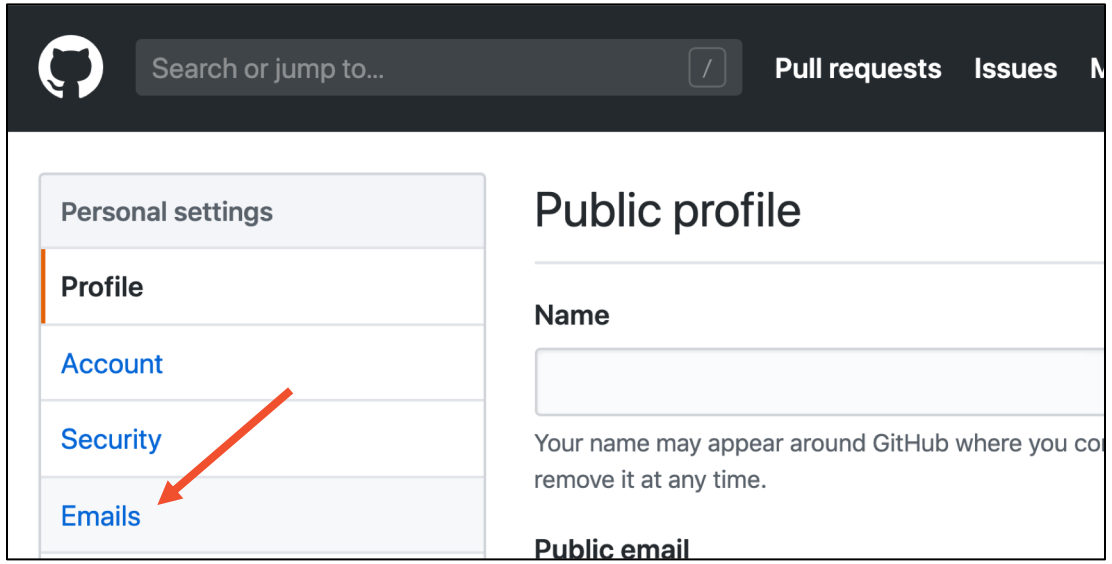
Some other choices:

Configure GitHub's Email Setting

1) Go to Settings

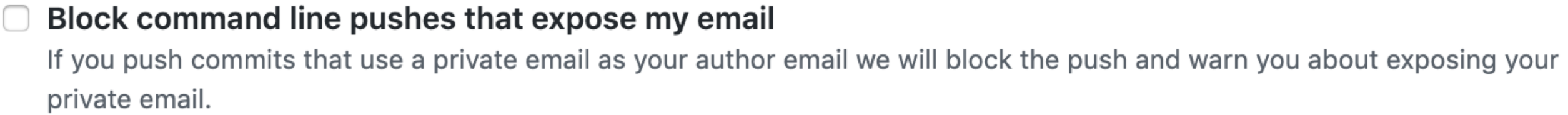


2) Click Emails



3) Enter email you used for workdir

4) For this tutorial, un-check 'Block command'



6. Create a repository on GitHub

make a repo

Create a remote repository

- Go to https://github.com/<your username>
- Click the **Repositories** tab
- Click 

- Enter **workdir** as the Repository name, click 

- **NOTE:** We will skip adding a README
- Click 

to copy the URL of this repository

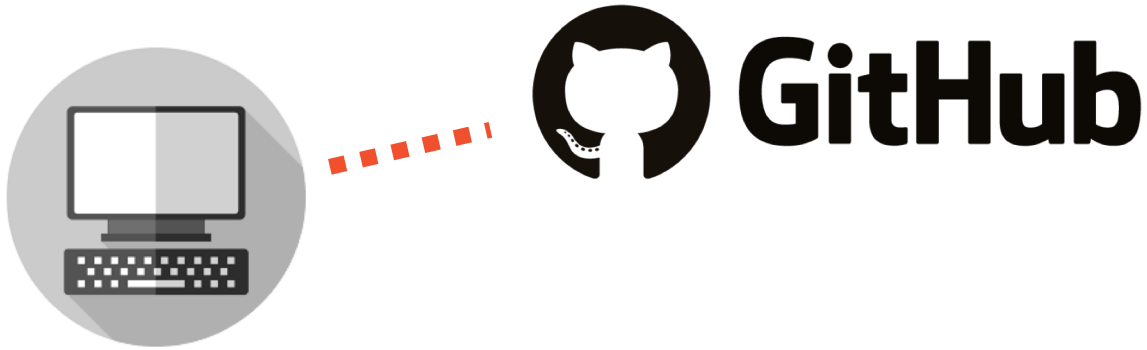
You now have an empty remote repository and some instructions to populate it!

7. Putting codes on GitHub

> git remote add & push

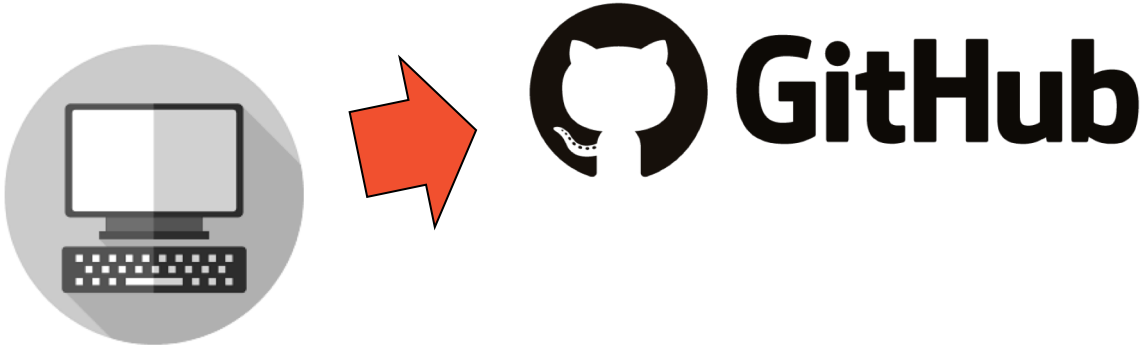
git remote add:

- Add a remote repository by its url so you can down/up-load from it.



git push:

- Upload a commit(s) and modifications to a remote repository.



7. Putting codes on GitHub

> git remote add & push

7.1 – Push an existing repository from your local computer

- Go to workdir & add a remote repository that you want to push your data to

```
> cd ~/workdir
> git remote add origin <copied URL>
```

https://github.com/<username>/workdir.git

- Push your local repository to your GitHub

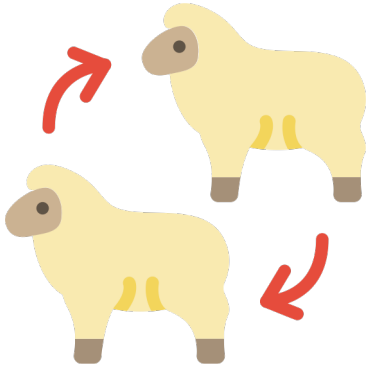
```
> git push –u origin master
```

Your code is now available on GitHub!

8. Getting a repository from GitHub

`git clone & fork`

Clone

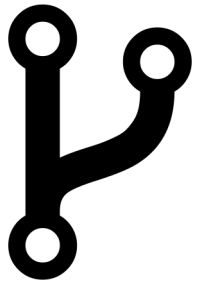


Copy a entire repository

Usually we clone from **GitHub** to our **computer**

GitHub → 

Fork



Copy a GitHub repository into your GitHub

Its essentially cloning from **GitHub** to **GitHub**

GitHub → GitHub

8. Getting a repository from GitHub

`Practice fork & clone`

8.1 – Lets Fork the repository with this presentation

- Go to https://github.com/mychan24/git_github_workshop
- Fork it by clicking 

- You'll be redirected to a copy of the same repository in **YOUR** GitHub account

8. Getting a repository from GitHub

Practice fork & clone

8.2 – Now lets clone that to your computer

- Go to https://github.com/**<username>**/git_github_workshop

- Click 

to copy the URL for cloning

- Clone it using the following command line

```
> cd ~
> git clone <copied URL>
```

You have forked & cloned someone else's repository! This is often how collaboration starts through GitHub.

9. Collaborating

9.1 – Setup a fake collaborator (i.e., yourself in another directory)

- Go to your GitHub account's workdir repository:
- Copy the URL 

- Clone it to a new directory call at "~/workdir_fake_collab"

```
> git clone <copied URL> ~/workdir_fake_collab
```

https://github.com/<username>/workdir.git

9. Collaborating

- **Make changes as fake collaborator and push to GitHub**
  - Go into the collab's directory, make a new file foofoo.txt

```
> cd ~/workdir_fake_collab
> touch foofoo.txt
```

  - Git add & commit foofoo.txt

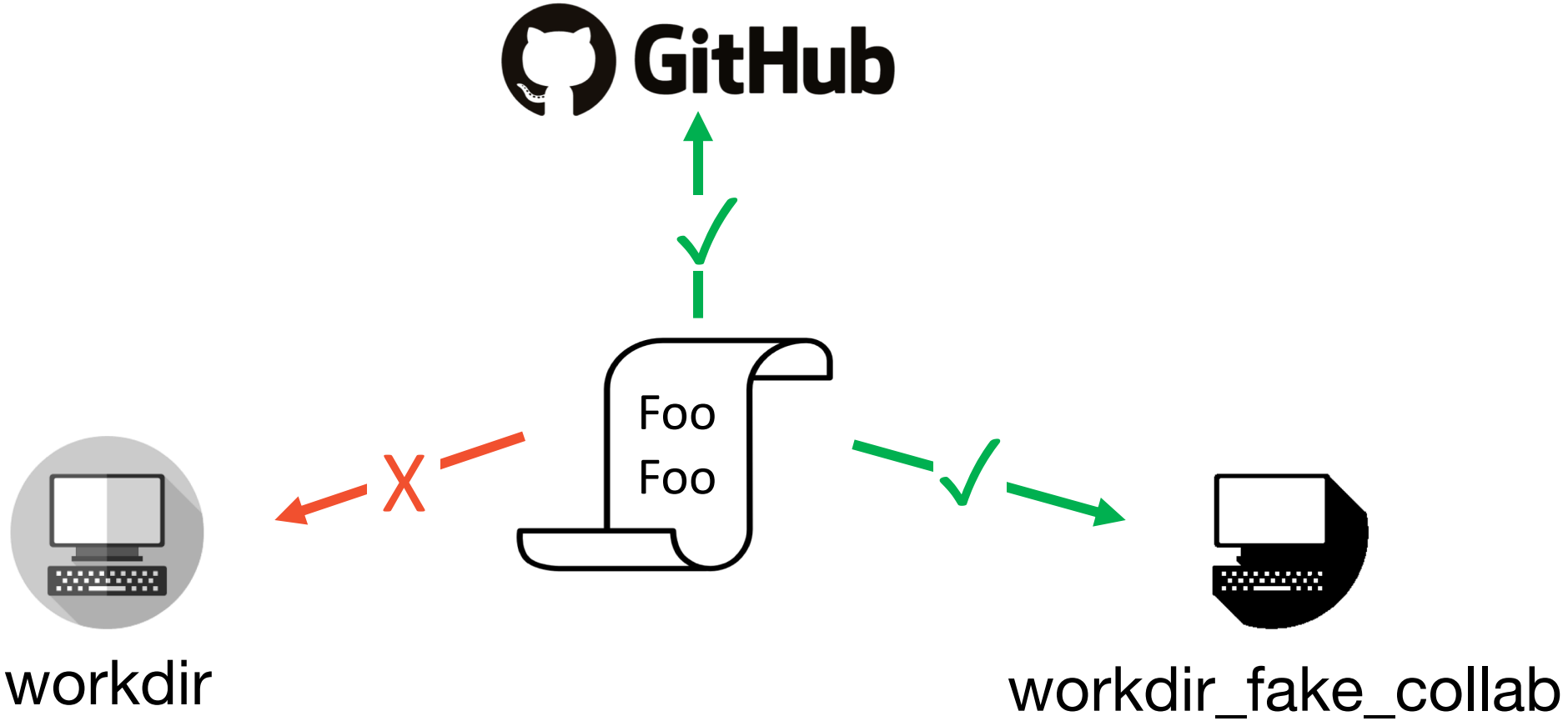
```
> git add foofoo.txt
> git commit -m "Add foofoo.txt"
```

- **Push this change to GitHub**

```
> git push origin master
```

9. Collaborating

Where is foofoo.txt?



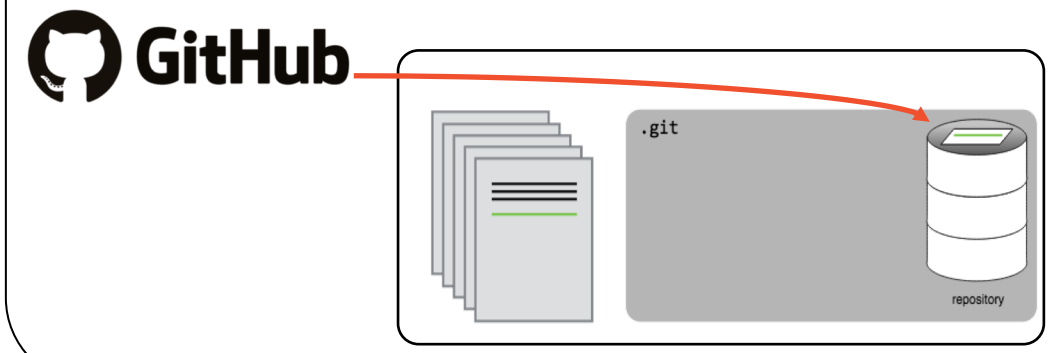
9. Collaborating

> git fetch & git merge

Fetch

Download snapshot of remote

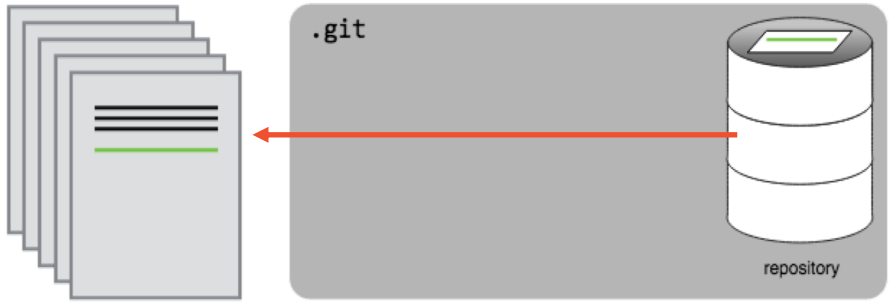
Downloading the changes into .git



Merge

Integrate fetched changes

Incorporate changes to local files



9. Collaborating

`> git fetch & git merge`

9.2 – Implement the changes in original workdir

- **Fetch** the changes your fake collaborator pushed to GitHub

```
> cd ~/workdir
> git fetch
```

- Check what your collaborator has changed compared to your local version

```
> git diff master origin/master
```

Your local master branch

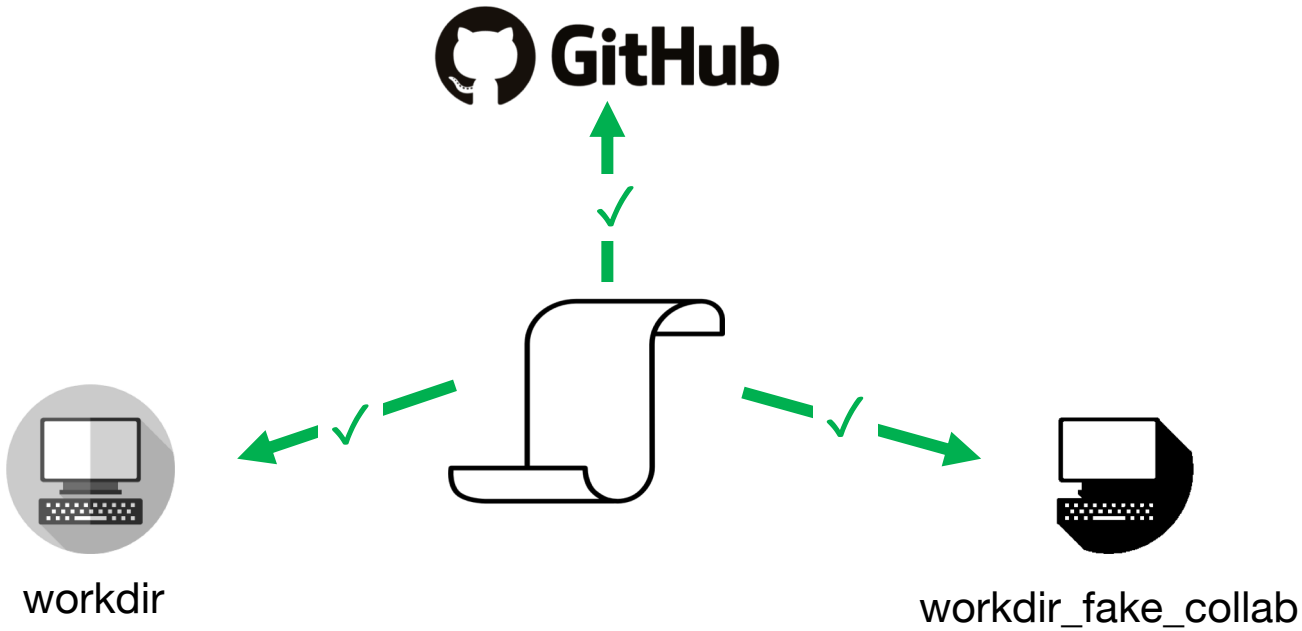
Master branch on GitHub (origin is what we nicknamed our remote repo on GitHub)

9. Collaborating

> git fetch & git merge

- If these are changes you want in your workdir, merge the changes

```
> git merge
```

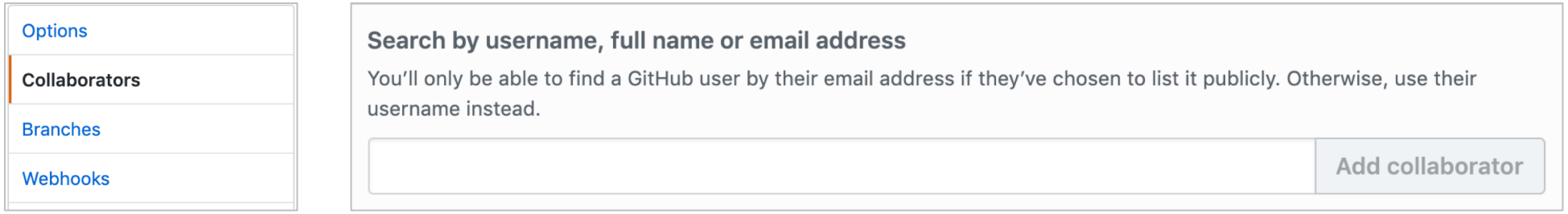


You have now fetched and merged changes from different people (or yourself on a different computer/folder)!

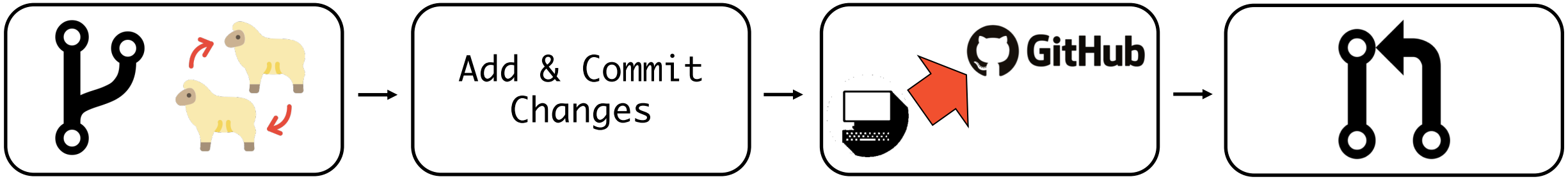
9. Collaborating

BONUS: Truly Collaborating on GitHub

1) Add collaborators in your GitHub repository



2) Using pull request (advance)



Fork & clone a repo on GitHub

Track your changes using Git

Push changes to GitHub

Make a **pull request**

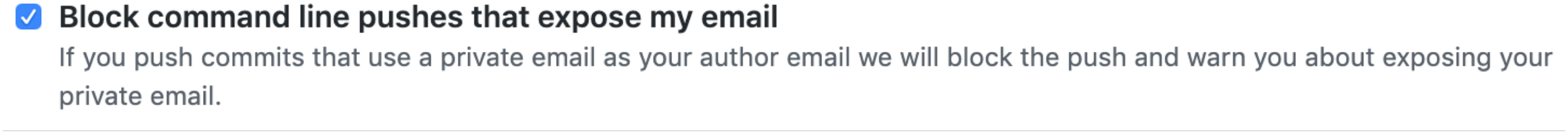
The original owner of the repository can review your changes before incorporating it

Email settings in real life

Setup the no-reply email provided by GitHub (in email setting)

```
> git config --global user.email ”shadywhale@users.noreply.github.com"
```

Check the following Setting on GitHub



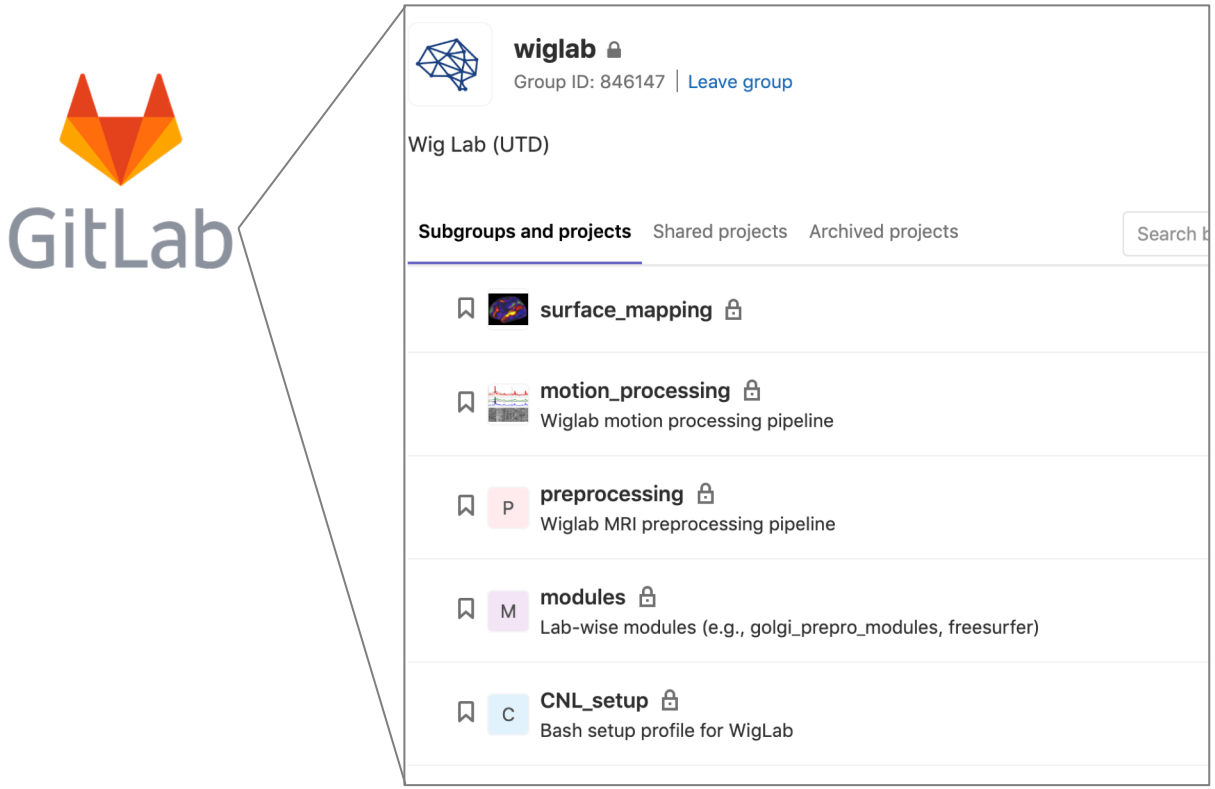
Advanced Git Topics

The following topics are covered using the bookdown https://gitbookdown.site

- Resolving conflicts
- Reverting to a previous commit
- Branching
- Stashing

Other resources

- GitHub Education Pack
  - https://education.github.com/
  - Pro account for students, academic researchers (i.e., postdoc) & faculties
- GitLab
  - https://gitlab.com



Acknowledgement