

GitHub

Version Control and Collaboration
with
Jojo Karlin

ITP Workshop
The Graduate Center
@jojokarlin jkarlin@gradcenter.cuny.edu



Workshop Objectives

What you will learn

1. Get a sense of the layout of GitHub
2. Get the basic purpose of git
3. Start your own repository
4. Push changes from your local repo
5. Follow a repository (and see where it goes!)

THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



What is git

?

Git is the open source distributed version control system that facilitates GitHub activities on your laptop.

In basic terms, git follows and records every change you make on your computer. It is doing the work behind GitHub.

Linux creator Linus Torvald released git in 2005 for Linux kernel development (which has thousands of collaborators). He named it for himself (the git!).

What is GitHub?

GitHub is the online face of Git.

GitHub is a way to access a vast scope of projects.

GitHub helps you collaborate with line by line documentation of revisions.

GitHub is a place to store your projects, explore new ones, and collaborate.



GitHub and you

git is a very powerful software, but one of its best assets is that it works for EVERYONE from the click-through beginner to the expert who dreams in code.

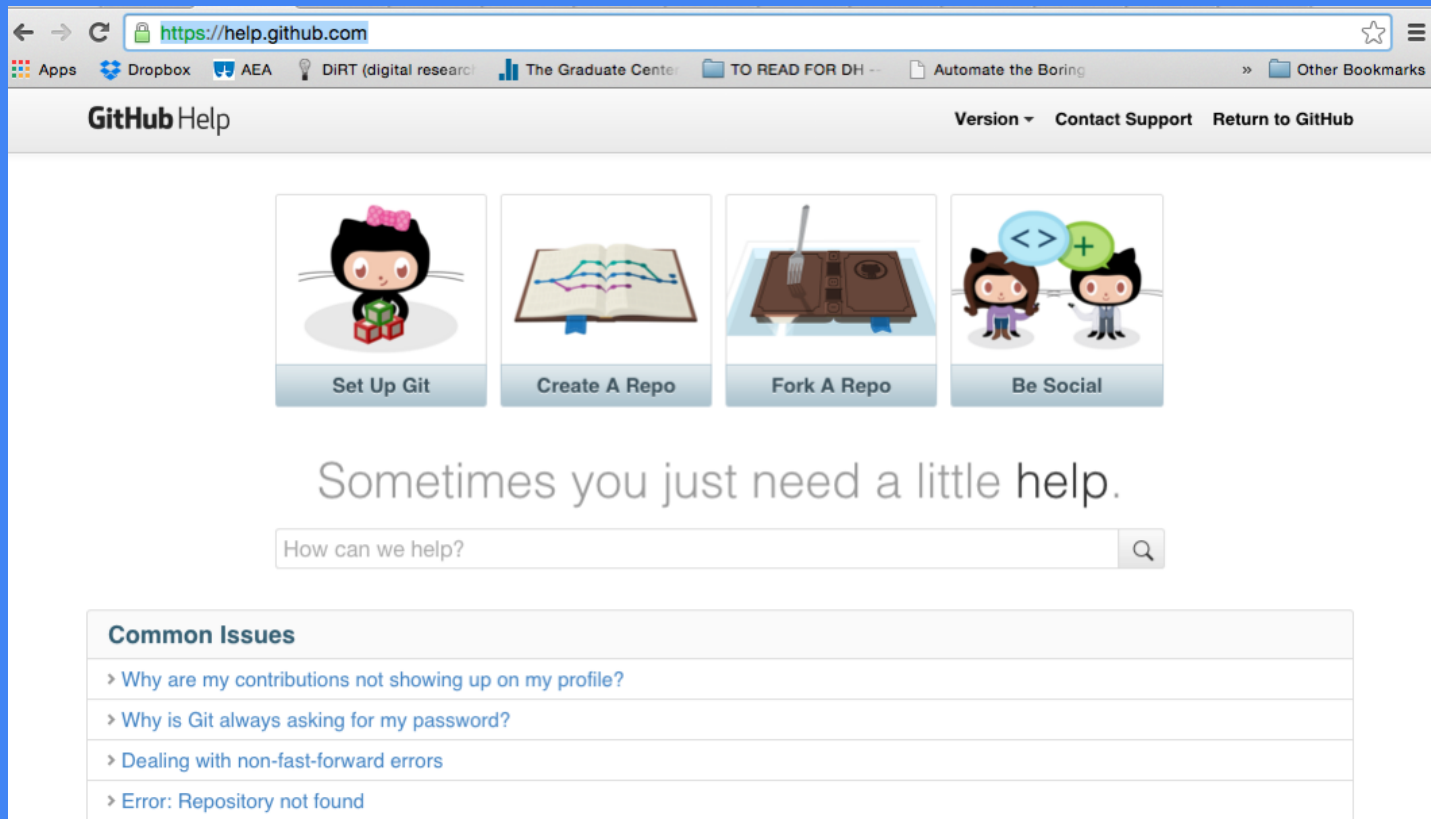
That said, there are many features on GitHub that you can leave alone until you are ready to participate in more advanced projects.

Don't worry! GitHub can work for you, too.

Let's begin.

Open your GitHub account at
github.com.

There is so much help available it can be overwhelming.
Remember, it's fine to stick to the simplest parts.



The screenshot shows the GitHub Help website in a web browser. The address bar displays `https://help.github.com`. The browser's bookmark bar includes links to Apps, Dropbox, AEA, DIRT (digital research), The Graduate Center, TO READ FOR DH, Automate the Boring, and Other Bookmarks. The GitHub Help header features the site name, a 'Version' dropdown, and links for 'Contact Support' and 'Return to GitHub'. The main content area presents four large, rounded rectangular buttons with illustrations and labels: 'Set Up Git' (with a cat character holding boxes), 'Create A Repo' (with an open book), 'Fork A Repo' (with a book being cut), and 'Be Social' (with two characters and code symbols). Below these buttons is the text 'Sometimes you just need a little help.' followed by a search bar containing the text 'How can we help?'. At the bottom, a 'Common Issues' section lists several topics with expandable arrows.

GitHub Help

Version ▾ Contact Support Return to GitHub

Set Up Git

Create A Repo

Fork A Repo

Be Social

Sometimes you just need a little help.

How can we help?

Common Issues

- › Why are my contributions not showing up on my profile?
- › Why is Git always asking for my password?
- › Dealing with non-fast-forward errors
- › Error: Repository not found

A repository is a project.

Create a repository!

To create a new repository on GitHub:

1. Click the icon next to your username, top-right.
2. Name your repository [descriptive title].
3. Write a short description.
4. Select **Initialize this repository with a README.**

In just a moment, we will create a repository using git on the Command Line!

README.

This file should serve as a title page and abstract for your project.

Your repository or project may wind up consisting of a number of files of different types and sizes. The README. should explain what everything is meant to do-- a mission statement and index all in one.

Good practice

Good project

initiate a repository with a README.

create a requirements file outlining
exactly which version of which
software you are using.

BE SUPER CLEAR ABOUT YOUR
ENVIRONMENT.

Using GitHub will help your digital projects.

DOCUMENTATION!

Even when you're not sure what you are documenting,
the exercise of articulating your edits will help you
proceed.

GitHub KEY TERMS

GitHub can be confusing because there are a lot of metaphors happening at once.

Family Tree type metaphors:

BRANCH
ROOT
ORIGIN
MASTER

Dinner table?

FORK

Relationships
language:

PUSH
PULL
COMMIT

Octocat?



Archive words:

REPOSITORY

Git-ionary

CLONE is your version of an existing project copied to your own computer.

FORK is a **CLONE** on the GitHub server side.

BRANCHes are for separate modules of BIG projects. (Don't worry about this)

AVOID CONFLICT! Don't merge multiple branches simultaneously.

*Don't worry about **PULL REQUESTS** on small projects.*

Using git at the Command Line
open Terminal

local git commands
(at your computer at home)

```
$ git init [myproject]
```

here's where you name your project
[you don't need the brackets]

```
$ cd myproject
```

this creates the directory with git already enabled, which starts the project, but also creates the control file which will store the information related to when you make any changes on the document.

```
$ git add .
```

This command connects with the software and puts your file in a holding zone readying it to be permanently committed to the project (the '.' means all, you can also choose to add one file you select)

```
$ git commit -m “[descriptive  
message]”
```

This command commits your changes to the official project. Be clear about what you've done. This is the “record this snapshot” command.

```
$ git clone [github url]
```

download an existing project (with all its controls ready to go!)

commit

some changes



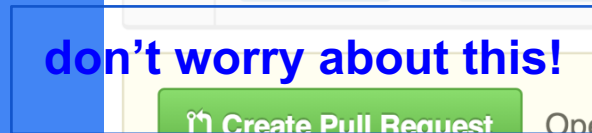


Here's where later on you might create or **fork** a **branch** to work on a portion of the project...

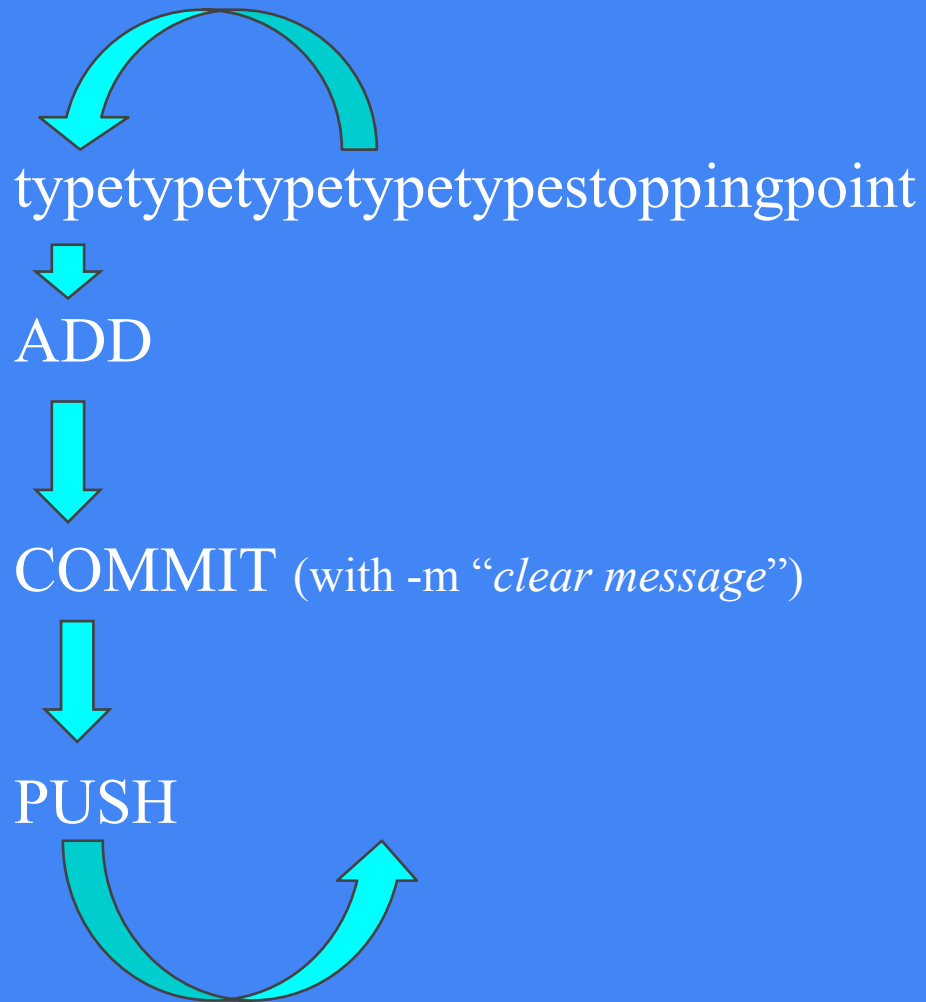


A **pull** request would then ask your master branch to check your changes

receive feedback
(git requires the administrator of the master branch to **review**)



...but for now, stick to using GitHub to compile the history for your own personal project. You can think of it as a nifty, nit-picky backlog of all the changes you make. It's a super powerful undo button.



make sure you committed
your changes locally first!

your master branch gives
the go and **commits** your
changes to the master



```
$ git add [file]
```

```
$ git commit -m “[your message!]”
```

```
$ git push [your local alias] [master]
```

\$ git pull

Makes sure you're up to date when you sit down at your desk

\$ git status

Shows you all the files you've changed that need to be committed to the master branch

\$ git diff

Shows file differences that haven't yet been staged (added)

Read more about GitHub and how much it can do for you

Librarian Git Arsenal: <http://bit.ly/2j0N4rS>

Github for beginners: <http://bit.ly/19vDdUS>

<https://github.com/learn-co-students/git-github-and-learn-oc-000>

<https://github.com/learn-co-students/git-version-control-101-v-000>

Got git?

There are lots of online tutorials with more information about git and GitHub. Learning git can be daunting and may take several attempts, but in the long run, it protects your data and improves your workflow.

<http://git-scm.com/video/what-is-git>

<https://vimeo.com/github>

PICK a project and see where it goes!

Social Paper

<https://github.com/cuny-academic-commons/social-paper>

Git-Lit

<https://github.com/Git-Lit>

Open Lab

<https://github.com/livinglab/openlab/>

Did we succeed?

How well did you get git?

1. What's your sense of the layout of GitHub?
2. Do you get how git and GitHub differ?
*hint: Git**Hub** is an online hub for git
3. Did you get a GitHub account?
4. Did you start your own repository?
5. Follow a repository?

Thank you for adding your
commitment, push and pull to master
version control.

Your projects will thank you.

Please contact me if you have any questions!

jkarlin@gradcenter.cuny.edu
[@jojokarlin](#)

The Graduate Center
ITP Core 1