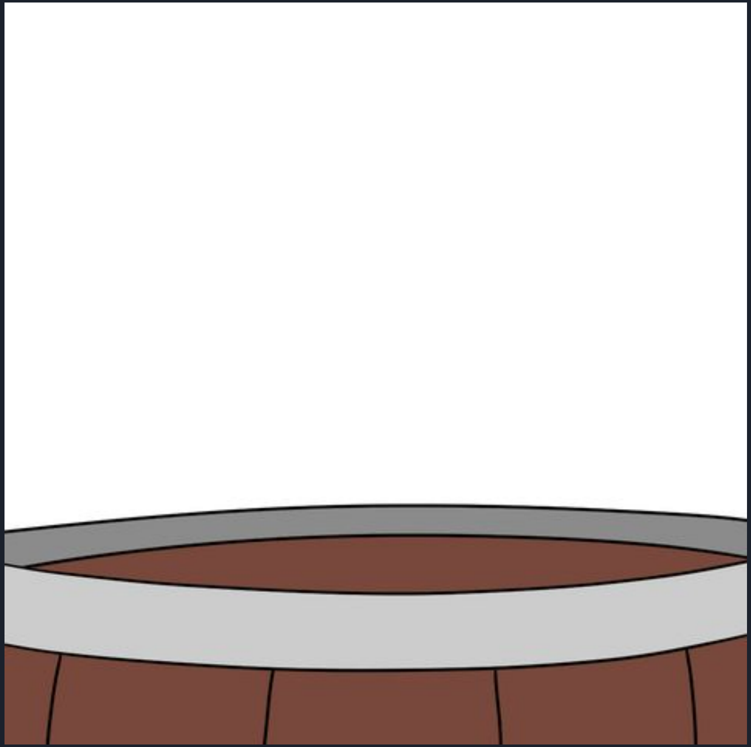
# Git/GitHub and CDNs

## Submit Homework

Make sure you submitted your Invision URL on Canvas



## This week overview

- CDNS
  - What are they
  - How to use them
- Git and GitHub
  - Create New Final Directory
  - Use GitHub Desktop
  - Start Using GitHub

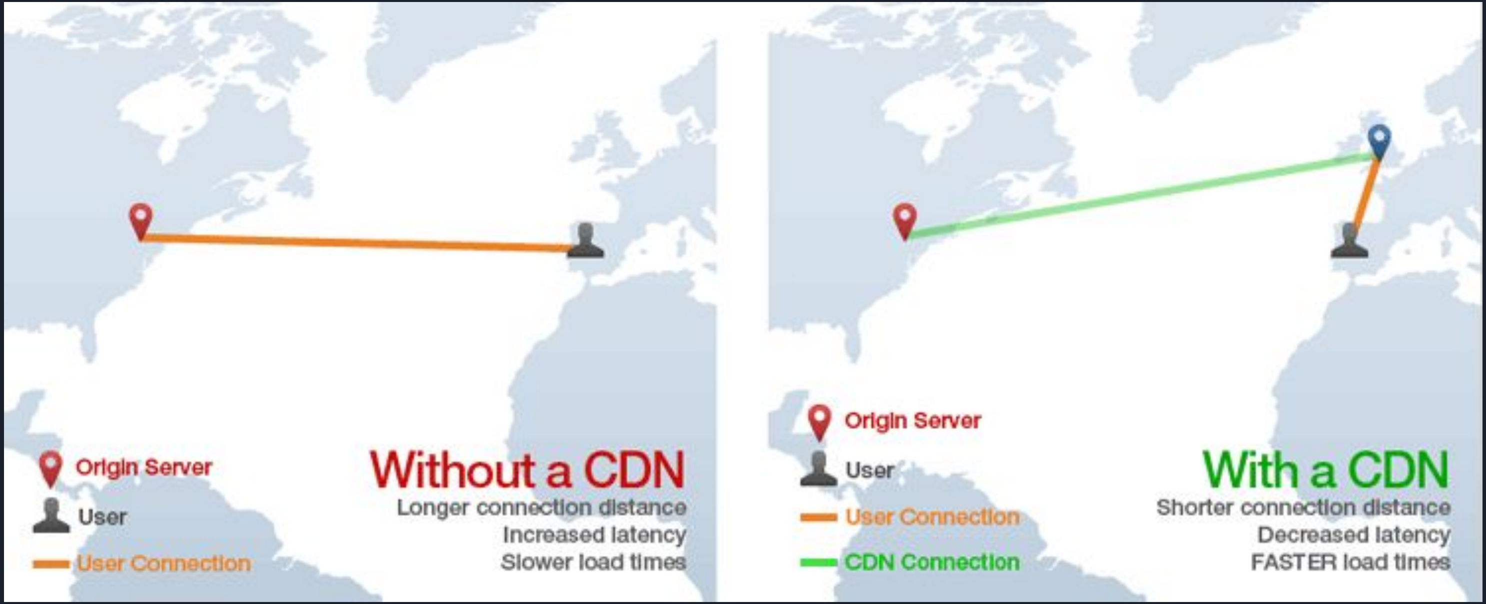
## CDNs

## CDN: What & Why

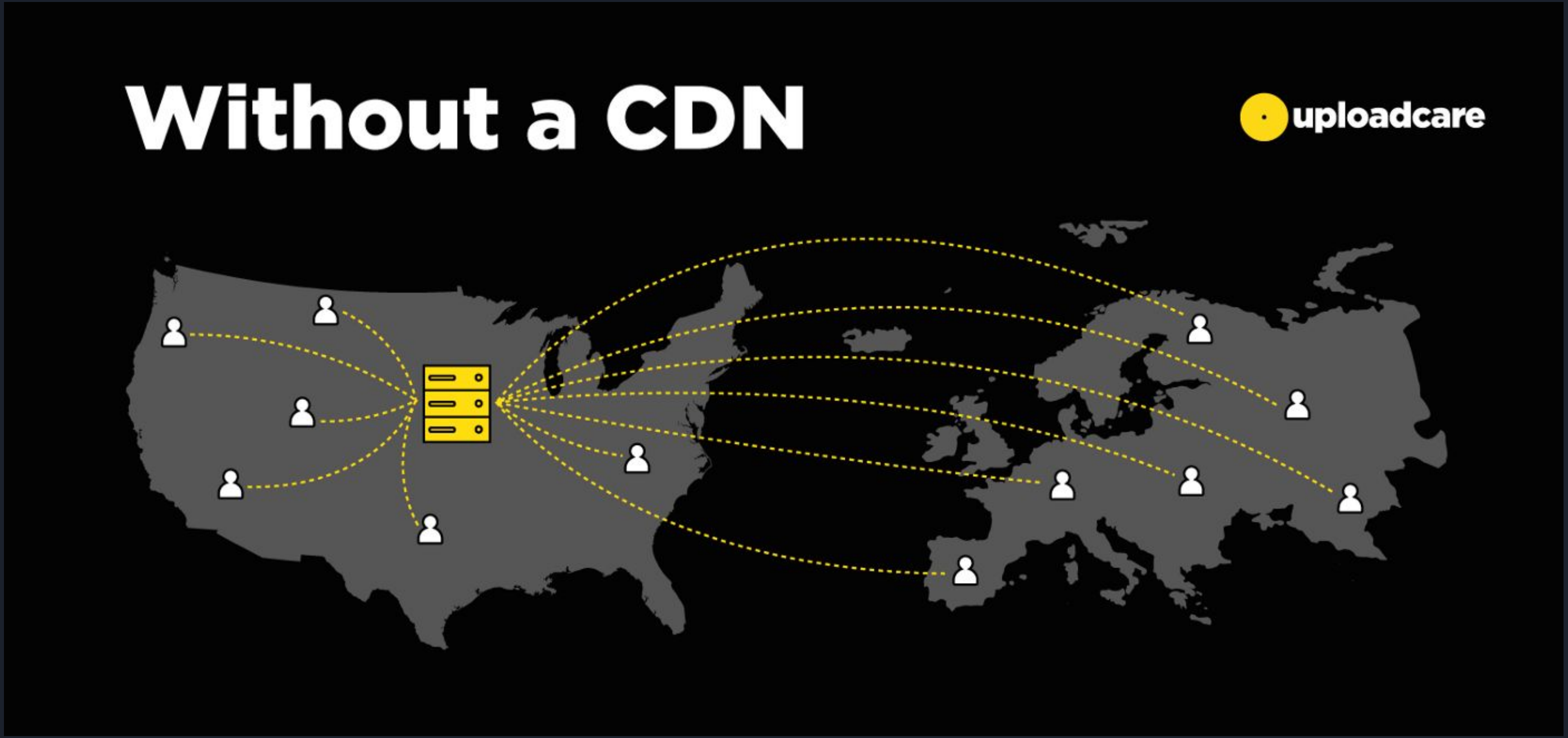
CDN stands for “content delivery network” or “content distribution network”

- A CDN allows for the quick transfer of assets needed for loading Internet content including HTML pages, JavaScript Files, Stylesheets, images, and videos
- “CDN nodes are usually deployed in multiple locations, often over multiple backbones, Benefits include reducing bandwidth costs, improving page load times, or increasing global availability of content”.

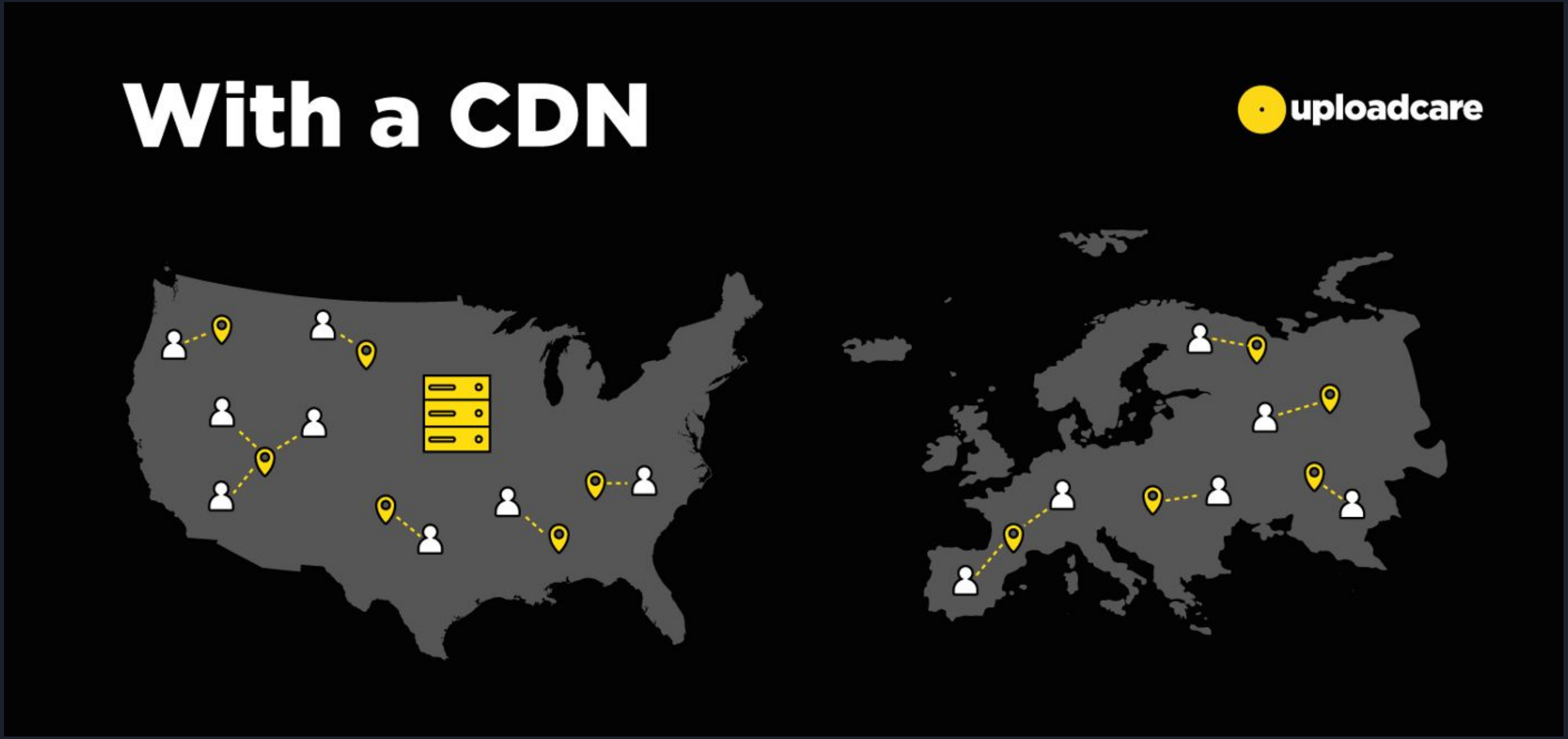
## Illustration of a CDN



## A Better illustration of a CDN (without)



## A Better illustration of a CDN (with)



## CDN: When will we use them?

CDN are vastly important and one way you can implement them is within your own projects.

For example let’s take a look at Twitter Bootstrap

Twitter Bootstrap is/was an extremely popular framework for building Responsive Frontend websites and application.

Let’s build a page that brings in Twitter Bootstrap to demonstrate how you might do this without the assistance of a CDN

## CDN: Demonstrate Twitter Bootstrap (TWBS)

1. Open class-directory
2. Create bootstrap.html file
3. Download [Twitter Bootstrap]
4. Unzip bootstrap-3.3.7-dist
    a. The reason we aren't using Bootstrap 4 for this is because it doesn't come bundled with fonts by default
5. Place the assets
    a. bootstrap-3.3.7-dist/css/bootstrap.min.css >>> class-directory/assets/css
    b. Don't worry about JavaScript files
    c. bootstrap-3.3.7-dist/fonts/* >>> class-directory/assets/fonts
        i. The * means all the files
6. On Twitter Bootstrap site visit Components and paste in some examples

## CDN: Manual file placement

With that example we had to manually go out to

1. Go to the website
2. Download the files
3. Unpack the files
4. Place the asset files
5. Code

That isn’t a ton of work in this example but what if

- You had to do this for 30 different tools, plugins, etc.?
- You are serving files from your server say in the US. What if your user is in China?

Coca-Cola
4GIFs.com

## CDN: How do we implement?

1. Staying in class-directory
2. Create a new file bootstrap-cdn.html
3. Navigate to Twitter Boostrap “Getting Started”
4. Copy the first CDN link for CSS
5. Paste that link inside your HTML file in the <head>
6. Now in the <body> copy all the elements you had from bootstrap.html

# CDN: FTW

As you can see with a single line of html and the power of CDN you

1. ~~Go to the website~~
2. ~~Download the files~~
3. ~~Unpack the files~~
4. ~~Place the asset files~~
5. Code

## CDN: Loading Fonts

Custom fonts are a popular and fun way to make your website unique and give your site a bunch of personality

Many popular sites like

- DaFont
- FontSpace

Are great places to download and use custom fonts

For this demonstration we are going to load in a custom Google Fonts

Google Fonts is a great site to get amazing professional looking fonts simply and easily for any project

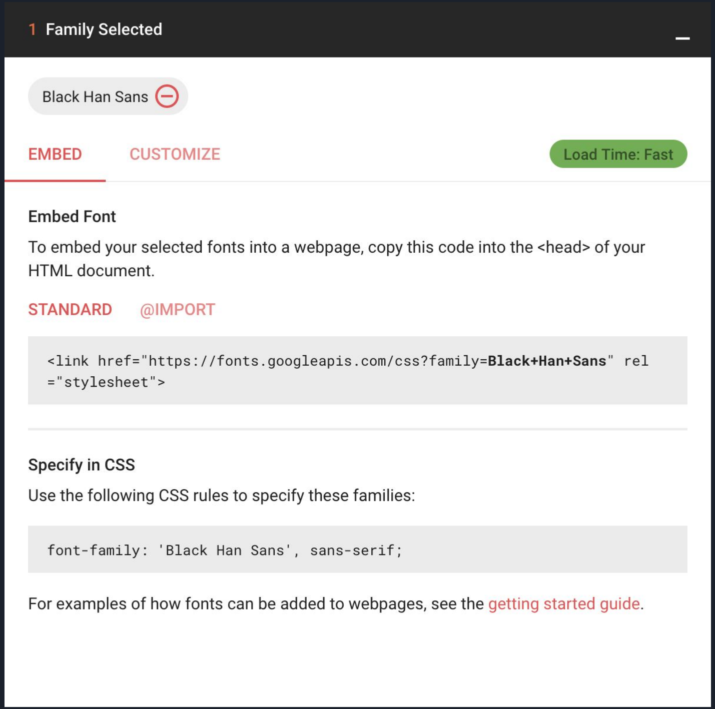
## CDN: Manual loading of fonts

1. Open class-directory
2. Create a new html file called fonts.html
3. Go to Google Fonts and download a font
4. Unzip the file of the font you selected
5. Take that font file and place inside of class-directory/assets/fonts
6. Create a fonts.css file inside of class-directory/assets/stylesheets
7. Inside the CSS file link up @font-face rule for your downloaded font
8. Back to fonts.html link up fonts.css
9. Test

y tho

# CDN: Load fonts by CDN

1. Create a file called fonts-cdn.html
2. Go back go Google Fonts
3. Select the font you had prior
4. Highlight the <link .../>
5. Paste link in <head> of your HTML file
6. Create a new CSS file
    a. assets/stylesheets/fonts-cdn.css
    b. Create a CSS rule and paste in the font family
7. View your code in the browser



Embed Font

To embed your selected fonts into a webpage, copy this code into the <head> of your HTML document.

```
<link href="https://fonts.googleapis.com/css?family=Black+Han+Sans" rel
="stylesheet">
```

Specify in CSS

Use the following CSS rules to specify these families:

```
font-family: 'Black Han Sans', sans-serif;
```

For examples of how fonts can be added to webpages, see the getting started guide.

## CDN: Recap

- Content delivery networks are the transparent backbone of the Internet in charge of content delivery
- No matter what you do or what type of content you consume chances are it is being powered by some type of CDN
- CDNs minimize the distance between users locations and the web servers allow for better performance
- Spreads out your content so it is not in a single location

## Git & GitHub

## Git & GitHub

Git & GitHub have been brought up on multiple occasions and it may not have sunk in yet what is being talked about

Git = Version Control

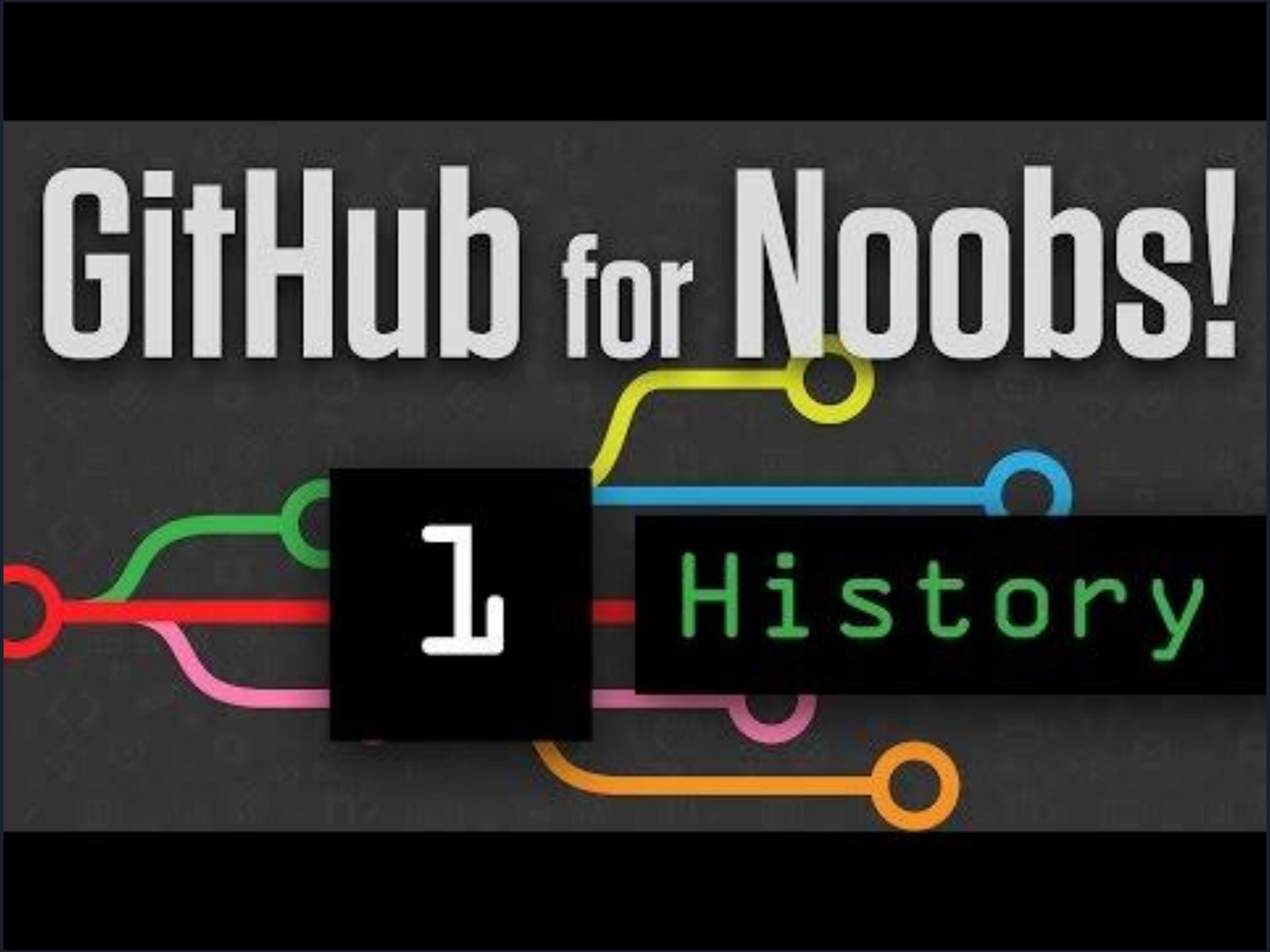
GitHub = Platform that uses Git

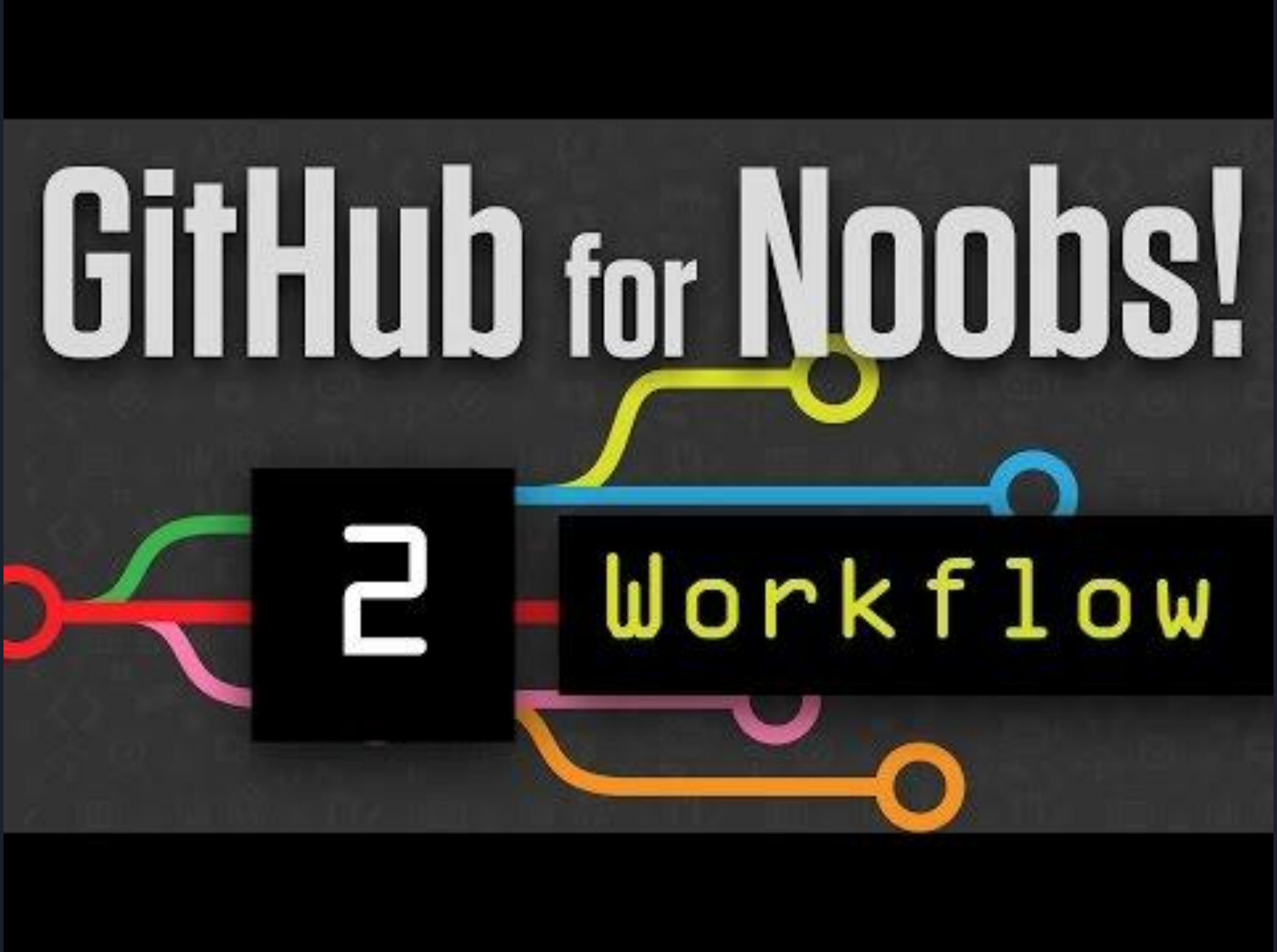
## What is Git?

“Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency” - git

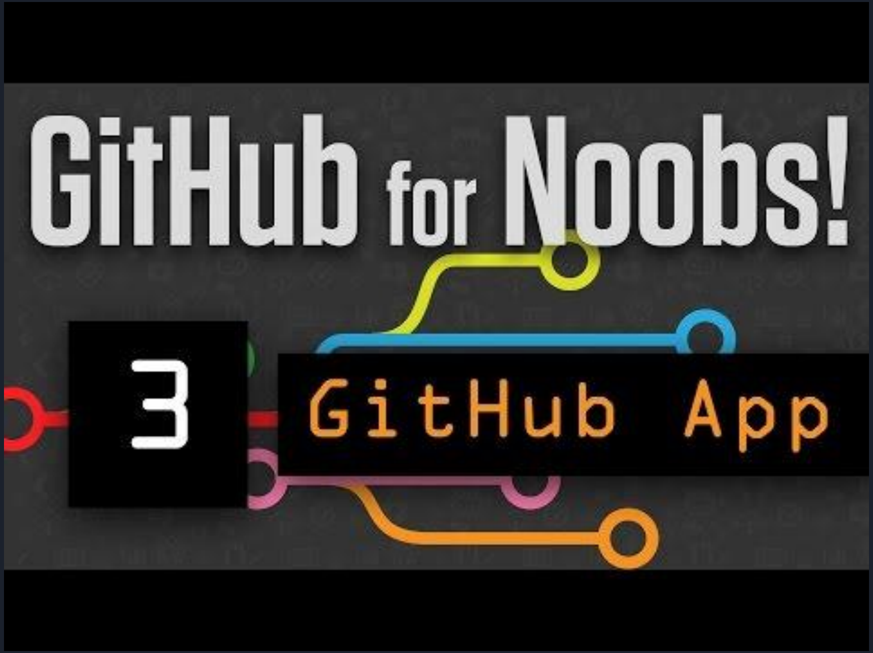
Git is a Version Control System

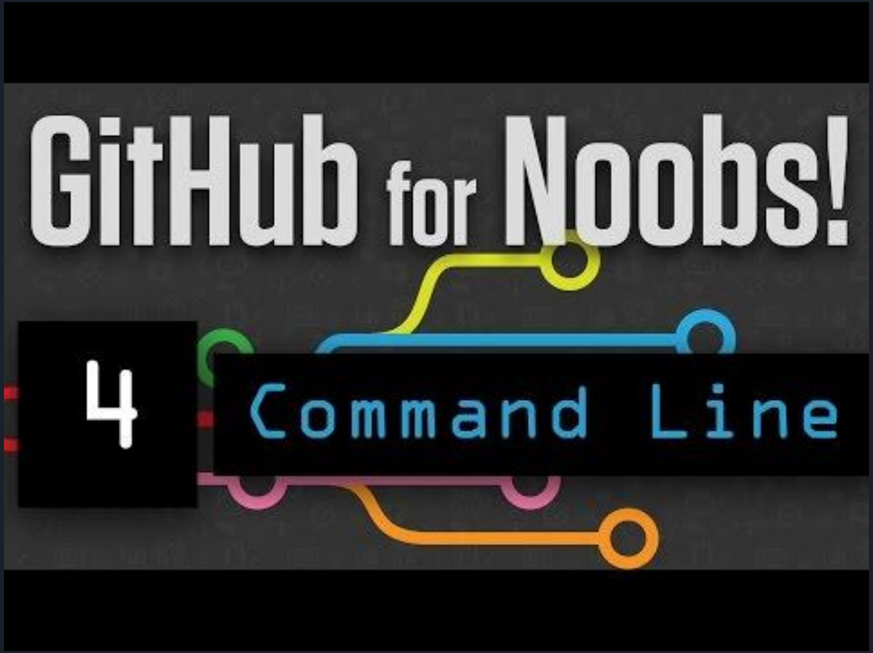
Version control is a system that records changes to a file or set of files over time so you can recall specific version later


GitHub for Noobs!
1
History


GitHub for Noobs!
2
Workflow

## Furthering Git and GitHub knowledge





## Deploy Site Using GitHub Desktop

## Preface

My workflow is a bit different than what will be demonstrated in this lecture. I do not use GitHub desktop professionally. However, that doesn’t mean it is not an acceptable tool to use in professional or personal environment. Tools are meant to get a job done and at the end of the day that’s all that matters is the job gets done.

Because of this you may find differing material and resources online. All that is completely acceptable to follow if it works for you or others.

Good YouTube Series:

1. Getting Started With GitHub Part 1
2. Getting Started With GitHub Part 2
3. Getting Started With GitHub Part 3

## What is GitHub Desktop?

GitHub Desktop is a Graphical User Interface for interacting with GitHub

Available on MacOS and Windows

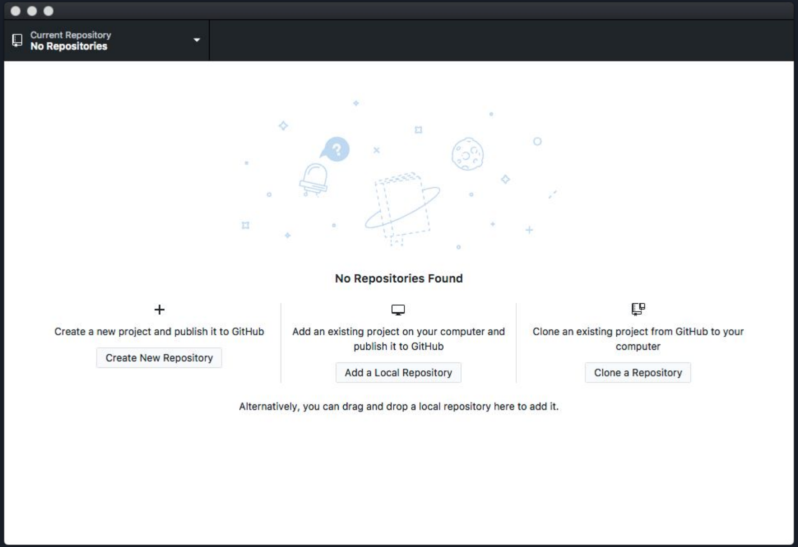
Designed by the folks at GitHub

Free

# Installing GitHub Desktop

1. Download GitHub Desktop
2. Install GitHub Desktop (locate GitHubDesktop.zip on your file system)
3. Once installed you will be prompted to enter your GitHub.com credentials
4. Step through the rest of the steps, you may receive an e-mail a 3rd party app has authorized your account

## Setting up your first Repo



No Repositories Found

Create a new project and publish it to GitHub

Add an existing project on your computer and publish it to GitHub

Clone an existing project from GitHub to your computer

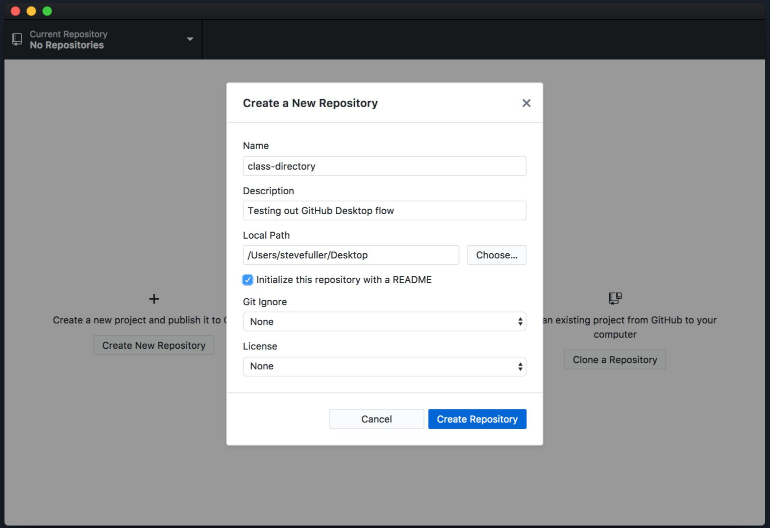
Alternatively, you can drag and drop a local repository here to add it.

- Create a new project and publish?
  - You don’t have Git files in your project
- Add an existing project and publish?
  - You have Git files in your project
- Clone an existing project from GitHub?
  - There’s code already on GitHub you want to download and work on

## Create a new project

- Name your project
- Write Description
- Where do you want your Repo to reside on your filesystem?
- Initialize with README?
  - A README is information about your project like installation instruction(s), how to use, FAQs, etc.
- Git Ignore
  - Files you don't want in GitHub
- License
  - Copyright protections

*Creating this will create a new folder with your essential Git files*


Current Repository
No Repositories
Create a new project and publish it to
Create New Repository
an existing project from GitHub to your computer
Clone a Repository

Create a New Repository

Name: class-directory

Description: Testing out GitHub Desktop flow

Local Path: /Users/stevefuller/Desktop

Choose...

[✓] Initialize this repository with a README

Git Ignore: None

License: None

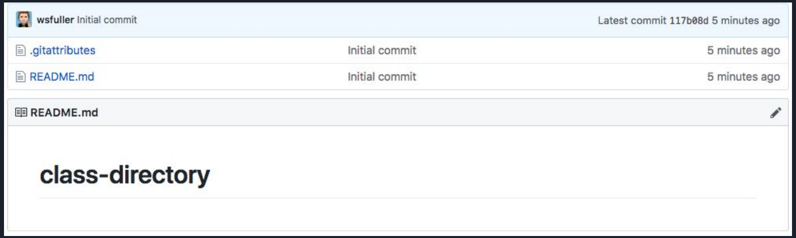
Cancel | Create Repository

# Publish Repository

In the top right corner press "Publish Repository"

Now go to:
github.com/:username/:repo-name

And you should see something like:

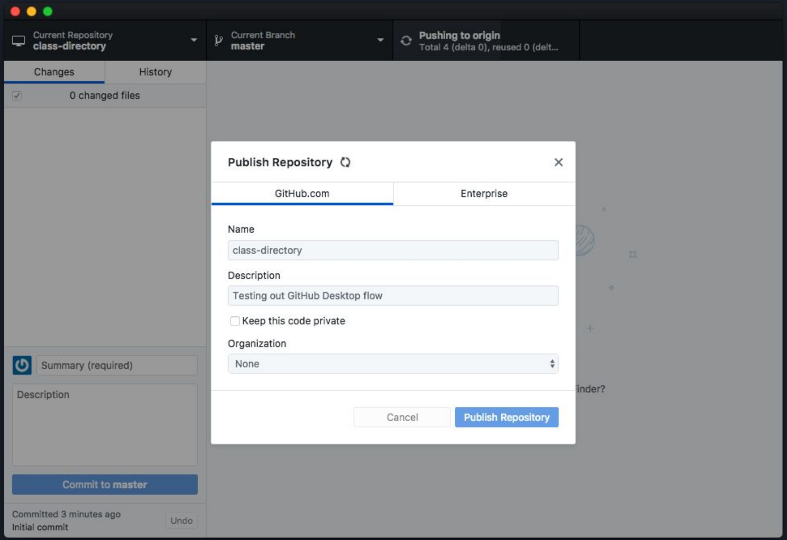


wsfuller Initial commit — Latest commit 117b08d 5 minutes ago

| | | |
|---|---|---|
| .gitattributes | Initial commit | 5 minutes ago |
| README.md | Initial commit | 5 minutes ago |

README.md

class-directory



Current Repository: class-directory
Current Branch: master
Pushing to origin — Total 4 (delta 0), reused 0 (delt...

Changes | History
0 changed files

Publish Repository

GitHub.com | Enterprise

Name: class-directory
Description: Testing out GitHub Desktop flow
☐ Keep this code private
Organization: None

Cancel | Publish Repository

Summary (required)
Description
Commit to master
Committed 3 minutes ago — Initial commit — Undo

## Congratulations you’ve just published to GitHub!!!

## But like, where’s my code and stuff?!?!

## One more thing... the .gitignore file

.gitignore is a file we use to omit certain files from being submitted to GitHub

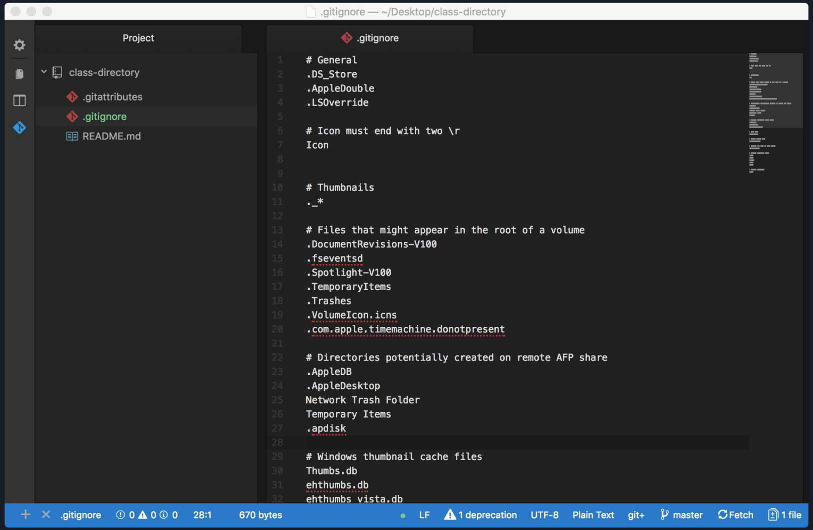
macOS gitignore

Windows gitignore

Here are a couple basic files that should cover our needs. For example though you don't want to be committing things like .psd, .sketch, etc files into GitHub. The .gitignore should always be updated before for code commits to avoid any unnecessary files from accidentally being committed.

## Adding your .gitignore

1. Open up your GitHub folder that was created in Atom
2. Make a new file called ".gitignore"
3. Paste in the sample code from the previous slide links
4. Commit this new file to your GitHub repo

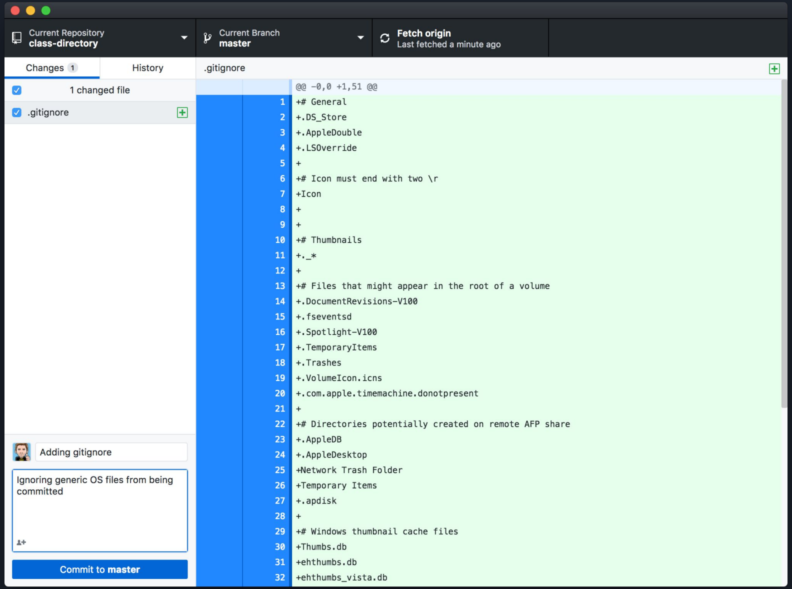


```
# General
.DS_Store
.AppleDouble
.LSOverride

# Icon must end with two \r
Icon


# Thumbnails
._*

# Files that might appear in the root of a volume
.DocumentRevisions-V100
.fseventsd
.Spotlight-V100
.TemporaryItems
.Trashes
.VolumeIcon.icns
.com.apple.timemachine.donotpresent

# Directories potentially created on remote AFP share
.AppleDB
.AppleDesktop
Network Trash Folder
Temporary Items
.apdisk

# Windows thumbnail cache files
Thumbs.db
ehthumbs.db
ehthumbs_vista.db
```

.gitignore 0 0 0 28:1 670 bytes LF 1 deprecation UTF-8 Plain Text git+ master Fetch 1 file

# Commit your .gitignore

1. Open GitHub desktop
2. Fill out Summary
3. Fill out Description (optional)
4. Commit to **master**
5. Click Push origin
6. Visit your repo



Adding gitignore

Ignoring generic OS files from being committed

```
@@ -0,0 +1,51 @@
+# General
+.DS_Store
+.AppleDouble
+.LSOverride
+
+# Icon must end with two \r
+Icon
+
+
+# Thumbnails
+._*
+
+# Files that might appear in the root of a volume
+.DocumentRevisions-V100
+.fseventsd
+.Spotlight-V100
+.TemporaryItems
+.Trashes
+.VolumeIcon.icns
+.com.apple.timemachine.donotpresent
+
+# Directories potentially created on remote AFP share
+.AppleDB
+.AppleDesktop
+Network Trash Folder
+Temporary Items
+.apdisk
+
+# Windows thumbnail cache files
+Thumbs.db
+ehthumbs.db
+ehthumbs_vista.db
```

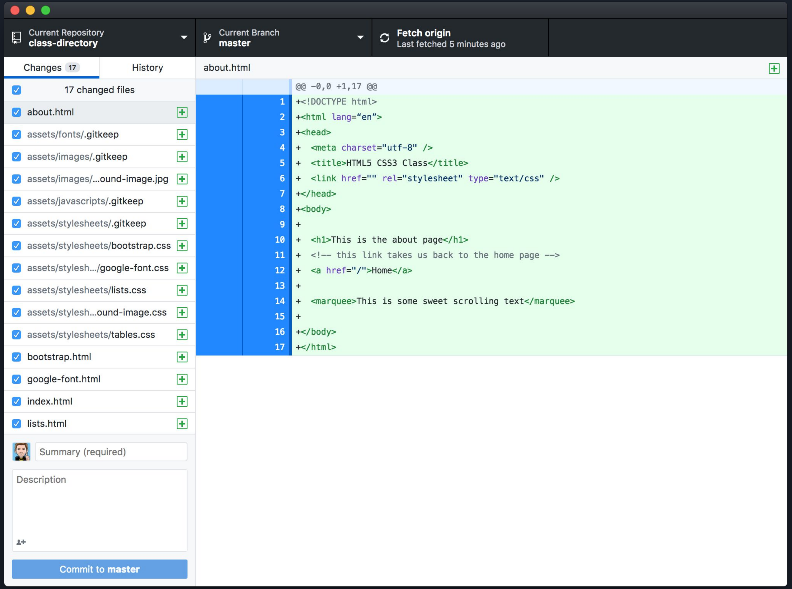
## You’ve just updated your repo!!!

## But like, where's my code and stuff?!?!

# Let’s get our actual code up

All we’ve done is prepare a folder that’s linked to GitHub, there’s a couple more steps to get this all working

1. Copy your working code into the one you just created for GitHub
2. Now you should see your files showing up
3. Summary
4. Description
5. Commit to **master**
6. Push to origin
7. Visit your GitHub Repo
8. Pat self on back



17 changed files

- ✓ about.html
- ✓ assets/fonts/.gitkeep
- ✓ assets/images/.gitkeep
- ✓ assets/images/...ound-image.jpg
- ✓ assets/javascripts/.gitkeep
- ✓ assets/stylesheets/.gitkeep
- ✓ assets/stylesheets/bootstrap.css
- ✓ assets/stylesh.../google-font.css
- ✓ assets/stylesheets/lists.css
- ✓ assets/stylesh...ound-image.css
- ✓ assets/stylesheets/tables.css
- ✓ bootstrap.html
- ✓ google-font.html
- ✓ index.html
- ✓ lists.html

```
@@ -0,0 +1,17 @@
+<!DOCTYPE html>
+<html lang=“en”>
+<head>
+  <meta charset="utf-8" />
+  <title>HTML5 CSS3 Class</title>
+  <link href="" rel="stylesheet" type="text/css" />
+</head>
+<body>
+
+  <h1>This is the about page</h1>
+  <!-- this link takes us back to the home page -->
+  <a href="/">Home</a>
+
+  <marquee>This is some sweet scrolling text</marquee>
+
+</body>
+</html>
```

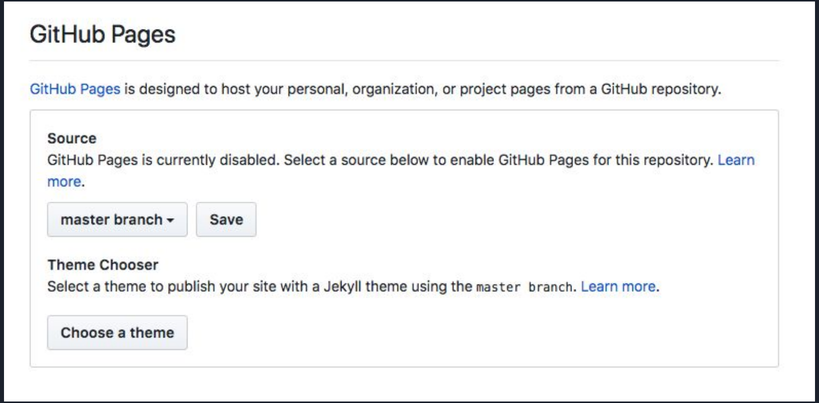
## Oh yeah

## But we aren’t quite done yet…

# Publish to GitHub Pages

GitHub pages will allow GitHub to take our code and publish a working website for everyone to view

1. Go to your GitHub Repo page
2. Click the Settings tab
3. Scroll down till you see “GitHub Pages”
4. Source should be your master branch
5. Click save
6. You should see a link: “Your site is ready to be published at :URL
7. Go grab a coffee and should be live in ~30 minutes



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Source
GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. Learn more.

Theme Chooser
Select a theme to publish your site with a Jekyll theme using the `master branch`. Learn more.

## Congrats on using GitHub and Deploying your site!!!

## Homework

- Submit: GitHub URLs (omit this till next week)
    - URL to your Repo
    - URL to your GitHub Page
- Read
    - Ch 17 & 18
- Code
    - About Page
    - Tour Dates