



Maydm

Web Development

Day 4: CSS Frameworks & Text Editors

Frameworks & Writing Code Like the Pros

Icebreaker!

Today's Schedule

Morning:

- CSS Frameworks
- Bootstrap
- Project: Portfolio Page
- Introduction to the Text Editor

Afternoon:

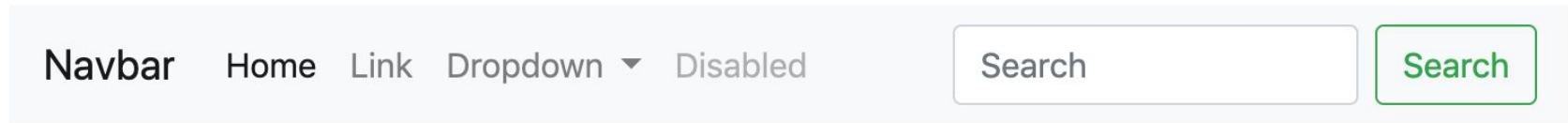
- HTML Head element
- Adding CSS links
- Moving projects to text editor
- Source control
- GitHub

CSS Frameworks

- CSS Frameworks provide a basic structure and prebuilt components to simplify creating webpages from scratch
- Why use a framework?
 - Saves time writing code
 - Responsive & mobile-friendly
 - Uniform styling across site
 - Cross-browser friendly
- Which framework to use?
 - Bootstrap -- built by Twitter, easy to learn
 - Materialize CSS -- easy to learn

Bootstrap Components

Navbars:

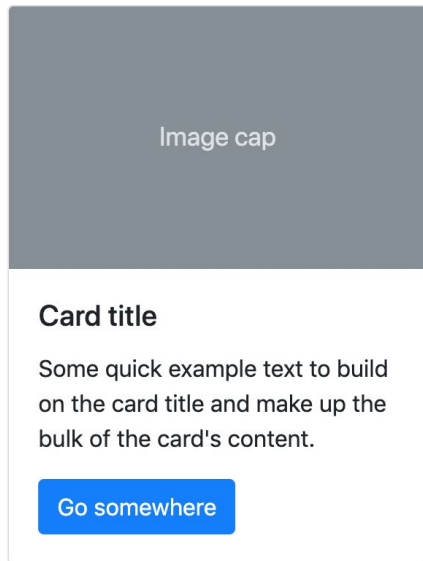


Buttons:

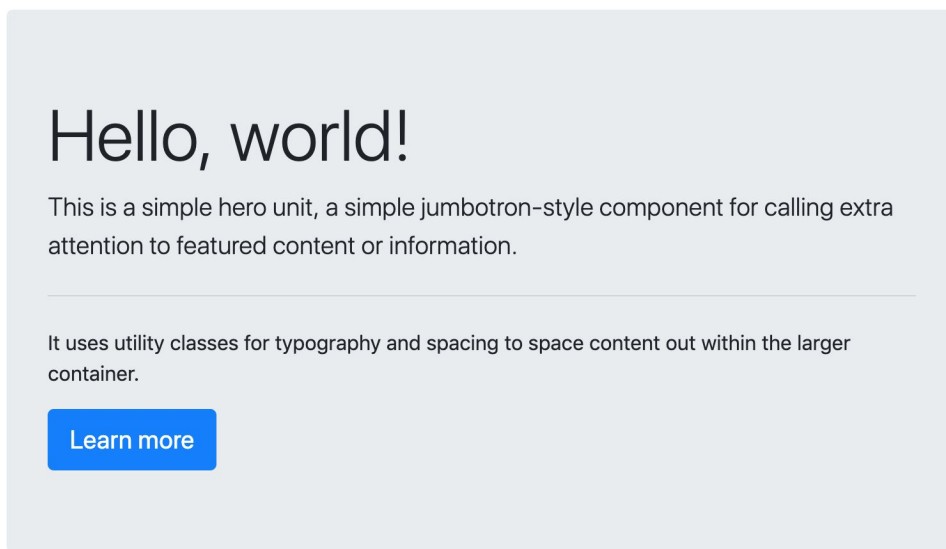


Bootstrap Components

Cards:



Jumbotron:



Bootstrap Utilities

Color:

`.text-primary`

`.text-secondary`

`.text-success`

`.text-danger`

`.text-warning`

`.text-info`

`.text-light`

`.text-dark`

`.text-body`

`.text-muted`

Text transform:

lowercased text.

UPPERCASED TEXT.

CapiTaliZed Text.

Bootstrap Utilities

- Width
- Height
- Alignment
- Margin
- Padding
- Borders
- Text align
- Font weight
- ...and more

Using Bootstrap

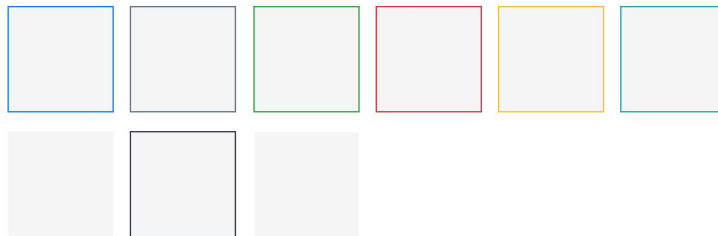
Bootstrap uses predefined classes.

Bootstrap includes excellent documentation with explanations of how to use all components and utilities.

Documentation includes example code and how that code will look.

Border color

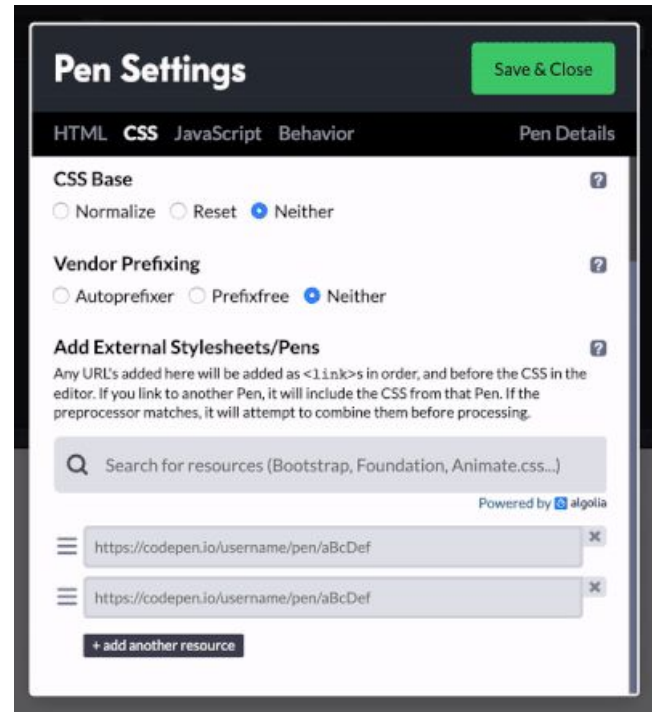
Change the border color using utilities built on our theme colors.



```
<span class="border border-primary"></span>  
<span class="border border-secondary"></span>  
<span class="border border-success"></span>  
<span class="border border-danger"></span>  
<span class="border border-warning"></span>  
<span class="border border-info"></span>  
<span class="border border-light"></span>  
<span class="border border-dark"></span>  
<span class="border border-white"></span>
```

Set up Bootstrap with CodePen

- Create a new Pen and click settings
- Go to the CSS tab
- Under “Add External Stylesheets / Pens” search for Bootstrap
- Click “twitter-bootstrap” to add the Bootstrap CSS to the project
- Switch to the JavaScript tab and repeat to add the required JS to the project



Internal Links

The `<a>` tag can also be used to send a user somewhere on the current page. To do that, the `HREF` attribute points to an ID on the current page.

```
<a href="#portfolio">Portfolio</a>
<a href="#about">About</a>

...

<section id="portfolio">
  <div>This is a portfolio item</div>
</section>
<section id="about">
  <div>This section is about me</div>
</section>
```

Project: Bootstrap Portfolio

- Adapt a basic portfolio page using Bootstrap
- Refer to Bootstrap documentation if you want to implement any components
- Use the navbar to add internal links -- links that go to a different part of the same page

Text Editor

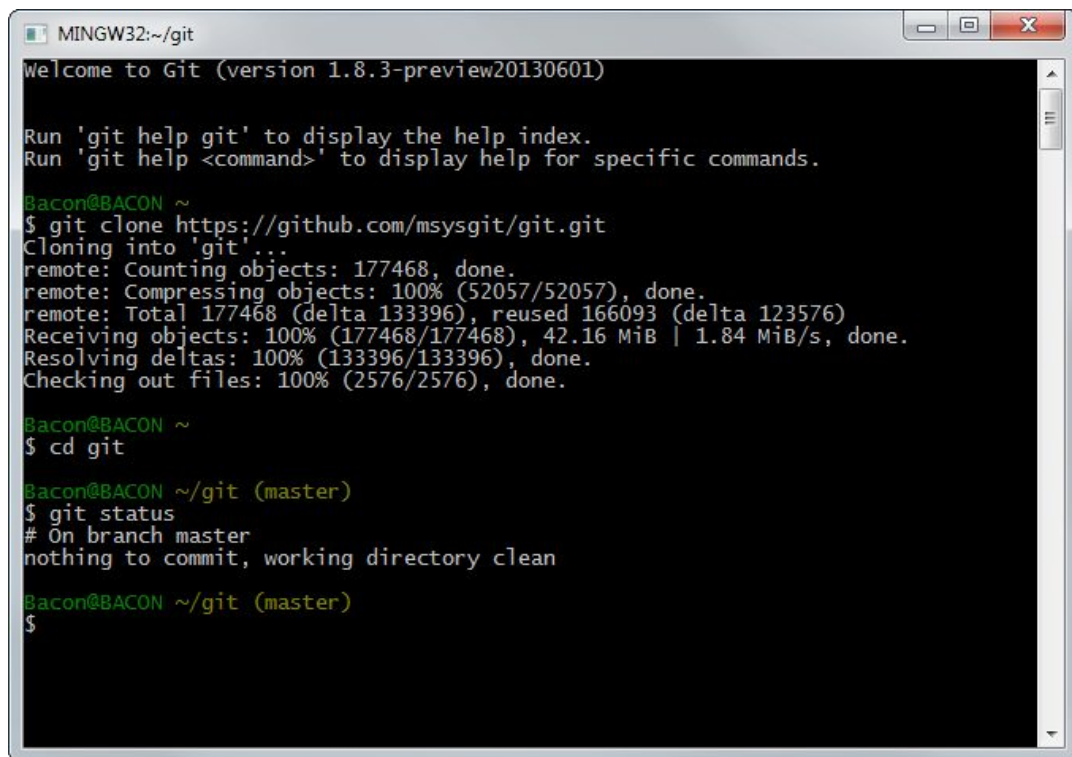
- What is a text editor?
 - A computer program that allows a user to edit plain text
- Is a text editor the same thing as a word processor?
 - No! Word processors includes formatting that can change how your code is read by computers.
- What text editor should I use?
 - We will be learning with Atom but other popular text editors include Visual Studio Code (free) and Sublime (unlimited free trial).

Command Line

Many professionals use the Command Line (aka the terminal) to do things, like creating files or folders, and backing up files.

We will be using Git Bash to do this.

Git Bash



```
MINGW32:~/git
Welcome to Git (version 1.8.3-preview20130601)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Bacon@BACON ~
$ git clone https://github.com/msysgit/git.git
Cloning into 'git'...
remote: Counting objects: 177468, done.
remote: Compressing objects: 100% (52057/52057), done.
remote: Total 177468 (delta 133396), reused 166093 (delta 123576)
Receiving objects: 100% (177468/177468), 42.16 MiB | 1.84 MiB/s, done.
Resolving deltas: 100% (133396/133396), done.
Checking out files: 100% (2576/2576), done.

Bacon@BACON ~
$ cd git

Bacon@BACON ~/git (master)
$ git status
# On branch master
nothing to commit, working directory clean

Bacon@BACON ~/git (master)
$
```


Command Line

```
:: make a new directory with "mkdir" plus the name of the folder  
$ mkdir new_folder
```

```
:: change into a directory with "cd" plus the name of the folder  
$ cd new_folder
```

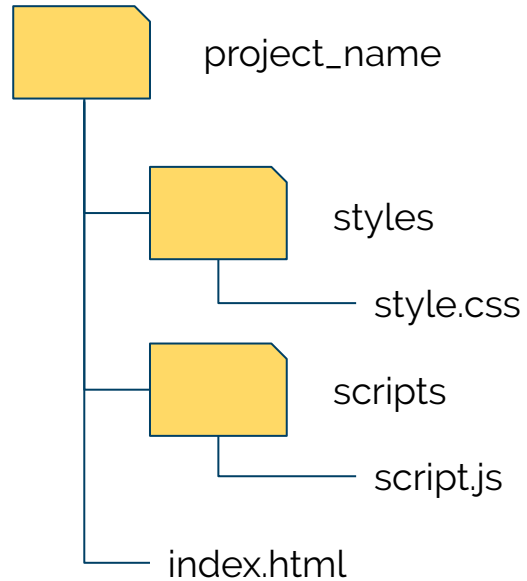
```
:: create a new file using "touch" plus the filename and extension  
$ touch index.html
```

```
:: open a directory with Atom using "atom" plus the directory name  
:: if you want to open the current directory, use a period  
$ atom .
```

Practice with Command Line

1. Create a directory using your first name
2. Move into that directory
3. Create a file called "README.md"
4. Open that file with Atom
5. Add to the Read Me file:
"Projects by **(your first name)** for Web Development, Summer 2019"
6. Save the file using ctrl-s

Folder Structure



HTML Files

So far we've been using CodePen to create sites, which is nice because it supplies some of the basic HTML code required.

Now that we're moving to the text editor and will be writing all our code, we have to add that code ourselves.

Note: The main HTML file for a project is always called index.html. The browser will look for this as the default file.

Anatomy of an HTML File

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>

    {Website goes here...}

  </body>
</html>
```

Anatomy of an HTML File

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title>Page Title</title>  
  </head>  
  <body>  
    {Website goes here...}  
  </body>  
</html>
```

The !DOCTYPE declaration tells the browser what kind of file it's receiving.

Older HTML versions require more specifications, but with the latest version, HTML5, "html" is all that's needed.

Anatomy of an HTML File

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>

    {Website goes here...}

  </body>
</html>
```

The html element is the root element of the web page. All other elements are nested inside it. Notice there's a closing html element as well.

Anatomy of an HTML File

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>

    {Website goes here...}

  </body>
</html>
```

The head element contains a lot of information about the page, including links to CSS and JavaScript files, the site title, and metadata about the site.

Anatomy of an HTML File

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    {Website goes here...}
  </body>
</html>
```

The website code stays in the body element.

All of the HTML from the CodePen projects exists inside a body element.

Adding CSS Files

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
    <link rel="stylesheet" href="styles/style.css" />
  </head>
  <body>
    {...}
  </body>
</html>
```

Project: Move to Text Editor

- Convert all CodePen projects to Text Editor
 - Fan Page, Mondrian, Poster, Portfolio Page
- Create folders for each project
- Add the “missing” HTML code for each project
 - DOCTYPE declaration
 - Head element
 - CSS links
 - Body element

Version Control

- What is version control?
 - Version control (aka source control) tracks changes in files.
- Why use it?
 - Go back to previous versions of code in case of bugs.
 - See differences between versions of files.
 - Try out new features without introducing bugs to stable code.
- Git is the most popular version control system.

Git + GitHub

By itself, Git creates **local repositories** (repos) but you can also add **remote repos** that are stored on the internet. This means your code is safe should something happen to your laptop.

GitHub is one of the most popular sites for web-based Git version control.

Sign up for a GitHub account at: <https://github.com>

Using Git

Git Tutorial for Windows:

<https://www.pluralsight.com/guides/using-git-and-github-on-windows>

Your First GitHub Repo


- Click on the green “New” button 
- Give your repo a name
 - Use the same name as your local folder
- Click on “Private” if you want to keep your code hidden

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner


Repository name *

 DisplacedTexan ▾


 /

Great repository names are short and memorable. Need inspiration? How about **potential-octo-eureka**?

Description (optional)

☒  **Public**

Anyone can see this repository. You choose who can commit.

☐  **Private**


You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

 |

Add a license: **None** ▾ 

Create repository

Worktime

- Create GitHub repos for all of your text editor projects.
- For each project, add the remote URL to the local Git repo and push the repo.
- Finish transferring your CodePen projects to the text editor.

Reflection

Write in your journal about how you feel or what you learned today.

Prompts:

- What do you think of Bootstrap and CSS frameworks?
- Would you rather use Bootstrap or write your CSS from scratch? Why?
- What did you find challenging about switching to the text editor from CodePen?
- Think about Git and version control. Why would you want to have multiple branches for a project?