GRAY AREA
FOUNDATION
FOR THE ARTS

Download the following…

- SublimeText
  - http://www.sublimetext.com/3
- Github for Mac
  - https://mac.github.com/

This Week

- Github/Github Pages
- HTML, Javascript, CSS
- Bootstrap Framework

Github

- Github is a service which stores code for your personal or public projects in the cloud
- Github utilizes git, a command-line tool for code for developers to maintain their code projects

Github Pages

- Github introduced Pages as way for developers to use their Github repositories as a platform for showcasing the work they were engaged in

Who uses Github Pages?

- Anyone with a Github Repository!
- Open source tools
- CVs/Resumés for developers

Why Use Github Pages

- Free!!!
- Easy to setup up and maintain
- We can share our work easily

First, Git!

- Git is primarily a Version Control tool
- Projects are stored in Repositories
- Developers work on Branches
- Developers Commit and Checkout code

First, Git!

- Github uses Git, as the name implies
- Github serves as the master storage for all repositories, allowing users to checkout these repositories locally

Github App or Terminal?

- Both choices are fine, depends on how comfortable you are with the command line
- For ease I’m going to teach from the app

Installing Github

- Download the Github app, extract, and install!

Check out that repository!

- Let’s go to the Repo!
- https://github.com/GrayAreaorg

How Github's App Works

- Github's app is a front-end for the command-line git tools
- The app allows you to interact with repositories, and push changes to code back to Github

How Git Works

- The first thing you do to an existing repository is clone it
- Cloning downloads the repository, but also sets it up with git to track changes
- You can always simply download as zip, but no changes can be tracked

Git and Branches

- Git uses branches, which allows multiple developers to work on the same code simultaneously
- Branches can be **merged** together when development is completed

Git and Branches

- Creating a new branch in the Github App is as simple as clicking the “Create new Branch icon”

How Git Works

- When you change your code, you can **commit** the code with git, which stores all of your changes
- Once complete, you can push these commits back to Github using the “Sync” button

How Git Works

- When your code is complete in your branch, you can merge your code back into the **master** or default branch

How Git Works

Conflicts, the bane of my existence

README.md

- Github, by convention, uses a file called README.md to describe your repository
- The .md file stands for **markdown**, a language for simple documents

README.md

- Some basic markdown:
  - #, ##, ### - Headings
  - > - Section
  - * - Bulleted List
  - [Link Text](http://www.something.com)
  -

Let’s Update our Pictures

- Checkout the Repo
- Create a new Branch
- Add your picture and information
- Commit and Sync
- Create a pull request

Creating a Project/Repo

- Create from Github
- Go to folder
- Create a README.md
- Add the files you wish

SublimeText

- Open up Sublime
- From finder, drag your Project Folder onto the Sublime Text icon

The Tools

- While we are coding, we can view our HTML page by double-clicking on any HTML file.
- I am going to suggest that we use Chrome to standardize, but this is not required
- That being said, the Chrome Development Tools are AMAZING!

VERY Basics of HTML

- In Sublime, create a new file called index.html
- Type html, then press the Tab key
- Add some text between <body> </body> and save

How Github Pages Works

- Github Pages requires a specific branch name:
    - gh-pages
- When we add an index.html file to our repo in this branch, Github renders it as a webpage

Pushing to Github Pages

- http://yourusername.github.io/repositoryname

Pushing to Github Pages

- It's good to point out how to make a Github page for your user account…
  - Create a repo called:
    - YOURUSERNAME.github.io

HTML

- HTML is a markup language
- Web browsers take this markup and turn it into something readable

HTML

- HTML consists of a head and body section
- The head contains metadata for the page, including the ability to load external files (i.e. javascript and css)

HTML and <tags>

- HTML is based on XML, which wraps data inside of ‘tags’
- There is a start tag (<tag>) and an end tag (</tag)
- Examples:
    - <html>, <head>, <body>
    - <h1>, <p>, <b>,

Head

- The head section has metadata related to the page itself

Body

- The body contains all the content of the page, including HTML, CSS, and Javascript

Tags for Structure

- Heading, paragraph, and list tags are examples of structural tags
- They tell the web browser how the text should be laid out on the page
- Structural tags come with default behavior in all the browsers

A brief bit about browsers….

Headings

- Headings denote titles, and other explanatory text
- <h1> through <h6>

Paragraph

-
- Denotes a separated area of text

Tags for Style

- Some tags do not dictate where content lives on a page, but instead adds style to given bits of text
- This typically will simply change how the content looks

Bold

-

Italic

-

Lists

-
-
-

Images

-

How paths work

- HTML looks for a folder structure similar to the URL you are requesting
  - index.html
    - —> /index.html
  - /bio
    - —> /bio/index.html
  - /bio/contact-info/
    - —> /bio/contact-info/index.html

Linking with Anchors

- Links in html are called “Anchors”, and use a tag like this:
- Link Text
- Links can either use relative paths, or absolute paths

Time to Code!

- Create two webpages:
  - index.html
    - 1 Heading, one paragraph, and an image
    - 1 Link to the bio
  - bio/index.html
    - 1 Heading, 1 paragraph, some bold, and italicized text

Let's Commit!

- Time to push our code to Github!
- Let's push this to the repository, using the gh-pages branch

A little bit of style…

- Long ago, style was handled within HTML tags
- Using the style attribute in HTML, we can add styles to our HTML

A little bit of style…

- Style can happen in 3 ways:
  - “style=“ attribute
  - <style> tag
  - <link> tag (external .css file)

A reminder for later…

- Style Order of Precedence:
  - style attribute (Highest)
  - style tag
  - css external file (Lowest)
- Style attributes are overwritten if repeated

CSS Styles

- Let’s set an entire paragraph to have bold text

CSS Styles

- Style attributes look like:
  - “background-color: green; text-align: center”

<divs> and

<divs> and

- The div and span tags do not add any default behavior, but allow you to use CSS to control each section
- divs are good for whole sections of text
- spans are good for text which exists within other text

<divs> and

- What if I wanted to control more than one div or span?

CSS Selectors

- CSS allows you to control multiple HTML elements at once
- CSS uses selectors to determine what to modify

CSS Selectors

- CSS Selectors come in 3 types:
    - elements
    - classes
    - ids
- A combination of these types can refer to specific elements

Class and ID

- When using class or id attributes in CSS, your HTML tags will need to specify which class or id references each element
- Remember:
    - Class means more than one element
    - ID means only one element is affected

CSS Selectors

- Element
  - i.e. “body”
- Class
    - .CLASS_NAME
- ID
    - #ID_NAME

CSS Selectors

- The body tag:
    - body: { }
- All divs with the “important_text” class
    - div.important_text : { }
- The element with the id: “important_picture”
    - #important_picture: { }

I’m Floating!

- float: left, right
- floating allows text, images, etc. to exist on the same line

Padding

- padding-left,-right,-top,-bottom
- Separates elements from each other for style and comfort

Putting our styles in one place

- The style tag allows you to create global styles for your webpage

Time to Code

- Let's create an updated version of our two web pages
    - Use divs to separate content
    - create a style section in the head for your page
    - Float the image to the left of your text, so it lines up neatly horizontally
    - Use padding to separate your content