Housekeeping

Classroom Culture

Let's all agree to...

- Be **PRESENT** and **INVOLVED**
- Be **RESPECTFUL** of other people's time
- **HONOR** your commitments
- Be **SUPPORTIVE** of each other

Setup Instructions

- Make sure your camera is on
- Download Github Desktop
- Download Visual Studio Code text editor
- Download Chrome
- Create free CodePen account

CSS Basics

Lesson 2

FEWD

GENERAL ASSEMBLY

Learning Objectives

- Describe the DOM and draw a simple DOM tree
- Understand the difference between absolute & relative paths, and apply relative paths to <img> and <a> tags
- Apply and explain CSS "cascade" including: importance, specificity, and inheritance
- Link to an external CSS stylesheet

Agenda

- Review Lesson 1
- Understanding the DOM
- Building a Simple Web Page
  - Image/link paths, Images, CSS, Colors, etc.
- Lab Time
  - GHE + Dev Environment Setup

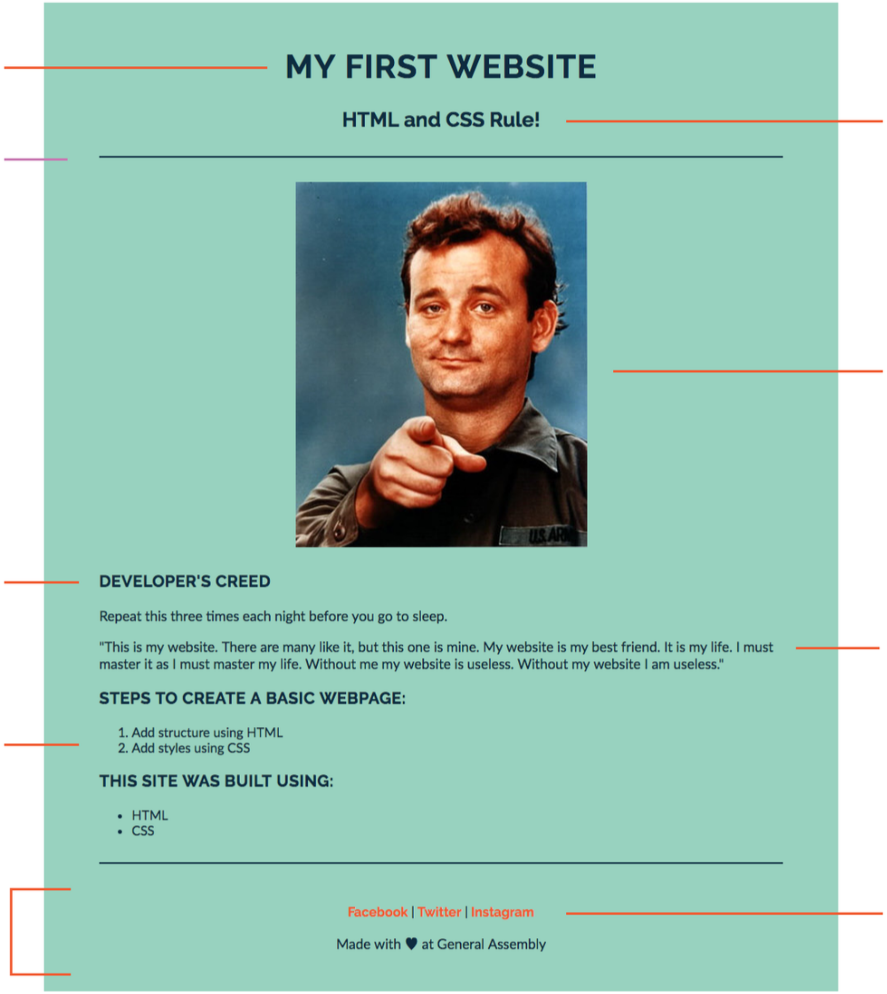
Lesson 1 Review

Lab Solution Review

Open: My First Website Solution CodePen

BODY:

BACKGROUND COLOR - #98D2BF
COLOR - #0D2C40
PADDING TOP - 20PX
PADDING RIGHT - 60PX
PADDING LEFT - 60PX



MY FIRST WEBSITE

HTML and CSS Rule!

DEVELOPER'S CREED

Repeat this three times each night before you go to sleep.

"This is my website. There are many like it, but this one is mine. My website is my best friend. It is my life. I must master it as I must master my life. Without me my website is useless. Without my website I am useless."

STEPS TO CREATE A BASIC WEBPAGE:

1. Add structure using HTML
2. Add styles using CSS

THIS SITE WAS BUILT USING:

- HTML
- CSS

Facebook | Twitter | Instagram

Made with ♥ at General Assembly

TEXT TRANSFORM - UPPERCASE
TEXT ALIGN - CENTER
FONT SIZE - 40PX
LETTER SPACING - 1PX
FONT FAMILY - "RALEWAY"

FONT FAMILY - "RALEWAY"
TEXT ALIGN - CENTER
FONT SIZE - 25PX

LOOK UP HR ELEMENT
BORDER - 0
BORDER BOTTOM - 2PX SOLID #0D2C40
MARGIN TOP - 30PX
MARGIN BOTTOM - 30PX

TO CENTER IMAGE:

```
display: block;
margin: 0 auto;
```

FONT FAMILY - "RALEWAY"
TEXT TRANSFORM - UPPERCASE
LETTER SPACING - .5PX
FONT SIZE - 20PX

FONT FAMILY - "LATO"
FONT SIZE - 17PX

FONT FAMILY - "LATO"

LOOK UP FOOTER ELEMENT
TEXT ALIGN - CENTER
PADDING - 20PX

TEXT DECORATION - NONE
FONT WEIGHT - BOLD
COLOR - #F15B31

Understanding the DOM

What is the DOM?

Document Object Model

- When a web page is loaded, the browser renders a **Document Object Model (DOM)** of the page in HTML.
- It represents the page so that programs can change the document structure, style, and content.
- The DOM model is constructed as a tree of objects (i.e. elements)

Basic HTML Document Structure

HTML Structure

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        <h1>Main Heading</h1>

        <p>This is my first paragraph.</p>
    </body>
</html>
```

DOCTYPE

Description

- Tells browser this file is written in latest version of HTML — HTML5

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

Best Practice

- Must be the very first thing in your HTML Document!

HTML Element

Description

- **ALL** of your HTML code goes between these tags

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

Head Element

Description

- For behind-the-scenes info
- Metadata that's **not displayed**
- Info used by browser and search engines

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

Best Practice

- Only one of these in each HTML file

Meta Charset

Description

- Tells our browser which character set to use
- Should always use UTF-8

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

Title

Description

- Tells the browser what this page is called
- Used for browser tabs, browser bookmarks, and search engines

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

Body

Description

- Wraps all content for our site
- **Everything we want displayed in the browser window goes between the body tags** -- i.e. <h1>, <p>, <img>, etc.

Best Practice

- Only one of these in each HTML file

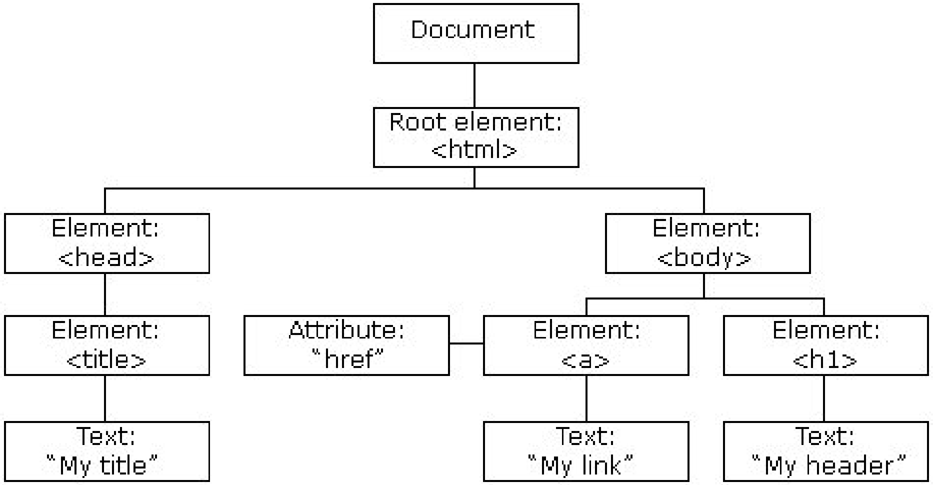
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

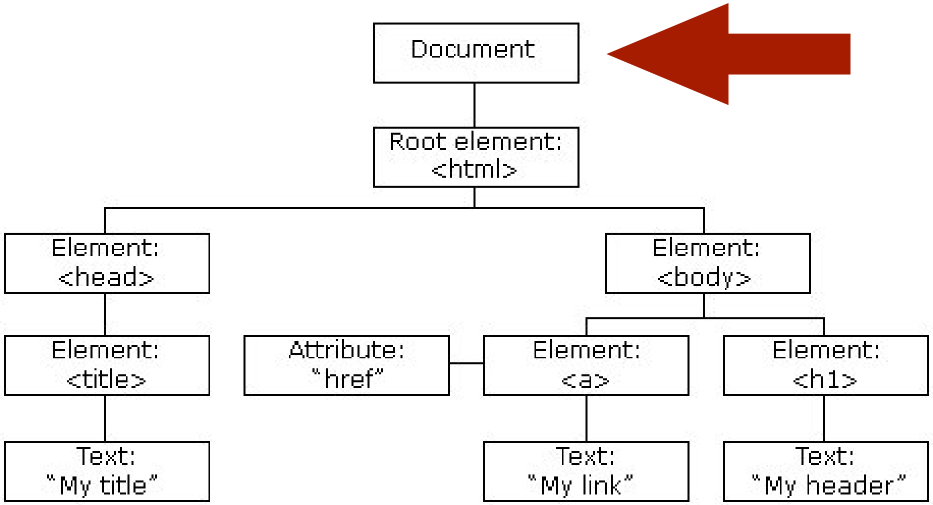
Body (cont'd)

All of the HTML you want displayed in the browser **MUST** go in between the <body> tags!!!

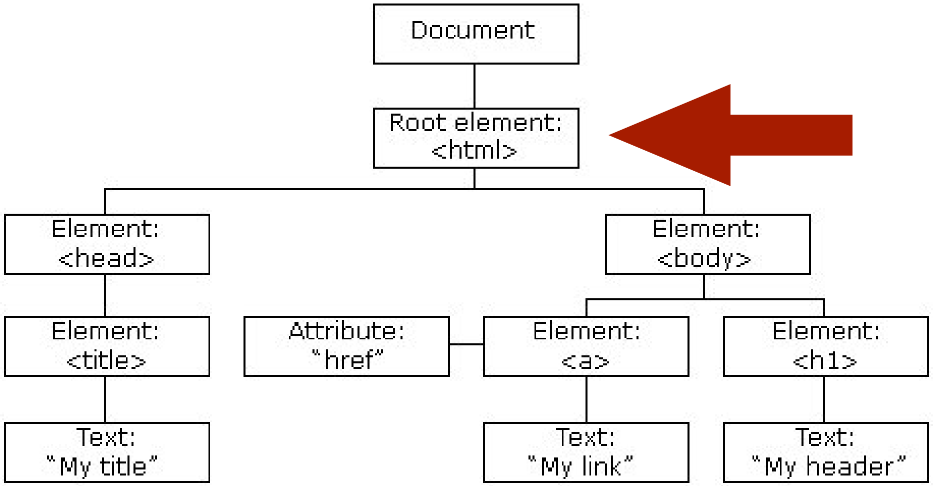
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        ...
    </body>
</html>
```

DOM Tree

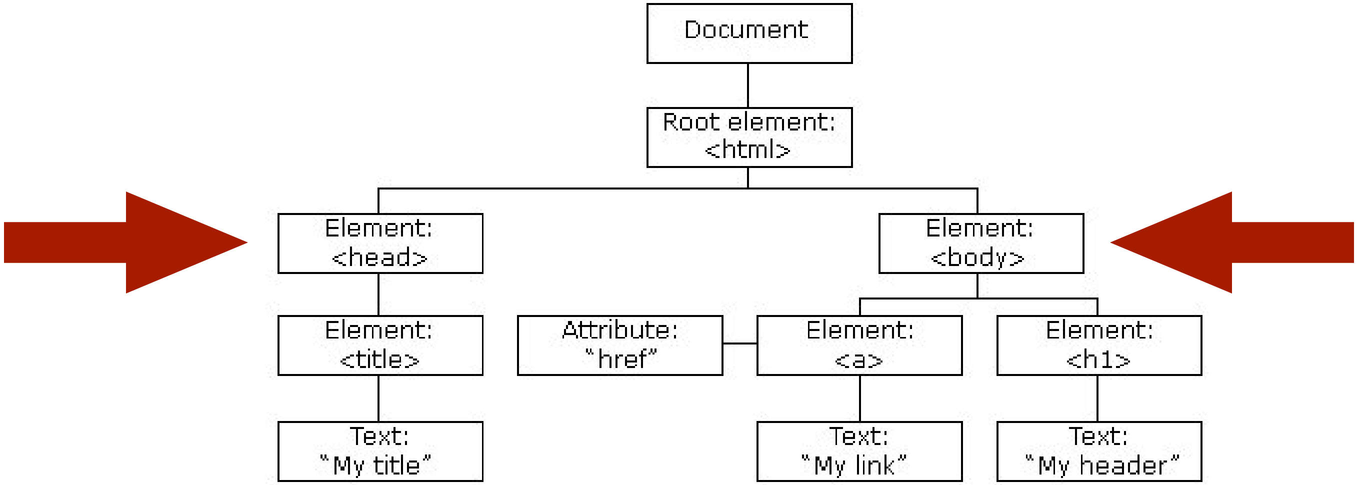

Document
Root element: <html>
Element: <head>
Element: <body>
Element: <title>
Attribute: "href"
Element: <a>
Element: <h1>
Text: "My title"
Text: "My link"
Text: "My header"



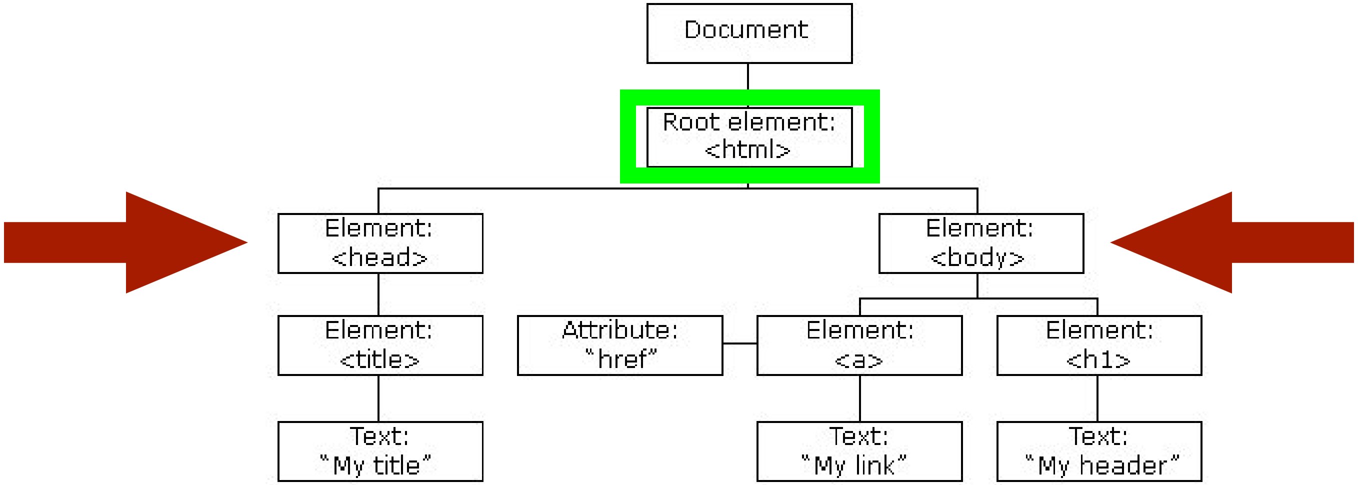
At the top, we have the **document object**



Next, we have the **root element**, which is always the **html** tag



Then comes the **head** and **body elements**. Notice they are on the same level -- this makes them **siblings**



Also, notice this makes them both the first **children** of the **html element**

The DOM Family Tree

Pulse Check

Key Objective:

In groups of 2-3, sketch a DOM tree for the following basic HTML document. Discuss the relationships between each of the different elements. Be prepared to share with the class.

Timing:

- **5 minutes** - Discuss with a partner in a breakout room
- **~1 minute** - Share your response with the class via the main Zoom room

Bonus:

Come up with another analogy that can describe the DOM!

HTML Document

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>My Title</title>
    </head>
    <body>
        <h1>Main Heading</h1>

        <p>This is my first paragraph.</p>
    </body>
</html>
```

Why should I care?

- Parent/child relationship of HTML elements
- CSS specificity and inheritance
- Javascript/jQuery integration
- Readability of HTML Code (indentation)

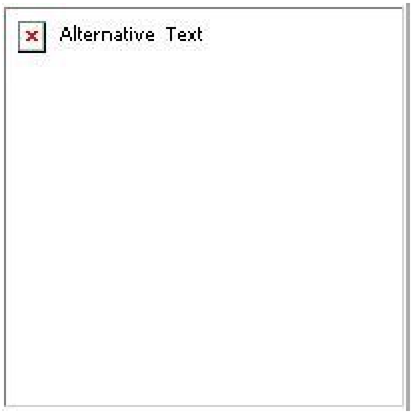
Building Websites: Image Paths

HTML images are placed using the **img** tag:

```
<img src="folder/image-name.jpg" alt="Alternative Text">
```

The **img** tag requires a **src** attribute to tell the browser where to find the image.

The **alt** attribute is optional, but strongly encouraged

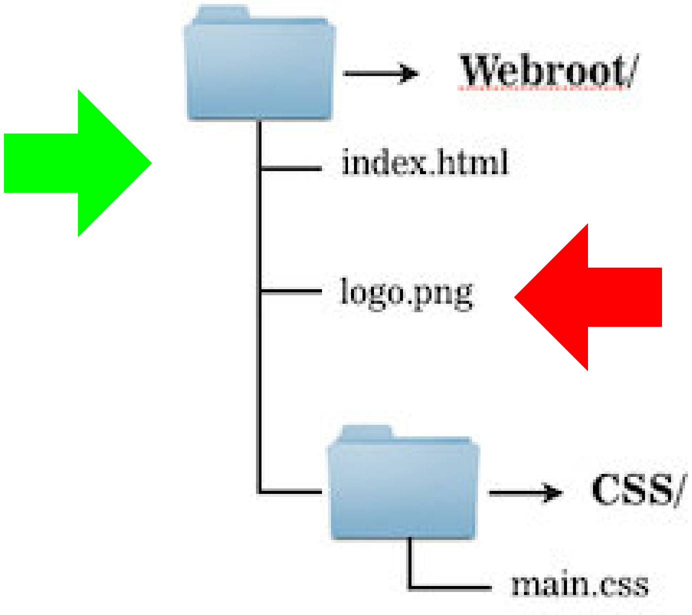


Alternative Text

Image Path URLs

Slack Check #1

How might you write the **src** path to access the **logo.png** from **index.html**?



CORRECT!

This is an example of a **relative URL**

```
<img src="logo.png">
```

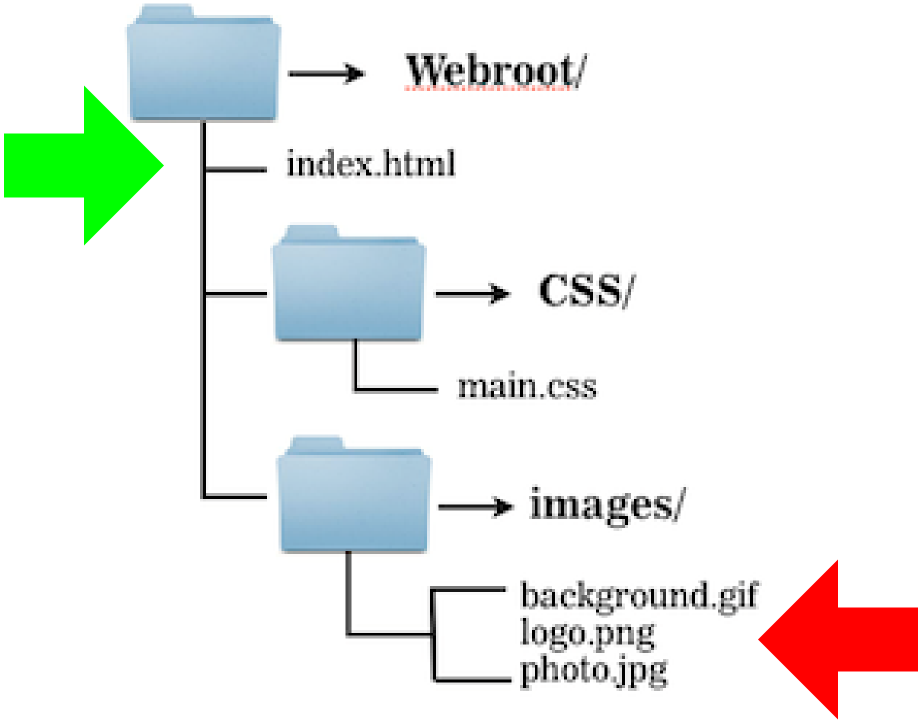
Relative vs Absolute URLs

Relative URLs

- A **relative URL** is any URL that doesn't explicitly specify the protocol (e.g., " http:// " or " https:// ") and/or domain (www.example.com)
- Meaning, as long as the image you are pointing to is inside your **webroot directory**, you can use a relative path

Slack Check #2

What about this example -- how might you write the **src** path to access the image file here?



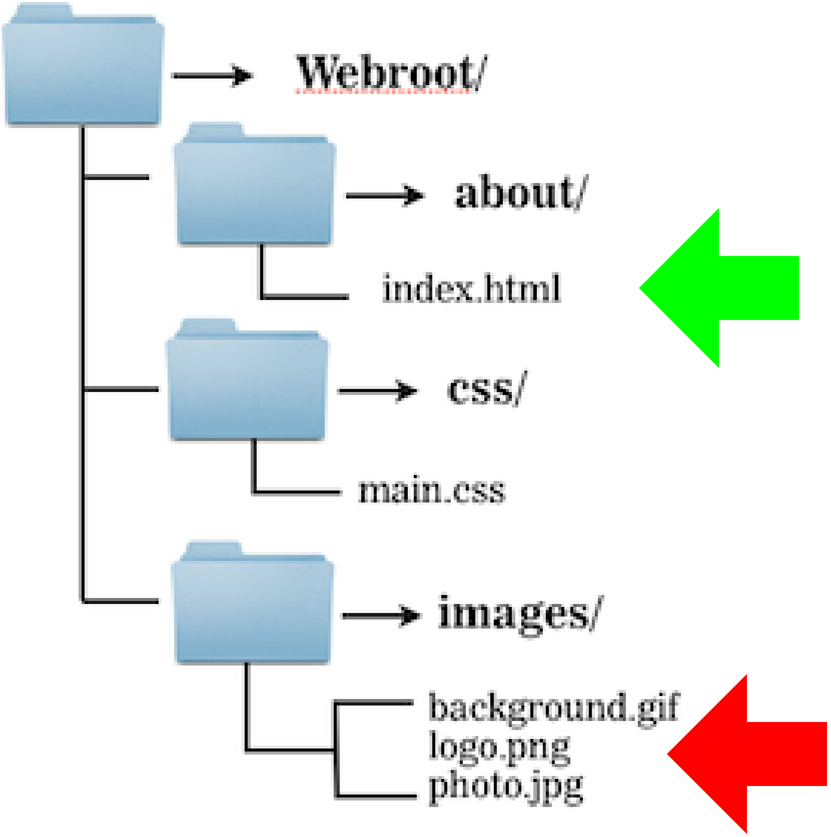
CORRECT!

Since it is in a subfolder in relation to **index.html**, we need to include that subfolder in our path:

```
<img src="images/logo.png">
```

Slack Check #3

Last one — what is the correct **src** path to access the "logo.png" file here?



CORRECT!

In this case, the **index.html** file is in its own subfolder.

When this happens, you need to go up a directory, and THEN go down into the subfolder containing the image.

```
<img src="../images/logo.png">
```

To go up a directory, we use 2 dots: **".."**

You can also chain these to go up multiple directories

For example, the following would go up 3 directories:

```
<img src="../../../folder-name/image-file.png">
```

Absolute URLs

- An **absolute URL** is a full URL that references any file located on an external or internal domain.
- Unlike **relative URLs**, they include the protocol + domain in the URL

```
<img src="http://www.example.com/images/logo.png">
```

REMEMBER!

These same rules apply where ever a **URL** is used in web development.

Building Websites: Image files & Colors

.jpg

Great for photos

.gif

Supports basic transparency and animations

.png

Supports transparency and semi-transparency

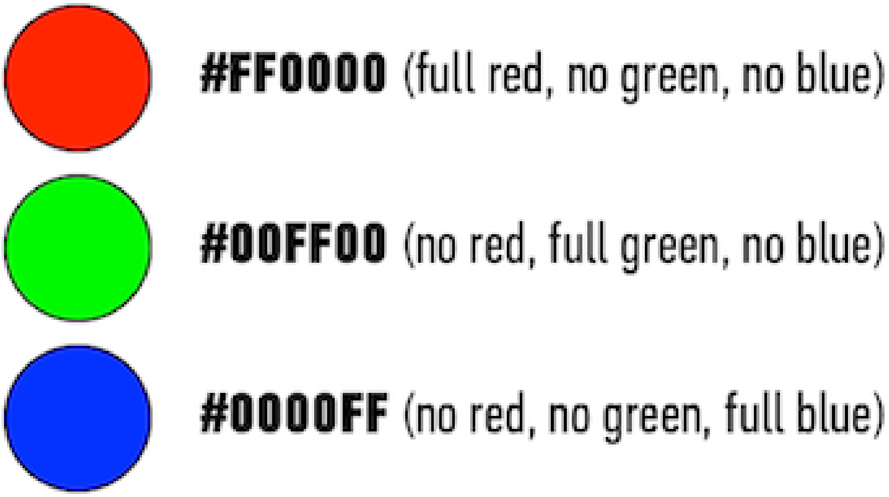
.svg

Great for logos and icons

Web Colors

- Color keywords
- Hexadecimal
- rgb/rgba

Hexadecimal



#FF0000 (full red, no green, no blue)

#00FF00 (no red, full green, no blue)

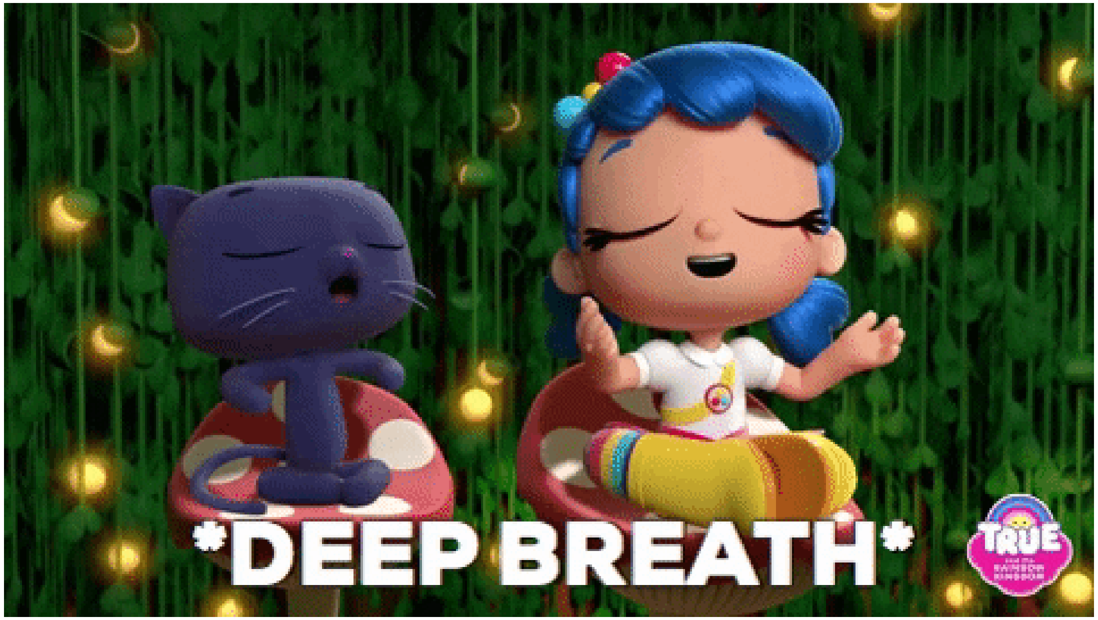
#0000FF (no red, no green, full blue)

RGB

```
rgb(255,255,255)
```

RGBA

```
rgba(255,255,255,0.5)
```



Break time!

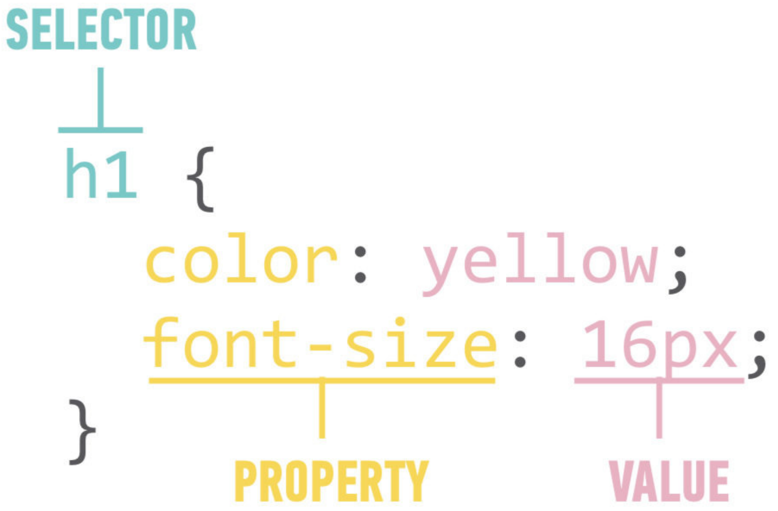
Let's take 5-10 minutes to decompress...

Building Websites: CSS

Cascading Style Sheets

- CSS stands for **Cascading Style Sheets**
- It allows you to **style** your content

CSS Rule Syntax



```
h1 {
  color: yellow;
  font-size: 16px;
}
```

CSS Properties

PROPERTY	VALUES	EXAMPLE
text-align	left, center, right, justify	text-align: center;
text-transform	uppercase, lowercase, capitalize	text-transform: uppercase;
text-decoration	none, underline	text-decoration: underline;
letter-spacing	px value	letter-spacing: 1px;

PROPERTY	VALUE	DESCRIPTION	EXAMPLE
color	color	Text color	color: #22475E;
background-color	color	Background color	background-color: green;

PROPERTY	VALUES	EXAMPLE
font-weight	normal, bold	font-weight: bold;
font-style	regular, italic	font-style: italic;
font-size	px value	font-size: 20px;

PROPERTY	VALUES	EXAMPLE
font-family	"font name", sans-serif, serif	font-family: "Georgia", serif;

Arial Georgia

CSS Font-Family Property

```css
h1 {
    font-family: "Arial", sans-serif;
}
```

- Font-family property specifies the font for a specific element
- Can specify a number of different fonts as a "fallback"
- Always start with the font you want, and then end with a generic font-family

CSS Using Custom Fonts

```css
@import url('http://www.website.com/custom/font/example');

h1 {
    font-family: "Custom Font Name", "Arial", sans-serif;
}
```

- When using a custom font, first link to it (in CSS or HTML)
    - Google Fonts is your friend (trust me)
    - Be sure to include any variations of it
- Then, specify the custom font name in your CSS declaration

Directions:

Let's change the custom font for the h1 element in our example Codepen from earlier: My First Website (Advanced)

- As a class, help me make the h1 element use the "Roboto" custom font
- BONUS: Make the font-weight extra-bold

Where does CSS go?

- External CSS
- Internal/Embedded CSS
- Inline CSS

External CSS

External CSS are styles placed in a **separate .css file**, and then linked to the HTML file via the **link tag**.

```
<!-- HTML file -->
<head>
    <link rel="stylesheet" href="style.css">
</head>
```

```
/* style.css file */
p {
    color: red;
    font-weight: bold;
}
```

Internal CSS

Internal CSS are styles placed directly in your HTML file via the **style tag**.

```
<!-- HTML file -->
<head>
    <style>
        /* Internal styles */
        p {
            color: red;
            font-weight: bold;
        }
    </style>
</head>
```

Inline CSS

Inline CSS are styles placed directly in an HTML element via the **style attribute**.

```
<!-- HTML file -->
<p style="color: red; font-weight: bold;">
    This is a paragraph.
</p>
```

Pulse Check

Directions:

I have a 2 page website -- which method should I choose to style the following elements? Why?

1. All **h1** elements on all pages
2. A specific **a** link on the home page
3. All **p** elements on the home page
4. All **img** elements on all pages
5. Specific **h3** element on the About page

CSS Specificity & Inheritance

"Cascading" Style Sheets...

Specificity

Essentially, how the browser calculates which property values to display in the event of a conflict.

Specificity Example #1

```
<!-- HTML content -->
<p>
    Hello there, <span>Mansoor!</span>
</p>
```

```
/* CSS styles */
span {
    color: blue;
}

...


span {
    color: red;
}
```

Specificity Example #2

```html
<!-- HTML content -->
<p>
    Hello there, <span>Mansoor!</span>
</p>
```

```css
/* CSS styles */
p span {
    color: blue;
}

...


span {
    color: red;
}
```

Code Along

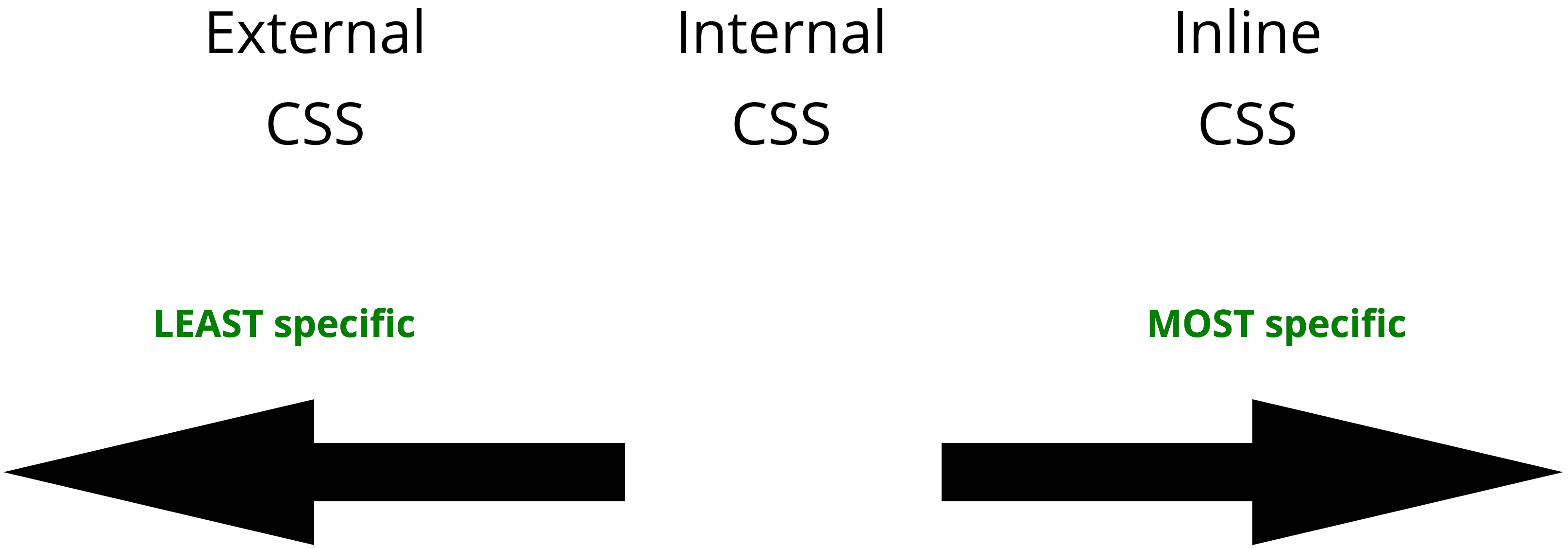
Open: CSS Specificity & Inheritance CodePen

Specificity (cont'd)

As you can see, CSS specifies a priority scheme to determine which style rules apply if more than one rule matches against a particular element.

This also comes into play when using a combination of inline, internal, and external styles.

Specificity Wars



External CSS

Internal CSS

Inline CSS

LEAST specific

MOST specific

Importance

You can override default specificity by simply including **!important** in your style declarations.

Importance Example

```html
<!-- HTML content -->
<p>
    Hello there, <span>Mansoor!</span>
</p>
```

```css
/* CSS styles */
p span {
    color: blue;
}

...


span {
    color: red !important;
}
```

Inheritance

A child element will **inherit** any applicable styles from its parent HTML element, unless otherwise specified.

Inheritance Example

```html
<!-- HTML content -->
<p>
    Hello there, <span>Mansoor!</span>
</p>
```

```css
/* CSS styles */
p {
    font-size: 36px;
}

...


span {
    color: red;
}
```

CSS Resources

- Cascade and Inheritance (MDN)
- Specificity (MDN)

Lab Time

Open: Assignment/starter_code/

GHE Setup

GHE/Github Desktop Setup

- **Fork** the FEWDR-422 repo
- **Open** GH Desktop
- File → Clone Repository... → Enterprise tab (you may need to login to GHE)
- **Select** your forked repo
- **Choose** "Desktop" as your "Local Path"
- Click **Clone**

Before Each Class

- **Open** GH Desktop
- Branch → Merge into Current Branch → upstream/master
- **Click** "Merge into master"
- **Click** "Pull"

Directions:

1. Get a jump start on this week's homework
2. Open the **Week_1_Basics/Assignment/starter_code** folder
3. Begin developing the **About Me html page** based on the assignment rubric

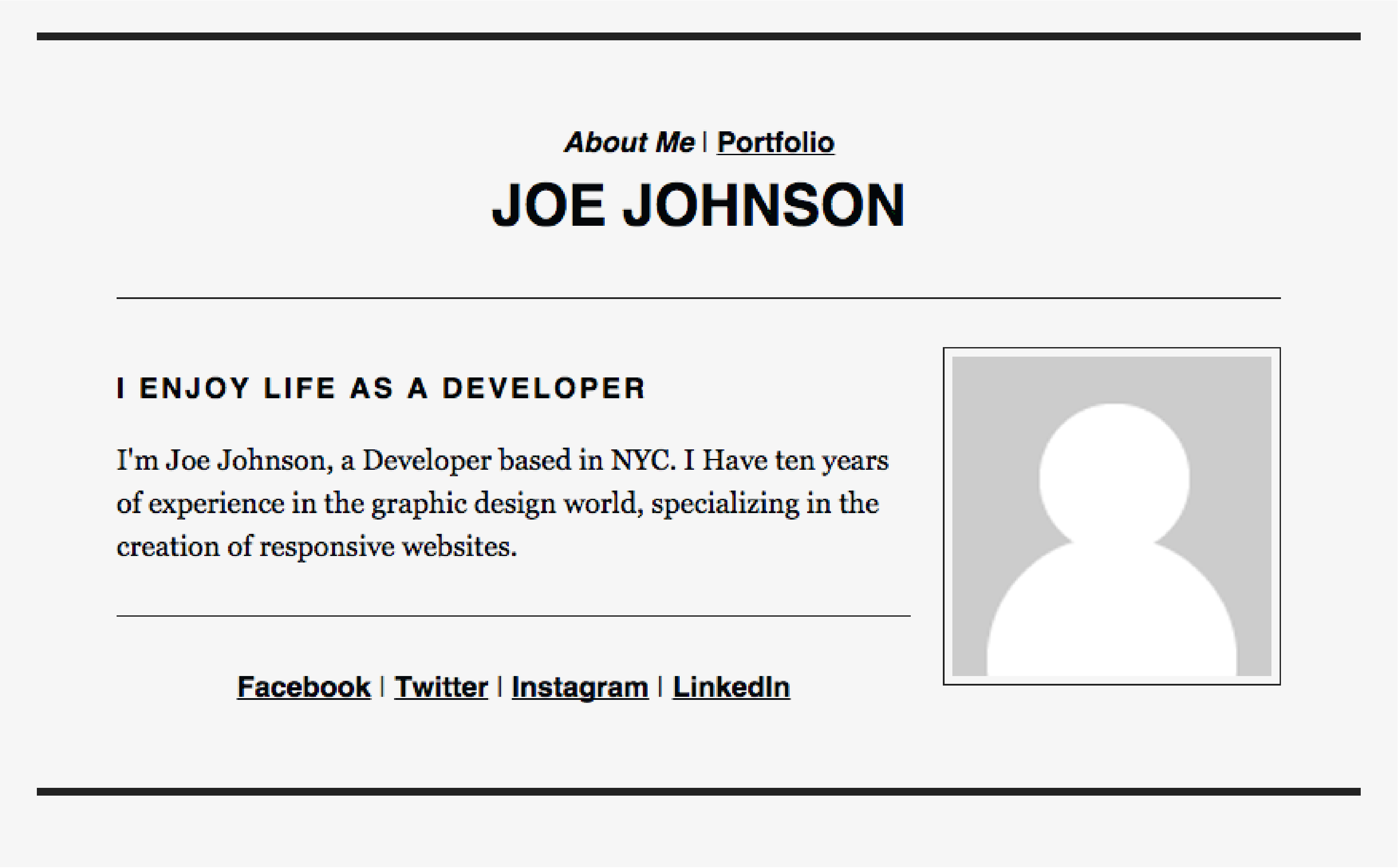
Timing:

- **5 minutes** - Get you started as a class
- **25 minutes** - finish building on your own

Bonus:

- I will help you get started
- Make it your own! Customize it as much as you want

About Me Page



About Me | Portfolio

JOE JOHNSON

I ENJOY LIFE AS A DEVELOPER

I'm Joe Johnson, a Developer based in NYC. I Have ten years of experience in the graphic design world, specializing in the creation of responsive websites.

Facebook | Twitter | Instagram | LinkedIn

Exit Tickets

Take 5-10 minutes to give us some feedback

Learning Objectives Review

- We described the DOM and drew a simple DOM tree
- We understand the difference between absolute & relative paths, and apply relative paths to <img> and <a> tags
- We applied and explained CSS "cascade" including: importance, specificity, and inheritance
- We linked to an external CSS stylesheet

FEWDR Homework

- All homework files will be included in the week's Assignment folder in GHE
- Look for this folder when you pull the remote class repo Monday before class
- Once completed, be sure to share the URL for the week's assignment with the TA via a PM in Slack
- The TA will grade all assignments and provide feedback directly to you via Slack
- **Communicate any issues!!**

Week 1 Homework

- Assignment: **About Me Page**
- We will submit this first assignment together at the start of class on MONDAY!! (you can still make changes and update until midnight)
- Grading rubric can be found in the **Assignment** folder
- **FOLLOW THE RUBRIC!**

Next Class...

Lesson 3 - CSS Box Model