GENERAL ASSEMBLY

HTML & CSS

DAY 1

Kim Goulbourne

LET'S ORGANIZE OUR FOLDERS!

1. Create a "HTML-CSS" folder anywhere on your laptop (this is where you'll do everything).
2. Inside that folder, create these folders:
   - Exercises
   - Assignments
   - Resources

AGENDA

- HTML basics
- Reading and understanding HTML structure
- CSS basics
- Understanding and using CSS properties
- External Style Sheets

WHAT IS HTML?
HYPERTEXT MARKUP LANGUAGE

WHAT DOES HTML STAND FOR?

Hypertext - at its most basic level, is text which contains links to other texts and other forms of media such as image or video.

Markup Language - a system for defining the presentation of text. HTML is a markup language for the web.

HOW DOES THE BROWSER UNDERSTAND HTML?

- HTML is the standard markup language used to create web pages on the world wide web.
- The language is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <html>).
- Browsers do not display the HTML tags and scripts, but use them to interpret the content of the page (which is particularly helpful for accessible devices - eg. what a visually impaired person would use).
- The latest version of HTML is HTML5.

WHAT IS CSS AND JS IN RELATION TO HTML?

- HTML - defines the structure of a webpage as well as the “type” of content that is on the page.
- CSS (Cascading Stylesheets) - define the look and feel of the webpage.
- JS (JavaScript) - affects to behavior of the webpage.

HTML SYNTAX

The “tag” defines the type of content

Example: <html>…</html>

IMPORTANT NOTES!

- Tags should be written in lowercase.
- All tags have an “opening” tag and “closing” tag.
  - For example: <html>…..</html>
- Some tags are “self-closing” because they don’t contain nested content.
  - versus <img></img>
  - Another self closing tag is input: <input type=“text” /> which is used for forms.
- Tags can have an attributes which provide additional information about the element. They are always specified in the opening tag and come in name/value pairs like: name=“value”.
- Most elements can be nested within another element.

QUICK RECAP!

WHAT IS HTML?

HTML TAGS

ONLY A FEW

HEADINGS

Heading tags are used to define the most important pieces of text on a page (the headings).

There are 6 available heading tags. h1 is the highest while h6 is the lowest in terms of usage hierarchy.

HEADING 1	...
HEADING 2	...
HEADING 3	...
HEADING 4	...
HEADING 5	...
HEADING 6	...

A NOTE ON HEADINGS AND THE <H1>

There can be multiple headings on a page but the <h1> should only be used once per page.

* This helps with SEO.

```
<h1>Page Title</h1>
<h2>Section Title 1</h2>
<h2>Section Title 2</h2>
<h3>Smaller Title</h2>
```

PARAGRAPH

The paragraph tag is used for longer form pieces of text (like in an article).

...

There is only one available tag for paragraphs.

LISTS

Lists define content that is connected in some way and are made up of 2 parts.

1. The container tag is used to define the type of list.
2. The list item tag is used for each item in the list.

OL are used for numbered lists where the order matters (1,2,3) while UL could be bullets, where the order doesn't matter.

UNORDERED LIST

ORDERED LIST

LIST ITEM …

USAGE:

```
<ul>
  <li>…</li>
</ul>
```

LINKS / ANCHOR TAG

The link/anchor tag is used to define linked content in a webpage.

HTML links are hyperlinks. A hyperlink is a text or an image you can click on, and jump to another document. The source of the link is set using the “href” attribute.

…

IMAGES

The image tag is used to insert images into the webpage.

```
<img src="image_url" />
```

The source of the image is set by the "src" attribute.

DIV

The tag is used to organize and separate elements on a page.

It is primarily used as a “container” tag similar to <ul> and tags. It contains other nested html elements.

```
<div>
  <h2>Title</h2>
  <p>Some text here</p>
</div>
```

HTML5 introduced more “semantic” container tags. These will be discussed in a later class:

- the <header> tag
- the <footer> tag
- the <nav> tag

SPAN

The tag is similar to a but used on a smaller scale. It can be used when no other tags make sense (it's not a paragraph, heading, link etc). It can also used to wrap content that needs to be targeted with CSS.

Eg. A word nested in a paragraph that needs to be styled with a different color.

```
<span>Credit Text</span>

<p>Hi, it's <span>me</span> Kim</p>
```

THE DIFFERENCE BETWEEN A SPAN AND A DIV

 is a naturally block level element - it sits on it's own line. Imagine a stack of boxes.
 is naturally an inline level element - it falls inline with the elements around like words in a paragraph.

* It's possible, using CSS, to set a div to be (inline) and a span to be (block) using the "display" property, which we will discuss later.

.HTML FILE

PUTTING IT ALL IN AN HTML DOCUMENT (.HTML FILE)

- All HTML documents must start with a type declaration.
  <!DOCTYPE html>
- The HTML document itself begins with the html tag (container tag) <html>…</html>
- Supporting files (such as CSS or JS) or information is set in the head tag (container tag) <head>…</head>.
- The (visible) part of the HTML document that is rendered is set in the body (container tag) <body>…</body>

```
<!DOCTYPE html>
<html>
  <head>…</head>
  <body>
      <h1>…</h1>
      <p>…</p>
  </body>
</html>
```

A QUICK NOTE ON INDEX.HTML

Every website must have a “index.html” file. This is typically the first file you will create for any project. It determines what the homepage of your site looks like.

* Unless I say otherwise, the first file you create for any project should be called *index.html*.

index.html

```
<!DOCTYPE html>
<html>
 <head>…</head>
 <body>
      <div>
        <h1>…</h1>
        <p>…</p>
      </div>
 </body>
</html>
```

LET'S PLAY

GA PRESS RELEASE: ADDING STRUCTURE

INSTRUCTIONS:

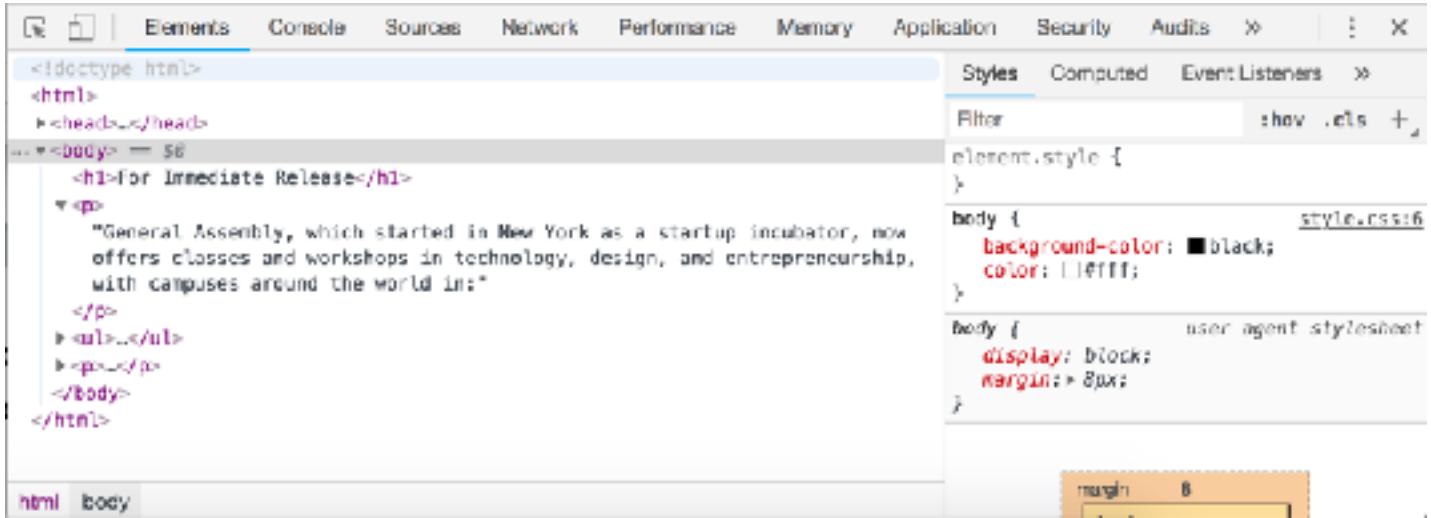
- Copy the GA Press Release folder to your desktop and place in the exercises folder we created.
- Let's play with HTML tags!

PLAYING IN THE BROWSER

USING DEV TOOLS TO PLAY

Your entire website can be viewed and manipulated using dev tools in the browser.

To view the dev tools in Chrome:
View > Developer > Developer Tools
OR Shortcut: Command + Alt + I



QUICK RECAP!

TAGS ON TAGS ON TAGS

WHAT IS CSS?
CASCADING STYLE SHEETS

WHAT DOES CSS STAND FOR?

Cascading - the format in which css rules are written.

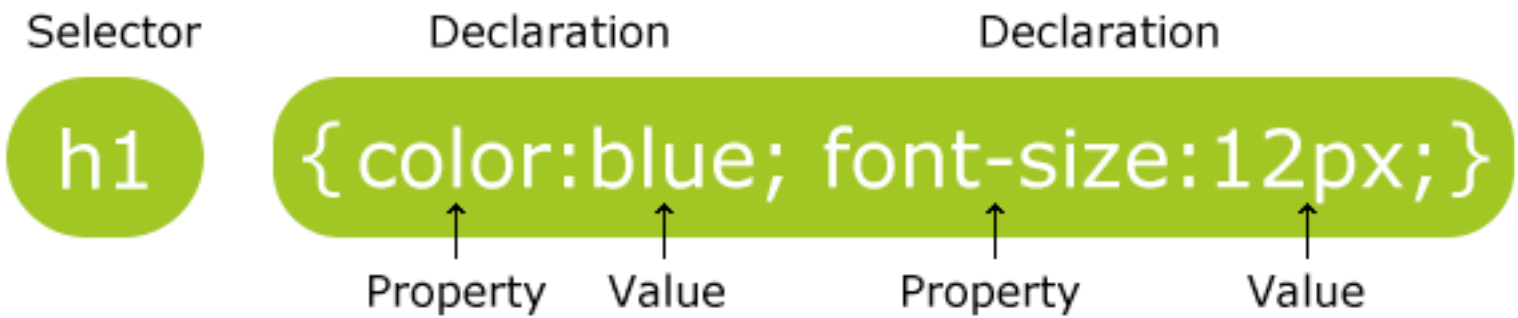
Style Sheets - describes the presentation of an HTML document.

HOW DOES THE BROWSER UNDERSTAND CSS?

- CSS is the stylesheet language used to style web pages on the world wide web.
- It describes how elements must be rendered on screen, on paper, or in other media (such as color or font size etc)
- Stylesheets may be written directly in an html document using a style tag or in an external css file but linked to the document.
- The latest version of CSS is CSS3.

CSS SYNTAX

(HTML tag)



Example:

```
h1 {
  color: blue;
  font-size: 12px;
}
```

IMPORTANT NOTES!

- Selectors (HTML tags) should be written in lowercase to match the html.
- Properties should also be written in lowercase.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

Example:

```
h3 {
  color: black;
  font-size: 12px;
}
```

CSS PROPERTIES

ONLY A FEW

FONT-FAMILY

Font family is used to set the main font that will be used to present a piece of text.

```
p {
   font-family: "Times New Roman", Times, serif;
}
```

It should only be used on tags that are used for text (like paragraphs).

The font-family property should hold several font names as a "fallback" system, to ensure compatibility between browsers/os. If the browser does not support the first font, it tries the next font.

http://www.w3schools.com/cssref/css_websafe_fonts.asp

FONT-SIZE

Font size is used to set the size of the font for a piece of text. It may be represented in different units - the most common is pixels.

```
p {
    font-size: 30px;
}
```

It should only be used on tags that are used for text (like paragraphs).

COLOR

Color is used to set the color of the font for a piece of text. It may be represented in different units - the most common is hex values.

using text

```
p {
   color: black;
}
```

using hex

```
p {
   color: #000000;
}
```

It should only be used on tags that are used for text (like paragraphs).

A hexadecimal(hex) color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color. All values must be between 00 and FF.

http://www.w3schools.com/tags/ref_colorpicker.asp

COLOR VALUES

Keywords - used less frequently but handy for basic colors (eg. black).
Hex codes - used more frequently and typically written with 6 digits (eg. #000000) but can be abbreviated with 3 for simple colors like(#000) or (#fff)
RGB - The first value is red, the second green, the third blue. Each value can range from 0 to 255. (Eg. rgb(0,0,0))
RGBA - Similar to RGB except it includes a 4th value called the "alpha" which determines the color's opacity. Value ranges from 0 and 1. (Eg. rgba(0,0,0,.5))

* HSL and HSLA are also values that can be used. Similar notation to RGB values, but specify colors using hue, saturation, and lightness. Read more: https://css-tricks.com/yay-for-hsla

EXAMPLE HEX VALUES

white = #fff or #ffffff

black = #000 or #000000

dark grey = #333 or #333333

red =#f00 or #ff0000

blue = #00f or #0000ff

yellow = #ff0 or #ffff00

BACKGROUND-COLOR

Background color is used to set the background color of any element on the page.

Like color, it may be represented in different units - the most common is hex values.

using text

```
body {
    background-color: black;
}
```

using hex

```
body {
    background-color: #000000;
}
```

* Both these declarations will set the background color of the entire document (body) to black.

BORDERS

Border is used to set the style of a border around any element on the page.

```
h1 {
   border: solid 1px #000;
}
```

SHORTHAND

*You can set just one side of the element by “border-top” vs just “border”.

```
h1 {
   border-color: #000;
   border-style: solid;
   border-width: 1px;
}
```

FULL

BORDERS

This will set just the top border.

```
h1 {
    border-top: solid 1px #000;
}
```

SHORTHAND

```
h1 {
    border-top-style: solid;
    border-top-width: 1px;
    border-top-color: #000;
}
```

FULL

MARGIN

Margin is used to set the space around (the outside) of an element.

```
h1 {
    margin: 10px;
}
```

ALL AROUND / SAME SIZE

```
h1 {
    margin-bottom: 10px;
}
```

JUST THE BOTTOM

PADDING

Padding is similar to margin except it is used to set the space around (the inside) of an element. It is also included in the final width or height of an element.

```
div {
   padding: 10px;
}
```

ALL AROUND / SAME SIZE

```
div {
   padding-right: 10px;
}
```

JUST THE RIGHT

DIFFERENT WAYS TO WRITE MARGIN OR PADDING

```
h1 {
    margin: 10px 5px 20px 15px;
}
```

ALL AROUND / DIFFERENT SIZES

```
h1 {
    margin-top: 10px;
    margin-right: 5px;
    margin-bottom: 20px;
    margin-left: 15px;
}
```

ALL AROUND / DIFFERENT SIZES

TAGS, CLASSES & IDS AS IT RELATES TO CSS

TAGS

These are pre-defined content types that help define the structure of an HTML document.

<h2>...</h2>

USING TAGS IN HTML

* A tag defines the type of html element
* An HTML element is the full tag and all its parts (attributes).

```
h2 {
  color:#000;
}
```

USING TAGS IN CSS

CLASSES (THE DOT .)

Classes make it possible to directly target individual html elements through css or js. It is defined as an attribute on an html element.

```
<h2 class="section-title">…</h2>
<h2 class="section-title subtitle">…</h2>
```

ADDING CLASSES IN HTML

* The same class name can be used on multiple elements.
* An element can have multiple class names (separated by a space)

```
.section-title {
  color:#000;
}
```

USING CLASSES IN CSS

```
h2.section-title {
  color:#000;
}
```

CSS ALTERNATIVE

IDS (THE HASHTAG #)

IDs make it possible to directly target individual html elements through css or js. It is defined as an attribute on an html element.

```
<h2 id="page-title">...</h2>
```

ADDING AN ID IN HTML

* Multiple elements can have IDs but an ID should only be used once per page.

```
#page-title {
  color:#000;
}
```

USING ID IN CSS

```
h2#page-title {
  color:#000;
}
```

CSS ALTERNATIVE

WHEN SHOULD I USE A CLASS OR ID?

- Classes are typically used when you want to reuse styles on multiple elements throughout the page/site.
- A given element can have more than one class associated with it, while an element can only have one id.
- Classes are also often used to define behavioral styles in addition to visual ones.
- Use an ID when you have a single element on the page that will take the style. Remember that IDs must be unique (like a header or footer). While a class may also be used for this purpose, it also helps with targeting in JavaScript (so you can directly target 1 element).

STYLE TAGS VS EXTERNAL STYLESHEETS

INLINE STYLE

These styles are written via the style attribute on an html element.

```
<h1 style="color: black;">Title</h1>
```

index.html

```
<!DOCTYPE html>
<html>
 <head></head>
 <body>
   <h1 style="color: black;">Title</h1>
 </body>
</html>
```

THE STYLE TAG

It is written like any other html tag except it exists in the head of the document.

```
<style>
    h1 {
      color: #000000;
    }
</style>
```

index.html

```
<!DOCTYPE html>
<html>
 <head>
  <style>
    h1 {
      color: #000000;
    }
  </style>
 </head>
 <body>
   <h1>Title</h1>
 </body>
</html>
```

EXTERNAL STYLESHEETS

- External stylesheets are used to keep your CSS separate from your HTML as a way of organizing your code.
- It is a file with extension (.css) that only contain css code (your declarations)
- It is linked to your HTML document by way of the link tag. Note the “rel” attribute which should always be included.

<link type="text/css" rel="stylesheet" href="*style.css*">

index.html

```
<!DOCTYPE html>
<html>
 <head>
   <link type="text/css" rel="stylesheet" href="style.css">
 </head>
 <body>
   <h1>Title</h1>
    <p>A paragraph of text.</p>
 </body>
</html>
```

style.css

```
h1 {
  color: red;
}

p {
   color: #000;
   font-size: 16px;
}
```

INHERITANCE, SPECIFICITY & IMPORTANCE

INHERITANCE

HTML elements will derive their initial styles from their parent.

```
body {
  color: black;
  font-family: Helvetica, Arial, sans-serif;
}
```

* All text elements (headings, paragraphs) will inherit these styles.

SPECIFICITY

HTML elements can be directly styled by either using classes and ids or by targeting nesting elements using their parent.

```
<ul class="list">
  <li>Item</li>
</ul>
```

```
ul li {
  color: black;
}
```

```
.list li {
  color: blue;
}
```

* Don't get too carried away with nesting selectors. You should be max 4 deep.

LET'S PRACTICE: SPECIFICITY

HTML

```
<div>
  <p style="color: blue;">Hello</p>
</div>
<div id="hi">
  <p>Goodbye</p>
</div>
<div>
  <p>Goodbye</p>
</div>
```

CSS

```
#hi p { color: #000; }
p { color: red; }
div p { color: #fff; }
```

IMPORTANCE

Any styles on an html element can be overwritten, no matter what, by using the “!important” rule.

```
h2 {
  color: black;
}
```

```
h2 {
  color:blue !important;
}
```

* Always look for a way to use specificity before even considering !important

IMPORTANT NOTES!

- The order in which styles are written matters.
- When classes and ids are used, it takes precedence over the order in which styles are written but not the location.
- Don’t go over board on nesting selectors. Use classes and ids to directly target deeply nested elements in this case.
- Always look for a way to use specificity before even considering to use the “!important” rule

LET'S PRACTICE: CSS

HTML

```
<div class="awesome">
  <h1>Title</h1>
  <p>Excuse me <span>!!!</span></p>
  <ul>
    <li class="active">Number 1</li>
    <li><a href="http://google.com">Number 2</a></li>
    <li>Number 3</li>
</div>
```

LET'S PLAY

GA PRESS RELEASE: ADDING STYLE

QUICK RECAP!

STYLES ON STYLES ON STYLES

LUNCHTIME

REVIEW!

AGENDA

- HTML tags for SEO
- CSS Cascade Order
- More CSS properties and Comments
- Folders, Relative and Absolute Paths
- Lab Time

HTML TAGS FOR SEO

META

The <meta> tag provides metadata about the HTML document. Meta elements are typically used to define page description, keywords, author of the document etc.

* It is important for SEO (Search Engine Optimization) purposes.

<meta name="description" content="This is my portfolio" />
<meta name="keywords" content="portfolio, developer, designer" />

TITLE

The <title> tag is a type of meta tag.

- It defines the title of the website.
- Title in the browser toolbar (tab)
- Title in bookmarks or favorites.
- Title in search-engine results

<title>Kim's Portfolio</title>

* It is very important for SEO (Search Engine Optimization) purposes.

CSS CASCADE ORDER

DETERMINE WHICH STYLES ARE APPLIED

ORDER - BY LINE

Firstly, styles are read in the order they are written (line by line). In the example below, the <h2> will be white.

```
h2 { color: red; }
h3 { color:blue; }
p { color: black; }
h2 { color: white; }
```

ORDER - BY ELEMENT/SELECTOR

1. tag
2. .class
3. #id

* The above represents the order in which styles are read. IDs will always have the last say.

HTML

```
<h2 id="section-title" class="group-title">Title</h2>
```

CSS *The order plays a limited role when classes and ids are involved.

```
h2 { color: red; }
.group-title { color:blue; }
#section-title { color: black; }
```

```
#section-title { color: black; }
h2 { color: red; }
.group-title { color:blue; }
```

ORDER - BY LOCATION

1. *Browser default* - defined by the browser (eg. google works on chrome).
2. *External style sheet* - included via the link tag
3. *Internal style sheet* - included via the style tag
4. *Inline style* - included via the style attribute on an html element.

* Inline style has the highest priority. It will overwrite the rest.

LET'S PRACTICE: ORDER

HTML

```
<h2>Title</h2>
<p>A cool paragraph.</p>
<h2 class="group-title">Title</h2>
```

CSS

```
.group-title { color:blue; }
h2 { color: red; }
```

HTML

```
<div id="hi"><h2>Hello</h2></div>
<div>
  <h2 style="color:blue;">Goodbye</h2>
</div>
```

CSS

```
#hi { background-color: #000; }
div { background-color: yellow;}
```

MORE CSS FUN

DISPLAY

The display property specifies the display behavior of an HTML element. Most common values are *block*, *inline*, *inline-block*, *flex*.

It can used on any html element.

```
span {
   display: block;
}
```

Note: Every element on a web page is a rectangular box. The CSS display property determines how that rectangular box behaves.

Check out all the available values: http://www.w3schools.com/cssref/pr_class_display.asp

A FEW ELEMENTS THAT ARE “DISPLAY: BLOCK” / “BLOCK LEVEL” BY DEFAULT

- Headings
- Paragraphs
- Lists

A FEW ELEMENTS THAT ARE NOT BLOCK LEVEL BY DEFAULT

- Spans
- Links
- Images

WIDTH

Width is used to set the width or horizontal size of an html element. It may be represented in different units - the most common are pixels (fixed) and percentages (fluid/flexible - used in responsive layouts).

It can used on any “block” level html element like a div.

```
div {
   width: 600px;
}
div {
   width: 80%;
}
```

HEIGHT

Height is used to set the height or vertical size of an html element. It may be represented in different units - the most common are pixels (fixed) and percentages (fluid/flexible - used in responsive layouts).

It can used on any “block” level html element.

```
div {
   height: 600px;
}
div {
   height: 90%;
}
```

BACKGROUND-IMAGE

Background image is used to set the background image of an html element.

```
div {
  background-image: url(“photo.jpg”);
  background-repeat: no-repeat;
  background-position: center center;
}
```

* Depending on the size of the image it will cover the space the element has, no more.
* It can used on any html element but it’s best practice not to use it on text based elements.
* background-repeat defaults to “repeat”
* background-position can be: X-axis (center/left/right) and Y-axis (center/top/bottom) or a pixel/percentage value.
* Also check out “background-size” (CSS3)

FONT-WEIGHT

Font weight is used to set the weight of the font on a text element. Common values are *normal*, *bold*, *bolder*, *lighter* and *(100* - light *, 200, 300, 400, 500, 600, 700, 900* - bold*)* along with *initial* (default) and *inherit* (from parent).

```
p {
  font-weight: bold;
}
```

* The or <b> tag can also be used to achieve the same effect as bold.

FONT-STYLE

Font style is used to set the style of the font on a text element. Available values are *normal, italic or oblique* along with *initial* (default) and *inherit* (from parent).

```
p {
  font-style:italic;
}
```

* The or <i> tag can also be used to achieve the same effect as italic.

FONT FACE: WHAT IS IT?

Before it was only possible to use “web safe” fonts on your website. Font face makes it possible to introduce more custom fonts in your webpage.

* You must have at least the 2 web font files to support multiple browsers. (.woff, .woff2, .ttf, .svg, .eot,)

FONT FACE: HOW TO USE IT

```
@font-face {
  font-family: 'MyWebFont';
  src: url('fonts/webfont.eot'); /* IE9 Compat Modes */
  src:  url('fonts/webfont.woff2') format('woff2'), /* Super Modern Browsers */
        url('fonts/webfont.woff') format('woff'), /* Pretty Modern Browsers */
        url('fonts/webfont.eot?#iefix') format('embedded-opentype'), /* IE6-IE8 */
        url('fonts/webfont.ttf')  format('truetype'), /* Safari, Android, iOS */
        url('fonts/webfont.svg#svgFontName') format('svg'); /* Legacy iOS */
}

h1 {  font-family: "MyWebFont", Times, serif; }
```

Folder structure:

```
index.html
fonts/
    webfoot.eot
    webfont.svg
    webfont.ttf
    webfont.woff
    webfont.woff2
```

CUSTOM FONTS FROM OTHER SERVICES

Custom fonts can also be used via these online services. They host the requested font and you must use their syntax to make it work in your code. It's usually as easy as adding a line of code in the <head> and simply referring to the font by name and using the repeated "font-weight".

- Google Fonts
- Typekit
- Hoefler&Co Cloud Typography

COMMENTS

COMMENTS: WHERE AND WHEN

Comments are used to describe or explain a piece of code. It can act as a reminder while also providing more information for any other dev that may be working on the project. They are not rendered on a web page.

SYNTAX

HTML: <!-- comment here —> (can be wrapped around lines of code)

CSS: /* comment here */ (can be wrapped around lines of code)

JS: /* comment here */ (can be wrapped around lines of code)

or: // comment here (for inline comments)

LET'S TALK ABOUT FOLDERS

TYPICAL FOLDER STRUCTURE FOR A PROJECT WITH 1 PAGE

```
repo-name/
    index.html
    style.css
```

This is the most basic setup for a project and what you've learned so far. But most projects will need much more.

Example for accessing the home page.

http://kim.com

http://kim.com/index.html

TYPICAL FOLDER STRUCTURE FOR A PROJECT WITH 1 PAGE

```
repo-name/
    css/
        style.css
    fonts/
        fontname.woff
        fontname.woff2
    images/
        image.jpg
    index.html
```

Example for accessing the home page.

http://kim.com

http://kim.com/index.html

TYPICAL FOLDER STRUCTURE FOR A PROJECT WITH 2 PAGES OPTION 1

Example for accessing the home page.

http://kim.com

http://kim.com/index.html

Example for accessing the about page.

http://kim.com/about.html

```
repo-name/
    about.html
    css/
        style.css
    fonts/
        fontname.woff
        fontname.woff2
    images/
        image.jpg
    index.html
```

TYPICAL FOLDER STRUCTURE FOR A PROJECT WITH 2 PAGES OPTION 2

Example for accessing the home page.

http://kim.com

http://kim.com/index.html

Example for accessing the about page.

http://kim.com/about

http://kim.com/about/index.html

```
repo-name/
    about/
        index.html
    css/
        style.css
    fonts/
        fontname.woff
        fontname.woff2
    images/
        image.jpg
    index.html
```

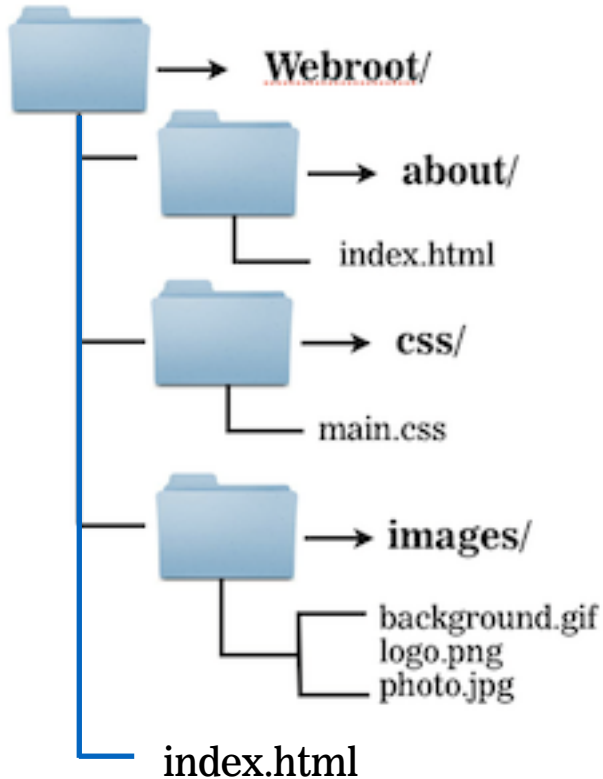
RELATIVE VS ABSOLUTE

FILE PATHS & LINKS

WORKING DIRECTORY

The working directory represents the current folder you are accessing.

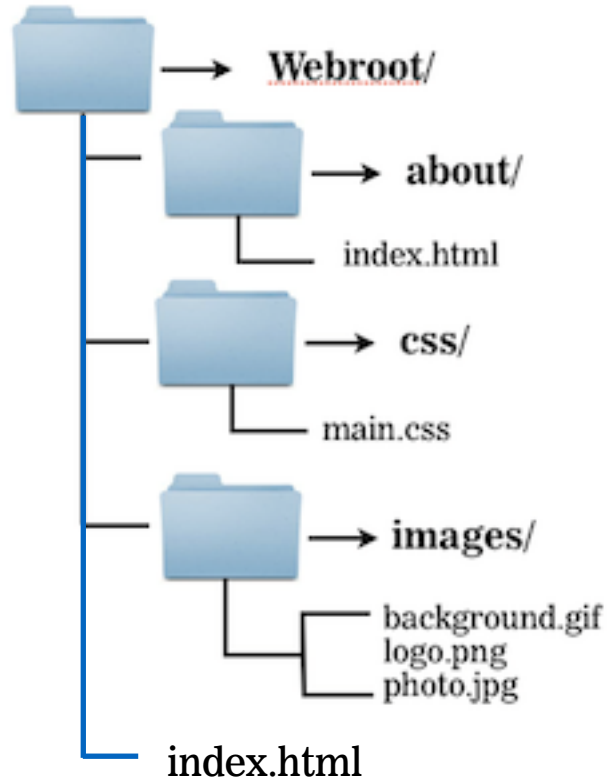
- If you visit the homepage, the “working directory” is the root (webroot).
- If you’re working in the about/index.html file or visiting the about page, the “working directory” is the about folder.
- If you’re working in the main.css file, the working directory is the CSS folder.



RELATIVE PATHS

Relative paths *do not* include the full website address (“http://www…”). It looks for the location of the file relative to the current page on which it appears (working directory).

```
<img src=“images/photo.jpg” />
<img src=“../images/photo.jpg” />
<a href=“about/index.html”>Link Title</a>
<link type=“text/css” href=“css/main.css” />
```



ABSOLUTE PATHS

Absolute paths typically include the full website address (includes “http://www…”) or start with a forward slash. It looks for the location of the file starting at the root of site’s directory so it ignores the current working directory.

```
<img src=“http://recipes.com/cookies.jpg” />
<img src=“/img/cookies.jpg” />

<a href=“http://recipes.com/about.html”>Link Title</a>
<a href=“/help/articles/how-to.html”>Link Title</a>
```

IMPORTANT NOTES!

- The forward slash (/) tells the browser to start by looking at the *root* of the site's directory.
  - on your local machine, it will start looking from your home directory (which will never work unless you specify the full ..crazy.. path)
  - on a local/remote server, it will start looking from your set "root" directory.
- If working without a "local server", never start paths with just a forward slash.
- The two dots + forward slash (../) tells the browser to go up/back a folder before looking for the file.
- Think about the "working directory" when writing your file paths in HTML, CSS and JS.

LAB TIME

CHOCOLATE COOKIE RECIPE

INSTRUCTIONS:

- Copy the cookie recipe folder to your desktop and place in the exercises folder we created.
- Add the appropriate HTML to the content in index.html as noted in the “readme.md” file.
- Using the “Cookie Recipe CSS guidelines, add the stated css properties in an external stylesheet.
- Add the “cookies.jpg” image after the first heading.

PERSONAL WEBSITE

CREATE A PORTFOLIO WEBSITE

REQUIREMENTS:

- Link both pages(about and portfolio) together using a tag
- Use a unique title (meta) and a single unique h1 tag
- Add images using img tags and relative paths (include unique alt attributes)
- Apply at least 5 different css properties to these elements
- Use at least 6 different HTML tags on your pages
- Commit at least 3 times

* BONUS: Experiment by adding additional HTML tags and CSS properties we did not cover in class and use classes and ids to target elements.

SUBMITTING THE HOMEWORK:

- Make sure you push the last changes via the GitHub app.
- Grab a link to your repo by visiting the repo on github.com under your profile. Click on the repo name and then grab the url in the address bar.
- Use this link to submit the assignment: https://docs.google.com/forms/d/e/1FAIpQLSfHooWfeWsNwIyRjFhTfr1oAlEZ3stRIahaNB7q8NEDh0eKTw/viewform

* Note: Make sure your repo isn't private. To make it public, visit "settings" under your repo on github.com and change it to public.

RESOURCES

A FEW LINKS

HTML

- http://www.w3schools.com/html/default.asp
- https://developer.mozilla.org/en-US/docs/Web/HTML/Element
- https://css-tricks.com/dom
- http://www.coffeecup.com/help/articles/absolute-vs-relative-pathslinks/

CSS

- http://www.w3schools.com/css/default.asp
- https://css-tricks.com
- http://monc.se/kitchen/38/cascading-order-and-inheritance-in-css
- http://www.w3schools.com/html/html_blocks.asp
- http://www.w3schools.com/cssref/pr_class_display.asp

* https://css-tricks.com/yay-for-hsla