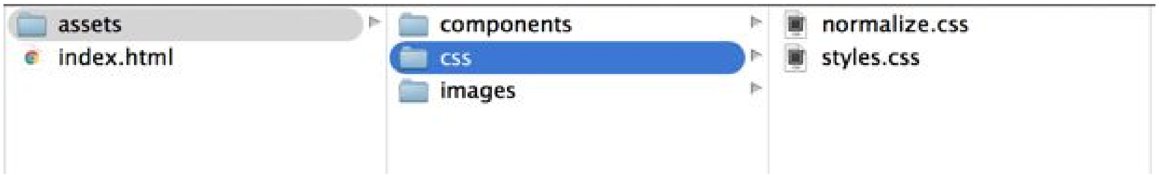
I399: Web Development

Knowledge Check #3

BASIC WEBSITE FILE STRUCTURE



assets
index.html

components
css
images

normalize.css
styles.css

NOTE: You might also have a fonts folder if you use a service like fontsquirrel. Other common folders might contain documents, PDFs, or other kinds of assets for your project. Name accordingly.

ESSENTIAL HTML STRUCTURE

```
<!DOCTYPE html>
<!--
        Documentation Comment (name, date, revisions, purpose)
-->
<html lang="en">
        <head>
                <meta charset="utf-8">
                <meta name="viewport" content="width=device-width,
        initial-scale=1, shrink-to-fit=no">
                <meta http-equiv="x-ua-compatible" content="ie=edge">
                <title></title>

                <!-- Stylesheets -->
                <link rel="stylesheet" type="text/css"
        href="assets/css/normalize.css">
                <link href="assets/css/styles.css" rel="stylesheet"
type="text/css" media="all">

                <!--[if lt IE 9]>
                <script
src="assets/components/html5shiv/html5shiv.min.js"></script>
```

```
        <![endif]-->
    </head>
    <body>

    </body>
</html>
```

- ***What purpose do these lines (in bold) serve? If you're not sure, can you guess?*** *(We'll discuss these again when we get to RWD)*

IMAGES

Vector Images

- Drawn by image programs using coordinates
- Infinitely re-sizeable
- Usually best for logos and illustrations
- Created in programs like Adobe Illustrator, or exported from code
- Formats used online: SVG

Raster Images

- Each image is made of pixels, each pixel is a color
- Most common, used for photographs (usually JPG)
- Created in programs like Adobe Photoshop or from a camera
- Formats used online: GIF, JPG, PNG
- PNG and GIF support transparency (alpha channel)

Alpha Channel

- Scale of 0 to 1, where 0 is completely transparent and 1 is opaque.
- GIF allows for transparency, but the quality is best for logos and items with limited colors
- PNG allows for a true alpha channel, but the final images are larger than the other formats

<u>To set opacity of an element and any elements inside of it</u>:

```
opacity: 0.6;
/* Element and all elements inside it set to 60% opacity */
```

<u>To set a color with a transparency use rgba()</u>:

```
background-color: rgba(0,0,0,0.6);
```

```
/* Element set to 60% opacity, elements inside remain 100% */
```

COLOR

Color can be defined in CSS in three ways:

1. **By name.**
   color: SlateBlue;

1. **By hexcode.**
   color: #00008B;
   Hexcodes are probably the most common method.

1. **By RGB or RGBA.**
   color: rgb(0, 0, 139);
   color: rgba(0, 0, 139, 1.0);

Hexcodes

#	**00**	**00**	**8B**
	Red	Green	Blue

- Hexcodes are made up of 3-PAIRS of numbers representing red, green then blue, in that order.
- They always start with a # (hashtag).

- The smallest number is 00 (or rgb: 0)
- The largest number is FF (or rgb: 255)

- #FFFFFF = white
- #000000 = black

- If the pairs are double digits, then you can shorten the code:
  #CCDDEE = #CDE
- If the pairs are all the same number, then you have a shade of grey:
  #8D8D8D = a dark grey

HTML ELEMENTS

Styling an entire element? Use a class:

In the HTML:

```
<nav class=”top-bar”>
        ...
</nav>
```

In the CSS:

```
nav.top-bar {   }
```

or

```
.top-bar {   }
```

Styling *PART* of an element? Use a span:

In the HTML:

```
<p>Salt <span class=”graphic-text”>and</span> Pepper</p>
```

In the CSS:

```
.graphic-text {   }
```

THE BOX MODEL

HTML elements are boxes. Any aspect of that box can be styled.
Example (base font size for websites is usually 16px):

In the HTML:

```
<h1>Mango Smoothie</h1>
```

In the CSS:

```
h1 {
        margin: 35px;
        padding: 2.0em;
        background-color: orange;
        text-align: center;
        border-bottom: 3px white dotted;
}
```

This creates an orange box that sits 35px down from where it would normally be positioned. The text is two baseline heights (32px) from the edge on all sides. The content is “Mango Smoothie” and it is centered in the box. A white, dotted line runs along the bottom of the box.

The sides of any HTML element (box) can be styled individually or together:

Styles all four sides at once.

```
padding: 1.0em;
```

Styles all four sides at once with different values.

```
padding: 1.0em 2.0em 1.0em 3.0em;
```

The order when using a shortcut dealing with the box model is always TOP, RIGHT, BOTTOM, LEFT.

Styles each side individually.

```
padding-top: 1.0em;
padding-right: 1.0em;
padding-bottom: 1.0em;
padding-left: 1.0em;
```

Styles the top/bottom and right/left.

```
padding: 1.0em 1.0em;
```

Styles the top, right/left then bottom.

```
padding: 1.0em 2.0em 1.0em;
```

CSS SELECTORS

CSS related to typography

Styles can be used on any HTML element containing text,
such as h1 - h6, p, li, td.

font-family
font-size
font-weight
font-style
color
text-transform
letter-spacing
word-spacing
line-height
text-align

Font shorthand:

Any of these options can be left out, but you wouldn't set line-height without setting font-size.

font: *font-weight font-style font-size / line-height font-family;*

example:
font: bold italic 12px/14px Arial, Helvetica, sans-serif;

Font stacks:

font-family: Georgia, 'Times New Roman', serif;
font-family: Helvetica, Arial, sans-serif;

- Always provide a **font stack**, not just one font choice.
- The stack should go from first choice, to back-up choice, possibly a second back-up choice, and finally a generic choice.
- Five generic options:
  serif, sans-serif, monospace, cursive, fantasy
- If the font name has more than one word, it MUST be placed in single quotes. *This is optional for single word fonts, but the quotes can be helpful. In TextWrangler, they will turn the font red, which may make your code easier for you to parse.*

How to change the background:

By setting each aspect individually:

```
body {
        background-color: #7d110c;        /* IU red */
        background-image: url(‘../images/iu-logo.png’);
        background-repeat: no-repeat;
        background-attachment: fixed;
        background-position: right top;
}
```

Using the background shortcut:

```
body {
        background: #7d110c url('../images/iu-logo.png') no-repeat
        fixed right top;
}
```

When using the shorthand property the order of the property values is:

1. background-color
2. background-image
3. background-repeat
4. background-attachment
5. background-position

- You can set the background on any block level HTML element.

- Always set a background color when you set a background image just in case the image doesn’t show up.

- DEBUGGING: The number one reason a background image doesn’t appear is that the pathname is incorrect.

- Use ../ (dot-dot-slash) before “images” to tell the browser to go up a level, then look for the images folder.

- Background-images can be layered, as in you can display more than one at a time in any given element. For example a textured background and a logo overlay that are both in the body.

- The options for background-repeat are: repeat (default), no-repeat, repeat-x, repeat-y -- IF YOU REPEAT, try to make your image “seamless”

- When **background-attachment** “fixed” is set, the content will appear to scroll on top of the image.

- Sometimes you want the image to appear in a certain spot in the browser and **background-position** will likely get you that effect.

- See our ***Code Snippets Page*** or search for ***Perfect fullscreen background*** by Chris Coyier at CSS Tricks to see how that effect can be done with CSS.

WEB FONTS

Measurement units
When styling typography, you must set what unit you’re using for size.
http://css-tricks.com/rems-ems/

- pt – points – sometimes seen, we won’t use this

- px – pixels – most useful to set base font sizes,
  and when working in *fixed width* sites

- em – ems – 1.0em = 1 line height, 1.2 or 1.3em = perfect for body type
  fonts stay in proportion to a base size when changed

- rem – point – ems relative to the root, good for components

- % – percent – percent difference based on root or default base size,
  usually better left for components, but depends.

Using Google Web Fonts

1. Load the google web font into your website.

In the HTML, in the <head> underneath the link to your normalize.css:

<link href='http://fonts.googleapis.com/css?family=Cinzel+Decorative' rel='stylesheet' type='text/css'>

OR

At the top of your CSS (okay under the documentation comment):

```
@import url(http://fonts.googleapis.com/css?family=Cinzel+Decorative);
```

2. Use the google web font to style your web pages.

In your CSS:
(okay)

```
font-family: 'Cinzel Decorative', cursive;
```

(better – provide a backup or two just in case)

```
font-family: 'Cinzel Decorative', ‘Times New Roman’, cursive;
```

Using @font-face

In your CSS towards the start of your CSS file:

```
@font-face {
        font-family: Chalkduster;
        src: url(‘../fonts/Chalkduster.otf’);
        /* best to have as many formats as possible listed here */
}
```

In your CSS wherever you need it:

```
font-family: ‘Chalkduster’, ‘Helvetica’, sans-serif;
```

- Google Webfonts WILL NOT APPEAR if you are looking at local files and are not connected to the Internet.
- @font-face will only work in all situations if you have *multiple files types* available for the call to reference.
- Be wise in your selection of special fonts. They add to a page’s load time dramatically and often they are meant for display purposes. *Be careful not to overdo the fancy.*

GOOD CODING PRACTICES

YES - *This practice is only true for zero. In general, units are good.*

```
margin: 0;
```

NO

```
margin: 0px;
```

YES

```
padding: 50px;
```

NO

```
padding: 50 px;
```

YES

```
background-color: black; /* set me in case something happens */
background-image: url(‘images/bkg.jpg’);
```

NO

```
background-image: url(‘images/bkg.jpg’);
```

GOOD DESIGN PRACTICES

#10: When reasonable, **style elements semantically** or by where they are located, rather than by adding in unnecessary classes.

```
.byline {
	font-weight: bold;
}

ul li {
	font-size: 18px;
	color: blue;
}


ol li {
	font-size: 24px;
	color: red;
}
```

#11: Use **text-align: justify;** sparingly. Flexible designs and justification can result in some pretty ugly typography.

#12: Always use a **font stack**, even when you're using a font downloaded and included in your website project folder. This serves to give you some control over what the design will look like if the browser can't read or find the first typeface choice.

#13: Don't put text up against the edge of a box. Use **padding** instead.

#14: Place related content closer together and unrelated content farther apart for added clarity and white space. (***proximity***)

#15: When pairing fonts, don't choose two that are similar. Make sure your choice doesn't look like an accident. The best pairings are from fonts in different groups, for example, a sans-serif and a serif. (***contrast***)