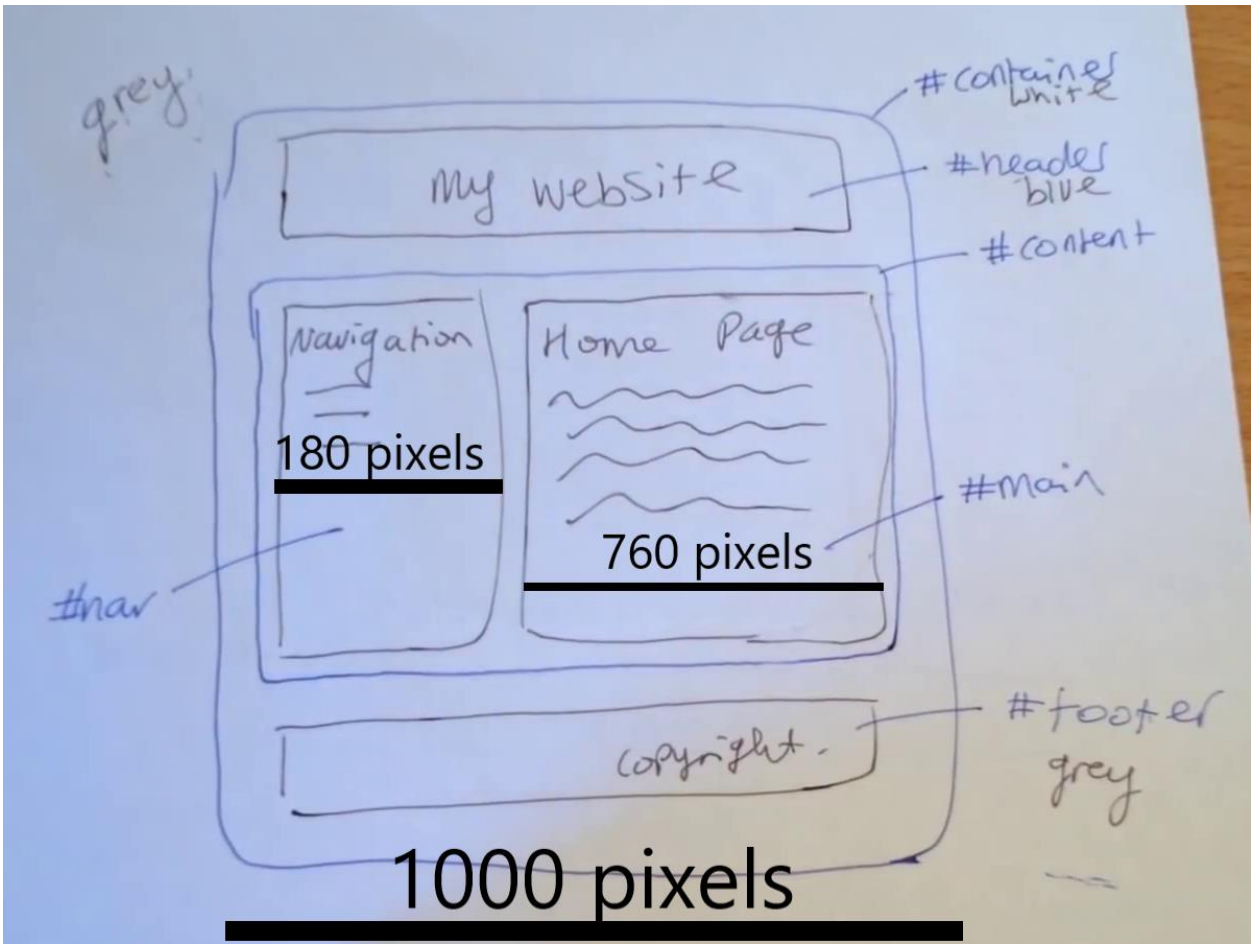
Itec/MODR 2635: html5 and CSS
Homework 2: Submission LINK.

Question 1: (100 points)
Design the following web page using an external CSS style (style.css):



To be displayed at your URL:

http://oldtown.glendon.yorku.ca/~lastname/css/2635/

Submit a short report with code and links to your web page.
This homework has only one question.

Colours, CSS (in HTML), Images

Colour

1. In HTML a colour can be specified as a name (Red, Blue, Green, Yellow and etc..), in RGB values, or in Hexadecimal

CSS

1. CSS stands for Cascading Style Sheets.
   a. CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**.
   b. CSS **saves a lot of work**. It can control the layout of multiple web pages all at once.
   c. CSS can be added to HTML elements in 3 ways:
      i. **Inline** - by using the style attribute in HTML elements
      ii. **Internal** - by using a <style> element in the <head> section
      iii. **External** - by using an external CSS file

   The most common way to add CSS, is to keep the styles in separate CSS files. However, here we will use inline and internal styling, because this is easier to demonstrate

2. Inline CSS is used to apply a unique style to a single HTML element. Inline CSS uses the *style attribute* of an HTML element.

```
<h1 style="color:blue;">This is a Blue Heading</h1>
```

3. Internal CSS is used to define a style for a single HTML page. An internal CSS is *defined in the <head> section* of an HTML page, by using a <style> tag:

```
<!DOCTYPE html>
<html>
<head>
<style>
body {background-color: powderblue;}
h1   {color: blue;}
p    {color: red;}
</style>
</head>
<body>

<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>

</body>
</html>
```

4. External CSS has an external style sheet that is used to define the style for many HTML pages. **With an external style sheet, you can change the look of an entire web site, by changing one file!** To use an external style sheet, add a link to it in the <head> section of the HTML page:

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

An external style sheet can be written in any text editor. The file must not contain any HTML code, and must be saved with a .css extension.

Here is how the "styles.css" looks:

```
body {
    background-color: powderblue;
}
h1 {
    color: blue;
}
p {
    color: red;
}
```

5. Common CSS Fonts contain the following information
    a. Colour
    b. Font-Family
    c. Font-Size

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
   color: blue;
   font-family: verdana;
   font-size: 300%;
}
p  {
   color: red;
   font-family: courier;
   font-size: 160%;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

6. CSS Border - The CSS **border** property defines a border around an HTML element.

```
p {
   border: 1px solid powderblue;
}
```

7. CSS Padding - The CSS **padding** property defines a padding (space) between the text and the border:

```
p {
   border: 1px solid powderblue;
   padding: 30px;
}
```

8. CSS Margin - The CSS **margin** property defines a margin (space) outside the border.

```
p {
   border: 1px solid powderblue;
```

```
    margin: 50px;
}
```

Note: You can also declare all the margins and paddings of an element in a single property as follows:

```
margin: 10px 10px 10px 10px;
```

If you declare all 4 values as above, the order is as follows (clockwise from the top):

1. top
2. right
3. bottom
4. left

If only one value is declared, it sets the margin on all sides.

```
margin: 10px;
```

If you only declare two or three values, the undeclared values are taken from the opposing side.

```
margin: 10px 10px; /* 2 values * Top and Sides/
margin: 10px 10px 10px; /* 3 values * Top, Right, Bottom/
```

You can set the margin property to negative values. If you do not declare the margin value of an element, the margin is 0 (zero).

```
margin: -10px;
```

All the coding we have done so far with CSS has ensured that all <p> will be altered via the CSS. However, what if we only wanted a few specific <p> tags to be manipulated via CSS and we wanted the others to stay the same? Here is where we use the id and class attributes.

9. The id Attribute - To define a specific style for one special element, add an id attribute to the element. (the # code)

   In the HTML we would type:

```
<p id="p01">I am different</p>
```

and in the CSS we would type:

```
#p01 {
   color: blue;
}
```

It is important to note that an id attribute goes hand in hand with the hashtag (#) code from CSS.

10. The class attribute - To define a style for a special type of elements, add a class attribute to the element:

In the HTML we would type:

```
<p class="error">I am different</p>
```

In the CSS we would type:

```
.error {
   color: red;
}
```

It is important to note that the class attribute goes hand in hand with the dot (.) code in CSS.

Images

1. The <img> tag is empty, it contains attributes only, and does not have a closing tag. The <img> tag needs a source (can be a URL or a file from your folder, an alternate attribute (alt) which states what the image is just in case the browser cannot find the image and a style attribute which defines the width and height of the image.

   <img src="*url*" alt="*some_text*" style="width:*width*;height:*height*;">

2. It is important to use the style attribute so that it the width and height of an image does not conflict with CSS coding. For example:
   http://www.w3schools.com/html/tryit.asp?filename=tryhtml_images_style

3. You can also use an image as a link.
   ```
   <a href="default.asp">
     <img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;border:0;">
   </a>
   ```

HTML 5

New HTML5 Elements
The most interesting new HTML5 elements are:

- New semantic elements like <header>, <footer>, <article>, and <section>.
- New attributes of form elements like number, date, time, calendar, and range.
- New graphic elements: <svg> and <canvas>.
- New multimedia elements: <audio> and <video>

New HTML5 API's (Application Programming Interfaces)
The most interesting new API's in HTML5 are:

- HTML Geolocation
- HTML Drag and Drop
- HTML Local Storage
- HTML Application Cache
- HTML Web Workers
- HTML SSE

Tip: HTML Local storage is a powerful replacement for cookies.

Removed Element	**Use Instead**
<acronym>	<abbr>
<applet>	<object>
<basefont>	CSS
<big>	CSS
	CSS
<dir>	
<font>	CSS
<frame>	
<frameset>	
<noframes>	
<strike>	CSS, <s>, or <del>
<tt>	CSS

HTML5 Browser Support
HTML5 is supported in all modern browsers.

- In addition, all browsers, old and new, automatically handle unrecognized elements as inline elements.
- Because of this, you can "teach" older browsers to handle "unknown" HTML elements.

Define Semantic Elements as Block Elements
HTML5 defines eight new semantic elements.

- All these are block-level elements.
- To secure correct behavior in older browsers, you can set the CSS display property for these HTML elements to block:

```
header, section, footer, aside, nav, main, article, figure {
  display: block;
}
```

New Elements in HTML5
Below is a list of the new HTML5 elements, and a description of what they are used for.

New Semantic/Structural Elements
HTML5 offers new elements for better document structure:

Tag	Description
<article>	Defines an article in a document
<aside>	Defines content aside from the page content
<bdi>	Isolates a part of text that might be formatted in a different direction from other text outside it
<details>	Defines additional details that the user can view or hide
<dialog>	Defines a dialog box or window
<figcaption>	Defines a caption for a <figure> element
<figure>	Defines self-contained content
<footer>	Defines a footer for a document or section
<header>	Defines a header for a document or section
<main>	Defines the main content of a document
<mark>	Defines marked/highlighted text
<meter>	Defines a scalar measurement within a known range (a gauge)
<nav>	Defines navigation links
<progress>	Represents the progress of a task
<rp>	Defines what to show in browsers that do not support ruby annotations
<rt>	Defines an explanation/pronunciation of characters (for East Asian typography)
<ruby>	Defines a ruby annotation (for East Asian typography)
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time
<wbr>	Defines a possible line-break

New Form Elements

Tag	Description

<datalist>	Specifies a list of pre-defined options for input controls
<output>	Defines the result of a calculation

New Input Types

New Input Types	New Input Attributes	
Color	autocomplete	placeholder
Date	autofocus	required
Datetime	form	step
Datetime-local	formaction	
Email	formenctype	
Month	formmethod	
Number	formnovalidate	
Range	formtarget	
Search	height and width	
Tel	list	
Time	min and max	
Url	multiple	
Week	pattern (regexp)	

HTML5 Graphics

Tag	Description
<canvas>	Draw graphics, on the fly, via scripting (usually JavaScript)
<svg>	Draw scalable vector graphics

New Media Elements

Tag	Description
<audio>	Defines sound content
<embed>	Defines a container for an external (non-HTML) application
<source>	Defines multiple media resources for media elements (<video> and <audio>)
<track>	Defines text tracks for media elements (<video> and <audio>)
<video>	Defines video or movie

HTML5 - New Attribute Syntax

HTML5 allows four different syntaxes for attributes.

This example demonstrates the different syntaxes used in an <input> tag:

Type	Example
Empty	<input type="text" value="John" disabled>
Unquoted	<input type="text" value=John>
Double-quoted	<input type="text" value="John Doe">
Single-quoted	<input type="text" value='John Doe'>

In HTML5, all four syntaxes may be used, depending on what is needed for the attribute.

HTML5 Semantic Elements
Semantics is the study of the meanings of words and phrases in a language.

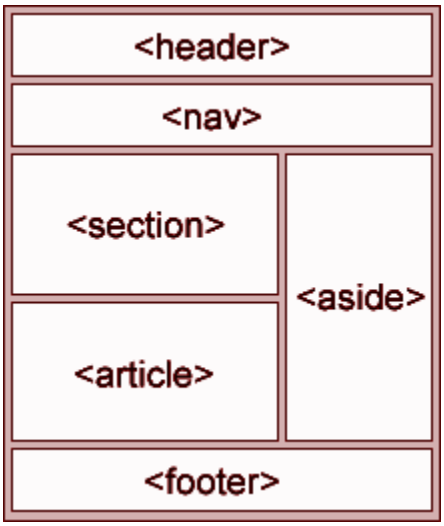
Semantic elements = elements with a meaning.

What are Semantic Elements?
A semantic element clearly describes its meaning to both the browser and the developer.

- Examples of non-semantic elements: and - Tells nothing about its content.
- Examples of semantic elements: <form>, <table>, and <article> - Clearly defines its content.

New Semantic Elements in HTML5
Many web sites contain HTML code like: to indicate navigation, header, and footer.

HTML5 offers new semantic elements to define different parts of a web page:



<article>
<aside>
<details>
<figcaption>
<figure>
<footer>
<header>
<main>
<mark>

<nav>
<section>
<summary>
<time>

HTML 4	**HTML 5**
	<header>
	<nav>
	<section>
	<article>
	<footer>

Use Lower Case Element Names
HTML5 allows mixing uppercase and lowercase letters in element names.

We recommend using lowercase element names because:

- Mixing uppercase and lowercase names is bad
- Developers normally use lowercase names (as in XHTML)
- Lowercase look cleaner
- Lowercase are easier to write

CSS: Introduction

The problem with HTML:

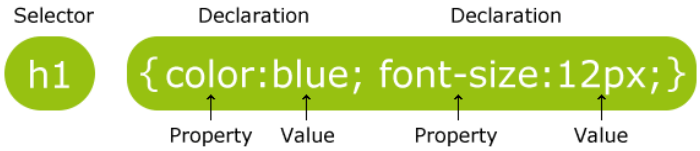
- HTML was originally intended to describe the content of a document
- Page authors didn't have to describe the layout--the browser would take care of that
- This is a good engineering approach, but it didn't satisfy advertisers and "artists"
- As a result, HTML acquired more and more tags to control appearance
    - Content and appearance became more intertwined
    - Different browsers displayed things differently, which is a real problem when appearance is important

Cascading Style Sheets

A Cascading Style Sheet (CSS) describes the appearance of an HTML page. It is usually a separate document included in the HTML but it can also be embedded inside the HTML.
Here is a demonstration of what can be done with CSS (click on the various stylesheet buttons to switch styles): https://www.w3schools.com/css/css_intro.asp
CSS has the following advantages:

- It lets you separate content from presentation
- It lets you define the appearance and layout of all the pages in your web site in a single place
- It can be used for both HTML and XML pages
- CSS has the following disadvantage:
- Most browsers don't support it very well

CSS syntax



CSS syntax is very simple--it's just a file containing a list of selectors (to choose tags) and descriptors (to tell what to do with them):
Example:
h1 {color: green; font-family: Verdana} says that everything included in h1 (HTML heading level 1) tags should be in the Verdana font and colored green

A CSS file is just a list of these selector/descriptor pairs. Selectors may be simple HTML tags or XML tags, but CSS also defines some ways to combine tags. Descriptors are defined in CSS itself, and there is quite a long list of them.
The general syntax is:

```
selector {property: value}
```

or

```
selector, ..., selector {
   property: value;
   . . .
   property: value
}
```

where *selector* is the tag to be affected (the selector is case-sensitive if and only if the document language is case-sensitive) *property* and *value* describe the appearance of that tag. Spaces after colons and semicolons are optional. A semicolon must be used *between* property:value pairs, but a semicolon after the last pair is optional.
Examples

```
/* This is a comment */
h1,h2,h3 {font-family: Arial, sans-serif;}   /* use 1st available font */
p, table, li, address {                /* apply to all these tags */
  font-family: "Courier New";      /* quote values containing spaces */
  margin-left: 15pt;                /* specify indentation */
}
p, li, th, td {font-size: 80%;}      /* 80% of size in containing element */
th {background-color:#FAEBD7}  /* colors can be specified in hex */
body { background-color: #ffffff;}
h1,h2,h3,hr {color:saddlebrown;} /* adds to what we said before */
a:link    {color:darkred}             /* an unvisited link */
a:visited {color:darkred}             /* a link that has been visited */
a:active  {color:red}                 /* a link now being visited */
a:hover   {color:red}                 /* when the mouse hovers over it */
```

CSS Units

Units appear in many CSS properties, for example font-size. It is always better to use percentages than absolute units such as pixels. For example a font-size of 200% will double the current size and will always be consistent with the display properties.

%	percentage
in	inch
cm	centimeter
mm	millimeter
em	1em is equal to the current font size. 2em means 2 times the size of the current font. E.g., if an element is displayed with a font of 12 pt, then '2em' is 24 pt. The 'em' is a very useful unit in CSS, since it can adapt automatically to the font that the reader uses
ex	one ex is the x-height of a font (x-height is usually about half the font-size)
pt	point (1 pt is the same as 1/72 inch)

pc	pica (1 pc is the same as 12 points)
px	pixels (a dot on the computer screen)

CSS Colors

Colors can be applied to many properties such as background-color. Color can be specified using names (“red”, “blue”) but for a full range of colors it is recommended to use HX or RGB values:

	Color HEX	Color RGB
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

CSS Selectors

In order to apply a style to an element, you need to defined a selector for that element. An HTML tag can be used as a simple element selector:

body { background-color: #ffffff }

You can use multiple selectors:

em, i {color: red}

You can repeat selectors:

h1, h2, h3 {font-family: Verdana; color: red}

h1, h3 {font-weight: bold; color: pink}

When values disagree, the last one overrides any earlier ones

The universal selector * applies to any and all elements:

* {color: blue}

When values disagree, more specific selectors override general ones (so em elements would still be red)

We have used the following CSS, in this order:

1. *em {color: red}*
2. * {color: blue}
3. **b {color: pink; color: black}**

A descendent selector chooses a tag with a specific ancestor:

p code { color: brown }

selects a code if it is somewhere inside a paragraph

A child selector > chooses a tag with a specific parent:

h3 > em { font-weight: bold }

selects an em only if its immediate parent is h3

An adjacent selector chooses an element that immediately follows another:

b + i { font-size: 8pt }

Example: I'm bold and I'm italic

Result will look something like: **I'm bold and** *I'm italic*

Examples

Selectors can apply to a type of element (example for any <p>), to a specific element (identified with an id), to a group of elements (identified by a class), or to more specific element defined by a combination of criterias. These are live examples of such selectors.

- The element selector
- The id selector
- The class selector (for all elements)
- The class selector (for only <p> elements)
- Grouping selectors

Attributes

It is possible to style HTML elements that have specific attributes or attribute values. A simple attribute selector allows you to choose elements that have a given attribute, regardless of its value:

Syntax: element[attribute] { ... }
Example: table[border] { ... }

An attribute value selector allows you to choose elements that have a given attribute with a given value:

Syntax: element[attribute="value"] { ... }
Example: table[border="0"] { ... }

The class attribute

Every HTML element can have a class attribute that defines one or many CSS classes. The class attribute allows you to have different styles for the same element:
In the style sheet:

```
p.important {font-size: 24pt; color: red}
p.fineprint {font-size: 8pt}
```

In the HTML:

```
<p class="important">The end is nigh!</p>
<p class="fineprint">Offer ends 1/1/97.</p>
```

To define a selector that applies to any element with that class, just omit the tag name (but keep the dot):

```
.fineprint {font-size: 8pt}
.label {background-color: blue}
```

Note that you can apply several classes to a single element:

```
<p class="important label">The end is nigh!</p>
```

The id attribute

The id attribute is used to identify a single element in the document: this means that the id value must be unique for each element in the document. So make sure to pick up different ids across your document. The id attribute is defined like the class attribute, but uses # instead of .
In the style sheet:

```
p#important {font-style: italic}     or
#important {font-style: italic}
```

In the HTML:

```
<p id="important">
```

class and id can both be used, and do not need to have different names:

<p class="important" id="important">

div and span

div and span are HTML elements whose only purpose is to hold CSS information. It's a bit like creating a block or node in the HTML tree structure. div ensures there is a line break before and after (so it's like a paragraph); span does not. Example:
CSS:

```
div {background-color: #66FFFF}
span.color {color: red}
```

HTML:

```
<div>This div is treated like a paragraph, but <span class="color">this span</span> is not.</div>
```

The box model

All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.
The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

Content - The content of the box, where text and images appear
Padding - Clears an area around the content. The padding is transparent
Border - A border that goes around the padding and content
Margin - Clears an area outside the border. The margin is transparent
Here is a CSS example to control these values:

```
div {
    width: 300px;
    border: 25px solid green;
    padding: 25px;
    margin: 25px;
}
```

What is the total width of the element?
Here is a live demo of the box model:

How to add CSS

There are three ways of inserting a style sheet in a HTML document:

1. External style sheet (example)
2. Internal style sheet (example)
3. Inline style (example)

External: This is the recommended technique since it is easy to share and modify independently from HTML documents. The HTML page includes a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section. For example:

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

The document style.css is edited as a regular text file and contains pure CSS, for example:

```
body {
    background-color: lightblue;
}
h1 {
    color: red;
    margin-top: 20px;
}
```

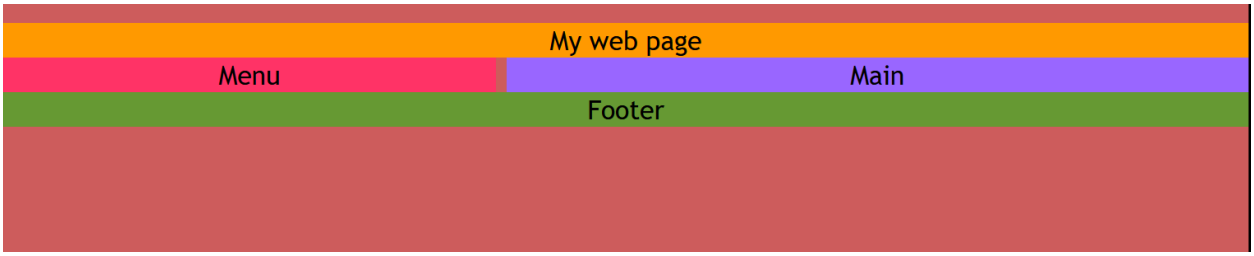
Internal: An internal style sheet is recommended if one single page has a unique style. Internal styles are defined within the <style> element, inside the <head> section of an HTML page, for example:

```
<head>
<style>
body {
    background-color: linen;
}
h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

Inline: inline style is used to apply a unique style for a targeted element. For example:

```
<h1 style="color:red;margin-left:20px;">Example of inline style</h1>
```

Example



References

1. HTML cheatsheet: http://bloggerspath.com/ultimate-html5-cheat-sheet-for-web-developers/
2. Chrome inspector: https://youtu.be/wcFnnxfA70g
3. CSS tutorial: https://www.w3schools.com/css/default.asp