TEENData Coding Learning Journal

7/1/2024
Started Codecademy "Learn HTML" course. Complete Lesson 1, which took 45 minutes. I learned the basic structure of HTML as well as common elements, such as body, paragraph, heading, divider, list, line breaks, font stylization, images, videos, and alternative text for visual media. The interactive activity allowed me to experiment with the elements and see the output immediately, which was helpful because I do not have experience with coding. Today's lesson was not very hard because I focused more on the function of each individual element, rather than combining them to form a comprehensive site.

7/2/2024
Completed the Codecademy HTML course Lesson 2, which took approximately 30 min. This activity focused on more complex code. Head tags and web page titles were more difficult than the elements that I learned yesterday because the effect is not visibly apparent. Hyperlinks were difficult because they require more complicated code to set up. I started to combine elements today, which was also challenging. For example, I made a hyperlink from an image to an external Wikipedia page. I also learned how to make the code easier to read and understand by using whitespace and indentation.

7/3/2024
Completed Lesson 3 in the Codecademy HTML course. It took about 20 min. I learned how to insert table rows, edit table data, create headings and footers, section off tables, and edit row and column spans. Rows and columns do not automatically populate, so it requires a long stream of code to create tables. I also learned how CSS can be used to style tables, including elements like font, size, and borders.

7/8/2024
Completed Lesson 4 of the HTML course. It took about 1 hour. I learned how to create different form elements, such as text inputs, number inputs, sliders, checkboxes, multiple choice buttons, dropdown lists, data lists, large text boxes, and submission buttons. I also learned how to code pre-filled text boxes and conceal sensitive information like passwords. This lesson was very difficult because it required me to keep track of a multiple of attributes that corresponded to different locations on and off the page, such as "value", "id", and "name".

7/10/2024
Completed Lesson 5 of the HTML course; it took about 45 minutes. I learned more complex coding for forms, such as validation for login credentials, required inputs, minimum/maximum values, and min/max character lengths. I also learned how to limit a user to certain types of

characters. For example, an input box can be set to only accommodate letters or it can exclude symbols. Setting a requisite character pattern for an input can be helpful for creating passwords; an input can be required to have a certain amount of letters, numbers, and symbols, respectively. This lesson was not as hard as the previous one because I already had experience working with the complex structure and only needed to learn the function of the new attributes. The second part of the lesson was semantic tags that replace confusing <div id=> tags, and instead consolidate divisions into specific tags, such as <header>, <nav>, <main>, <footer>, <section>, <article>, and <aside>. These tags split up sections of the code so that they are easier to interpret and easier to stylize in groups. This also makes a website more user-friendly, especially to those who have accessibility accommodations. I also learned how to create <figure> and <figcaption> tags to encapsulate media (photos, videos, audio, etc.) and group them with their captions. I also learned how to add audio and video controls for the user, such as play/pause, mute, time elapsed/remaining, and full screen. This lesson was very helpful because it showed me how to make the code condensed and easier to read.

7/13/2024
Today I started the Codecademy "Learn CSS" course, which took about 20 minutes. CSS stands for Cascading Style Sheets and is used to stylize HTML with factors like font, color, and positioning. This makes websites more aesthetically appealing. There are two CSS syntaxes--one is called ruleset and the other is called inline. The two methods have different ways of indicating which HTML element is being modified, but both contain a declaration consisting of a property-value pair that specifies the stylization. Inline syntax is integrated into HTML code. It is not commonly used except in a few situations. I also learned another way to write CSS code, which is an internal stylesheet. This can lead to long and confusing HTML pages, so external stylesheets, which separate HTML and CSS code, are a more commonly used method. External stylesheets use ruleset CSS syntax. I learned how to link a CSS file to the corresponding HTML file using the <link> tag so that the modifications can be applied. In this lesson, I learned how to use inline coding, internal stylesheets, and external stylesheets to modify the color of text. It was a bit confusing to learn that there are three different methods and two syntaxes to incorporate CSS into HTML. However, the individual methods are not hard to use.

7/16/2024
I completed half of Lesson 2 of the CSS course, which took about 45 minutes. I learned how to use a selector to specify what elements are to be stylized by the following code. The type selector precedes the declaration and specifies a tag that needs to be stylized according to the declaration--such as h1 or p. The universal selector applies a style to all elements and uses an asterisk (*). The class selector applies a style to elements that have the same class, such as "class=title" or "class=media". It uses a period before the class (ex: .title). Classes can be associated with elements of different tags. Multiple classes can be used with one element for ease

of combining styles. The id selector is used when one element has a unique styling need and using the # symbol. IDs can only correspond to one value, whereas class can correspond to multiple. The attribute selector applies a style to elements with a similar attribute--including href, src, class, and id. In fact, the attribute selector can be used instead of the class or id selectors. It uses the [] symbols. One can create very specific selectors by combining tag and attribute selectors. For example, a coder can apply a style to all images with a source that mentions “dog” using the selector img[src*=’dog’]. This lesson used a lot of HTML vocabulary terms that I haven’t quite grasped yet, so I had to review the differences and similarities between tags, elements, and attributes. The individual selectors were not hard to learn, but combining them was more difficult. It was interesting to see how different selectors can have the same function. I will work on discerning the most efficient way to stylize content as I become more familiar with CSS.

7/17/2024
Completed the second half of the CSS Lesson 2. It took about 1 hour. I learned about pseudo-class selectors, which have different styles according to how the element has been interacted with. For example, a link turning a different color after being clicked or text having a certain background when hovered over. I also learned about specificity, which is the order in which styles will be prioritized. The most specific selector, id, will override the most general selector, type. The best practice is to use the lowest degree of specificity possible because it makes it edit an elements style later on. I learned a CSS method called chaining, which combines selectors. This is similar to combining selectors from yesterday. Descendant selectors add another requirement for stylization. For example, in the combined selector .dog h2, the style will be applied to all h2 headings within elements with a class of “dog”. If the same style needed to be applied to multiple elements, but they are mutually exclusive, the two selectors can be separated by a comma

7/23
Completed Lesson 3 of the Codecademy CSS course, which took about 15 minutes. I learned how to style many different visual properties with CSS, such as font family, font size, font weight, text alignment, foreground color, background color, opacity, and background image. Not all font families are supported on every browser, but some are such as Arial, Calibri, Tahoma, and Georgia. Font size is measured in pixels. Font weight can either be bold or normal. Text alignment has four settings--left, center, right, and justify. Opacity is on a scale from 0-1, with 0 being completely opaque and 1 being fully visible. I also learned how to use the !important tag, which overrides all other styling commands pertaining to an element. This lesson was not difficult because I only had to learn the tags for each property and add a declaration to a CSS stylesheet.

8/21/2024
Started Lesson 4 of the CSS course. I learned about the box model for content, which includes the content, padding, border, and margin. I learned how to adjust the height and width of an element in pixels. However, this makes an element the same size at all times, which makes the site less accessible across all devices. I also learned how to adjust the width, style, and color of a border. For border curvature, border-radius can be used, either with pixels or percentages. This lesson was not hard because the properties had straightforward commands and I was able to see the result of stylization immediately.

8/22/2024
Continued Lesson 4 of the CSS course. I learned shorthands for specifying padding width on the top, bottom, right, and left of an element. The last component of the box model is margin, which specifies how much space is between an element's border and other elements. I learned the margin command, specific commands for the margins of each side of an element, and shorthands. I learned how to center content using "margin: 0 auto;", which sets the vertical margins to 0 and the horizontal margins to values that allow the content to be in the middle of the element. For this command to work, the containing element must have a width specified.

I also created the HTML file for the website. I used the TextEdit application on my MacBook and inserted a placeholder title for the webpage and a sample paragraph.

8/23/2024
Finished Lesson 4 of the CSS course. I learned about a concept called margin collapse. With right and left margins, the space between the borders of two elements is the sum of the margins. However, with top and bottom margins, the space is determined by the largest margin specification. I also learned how to set minimum and maximum heights and widths. Setting the min and max were not difficult, but I wasn't able to easily visualize the changes unless the content overflowed. Sometimes, if the total height and width of an element is too large for the containing element, the content will overflow. Using the overflow property allows a coder to determine whether content will be hidden, whether a scrollbar will be inserted, or whether the content will be visible (default setting). I learned how to reset the default margin and padding to 0. Developers do this to override a web browsers' default settings because those defaults often interfere with the desired design of a webpage. The visibility property hides an element from viewers, but keeps the space that it would occupy. At first, I was confused with the visibility property because I could not think of a scenario in which it would be useful. However, after some research I discovered that developers can use visibility: hidden if an element isn't applicable to a site yet, if user interaction is required to display the element, and if the developer would like to insert transitions.

11/18/2024
Completed Lesson 5 of CSS course. The box model is affected by changes to elements other than the content area, including borders and padding. Height and width are based on the content only. The border-box model is not affected by borders and padding. Height and width are determined by the entire border-box model.

I transferred my code to Visual Studio Code (VS Code) for formatting reasons and to streamline the coding experience.

6/10/2025
Set up outline of website and populated social media page with pertaining research. The website now includes six pages: *Home* (overview of project), *Bibliography* (annotated bibliography for the content of the website), *Code Journal* (snippets of code from VS Code and learning journal), *Social Media*, *Healthcare*, and *Education*.

7/6/2025
Learning CSS through a Girls Who Code summer course instead of Codecademy. Chose color scheme and started styling the <nav> bar and home page. Used properties such as background-color, text-align, text-decoration, and font-family.

7/8/2025
Working on finer details in CSS, such as keeping elements in line with each other and providing appropriate margins and padding. W3schools has also been a helpful resource for both HTML and CSS. Redesigned nav bar to stay constant across pages and re-structured the website layout for user accessibility. Included images for each page to make the website appealing.

7/9/2025
Learned how to embed Word Documents as PDFs in HTML sites using Geeks for Geeks coding instructions. I had to use the <iframe> element. Populated annotated bibliography into the pertaining page.

7/10/2025
Used w3schools tutorial to learn how to embed screenshots into an HTML page using the <img> element. Will embed the coding journal using <iframe> after I download the finished document as a pdf. Tested user accessibility with a family member.