***CSS Introduction***

* CSS stands for Cascading Style Sheets
* CSS describes how HTML elements are displayed on screen, paper or in other media.
* CSS saves a lot of work. It can control the layout of multiple web pages all at once.
* External stylesheets are stored in CSS files.

*Why use CSS?*

CSS is used to define styles for our web pages, including the design, layout and variations in display for different devices and screen sizes.

*CSS solved a big problem:*

HTML was never intended to contain tags for formatting a web page HTML was created to describe the content of a web page, like:

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

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

When tags like <font> and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process. To solve this problem, the World Wide Web Consortium (W3C) created CSS.

CSS removed the style formatting from the HTML page.

*CSS Saves a lot of Work:* The style definitions are normally saved in external .css files. With an external stylesheet file, you can change the look and feel of an entire website by changing just one file.

How to add CSS?

There are three ways to add CSS to our HTML page:

1. Inline CSS <tag style=”css” />
2. Internal CSS <style>css</style>
3. External Stylesheet <link href=”style.css”/>

Inline CSS is useful, when we need to add styling only for a single element on the web page.

Internal CSS is useful when we are adding styles to a single page website or only for one web page.

External CSS is the most crucial and is used frequently, as it supports styling for multi-page websites. We can use a single CSS file that can be embedded into multiple HTML documents to provide styling.

Examples of all three ways to add CSS is:

index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>CSS</title>

    <link rel="stylesheet" href="style.css">

    <style>

        p{

            background-color: blueviolet;  /\* internal css \*/

        }

    </style>

</head>

<body>

    <h1 style="background-color: burlywood;"> <!-- inline CSS -->

        hello inline css

    </h1>

    <p>

        hello internal css.

    </p>

    <div>

        hello external css

    </div>

</body>

</html>

style.css

div{

    background-color: brown;

}

Output:



***CSS Selectors***

CSS Selectors are used to “find” or select the HTML elements you want to style. There are 5 types of selectors:

1. Element/Type selector
2. Class Selector
3. Id selector
4. Attribute Selector
5. Universal selector

Element/Type Selector: The CSS type/element selector matches elements by node name. in other words, it selects all elements of the given type within a document.

Class Selector: The CSS class selector matches elements based on the contents of their class attribute.

Id Selector: The CSS id selector matches elements based on the contents of their id attribute.

Attribute Selector: The CSS attribute selector matches elements based on the element having a given attribute explicitly set, with options for defining an attribute value or substring value match.

Universal Selector: The CSS universal selector matches elements of any type.

We need to achieve the following:

A close-up of a text

Description automatically generated

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>CSS</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <!-- selectors example -->

    <h1>CSS Selectors</h1>

    <h2>Applying CSS to Different Parts of HTML</h2>

    <!-- TODO 1: Set the CSS for all paragraph tags to "color: red" -->

    <p class="note">1. The element selector targets elements based on their HTML tag name.</p>

    <ol>

        <!-- TODO 2: Set the CSS for all elements with a class of "note" to "font-size: 20px" -->

        <li class="note" value="2">Class selectors target elements based on the value of the class attribute.</li>

        <!-- TODO 3: Set the CSS for the element with an id of "id-selector-demo" to "color: green" -->

        <li class="note" id="id-selector-demo" value="3">ID selectors target elements based on the value of the id

        attribute.</li>

        <!-- TODO 4: Set the CSS for the li elements that have the "value" attribute set to "4" to have "color: blue" -->

        <li class="note" value="4">Attribute selectors target elements based on their attributes and values.</li>

        <!-- TODO 5: Set all elements to have "text-align: center" -->

        <li class="note" value="5">The universal selector targets all elements.</li>

    </ol>

</body>

</html>

Style.css

ol {

    margin-left: -40px;

    margin-top: -20px;

    list-style-position: inside;

  }

/\* TODO 1: Set the CSS for all paragraph tags to "color: red" \*/

p{

    color: red;

}

/\* TODO 2: Set the CSS for all elements with a class of "note" to "font-size: 20px" \*/

.note{

    font-size: 20px;

}

/\* TODO 3: Set the CSS for the element with an id of "id-selector-demo" to "color: green" \*/

#id-selector-demo{

    color: green;

}

/\* TODO 4: Set the CSS for the li elements that have the "value" attribute set to "4" to have "color: blue" \*/

li[value="4"]{

    color: blue;

}

/\* TODO 5: Set all elements to have "text-align: center" \*/

\*{

    text-align: center;

}

Project: Color Vocab Website

* Code is in the repository.

***CSS Properties***

**Color Properties:** we use two types of color properties, the “background-color” and “color”. The background color property is used to specify the shade/color we want for background, and the color property is used to set the color of the text.

Example: Achieve the following using color properties:

A close up of a line

Description automatically generated with medium confidence

Code is in CSS folder in outside index.html and style.css.

**Font Properties:** there are different font properties available in CSS, like “font-size”, “font-weight”, “font-family”, etc. “color” is also a type of font property, as it changes the text/font color.

1. “font-size”: there are different ways in which we can define the font size, such as px(pixels), pt(points), em() and rem().

* 1px: 1 pixel is 1/96th of an inch. This means around 0.26 mm. This is the size that it takes up as a square pixel.
* 1pt: 1 point is the 1/72th of an inch, this means around 0.35 mm, little bit larger than the pixel. While writing the word document, we use the points, like if font size is selected to be 12, then it represents 12 points.
* 1em: em is pronounced as “m” letter. This em is 100% of its parent, for example, if we are applying this property as “h1: {font-size: 1 em; }” and this h1 is enclosed into the body tag, then it will take up the value related to the body tag, say body tag is having 20px size, so 1 em will be equal to “100% of parent” i.e., 20px, while if we set h1 to 2em, then it will be 2\*100% of parent, i.e., 2\*20px = 40px.
* 1rem: rem is also a relative property, but it is “100% of root” and we know that “html” element is our root element. So, rem takes up the size relative to html element.

Example of em vs. rem: let’s say we have below code snippet:

<html>

    <body>

        <h1>hello</h1>

        <footer>

            <h2>there</h2>

        </footer>

    </body>

</html>

1. I am setting the font-size of footer as 20px and font-size of h2 as 2em. So, h2 will take the size relative to its parent in this case, i.e., of footer element. So, 2em = 2\*20px = 40px.
2. Now in this case, font-size of footer is same as above, 20px, and font-size of h2 this time is 2rem. Now, this 2rem will not be affected, if I change the font-size of footer, because it takes up the size relative to its root element, and here root is html element, so if font-size of html is defined to be 10px let’s say, then 2rem will be 2\*10px = 20px.

NOTE: it is recommended to use rem while defining the font-size of any element, because root is only one for each case, while parent can be multiple elements. In the above example, we can see, h2 has footer as its parent element, footer has body and body has html as its parent element. So, this might be a bit confusing, and can lead to inconsistency in the webpages.

1. “font-weight”: we can specify font-weight into three ways – using keywords (bold, normal, etc.), using numbers (100 to 900) and relative to parent (lighter (-100), bolder (+100), etc.)
2. “font-family”: font-family is used to specify the typeface or the font style we want in our website. These can be of two types, “serif” and “sans-serif”. Below is the difference between them.

|  |  |
| --- | --- |
| A black text on a white background  Description automatically generated | A black text on a white background  Description automatically generated |

To include custom fonts in our website, we can go to “fonts.google.com” and can choose any font of our choice and can embed that in our website.

1. “text-align”: This property allows us to manage the text position, where we want our text to be, it has different values available, like – center, start, end, left, right, justify, etc.

Example:

Index.html

<!-- 6.1 font properties -->

    <!-- to-do:

        1. change the color of <p>color</p> to coral color

        2. change the font size of <p>font size</p> to 2X the size of the root font size.

        3. change the font weight of <p>font weight</p> to 900

        4. change the font family of <p>font family</p> to the Google font Caveat with regular(400) font weignt.

        5. change the <p>text align</p> to right aling.

        6. change the root (html element) font size to 30px.

    -->

    <h1>Important CSS properties</h1>

    <p id="color">color</p>

    <p id="size">font size</p>

    <p id="weight">font weight</p>

    <p id="family">font family</p>

    <p id="alignment">text align</p>

Style.css

#color{

      color: coral;

  }

  #size{

      font-size: 2rem;

  }

  #weight{

      font-weight: 900;

  }

  #family{

    font-family: 'Caveat', cursive;

  }

  #alignment{

      text-align: right;

  }

  html{

      font-size: 30px;

  }

Output:



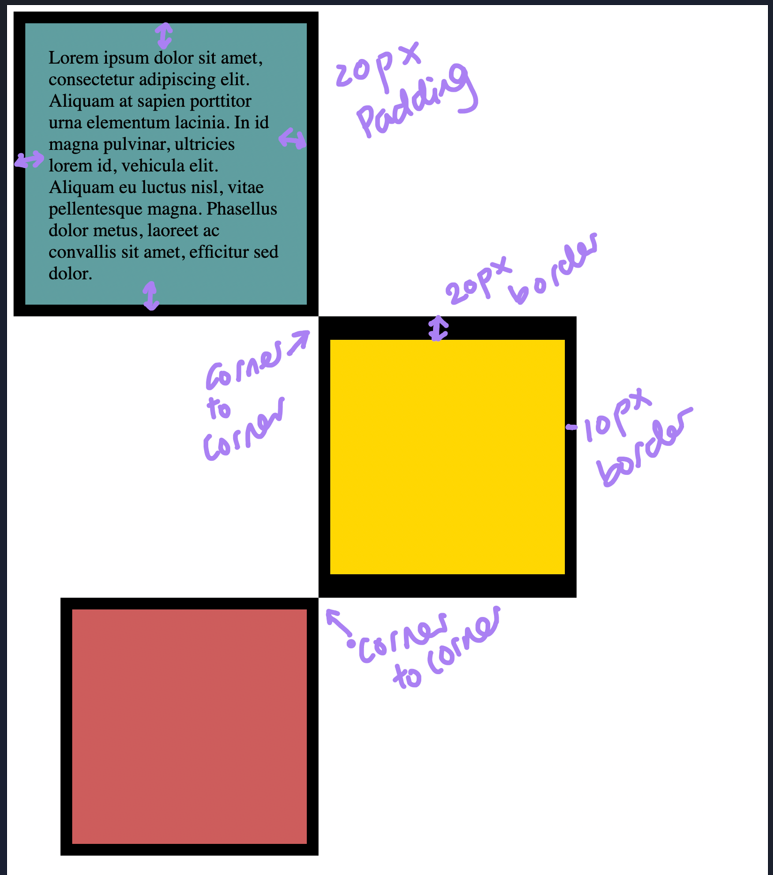
**Inspecting CSS:** we can try and play around using Chrome developer tools.

**The CSS Box Model – Margin, Padding and Border:** The CSS box model is essentially a box that wraps around every HTML element. It consists of margin, border, padding and the actual content. It defines how all these different parts work.

A diagram of a website

Description automatically generated with medium confidence

Complete the below task to get some hands-on experience on the box model. We need to achieve the following:



Code:

index.html

<!-- 6.3 box model in CSS -->

    <!-- to-do:

        1. create 3 boxes using the div element

        2. set their sizes to 200px high by 200px wide.

        3. set different background colors for each of the boxes

        4. add a paragraph element to first div element and add some content to it

        5. set the first div to have 20px padding all around with a black 10px border

        6. fix the style of <p> element to remove all margins.

        7. set the second div to have a 20px border on top and bottom and 10px border on left and right.

        8. set the third div to have a 10px border.

        9. set the margins for the dives so that each box corner touches the other.

    -->

    <div id="first">

        <p>Lorem ipsum dolor, sit amet consectetur adipisicing elit. Consectetur blanditiis quidem magni explicabo excepturi commodi harum repellendus laudantium! Iure excepturi, eos non incidunt assumenda officiis sit beatae suscipit minima voluptates.</p>

    </div>

    <div id="second"></div>

    <div id="third"></div>

style.css

  div{

      width: 200px;

      height: 200px;

  }

  p{

      margin: 0;

  }

  #first{

    background-color: aquamarine;

    padding: 20px;

    border: 10px solid black;

  }

  #second{

    background-color: bisque;

    border: solid black;

    border-width: 20px 10px;

    margin-left: 260px;

  }

  #third{

    background-color: burlywood;

    border: 10px solid black;

    margin-left: 40px;

  }

Output:

A close-up of a screen

Description automatically generated