CSS3 (Cascading Style Sheets)

CSS stands for Cascading Style Sheets. CSS describes how HTML elements are to be 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. In other words, it's a style sheet language that determines how the elements/contents in the page are looked/shown. CSS is used to develop a consistent look and feel for all the pages.

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

CSS enables the separation of the content from the presentation. This separation provides a lot of flexibility and control over how the website has to look like. This is the main advantage of using CSS.

A CSS rule consists of a selector and a declaration block.

CSS syntax:

```
Selector {
            Property1: value1;
            Property2: value2;
            …..
      }
```

The selector can be a HTML element that we want to style.

Styling can be applied in 3 different ways:

1. Inline style sheets
2. Internal style sheets
3. External style sheets

Or

Local style, Page-Level style, and External Styles.

1. Local styles

Also known as inline. This form is defined within your HTML tags/elements. It's mostly used to style specific elements in your code.

```
<html>
   <head>
     <title>Cascading Style Sheets</title>
   </head>
   <body>
     <p style = "font-family: sans-serif;
        font-size: 1.2em
        font-style: italic;">
```

```
        This paragraph is an example of a local style.
      </p>
      <p>This is an Unaffected paragraph</p>
  </body>
</html>
```

This code snippet edits the font of the p tag within the body. However, it only changes the contents of the first p tag. The second p tag maintains the webpage's default style.

Pros:

- Easy to test designs on individual elements.
- You can use one document to load CSS styles.

Cons:

- It can take a long time to load or render a page since multiple individual elements are styled within.
- This final file will look disorganized, thus making it harder to read.

2. Page-Level styles

Page-level styles are defined at the header area of the HTML file. All similar tags, whether elements members of the class or ID selector within the body of the HTML will undergo the changes at once. An ID selector can only identify one element each while Class selectors can identify more than just one at a time.

```
<html>
  <head>
    <title>Cascading Style Sheets</title>
    <meta charset="utf-8">
    <style type="text/css">
      body {
        color: yellow;
        background-color: red;
      }
      p {
        color: red;
        background-color: yellow;
      }
    </style>
  </head>
  <body>
    <h1>Heading</h1>
    <p>This paragraph has been styled using page level styling</p>
  </body>
</html>
```

Pros:

- Easy to manage during initial development stages.

- The programmer doesn't need to keep switching files to make minor adjustments.

Cons:

- Adding code to the document increases the size of the file.
- In order to manipulate a website design, you would have to acquire all the files that contain CSS.

3. External Styles

The styles used for the webpage are located in a completely different file. This other file purely contains CSS code and is saved with a .css extension. The .HTML file is linked to the .css file that can be imported to modify the webpage style.

```
<!DOCTYPE html>
 <html>
  <head>
    <meta charset="utf-8">
    <title>External Styles</title>
    <link rel="stylesheet" type="text/css" href="css/myStyle.css"/>
  </head>
  <body>
   <h1>Heading</h1>
    <p>This is an example of External CSS</p>
  </body>
 </html>

/*This is the CSS file called myStyle.css*/

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

 p {
  color: purple;
 }
```

Pros:

- One style sheet controls many pages.
- Separation of content and design.

Cons:

- You cannot render the page until the entire CSS file is downloaded.
- Linking multiple CSS files to the page can lead to downtime.

Features of CSS3:

1. CSS Animations and Transitions

2. Calculating values with calc()
3. Advanced Selectors
4. Generated Content and Counters
5. Gradient
6. Web fonts
7. Box sizing
8. Border images
9. Media queries
10. Multiple backgrounds
11. CSS columns
12. CSS 3D transforms

Some of the key modules are:

- Box model
- Image values and replaced content
- Text effects
- Selectors
- Backgrounds and borders
- Animations
- User interface (UI)
- Multiple column layouts
- 2D/3D transformations

Advantages of CSS3

- CSS3 provides a consistent and precise positioning of navigable elements.
- It is easy to customize a web page as it can be done by merely altering a modular file.
- Graphics are easier in CSS3, thus making it easy to make the site appealing.
- It permits online videos to be seen without using third-party plug-ins.
- CSS3 is economical, time-saving, and most browsers support it.

CSS frameworks are the pre-planned libraries which make easy and more standard compliant web page styling. The frequently used CSS frameworks are: -

- Bootstrap
- Foundation
- Semantic UI
- Gumby
- Ulkit

CSS selectors:

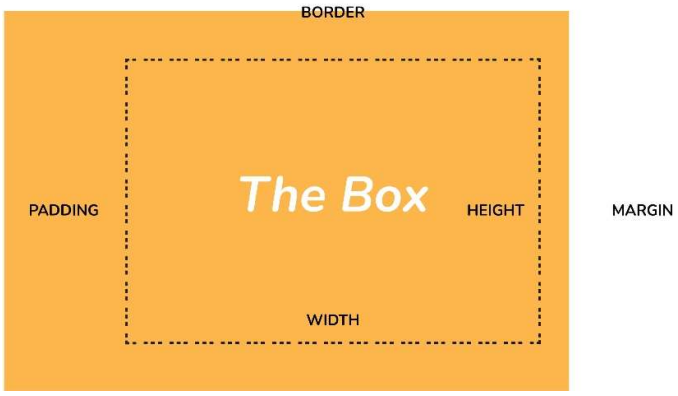
A CSS selector selects the HTML element(s) you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinator selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

CSS Box Model and the related properties:

A rectangle box is wrapped around every HTML element. The box model is used to determine the height and width of the rectangular box. The CSS Box consists of Width and height (or in the absence of that, default values and the content inside), padding, borders, margin.



- **Content:** Actual Content of the box where the text or image is placed.
- **Padding:** Area surrounding the content (Space between the border and content).
- **Border:** Area surrounding the padding.
- **Margin:** Area surrounding the border.

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-color: lightgrey;
  width: 100px;
  border: 15px solid green;
  padding: 50px;
  margin: 15px;
  font-size: 20px;
}
</style>
</head>
```

```
<body>
<h2>Demonstrating the Box Model</h2>
<div>CSE</div>
<div>AI and DS</div>
<div>AI and ML</div>
</body>
</html>
```

Specificity:

If there are two or more CSS rules that point to the same element, the selector with the highest specificity value will "win", and its style declaration will be applied to that HTML element.

Think of specificity as a score/rank that determines which style declaration are ultimately applied to an element.

Look at the following examples: In this example, we have added a class selector (named "test"), and specified a green color for this class. The text will now be green (even though we have specified a red color for the element selector "p". This is because the class selector is given higher priority:

```
<!DOCTYPE html>
<html>
<head>
<style>
.test {color: green;}
    p {color: red;}
</style>
</head>
<body>
<p class="test">Hello World! </p>
</body>
</html>
```

Advanced Features:

1) Rounder corners to the elements:

```
<!DOCTYPE html>
<html>
<head>
<style>
#rcorners1 {
  border-radius: 25px;
  background: #73AD21;
```

```
  padding: 20px;
  width: 100px;
  height: 50px;
  text-align: center;
}

#rcorners2 {
  border-radius: 25px;
  background: url(paper.gif);
  background-position: left top;
  background-repeat: repeat;
  padding: 20px;
  width: 100px;
  height: 50px;
}
</style>
</head>
<body>

<h1>The border-radius Property</h1>
<p>Rounded corners for an element with a specified background color:</p>
<p id="rcorners1">Rounded corners</p>
<p id="rcorners2">Rounded corners</p>
</body>
</html>
```

2) CSS Colors:

RGBA Colors: RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color. An RGBA color value is specified with: rgba(red, green, blue, alpha). The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:rgba(255,0,0,0.3);}
#p2 {background-color:rgba(0,255,0,0.3);}
#p3 {background-color:rgba(0,0,255,0.3);}
</style>
</head>
<body>
```

```
<h1>Define Colors With RGBA Values</h1>

<p id="p1">Red</p>
<p id="p2">Green</p>
<p id="p3">Blue</p>

</body>
</html>
```

HSL Colors: HSL stands for Hue, Saturation and Lightness. An HSL color value is specified with: hsl(hue, saturation, lightness).

1. Hue is a degree on the color wheel (from 0 to 360):
    - 0 (or 360) is red
    - 120 is green
    - 240 is blue
2. Saturation is a percentage value: 100% is the full color.
3. Lightness is also a percentage; 0% is dark (black) and 100% is white.

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:hsl(120,100%,50%);}
#p2 {background-color:hsl(120,100%,75%);}
#p3 {background-color:hsl(120,100%,25%);}
</style>
</head>
<body>

<h1>Define Colors With HSL Values</h1>

<p id="p1">Green</p>
<p id="p2">Light green</p>
<p id="p3">Dark green</p>

</body>
</html>
```

Opacity

The CSS opacity property sets the opacity for the whole element (both background color and text will be opaque/transparent).

The opacity property value must be a number between 0.0 (fully transparent) and 1.0 (fully opaque).

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:rgb(255,0,0);opacity:0.6;}
#p2 {background-color:rgb(0,255,0);opacity:0.6;}
#p3 {background-color:rgb(0,0,255);opacity:0.6;}

</style>
</head>
<body>

<h1>Define Colors With Opacity</h1>

<p id="p1">Red</p>
<p id="p2">Green</p>
<p id="p3">Blue</p>

</body>
</html>
```

CSS Backgrounds: The CSS background properties are used to add background effects for elements. The following are the CSS background properties:

- `background-color`
- `background-image`
- `background-repeat`
- `background-attachment`
- `background-position`
- `background` (shorthand property)

background-color: with CSS, a color is most often specified by:

- a valid color name - like "red"
- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"

background-image: The `background-image` property specifies an image to use as the background of an element. By default, the image is repeated so it covers the entire element.

```
Body { background-image: url("paper.gif");

}
```

background-repeat: By default, the `background-image` property repeats an image both horizontally and vertically. Some images should be repeated only horizontally or vertically, or they will look strange, like this:

repeat-x, repeat-y, no-repeat

```
body {
  background-image: url("gradient_bg.png");
  background-repeat: repeat-x;
}
```

background-position

The `background-position` property is used to specify the position of the background image.

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
}
```

background-attachment

The `background-attachment` property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page):

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
  background-attachment: fixed;
}
```

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
  background-attachment: scroll;
}
```

CSS Animation:

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: sample;
  animation-duration: 4s;
}

@keyframes sample {
  from {background-color: red;}
  to {background-color: yellow;}
}
</style>
</head>
<body>

<h1>CSS Animation</h1>

<div></div>

<p><b>Note:</b> When an animation is finished, it goes back to its original style.</p>

</body>
</html>
```

CSS Transitions

CSS transitions allows you to change property values smoothly, over a given duration.

Mouse over the element below to see a CSS transition effect:

- transition
- transition-delay
- transition-duration

- transition-property
- transition-timing-function

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

Note: If the duration part is not specified, the transition will have no effect, because the default value is 0.

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}

div:hover {
  width: 300px;
}
</style>
</head>
<body>
<h1>The transition Property</h1>
<p>Hover over the div element below, to see the transition effect:</p>
<div></div>

</body>
</html>
```

CSS 2D Transforms: CSS transforms allow you to move, rotate, scale, and skew elements. Mouse over the element below to see a 2D transformation. With the CSS transform property you can use the following 2D transformation methods:

- translate()
- rotate()

- scaleX()
- scaleY()
- scale()
- skewX()
- skewY()
- skew()
- matrix()

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 300px;
  height: 100px;
  background-color: yellow;
  border: 1px solid black;
}
div#one {
  transform: rotate(20deg);
}
div#two {
 transform: rotate(-20deg);
}
div#three {
 transform: scale(2, 3);
}
div#four {
 transform: scale(0.5, 0.5);
}
div#five {
 transform: scaleX(2);
}
</style>
</head>
<body>
<h1>The rotate() Method</h1>
<p>The rotate() method rotates an element clockwise or counter-clockwise.</p>

<div>
This a normal div element.
```

```
</div>
<div id="one">
This div element is rotated clockwise 20 degrees.
</div>
<div id="two">
This div element is rotated counter clockwise 20 degrees.
</div>
<div id="three">
This div element is scaled
</div>
<div id="four">
This div element is reduced
</div>
<div id="five">
This div scaling in X direction
</div>
</body>
</html>
```

CSS Multiple Columns:

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
.newspaper {
  column-count: 3;
}
</style>
</head>
<body>

<h1>Javascript notes in Create Multiple Columns</h1>

<div class="newspaper">
```

JavaScript is a lightweight, cross-platform, and interpreted scripting language. It is well-known for the development of web pages; many non-browser environments also use it. JavaScript can be used for Client-side developments as well as Server-side developments. JavaScript contains a standard library of objects, like Array, Date, and Math, and a core set of language elements like operators, control structures, and statements.

```
</div>

</body>
</html>
```

CSS Flex: Before the Flexbox Layout module, there were four layout modes:

- Block, for sections in a webpage
- Inline, for text
- Table, for two-dimensional table data
- Positioned, for explicit position of an element

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

The flex container properties are:

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

Ex:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  background-color: #FF0000;
}

.flex-container > div {
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
</style>
</head>
```

```
<body>
<h1>Create a Flex Container</h1>
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
</body>
</html>
```