Web Technology I

Unit 5

Cascading Style Sheet (CSS)

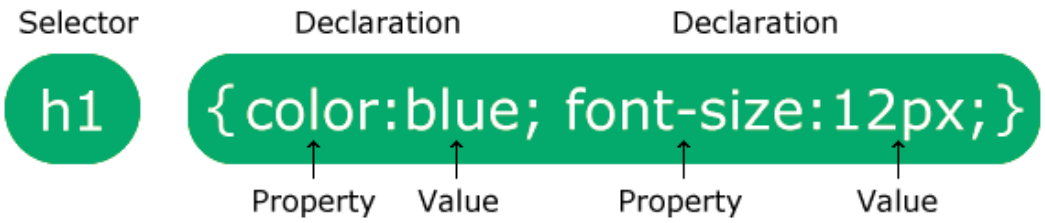
- Introduction to Cascading Style Sheets (CSS)
- Advantages of using CSS
- Basic Syntax: Creating Cascading Style Sheets (CSS) using **<style>** tag
- Types of Style Sheets
    - Inline Style Sheets
    - Internal/ Embedded Style Sheets
    - External Style Sheets
- Introduction to different Styles and their Attributes
    - Backgrounds and Color Styles and Attributes
    - Fonts and Text Styles and Attributes
    - Margin, Padding and Border Styles and Attributes
    - List Styles and Table Layouts
    - Additional Features - Grouping Style Sheets, Assigning Classes and Span
    - DIV Tag
    - Responsive Web Design

CSS

- CSS stands for **Cascading Style Sheet**.
- CSS is the language used to style an HTML document.
- CSS describes how HTML elements should be displayed
- CSS3 is the latest version of the CSS language.

Syntax:

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



Example:

```
body {
  background-color: lightblue;
}

h1 {
  color: white;
  text-align: center;
}
```

Advantages of using CSS

- CSS separates presentation and content for easier maintenance.
- CSS ensures consistency across webpages for better branding and user experience.
- CSS reduces page load times for faster performance.
- CSS enhances accessibility for users with disabilities.
- CSS provides greater design flexibility for creative and appealing designs.

Types of Style Sheets

There are 3 types of CSS

1. Inline Style Sheets
2. Internal/ Embedded Style Sheets
3. External Style Sheets

i. Inline Style Sheets

- Inline CSS is used to apply a style to a single HTML element.
- It is defined within the HTML tag using the "style" attribute.
- Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Inline CSS</title>
  </head>
  <body style="background-color: white">
    <h1 style="color: Red;">Inline CSS</h1>
    <p style="color: blue">Hey there.</p>
  </body>
</html>
```

ii. Internal/ Embedded Style Sheets

- Internal CSS is used to apply a style within the same HTML document.
- It is defined within the **head** section of the HTML document using the **<style>** tag.
- Example:

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {
        background-color: black;
      }
      h1 {
        color: red;
        padding: 50px;
```

```
      }
    </style>
  </head>
  <body>
    <h2>CSS types</h2>
    <p>Cascading Style sheet types: internal CSS</p>
  </body>
</html>
```

iii. External Style Sheets

- In external CSS we create separate CSS file. Eg. **style.css**
- The external CSS file is linked to HTML file using **<link>** tag.
- The reference of file is given with the attribute **href="styles.css".**

File Name: index.html

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h2>CSS types</h2>
    <p>Cascading Style sheet types: internal CSS</p>
  </body>
</html>
```

File Name: **style.css**

```
body {
  background-color: black;
}
h1 {
  color: red;
  padding: 50px;
}
```

CSS Background

- CSS background property is used to define the background effects on element.
- Some background properties are given below:

1. background-color : specifies the background color of the element.

2. background-image : specifies the background image of the element.

3. background-repeat : The background-repeat property sets if/how a background image will be repeated. It has following properties:
   - repeat : background image is repeated vertically and horizontally.
   - repeat-x : background image is repeated horizontally.
   - repeat-y : background image is repeated vertically.
   - no-repeat : background image is not repeated.

4. background-attachment : specifies if the background image is fixed or scroll with the rest of the page. It has the following properties.
   - Scroll: The background image will scroll with the page. This is default.
   - Fixed: The background image will not scroll with the page.

5. background-position: The background-position property sets the starting position of a background image. It has following properties.
   1. center
   2. top
   3. bottom
   4. left
   5. right

6. background-size: specifies the size of the background images. It has following properties:

   - auto : Default value. The background image is displayed in its original size
   - contain : Resize the background image to make sure the image is fully visible.
   - cover : Resize the background image to cover the entire container.
   - length : Sets the width and height of the background image.
   - percentage: Sets the width and height of the background image in percent.

Example:

```
body {
  background-color: red;
  background-image: url("cat.jpg");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  background-attachment: fixed;
}
```

CSS Text

CSS has a lot of properties for formatting text. Some of them are:

- color : The color property is used to set the color of the text. The color is specified by:
  - a color name - like "red"
  - a HEX value - like "#ff0000"
  - an RGB value - like "rgb(255,0,0)"
- direction : specifies the direction of an element. It has following values.
  - ltr : text direction from left-to-right.
  - rtl : text direction from right-to-left.
- text-decoration : The text-decoration CSS property is used to add decorative lines or effects to text. It is a shorthand property for:
  - text-decoration-line (required) : none | underline | overline | line-through | blink
  - text-decoration-color : color of the text decoration.
  - text-decoration-style : solid | wavy | dotted | dashed | double
  - text-decoration-thickness: thickness of the decoration.
- text-transform : The text-transform property controls the capitalization of text. It's values are none, uppercase, lowercase , capitalize.
- letter-spacing : Specifies the space between letters.
- word-spacing : Specifies the space between words.

Example:

```
p {
  color: green;
  direction: rtl;
  text-decoration: underline;
  text-transform: uppercase;
  word-spacing: 2px;
```

```
  line-height: 1.5;
}
```

CSS Fonts

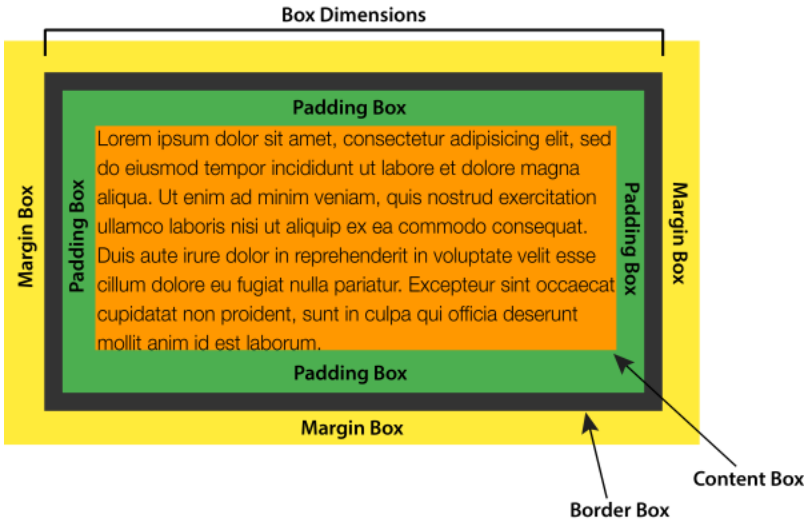
In CSS, we can style fonts by changing properties such as:

i. font-family: specifies the type of font. Eg. Serif, sans-serif, monospace, cursive, times new roman etc.
ii. font-size: species size of the font. Eg. 16px, 2em, 1.5rem, 5%, 10vw etc.
iii. font-weight: specifies the boldness of a font. Eg. Normal, bold, 100-900 etc.
iv. font-style: used to specify italic text.

Example:

```
p{
  font-family: sans-serif;
  font-size: 20px;
  font-weight: bold;
  font-style: italic;
}
```

Margin, Padding and Border Styles and Attributes



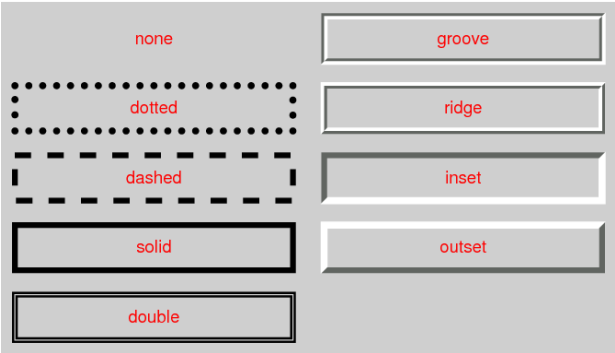
Every html element consists of the properties:

1. Content: This is the actual content of the element, such as text or an image.
2. Padding: This is the space between the content and the border of the element.
3. Border: This is a line that around the padding and content.
4. Margin: This is the space between the border and the neighboring elements.

CSS Borders:

We can style border with various properties such as:

1. border-style: none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset.



2. border-width: width of border. Eg. 2px
3. border-color: color of the border. Eg. Red, yellow, #ffffff, rgb(255,232,200) etc.
4. border-radius: roundness of border. 8px, 50% etc.
5. border-sides: (border-top, border-bottom, border-left, border-right)

border shorthand:

selector{

border: size style color;

}

Example:

```
p {
  border: 5px solid red;
}
```

CSS Margin

The properties for setting margins are:

- margin-top: 10px (margin only on top)

- margin-right: 10px (margin only on right)
- margin-bottom : 10px (margin only on bottom)
- margin-left: 10px (margin only on left)

margin shorthand:

- margin: 10px 8px 20px 25px; (10px top, 8px right, 20px bottom, 25px right)
- margin: 20px; (20px on all side)
- margin: 10px 20px; (10px tob and bottom, 20px left and right)

CSS Padding

The properties for setting paddings are:

- padding-top: 10px (padding only on top)
- padding-right: 10px (padding only on right)
- padding-bottom : 10px (padding only on bottom)
- padding-left: 10px (padding only on left)

padding shorthand:

- padding: 10px 8px 20px 25px; (10px top, 8px right, 20px bottom, 25px right)
- padding: 20px; (20px on all side)
- padding: 10px 20px; (10px tob and bottom, 20px left and right)

List Styles

In CSS, we can style list items using several properties:

- list-style-type: none, disc, circle, square, decimal, lower-roman, upper-alpha, etc.
- list-style-image: specifies an image as the list item marker. Eg. url("cat.jpg)
- list-style-position: specifies the position of list item maker.

Example:

```
ul {
  list-style-type: disc;
  list-style-image: url('my-marker.png');
  list-style-position: inside;
}
```

Table Styles:

In CSS, we can style table using several properties:

- border: specifies the border of the table.
- border-collapse: collapse. (Collapses the border between cells)
- height: specifies height of the table.
- width: specifies width of the table.
- text-align: specifies the alignment of the text. (left, right, center)
- vertical-align: specifies vertical alignment the text. (top, bottom, middle)
- padding: specifies padding for the content.

Example:

```
table {
  border: 1px solid black;
  border-collapse: collapse;
  background-color: #f9f9f9;
  height: 250px;
  width: 250px;
}

th, td {
  padding: 8px;
  text-align: center;
  vertical-align: middle;
}
```

Grouping Style Sheets

- In CSS (Cascading Style Sheets), grouping styles to apply the same styles to multiple elements at once.
- We can select multiple elements or classes using comma (,).
- Example:

```
h1, h2, p, .section {
  font-family: sans-serif;
  color: #333333;
}
```

Assigning Classes and ID

- Classes and IDs are attributes in HTML.
- They are used to identify and group elements in a webpage.
- They are commonly used for targeting elements in CSS and JavaScript.

Class attribute:

- A class is defined using the "class" attribute.
- A class can have multiple values.
- Multiple elements can have same class.
- In css we use ".classname" to select an element.

Example:

HTML:

```
<div class="box">This is a box</div>
<div class="box">This is another box</div>
```

CSS:

```
.box {
  background-color: yellow;
  border: 1px solid black;
  padding: 10px;
}
```

Id attribute:

- The id is defined using the "id" attribute.
- Each element can only have one unique id attribute.
- In css we use "#idname" to select an element.

Example:

HTML:

```
<div id="header">This is the header</div>
```

CSS:

```css
#header {
  background-color: blue;
  color: white;
  padding: 10px;
}
```

Div tag and Span tag

- Both and tags are used for organizing
- They are also used for styling content on a web page with CSS.

tag

- The ** ** tag is used to group elements to create a section or container.
- Content within a ** ** starts on a new line.
- It is commonly used to create sections or areas on a page, such as headers, footers, or content blocks.

Example

```html
<div id="header">
  <h1>Welcome to my website!</h1>
  <nav>
    <ul>
      <li><a href="#">Home</a></li>
      <li><a href="#">About</a></li>
      <li><a href="#">Contact</a></li>
    </ul>
  </nav>
</div>
```

tag

- **** groups inline elements together.
- Content in **** is displayed inline with other elements.

- **** does not create a new line.
- It's often used to style specific text parts, like highlighting words.

Example:

```
<p>This is a <span class="highlight"> highlighted </span> word in a paragraph.</p>
```

Responsive Web Design

- Responsive web design is a web design approach.
- We use responsive design to make contents adapt and look good on any screen such as mobile, TV, tablet, laptop, desktop etc.
- It improves website performance and load times on different devices.
- Responsive design is important for SEO and helps users find relevant content on mobile devices.
- It helps to reach and engage a wider audience, especially with the increasing use of mobile devices.

There are several techniques we can apply to make responsive webpages:

1. The viewport meta tag:
    - The viewport meta tag is used to tell mobile browsers how to display a web page.

- This tag instructs mobile browsers to set the width of the viewport to match the device width.
- Additionally, it tells the browser to scale the document to 100% of its intended size.
- This ensures that the page appears in a mobile-optimized size on the device.

Example:

```
<meta name="viewport" content="width=device-width,initial-scale=1" />
```

2. CSS media queries:
   - Media queries are commonly used in responsive web design.
   - These allow designers to specify different styles for different screen sizes and orientations.
   - Media queries use @media rule and CSS styles.

```
@media (min-width: 400px) {
  /* Styles for viewports wider than 400 pixels. */
}

@media (max-width: 400px) {
  /* Styles for viewports narrower than 400 pixels. */
}

@media (min-width: 50em) and (max-width: 60em) {
  /* Styles for viewports wider than 50em and narrower than 60em. */
}
```

   - Example:

```
img{
    width: 50px;
}
@media screen and (max-width: 600px) {
  img {
    width: 25px;
  }
}
```

3. Modern CSS layout techniques such as Flexbox, Grid Layout, and Multicol: These allow web page elements to adjust their size and position based on the screen size.

4. Responsive images: These use techniques such as srcset and sizes attributes to deliver different sized images to different devices, improving load times and performance. To ensure media is never larger than its responsive container, the following approach can be used:

Example:

```
img, picture, video {
  max-width: 100%;
}
```

5. Responsive typography: We can achieve responsive texts using media-queries and responsive units such as (em, rem and vw)
   Example:

```
html {
  font-size: 1em;
}

h1 {
  font-size: 2rem;
}

p{
  font-size: 6vw;
}

@media (min-width: 1200px) {
  h1 {
    font-size: 4rem;
  }
}
```

6. Mobile-first design: This involves designing for smaller screens first, then scaling up for larger screens, to ensure that the website is optimized for mobile devices.
7. Frameworks and libraries: CSS frameworks such as (Bootstrap, tailwindcss etc.) provide pre-built responsive design components, such as grids, navigation menus, and form elements, that can be easily integrated into a website.