UNIT-III
CASCADING STYLE SHEET(CSS)

CSS Overview:

CSS (Cascading Style Sheets) is a language designed to simplify the process of making web pages presentable.

- It allows you to apply styles to HTML documents by prescribing colors, fonts, spacing, and positioning.
- The main advantages are the separation of content (in HTML) and styling (in CSS) and the same CSS rules can be used across all pages and not have to be rewritten.
- HTML uses tags and CSS uses rule sets.
- CSS styles are applied to the HTML element using selectors.

CSS Syntax

CSS consists of style rules that are interpreted by the browser and applied to the corresponding elements. A style rule set includes a selector and a declaration block.

- **Selector:** Targets specific HTML elements to apply styles.
- **Declaration:** Combination of a property and its corresponding value.

Example

```
p {
   color: blue;
   text-align: center;
}
```

CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces. In this example, all paragraph element (<p> tag) will be centre-aligned, with a blue text color.

Types of CSS (Cascading Style Sheet)

CSS (Cascading Style Sheets) is used to style and layout of web pages, and controlling the appearance of HTML elements. CSS targets HTML elements and applies style rules to dictate their appearance.

Below are the types of CSS:

- Inline CSS

- Internal or Embedded CSS
- External CSS

1. Inline CSS

Inline CSS involves applying styles directly to individual HTML elements using the style attribute. This method allows for specific styling of elements within the HTML document, overriding any external or internal styles.

```
<p style="color:#009900;
        font-size:50px;
        font-style:italic;
        text-align:center;">
    Inline CSS
</p>
```

2. Internal or Embedded CSS

Internal or Embedded CSS is defined within the HTML document's <style> element. It applies styles to specified HTML elements. The CSS rule set should be within the HTML file in the head section i.e. the CSS is embedded within the <style> tag inside the head section of the HTML file.

```
<!DOCTYPE html>
<html>
<head>
   <style>
      .main {
         text-align: center;
      }
      .GFG {
         color: #009900;
         font-size: 50px;
         font-weight: bold;
      }
      .geeks {
         font-style: bold;
         font-size: 20px;
```

```
        }
    </style>
</head>
<body>
    <div class="main">
        <div class="GFG">Internal CSS</div>
        <div class="geeks">
            Implementation of Internal CSS
        </div>
    </div>
</body>

</html>
```

3. External CSS

External CSS contains separate CSS files that contain only style properties with the help of tag attributes (For example class, id, heading, … etc). CSS property is written in a separate file with a .css extension and should be linked to the HTML document using a **link** tag. It means that, for each element, style can be set only once and will be applied across web pages.

HTMLCSS

```
body {
    background-color: powderblue;
}
.main {
    text-align: center;
}
.GFG {
    color: #009900;
    font-size: 50px;
    font-weight: bold;
}
#geeks {
```

```
   font-style: bold;
   font-size: 20px;
}
```

CSS Rules:

A CSS rule is a grouping of one or more CSS properties which are to be applied to one or more target HTML elements. A CSS rule consists of a CSS selector and a set of CSS properties. The CSS selector determines what HTML elements to target with the CSS rule.

A *CSS rule* is a grouping of one or more CSS properties which are to be applied to one or more target HTML elements.

A CSS rule consists of a CSS selector and a set of CSS properties. The CSS selector determines what HTML elements to target with the CSS rule. The CSS properties specifies what to style of the targeted HTML elements.

Here is a CSS rule example:

```
div {
   border    : 1px solid black;
   font-size : 18px;
}
```

This example creates a CSS rule that targets all div elements, and the set the CSS properties border and font-size for the targeted elements.

The CSS selector part a CSS rule is the part before the first {. In the example above it is the div part of the CSS rule. The CSS properties are listed inside the { ... } block.

CSS rules have to be specified inside either a style element or inside an external CSS file.

Example with type Selectors and the Universal Selector:

Selectors:

CSS selectors are used to target HTML elements on your pages, allowing you to apply styles based on their ID, class, type attributes, and more. There are mainly 5 types of selectors.

- **Basic CSS Selectors:** These are used to target elements by tag, .class, or #id for fundamental styling needs.
- **Combinators:** Ideal for styling elements based on their DOM relationships (e.g., parent-child or sibling relationships).
- **Group Selectors:** Use to apply the same styles to multiple, unrelated elements simultaneously.
- **Attribute Selectors:** Perfect for styling elements based on specific attributes or values, such as form inputs or links with certain prefixes or states.
- **Pseudo-Classes:** Best for styling elements dynamically or interactively, like :hover for user interaction or :nth-child() for structural styling.

Types of CSS Selectors

Basic Selectors

Basic selectors in CSS are simple tools used to target specific HTML elements for styling. These include selecting by element name (e.g., h1), class (.class Name), ID (#idName), or universally (* for all elements).

1. Universal Selector (*):The universal CSS selector is used to select all elements. It is marked with a *. Here is a universal CSS selector example:

```
* {
   font-size: 18px;
}
```

This example selects all HTML elements and set their font-size CSS property.

The universal CSS selector is not so often used alone. It is more often used with a child selector or descendant selector.

2. Element Selector: Targets all elements of a specific type, such as paragraphs or headers. For example, setting a common font size for all paragraphs

```
{...}
  <style>
    p {
      font-size: 16px;
    }
  </style>
{...}
```

3. Class Selector (.): Applies styles to elements with a specific class attribute. For instance, making all buttons have a blue background.

```
{...}
  <style>
    .button {
      background-color: blue;
      color: white;
    }
  </style>
{...}
```

4. ID Selector (#): Styles a single element identified by its unique id. For example, changing the background color of a header

```
{...}
  <style>
    #header {
      background-color: gray;
    }
  </style>
{...}
```

Span and div Elements:

Span:

The HTML span element is a **generic inline container** for inline elements and content. It used to group elements for styling purposes (by using the class or id attributes). A better way to use it when no other semantic element is available.

```
<!DOCTYPE html>
<html>
<head>
   <title>span tag</title>
</head>
<body>
   <p>
      <span style="background-color:lightgreen">
         GeeksforGeeks
      </span>
      is A Computer Science Portal where you can
      <span style="color:blue;">
         Publish
      </span> your own
      <span style="background-color:lightblue">
         articles
      </span>
      and share your knowledge with the world!!
   </p>
</body>
</html>
```

div Element:

The div tag is known as the **Division tag**. The HTML tag is a block-level element used for grouping and structuring content. It provides a container to organize and style sections of a webpage, facilitating layout design and CSS styling.

```
<!DOCTYPE html>
<html>
<head>
   <title>Div tag</title>
   <style>
      div {
         color: white;
         margin: 2px;
         font-size: 25px;
      }
```

```
      .div1 {
         background-color: rgb(142, 142, 245);
      }
      .div2 {
         background-color: #9af19a;
      }
      .div3 {
         background-color: rgb(232, 232, 137)
      }
      .div0 {
         background-color: #009900;
      }
   </style>
</head>
<body>
   <div class="div1"> div tag </div>
   <div class="div2"> div tag </div>
   <div class="div3"> div tag </div>
   <div class="div0"> div tag </div>
</body>
</html>
```

Cascading Style Attribute:

Attributes

Attribute selectors in CSS target elements based on the presence or value of their attributes. Examples include [attr] (selects elements with the attribute), [attr=”value”] (matches specific values), and [attr^=”val”] (matches values starting with “val”).

1. Presence Selector: It selects elements that contain a specific attribute. For example, styling all inputs with a type attribute.

```
{...}
   <style>
      input[type] {
         border: 2px solid black;
      }
   </style>
{...}
```

2. Attribute Value Selector: It targets elements with a particular attribute value. For example, styling text inputs.

```
{...}
  <style>
    input[type="text"] {
      background-color: yellow;
    }
  </style>
{...}
```

3. Substring Matching(^=): It matches elements where the attribute contains a substring. For example, styling links with https in their href.

```
{...}
  <style>
    a[href^="https"] {
      color: green;
    }
  </style>
{...}
```

4. Wildcard Selector (*=): Matches elements where the attribute value contains a specific string. For example, underlining links with example in the URL.

```
{...}
  <style>
    a[href*="example"] {
      text-decoration: underline;
    }
  </style>
{...}
```

CSS Style container:

The **@container** CSS at-rule is a conditional group rule that applies styles to a containment context. Style declarations are filtered by a condition and applied to the container if the condition

is true. The condition is evaluated when the queried container size, <style-feature>, or scroll-state changes.

The container-name property specifies a list of query container names. These names can be used by @container rules to filter which query containers are targeted. The optional, case-sensitive <container-name> filters the query containers that are targeted by the query.

Once an eligible query container has been selected for an element, each container feature in the <container-condition> is evaluated against that query container.

Syntax

The @container at-rule has the following syntax:

```
@container <container-condition># {
  <stylesheet>
}
<!DOCTYPE html>
<html>
<head>
<style>
.parent {
  container-name: myContainer;
  container-type: inline-size;
}
/* Add styles if myContainer is less than 500px wide */
@container myContainer (width < 500px) {
  .child {
    width: 50%;
    border: 2px solid maroon;
    background-color: salmon;
  }
}
```

```
</style>
</head>
<body>
<h1>Demo of @container</h1>
<p>Resize the page to see the effect.</p>
<div class="parent">
  <div class="child">
    <h2>A header</h2>
    <p>Some text.</p>
  </div>
</div>
</body>
</html>
```

External CSS Files:

External CSS is a method used to style multiple HTML pages with a single stylesheet. This approach involves creating a separate CSS file with a .css extension that contains style properties applied to various selectors (such as classes, IDs, headings, etc.). By using external CSS, you can maintain a consistent design across multiple web pages efficiently.

External CSS to HTML

To link an external CSS file to an HTML document, you need to use the <link> element within the <head> section of your HTML file. The <link> element should have the rel attribute set to "stylesheet" and the href attribute specifying the path to your CSS file.

Syntax:

To link an external CSS file to an HTML document, you need to use the <link> element within the <head> section of your HTML file. The <link> element should have the rel attribute set to "stylesheet" and the href attribute specifying the path to your CSS file.

```
<link rel="stylesheet" href="path/to/your/styles.css">
```

Example:

```
<!DOCTYPE html>
<html>
<head>
```

```
    <link rel="stylesheet" href="geeks.css" />
</head>
<body>
    <div class="main">
        <h1 class="GFG">
            GeeksForGeeks
        </h1>
        <p id="geeks">
            A computer science portal for geeks
        </p>
    </div>
</body>
</html>
```

CSS Properties:

CSS properties are the foundation of web design, used to style and control the behaviour of HTML elements. They define how elements look and interact on a webpage.

- Used to control layout, colors, fonts, spacing, and animations on web pages.
- It is essential for making web pages responsive and accessible across devices.
- Help maintain consistency and efficiency in web design and development.
- The list of complete CSS properties is given below.

Properties	Descriptions
@charset Rule	Specifies the character encoding used in the style sheet.
@keyframes Rule	CSS @keyframes specify the animation rule.
@media Rule	Set of styles for different media/devices using the Media Queries.
align-content	It is used to change the behavior of the flex-wrap property.

Properties	Descriptions
align-items	Set the alignment of items inside the flexible container or in the given window.
align-self	Align the selected items in the flexible container.
all	Set all the elements' values to their initial or inherited values.

Color Properties:

The **color property** is used to specify the text color. It accepts color values as color name, HEX, RGB, RGBA, HSL, or HSLA values. This property plays a crucial role in defining text appearance, ensuring readability, and enhancing the overall design aesthetics of web content.

Syntax

```
color: color | initial | inherit;
```

Property Values

- **color:** Specifies the color to apply. It can be a keyword, hex code, RGB/RGBA, HEX, HSL/HSLA value, or global values.
    - **RGB/RGBA Values:** Use the red, green, and blue color model, with optional alpha transparency.
    - **Hexadecimal Values:** Colors represented in hexadecimal format starting with #.
    - **HSL/HSLA Values:** Define colors using hue, saturation, lightness, and optional alpha transparency.
- **Initial:** Sets the color to its default value.
- **Inherit:** Inherits the color value from its parent element.

Using Color Keywords

Named colors are predefined names in CSS for specific colors such as "blue," "green," and "red." It provides a simple way to assign colors without needing to know their RGB or HEX values.

Syntax

```
color: color_name;
```

```
<!DOCTYPE html>
```

```
<html>
<head>
   <style>
      h1 {
         color: black;
      }
      p {
         font-size: 20px;
         color: green;
      }
      .red-color {
         font-size: 20px;
         color: red;
      }
      .blue-color {
         font-size: 20px;
         color: blue;
      }
   </style>
</head>
<body>
   <h1>
      CSS Color Property
   </h1>
   <p>
      GEEKSFORGEEKS: A computer science portal
   </p>
   <p class="red-color">
      GEEKSFORGEEKS: A computer science portal
   </p>
   <p class="blue-color">
      GEEKSFORGEEKS: A computer science portal
   </p>
</body>
</html>
```

RGB/RGBA Value:

Here R stands for Red, G stands for Green, and B stands for Blue. The color will be assigned to the text by using the range of these values. These values range from 0 to 255. And, A stands for Alpha channel. Which represents the opacity or opaque of the color.

Syntax

```
color: RGBA(value, value, value, value);
```

```
<h1 style="color: RGB(0, 0, 0);">
   CSS color property
```

```
</h1>
<p style="color: RGB(0, 150, 0);">
   RGB(0, 150, 0)-This is the code for green color.
</p>
<p style="color: RGB(255, 0, 0);">
   RGB(255, 0, 0)-This is the code for red color.
</p>
<p style="color: RGB(0, 0, 255);">
   RGB(0, 0, 255)-This is the code for blue color.
</p>
```

Opacity Values for Color:

The **opacity** in CSS is the property of an element that describes the transparency of the element. It is the opposite of transparency & represents the degree to which the content will be hidden behind an element.

Try It:

Opacity: 0.5

Opacity: 0.2

Opacity: 0.8

Opacity: 1 (Fully Visible)

Currently Active Property:

Opacity: 0.5

HSL and HSLA Values for Color:

HSL stands for hue, saturation, and lightness.

HSLA color values are an extension of HSL with an Alpha channel (opacity).

HSL Color Values

In HTML, a color can be specified using hue, saturation, and lightness (HSL) in the form:

hsl(*hue, saturation, lightness*)

Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.

Saturation is a percentage value. 0% means a shade of gray, and 100% is the full color.

Lightness is also a percentage value. 0% is black, and 100% is white.

Experiment by mixing the HSL values below:

hsl(0, 100%, 50%)

HSLA Color Values

HSLA color values are an extension of HSL color values, with an Alpha channel - which specifies the opacity for a color.

An HSLA color value is specified with:

hsla(*hue, saturation, lightness, alpha*)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

Experiment by mixing the HSLA values below:

```
<!DOCTYPE html>
<html>
<body>
<h1 style="background-color:hsl(0, 100%, 50%);">hsl(0, 100%, 50%)</h1>
<h1 style="background-color:hsl(0, 80%, 50%);">hsl(0, 80%, 50%)</h1>
<h1 style="background-color:hsl(0, 60%, 50%);">hsl(0, 60%, 50%)</h1>
<h1 style="background-color:hsl(0, 40%, 50%);">hsl(0, 40%, 50%)</h1>
<h1 style="background-color:hsl(0, 20%, 50%);">hsl(0, 20%, 50%)</h1>
```

```
<h1 style="background-color:hsl(0, 0%, 50%);">hsl(0, 0%, 50%)</h1>
<p>With HSL colors, less saturation mean less color. 0% is completely gray.</p>
</body>
</html>
```

Font Properties:

CSS fonts control how text appears on a webpage. With CSS, you can specify various properties like font family, size, weight, style, and line height to create visually appealing and readable typography

```
<html>
<head>
  <style>
    .gfg {
      font-family: "Arial, Helvetica, sans-serif";
      font-size: 60px;
      color: #090;
      text-align: center;
    }
  </style>
</head>
<body>
  <div class="gfg">GeeksforGeeks</div>
</body>
</html>
```

- The code applies a green, 60px Arial font to the text “GeeksforGeeks” and centers it on the page.
- The text is styled using the .gfg class in the HTML.

Key Properties of CSS Fonts

To customize fonts effectively in web design, it’s crucial to understand the main CSS font properties:

- **font-family:** Specifies the font type.
- **font-size:** Determines the size of the text.

- **font-weight:** Adjusts the thickness of the text.
- **font-style:** Controls the slant of the text (e.g., italic).
- **line-height:** Sets the space between lines of text.
- **letter-spacing:** Modifies the space between characters.
- **text-transform:** Controls the capitalization of text.

Line – Height Property:

The **CSS line height property** determines the height of each line of text within an element. It can be set as a specific length, percentage of the font size, or unitless number, affecting spacing between lines for improved readability and aesthetics.

Syntax:

```
line-height: normal|number|length|percentage|initial|inherit;
```

Property values:

Value	Description
normal	Represents the default line-height of the element.
initial	Sets the line height property to its default value.
number	Unitless numbers are multiplied by the element's font size to determine line height.
length	Specifies a fixed line height using a length unit (e.g., px, em).
percentage	Sets the line height as a percentage of the element's font size.

line-height: normal; property

The line-height: normal; property sets the default line height for text, typically ensuring optimal readability and spacing within the element.

line-height: number; property

The line-height: number; property sets the line height to a unitless number multiplied by the current font size, influencing text spacing and readability effectively.

line-height: length; property

The line-height: length; property sets a specific fixed line height using a length unit (e.g., pixels, em), adjusting text spacing within the element accordingly.

line-height: percentage; property

The `line-height: percentage;` property sets the line height as a percentage of the current font size, adjusting text spacing proportionally within the element.

Text Properties:

CSS Text Formatting allows you to control the visual presentation of text on a webpage. From changing fonts to adjusting spacing and alignment, CSS provides powerful properties to enhance the readability and aesthetic of text.

- CSS lets you adjust font properties, text alignment, spacing, and decorations.
- It helps in creating visually appealing text and improving the user experience.
- Various text-related properties can be combined to achieve unique text styles and layouts.

```
<html>
<head>
  <style>
    .initials {
      font-size: 40px;
      font-weight: bold;
      color: #4CAF50;
      text-transform: uppercase;
      font-family: Arial, sans-serif;
    }
  </style>
</head>
<body>
   <p class="initials">A.B.</p>
</body>
</html>
```

In this example

- **Font Size:** The text size is set to 40px, making it large and prominent.

- **Font Weight:** The text is bold, making it stand out more.
- **Color:** The text color is set to a green shade (#4CAF50), giving it a fresh look.
- **Text Transform:** The text is converted to uppercase, so the letters appear in capital letters.
- **Font Family:** The text uses Arial, which is a sans-serif font, for a clean, modern appearance.
    - **CSS Text Formatting Properties**
    - These are the following text formatting properties.

Property	Description
text-color	Sets the color of the text using color name, hex value, or RGB value.
text-align	Specifies horizontal alignment of text in a block or table-cell element.
text-align-last	Sets alignment of last lines occurring in an element.
text-decoration	Decorates text content.
text-decoration-color	Sets color of text decorations like overlines, underlines, and line-throughs.

Border Properties:

Borders in CSS are used to create a visible outline around an element. They can be customized in terms of

- **Width:** The thickness of the border.
- **Style:** The appearance of the border (solid, dashed, dotted, etc.).
- **Color:** The color of the border.

You can try different types of borders here-

Syntax:

```
element {
    border: 1px solid black;
}
```

Example

```
<html>
<head>
   <style>
      .simple-border {
         border: 2px solid black;
         padding: 20px;
         text-align: center;
      }
   </style>
</head>
<body>
   <div class="simple-border">This div has a simple black border.</div>
</body>
</html>
```

CSS Border Properties

CSS provides several properties to control and customize borders:

Property	Description
border-style	Determines the type of border (e.g., solid, dashed, dotted).
border-width	Sets the width of the border (in pixels, points, or other units).

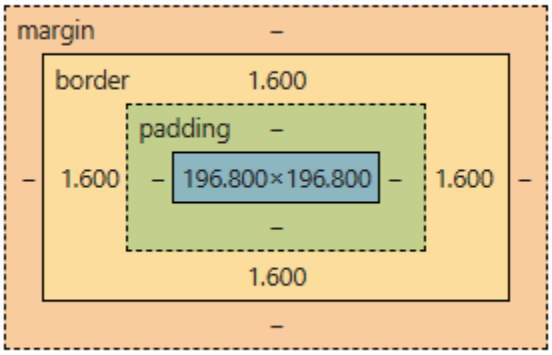
Property	Description
border-color	Specifies the border color.
border-radius	Creates rounded corners for elements.

Element Box:

The CSS Box Model defines how elements are sized, positioned, and rendered on a webpage. When a browser loads an HTML document, it creates a DOM tree and assigns a box to each element. This box calculates the element's dimensions and position relative to its parent or the root <html> element, ensuring accurate layout and spacing.

Box Model Component Layout

- **Content**: The area where text or other content is displayed.
- **Padding**: Space between the content and the element's border.
- **Border**: A frame that wraps around the padding and content.
- **Margin**: Space between the element's border and neighbouring elements.



Padding Property:

Padding is the space between the content and the defined border of an element. Padding means adding spaces inside the element, controlling its internal space, thus affecting its dimensions and appearance.

CSS Padding property is used to create space between the element's content and the element's border. It only affects the content inside the element.

CSS padding is different from CSS margin as the margin is the space between adjacent element borders and padding is the space between content and element's border.

We can independently change the top, bottom, left, and right padding using padding properties.
CSS Padding Properties

CSS provides properties to specify padding for individual sides of an element which are defined as follows:

- **padding-top:** Sets the padding for the top side of the element.
- **padding-right:** Sets the padding for the right side of the element.
- **padding-bottom:** Sets the padding for the bottom side of the element.
- **padding-left:** Sets the padding for the left side of the element.

Padding properties can have the following padding values:

- Length- in cm, px, pt, etc.
- Width- % width of the element.
- inherit- inherit padding from the parent element

Syntax:

```
/* Applying padding to each side individually */
.myDiv {
   padding-top: 80px;
   padding-right: 100px;
   padding-bottom: 50px;
   padding-left: 80px;
}
```

Example

```
<!DOCTYPE html>
<html>
<head>
   <title>Padding Example</title>
   <style>
      body {
         margin: 0;
         padding: 20px;
         width: 50%;
      }
      h2 {
         color: green;
      }
      .myDiv {
         background-color: lightblue;
         border: 2px solid black;
         /* Applying padding to each side individually */
         padding-top: 80px;
         padding-right: 100px;
         padding-bottom: 50px;
         padding-left: 80px;
      }

      .inner {
         background-color: pink;
```

```
        border: 2px solid black;
        width: 70px;
        height: 50px;
        display: flex;
        align-items: center;
        justify-content: center;
      }
  </style>
</head>
<body>
  <div class="myDiv">
    <div class="inner">Pad_Box</div>
  </div>
</body>
</html>
```

Margin Property:

CSS Margin is the space outside an element, separating it from other elements. It adds space around an element, affecting its positioning and creating gaps between other elements.

CSS provides properties to specify the margin for each side of an element individually.

- **margin-top:** Sets the margin space above the element.
- **margin-right:** Sets the margin space to the right of the element.
- **margin-bottom:** Sets the margin space below the element.
- **margin-left:** Sets the margin space to the left of the element.

Margin properties can have the following values:

- Length in cm, px, pt, etc.
- Width % of the element.
- Margin calculated by the browser: auto.

Syntax:

```
.myDiv {
  margin-top: 80px;
  margin-right: 100px;
  margin-bottom: 50px;
  margin-left: 80px;
}
```

Example:

```
<!DOCTYPE html>
<html>
<head>
   <title>Margin Example</title>
   <style>
      body {
         margin: 0;
         padding: 20px;
      }
      h2 {
         color: green;
      }
      .parentDiv {
         background-color: lightblue;
         border: 1px solid black;
         padding: 20px;
         width: 40%;
         /* Applying margin to the child div */
      }
      .childDiv {
         background-color: wheat;
         /* Applying margin to the child div */
         margin-top: 20px;
         margin-right: 20px;
         margin-bottom: 15px;
         margin-left: 30px;
      }
   </style>
</head>
<body>
   <h2>GeeksforGeeks</h2>
   <div class="parentDiv">
      <div class="childDiv">
         Margin box
      </div>
   </div>
</body>
</html>
```