CSS

CSS stands for Cascading Style Sheet.

Used to give styling to the web pages which is structured by the HTML language.

There are three ways in CSS to Style the Web pages

1. Inline CSS
2. Internal CSS
3. External CSS
4. Import.

1. **Inline CSS:** Inline CSS code is written in the opening tag of the HTML element by using style attribute.
   Syntax:

2. **Internal CSS:** Internal CSS code is written in the head section of an HTML element using <style> tag.
   Syntax:
   ```
   <head>
       <style>
           tag_name{
               property: value;
           }
       <style>
   </head>
   ```

3. **External CSS:** External CSS styling can be done by creating a external CSS file with file_name.css extention and providing the link between that CSS file to HTML file by <link> tag in the head section of HTML element.
   Syntax:
   ```
   <head>
       <link rel="stylesheet" href=" file_name.css">
   </head>
   ```

4. **@import**: This Styling can be done by linking one CSS file to another CSS file.
   Syntax: @import url(file_name.css)

Import should be written in very first line of the CSS file which has to be linked.

NOTE: The first priority is inline CSS

But the priorities of the Internal and External CSS varies accordingly depending on the declared position

if, <link rel="stylesheet href="index.css">
<style> </style>
Here, the first priority is for Internal CSS.

if, <style> </style>
<link rel="stylesheet href="index.css">
Here, the first priority is for External CSS.

SELECTOR

Selectors are used to select the particular HTML element and to style them.

There are five types of Selectors

1. Simple Selectors
2. Combinator Selectors
3. Attribute Selectors
4. Pseudo Element Selectors.
5. Pseudo Classes Selectors.

Simple Selectors

Simple selectors style the HTML element in *five* ways:

1. Id Selector.
2. Class Selector.
3. Tagname.
4. Groupname.
5. Universe.

Id selectors:

Id selector targets only individual element in the HTML document.

Prefix symbol used for id selector is # (hash)

Example:

```
#demo{
    color: red;
    background: yellow;
}
```

Class Selector

Class selector targets multiple elements in the HTML document.

Prefix symbol used for id selector is . (dot)

Example:

```
.test{
    color: red;
    background: yellow;
}
```

Tagname Selector

Tagname selector targets the HTML elements by tag name.

There is no symbol used in tagname selectors.

Example:

```
h1{
    color: magenta;
    background: green;
}
h2{
    color: red;
    background: black;
}
h3{
    color: yellow;
    background: orange;
}
```

Grouping Selector

Groupname selector targets a group of HTML elements by tagname separated by commas.

Example:

```
p, div, h4{
    color: magenta;
    background: green;
}
```

Universal Selector

Universal Selectors targets every HMTL element.
The symbol of Universal Selector is *
There is no selector is declared in css.

Example:

```
*{
    color: yellow;
    background: magenta;
}
```

Combinator Selector

Combinator Selectors are used to style the combination of HTML elements.

There are *four* Combinator Selectors:

1. Descendent Selectors
2. Child Selectors
3. Adjacent Sibling Selectors
4. General Sibling Selectors

1. Descendent Selectors: This Selector targets both direct and indirect children of a parent tag. The symbol is (space)

Syntax: parent_tag child_tag{property: value;}

2. Child Selectors: This Selector targets the only the direct children of the parent tag. The symbol is > (greater than)

Syntax: parent_tag > child_tag{property: value;}

3. Adjacent Selectors: This Selector targets the element which is the first sibling of the targeted HTML tag or element.

(or)

This Selector targets the element which is exactly adjacent to the targeted HTML tag or element.

The symbol is + (plus)

Syntax: parent_tag + target_tag{property: value;}

4. General Sibling Selectors: This Selector targets all the siblings of the target HTML tag or element.

(or)

This Selector targets all the adjacent tags of the target HTML tag or element. The symbol is ~ (tilde)

Syntax: parent_tag ~ target_tag{property: value;}

Attribute Selector: Attribute Selectors are used to style the HTML element by targeting the respective attributes and with their values.

Types of Attribute Selector:

1. **tag_name[attribute_name]:** Represents elements with an attribute name of ***attribute_name***.

2. **tag_name [attribute_name= "value"]:** Represents element with an attribute name of ***attribute_name*** whose value is exactly ***value***.

3. **tag_name [attribute_name ~= "value"]:** Represents element with an attribute name of ***attribute_name*** whose value is a *whitespace* separated list of words, one of which is exactly ***value***.

4. **tag_name[attribute_name |= "value"]:** Represents element with an attribute name of ***attribute_name***. Whose value can be exactly value or can begin with value immediately followed by a *hyphen*.
   (also allows the space in " value")

5. **tag_name[attribute_name ^= "value"]:** Represents element with an attribute name of ***attribute_name*** whose value is *prefixed* (preceded) by ***value***.
   (also allows the space in " value")

6. **tag_name[attribute_name $= "value"]:** Represents element with an attribute name of ***attribute_name*** whose value is *suffixed* (followed)by ***value***.
   (also allows the space in " value")

7. **tag_name [attribute_name *= "value"]:** Represents element with an attribute name of ***attribute_name*** whose value contains at least one occurrence of ***value*** (*one character or combination of character*) within the string.

Pseudo Element Selectors.

Pseudo Elements targets the content of the HTML element by the following ways:

1. First Letter
2. First Line
3. Before
4. After
5. Selection
6. Marker

Each pseudo element selectors is declared with the double colon **: :**

First Letter: The first letter styles the very first letter of the content in the targeted HTML element.

Example:
```
p : : first-letter{
    font-size: 30px;
    color: black;
}
```

First Line: The first line styles the very first line of the content in the targeted HTML element.

Example:
```
p : : first-line{
    background-color: black;
    color: white;
}
```

Before: The before pseudo element is used to place the specific content before the targeted HTML element.

Example:
```
p : : before{
    content: “&phone”;
    background-color: black;
    color: white;
}
```

After: The after pseudo element is used to place the specific content after the targeted HTML element.

Example:
```
p : : after{
        content: “Thank You”;
        color: blue;
}
```

Selection: The selection pseudo element is used to style the content when the cursor is dragged on the content (when the content is selected) on the targeted HTML element.

Also copying the text content from the user can also be disabled, by using the property user-select: none.

Example:
```
p : : selection{
        background-color: pink;
        color: magenta;
}
```

Marker: Marker pseudo element is used to style the lists in the HTML document.

Marker is only used to style the list type not the content.

Declaration of selector is not mandatory here.

Background color will not work for Marker.

Example:
```
li : : marker{
        color: red;
        font-size: 30px;
}

(or)

: : marker{
        color: red;
        font-size: 30px;
}
```

Pseudo Classes

Pseudo Class Selector specifies the special state of the content.
The Pseudo Class Selector are:

1. Link
2. Visited
3. Hover
4. Active

Example:

```
a : link {
        color: aqua;
        background-color : red;
        }

a : visited {
        color : green;
            }

a : hover {
        color : yellow;
           }

a : active {
         color : chocolate;
           }
```

BOX Model

Box Model is essentially a box that wraps around every HTML element. Box Model is used to design and layout.

The CSS Box Model consists of Margin, Border, Padding and the actual Content.

Content: *It is the content of the HTML element.*
Padding: *It is the space after the content.*
Border: *It is the Border of the HTML element.*
Margin: *It is the space after the Border of an HTML element.*
Padding and Margin allow up to four values.

Syntax:

```
padding:  10px ;
padding:  10px   20px;
padding:  10px   20px   30px;
padding:  10px   20px   30px   40px;
```

Text Properties

Formatting text properties:

color- changes the text color
Example - color: red;

text-align- Horizontal alignment of the text (here we have right, left, center, justify).
Example- text-align: center;

text-transform- converts the text-letter format(UPPERCASE, lowercase, Capitalize).
Example- text-transform: capitalize;

text-shadow- it allows 4 values they are x-direction y-direction opacity and color
Example- text-shadow: 2px 2px 5px red;

text-decoration- underline, overline, line-through
Example- text-decoration: none;

letter-spacing- to get the space between letters of a text we use it
Example- letter-spacing:2px;

word-spacing- to get the space between words of a text we use it
Example- word-spacing:5px;

text-indent- it'll be saying from where the text should start.
Example- text-indent:2px;

Background Styling Properties

background-color: The background-color property sets the background color of an element.
Syntax: background-color: color|transparent|initial|inherit;

background-image: The background-image property sets one or more background images for an element.
Syntax: background-image: url|none|initial|inherit;

background-origin: The background-origin property specifies the origin position (the background positioning area) of a background image.
Syntax: background-origin: padding-box|border-box|content-box|initial|inherit;

background-position: The background-position property sets the starting position of a background image.
Syntax: background-position: value;

background-repeat: The background-repeat property sets if/how a background image will be repeated. By default, a background-image is repeated both vertically and horizontally.
Syntax: background-repeat: repeat|repeat-x|repeat-y|no-repeat|initial|inherit;

background-size: The background-size property specifies the size of the background images.
Syntax: background-size: auto|length|cover|contain|initial|inherit;

Height and Width properties

The CSS `height` and `width` properties are used to set the height and width of an element.
The height and width properties do not include padding, borders, or margins. It sets the height/width of the area inside the padding, border, and margin of the element.

The `height` and `width` properties may have the following values:

- **`auto`** - This is default. The browser calculates the height and width
- **`length`** - Defines the height/width in px, cm, etc.
- **`%`** - Defines the height/width in percent of the containing block
- **`initial`** - Sets the height/width to its default value
- **`inherit`** - The height/width will be inherited from its parent value

The CSS height and width properties are:

- Height
- Width
- Max-height
- Max-width
- Min-height
- Min-width

Max-height

Sets the maximum height of an element.

If the content is larger than the maximum height, it will overflow. So to handle the content overflow we go for **overflow** property.

If the content is smaller than the maximum height, the **`max-height`** property has no effect.

Note: This prevents the value of the height property from becoming larger than **`max-height`**. The value of the **`max-height`** property overrides the height property.

Max-width

The **max-width** property defines the maximum width of an element.

If the content is larger than the maximum width, it will automatically change the height of the element.

If the content is smaller than the maximum width, the **max-width** property has no effect.

Note: This prevents the value of the **width** property from becoming larger than **max-width**. The value of the **max-width** property overrides the width property.

When the width exceeds the view port of the browser the horizontal scroll bar is added.

Using **max-width** instead, in this situation, will improve the browser's handling of small windows.

Min-height

The **min-height** property defines the minimum height of an element.

If the content is smaller than the minimum height, the minimum height will be applied.

If the content is larger than the minimum height, the **min-height** property has no effect.

Note: This prevents the value of the **height** property from becoming smaller than **min-height**.

Min-width

The **min-width** property defines the minimum width of an element.

If the content is smaller than the minimum width, the minimum width will be applied.

It must be included with the property **display : inline-block**

If the content is larger than the minimum width, the **min-width** property has no effect.

Note: This prevents the value of the **width** property from becoming smaller than **`min-width`**.

Transition Properties

transition-property: Specifies the name of the CSS property to which the transition is applied.

transition-duration: Specifies the duration of the transition process form the old value to the new value.

transition-timing-function: Describes how the intermediate values used during a transition will be calculated.

transition-delay: Defines when the transition will start. It allows a transition to begin execution some period of time from when it is applied.

Transform Properties

The transform property **applies a 2D or 3D transformation to an element**. This property allows you to rotate, scale, move, skew elements**.**

- rotate and skew should be declared in terms of degrees.
- scale should be declared in terms of integer.

Syntax: transform: rotate(30deg);

CSS Animation

CSS Animation is used to animate HTML elements from one style to another style.

Without java script we can give 5% and 10% of functionality to the web pages through CSS animation.

We have following animation property:

- animation: it is an identifier which means name of the animation.

- animation-duration: Specifies the time in seconds to perform the animation only till that specified period of time.

- animation-iteration-count: Specifies the number of times the animation should perform.

- animation-timing-function: Specifies the mode of transform in animation. The values are ease-in, ease-out, ease-in-out.

We can animate the HTML element through keyframes

Syntax:

```
@keyframes animation_name{

        0%{ property:value;}

        25%{ property:value;}

        50%{ property:value;}

        100%{ property:value;}

}
```

This specifies that at 0% at the given animation-duration the particular task should be performed, at 25% at the given animation-duration the particular task should be performed, so on.

Flex and Grid

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

A Flexible Layout must have a parent element with the *display* property set to *flex*.

Ex:
```
div {
  display: flex;
}
```

The flex-container properties are

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

flex-direction: column, row, column-reverse, row-reverse

flex-wrap: wrap, nowrap, wrap-reverse

flex-flow: row wrap

justify-content: center, flex-start, flex-end, space-around, space-between

align-items: center, flex-start, flex-end, stretch, baseline.

align-content: space-between, space-around, stretch, center, flex-start, flex-end