Topic : Introduction to CSS

1. CSS INTRODUCTION

- **CSS** stands for **C**ascading **S**tyle **S**heets
- CSS describes **how HTML elements are to be presented on screen**
- CSS can control the layout of multiple web pages all at once
- Includes adding visuals like **color, fonts, layouts,** etc.

The syntax for CSS is:

```
selector {
    properties;
}
```

Each property ends with a semicolon.
Each property includes a CSS property name and a value, separated by a colon.

Below is an example, to see how CSS modifies the HTML code:

```
<h1>Welcome to Coding Ninjas!</h1>
<h2>Where coding is a way of life..</h2>
```

Now, adding CSS to the above HTML code:

```
h1 { font-family: monospace; }
h2 { color: blue; }
```

The code snippet would now look like this:

Welcome to Coding Ninjas!

Where coding is a way of life..

2. CSS COMMENTS

Just like HTML, CSS also contains comments and can be used to define the various sections of HTML element styles that can help in changes at later stage, easily. You can also describe where certain generic style(s) can be used.

A CSS comment *starts with* **/*** *and ends with* ***/**. Comments can also span multiple lines.

3. ADDING CSS TO HTML PAGE

The browser formats the HTML document based upon the information in the stylesheet. The browser access the stylesheets from the HTML document itself. There are 3 ways to add CSS styles in your document. Each of them can contain multiple properties:

- Inline styles
- Internal styles
- External styles

Cascading order defines which style will be applied to element when multiple styles are used.
Cascading order priority: ***Inline > (internal ≅ external) > browser default.***

The internal and external CSS are treated equally by the browser. But the order of their definition decides which one's property will get preference.
If *link to external CSS is defined before the internal CSS*, then properties of internal CSS will get the preference, i.e ***internal CSS > external CSS***.
If *link to external CSS is defined after the internal CSS*, then properties of external CSS will get preference over internal CSS, i.e. ***external CSS > internal CSS***.

3.1. Inline styles

Inline stylesheet is applied directly to our HTML code using the **style** attribute. The inline stylesheet syntax contains properties written inside the style attribute.
Multiple properties can be defined at a time.
An inline style is used to apply a **unique style** for a single element.

You can use inline styles like this:

```
<p style="color:blue;font-size:40px">Inline CSS</p>
```

3.2. Internal styles

An **internal** or **in-page** stylesheet holds the CSS code for the web page. The internal stylesheet contains the styles for that HTML document only. They cannot be reused.
Internal CSS can be applied using **<style>** tag inside the **<head>** tag.

You can use internal styles like this:

```
<style>
    h1 { color:blue; }
    h2 { color:red; }
    p { color:green; }
</style>
```

3.3. External styles

With an **external** stylesheet, you can change the look of an entire website by changing just one file. The syntax is similar to internal stylesheets, but it is applied using a complete different **CSS file**. It is saved with **'.css'** extension. Eg. *'styles.css'*.

To use external stylesheet, a reference is provided to file inside the **<link>** element:

```
<link rel="stylesheet" type="text/css" href="styles.css">
```

The **rel** defines the *relationship with the linked document*(here, favicon).
The **href** defines the *location of the lined document*(here, favicon).
The **type** defines the *media type of the linked document*(here, favicon)

***NOTE**: The <link> element goes inside the <head> section. The 'styles.css' contains the CSS syntax code only.*

4. SELECTORS

Selectors point to the HTML element which we want to style. We use selectors in internal and external stylesheets. There are mainly 3 types of selectors that are used to apply styles:

- Element selector
- Class selector
- Id selector

Specificity here defines which style will be applied when same multiple styles are applied to an element. If the specificity is same, then the latest rule is applied.
*Specificity order: **inline > id selector > class selector > tag selector > browser default***.

***NOTE:** If same property is defined inside the same type of selector, then the property which is defined at the last, will be used by the browser.*

4.1. Element Selector

The element selector selects all elements with the specified element name. This will select all the elements in the HTML document. But most of the time we don't want this. So, to apply styles to only some elements we need to use some restrictions. We will look into this later in this section only.

Syntax: `element { css declarations; }`

Eg., applying style to h2 tag like this:

```
<h1>Blue Color</h1>
```

and applying css like this:

```
h1 { color: blue; }
```

will show on the browser like this:

4.2. Class Selector

The class selector selects multiple elements with a specific class attribute. To select elements with a specific class, write a period(.) character, followed by the name of the class.

Syntax: `.class-name {css declarations; }`

To use the class selector, the **class attribute** is used in the element's opening tag. The value of the class attribute contains the name of the class. There can be **multiple classes** added to the tag by giving a space in between.

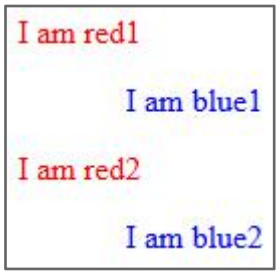
Eg., defining the classes in the HTML file like this:

```
<p class="red">I am red1</p>
<p class="blue right">I am blue1</p>
<p class="red">I am red2</p>
<p class="blue right">I am blue2</p>
```

and applying css like this:

```
.red { color:red; }
.blue { color:blue; }
.right { text-align:right; }
```

will show on the browser like this:



4.3. Id Selector

The id selector selects only one element with a specific id attribute. To select element with a specific id, write a hash(#) character, followed by the name of the id.

Syntax: `#class-name {css declarations; }`

To use the id selector, the **id attribute** is used in the element's opening tag. The value of the id attribute contains the name of the id. There can only be **one id** in the tag. The id is **unique** in a HTML page. If another element is given the same id, the styles would not be applied by the browser.

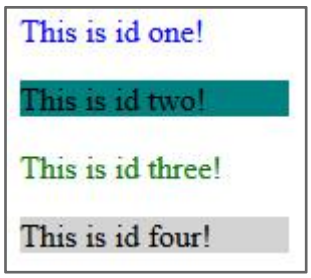
Eg., defining the ids in the HTML file like this:

```
<p id="one">This is id one!</p>
<p id="two">This is id two!</p>
<p id="three">This is id three!</p>
<p id="four">This is id four!</p>
```

and applying css like this:

```
#one { color: blue; }
#two { background-color: teal; }
#three { color: green; }
#four { background-color: lightgrey; }
```

will show on the browser like this:



4.4. Grouping Selectors

It is most of the time that we use same css for multiple elements and we can't use too much classes. This would bring classes without any meaning in the scenario and would become difficult to manage.
So, CSS provides with a grouping feature where you can apply CSS rules to multiple elements with the use of combination of either tag, class or id. For grouping, we use **comma between the different selectors**.

These are the few examples how you can use grouping:

- `p, .class-name { CSS properties }` - apply styles to 'para' and element with class as 'class-name'
- `#id1, #id2, span { CSS properties }` - apply styles to 'span' and elements with ids as 'id1'and 'id2'
- `.class-name, #id1, div { CSS properties }` - apply styles to 'div' and elements with class 'class-name' and id as 'id1'

4.5. Nesting Selectors

There are times when we want to target elements inside a specific section of HTML page. So, instead of the using the classes we can use nesting that works like a hierarchy and is easy to understand.

To use nesting, use a **space between the selectors** such that the **sequence represents a hierarchy, starting from top**.

These are the few examples:

- `.class-name span { CSS declarations }` - apply styles to only those 'span', which are inside the element with class 'class-name'
- `#id1 .class-name span { CSS declarations }` - apply styles to only those 'span', which are inside the element with class 'class-name' and 'class-name' is inside the element with id 'id1'

4.6. Chaining Selectors

There are a times when we have(or want to have) same class for multiple elements and we want to apply style/s to one/some of them. Then we can use chaining

selectors.

To use chaining we use the combination of selectors without any space in between them.

Eg., we have a class 'header-style' applied to every heading. We can apply different styles to them like this:

```
h1.header-style { CSS declarations }
h2.header-style { CSS declarations }
```