

Styles and CSS

- Styles are set of attributes defined for HTML elements to make the presentation more interactive and responsive.
- HTML elements have attributes but limited in functionality, style attributes make HTML more effective. They extend HTML element.
- Styles can be defined in 3 ways
 - Inline Styles
 - Embedded Styles
 - External Style Sheet

Inline Styles:

- The styles are defined for elements by using “style” attribute.
- Every element has its own individual styles.
- The styles defined for one element can't be re-used for other elements.
- These styles are faster in rendering as they are local to element.

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Styles</title>
```

```
  </head>
```

```
<body>
    <h2 style="background-color: green; color:white; text-align: center;">HTML</h2>
    <h2>CSS</h2>
    <h2>JavaScript</h2>
</body>
</html>
```

Embedded Styles:

- Styles are defined in page by using <style> element.
- You configure in head or body section.
- You can keep all your styles at one location and use across various elements.
- It is good for reusing styles.
- Slower than inline.

Ex:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Styles</title>
        <style>
            h2 {
                background-color: green;
```

```
        color: white;
        text-align: center;
    }
</style>
</head>
<body>
    <h2>HTML</h2>
    <h2>CSS</h2>
    <h2>JavaScript</h2>
</body>
</html>
```

FAQ: Where to embed the styles, in head or in body?

A. If you want to configure a set of style, which are loaded into browser memory, and used later by the elements according to the requirement then keep in <head> section.

If you want to configure a set of styles, which are applied directly on body load then better define them in <body>.

Ex:

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

```
<title>Styles</title>
<style>
    .heading {
        background-color: green;
        color: white;
        text-align: center;
    }
</style>
</head>
<body>
    <style>
        body {
            background-color: lightgreen;
        }
    </style>
    <h2 class="heading">HTML</h2>
    <h2>CSS</h2>
    <h2>JavaScript</h2>
</body>
</html>
```

FAQ: What is MIME type for Styles?

- MIME type defines the type content present in element.
- The MIME type is used by browser to understand the type of content.
- Styles MIME type is "text/css"

Syntax:

```
<style type="text/css"> </style>
```

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Styles</title>
```

```
<style type="text/css">
```

```
  .heading {
```

```
    background-color: green;
```

```
    color: white;
```

```
    text-align: center;
```

```
  }
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<style type="text/css">
```

```
  body {
```

```
    background-color: lightgreen;
```

```
    }  
</style>  
<h2 class="heading">HTML</h2>  
<h2>CSS</h2>  
<h2>JavaScript</h2>  
</body>  
</html>
```

Styles from External File:

- The styles are maintained in a separate style sheet that have the extension “.css”
- You can link the style sheet to any HTML page.
- Styles are accessible across several pages.
- If you use external style sheet then number of requests for page will increase and also the page load time.

Ex:

- Add a new folder “Styles”
- Add a new style sheet into folder “effects.css”
- Add effects into style sheet

```
h2 {  
    background-color:blue;  
    color:white;  
    text-align: center;  
}
```
- Link the stylesheet to your web page.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Styles</title>
    <link type="text/css" rel="stylesheet"
href="../Styles/effects.css">
  </head>
  <body>
    <h2>HTML</h2>
    <h2>CSS</h2>
    <h2>JavaScript</h2>
  </body>
</html>
```

FAQ: What is CDN?

- CDN is Content Distribution Network
- We can maintain all style sheets in a repository server [SandBox]
- We can directly connect and access the style sheet from repository server instead of download into project.

Ex:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Styles</title>
```

```
<link type="text/css" rel="stylesheet"
href="http://127.0.0.1/cdn/styles/effects.css">
</head>
<body>
  <h2>HTML</h2>
  <h2>CSS</h2>
  <h2>JavaScript</h2>
</body>
</html>
```

Minification of CSS

- Minification is the process of compressing CSS.
- It is always recommended to Minify and use the CSS for production. [Live]
- CSS original file will occupy more space, we have to use them for development but not for production.

FAQ: What is “media” type for styles?

- Media specifies the styles target, which can be for Print, Screen, Speech.
- “media” is an attribute used to configure styles targeting different sources like printer, screen, audio out etc.
- You can configure “media” attribute for <style> or <link> element.
- Media values can be
 - All

- Print
 - Screen
 - Speech
- You can configure effects which will work for specific media.

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style type="text/css" media="screen">
```

```
body {
```

```
    border:2px solid darkcyan;
```

```
    padding:20px;
```

```
}
```

```
h1{
```

```
    border:1px solid red;
```

```
    box-shadow: 2px 3px 4px red;
```

```
    text-align: center;
```

```
    padding: 10px;
```

```
}
```

```
</style>
```

```
<style type="text/css" media="print">
```

```
h1{
```

```

        border:1px solid red;
        text-align: center;
        padding: 10px;
    }
</style>
</head>
<body>
    <h1>Amazon Shopping</h1>
</body>
</html>

```

Writing Styles for Elements

- If you are writing inline style for any element

```
<div style="stylePropertyName:value; stylePropertyName:value"> </div>
```
- If you are writing styles embedded or in external file

```
<style>
    selector
    {
        stylePropertyName: value;
        stylePropertyName: value;
    }
</style>
```
- Selector is used to define the target where the given styles need to apply.

- CSS can use various types of selectors
- The primary selectors used in styles are:
 - Type Selector
 - ID Selector
 - Class Selector

Type Selector

- Type selector refers to HTML element tag name [Image , Bold].
- The given styles will be applied to specified tag where ever it is used in page.
- It will apply effects to every occurrence of the tag in page. You can't disable for any specific.

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <style>
```

```
      h2
```

```
      {
```

```
        background-color:red;
```

```
        color:white;
```

```
        text-align:center;
```

```
      }
```

```
</style>
</head>
<body>
  <h2>HTML</h2>
  <p>It is a markup language.</p>
  <h2>CSS</h2>
  <p>Defines styles of HTML.</p>
  <h2>JavaScript</h2>
  <p>Handles client side interactions.</p>
</body>
</html>
```

ID Selector

- Every element can be defined with ID.
- You can use ID to access the element and apply effects.
- You can choose to which element you want the effects.
- Element is defined with ID

```
<div id="effects"> </div>
```
- You can access the ID in styles by using “#” reference

```
<style>
#effects
{
}
</style>
```
- Every tag can use only one ID reference.

- If you have configured multiple categories of styles with ID selector and want to use for specific tag, then it is not possible to define all effects to one element.

Ex:

```
<!DOCTYPE html>

<html>

  <head>

    <style>

      #textEffects {

        text-align: center;

        color:yellow;

      }

      #bgEffects {

        background-color: red;

      }

    </style>

  </head>

  <body>

    <h2 id="textEffects">HTML</h2>

    <p>It is a markup language.</p>

    <h2 id="bgEffects">CSS</h2>

    <p>Defines styles of HTML.</p>
```

```
<h2>JavaScript</h2>
<p>Handles client side interactions.</p>
</body>
</html>
```

Class Selector

- A class selector is defined by using “.”
- Class is accessed and applied to element by using “class” attribute.
- Every tag can implement multiple classes.
- Multiple classes are specified with space.

```
<style>
```

```
.cssClassName
```

```
{
```

```
}
```

```
</style>
```

```
<div class="cssClassName1 cssClassName2"> </div>
```

- The CSS selectors are further classified into various groups based on behaviour
 - **Combinators / Rational Selectors**
 - **Attribute Selectors**
 - **Pseudo Selectors**
 - **Structural Pseudo Selectors**

Rational or Combinators

- These selector default with parent and child elements as well as with elements that have relation.
- Relation like adjacent, below, above, before, after, first, last etc..

Selector	Description
Descendent Selector	<p>Targets all tags under specified parent. It includes any level hierarchy.</p> <p>It defines the parent element and the child element by using space.</p> <p>Syntax:</p> <pre>parentElement childElement {</pre> <pre>}</pre> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <style> ol li { color: red; } div p { color:green; } </style> </head> <body> <h2>Web Technologies</h2> </pre>

```
<li>HTML
  <ol>
    <li>Void Elements</li>
    <li>Normal Elements</li>
  </ol>
</li>
<li>CSS</li>
<li>JavaScript</li>
</ol>
<div>
```

```
<blockquote>Blockquote...</blockquote>
  <p>Para-1</p>
  <div>
    <p>Para-2</p>
  </div>
</div>
<p>Para-3</p>
</body>
</html>
```

Output:

Web Technologies

1. HTML
 1. Void Elements
 2. Normal Elements
2. CSS
3. JavaScript

Blockquote...

Para-1

Para-2

Para-3

Child Selector	<p>It applies effects only to the direct child of parent element.</p> <p>Syntax:</p> <pre>Parent > child {</pre> <pre>}</pre> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <style> div>p { color:red; } </style> </head> <body> <div> <p>Para-1</p> </div> <div> <p>Para-2</p> </div> </body> </html></pre> <p>Output:</p> <p>Para-1</p> <p>Para-2</p>

Adjacent Sibling	<p>It defines effects to an element which is specified immediately after current element.</p> <p>It is not parent and child, it is one below another.</p> <p>It will apply only to the first adjacent element.</p> <p>Syntax:</p> <p>FirstElement + adjacentElement</p> <pre>{ }</pre> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <style> h2+p { color:red; } </style> </head> <body> <h2>HTML Elements</h2> <p>Para-1</p> <p>Para-2</p> <p>Para-3</p> <p>Para-4</p> </body> </html></pre>

	<p>Output:</p> <p>HTML Elements</p> <p>Para-1</p> <p>Para-2</p> <p>Para-3</p> <p>Para-4</p>
General Sibling	<p>It defines effects to all elements which are specified after the current element.</p> <p>Syntax:</p> <p>FirstElement ~ AdjacentElements</p> <pre>{ }</pre> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <style> h2~p { color:red; } </style> </head> <body> <h2>HTML Elements</h2> <p>Para-1</p> <p>Para-2</p> <p>Para-3</p> <p>Para-4</p> </body></pre>

	</html> Output: HTML Elements Para-1 Para-2 Para-3 Para-4
--	---

Attribute Selectors

- Several elements in HTML are presented by using attribute of tag.

<input type="button">

<input type="radio">

- "type" is attribute.
- We have to apply effects based on attribute and value.

Syntax:

tagName["attribute"] { }

tagName["attribute=value"] { }

Ex: Attribute and Value

<!DOCTYPE html>

<html>

<head>

```
<style>
  input[type="button"] {
    background-color: lightgreen;
  }
  input[type="password"] {
    background-color: lightpink;
  }
</style>
</head>
<body>
  <form>
    <dl>
      <dt>Name</dt>
      <dd><input type="text"></dd>
      <dt>Password</dt>
      <dd><input type="password"></dd>
    </dl>
    <input type="button" value="Register">
  </form>
</body>
```

</html>

Output:

Name

Password

Register

Ex: Only Attribute

<!DOCTYPE html>

<html>

<head>

<style>

p[id] {

color: red;

}

</style>

</head>

<body>

<p>Para-1</p>

<p id="p2">Para-2</p>

<p>Para-3</p>

```
<p id="p4">Para-4</p>
</body>
</html>
```

Output:

Para-1
Para-2
Para-3
Para-4

- Attribute selectors can be defined with conditions.
- Effects are applied only to attribute that match the given condition.

Condition	Purpose
[attribute="val"]	<p>Equal specifies that it should be exact match.</p> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <style> p[class="Effect"] { color:red; } </style> </head></pre>

	<pre> <body> <p class="paraEffect">Para-1</p> <p class="para">Para- 2</p> <p class="Effectpara">Para-3</p> <p class="Effect">Para- 4</p> </body> </html> </pre> <p> Para-1 Para-2 Para-3 Para-4 </p>
[attribute^="val"]	<p>It refers the value starting with specified term.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <style> p[class^="Effect"] { color:red; } </style> </head> <body> </pre>

	<pre> <p class="paraEffect">Para-1</p> <p class="para">Para- 2</p> <p class="Effectpara">Para-3</p> <p class="Effect">Para- 4</p> </body> </html> </pre> <p> Para-1 Para-2 Para-3 Para-4 </p>
[attribute\$="val"]	<p>It specifies that the value ending with given term.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <style> p[class\$="Effect"] { color:red; } </style> </head> <body> </pre>

	<pre> <p class="paraEffect">Para-1</p> <p class="para">Para- 2</p> <p class="Effectpara">Para-3</p> <p class="Effect">Para- 4</p> </body> </html> </pre> <p> Para-1 Para-2 Para-3 Para-4 </p>
[attribute*="val"]	<p>It matches the term at any location.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <style> p[class*="Effect"] { color:red; } </style> </head> <body> </pre>

	<pre> <p class="paraEffect">Para-1</p> <p class="para">Para- 2</p> <p class="Effectpara">Para-3</p> <p class="Effect">Para- 4</p> </body> </html> </pre> <p> Para-1 Para-2 Para-3 Para-4 </p>
[attribute ="val"]	<p>Name starts with specified term and separated with “-“.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <style> p[class ="Effect"] { color:red; } </style> </head> <body> </pre>

	<pre> <p class="para- Effect">Para-1</p> <p class="para">Para- 2</p> <p class="Effect- para">Para-3</p> <p class="Effect">Para- 4</p> </body> </html> </pre> <p>Output:</p> <p>Para-1 Para-2 Para-3 Para-4</p>
[attribute~="val"]	<p>Name start with specified term and contain blank space.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <style> p[class~="Effect"] { color:red; } </style> </head> <body> </pre>

	<pre> <p class="para- Effect">Para-1</p> <p class="Effect para">Para-2</p> <p class="Effect- para">Para-3</p> <p class="Effect">Para- 4</p> </body> </html> </pre> <p> Para-1 Para-2 Para-3 Para-4 </p>
--	--

Dynamic Pseudo-Classes

- Dynamic indicates that the effect can change according to state and situation.
- Pseudo indicates that it is not referring to exactly the element which is having the same name as selector name.
- The selector name and the element it effects may differ.

Syntax:

link - not <link> element, it refers to <a>

class/Id/type: pseudoClass {
}

Selector	Description
:link	Specifies effect for Hyperlink.
:visited	It defines effects for visited links.
:hover	It defines effects when mouse pointer is over element.
:active	It defines effects when link is in active state.
:focus	<p>It defines effects when element get focus.</p> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <style> .txtName+span { display: none; } .txtName:focus+span { display: inline; } </style> </head> <body></pre>

	<pre> <div> <label>Name</label> <div> <input class="txtName" type="text"> Name 4 Chars </div> </div> </body> </html> </pre> <p>Output: Without focus on text box</p> <p>Name</p> <input data-bbox="571 1010 711 1055" type="text"/> <p>With focus on text box</p> <p>Name</p> <input data-bbox="571 1196 711 1240" type="text"/> Name 4 Chars
--	---

Syntax:

Element:Link { }

#heading:hover { }

.txtName:focus { }

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <style>
```

```
      .txtName+span {
```

```
        display: none;
```

```
      }
```

```
      .txtName:focus+span {
```

```
        display: inline;
```

```
      }
```

```
      input:focus {
```

```
        border:2px solid darkcyan;
```

```
        box-shadow: 2px 2px 3px darkcyan;
```

```
      }
```

```
      a{
```

```
        text-decoration: none;
```

```
      }
```

```
      a:hover {
```

```
        text-decoration: underline;
```

```
      }
```

```
      a:active {
```



```
        color:red;
    }
    a:visited {
        color:green
    }
    a:link {
        color: gray;
    }
</style>
</head>
<body>
    <div>
        <label>Name</label>
        <div>
            <input class="txtName" type="text">
            <span>Name 4 Chars</span>
        </div>
    </div>
    <div>
        <a href="home.html">Home</a>
        <span>|</span>
        <a href="http://www.flipkart.com">Flipkart</a>
```

```
</div>
</body>
</html>
```

Target pseudo class

Selector	Description
:target	<ul style="list-style-type: none">- It defines effects to any element when it becomes target of a link.- You can implement in intra document navigation.

Ex:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Target</title>
    <style>
      ul {
        list-style: none;
        display: flex;
      }
      li {
        margin-left: 50px;
```

```
        border:2px solid darkblue;
        padding: 10px;
        width: 200px;
        text-align: center;
        border-radius: 10px;
    }

    .group {
        border:2px solid darkgreen;
        background-color: lightgreen;
        color:black;
        margin-top: 20px;
        padding: 10px;
    }

    .group:target {
        background-color: black;
        color:white;
    }
</style>
</head>
<body>
    <header>
        <nav>
```

```
<ul>
  <li><a href="#html">HTML</a></li>
  <li><a href="#css">CSS</a></li>
  <li><a href="#js">JavaScript</a></li>
</ul>
</nav>
</header>
<section>
  <div id="html" class="group">
    <h3>HTML</h3>
    <p>It is a markup language.</p>
  </div>
  <div id="css" class="group">
    <h3>CSS</h3>
    <p>It is to define styles</p>
  </div>
  <div id="js" class="group">
    <h3>JavaScript</h3>
    <p>It is a language.</p>
  </div>
</section>
</body>
```

</html>

The UI element state pseudo-classes

- Element state indicates the state of element like enabled, disabled, readonly, checked.

Selector	Description
:enabled	It defines effects when element is enabled.
:disabled	<p>It defines effects when element is disabled.</p> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <title>State</title> <style> input:read-only { background-color: gainsboro; color: gray; } button:disabled { cursor: not-allowed; } button:enabled { cursor: grab; } </style></pre>

	<pre> </head> <body> <fieldset> <legend>User Name</legend> <div> <input readonly type="text" value="John"> <button disabled>Submit</button> </div> </fieldset> </body> </html> </pre>
:read-only	<p>It defines effects when element is set to read-only.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>State</title> <style> input:read-only { background-color: gainsboro; color: gray; } </style> </head> </pre>

	<pre> <body> <fieldset> <legend>User Name</legend> <div> <input readonly type="text" value="John"> <button>Submit</button> </div> </fieldset> </body> </html> </pre>
:checked	<p>It defines effects when element is checked.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>State</title> <style> input[type="checkbox"]+span { color:red; } input[type="checkbox"]:checked+span { color: green; } </style> </head> </pre>

	<pre> <body> <fieldset> <legend>Terms of Service</legend> <textarea rows="4" cols="40"> Read our terms and conditions.. </textarea> <div> <input type="checkbox"> I Accept </div> </fieldset> </body> </html> </pre>
--	---

The UI element validation state pseudo classes:

- HTML 5 provides pre-defined form validations like require, email, url, pattern etc.
- CSS can use HTML 5 validations to verify the state valid or not and can apply effects.

Selector	Description
:valid	It defines effects for element if its value is valid against the validation defined. Validation can be verified by using: <ul style="list-style-type: none"> - Minlength

	<ul style="list-style-type: none"> - Maxlength - Required - Pattern - Email - URL etc.
:invalid	<p>It defines effect for element when it is invalid.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>State</title> <style> #txtName:valid+span { display: none; } #txtName:invalid+span { display: inline; } #txtName:valid { border:2px solid green; box-shadow: 2px 2px 3px green; } #txtName:invalid { border:2px solid red; box-shadow: 2px 2px 3px red; } </style> </head> </pre>

	<pre> <body> <div class="form-group"> <label>User Name</label> <div> <input id="txtName" type="text" minlength="4"> Name too short </div> </div> </body> </html> </pre>
:in-range	<p>It defines effects for element when input value is within the specified range.</p>
:out-of-range	<p>It defines effects for element when input value is out of given range.</p> <p>Range is verified with “min and max” values defined for input element.</p> <p>EX:</p> <pre> <!DOCTYPE html> <html> <head> <title>State</title> <style> input:in-range { border: 2px solid green; box-shadow: 2px 3px 4px green; } input:out-of-range { border: 2px solid red; </pre>

	<pre> box-shadow: 2px 3px 4px red; } </style> </head> <body> <div class="form-group"> <label>Age</label> <div> <input type="number" min="16" max="35"> </div> </div> </body> </html> </pre>
:required	It defines effects to element when it verified with required error. It is not validating required, It is just verifying whether the required defined or not.
:optional	If it is not defined with required validation then it is treated as optional.

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>State</title>
```

```
<style>
```

```
    input:in-range {
```

```
        border: 2px solid green;
        box-shadow: 2px 3px 4px green;
    }

    input:out-of-range {
        border: 2px solid red;
        box-shadow: 2px 3px 4px red;
    }

    .form-group {
        margin-top: 20px;
    }

    #txtName:optional+div {
        display: none;
    }

    #txtName:required+div {
        display: block;
        color:red;
    }

    #txtName:valid+div {
        display: none;
    }

</style>
</head>
```

```
<body>
  <div class="form-group">
    <label>Age</label>
    <div>
      <input type="number" min="16" max="35">
    </div>
  </div>
  <div class="form-group">
    <label>Name</label>
    <div>
      <input required id="txtName" type="text">
      <div>Name required</div>
    </div>
  </div>
</body>
</html>
```

Structural Pseudo Selector:

- You can target your effects based on the position of element in parent and child hierarchy.

Selector	Description
:first-child	It defines effects only for first child element.

:last-child	It defines effects only for last child element.
:nth-child(LevelNumber)	<p>It defines effects only to specific child element that occurs at given level.</p> <p>Level number starts with 1.</p> <p>Index number starts with 0.</p> <p>You can also define the pre-set values like 'even & odd' to apply effects based on even and odd occurrences.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>Structure</title> <style> ol > li:first-child { color:red; } ol > li:last-child { color:blue; } ol > li:nth-child(3){ color: green; font-size: 30px; } </style> </head> <body> </pre>

```
<ol>
  <li>Item-1</li>
  <li>Item-2</li>
  <li>Item-3</li>
  <li>Item-4</li>
  <li>Item-5</li>
</ol>
</body>
</html>
```

Ex: Even and Odd occurrence

```
<!DOCTYPE html>
<html>
  <head>
    <title>Odd Even</title>
    <style>
      thead > tr {
        background-color:
darkcyan;
        color:white;
      }
      tbody > tr:nth-child(even)
{
        background-color:
lightcyan;
      }
      tbody > tr:nth-child(odd){
        background-color:
lightgreen;
      }
    </style>
```

```
</head>
<body>
  <table border="1"
width="400">
    <thead>
      <tr>
        <th>Name</th>
        <th>Price</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>TV</td>
        <td>45000.55</td>
      </tr>
      <tr>
        <td>Mobile</td>
        <td>41000.22</td>
      </tr>
      <tr>
        <td>Nike</td>
        <td>5200.33</td>
      </tr>
      <tr>
        <td>Shirt</td>
        <td>4100.33</td>
      </tr>
    </tbody>
  </table>
</body>
</html>
```


<p>:nth-of-type(LevelNumber[n])</p> <p>:nth-of-type(2n)</p> <p>:nth-of-type(2n+startNumber)</p>	<p>It will repeat the effect for every nth occurrence.</p> <p>It will repeat the effect for every 2nd occurrence.</p> <p>It will start with specific level.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>Structure</title> <style> ol > li:nth-of-type(2n+1){ color:red; } </style> </head> <body> Item-1 Item-2 Item-3 Item-4 Item-5 </body> </html> </pre>
<p>:nth-last-of-type(n)</p>	<p>It will apply effect for every nth occurrence from bottom.</p> <p>Ex:</p>

	<pre> <!DOCTYPE html> <html> <head> <title>Structure</title> <style> ol > li:nth-last-of- type(2n+1){ color:red; } </style> </head> <body> Item-1 Item-2 Item-3 Item-4 Item-5 Item-6 Item-7 Item-8 </body> </html> </pre>
:nth-last-child(n)	<p>It will apply from bottom without repeating.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> </pre>

	<pre> <title>Structure</title> <style> ol > li:nth-last-child(2){ color:red; } </style> </head> <body> Item-1 Item-2 Item-3 Item-4 Item-5 Item-6 Item-7 Item-8 </body> </html> </pre>
:root	<p>It refers to root of document, which is 'body'</p> <p>Ex:</p> <pre> :root { font-family:Arial; } </pre>
:empty	<p>If any element is empty, without any content then its will define the given effects.</p>

You can configure for containers like `<div>`, ``, `<td>`, `<dd>`, `<p>` etc.

Ex:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Odd Even</title>
    <style>
      tbody > tr {
        background-color:
darkcyan;
        color:white;
      }
      tbody > tr:nth-child(even)
{
        background-color:
lightcyan;
      }
      tbody > tr:nth-child(odd){
        background-color:
lightgreen;
      }
      tbody > tr > td:empty {
        background-color: red;
      }
    </style>
  </head>
  <body>
```

```
<table border="1"
width="400">
  <thead>
    <tr>
      <th>Name</th>
      <th>Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>TV</td>
      <td>45000.55</td>
    </tr>
    <tr>
      <td>Mobile</td>
      <td>41000.22</td>
    </tr>
    <tr>
      <td>Nike</td>
      <td></td>
    </tr>
    <tr>
      <td>Shirt</td>
      <td>4100.33</td>
    </tr>
  </tbody>
</table>
</body>
</html>
```

Pseudo-Element Selectors:

Selector	Description
::first-line	Effects for first line in paragraph.
::first-letter	<p>Effects for first character.</p> <p>Ex:</p> <pre><!DOCTYPE html> <html> <head> <title>Element Selectors</title> <style> p::first-letter { font-family: Arial; font-size: 30px; color:red; } p::first-line { color:red; } </style> </head> <body> <p>Depending on how you obtained the Windows software, this is a license agreement between (i) you and the device manufacturer or software installer that distributes the software with your device; or (ii) you and Microsoft Corporation (or, based on where you live or, if a business, where your principal place of business is located, one of its affiliates) if you acquired the software from a retailer. Microsoft is the</pre>

	<p>device manufacturer for devices produced by Microsoft or one of its affiliates, and Microsoft is the retailer if you acquired the software directly from Microsoft. Note that if you are a volume license customer, use of this software is subject to your volume license agreement rather than this agreement.</p> <pre> </p> </body> </html> </pre>
::before	Effect or content to add before the current element.
::after	<p>Effect or content to add after the current element.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>Before After</title> <style> ul { display: flex; list-style: none; } li::before { content: "-->"; } li:first-child::before { content: ""; } </style> </head> <body> Item 1 Item 2 Item 3 </body> </html> </pre>

	<pre> } </style> </head> <body> Site Map <nav> Home About Contact Login </nav> </body> </html> </pre>
::placeholder	It will apply effects for placeholder.
::selection	<p>It will apply effects for selection.</p> <p>Ex:</p> <pre> <!DOCTYPE html> <html> <head> <title>Languages</title> <style> * { font-style: italic; } input::placeholder { color:lightgreen; } p::selection { background-color: yellow; } </style> </head> <body> <p>Languages</p> <input type="text" value="Placeholder" /> </body> </html> </pre>

	<pre> } </style> </head> <body> <p>Some content.. select and see..</p> <dl> <dt>Name</dt> <dd><input placeholder="Name 4 chars" type="text"></dd> <dt>Password</dt> <dd><input disabled type="password"></dd> <dt>Mobile</dt> <dd><input required type="text"></dd> </dl> </body> </html> </pre>
--	--

Language Selector:

- It defines effects based on lang configured for element.
- If you page is multi lingual then you can define effects to content based on specific language.

“:lang()”

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
    <head>
```

```
<title>Languages</title>
<style>
  p:lang(en){
    font-style: italic;
  }
</style>
</head>
<body>
  <h2>Language Selector</h2>
  <p>Some Text</p>
  <p lang="en">English US</p>
</body>
</html>
```

Negation Selector

- It is used to define effects for the elements which are not matching with specified criteria.
- The negation selector is defined using “:not()”
- It will ignore effects for specific element and apply for other.

Ex:

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

```
<head>
  <title>Languages</title>
  <style>
    p:not(#effects){
      color:red;
    }
  </style>
</head>
<body>

  <p>Para-1</p>
  <p id="effects">Para-2</p>
  <p>Para-3</p>
  <p id="styles">Para-4</p>
  <p id="effects">Para-5</p>
</body>
</html>
```

- You can also configure for properties.

Ex:

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

```
<head>
  <title>Languages</title>
  <style>
    input:not([disabled]) {
      background-color: lightgreen;
    }
  </style>
</head>
<body>
  <dl>
    <dt>Name</dt>
    <dd><input type="text"></dd>
    <dt>Password</dt>
    <dd><input disabled type="password"></dd>
    <dt>Mobile</dt>
    <dd><input required type="text"></dd>
  </dl>
</body>
</html>
```

Universal Selector:

- It is defined by using “*” that represents all.
- It apply effects to all elements.

Ex:

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Languages</title>
```

```
    <style>
```

```
      * {
```

```
        font-style: italic;
```

```
      }
```

```
    </style>
```

```
  </head>
```

```
  <body>
```

```
    <dl>
```

```
      <dt>Name</dt>
```

```
      <dd><input type="text"></dd>
```

```
      <dt>Password</dt>
```

```
      <dd><input disabled type="password"></dd>
```

```
      <dt>Mobile</dt>
```

```
<dd><input required type="text"></dd>
```

```
</dl>
```

```
</body>
```

```
</html>
```