Web Designing
Overview of CSS

Outline

- Overview of CSS
  - Evolution of CSS, Syntax of CSS, Exploring CSS Selectors
  - Inserting CSS in an HTML Document
  - Exploring Background of Webpage using CSS
  - Exploring Font Properties in CSS
  - Controlling the Display of an Element using CSS
  - Positioning of Element using CSS
  - Floating and Element using CSS
  - Exploring Different Model in CSS – Box Model, Line Box Model, Template Layout Model, Multi- Column Model.

CSS

- What is CSS ?
  - CSS stands for Cascading Style Sheet, is a text file with .css extension and is commonly used to define styles and layouts of Web pages written in HTML and Extensible Hypertext Markup Language (XHTML).
- **Cascading**: refers to the procedure that determines which style will apply to a certain section, if you have more than one style rule.
- **Style:** how you want a certain part of your page to look. You can set things like color, margins, font etc for things like tables, paragraphs, and headings**.**
- **Sheets:** the "sheets" are like templates, or a set of rules, for determining how the webpage will look.
- So, CSS (all together) is a styling language – a set of rules to tell browsers how your webpage should look.

3

CSS

- **A Brief History of CSS**
  - CSS was invented by Hakon Wium Lie on 10th Oct, 1994 and maintained by a group of people within World Wide Web Consortium (W3C).
- **Why CSS ?**
  - CSS simplifies the task of maintaining a Web document by separating its style information, such as font size, font color, line width, and background color etc. This separation allows you to apply the same style rules to multiple Web pages. CSS also allows you to apply a style multiple times in a single Web page.
  - CSS defines layout of HTML documents. For example, CSS covers Fonts, colors, margins, lines, height, width, background images, advanced positions and many other things.

4

CSS

- **Why to Use CSS ?**
  - Suppose, you have a Web page that contains multiple tables and you want to apply some style on the table caption, table header, and table cells.
  - To do this, you just need to write the code once in a CSS style sheet and apply this style sheet to all the tables of your Web page.
  - This reduces the complexity and redundancy of code in the Web page and saves your time, as you do not need to write the same code again and again.

5

CSS

- **CSS Recommendation :**
  - The CSS file contains the style code for the structure, such as headings, paragraphs, and links. The styles patterns and layouts defined in a CSS file can be modified by making the required changes in the code of the CSS file.
  - CSS also provides a pattern that helps in applying the style rules on specific elements. This pattern is known as a selector. Some of the most-commonly used CSS selectors are universal, type, and class.

6

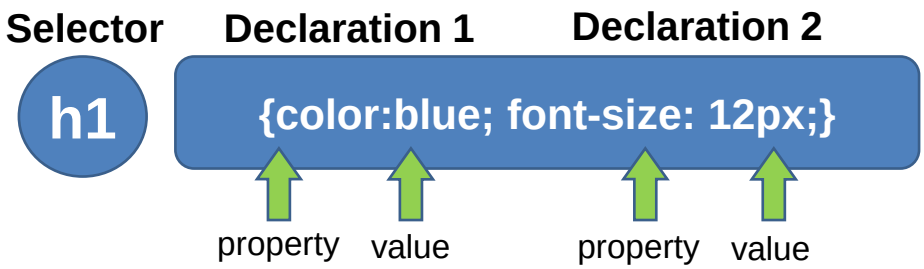
Advantages and Disadvantages of CSS

- CSS saves more time
- External CSS helps Web Pages Load Faster
- CSS is easy to maintain
- CSS has superior styles to HTML
- CSS provides multiple device compatibility
- CSS provides global web standards
- CSS provides offline browsing
- CSS is platform independence
- External CSS makes Updates Easier and Smoother
- **Disadvantages :**
  - Browser Dependent
  - Difficult to retrofit in old websites

7

Basic Syntax of CSS

- A CSS rule has two main parts: a selector, and one or more declarations



- The **selector** can be HTML element, id or class.
- Each **declaration** consists of a **property** and a **value**.
- The **property** is the style attribute you want to change. Each property has a **value**.

8

Using CSS

- CSS can be added to HTML documents in 3 ways:
  - Inline - by using the style attribute inside HTML elements
  - Internal - by using a <style> element in the <head> section
  - External - by using a <link> element to link to an external CSS file

9

Inline Style

- To define an inline CSS style, we simply add the style attribute to an XHTML element with the CSS declaration as the attribute value:

```
<h2 style="color:red;">CAUTION: Icy Road Conditions</h2>
<h2>Please Slow Down!</h2>
```

An inline style declaration is highly specific and formats just one element on the page. No other elements, including other <h2> elements on the page, will be affected by this CSS style.

Since inline styles have limited scope and do not separate content from presentation, their use is generally discouraged. We won't be using inline styles much in this class.

10

Inline Style

- It is possible to place CSS right in your HTML code, and this method of CSS usage is referred to as inline css.
- Inline CSS has the highest priority out of external, internal, and inline CSS.
- This means that you can override styles that are defined in external or internal by using inline CSS.
- If you want to add a style inside an HTML element all you have to do is specify the desired CSS properties with the style HTML attribute.
- Example :

HTML

```
<p style="background: blue; color: white;"> My Inline CSS </p>
```

11

Internal Inline CSS

```
<!DOCTYPE HTML>
    <HEAD>
        <TITLE>CSS Example </TITLE>
        <STYLE>
            *{margin: 0;}
            body
            {
                color:Red;
                background-color:lightgrey;
                font-family: sans-serif;
            }
            h1{font-size:18pt}
            p{font-size:12pt}
```

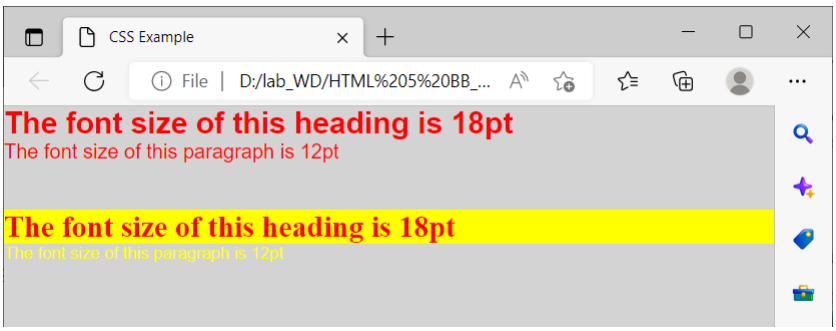
12

Internal Inline CSS

```
        </STYLE>
    </HEAD>
    <BODY>
        <H1>The font size of this heading is 18pt</H1>
        <P>The font size of this paragraph is 12pt</P>
        <BR/>
        <BR/>
        <H1 style="color:#ff0000;background-
         color:#ffff00;font-family: Ariel;">The font size of
          this heading is 18pt</H1>
        <P style="color:yellow; font-size:10pt">The font
         size of this paragraph is 12pt</P>
    </BODY>
</HTML>
```

13

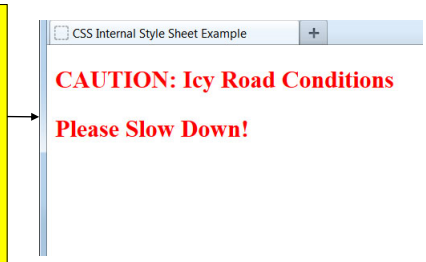
output



Internal Style

- To use an internal CSS style sheet, we add a <style> section within the <head> of the page. All our CSS declarations go within this section:

```
<head>
  ...
  <style type="text/css">
    h2 {color:red;}
  </style>
</head>
<body>
  <h2>CAUTION: Icy Road Conditions</h2>
  <h2>Please Slow Down!</h2>
</body>
```



Styles declared in the internal style sheet affect all matching elements on the page. In this example, all <h2> page elements are displayed in the color red.

Since formatting declarations are entirely in the <head> section, away from the actual page content, internal CSS style sheets do a much better job than inline styles at separating content from presentation.

External Style

- When using CSS it is preferable to keep the CSS separate from your HTML.
- Placing CSS in a separate file allows the web designer to completely differentiate between content (HTML) and design (CSS).
- External CSS is a file that contains **only CSS** code and is saved with a **".css"** file extension.
- This CSS file is then referenced in your HTML using the <**link**> instead of <**style**>.

External Style

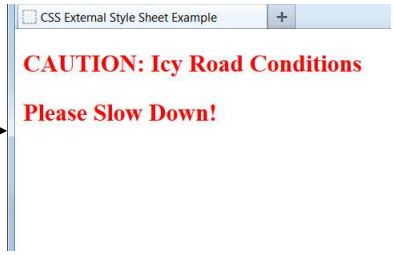
- To use an external CSS style sheet, we create a new file (with a .css extension) and write our style declarations into this file. We then add a <link> element into our HTML file, right after the opening <head> tag:

style.css (separate file):

```
h2 {color:red;}
```

example.html file:

```
<head>
  <link rel="stylesheet" type="text/css"
   href="style.css" />
  ...
</head>
<body>
  <h2>CAUTION: Icy Road Conditions</h2>
  <h2>Please Slow Down!</h2>
</body>
```



The <link> element instructs the browser to load the external file specified by the href attribute and to apply the CSS style declarations contained there.

- Example :

Demo.html

```
<html>
<head>
<link rel="stylesheet" type="text/css"
href="test.css">
</head>
<body>

<p> Hello Friends </p>
<p id="para1"> How are you? </p>

</body>
</html>
```

test.css

```
#para1{
        text-align: center;
}
p
{
        color : blue;
}
```

Output

Hello Friends

How are you?

External Style Sheets

- In an external style sheet, the style sheet rules are saved into a text file with the .css extension.
- Once you have a style sheet document, you can link it with your Web pages in the following two ways.
  - **Linking** - Refers to the HTML LINK element, which is used to link a style sheet. This element has the following three attributes.
    - rel : The rel attribute specifies what you are linking (style sheet in this case).
    - type : The type specifies the MIME type for the browser
    - href : The href attribute specifies the path of the .css file

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

19

External Style Sheets

- **Importing** - Helps you in accessing the style rules from other CSS style sheets. The @import keyword is used, followed by the Uniform Resource Identifier (URI) of the style sheet to which you want to import the style rules.
- Example:

```
<style type="text/css">
    @import url("mystylesheet.css")
     h1 { color: blue }
</style>
```

- In the above code fragment, we have used the @import keyword followed by the URL of the style sheet, named mystylesheet.css. In addition to the @import rule the @media rule of CSS helps you in applying the styles to the media device depending on the type of the device a page is displaying. Some of the media devices supported by the CSS are computer screens, printers, televisions, handhelds, speech synthesizers, and projectors. Please note that all the media types are not supported by all the Web browsers.

20

External Style Sheets

- The following code fragment shows an example of using @media rule.

```
<style type="text/css">
      @media screen
       {
          body { font-size: 13px; }
       }
</style>
```

21

Import Rule

```
<!DOCTYPE HTML>
        <HEAD>
                <TITLE>CSS Example </TITLE>
                <STYLE TYPE="text/css">
                        @import url("example.css");
                          // can import more css also
                        P {color:blue}
                </STYLE>
        </HEAD>
        <BODY>
                <H1>The font size of this heading is 18pt</H1>
                <P>The font size of this paragraph is 12pt</P>
```

22

Import Rule

```
<TABLE>
    <TR>
        <TH>living being</TH>
        <TH>shelter</TH>
    </TR>
    <TR>
        <TD class="code">Lion</TD>
        <TD>Lion lives in the den</TD>
    </TR>
    <TR>
        <TD class="code">Man</TD>
        <TD>Man lives in house</TD>
    </TR>
```

23

Import Rule

```
    <TR>
        <TD class="code">Fish</TD>
        <TD>Fish lives in water</TD>
    </TR>
    <TR>
        <TD class="code">Bird</TD>
        <TD>Bird lives in nest</TD>
    </TR>
</TABLE>
</BODY>
</HTML>
```

24

example.css

```
/* style sheet for sample.html document */
*{margin: 0;}
body
{
color:#ff0000;
background-color:#ffff00;
font-family: sans-serif;
}
h1{font-size:18pt}
p{font-size:12pt}
```

25

example.css

```
table
{
background-color:#efefef;
border-style:solid;
border-width:2px;
border-color:#999900;
}
th
{
background-color:#cccc00;
font-weight:bold;
padding:3px;
}
```

26

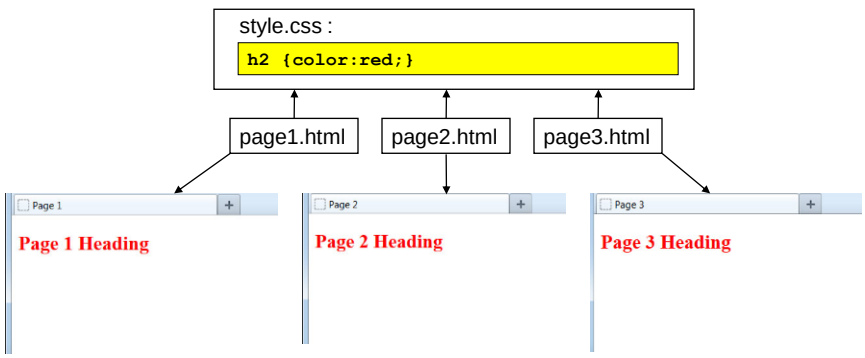
example.css

```
td{padding:3px}
td.code
{
font-family: serif;
font-weight:bold;
}
```

27

Benefit of External Style Sheet

- The real power of using an external style sheet is that multiple web pages on our site can link to the same style sheet:



28

Internal vs. External Style Sheets

- Internal Style Sheets:
  - are appropriate for very small sites, especially those that have just a single page.
  - might also make sense when each page of a site needs to have a completely different look.
- External Style Sheets:
  - are better for multi-page websites that need to have a uniform look and feel to all pages.
  - make for faster-loading sites (less redundant code).
  - allow designers to make site-wide changes quickly and easily.

29

How to write style rules

- Two parts: (1) selector and (2) declaration.
  - **Selector**: the HTML element you want to add style to. <p> <h1> <table> etc
  - **Declaration**: the statement of style for that element. Made up of property and value.
- **Selector**: the thing I want to make styled: a paragraph, a table, a header, etc.
- **Declaration**: what I want to do to my paragraph, table, header, etc.
- Selector {declaration;}
- Declaration = {property: value;}
- **Property**: what aspect you want to change. ex: color, font, margins, etc.
- **Value**: the exact setting for that aspect. ex: red, italic, 40px, etc

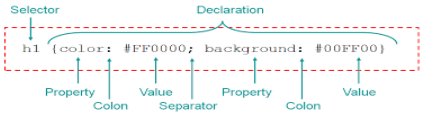
30

How to write style rules

- selector {property: value;}
- Essentially means: The thing I want to change
  {the aspect of that thing I want to change: what I want it to be;}
- Selector {property: value;}
  - h1 {color: red;}
  - Means: Speaking of my heading1, I want the text color to be red.
- A CSS declaration always ends with a semicolon, and declaration groups are surrounded by curly braces.

31

How to write style rules



- selector {property: value;}
- Essentially means: The thing I want to change
  {the aspect of that thing I want to change: what I want it to be;}
- Selector {property: value;}
  - h1 {color: red;}
  - Means: Speaking of my heading1, I want the text color to be red.
- Selector
  {
  1st property : value;
  2nd property : value;
  3rd property : value;
  . . .
  Nth property : value;
  }
- The selector is the name of the element to which you want to apply the CSS properties.

Setting Multiple Properties

- We can define as many properties as we wish for a selector:

```
p {color:red;font-style:italic;text-align:center;}
```

In this example, all text within paragraph elements will show in red italics that is centered on the page.

```
p {
  color: red;
  font-style: italic;
  text-align: center;
}
```

Just as with HTML, browsers ignore space characters in CSS code. Many designers take advantage of this fact by placing the opening and closing curly brackets on their own dedicated lines. Each of the property and value pairings are placed on their own indented line, with a space after the colon. This makes the code far easier to read.

33

Example

```
<!DOCTYPE html>
<html>
<head>
        <title>CSS Syntax Example</title>
        <style>
                p
                {
                        color: red;
                        text-align: center;
                }
        </style>
</head>
<body>
    <p>Hello Browser!</p>
    <p>I am center-aligned with red color.</p>
    </body>
    </html>
```

34

CSS Selectors

- CSS selectors are used to "find" (or select) the HTML elements you want to style.
- The different types of selectors are as follows.
  - The universal selector
  - The type selector
  - The class selector
  - The id selector
  - The child selector
  - The descendant selector
  - The adjacent sibling selector
  - The attribute selector
  - The query selector

35

CSS Selectors

- The universal selector : The universal selector (*) selects all the elements that are present in an HTML document.
- The universal selector is represented by an asterisk symbol, as shown in the following code snippet. ***{}**
- The following code fragment shows the use of universal selector. *** { margin:0; padding:0;** color: green; **}**
- In the above code fragment, the margin and the padding properties are set to 0 for all the elements in the HTML or XHTML document on which the CSS rule is applied.
- All the font have green color for all elements.
- **Example :** The CSS rule below will affect every HTML element on the page:

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

36

Selector Example

- **The CSS universal Selector Example**

```
<!DOCTYPE html>
    <html>
      <head>
       <style>
            * {
                text-align: center;
                color: blue;
              }
         </style>
      </head>
       <body>
           <h1>Hello world!</h1>
          <p>Every element on the page will be affected by the style.</p>
           <p id="para1">Me too!</p>
           <p>And me!</p>
       </body>
</html>
```

37

CSS Selectors

- **CSS Type Selector/Grouping Selector :** The type selector matches all the elements specified in a list with the given value to determine the elements to which the CSS rules are to be applied.
- The rules applied to several elements of an HTML or XHTML document are similar to the ones applied to a CSS file.
- The following code fragment shows how to use the type selector in CSS. **h1, h2, h3, p { font-family: sans-serif }**
- In the above code fragment, we have specified the font-family property for the different heading elements (h1, h2, h3) and for the paragraph element (p).

38

Selector Example

- **The CSS Grouping Selector :** The grouping selector selects all the HTML elements with the same style definitions.
- Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

```
h1 {
  text-align: center;
  color: red;
}
h2 {
  text-align: center;
  color: red;
}
p {
  text-align: center;
  color: red;
}
```

39

Selector Example

- It will be better to group the selectors, to minimize the code.
- To group selectors, separate each selector with a comma.
- **Example :** In this example we have grouped the selectors from the code above:

```
h1, h2, p {
  text-align: center;
  color: red;
}
```

40

Selector Example

- **The CSS Grouping Selector Example**

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         h1, h2, p {
                        text-align: center;
                        color: red;
                      }
        </style>
      </head>
      <body>
         <h1>Hello World!</h1>
         <h2>Smaller heading!</h2>
          <p>This is a paragraph.</p>
</body>     </html>
```

41

CSS Selectors

- **Class Selector :** The class selector allows you to apply CSS rules to the elements that carry a class attribute whose value matches with the class attribute specified in the class selector.
- To select elements with a specific class, write a period (.) character, followed by the class name.
- Let's consider that you have an element, H1, with a class attribute whose values is intro, as shown in the following code fragment. **<H1 class="intro">Header 1</H1>**
- You can use a class selector in either of the two ways.
  - **(i)** By applying the CSS rule to all the elements that have the class attribute of the same value. The following code fragment shows how to apply the CSS rule.

    .intro { font-family: sans-serif}
  - In the above code fragment, a period is followed by a value. The value is followed by braces which embeds the CSS rule within it. The CSS rule is applied to all the elements having the class attribute with ***intro*** as its value.

42

CSS Selectors

- **(ii)** By applying the CSS rule to the H1 elements, whose class attribute contains *intro* as its value. The following code fragment shows how to apply the CSS style on H1 elements.
  h1.intro { font-family: sans-serif}
- In the above code fragment, an element name is followed by a value.
- The value is followed by braces, which embeds the CSS rule within it.
- The CSS rule is applied to all the H1 elements having the class attribute with ***intro*** as its value.

43

Selector Example

The CSS class Selector Example

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         .center {
               text-align: center;
               color: red;
          }
      </style>
    </head>
    <body>
       <h1 class="center">Red and center-aligned heading</h1>
       <p class="center">Red and center-aligned paragraph.</p>
     </body>
</html>
```

44

Selector Example

- **The CSS class Selector Example (**You can also specify that only specific HTML elements should be affected by a class.)
- A class name cannot start with a number.
- In this example only <p> elements with class="center" will be red and center-aligned:

```
p.center {
    text-align: center;
    color: red;
    }
```

45

Selector Example

The CSS class Selector Example (In this example only <p> elements with class="center" will be red and center-aligned:)

```
<!DOCTYPE html>
<html>
<head>
<style>
p.center {
    text-align: center;
    color: red;
    }
</style>
</head>
<body>
<h1 class="center">This heading will not be affected</h1>
<p class="center">This paragraph will be red and center-aligned.</p>
</body>
</html>
```

46

Selector Example

- **The CSS class Selector Example (**HTML elements can also refer to more than one class.)
- In this example the <p> element will be styled according to class="center" and to class="large":

```
<p class="center large">This paragraph refers to two classes.</p>
```

47

Selector Example

```
<!DOCTYPE html>
<html>
    <head>
      <style>
        p.center {
              text-align: center;
              color: red;
                 }
         p.large {
              font-size: 300%;
                 }
      </style>
    </head>
```

48

Selector Example

```
<body>
    <h1 class="center">This heading will not be affected</h1>
    <p class="center">This paragraph will be red and center-
     aligned.</p>
    <p class="center large">This paragraph will be red, center-
     aligned, and in a large font-size.</p>
</body>
</html>
```

49

Class Selector

```
<!DOCTYPE HTML>
<html>
  <head>
    <title> Class selector example</title>
    <style>
            h2.green    {    color: green;    }
            .green      {    color: yellow;   }
            p.red       {    color: red;      }
      </style>
  </head>
  <body>
    <h2 class="green">CSS Class Selector Tutorial</h2>
    <p class="green">This is tutorial on CSS class selector.</p>
```

50

Class Selector

```
        <h2 class="red">CSS Class Selector Example</h2>
        <p class="red">This is example on CSS class selector.</p>
    </body>
</html>
```

51

CSS Selectors

- **CSS ID Selector :** The value of the id attribute is unique within a document; therefore, the selector is applied only to the content of one element.
- The following code fragment shows the h1 element having myHeader as the value of the id attribute.
  <H1 id="myHeader">Hello World!</H1>
- The following code fragment shows the id selector, which is represented by a hash symbol (#) and followed by the value of the id attribute.
- **Note:** An id name cannot start with a number.
  #myHeader{ font-family: sans-serif }
- In the above code fragment, myHeader is the value of the id attribute.
- The CSS rule is applied to the value of the id attribute.

52

Selector Example

- Example : The CSS rule below will be applied to the HTML element with id="para1":

```
#para1 {
  text-align: center;
  color: red;
}
```

53

Selector Example

The CSS id Selector Example

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      #para1 {
       text-align: center;
       color: red;
      }
    </style>
  </head>
  <body>
     <p id="para1">Hello World!</p>
     <p>This paragraph is not affected by the style.</p>
  </body>
</html>
```

54

CSS Selectors

- **CSS Child Selector :** The child selector matches the element that is an immediate child of another element.
- In the child selector, greater than symbol (>) is used as the combinator, as shown in the following code fragment.
  TD>b{ font-family: sans-serif }
- A combinator is a symbol, such as >, <, and +, which shows the relationship between the two elements.
- In the preceding code fragment, the b element is the immediate child of the TD element.
- The CSS rule is applied to all the b elements that are immediate children of TD elements.

55

CSS Selectors

```
<style>
        body > p
        {
            color: green;
        }
    </style>
<body>
<h2>CSS Child Selector</h2>
<hr/>
<p>Hello Browser!</p>
<p>This is CSS child selector</p>
<hr/>
<div>
<p>Paragraph in <div></p>
</div>
<p>Paragraph out of </div></p>
```

56

CSS Selectors

- **CSS Descendant Selector :** The descendant selector matches an element that is a descendant of another element.
- A descendant element is an element that is nested inside another element.
- In case of the descendant selector, white space is used as the combinator, as shown in the following code fragment.
  table b { font-family: sans-serif }
- In the above code fragment, CSS is applied to all the b elements that are nested within the table element.

57

CSS Selectors

```
<style>
        ul em
        {
                color: green;
        }
</style>
<h2>CSS Descendant Selector</h2>    <hr/>
<p>Hello Browser!</p>
<p>This is CSS descendant selector.</p>    <hr/>
Here is the unorganized lists:<br/>
<ul>
        <li>list <em>item</em> 1</li>
        <li>list <em>item</em> 2</li>
        <li>list <em>item</em> 3</li>
</ul>
```

58

CSS Selectors

- **CSS Adjacent Sibling Selector :** The adjacent sibling selector selects all the elements that are adjacent siblings of a specified element. Sibling elements must have the same parent element.
- The word adjacent means side-by-side, so no other element could exist between the adjacent sibling elements. To use an adjacent sibling selector, the plus symbol (+) is used as its combinator, as shown in the following code fragment of a CSS file.

  H2+P { font-family: sans-serif }
- Let's apply the preceding code fragment of a CSS file to the following HTML code fragment.

  <H2>Heading</H2>
  <P>The selector above matches this paragraph.</P>
  <P>The selector above does not match this paragraph.</P>

59

CSS Selectors

- In the above code fragment, the first paragraph matches the adjacent sibling selector, H2+P, because the P element is an adjacent sibling to the H2 element.
- The second paragraph does not match with the selector. Although it is a sibling of the H2 element, it is not adjacent to the element.

60

CSS Selectors

- **CSS Attribute Selector :** The CSS attribute selector selects elements on the basis of some specific attributes or attribute values.
- The following table describes most common types of attribute selectors.
- Example:

```
<style>
        input[type="text"]
        {
            color: green;
        }
</style>
```

61

Attribute Selectors

Name	Syntax	Match	Example
Hyphen selector	[attribute|=value]	Matches if the element has an attribute with a value followed by a hyphen	[lang|=fr] { background-color:red; }
Existence selector	[attribute]	Matches if the element has a specific attribute	a[title] { color:green; }
Equality selector	[attribute=value]	Matches if the element has an attribute with a specific value	a[href=http://www.ddu.ac.in/] { font-weight:bold; }
Space selector	[attribute~=value]	Matches if the element has an attribute with space separated items that match with the value	a[title~=Web] { background-color:red; }

62

CSS Selectors

- **CSS Query Selector :**
  The *querySelector()* and *querySelectorAll()* methods accepts CSS selectors as parameters and return the matching element node in the document tree.
- The *querySelector()* method helps in querying the entire document or a specific element of the document.
- You can use all the CSS selectors with this method as parameters.
- If multiple elements are available, CSS selectors return the first matching element; or returns null, if no element is available.
- The *querySelectorAll()* method returns all the available elements as a single static collection of elements known as *staticNodeList*.
- This collection of elements is not affected by any change made in the document tree, for instance removing or inserting a node does not affect *staticNodeList*.

63

Working with querySelector

```
<!DOCTYPE HTML>
    <HEAD>
        <TITLE> Working with querySelector </TITLE>
    </HEAD>
    <BODY>
        <H1>Working with querySelector</H1>
        <div id="div1" style="padding:50px; width:100px;
          height:100px; border:1px solid black">
        </div>
        <P>Move the mouse cursor over the color
         name</P>
        <LABEL id="label1" >Blue</LABEL>--
        <LABEL id="label2" >Red</LABEL>--
```

64

Working with querySelector

```
<LABEL id="label3" >Yellow</LABEL>--
<INPUT id="text">
<script type="text/javascript">
        if (document.querySelector)
        {
                var
        lbl1=document.querySelector('#label1')
                var
       lbl2=document.querySelector('#label2')
                var
        lbl3=document.querySelector('#label3')
                lbl1.onmouseover=function(){
        document.querySelector('#text').value="This
        is Blue color";
```


Working with querySelector

```
        document.querySelector('#text').style.color="blue";
        document.querySelector('#div1').style.background="blue"
                        }
                        lbl2.onmouseover=function(){
        document.querySelector('#text').value="This is Red color";
        document.querySelector('#text').style.color="red"
        document.querySelector('#div1').style.background="red"
                        }
                        lbl3.onmouseover=function(){
        document.querySelector('#text').value="This is Yellow
                                color";
        document.querySelector('#text').style.color="yellow"
        document.querySelector('#div1').style.background="yellow
"                       }
                }
```

Working with querySelector

```
            </script>
        </BODY>
</HTML>
```

67

Working with querySelectorALL

```
<!DOCTYPE HTML>
 <HTML>
   <HEAD>
     <TITLE>Working with the querySelectorAll</TITLE>
   </HEAD>
   <BODY>
     <H1>Working with the querySelectorAll</H1>
     <FORM id="myform">
       <B>Select your favourite cars:</B> <BR/>
       <INPUT name="cars" type="checkbox" value="Ferrari"
        />Ferrari <BR/>
       <INPUT name="cars" type="checkbox" value="Audi"
       />Audi <BR/>
```

68

Working with querySelectorALL

```
<INPUT name="cars" type="checkbox" value="BMW" />BMW <BR/>
<INPUT name="cars" type="checkbox" value="Mercedez" />Mercedez<BR/>
<BR />
<INPUT type="submit" />
</FORM>
<SCRIPT type="text/javascript">
if (document.querySelector){
document.querySelector('#myform').onsubmit=function(){
var checkcars=this.querySelectorAll('input[name="cars"]:checked')
document.write("<B>You have selected the following cars: </B><BR/>");
```


Working with querySelectorALL

```
for (var i=0; i<checkcars.length; i++){
var value="“;
value += checkcars[i].value + "<BR/>"
document.write("<LI>"+value+"</LI>"); }

return false;
} // end of documentqueryselector()
}//end of if
</SCRIPT>
</BODY>
</HTML>
```

CSS Selectors

- **CSS element Selectors :** The element selector selects elements based on the element name.
- You can select all <p> elements on a page as shown in the example given below.

```
<head>
        <style>
                p
                {
                        text-align: center;
                        color: red;
                }
        </style>
</head>
<body>
<p>Hello Browser!</p>
<p>This is CSS element selector.</p>
```

71

Selector Example

```
<!DOCTYPE html> // Element selector
<html>
  <head>
    <style>
      p {
          text-align: center;
          color: red;
      }
   </style>
  </head>
  <body>
      <p>Every paragraph will be affected by the style.</p>
      <p id="para1">Me too!</p>
       <p>And me!</p>

   </body>
</html>
```

72

CSS Selectors

- **CSS Multiple Style Rules :** You may need to define multiple style rules for a single element.
- You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example.

```
h1
{
   color: #36C;
   font-weight: normal;
   letter-spacing: .4em;       margin-bottom: 1em;
   text-transform: lowercase;
}
```

- Here all the property and value pairs are separated by a semi colon (;). You can keep them in a single line or multiple lines.
- For better readability we keep them into separate lines.

73

CSS Selectors

- **CSS Grouping Selectors :** You can apply a style to many selectors if you like. Just separate the selectors with a comma as given in the following example.

```
h1, h2, h3
{
   color: #36C;
   font-weight: normal;
   letter-spacing: .4em;
   margin-bottom: 1em;
   text-transform: lowercase;
}
```

- This define style rule will be applicable to h1, h2 and h3 element as well.

74

CSS Selectors

- The order of the list is irrelevant. All the elements in the selector will have the corresponding declarations applied to them.
- You can combine various class selectors together as shown below.

```
#content, #footer, #supplement
{
    position: absolute;
    left: 510px;
    width: 200px;
}
```

75

CSS Inheritance

- In CSS, a property that is applied to an element is also inherited by the child elements of that element.
- For example, if the font-family property is declared for the BODY element, it is automatically applied to all the elements present inside the BODY element.
- This inheritance saves your time in writing the repeated code for every single element that constitutes the Web page.
- The following code fragment shows inheritance in CSS.

```
<div style="font-family:serif; border:1px solid red; padding:10px;>
        This text will be in a serif font.
        <p>
            This text is also in a serif font, because font is inherited
            by default.
            But the border and padding properties are not
            inherited from the parent div.
        </p>
</div>
```

76

CSS Inheritance

- In the above code fragment, font-family is automatically inherited by the **p** element from the parent **div** element.
- In this case, border and padding properties are not inherited from the parent element.

77

CSS properties that are inherited automatically

Properties	Description
border-collapse	Represents the border display
border-spacing	Represents the thickness of the border
caption-side	Represents the place of the table caption
color	Represents the color of a text
cursor	Represents the cursor to be displayed
direction	Represents the direction of a text
empty-cells	Specifies whether to display the border and background on empty cells
font	Specifies all the font properties
font-family	Represents the font family of a text
font-stretch	Represents text as stretched or condensed

78

CSS properties that are inherited automatically

Properties	Description
font-size	Represents the size of a text
font-size-adjust	Represents the size of the text on the basis of an aspect value
font-style	Represents the style of a text
font-variant	Represents the fonts variant, such as small caps
font-weight	Represents the font as bold
letter-spacing	Represents the space between the characters in a text
line-height	Represents the height of a line
list-style	Represents the style of the list
list-style-image	Represents an image as a list marker

79

CSS properties that are inherited automatically

Properties	Description
list-style-type	Represents the type of a list marker
quotes	Represents the quotation marks in a text
text-align	Represents the horizontal alignment of a text
text-indent	Represents the indentation of a text
text-transform	Represents the transformation of text, such as uppercase, capitalize, or lowercase
white-space	Handles the spaces in an element
word-spacing	Represents the spacing between the words in a text

80

81