



CSS

CASCADING STYLE SHEET

CSS stands for Cascading Style Sheet.

CSS is a plain text file to store properties.

CSS describing look and formatting properties for how content should be display on web-page.

CSS give we control to make global change and apply all web-page.

Define CSS Comments : CSS comment we can define in to <style> section or Style Sheet using /* and */.

Syntax :

```
<html>
  <head>
    <style type = "text/css">
      /* comment line or statement. */
    </style>
  </head>
  <body>
    ...
    ...
    ...
  </body>
</html>
```

<style> section
or
Style Sheet

Easy Manage : CSS with we can better manage whole web-page. CSS allow to manage entire site elements looks and formatting in single style sheet file.

Global Change: CSS Style Sheet changes apply to global change for all web-page. When Style Sheet appear with web-page it's know as Cascading Style Sheet.

Save time: When we create HTML document, we define separate set attributes value in each element. but it is limited use. But, CSS give a lot of flexibility to set properties and values either group of element or single element. So it's benefit to avoid same code write again and again.

Easy Maintain/Update: CSS style sheet maintain easier and anytime you can edit elements properties and values.

Three way to write CSS: We can write CSS styles three way in-line element line (scope is only that element), Internal style write in header section (scope is only that web-page), or External style sheets write in external .css extension files. External style sheets enable to change the elements and layout style of all the pages in a Web site, just by editing one single file.

Page Load Faster: Web-page with stylesheets take a bit of second to load very fastly.

In CSS style. we can define and implement CSS style four different way (In-line style, Internal style, external style sheet and @import). So, we can define and implement CSS style in following different ways.

1. In-line CSS style
2. Internal CSS Style
3. External Style Sheet
4. @import Style Sheet

1. In-line CSS Style

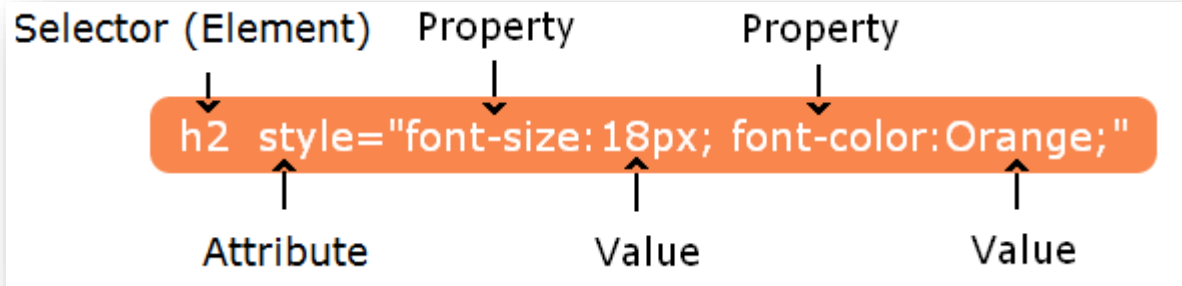
In-line CSS Style write in element line using style attribute. All most every HTML element support style attribute. In-line stylesheet priority high more than other three.

In-line CSS style consists set of rules in four part:

1. Selector (Element)
2. Style (Attribute)
3. Property and
4. Value

Syntax :

Selector is normally HTML element that element you want to assign CSS style. And style is attribute to assign CSS property and set suitable value.



CSS Style : There are various types of style are as follows.

- CSS Text
- CSS Font
- CSS Background
- CSS Links
- CSS Lists
- CSS Display
- CSS Visibility

CSS TEXT properties are various type like text color, text-align, text-decoration, letter-spacing and many more properties. Using these properties we can change the text formatting style.

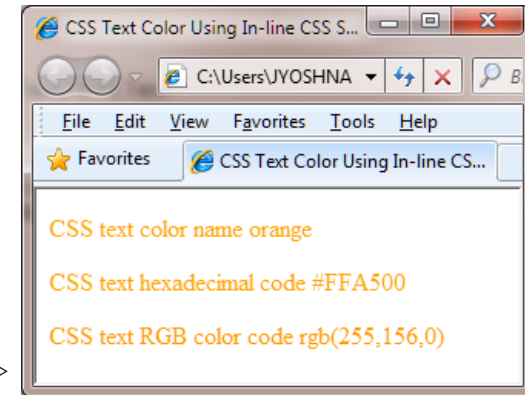
Following are some CSS text properties listed. Using these properties we can play with text formatting style.

CSS Color :

CSS color property is used to set the Text color. The color value can be specified following three types:

- Color Name: Orange
- Color Hexadecimal Code: #FFA500
- Color RGB: rgb(255, 165, 0)

```
<!DOCTYPE html>
<html>
<head>
<title>CSS Text Color Using In-line CSS Style</title>
</head>
<body>
<p style="color: orange;">CSS text color name orange</p>
<p style="color: #FFA500;">CSS text hexadecimal code #FFA500</p>
<p style="color: rgb(255,156,0);">CSS text RGB color code rgb(255,156,0)</p>
</body>
</html>
```



2. Internal CSS Style

Internal CSS Style includes within web page using `<style type="text/css">.....</style>` element and between this element CSS style properties are listed. Internal CSS style normally written within `<head>.....</head>` element of the web page.

Internal CSS style consists set of rules in three part:

1. Selector (element, class, id)
2. Property and
3. Value

Syntax :

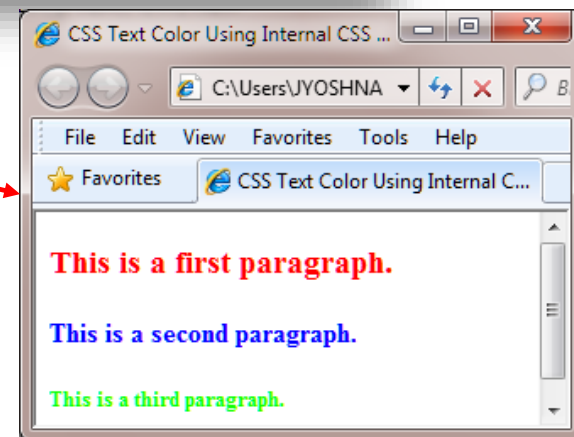
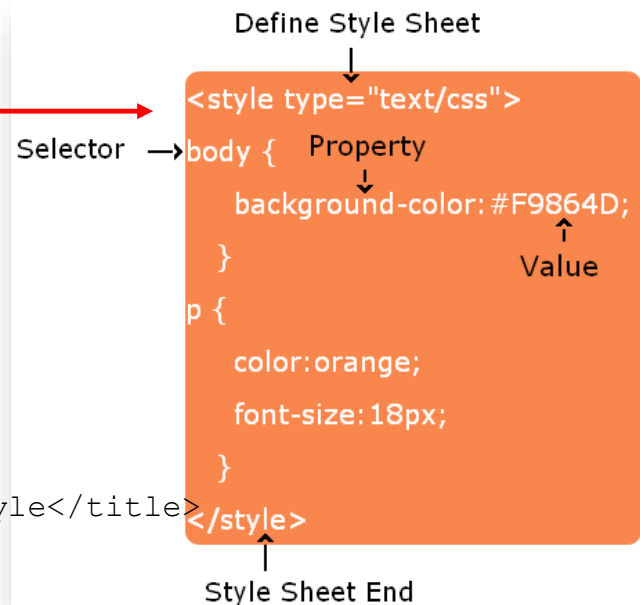
Selector is normally HTML element that element you want to assign CSS style. All elements within web page that elements assign CSS properties and set suitable values.

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

</style>
</head>
<body>
</body>
</html>
```

Example of CSS Text Color Using Internal CSS Style :

```
<!DOCTYPE html>
<html><head>
<title>CSS Text Color Using Internal CSS Style</title>
<style type="text/css">
  h3 {
    color:red;
  }
  h4{
    color:#0000FF;
  }
  h5{
    color:rgb(10,255,0);
  }
</style></head><body>
  <h3>This is a first paragraph.</h3>
  <h4>This is a second paragraph.</h4>
  <h5>This is a third paragraph.</h5>
</body></html>
```



3. External Style Sheet

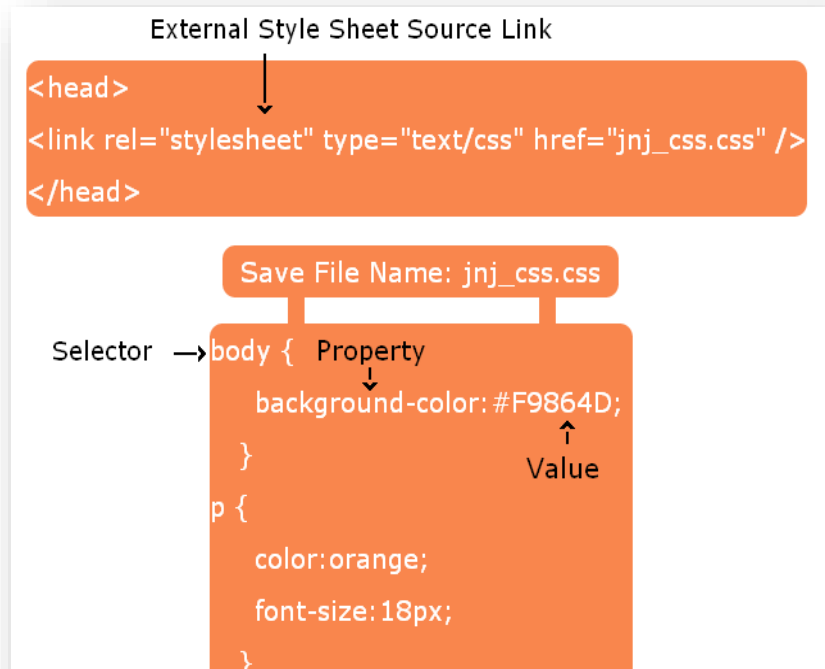
External Style Sheet define in separate .css extension file. and used to make global change also manage all web page from a single CSS document.

External style sheets give you control to change formatting and layout styles of every single elements in web pages. And only those web page who are linked with external CSS document.

External style sheet consists set of rules in four part:

1. External Source link
2. Selector (element, class, id)
3. Property and
4. Value

External stylesheet linked to a web page. Selector is normally HTML element (or class, id) to assign CSS properties and set suitable values.



For External style sheet, we create two files : .css file and .html file

Step no:01

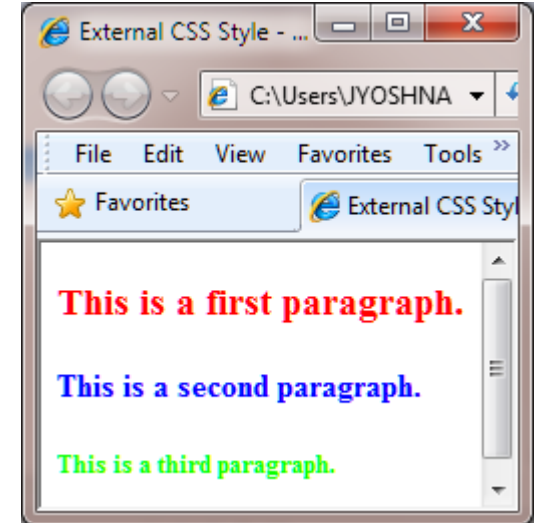
Now, create style.css file.

```
h3{
    color:red;
}
h4{
    color:#0000FF;
}
h5{
    color:rgb(10,255,0);
}
```

Step no:02

Now, create ExternalCSS.html file.

```
<!DOCTYPE html>
<html>
<head>
<title>External CSS Style</title>
<link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
<h3>This is a first paragraph.</h3>
<h4>This is a second paragraph.</h4>
<h5>This is a third paragraph.</h5>
</body>
</html>
```



4. @import Style Sheet

@import CSS Style is another way to loading a CSS file.

@import CSS Style define within `<style type="text/css">.....</style>` element in the `<head>.....</head>` of web page.

@import CSS style consists set of rules in three part:

1. @import (keyword)
2. url()
3. CSS file path

@import url('style.css');

For @import style sheet, we create two files : .css file and .html file
Step no:01

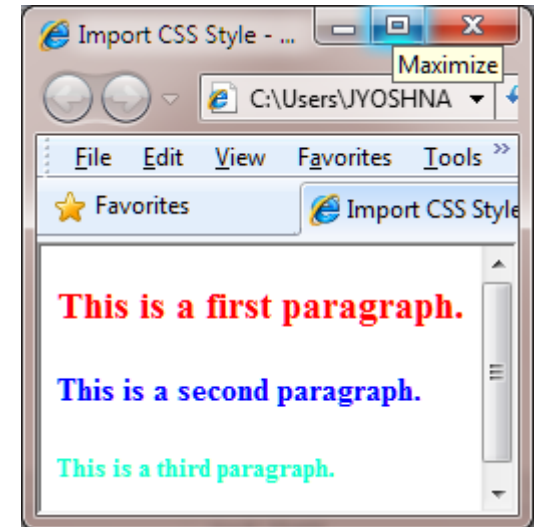
Now, create style.css file.

```
h3{
    color:red;
}
h4{
    color:#0000FF;
}
h5{
    color:rgb(10,255,0);
}
```

Step no:02

Now, create Example@.html file.

```
<!DOCTYPE html>
<html>
<head>
  <title>Import CSS Style</title>
  <style>
    @import url("style.css");
  </style>
</head>
<body>
  <h3>This is a first paragraph.</h3>
  <h4>This is a second paragraph.</h4>
  <h5>This is a third paragraph.</h5>
</body>
</html>
```



CSS Text Formatting Properties

CSS TEXT properties are various type like text color, text-align, text-decoration, letter-spacing and many more properties. Using these properties we can change the text formatting style. Following are some CSS text properties listed. Using these properties we can play with text formatting style.

CSS Color

CSS color property is used to set the Text color. The color value can be specified following three types:

1. Color Name: Orange
2. Color Hexadecimal Code: #FFA500
3. Color RGB: rgb(255, 165, 0)

EXAMPLE:

```
<!DOCTYPE html>
<html>
<head>
  <title>CSS Text Color</title>
</head>
<body>
  <p style="color: orange;">CSS text color name orange</p>
  <p style="color: #FFA500;">CSS text hexadecimal code #FFA500</p>
  <p style="color: rgb(255,156,0);">CSS text RGB color code rgb(255,156,0)</p>
</body>
</html>
```

CSS text-align

CSS text-align property use to set the **horizontal alignment** of text. text-align possible value center, left, right, or justify. When we set **text-align** property value **justify** than the effect is both width (left or right) equal like newspaper or books type.

EXAMPLE:

```
<!DOCTYPE html>
<html>
<head>
  <title>CSS Text Align</title>
</head>
<body>
  <p style="text-align: right;">CSS text align right</p>

  <p style="text-align: center;">CSS text align center</p>

  <p style="text-align: left;">CSS text align left</p>

  <p style="text-align: justify;">Hello, this is example of CSS text-align justify type. Both
side left    and right are equal. Its like newspaper or book type. Hello, this is example of CSS
text-align justify type. Both side left and right are equal. Its like newspaper or book type.
Hello, this is example of CSS text-align justify type. Both side left and right are equal. Its
like newspaper or book type.</p>
</body>
</html>
```

CSS text-indent

CSS text-indent property is used to set the paragraph first line left side leave the blank space.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>CSS text-indent</title>
</head>
<body>
<p style="text-indent: 35px;">This paragraph is example of CSS text-indent property
and value set 35px (pixel).So it means paragraph first line left side leave blank
space end of blank space start a first line paragraph text.</p>
</body>
</html>
```

CSS text-decoration

CSS text-decoration property use to decorate the text.

CSS text-decoration possible value underline, overline, blink, through etc.

EXAMPLE

```
<!DOCTYPE html>
<html>
<head>
  <title>CSS text-decoration</title>
</head>
<body>
  <p style="text-decoration: underline;">Text is underline decorate</p>
  <p style="text-decoration: overline;">Text is overline decorate</p>
  <p style="text-decoration: blink;">Text is blink decorate</p>
  <p style="text-decoration: line-through">Text is line delete decorate</p>
  <p style="text-decoration: none;">Text is nothing any decorate value</p>
</body>
</html>
```


CSS text-transformation

CSS text-transformation property use to text transform.

CSS text-transformation possible value capitalize, lowercase and uppercase in a text.

CSS text-transformation property value capitalize, it means first letter capital for all word.

EXAMPLE

```
<!DOCTYPE html>
<html>
<head>
  <title>CSS text-transform</title>
</head>
<body>
  <p style="text-transform: capitalize">This text transform to capital.</p>
  <p style="text-transform: lowercase">This text transform to lowercase.</p>
  <p style="text-transform: uppercase">This text transform to uppercase.</p>
</body>
</html>
```

CSS letter-spacing

CSS letter-spacing property set blank space between each two letter.

EXAMPLE

```
<!DOCTYPE html>
<html>
<head>
  <title>CSS letter-spacing</title>
</head>
<body>
  <p style="letter-spacing: 5px;">This text is represent letter spacing.</p>
</body>
</html>
```

CSS word-spacing

CSS word-spacing property set blank space between each two word.

EXAMPLE

```
<!DOCTYPE html><html><head>
<title>CSS word-spacing</title>
</head><body>
<p style="word-spacing: 25px;">This text is represent word spacing.</p>
</body></html>
```

CSS white-space

CSS white-space property use to set a predefined text. CSS white-space possible value is 'normal', 'pre'.

EXAMPLE

```
<!DOCTYPE html><html><head>
<title>CSS white-space</title>
</head><body>
<p style="white-space: pre;">This text is represent predefined text predefined area
another text</p>
</body></html>
```

CSS text-shadow

CSS text-shadow property is use to decorate text and apply shadow effect style.

EXAMPLE

```
<!DOCTYPE html><html> <head>
<title>CSS text-shadow</title>
</head><body>
<p style="text-shadow: 4px 4px 8px orange;">This text is represent text shadow
effect.</p>
</body></html>
```

The background property is one of the most useful ones and really fun to play around with. It is used to manipulate page backgrounds as we shall soon see.

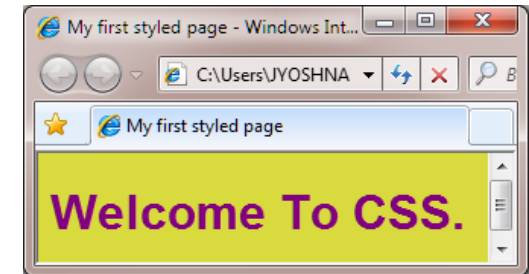
attributes	values
color	Sets an element's text-color (color name or color code).
background- color	Specifies the color in an element's background (color name or color code).
background- image	Set the background image (a URL or name).
background- repeat	With a background image specifies , sets up how the image repeats throughout page (Repeat-x, Repeat-y , Repeat, no- Repeat).

CSS Font

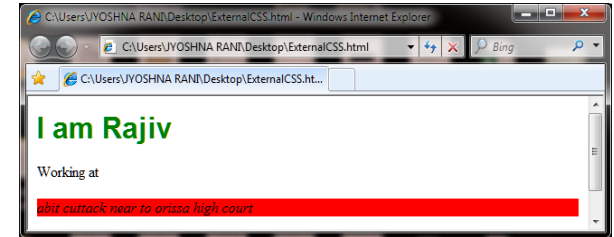
The font properties are used to set the various font oriented properties.

attributes	Values
font-family	A comma-delimited sequence of font family names (serif , Times new Roman).
font-style	Normal, Italic or Oblique.
font-weight	Normal, bold, bolder, lighter or one of the numerical values (100 - 900)
font-size	A term that denotes absolute size.

```
<!DOCTYPE html>
<html>
<head>
  <title>My first styled page</title>
  <style type="text/css">
    body {
      font-family: Georgia, "Times New Roman", Times, serif;
      color: purple;
      background-color: #d8da3d }
    h1 {
      font-family: Helvetica, Geneva, Arial, SunSans-Regular, sans-serif }
  </style>
</head>
<body>
  <h1>Welcome To CSS.</h1>
</body>
</html>
```



```
<html>
<head>
<style type="text/css">
H1 { font-family:arial,helvetica; color:green }
p { font-size:12pt;font-style:italic;background-color:red }
</style>
</head>
<body>
<H1> I am Rajiv </h1> Working at <p> abit cuttack near to orissa high court</p>.
</body>
</html>
```



The border are defined under the box properties category.

CSS supports the following border properties:

1. Border-style

The values for border-style properties are:

- solid
- Double
- Groove
- dotted
- dashed
- inset
- outset
- ridge
- hidden

2. Border-width

This property specifies the width of the four borders.

The values for border-width property are:

- thick
- thin
- medium
- or the give specific size in px, pt, cm, em, etc.

Note: The border-style property must be defined to get the result of border-width.

3. Border-color

This property sets the color for the borders.

Border colors are set by:

- Specifying a color name, like “orange”.
- Specifying hex value, like “#ff0000”.
- Specifying RGB value, like “rgb(255,0,0)”.

4. Border-direction

If you want an individual look for each side of the border, this property can be used.

The border direction values are: top/ bottom/ right/ left

Example:

- border-bottom-style: solid;
- border-bottom-color: red;
- border-bottom-width: 5 px;
- border-All in one

It is used to create a uniform border.

Example:

- border: 10px outset green;
- border: 20px solid;


```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  border: 5px solid red;
}

h2 {
  border: 4px dotted blue;
}

h3 {
  border: double;
}
</style>
</head>
<body>

<h1>A heading with a solid red border</h1>

<h2>A heading with a dotted blue border</h2>

<h3>A heading element with a double border.</h3>

</body>
</html>
```

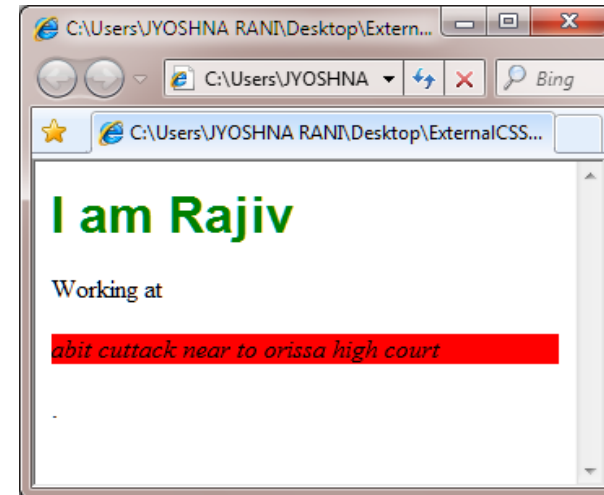


The style sheets support classes or set of style changes for documents. A class can be defined to change the style in a specific way for any element is applied to, and classes can be used to identify logical sets of styles changes that might be different HTML element

The style changes can be applied directly to each HTML element or applied to part of a document with the `<style>` `</style>` tags.

If any element is made a member of a class by inserting `class = <classname>` into opening tag , it conforms to that class's specification.

```
<html>
<head>
<style type="text/css">
.xyz { font-family:arial,helvetica; color:green }
  p   { font-size:12pt;font-style:italic;background-color:red }
</style>
</head>
<body>
<H1 class = "xyz"> I am Rajiv </h1>
Working at <p> abit cuttack near to orissa high court</p>.
</body>
</html>
```



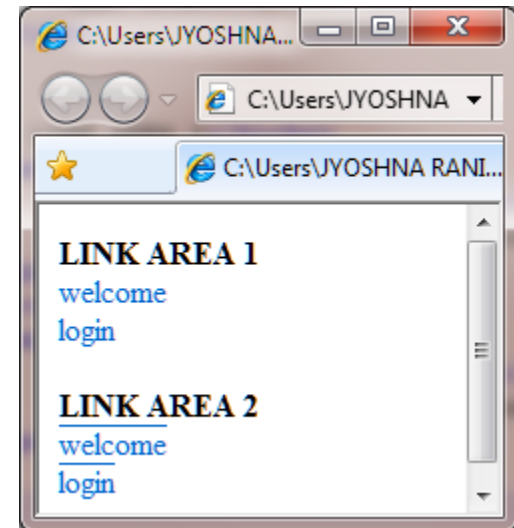
** tag**

- The span is similar to the div tag. Both of them divide the content into individual sections.
- The difference is that span goes into a refined level so that by using span a single character is formatted if required.
- It is used for text-level element and not for a block-level element.
- There is no line feed after the tag.
- It is used as a selector in style sheet and it includes STYLE class and ID as its attribute.
- It is a CSS element and used purely to apply style.
- It has no effect when style sheet is disabled.

```

<html><head>
  <style type="text/css">
    .typ1 A:link{text-decoration:none}
    .typ1 A:visited{text-decoration:none; color: red}
    .typ1 A:hover{text-decoration:underline; color:green}
    .typ1 A:active{text-decoration:none; color:purple}
    .typ2 A:link{text-decoration: overline}
    .typ2 A:visited{text-decoration:overline; color: red}
    .typ2 A:hover{text-decoration:underline; color:green}
    .typ2 A:active{text-decoration:overline; color:purple}
  </style>
</head>
<body>
  <b> LINK AREA 1</b><br>
  <span class="typ1">
    <a href="color.html">welcome</a><br>
    <a href="textcolor.html">login</a>
  </span>
  <br><br><b> LINK AREA 2</b><br>
  <span class="typ2">
    <a href="color.html">welcome</a><br>
    <a href="textcolor.html">login</a>
  </span>
</body></html>

```



- It divides the content into individual sections.
- <div> may contain paragraphs, headings, tables and other division also.
- Each section can have its own formatting.
- It is a block-level element which means that there can be a line feed after the</div>tag.
- The <div> elements useful for marking large section of a document.

Example of div tag

```
<html>
  <head>
    <style type="text/css">
      .emb{border: double; font-variant:small-caps; border-color: green;color:red; width:250px;}
    </style>
  </head>
  <body>
    <h2>Example of div element</h2>
    <div class="emb">The div is a block level element</div>
  </body>
</html>
```

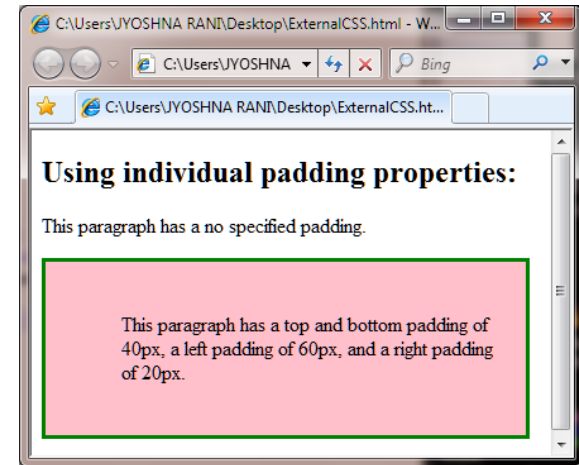


This property specifies how much space should appear between the content of an element and its border.

The values of padding properties are:

- padding-bottom
- padding-top
- padding-left
- padding-right
- padding

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      p.padding
      {
        border: 3px solid green;
        background-color: pink;
        padding-top: 40px;
        padding-right: 20px;
        padding-bottom: 40px;
        padding-left: 60px;
        width: 300px;
      }
    </style>
  </head>
  <body>
    <h2>Using individual padding properties:</h2>
    <p>This paragraph has a no specified padding.</p>
    <p class="padding">This paragraph has a top and bottom padding of 40px, a
left padding of 60px, and a right padding of 20px.</p>
  </body>
</html>
```



HTML Lists and CSS List Properties

In HTML, there are two main types of lists:

- unordered lists () - the list items are marked with bullets
- ordered lists () - the list items are marked with numbers or letters

The CSS list properties allow you to:

- Set different list item markers for ordered lists
- Set different list item markers for unordered lists
- Set an image as the list item marker
- Add background colors to lists and list items

Different List Item Markers

The list-style-type property specifies the type of list item marker.

```
<!DOCTYPE html>
<html>
<head>
<style>
ul.a {
    list-style-type: circle;
}

ul.b {
    list-style-type: square;
}

ol.c {
    list-style-type: upper-roman;
}

ol.d {
    list-style-type: lower-alpha;
}
</style>
</head>
```

```
<body>
<p>Example of unordered lists:</p>
<ul class="a">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

<ul class="b">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

<p>Example of ordered lists:</p>
<ol class="c">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

<ol class="d">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<style>
ol {
    background: #ff9999;
    padding: 20px;
}

ul {
    background: #3399ff;
    padding: 20px;
}

ol li {
    background: #ffe5e5;
    padding: 5px;
    margin-left: 35px;
}

ul li {
    background: #cce5ff;
    margin: 5px;
}
</style>
</head>
```

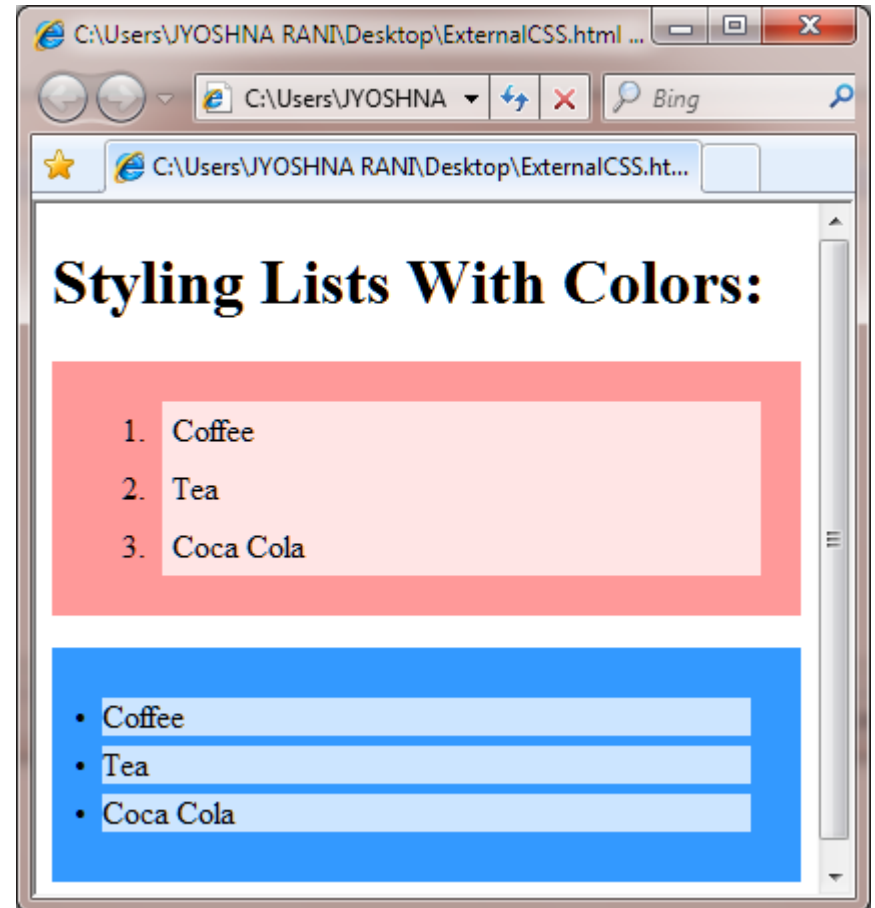
```
<body>

<h1>Styling Lists With Colors:</h1>

<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

</body>
</html>
</head>
```



```
<!DOCTYPE html>
<html>
<head>
<style>
.center {
    text-align: center;
}

.pagination {
    display: inline-block;
}

.pagination a {
    color: black;
    float: left;
    padding: 8px 16px;
    text-decoration: none;
    transition: background-color .3s;
    border: 1px solid #ddd;
    margin: 0 4px;
}

.pagination a.active {
    background-color: #4CAF50;
    color: white;
    border: 1px solid #4CAF50;
}

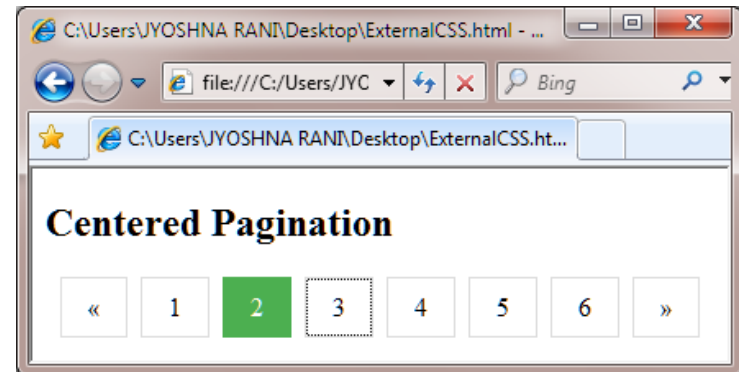
.pagination a:hover:not(.active) {background-color: #ddd;}
</style></head>
```

```
<body>

<h2>Centered Pagination</h2>

<div class="center">
  <div class="pagination">
    <a href="#">&laquo;</a>
    <a href="#">1</a>
    <a href="#" class="active">2</a>
    <a href="#">3</a>
    <a href="#">4</a>
    <a href="#">5</a>
    <a href="#">6</a>
    <a href="#">&raquo;</a>
  </div>
</div>

</body>
</html>
```



DHTML

Dynamic Hyper Text Markup Language

What is DHTML ?

DHTML called Dynamic Hyper Text Markup Language used for creating interactive and dynamic websites. DHTML is a combination of various technologies that are gathered as under one roof. Combination of these various technologies are the HTML (static markup language), JavaScript (client-side scripting), CSS (presentation definition language), DOM (Document Object Model).

Uses of DHTML

- It is used for designing the animated and interactive web pages that are developed in real-time.
- DHTML helps users by animating the text and images in their documents.
- It allows the authors for adding the effects on their pages.
- It also allows the page authors for including the drop-down menus or rollover buttons.
- This term is also used to create various browser-based action games.
- It is also used to add the ticker on various websites, which needs to refresh their content automatically.

Features of DHTML

- Its simplest and main feature is that we can create the web page dynamically.
- Dynamic Style is a feature, that allows the users to alter the font, size, color, and content of a web page.
- It provides the facility for using the events, methods, and properties. And, also provides the feature of code reusability.
- It also provides the feature in browsers for data binding.
- Using DHTML, users can easily create dynamic fonts for their web sites or web pages.
- With the help of DHTML, users can easily change the tags and their properties.
- The web page functionality is enhanced because the DHTML uses low-bandwidth effect.

HTML (Hypertext Markup language)	DHTML (Dynamic Hypertext Markup language)
1. HTML is simply a markup language.	1. DHTML is not a language, but it is a set of technologies of web development.
2. It is used for developing and creating web pages.	2. It is used for creating and designing the animated and interactive web sites or pages.
3. This markup language creates static web pages.	3. This concept creates dynamic web pages.
4. It does not contain any server-side scripting code.	4. It may contain the code of server-side scripting.
5. The files of HTML are stored with the .html or .htm extension in a system.	5. The files of DHTML are stored with the .dhtm extension in a system.
6. A simple page which is created by a user without using the scripts or styles called as an HTML page.	6. A page which is created by a user using the HTML, CSS, DOM, and JavaScript technologies called a DHTML page.
7. This markup language does not need database connectivity.	7. This concept needs database connectivity because it interacts with users.

Components of Dynamic HTML

DHTML consists of the following four components or languages:

- HTML 4.0
- CSS
- JavaScript
- DOM.

HTML 4.0

HTML is a client-side markup language, which is a core component of the DHTML. It defines the structure of a web page with various defined basic elements or tags.

JavaScript

JavaScript is a scripting language which is done on a client-side. The various browser supports JavaScript technology. DHTML uses the JavaScript technology for accessing, controlling, and manipulating the HTML elements. The statements in JavaScript are the commands which tell the browser for performing an action.

Dynamic HTML(DHTML) = HTML 4.0 + JavaScript
= DHTML JavaScript

DHTML JavaScript

JavaScript can be included in HTML pages, which creates the content of the page as dynamic. We can easily type the JavaScript code within the <head> or <body> tag of a HTML page. If we want to add the external source file of JavaScript, we can easily add using the <src> attribute

Document.write() Method : The document.write() method of JavaScript, writes the output to a web page.

Example : The example uses the **document.write()** method of JavaScript in the DHTML. In this example, we type the JavaScript code in the <body> tag.

```
<html>
```

```
<body>
```

```
<script type="text/javascript">
```

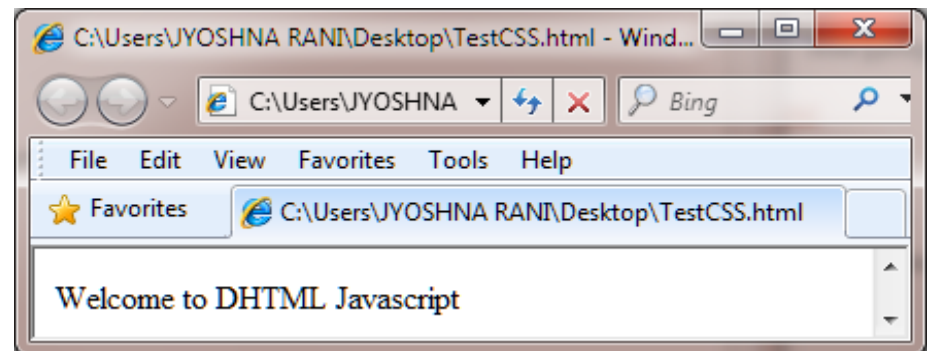
```
    document.write("Welcome to DHTML Javascript");
```

```
</script>
```

```
</body>
```

```
</html>
```

Save As : First.html



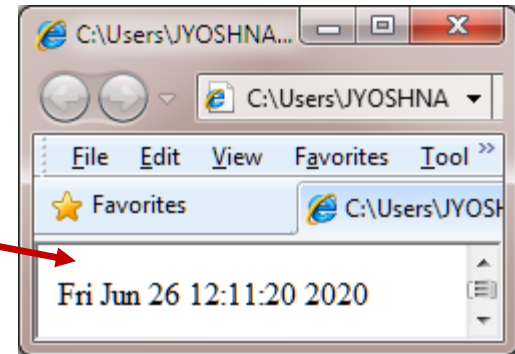
Example

The example uses JavaScript to display the current date and time on a page:

```
<html>
<body>

<script type="text/javascript">
    document.write(Date());
</script>

</body>
</html>
```

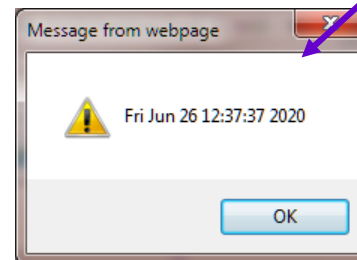
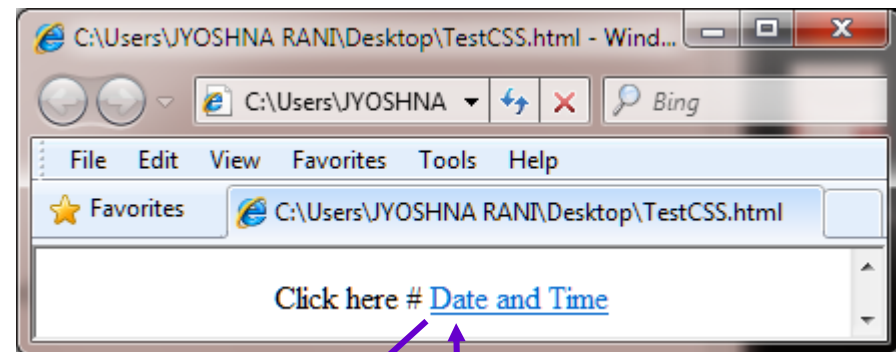


JavaScript and HTML event

A JavaScript code can also be executed when some event occurs. Suppose, a user clicks an HTML element on a webpage, and after clicking, the JavaScript function associated with that HTML element is automatically invoked. And, then the statements in the function are performed.

Example : The example shows the current date and time with the JavaScript and HTML event (OnClick).

```
<html>
<body>
<script type="text/javascript">
function dateandtime()
{
alert(Date());
}
</script>
<center> <p>
Click here # <a href="#" onClick="dateandtime();" > Date and Time </a>
</p> </center>
</body>
</html>
```

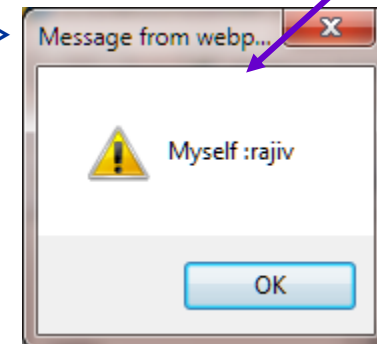
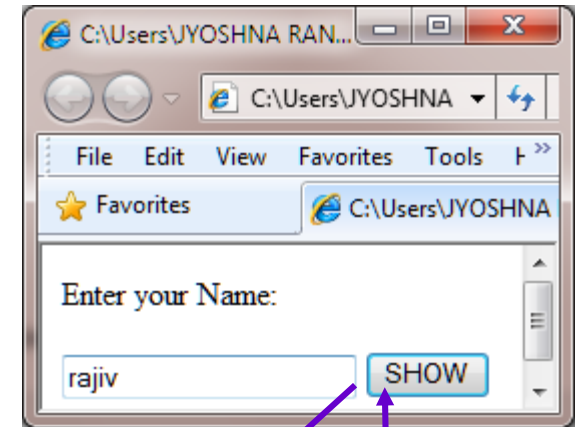


Click on

JavaScript and HTML DOM

With version 4 of HTML, JavaScript code can also change the inner content and attributes of the HTML event.

```
<html>
<body>
<script type="text/javascript">
function show()
{
var x,p;
p = document.getElementById("percentage").value;
alert("Myself :"+p);
}
</script>
<p>Enter your Name:</p>
<input type="text" id="percentage">
<input type="button" value = "SHOW" onclick="show()">
</body>
</html>
```



Click on

CSS

CSS stands for Cascading Style Sheet, which allows the web users or developers for controlling the style and layout of the HTML elements on the web pages.

DHTML CSS

We can easily use the CSS with the DHTML page with the help of JavaScript and HTML DOM.

With the help of `this.style.property=new style statement`, we can change the style of the currently used HTML element.

DHTML CSS = DHTML + CSS

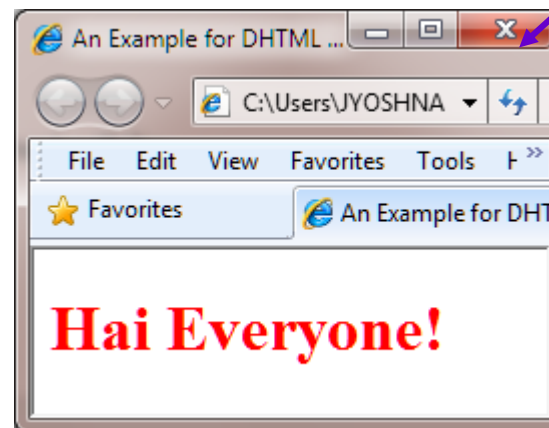
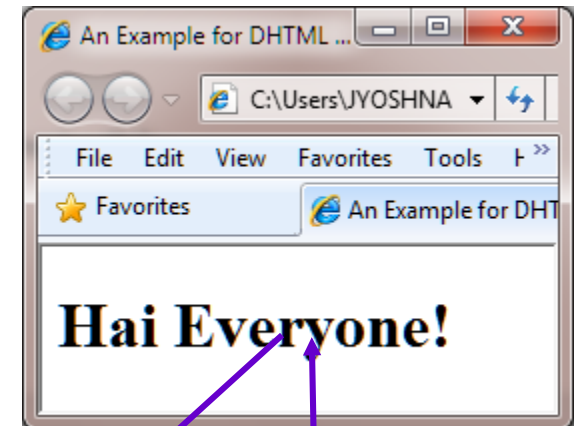
Or,

we can also update the style of any particular HTML element by `document.getElementById(id).style.property = new_style statement`.

DHTML CSS = DHTML + CSS + JAVASCRIPT

Example : The following example uses the DHTML CSS for changing the style of current element:

```
<!DOCTYPE html>
<html lang="en-US">
<head>
<title>An Example for DHTML CSS</title>
</head>
<body>
<h1 onclick="this.style.color='red'">Hai Everyone!</h1>
</body>
</html>
```



Click on

Change Style of a Specific HTML Element

To change the style of a specific HTML element, use the following statement:

```
document.getElementById(id).style.property=new style
```

Example

```
<html>  
<body>  
<h1 id="h1" onclick="document.getElementById('h1').style.color='red'">Hai Everyone!</h1>  
</body>  
</html>
```

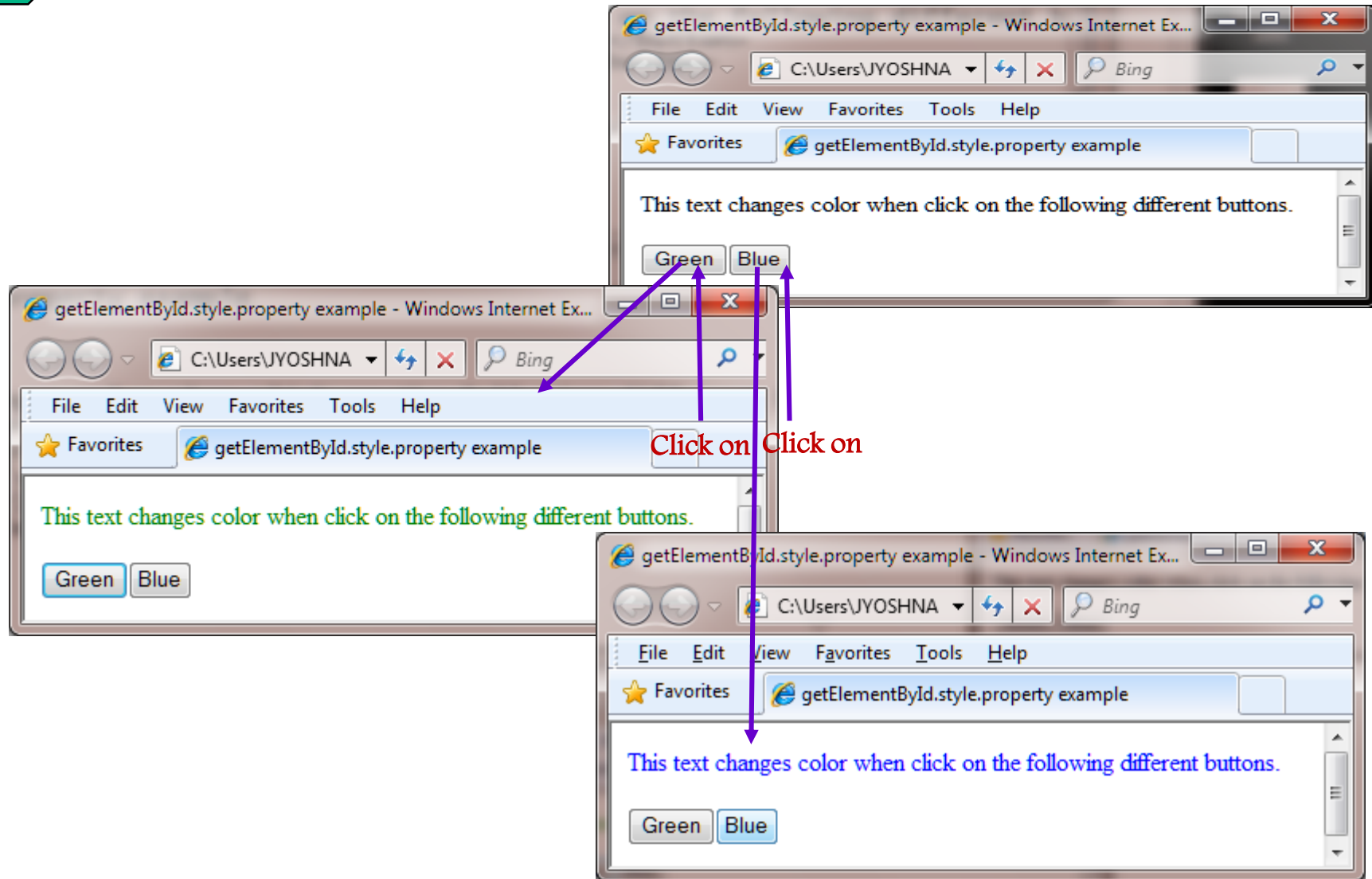


CSS with JavaScript in DHTML

With version 4 of HTML, JavaScript code can also change the style such as size, color, and face of an HTML element.

Example : The following example changes the color of a text.

```
<html>
<head>
<title>
    getElementById.style.property example
</title>
</head>
<body>
    <p id="demo">
        This text changes color when click on the following different buttons.
    </p>
    <button onclick="change_Color('green');"> Green </button>
    <button onclick="change_Color('blue');"> Blue </button>
<script type="text/javascript">
function change_Color(newColor) {
    var element = document.getElementById('demo').style.color = newColor;
}
</script>
</body>
</html>
```



DHTML Events

- An event is defined as changing the occurrence of an object.
- It is compulsory to add the events in the DHTML page. Without events, there will be no dynamic content on the HTML page. The event is a term in the HTML, which triggers the actions in the web browsers.
- Suppose, any user clicks an HTML element, then the JavaScript code associated with that element is executed. Actually, the event handlers catch the events performed by the user and then execute the code.

Example of events:

- Click a button.
- Submitting a form.
- An image loading or a web page loading, etc.

Following table describes the Event Handlers used in the DHTML:

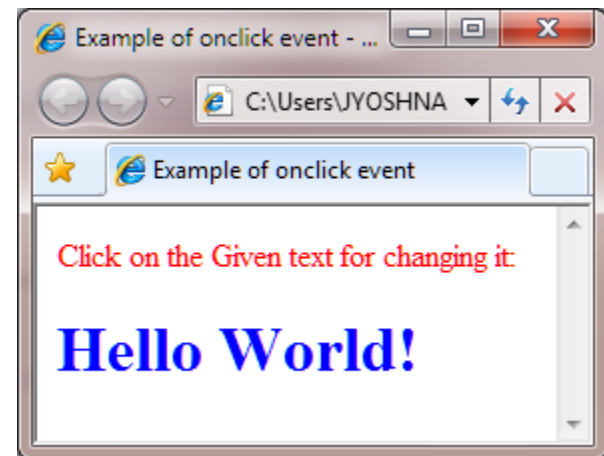
Event	When it occurs
onabort	It occurs when the user aborts the page or media file loading.
onblur	It occurs when the user leaves an HTML object.
onchange	It occurs when the user changes or updates the value of an object.
onclick	It occurs or triggers when any user clicks on an HTML element.

Event	When it occurs
ondblclick	It occurs when the user clicks on an HTML element two times together.
onfocus	It occurs when the user focuses on an HTML element. This event handler works opposite to onblur.
onkeydown	It triggers when a user is pressing a key on a keyboard device. This event handler works for all the keys.
onkeypress	It triggers when the users press a key on a keyboard. This event handler is not triggered for all the keys.
onkeyup	It occurs when a user released a key from a keyboard after pressing on an object or element.
onload	It occurs when an object is completely loaded.
onmousedown	It occurs when a user presses the button of a mouse over an HTML element.
onmousemove	It occurs when a user moves the cursor on an HTML object.
onmouseover	It occurs when a user moves the cursor over an HTML object.
onmouseout	It occurs or triggers when the mouse pointer is moved out of an HTML element.
onmouseup	It occurs or triggers when the mouse button is released over an HTML element.
onreset	It is used by the user to reset the form.
onselect	It occurs after selecting the content or text on a web page.
onsubmit	It is triggered when the user clicks a button after the submission of a form.
onunload	It is triggered when the user closes a web page.
Event	When it occurs
onabort	It occurs when the user aborts the page or media file loading.

Event	When it occurs
onblur	It occurs when the user leaves an HTML object.
onchange	It occurs when the user changes or updates the value of an object.
onclick	It occurs or triggers when any user clicks on an HTML element.
ondblclick	It occurs when the user clicks on an HTML element two times together.
onfocus	It occurs when the user focuses on an HTML element. This event handler works opposite to onblur.
onkeydown	It triggers when a user is pressing a key on a keyboard device. This event handler works for all the keys.
onkeypress	It triggers when the users press a key on a keyboard. This event handler is not triggered for all the keys.
onkeyup	It occurs when a user released a key from a keyboard after pressing on an object or element.
onload	It occurs when an object is completely loaded.
onmousedown	It occurs when a user presses the button of a mouse over an HTML element.
onmousemove	It occurs when a user moves the cursor on an HTML object.
onmouseover	It occurs when a user moves the cursor over an HTML object.
onmouseout	It occurs or triggers when the mouse pointer is moved out of an HTML element.
onmouseup	It occurs or triggers when the mouse button is released over an HTML element.
onreset	It is used by the user to reset the form.
onselect	It occurs after selecting the content or text on a web page.
onsubmit	It is triggered when the user clicks a button after the submission of a form.
onunload	It is triggered when the user closes a web page.

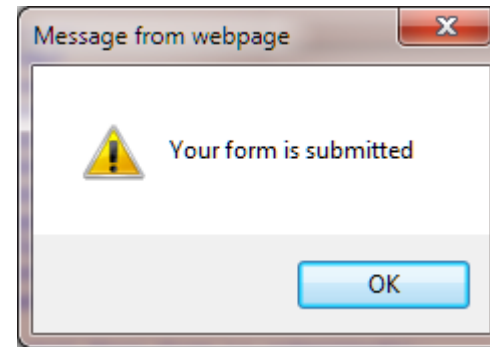
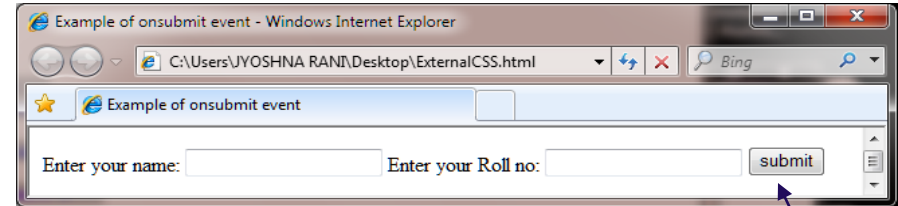
Example of onclick event handler

```
<html>
<head>
<title> Example of onclick event </title>
<script type="text/javascript">
function ChangeText(ctext)
{
ctext.innerHTML=" Hi Hello Rajiv! ";
}
</script>
</head>
<body>
<font color="red"> Click on the Given text for changing it: <br>
</font>
<font color="blue">
<h1 onclick="ChangeText(this)"> Hello World! </h1>
</font>
</body>
</html>
```



Example of onsubmit event handler

```
<html>
<head>
<title>
Example of onsubmit event
</title>
</head>
<body>
<form onsubmit="Submit_Form()">
<label> Enter your name: </label>
<input type="text">
<label> Enter your Roll no: </label>
<input type="Number">
<input type="submit" value="submit">
</form>
<script type="text/javascript">
function Submit_Form()
{
alert(" Your form is submitted");
}
</script>
</body>
</html>
```

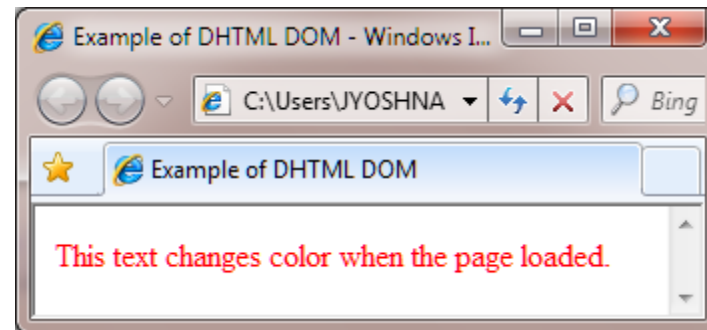


Click on

DHTML DHTML DOM

- DHTML DOM stands for Dynamic HTML Document Object Model.
- It is a w3c standard, which is a standard interface of programming for HTML. It is mainly used for defining the objects and properties of all elements in HTML. It also defines the methods for accessing the HTML elements.

```
<html>
<head>
  <title>
Example of DHTML DOM
  </title>
</head>
<body>
<font color = "blue">
  <p id="demo"> This text changes color when the page loaded. </p>
</font>
<script type="text/javascript">
document.getElementById('demo').style.color = "red";
</script>
</body>
</html>
```



Advantages of DHTML

The various benefits or the advantages of DHTML (Dynamic HTML):

- Those web sites and web pages which are created using this concept are fast.
- There is no plug-in required for creating the web page dynamically.
- Due to the low-bandwidth effect by the dynamic HTML, the web page functionality is enhanced.
- This concept provides advanced functionalities than the static HTML.
- It is highly flexible, and the user can make changes easily in the web pages.

Disadvantages of DHTML

The various disadvantages or limitations of DHTML (Dynamic HTML):

- The scripts of DHTML does not run properly in various web browsers. Or in simple words, we can say that various web browsers do not support the DHTML. It is only supported by the latest browsers.
- The coding of those websites that are created using DHTML is long and complex.
- For understanding the DHTML, users must know about HTML, CSS, and JavaScript. If any user does not know these languages, then it is a time-consuming and long process in itself.

The XML logo is located in the top-left corner. It consists of a teal octagon with the letters 'XML' in red, serif font. To the right of the octagon is a long, horizontal teal bar with a gradient from dark teal on the left to light teal on the right, ending in a rounded right side.

XML

- XML stands for Extensible Markup language.
- It was developed by W3C (World Wide Web Consortium).
- XML is a self-descriptive language.
- XML is used to store and carry data.

Applications of XML

- Mobiles: XML is used to display the text, images and sounds in the mobile softwares.
- Converters: Converters are used to convert existing documents into XML format.
- Example: PDF to XML.
- Voice XML: Converts the XML documents into an audio format.

Features of XML

- XML provides domain specific self describing tags.
- It carries data with HTML.
- XML allows interchanging of data between different computer application.
- User defined tags in XML allow to search.

XML	HTML
Designed to transport and store data.	Designed to display data.
Provides granular update facility.	Granular update facility is unavailable.
XML is case sensitive.	HTML is not case sensitive.

- Write XML code in the notepad and save with .xml extension.
- To view XML file, open it in web browsers.

XML document consists of a number of components used to represent information in hierarchical manner. These components are as follows.

1. Processing Instruction

- XML document begins with processing instruction (PI).
- PI indicates the version of XML and it is optional.

Example

PI statement: `<?xml version="1.0"?>`

Rules of XML PI declaration

- a) XML declaration are case sensitive.
- b) XML declaration appear at very first line.

2. Tags

- Tags are used to identify the data.
- XML tags begin with "<" and end with ">"

Start tag: `<element>`

End tag: `</element>`

Example

`<person> Name</person>`

3. Elements

Elements are used to identify and describe the XML data.
Start with a start tag, <element> and End with </element>

Example

```
<person><info>Employee</info>Ricky</person>
```

Rules to write XML elements

a) Start and end tags must be the same.

Example:

```
<person><info>student</info>Bob</person>
```

b) XML document must have exactly one root element.

Example:

```
<c>
```

```
  <a>...</a>
```

```
  <b>...</b>
```

```
</c>
```

In the above example, c is root element and elements a and b are within the root element c

c) XML elements are case sensitive.

Example:

```
<name>Bob</name> and not as <Name>Bob</name>
```

4. Attributes

Attributes are name-value pairs and occur inside start-tags after the element name.

Example:

```
<student gender =“male”>
```

In above example, gender is an attribute and student is an element.

5. Entities

- Entities are used to represent the special characters.
- Every entity should have a unique name.
- Entity names begin with the &(ampersand) and end with semicolon(;).

Internal Entities:

- Internal entities associate a name with string of literal text.
- Internal entities define the shortcuts for frequently typed text or frequently changing text.

The XML predefines internal entities which are as follows:

- a) & lt; used for less than symbol, <
- b) & gt; used for greater than symbol, >
- c) & amp; produces ampersand , &
- d) & apos; used for a single quote character (apostrophe),'
- e) & quot; used for double quote character, “ ”

Example : Internal entities

```
<NUMBERS>
```

```
  <LARGE>50 & gt; 5</LARGE>
```

```
  <SMALL> 5 & lt; 50</SMALL>
```

```
</NUMBERS>
```

Output:

number entities

6. Comments

XML comments begin with <!-- and End with - - >

Comments are not part of the textual content of XML document.

Syntax

```
<!-- - - -comment- - ->
```

7. Content

Content represents the information of elements in XML document.

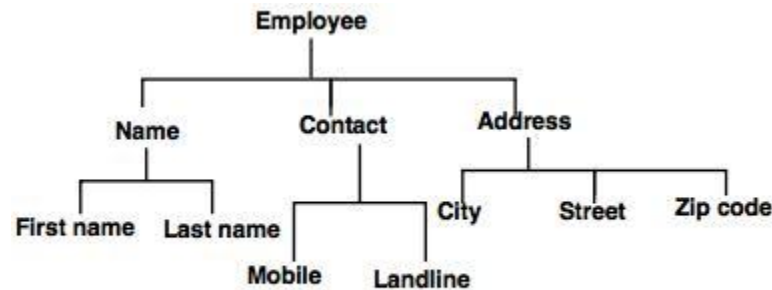
Example

```
<Bookname>Java and XML</Bookname>
```

XML Tree Structure

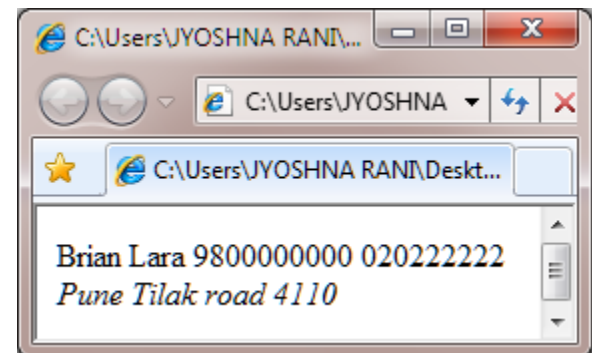
- XML tree structure is also called as tree model or hierarchical model.
- XML document must contain a root element and all elements in the document can contain sub-elements, text and attributes.
- The complex elements are represented with the help of internal nodes and simple elements are represented with leaf node.

Example: The employee information is represented with tree structure, which can be used in XML document.



XML tree structure

```
<Employee>
  <Name>
    <First name>Brian</First name>
    <Last name>Lara</Last name>
  </Name>
  <Contact>
    <Mobile>9800000000</Mobile>
    <Landline>020222222</Landline>
  </Contact>
  <Address>
    <City>Pune</city>
    <Street>Tilak road</Street>
    <Zip code>4110</Zip code>
  </Address>
</Employee>
```



XML Document Type Definition (XML DTD)

- DTD stands for Document Type Definition.
- It is used to validate the XML documents.
- XML provides facility to create your own DTDs for XML document.
- DTD specifies the structure of the XML document.
- DTD is part of the file or separate document file.

Types of DTD

There are two types of DTDs:

- a) Internal DTD.
- b) External DTD.

Internal DTD vs External DTD.

Internal DTD	External DTD
Internal DTD is part of the document.	External DTD is a separate file.
The scope is limited to the document which is created.	It is accessible across the multiple documents.
Syntax: <!DOCTYPE rootelement[element and attribute declarations]>	Syntax: <!DOCTYPE rootelement SYSTEM "path of .dtd file">

XML Document Type Definition (XML DTD) (conti.)

Components of DTD

We have to associate the .xml file with the DTD and run the same .xml file in the browser.

Syntax for declaring elements in DTD:

```
<!ELEMENT elementname (content-type or content-model)>
```

Where elementname specifies the name of the element present in the xml document and content-type or content-model specifies whether the element contains textual data or other elements.

The three types of elements are as follows:

Element type	Description	DTD Declaration
Empty	Contains attributes, but can't contain text or any other element.	<!Element elementname EMPTY>
Unrestricted	Contains text content or any other element.	<!ELEMENT elementname ANY>
Container	Contain another elements.	<!ELEMENT elementname (elementname, elementname [elementname])>

Declaring Attributes in DTD

Syntax:

```
<!!ATTLIST elementname atributename valuetype [atributetype] ["default"]>
```

Each attribute declaration must include at least the attribute name in a DTD as follows:

Value Type	Description
PCDATA	Represents the plain text values.
ID	Assigns unique value to each element in the document.
(enumerated)	Assigns a specific range of values which is specified within parenthesis.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
  <!DOCTYPE Employee
  [
    <!ELEMENT address (name,company,phone)>
    <!ELEMENT name (#PCDATA)>
    <!ELEMENT company (#PCDATA)>
    <!ELEMENT phone (#PCDATA)>
  ]>
  <address>
    <name>Mayuri S</name>
    <company>ABIT</company>
    <phone>91-9800000000</phone>
  </address>
```

What is XML schema?

- XML schema defines the structure of an XML document.
- A schema defines, list of elements, attributes used in XML documents and data type of these elements.
- It is known as XSD.

XSD vs DTD

XSD	DTD
Defines list, order, data types of elements and attributes.	Defines list, order of elements and attributes.
Provides control over the elements and attributes used in XML documents.	It does not provide control over elements and attributes.
XSD allows to create customized datatype.	DTD does not allow to create customized datatype.
Syntax of XSD is similar to XML document.	Syntax of DTD is different from XML document.
XSD allows to define restrictions on data. For example: Define the content in a document by using only integer data type.	DTD does not allows to define restrictions on data.

Declaring Elements in XSD

Syntax: `<xsd:element name =“elementname” type=“datatype” minOccurs = “notNegativeInteger” maxOccurs=“nonNegativeInteger | unbounded”/>`

Where,

name: Defines the element name

type: Defines data type.

minOccurs: If its value is zero, the use of element is optional and if its value is greater than zero, the use of element is compulsory, and should occur at least for specified number of times.

maxOccurs: If value is set as unbounded, the use of element can appear any number of times in the XML document without any limitation.

XSD for simple type elements using predefined datatypes

```
<xsd:element name="CARNAME" TYPE="xsd:string"/>
<xsd:element name="COMPANYNAME" TYPE="xsd:string"/>
<xsd:element name="PRICE" TYPE="xsd:positiveInteger"/>
```

- XML document with simple type elements and associated XSD using creation of custom data type.

```
<EMPLOYEEENAM>Surabhi</EMPLOYEEENAM>
<EMPLOYEEEPHONE>9800000000</EMPLOYEEEPHONE>
<xsd:element name="EMPLOYEEENAM" TYPE="xsd:string"/>
<xsd:element name="EMPLOYEEEPHONE" TYPE="xsd:string"/>
  <xsd:simpleType name="phoneno">
    <xsd:restriction base="xsd:string">
      <xsd:length value="15"/>
      <xsd:pattern value="\d{3}-\d{6}"/>
    </xsd:restriction>
  </xsd:simpleType>
```

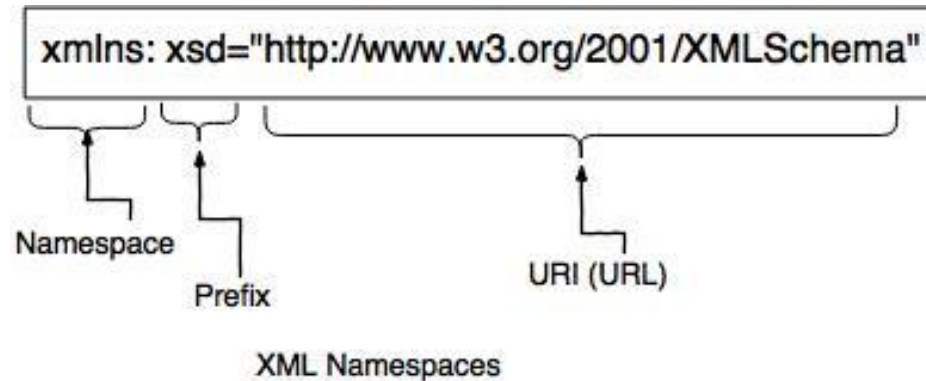
In above example, the XSD defines the custom data type "phoneno".

The data type "phoneno" has restrictions as, it can hold predefined data type string and it should be 12 characters long and \d presents 'digits'.

Inbuilt characters classes commonly used in XSD.

Class	Method	Meaning
[0-9]	\d	Any digit.
[\f\r\t\n\v]	\s	Any white space.
[A-Za-z0-9]	\w	Any word character.
[^0-9]	\D	Not a digit.
[^\f\r\t\n\v]	\S	Not a White space.
[^A-Za-z0-9]	\W	Not a word character.

- XML Namespace is a unique name given to group of elements and attributes to avoid naming collisions.
- URI (Uniform Resource Identifier) is used to assign namespace.



- The keyword xmlns is used to declare namespace in an XSD document.
- A namespace is declared at the beginning of the document.

Syntax:

```
<xmlns:prefix="URI">
```

Where,

- xmlns is the name of the attribute.
- prefix is optional.
- URI is the value of the xmlns attribute which is associated with XSD document.

There are two ways of declaring namespaces:

1. Default declaration: In this type of declaration prefix is not used.
2. Explicit declaration: In this type of declaration, the prefix is defined with xmlns and with URI.

Default declaration

```
<xml version="1.0 encoding="UTF-8?">
<schema xmlns="w3.org.com/XMLSchema">
  <name>Mayuri</name>
  <company>CareerRide</company>
  <phone>9800000000</phone>
</information>
```

In above example, a prefix is not defined with xmlns and URI.

Explicit declaration

```
<xml version="1.0 encoding="UTF-8?">
<schema:information xmlns:info="w3.org.com/XMLSchema">
  <info:name>Mayuri</info:name>
  <info:company>CareerRide</info:company>
  <info:phone>9800000000</info:phone>
</info:information>
```

In above example prefix info is used with xmlns and URI.

What is XSL?

CSS has drawbacks, when linked with XML documents like, sorting and reordering elements based on a condition and displaying only sensitive elements which is possible in XSL.

XSL is made of two parts

a) XSL Transformation (XSLT)

Transformation between two XML is possible using XSLT.

b) XML Path (XPath)

XML Path is an XML based language which is used to access elements and attributes present in the XML document.

CSS vs XSLT

CSS	XSLT
CSS is unable to perform operations on elements like add, delete, reordering etc.	XSLT is able to perform operations on elements like add, delete, reordering.
Access to PI and attributes of XML document is not possible in CSS.	XSLT allows to access and manipulate the comments, PI, attribute names and values within the XML document.
Uses less memory.	Requires more memory to manipulate the document.
Syntax is different than XML.	Syntax is similar to XML.
It is simple to use and suitable for only small documents.	It is complex to use.

XSLT elements are used for selecting and formatting data.

1. Stylesheet element: Stylesheet declaration statement is used to inform the browser that, this is a style sheet file.

Syntax

```
<xsl:stylesheet xmlns:xsl= "http://www.w3.org/XSL/Transform" version="1.0">
```

2. Value-of element: Value-of element returns the value of specified element or an attribute.

Syntax

Value of Element: `<xsl:value-of select="elementname"/>`

Value of Attribute: `<xsl:value-of select="@attributename"/>`

3. For-each element: It instructs the XSLT processor to process the information for each instance of the specified pattern.

Syntax

```
<xsl:for-each select="pattern">
```

```
  <!--XSLT code-->
```

```
</xsl:for-each>
```

4. Sort element: It sorts the data depending on the values assigned to elements and attributes.

Syntax

```
<xsl:sort select="expression" order="ascending|descending"  
case-order="upper-first|lower-first"  
data-type="text|number|userdefined"/>
```

Where,

Select: Represents the element name.

Order: Represents the sort order and the default value is ascending.

case-order: Represents the data type of the data to be sorted and default value is text.

data-type: Represents the data type of the data to be sorted and default value is text.

5. Text element: This element generates constant text in the output and used to display labels.

Syntax

```
<xsl:text>Name:</xsl:text>
```

Explain the transformation of XML document using XSLT

Step 1: Create an XML document which contains information of employee and save with employee.xml.

```
<?xml version="1.0"?>
<class>
  <Employee Id="101">
    <firstname>Nirja</firstname>
    <lastname>Shah</lastname>
    <salary>85000</salary>
  </Employee>
  <Employee Id="102">
    <firstname>Prashant</firstname>
    <lastname>Saxena</lastname>
    <salary>95000</salary>
  </Employee>
</class>
```

In above XML document we have mentioned a employee information.

Step 2: Create XSLT document to fulfill the above requirement and save with employee.xsl in same folder along with employee.xml file.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html><body>
  <h2>Employee</h2>
  <table>
    <tr>
      <th>Id</th>
      <th>FirstName</th>
      <th>LastName</th>
      <th>Salary</th>
    </tr>
    <xsl:for-each select="class/Employee">
      <tr>
        <td><xsl:value-of select="@Id"/></td>
        <td><xsl:value-of select="firstname"/></td>
        <td><xsl:value-of select="lastname"/></td>
        <td><xsl:value-of select="salary"/></td>
      </tr>
    </xsl:for-each>
  </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Step 3: Link the XSLT document to XML document by following xml-stylesheet tag.
<?xml-stylesheet type="text/xsl" href="employee.xsl"?>

Step 4: Run the employee.xml file in the browser.

Output:

Employee

Id	FirstName	LastName	Salary
101	Nirja	Shah	85000
102	Prashant	Saxena	95000

What is XML Parsers?

- XML parsers are libraries used for checking and validating XML document schema.

DOM Parser


- DOM (Document Object Model) provides a programming interface for manipulating XML documents.
- DOM includes a set of objects and interfaces, which represent the content and structure of an XML document and enables a program to traverse XML tree structure.
- DOM allows to create new XML document with programming interfaces, for example, we can create XML document through ASP.net.

SAX Parser

- SAX stands for Simple API for XML.
- API (Application Programming interface) allows programmer to read and write XML data.
- SAX parser is based on events.
- SAX Parser follows a push model which allows sequential access.

SAX Parser vs DOM Parser.

SAX Parser	DOM Parser
Instead of creating internal structure in the document, it takes the occurrences of components as events.	DOM parser creates a tree structure in memory from an input document and waits for client request.
SAX Parser allows to extract the information in the document according to the users interest.	User can not extract the information of his interest.
SAX parser is space efficient.	DOM Parser is space inefficient.
Users have to create their own data structures.	DOM parser allows user to access any part of the document and modify the DOM tree.



Thank
You!