CASCADING STYLE SHEET

Cascading Style Sheets (CSS) is a language that allows us to define how the **HTML** document to appear. This concerns features such as typeface, background, link colors, margins, and placement of objects on a page. The set of instructions for these features are called **"style rules"** in CSS language. They allow web page designers to improve and change the appearance of their web pages very efficiently. Cascading Style Sheet (CSS) provides the way of incorporating style information in a style sheet. The term **"cascading"** indicates that several style sheets can be blended to present a document on the browser's screen.

A **style sheet** is a document that contains style information about one or more documents written in markup languages. It enables us to control rendering of styles such as fonts, color, typeface, size, spacing, margins, and other aspects of documents style. A style sheet is composed of a set of style rules written in a specified format. This set of style rules instructs browsers on how to present a document on the screen.

Advantages of CSS

- ✓ The primary intention of **CSS** is to separate document presentation from document content written in markup languages.
- ✓ CSS will allow designers to create tighter, more dynamic content for the Web.
- ✓ Web pages using CSS will transfer faster to users and will be easy to maintain.
- ✓ CSS will lead to improved accessibility, maintainability and performance on the Web.

CSS Syntax

There are two parts in a style rule. The ***selector*** determines what HTML tag we are styling, and the ***declaration*** is made up of one or more ***property: value*** pairs, ending in a semi-colon and enclosed in curly brackets. In this case a *property* is a display feature such as color, and a *value* is what we want that property to look like.

Syntax:

```
selector {
property1: value1;
property2: value2;
……
propertyN: valueN;
}
```

For example,

```
h1      {  color:    red;    }
selector   property  value
           |_____declaration_____|
```

means "All <h1> headers' text color to be red in entire HTML Document."

LINKING CSS TO A WEB DOCUMENT

For any style sheet to affect an HTML document, it must be attached. This can be done in several ways, depending on which kind of style sheet we are attaching. There are three basic methods of embedding our CSS into the HTML.

- **External style sheet**
- **Embedded style sheet**
- **Inline style sheet**

External Style Sheet

In this case, style information is written in a separate file and is referenced from an HTML document. An external style sheet is useful when the same style is applied on different documents. If the document is an HTML document, the external style sheet is specified using the HTML **<link>** tag. Following is an example:

<link rel = ”stylesheet” type = ”text/css” href = ”mystyle.css”/>

This tag specifies that the style information to be used to display this document is stored in a file named **mystyle.css**. It is inserted in the <**head**> section of an HTML document.

According to W3C specification, external style sheets can be of three types: **Persistent, Alternate, and Preferred**. A persistent style sheet is one that is always applied. Document authors may also specify a set of alternate style sheets, one of which is selected by the user depending on their choice. Authors may also specify one of the alternate style sheets as the preferred style sheet which is applied when no style sheet is selected.

The attribute **rel** specifies the type of style sheet used. Its value is **“stylesheet”** for persistent and preferred style sheets and **“alternate stylesheet”** for alternate style sheets.

Example **CSS Document(styles.css)** file

```
hr {color:#6845ff;}
p {margin-left:20px;}
body {font-family:arial,Tahoma; }
```

HTML Document

```
<html>
<head>
<title>Example Web Page</title>
    <link rel = ”stylesheet” type = ”text/css” href = ”styles.css”/>
```

```
</head>
<body>
<p>      Text that will be formatted.    </p>
<hr>
<p>      This will also be formatted.    </p>
</body>
</html>
```

Embedded Style Sheet

An internal style sheet should be used when a single document has a unique style. In this method, style information is placed under the **<style>** tag in the **<head>** section of an HTML page.

```
<html>
<head>
<title>Example Web Page</title>
<style type="text/css">
        hr {color:#6845ff;}
        p {margin-left:20px;}
        body {font-family:arial,Tahoma; }
</style>
</head>
<body>
<p>  Text that will be formatted.    </p>
<hr>
<p>      This will also be formatted.    </p>
</body>
</html>
```

Inline Style Sheet

Style may be in lined using the STYLE attribute. The STYLE attribute may be applied to any BODY elements including BODY itself. The attribute takes as its value any number of CSS declarations, where each declaration is separated by a semicolon. An example follows:

```
<html>
<body>
<p style=”margin-left:20px; direction=ltr”>
Text that will be left aligned paragraph.
</p>
<hr style=”color:#6845ff;”>
```

```
<p style=”margin-right:20px; direction=rtl”>
This will be right aligned paragraph
</p>
</body>
</html>
```

This is the least flexible styling method. If we want to modify the style of an element, we have to work through the document and modify it.

CASCADING RULE

There are several ways to specify style rules such as external, embedded and inline methods. If more than one rule is specified for a tag, then the rules are conflicted. For conflicting rules, latter rules get preference over the earlier rules. That means the inline style sheet definition overrides the embedded style sheet, which in turn overrides the external style sheet for the HTML document.

TEXT FORMATTING CSS

The text displayed in the web page is styled with some of the text formatting properties. The heading uses the text-align, text-transform, and color properties. The paragraph is indented, aligned, and the space between characters is specified. Some of the text formatting properties are

- Text Color
- Text Alignment
- Text Vertical Alignment
- Text Decoration
- Text Transformation
- Text Indentation
- Letter Spacing
- Word Spacing
- Text Direction

Text Color

The color property is used to set the color of the text. The color can be specified by:

- name - a color name, like "red"
- RGB - an RGB value, like "rgb(255,0,0)"
- Hex - a hex value, like "#ff0000"

The default color for a page is defined in the body selector.

Example

```
Body{color:blue;}
h1{color:#00ff00;}
h2 {color:rgb(255,0,0);}
```

Text Alignment

The text-align property is used to set the horizontal alignment of a text. Text can be centered, or aligned to the left or right, or justified.

When text-align is set to "justify", each line is stretched so that every line has equal width, and the left and right margins are straight

Example

```
h1 {text-align: center;}
p {text-align: right;}
td {text-align: justify;}
```

Text Vertical Alignment

The vertical-alignment property is useful when working with inline elements such as images, tables. It allows us to control their vertical positioning within the containing element. It can take several values like.

Value	Description
length	Raises or lower an element by the specified length. Negative values are allowed
%	Raises or lower an element in a percent of the "line-height" property. Negative values are allowed
baseline	Align the baseline of the element with the baseline of the parent element. This is default
sub	Aligns the element as it was subscript
super	Aligns the element as it was superscript
top	The top of the element is aligned with the top of the tallest element on the line
text-top	The top of the element is aligned with the top of the parent element's font
middle	The element is placed in the middle of the parent element
bottom	The bottom of the element is aligned with the lowest element on the line
text-bottom	The bottom of the element is aligned with the bottom of the parent element's font

Examples

```
img { vertical-align: text-top; }
table  { vertical-align: middle; }
```

Text Decoration

The text-decoration property is used to set or remove decorations from text. The possible values for this property are none, overline, line-through, underline and blink.

Examples:

```
a {text-decoration: none;}
h1 {text-decoration: overline;}
h2 {text-decoration: line-through;}
h3 {text-decoration: underline;}
h4 {text-decoration: blink;}
```

Text Transformation

The text-transform property is used to specify either uppercase or lowercase letters in a text. It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word.

Examples:

```
p {text-transform: uppercase;}
h1 {text-transform: lowercase;}
td{text-transform: capitalize;}
```

Text Indentation

The text-indentation property is used to specify the indentation of the first line of a text with in an element. The value for this property is in terms of px, pc, in, mm

Example

```
p {text-indent:50px;}
```

Letter-spacing Property

The letter-spacing property increases or decreases the space between characters in a text within the element. The value for this property is in terms of px, pc, in, mm.

Example:

```
p {letter-spacing:2px}
h2 {letter-spacing:1px}
```

Word-spacing Property

The word-spacing property increases or decreases the white space between words. The value for this property is in terms of px, pc, in, mm.

Example

```
P {  word-spacing:30px; }
```

Text Direction

The direction property specifies the direction in which the text should flow. The possible values are ltr (left to right), rtl (right to left).

Example

```
p  { direction: rtl; }
```

Line – height Property

The line-height property is used to specify the gap between lines in the text. This should be specified in terms of px, pc, in, mm and percentages.

Example

```
p.small {line-height:90%}
p.big {line-height:200%}
```

Text Pseudo- Classes

There are two very useful pseudo-classes that can help us to work with text. These pseudo-classes allows us to render either the first letter or the first line of an element in a different way than the rest of that element. The name of the pseudo-class is separated from the element by a colon (:).

1. First - letter pseudo-class
2. First – line pseudo-class

first-letter pseudo-class

The first-letter pseudo-class allows us to specify a rule just for the first letter of an element. This is most commonly used on the first character of a new page.

Example

```
p:first-letter {  font-size:200%;  color:#8A2BE2; }
```

first-line pseudo-class

The first-line pseudo-class allows us to render the first line of any paragraph differently from the rest of the paragraph.

Example

```
p:first-line {  background-color : yellow; }
```

TEXT FORMATTING CSS PROPERTIES		
Property	**Description**	**Values**
color	Sets the color of a text	*color*
text-align	Aligns the text in an element	Left, right, center, justify
text-decoration	Adds decoration to text	None, underline, overline, line-through, blink
text-indent	Indents the first line of text in an element	*Length, %*
text-transform	Controls the letters in an element	None, capitalize, uppercase, lowercase
vertical-align	Sets the vertical alignment of an element	Baseline, sub, super, top, text-top middle, bottom text-bottom, *length, %*
letter-spacing	Increase or decrease the space between characters	Normal, *length*
word-spacing	Increase or decrease the space between words	Normal, *length*
direction	Sets the text direction	Ltr, rtl
line-height	Sets the distance between lines	Normal, *number length, %*

CONTROLLING FONTS USING CSS

Several properties related to font allow us to control the appearance of text in our document. CSS font properties define the font family, boldness, size, and the style of a text. Some of the font properties are

- ✓ font
- ✓ Font-family
- ✓ Font-size
- ✓ Font-weight
- ✓ Font-style
- ✓ Font-stretch
- ✓ Font-variant

Font Family

The font family of a text is set with the font-family property. The font-family property should hold several font names as a comma separated list. If the browser does not support the first font, it tries the next font. We provide the list of fonts starts with the font we want, and ends with a generic family, to let the browser pick a similar font in the generic family, if no other fonts are available.

In CSS, there are two types of font family names:

- **generic family** - A group of font families with a similar look such as "Serif" or "Sans Serif", "Monospace".
- **font family** - a specific font family like "Times New Roman" or "Arial".

Generic family	Font family
Serif	Times New Roman Georgia
Sans-serif	Arial Verdana
Monospace	Courier New Lucida Console

Example

```
p{font-family:"Times New Roman", Times, serif;}
```

Font Style

The font-style property is mostly used to specify italic text. This property has three values:

- normal - The text is shown normally
- italic - The text is shown in italics
- oblique - The text is "leaning" which is very similar to italic

Example

```
p.normal {font-style:normal;}
p.italic {font-style:italic;}
p.oblique {font-style:oblique;}
```

Font-size Property:

The font-size property enables us to specify a size for the font. We can specify a value for this property in several ways like

- Absolute size
- Relative size
- Length
- Percentage

xx-small, x-small, small, medium, large, x-large, xx-large are the values for the absolute sizes.

Smaller, larger are the values for relative sizes.

Length can be expressed in the form of **px, em, pt, in, cm, pc, mm**.

A **percentage** is calculated as a proportion of the element that contains the text like 20%, 10%

Example

```
H1{ font-size:xx-large;}
H6{font-size:smaller;}
P{font-size:12px;}
B{font-size:10%;}
```

Font-weight Property

The font-weight property sets how thick or thin characters in text should be displayed. The possible values for font-weight are nomal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900.

Example

```
p{font-weight:normal;}
h1 {font-weight:bold;}
h6{font-weight:100;}
```

Font-stretch Property

The font-stretch property allows us to make text wider or narrower that means the width of the actual letters in a font. It can take the values like normal, wider, narrower, and expanded.

Font-variant Property

The font-variant property specifies whether or not a text should be displayed in a small-caps font. In a small-caps font, all lowercase letters are converted to uppercase letters. However, the converted uppercase letters appears in a smaller font size than the original uppercase letters in the text.

There are two possible values for the font-variant property like normal and small-caps.

```
p.small{font-variant:small-caps;}
```

font property

The font shorthand property sets all the font properties in one declaration. The properties that can be set, are in order of "font-style font-variant font-weight font-size/line-height font-family". If one of the other values are missing, the default values will be inserted.0

Example

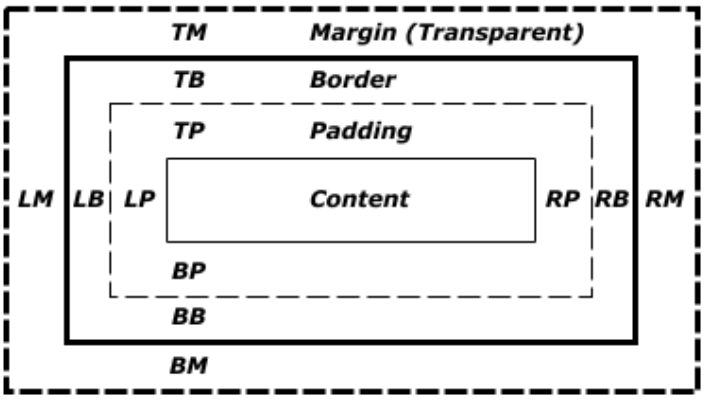
p.first { font:15px arial,sans-serif; }

p.second { font:italic bold 12px/30px Georgia, serif; }

FONT PROPERTIES		
Property	**Description**	**Values**
font	Sets all the font properties in one declaration	*font-style,font-variant, font-weight,font-size, font-family.*
font-family	Specifies the font family for text	*family-name, generic-family*
font-size	Specifies the font size of text	xx-small, x-small, small medium, large ,x-large, xx-large, smaller, larger *length, %*
font-style	Specifies the font style for text	Normal, italic, oblique
font-variant	Specifies whether or not a text should be displayed in a small-caps font	Normal, small-caps
font-weight	Specifies the weight of a font	Normal, bold, bolder, lighter,100,200,300,400 500, 600, 700, 800,900
font-stretch	make text wider or narrower	normal, wider, narrower, expanded.

THE CSS BOX MODEL

All HTML elements can be considered as boxes. The CSS box model is essentially a box that wraps around HTML elements, and it consists of margins, borders, padding, and the actual content. The box model allows us to place a border around elements and arrange the elements inside the border.



The parts of Box Model are:

- **Margin –** The Margin is the distance between the edge of a box and the box next to it.
- **Border** - A border that goes around the padding and content. The border is affected by the background color of the box
- **Padding –** This padding is the space between the content of the box and its border.
- **Content** - The content of the box, where text and images appear

Margin Property for Box Model

The CSS margin properties define the space around elements. The margin does not have a background color, and is completely transparent. The top, right, bottom, and left margin can be changed independently using separate properties. A shorthand margin property can also be used, to change all margins at once. The value of this property is either a length in terms of px, pm, in or percentage.

In CSS, it is possible to specify different margins for different sides like

```
margin-top:100px;
margin-bottom:100px;
margin-right:50px;
margin-left:50px;
```

It is possible to specify all the margin properties in one property. This is called a shorthand property. The shorthand property for all the margin properties is "margin".

The margin property can have from one to four values.

- **margin:25px 50px 75px 100px;**
  - top margin is 25px
  - right margin is 50px
  - bottom margin is 75px
  - left margin is 100px
- **margin:25px 50px 75px;**
  - top margin is 25px
  - right and left margins are 50px
  - bottom margin is 75px
  -
- **margin:25px 50px;**
  - top and bottom margins are 25px
  - right and left margins are 50px
- **margin:25px;**
  - all four margins are 25px

MARGIN CSS PROPERTIES		
Property	**Description**	**Values**
margin	A shorthand property for setting the margin properties in one declaration	*margin-top, margin-right, margin-bottom margin-left*
margin-bottom	Sets the bottom margin of an element	Auto, *length,%*
margin-left	Sets the left margin of an element	Auto, *length, %*
margin-right	Sets the right margin of an element	Auto, *length, %*
margin-top	Sets the top margin of an element	Auto, *length, %*

Border Property for Box Model

The CSS border properties allow us to specify the style and color of an element's border. There are three properties of a border.

1. border-style
2. border-color
3. border-width

border-style Property

The border –style property allows us to specify the style of the border.

P{border-style:solid;}

The values of the border are

none: Defines no border

dotted: Defines a dotted border

dashed: Defines a dashed border

solid: Defines a solid border

double: Defines two borders. The width of the two borders are the same as the border-width value

groove: Defines a 3D grooved border. The effect depends on the border-color value

ridge: Defines a 3D ridged border. The effect depends on the border-color value

inset: Defines a 3D inset border. The effect depends on the border-color value

outset: Defines a 3D outset border. The effect depends on the border-color value

In CSS it is possible to specify different borders for different sides.

For example

```
p
{
border-top-style:dotted;
border-right-style:solid;
border-bottom-style:dotted;
border-left-style:solid;
}
```

border-color Property

The border-color property is used to set the color of the border. The color can be set by two methods.

- name - specify a color name, like "red"
- Hex - specify a hex value, like "#ff0000"

Example

P{border-style:solid; border-color:red; }

In CSS it is possible to specify different borders for different sides

```
P                                                                    {
border-top-style:dotted;
border-right-style:solid;
border-bottom-style:dotted;
border-left-style:solid;
}
```

border-width property

The border-width property is used to set the width of the border. The width is set in pixels, or by using one of the three pre-defined values: thin, medium, or thick.

p{ border-style:solid; border-width:5px;}

Border Property

It is also possible to specify all the individual border properties in one property. The border property is a shorthand for the following individual border properties:

- border-width
- border-style
- border-color

Example

P{border:5px solid red;}

Padding Property for Box Model

The CSS padding properties define the space between the element border and the element content. The top, right, bottom, and left padding can be changed independently using separate properties. A shorthand padding property can also be used, to change all paddings' at once.

Padding - Individual sides

In CSS, it is possible to specify different padding for different sides

padding-top:25px;

padding-bottom:25px;

padding-right:50px;

padding-left:50px;

Padding - Shorthand property

It is possible to specify all the padding properties in one property. The shorthand property for all the padding properties is "padding":

P{padding:25px 50px 75px 100px;}

Property	Description	Values
padding	A shorthand property for setting all the padding properties in one declaration	*padding-top, padding-right, padding-bottom padding-left*
padding-bottom	Sets the bottom padding of an element	*Length, %*
padding-left	Sets the left padding of an element	*Length, %*
padding-right	Sets the right padding of an element	*Length, %*
padding-top	Sets the top padding of an element	*Length, %*

Dimensions for the Box Model

We can set the dimensions for the box in terms of width and height.

Property	Purpose
Height	Sets the height of the box
Width	Sets the width of the box
Max-height	Sets the maximum height of a box
Min-height	Sets the minimum height of a box
Max-width	Sets the maximum width of a box
Min-width	Sets the minimum width of a box

BACKGROUND PROPERTIES

CSS treats each element as in its own box. We can control the background of these boxes using the background properties.

Property	Purpose
background-color	Specifies a background color
background-image	Specifies an image to use as the background
background-repeat	Indicates whether the background image should be repeated
background-attachment	Indicates a background image should be fixed in one position on the page, and whether it should stay in that position when the user scrolls down the page
background-position	Indicates where an image should be positioned
background	A shorthand form that allows you to specify all of these properties

background Color: background-color: <color> | transparent

The background-color property sets the background color of an element.

For example:

body { background-color: white }

h1 { background-color: #000080 }

The value transparent indicates that whatever is behind the element can be seen.

background-image : background-image: <url> | none

The background-image property sets the background image of an element. For example:

body { background-image: url("/images/monkey.gif") }

p { background-image: url("http://www.cclrc.com/pretty.png") }

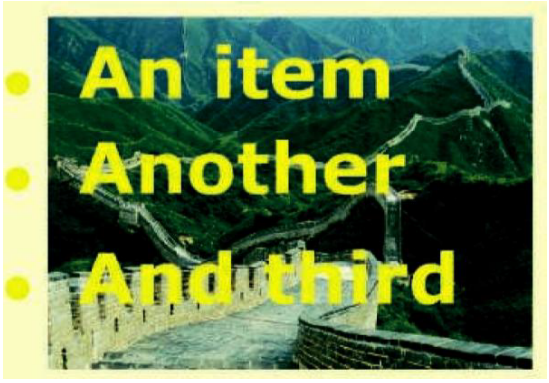
h1 { background-image: none }

We have the HTML like

```
<ul style="background-
image:url('wall.png')"> <li>An item</li>

<li>Another
</li>
<li>And
third</li>
</ul>
```

It shows the output like



Background-repeat : background-repeat: repeat | repeat-x | repeat-y | no-repeat

If we do not want our image to repeat all over the background of the box, we should use the background - repeat property, which has four helpful values

Value	Purpose
repeat	This causes the image to repeat to cover the whole page (it is the default therefore rarely used).
repeat-x	The image will be repeated horizontally across the page (not down the whole page vertically).
repeat-y	The image will be repeated vertically down the page (not across horizontally).
no-repeat	The image is displayed only once.

For example

```
body { background-image:
url(/images/monkey.gif) } body { background-
repeat: repeat-x }
```

In the above example, the monkey will only be tiled horizontally

Background-attachment : background-attachment: scroll | fixed

When we specify a background image we can use the background - attachment property to specify whether the image is fixed in its position, or whether it moves as the user scrolls up and down the page.

For example

```
body { background-image:
url("monkey.jpg") } body { background-
attachment: fixed }
```

background-position : background-position: [percentage> | length]{1,2} | [[top | center | bottom] || [left | center | right]]

We may want to alter the position of this image, and we can do this using the background – position property, which takes the values like

Value	Meaning
x% y%	Percentages along the *x* (horizontal) and *y* (vertical) axis
x y	Absolute lengths along the *x* (horizontal) and *y* (vertical) axis in pixels
left	Shown to the left of the page or containing element
center	Shown to the center of the page or containing element
right	Shown to the right of the page or containing element
top	Shown at the top of the page or containing element
center	Shown at the center of the page or containing element
bottom	Shown at the bottom of the page or containing element

Background

The background property allows us to specify several of the background properties at once. The values can be given in any order, and if we do not supply one of the values, the default value will be used.

background: [<background-color> || <background-image> || <background-repeat> || <background-attachment> || <background-position>]

For Example

```
body { background: white url(http://www.clrc/xyz.gif) }
p { background: url(../backgrounds/lion.png) #F0F000 fixed }
h1 { background: #0F0 url(grass.jpg) no-repeat bottom left }
ul { background: white url('monkey.jpg') repeat scroll 0% 50%}
```

SELECTORS

The basic building blocks of style rules are selectors. Selectors determine elements on which rules are to be applied. The elements selected by selectors are called **subjects** of selectors. A selector may simply be the name of an element called **simple selector** or may consist of a rich contextual pattern called **complex selector**. A complex selector consists of one or more simple selectors separated by combinatory such as white space, ">", and "+".

Grouping

Selectors having common declarations are grouped into a comma-separated (,) list. For example the below code

h1 { color ; red; }
h2 { color ; red; }
h3 { color ; red; }

This is equivalent to: **h1, h2, h3 { color: red; }**

Type Selectors

A type selector is a simple selector, which is the name of a document element and it matches every single element of the document. For example, the selector **p** selects every **<p>** element in the document. Similarly, the selector **b** selects every **<b>** element.

Universal Selectors

CSS has a special selector *, which matches with every single element in the document. For example,

*** { color: red; }**

It makes all the text in the document red.

The universal selector is useful when element names are not known in advance during the development of a style sheet.

Descendant Selectors

Descendant selectors, also called **contextual selectors**, allow us to determine the elements depending upon their hierarchical relationship. A descendant selector selects only those elements that are descendants of a specified element. A descendant selector is made up of two or more selectors separated by white space. A descendant selector of the form "A B" matches when an element B is the descendant of some ancestor element A.

For Example

h1 { color: red }
em { color: red }

Child Selector

Child selectors select elements that are immediate children of a specified element. The combinatory used for child selector is **“>”.**

For example

body > P { line-height: 1.3 }

Attribute Selectors

Attribute selectors provide a ways of selecting elements depending on the presence of an attribute or the presence of certain attribute values. Attribute selectors are not supported by IE but are supported by Firefox and opera.

Simple Attribute Selector

It selects elements having a specified attribute. The general syntax of simple attribute selectors is as follows:

element[attribute_name]

or

[attribute_name]

For example

a [href]

selects all <a> elements having the attribute href. If the element name is omitted, all elements are assumes.

Attribute Value Selector

It allows us to select elements having an attribute with a value equal to given value in CSS. It is used as:

element[attribute_name="value"]

or

[attribute_name="value"]

The following example

P[class="bold"]

matches with the following:

…

One of many attribute value selector

It allows us to select elements having an attribute with a value equal to any one of the values separated by white spaces. It is used as:

element[attribute_name ~ ="value"]

or

[attribute_name~="value"]

The following example

P[class~="bold"]

matches with the following:

…

…

Hyphen attribute value selector

This selector selects an element having attribute value exactly equal to the specified value or beginning with the specified value immediately followed by **"-"**. The basic purpose of this selector is to match language sub-code. Here is an example:

p[lang |="en"]

that matches … and … .

Starts-with attribute value selector

This selector selects elements having an attribute value that starts with the value specified. The general syntax is

element[attribute_name^="attribute_value"]

or

[attribute_name^="attribute_value"]

For example, the selector p[type^= "copy"] matches copyright… as well as copyleft…

Ends-with attribute value selector

This attribute selects elements having attribute value that ends with the value specified. The general syntax is

element[attribute_name$="attribute_value"]

or

[attribute_name$="attribute_value"]

For example, a[href$=".com"] selects those anchor tags that point to .com website. So, it matches

yahoo

rediff mail

but not

jadavpur university

Substring match attribute value selector

This selector selects those elements having an attribute value containing at least one occurrence of the value specified. The general syntax is

element[attribute_name*="attribute_value"]

or [attribute_name*="attribute_value"]

For example, a[href*="image"] selects those anchor tags that have image in the href attribute.

Pseudo Classes and Elements

A pseudo-class is way of selecting certain parts of a HTML document, based in principle not on the HTML document tree itself, but on other phantom conditions like language encoding or the dynamic state of an element. Some of the pseudo classes and elements are

- :link
- :visited
- :active
- :hover
- :first-child
- :last-child
- :only-child
- :after
- :before

:first-child Pseudo-class

The :first-child pseudo-class matches a specified element that is the first child of another element.

In the following example, the selector matches any <p> element that is the first child of any element:

Example

```
<html>
<head>
<style type="text/css">
p:first-child { color:blue; }
</style>
</head>
<body>
<p>I am a strong man.</p>
<p>I am a strong man.</p>
</body>
</html>
```

:last-child Pseudo-class

The :last-child selector matches every element that is the last child of its parent.

Example

```
p:last-child { background:#ff0000; }
```

:only-child Pseudo-class

The :only-child selector matches every element that is the only child of its parent

Example

p:only-child { background:#ff0000; }

:before Pseudo-element

The ":before" pseudo-element can be used to insert some content before the content of an element.

Example

h1:before { content:url(smiley.gif); }

:after Pseudo-element

The ":after" pseudo-element can be used to insert some content after the content of an element.

Example

h1:after { content:url(smiley.gif); }

CSS FOR LINKS

Links can be styled with any CSS property such as color, font-family, background, etc. Special CSS for links are in such a way that they can be styled differently depending on what state they are in.

CSS defined some pseudo classes for links that can give greater control over presentation of links. They are

- **a: link - a normal, unvisited link**
- **a: visited - a link the user has visited**
- **a: hover - a link when the user moves over it**
- **a: active - a link the moment it is clicked**

- The **:link** pseudo class applies to those hyper links that have not yet been visited.
- The **:visited** pseudo class applies to hyperlinks that have already been visited at least once.
- The **:hover** pseudo class selects the hyperlinks that are being designated by the user with a pointing device.
- The **:active** pseudo class applies to an element that is currently being activated by the user.

For example :

```
body {background-color:#ffffff;}
a { font-family: arial, verdana, sans-serif; font-size:12px; font-weight:bold;}
a:link { color:#0000ff; text-decoration:none;}
a:visited { color:#333399; text-decoration:none;}
a:link:hover { background-color:#e9e9e9; text-decoration:underline;}
a:active { color:#0033ff; text-decoration:underline;}
```

CSS FOR LISTS

CSS allows us to customize the lists that can be made with HTML. The CSS list properties facilitate us 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

Property	Purpose
list-style-type	Allows you to control the shape or appearance of the marker (the marker is another name for the bullet point or number).
list-style-position	When a list item takes up more than one line, this property specifies where the marker should appear in relation to the text.
list-style-image	Specifies an image for the marker rather than a bullet point or number.
list-style	Serves as shorthand for the preceding properties.
marker-offset	Specifies the distance between a marker and the text in the list.

The list - style - type Property

The list - style - type property allows you to control the shape or style of bullet point in the case of unordered lists and the style of numbering characters in ordered lists.

The values to this property for unordered list are

Value	Marker
none	None
disc (default)	A filled-in circle
circle	An empty circle
square	A filled-in square

The values to this property for ordered list are

Value	Meaning	Example
decimal	Number	1, 2, 3, 4, 5
decimal-leading-zero	0 before the number	01, 02, 03, 04, 05
lower-alpha	Lowercase alphanumeric characters	a, b, c, d, e
upper-alpha	Uppercase alphanumeric characters	A, B, C, D, E
lower-roman	Lowercase Roman numerals	i, ii, iii, iv, v
upper-roman	Uppercase Roman numerals	I, II, III, IV, V

The list - style - type property can either be used on the < ul > and < ol > elements in which case it applies to the entire list or on the individual < li > elements.

For example

ul {list-style-type: circle;}

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

li {list-style-type:disc;}

The list - style - position Property

The list - style - position property indicates whether the marker should appear inside or outside of the box containing the main points. There are two values for this property. They are

Value	Purpose
inside	The marker is inside the block of text (which is indented).
outside	The marker sits to the left of the block of text (this is the default value if this is not specified).

Example

ul {list-style-position:outside;}

ol {list-style-position:inside;}

The list - style - image Property

The list - style - image property allows us to specify an image so that we can use our own bullet style. For this property we must supply the URL of marker image as the value.

For example

li {list-style-image: url("images/bulletpoint.gif");}

The list - style Property

The list - style property is a way of expressing more than one of these properties at once. They can appear in any order. For example:

ul {list-style: inside circle;}

marker-offset property

The *marker-offset* property defines the distance between the marker and text associated to that marker. The value for this property is the length to be defined in terms of px, em, pc, mm. For example

Li { marker-offset: 10em;}

CSS FOR TABLES

Properties that are commonly used with the < table > , < td > , and < th > elements are

- border to set the properties of the border of a table.
- padding to set the amount of space between the border of a table cell and its content — this property is very important to make tables easier to read.
- Properties to change text and fonts.
- text - align to align writing to the left, right, or center of a cell.
- vertical - align to align writing to the top, middle, or bottom of a cell.
- width to set the width of a table or cell.
- height to set the height of a cell (often used on a row as well).
- background - color to change the background color of a table or cell.
- background - image to add an image to the background of a table or cell.

Table specific properties

CSS defined some properties that can only be used with tables. They are

Property	Purpose
border-collapse	Where the borders of two table cells touch, this property indicates whether *both* borders should be visible, or whether the browser should pick just one of the borders to show.
border-spacing	Specifies the width of the space that should appear between table cells.
caption-side	Specifies which side of a table the caption should appear on.
empty-cells	Specifies whether the border should be shown if a cell is empty.
table-layout	If the space you have allocated for a table is not enough to fit the contents, browsers will often increase the size of the table to fit the content in — this property can force a table to use the dimensions you specify.

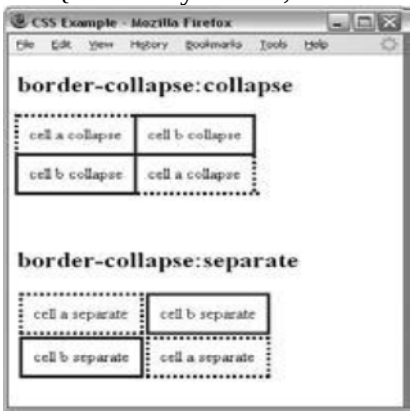
The border - collapse Property

When two table cells meet, we can tell the browser to show just one of the borders. We can do this using the border - collapse property, which can take two values.

Value	Purpose
collapse	Horizontal borders will be collapsed and vertical borders will abut one another.
separate	Separate rules are observed. This value opens up additional properties to give you further control.

For example

```
table.one {border-collapse:collapse;}
table.two {border-collapse:separate;}
td.a {border-style:dotted; border-width:3px; border-color:#000000; padding:10px;}
td.b {border-style:solid; border-width:3px; border-color:#333333; padding:10px;}
```



The border - spacing Property

The border - spacing property specifies the distance that separates adjacent cell's borders. If we provide one value, it will apply to both vertical and horizontal borders.

table.one {border-spacing:15px;}

We can also can specify two values, in which case the first refers to the horizontal spacing and the second to the vertical spacing:

table.two {border-spacing:2px 4px;}

The empty - cells Property

The empty - cells property indicates whether a cell without any content should have a border displayed. It can take one of two values.

Value	Purpose
show	Borders will be shown even if the cell is empty (this is the default value).
hide	Borders will be hidden if cell is empty.

CSS Code

table { width:350px; border-collapse:separate; empty-cells:hide;}

td {padding:5px; border-style:solid; border-width:1px; border-color:#999999;}

HTML Code

< table > < tr > < th > < /th >

< th > Title one < /th > < th > Title two < /th > < /tr >

< tr > < th > Row Title < /th > < td > value < /td > < td > value < /td >

< /tr >

< tr > < th > Row Title < /th > < td > value < /td > < td > < /td > < /tr >

< /table >

The caption - side Property

The caption - side property is for use with the < caption > element to indicate on which side of the table the caption should go. The values for the caption-side property are

Value	Purpose
top	The caption will appear above the table (the default).
right	The caption will appear to the right of the table.
bottom	The caption will appear below the table.
left	The caption will appear on the left side of the table.

For example

caption {caption-side:bottom}

The table - layout Property

The table-layout property sets the table layout algorithm to be used for a table.

Value	Description
auto	Automatic table layout algorithm (this is default): • The column width is set by the widest unbreakable content in the cells • Can be slow, since it needs to read through all the content in the table, before determining the final layout
fixed	Fixed table layout algorithm: • The horizontal layout only depends on the table's width and the width of the columns, not the contents of the cells • Allows a browser to lay out the table faster than the automatic table layout • The browser can begin to display the table once the first row has been received