**CSS DOC**

**Selectors** in CSS: id ‘#‘ and class ‘.‘

Three ways to insert style: External, Internal and Inline

**Multiple styles for same selectors:**

If any selector has style mentioned in external as well as in internal then common properties will get replaced by internal style and remaining will get inherited from external.

Cascading Order :

1. Browser default
2. External style sheet
3. Internal style sheet (in the head section)
4. Inline style (inside an HTML element)

*Inline style has higher priority.*

***Note****: If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will* ***override*** *the internal style sheet!*

**CSS Properties:**

1. **Background**

By default, the **background-image** property repeats an image both horizontally and vertically.

To repeat horizontally: *background-repeat:****repeat-x****;*

To repeat Vertically: *background-repeat:****repeat-y****;*

To avoid repetition of image: *background-repeat:****no-repeat****;*

To apply position to image: *background-position:right top;*

**Short-hand property for background**:

*background:#ffffff url('img\_tree.png') no-repeat right top;*

1. **Text**

The **text-align** property is used to set the horizontal alignment of a text.

Values: center, left, right and **justify.**

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 (like in magazines and newspapers).

**text-decoration**: [values: none, line-through, underline, overline]

The **text-transform** property is used to specify uppercase and lowercase letters of text

Values: uppercase, lowercase and capitalize.

The capitalize makes capitalize the first letter of each word.

The **text-indent** property is used to specify the indentation [space for first line like newspaper] of the first line of a text.

The **vertical-align** sets the vertical alignment of an element

e.g.

Hello <img style="vertical-align:text-top" src="w3schools\_logo.gif" alt="W3Schools" width="270" height="50" /> Text End

Hello W3Schools Text End.

**Some more properties of Text**:

color:

direction: rtl and ltr(default). To use rtl we will have to use -------

letter-spacing: Specify space between characters in text.

line-height: Sets the line height e.g *line-height:70%*

text-shadow: e.g. *text-shadow: 2px 2px #ff0000;*

white-space: e.g. *white-space:nowrap;*

word-spacing: Increases or decreases the space between words in a text

1. **Font**

**font-family**: "Times New Roman", Times, serif;

***Note****: If the name of a font family is more than one word, it must be in quotation marks, like: "Times New Roman".*

More than one font family is specified in a comma-separated list:

**font-style**: [ values: normal, italic, oblique ]

**font-size**:

With pixels- font-size:40px;

With Em- font-size:2.5em

With Percentage and Em- body{font-size:100%} h1{font-size:2.5em}

Here 2.5 em means 40px because body font-size is 100%.

If we decrease the percentage of font-size of body then h1 size will become small.

***Note****: If you do not specify a font size, the default size for normal text, like paragraphs, is 16px (****16px=1em****).*

**font-weight**: [values: normal, lighter, bold, 900]

**font-variant:** [Values: small-caps]

e.g. My name is hege Refsnes ---------> My name is hege Refsnes.

**Short-hand property:**

font: italic bold 12px/30px Georgia,serif;

Where, 12px – font-size and

30px – line-height

1. **Link**

Link states:

*e.g. a:link {color:#FF0000;} /\* unvisited link \*/*

*a:visited {color:#00FF00;} /\* visited link \*/*

*a:hover {color:#FF00FF;} /\* mouse over link \*/*

*a:active {color:#0000FF;} /\* selected link \*/*

Above sequence is required for hover and active

1. **List** [unordered lists and ordered list]

**list-style-type**: [Lot of values like circle, upper-roman,lower-alpha etc.]

Use this property according to list type

**list-style-image**: url('sqpurple.gif');

**list-style-position:**[values: inside,outside]

1. **Tables**

**border**: e.g. table, th, td {border: 1px solid black} o/p- table will show box for each cell.

**border-collapse**: To avoid above cell border we use *border-collapse:collapse* property.

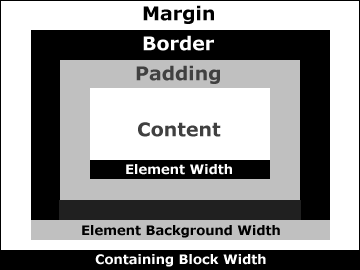
**text-align**: [values: left, right and center]e.g.  *td {text-align : right;}*

**vertical-align**: [values: top, bottom and middle]e.g.  *td {vertical-align : top;}*

**padding:**

**color:**

1. **CSS Box Modal:**



***Important****: When you set the width and height properties of an element with CSS, you just set the width and height of the content area. To calculate the full size of an element, you must also add the* ***padding, borders and margins.***

1. **Border:**

border-style:[values: solid, dotted, dashed etc]

border-width: [values: in px e..g 1px and medium]

**Shorthand**: *border: 1px solid red;*

***Note****: The "border-width" property does not work if it is used alone. Use the "border-style" property to set the borders first.*

We can apply style to each side of the element as below:

*e.g. border-top-style:dotted;  
 border-right-style:solid;  
 border-bottom-style:dotted;  
 border-left-style:solid;*

***Shorthand*** *for above example is border-style:dotted solid double dashed*

1. **Outline:**

An outline is a line that is drawn around elements (outside the borders) to make the element "stand out".

e.g. outline: green dotted 2px;

*Note: The outline is not a part of an element's dimensions; the element's total width and height is not affected by the width of the outline.*

1. **Margin**:

Define the space around elements.

The margin does not have a background color, and is completely transparent.

*Note: It is also possible to use negative values, to overlap content.*

1. **Padding**:

Define the space between the element border and the element content.

The padding is affected by the background color of the element.

1. **Display and Visibility:**

display**:none** hides the element.

visibility**:hidden** hides an element, but it will still take up the same space as before. The element will be hidden, but still affect the layout.

**Display block**:

A block element is an element that takes up the full width available, and has a line break before and after it. e.g. <h1>, <div>, <p> etc.

**Display inline**:

An inline element only takes up as much width as necessary, and does not force line breaks. e.g <span>, <a>

Inline elements have left, right, padding and margin but no top and bottom. Cannot have width and height.

**Display inline-block:**

allow other elements to sit to their left and right; respect top & bottom margins and padding; respect height and width.

**Changing How an Element is Displayed**

*e.g. li{display:inline} -*  li is block level element but we can change it to inline

*span{display: block} -* span is inline level element but we can change it to block

1. **Positioning:**

Elements can be positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the positioning method.

There are four different positioning methods.

1. **Static** [default]: Static positioned are not affected by top, left, right and bottom.
2. **Fixed**: element become fix even window has scrolled.

***Note****: IE7 and IE8 support the fixed value only if a !DOCTYPE is specified.*

1. **Relative**: A relative positioned element is positioned relative to its normal position.

Relatively positioned elements are often used as container blocks for absolutely positioned elements.

1. **Absolute**:

An absolute position element is positioned relative to the first parent element that has a position other than static. If no such element is found, the containing block is <html>

Absolutely positioned elements are removed from the normal flow. The document and other elements behave like the absolutely positioned element does not exist.

The **z-index** is used to overlapping between elements.

**Note:** If two positioned elements overlap without a z-index specified, the element positioned last in the HTML code will be shown on top.

**Set shape of an element:**

***clip****="auto | rect(top right bottom left) | inherit" .* But position should be absolute.

e.g. img { position:absolute; clip:rect(0px,60px,200px,0px); }

**overflow** property specifies what to do if the content of an element exceeds the size of the element's box.

1. visible: Default. It renders outside the element box.
2. hidden: overflow is clipped and remaining content will be invisible.
3. scroll: overflow is clipped and add the scroll-bar
4. auto: If overflow is clipped, a scroll-bar should be added to see rest of content. (It shows vertical scroll)
5. inherit: inherited from parent
6. **Float**: [values: none, left and right]

Elements are floated horizontally means [floated left or right, not up or down]

The elements after the floating element will flow around it.

The elements before the floating element will not be affected.

If you place several floating elements after each other, they will float next to each other if there is room.

Elements after the floating element will flow around it. To avoid this, use the clear property ***clear:both*** [values: left, right and both]

*Note: float property will not work with absolute position.*

1. **Align**: [Will not work with absolute position]

Horizontally center-> *margin-left and margin-right to auto*.

**Cross-browser Compatibility Issues**

When aligning elements it is always a good idea to predefine margin and padding for the <body> element. This is to avoid visual differences in different browsers.

There is a problem with IE8 and earlier when using the float property. If the !DOCTYPE declaration is missing, IE8 and earlier versions will add a 17px margin on the right side. This seems to be space reserved for a scrollbar. Always set the !DOCTYPE declaration when using the float property.

1. **CSS pseudo-classes And CSS pseudo elements**

These are used to add special effects to some selectors.

*selector:pseudo-class {property:value;}*

e.g. *a:visited{color:red}*

*selector.class:pseudo-class {property:value;}* // pseudo-classes with css class

e.g. *a .myclass : visited {color:pink};*

**first-child pseudo class:**

p: first-child // Match the first <p> element

p > i : first-child // Match first i (i.e. italic) in all p elements

p: first-child i // Match all <i> elements in first child <p> elements

p:first-child i:first-child // Match first <i> in first <p> element

p:first-letter // Work with block-level elements

p:first-line // Work with block-level elements

p:before { content:"Read this -"; } // Add content before paragraph

p:after // Add content after paragraph

1. **Image transparency:**

img  
{  
opacity:0.4;  
filter:alpha(opacity=40); /\* For IE8 and earlier \*/  
}  
img:hover  
{  
opacity:1.0;  
filter:alpha(opacity=100); /\* For IE8 and earlier \*/  
}

**Image Sprites:**

An image sprite is a collection of images put into a single image

e.g. width:46px;  
height:44px;  
background:url(img\_navsprites.gif) 0 0;

1. **Media Types : [DO IT LATER]**
2. **Attribute Selectors:**

Style HTML elements which having specific attribute.

E.g. *[title] {color:red};*

Specific attribute with values ‘*W3Schools’* e.g. *[title=W3Schools] {color:red}*

The title attribute which value having ‘hello’ word. e.g. *[title~=hello] { color:blue;}*

Lang attribute which value start with ‘en’ *[lang|=en] {color:blue;}*

**Styling Form without using class and ID:**

*input [type=”text”] { background-color:red };*

*input[type="text"]:hover { background-color:blue } // On hover of text input*

1. Conditional CSS

Conditional Operators:

lt - Less than

lte - Less than or equal to

eq - Equal to

gte - Greater than or equal to

gt - Greater then

1. Style for different-browsers (Hacks)

*p {*

*color: pink;*

*color: red\0/; /\* IE 8 \*/*

*\*color: blue; /\* IE 7 \*/*

*\_color: yellow; /\* IE 6 \*/*

*}*