

Cascading Style Sheets Notes (Source: from w3schools.org)

What is CSS?

- **CSS** stands for **Cascading Style Sheets**
- Styles define **how to display** HTML elements
- Styles were added to HTML 4.0 **to solve a problem**
- **External Style Sheets** can save a lot of work
- External Style Sheets are stored in **CSS files**

Styles Solved a Big Problem

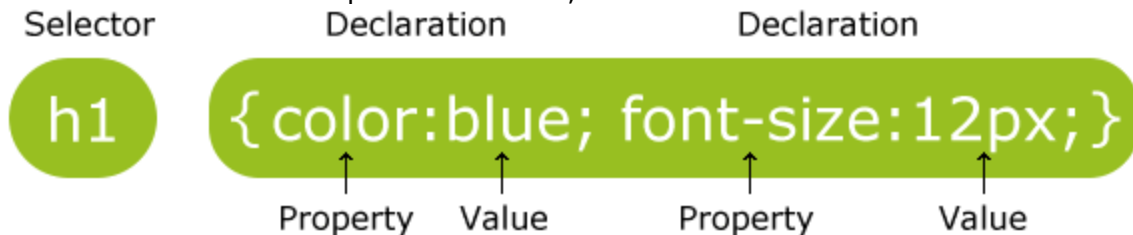
- HTML was never intended to contain tags for formatting a document.
- HTML was intended to define the content of a document, like:
 - `<h1>This is a heading</h1>`
 - `<p>This is a paragraph.</p>`
- When tags like ``, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large web sites, where fonts and color information were added to every single page, became a long and expensive process.
- To solve this problem, the World Wide Web Consortium (W3C) created CSS.
- In HTML 4.0, all formatting could be removed from the HTML document, and stored in a separate CSS file.
- All browsers support CSS today.

CSS Saves a Lot of Work!

- CSS defines HOW HTML elements are to be displayed.
- Styles are normally saved in external .css files. External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single file!

CSS Syntax

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



- The selector is normally the HTML element you want to style.
- Each declaration consists of a property and a value.
- The property is the style attribute you want to change. Each property has a value.

CSS Selectors

- Three types of CSS Selectors
 - Element
 - Class
 - ID

CSS Element Selector Example

- A CSS declaration always ends with a semicolon, and declaration groups are surrounded by curly brackets:
- `p {color:red;text-align:center;}`
- To make the CSS more readable, you can put one declaration on each line, like this:
- ```
p
{
color:red;
text-align:center;
}
```

## CSS Comments

- Comments are used to explain your code, and may help you when you edit the source code at a later date. Comments are ignored by browsers.
- A CSS comment begins with "/\*", and ends with "\*/", like this:
- ```
/*This is a comment*/
p
{
text-align:center;
/*This is another comment*/
color:black;
font-family:arial;
}
```

The id and class Selectors

- In addition to setting a style for a HTML element, CSS allows you to specify your own selectors called "id" and "class".

The id Selector

- The id selector is used to specify a style for a single, unique element.
- The id selector uses the id attribute of the HTML element, and is defined with a "#".
- **Example**

Imagine within the body element of our html page, we have the following paragraph element

```
<p id="welcome">Welcome to the wonderful world of HTML</p>
```

We can then create a CSS rule with the id selector:

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

The class Selector

- The class selector is used to specify a style for a group of elements. Unlike the id selector, the class selector is most often used on several elements.
- This allows you to set a particular style for many HTML elements with the same class.
- The class selector uses the HTML class attribute, and is defined with a "."
- In the example below, all HTML elements with class="center" will be center-aligned:
- **Example**

Imagine within the body element of our html page, we have the following header element

```
<h2 class="center">Summary</h2>
```

We can then create a CSS rule with the class selector:

```
.center {text-align:center;}
```

- You can also specify that only specific HTML elements should be affected by a class.
- In the example below, all p elements with class="center" will be center-aligned:
- **Example**
- `p.center {text-align:center;}`

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style


External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
```

An external style sheet can be written in any text editor. The file should not contain any html tags. Your style sheet should be saved with a .css extension. An example of a style sheet file is shown below:

```
hr {color:sienna;}
p {margin-left:20px;}
body {background-image:url("images/back40.gif");}
```

 Do not leave spaces between the property value and the units! "margin-left:20 px" (instead of "margin-left:20px") will work in IE, but not in Firefox or Opera.

Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, by using the <style> tag, like this:

```
<head>
<style type="text/css">
hr {color:sienna;}
p {margin-left:20px;}
body {background-image:url("images/back40.gif");}
</style>
</head>
```

Inline Styles

An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly!

To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a paragraph:

```
<p style="color:sienna;margin-left:20px">This is a paragraph.</p>
```

Multiple Styles Will Cascade into One

Styles can be specified:

- inside an HTML element
- inside the head section of an HTML page
- in an external CSS file

Tip: Even multiple external style sheets can be referenced inside a single HTML document.

Cascading order

What style will be used when there is more than one style specified for an HTML element?

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:

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

So, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).



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 Background Properties

CSS background properties are used to define the background effects of an element.

CSS properties used for background effects:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position

Background Color

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

The background color of a page is defined in the body selector:

Example

```
body {background-color:#b0c4de;}
```

With CSS, a color is most often specified by:

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

Look at [CSS Color Values](#) for a complete list of possible color values.

In the example below, the h1, p, and div elements have different background colors:

Example

```
h1 {background-color:#6495ed;}
p {background-color:#e0ffff;}
div {background-color:#b0c4de;}
```

Background Image

The background-image property specifies an image to use as the background of an element.

By default, the image is repeated so it covers the entire element.

The background image for a page can be set like this:

Example

```
body {background-image:url('paper.gif');}
```

Background Image - Repeat Horizontally or Vertically

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

Some images should be repeated only horizontally or vertically, or they will look strange.

To repeat an image only horizontally or vertically, use the background-repeat property.

```
body
{
background-image:url('gradient2.png');
background-repeat:repeat-x;
}
```

Background Image - Set position and no-repeat



When using a background image, use an image that does not disturb the text.

Showing the image only once is specified by the background-repeat property:

Example

```
body
{
background-image:url('img_tree.png');
background-repeat:no-repeat;
}
```

Background - Shorthand property

As you can see from the examples above, there are many properties to consider when dealing with backgrounds.

To shorten the code, it is also possible to specify all the properties in one single property. This is called a shorthand property.

The shorthand property for background is simply "background":

Example

```
body {background:#ffffff url('img_tree.png') no-repeat right top;}
```

When using the shorthand property the order of the property values are:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position

It does not matter if one of the property values is missing, as long as the ones that are present are in this order.

CSS Border

CSS Border Properties

The CSS border properties allow you to specify the style and color of an element's border.

Border Style

The border-style property specifies what kind of border to display.



None of the border properties will have ANY effect unless the **border-style** property is set!

border-style values:

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

Border Width

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.

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

Example

```
p.one
{
border-style:solid;
border-width:5px;
}
p.two
{
border-style:solid;
border-width:medium;
}
```

Border Color

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

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

You can also set the border color to "transparent".

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

Example

```
p.one
{
border-style:solid;
border-color:red;
}
p.two
{
border-style:solid;
border-color:#98bf21;
}
```

CSS Text

Text Color

The color property is used to set the color of the text.

With CSS, a color is most often specified by:

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

Look at [CSS Color Values](#) for a complete list of possible color values.

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);}
```



For W3C compliant CSS: If you define the color property, you must also define the background-color property.

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 (like in magazines and newspapers).

Example

```
h1 {text-align:center;}
p.date {text-align:right;}
p.main {text-align:justify;}
```

Text Decoration

The text-decoration property is used to set or remove decorations from text.

The text-decoration property is mostly used to remove underlines from links for design purposes:

Example

```
a {text-decoration:none;}
```

It can also be used to decorate text:

Example

```
h1 {text-decoration:overline;}
h2 {text-decoration:line-through;}
h3 {text-decoration:underline;}
h4 {text-decoration:blink;}
```

Try it yourself >>



It is not recommended to underline text that is not a link, as this often confuses users.

Text Transformation

The text-transform property is used to specify uppercase and 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.

Example

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

Text Indentation

The text-indentation property is used to specify the indentation of the first line of a text.

Example

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

All CSS Text Properties

Property	Description
color	Sets the color of text
direction	Specifies the text direction/writing direction
letter-spacing	Increases or decreases the space between characters in a text
line-height	Sets the line height
text-align	Specifies the horizontal alignment of text
text-decoration	Specifies the decoration added to text
text-indent	Specifies the indentation of the first line in a text-block
text-shadow	Specifies the shadow effect added to text
text-transform	Controls the capitalization of text
vertical-align	Sets the vertical alignment of an element
white-space	Specifies how white-space inside an element is handled
word-spacing	Increases or decreases the space between words in a text

Grouping Selectors

In style sheets there are often elements with the same style.

```
h1
{
color:green;
}
h2
{
```

```
color:green;
}
p
{
color:green;
}
```

To minimize the code, you can group selectors.

Separate each selector with a comma.

In the example below we have grouped the selectors from the code above:

Example

```
h1,h2,p
{
color:green;
}
```

Nesting Selectors

It is possible to apply a style for a selector within a selector.

In the example below, one style is specified for all p elements, one style is specified for all elements with class="marked", and a third style is specified only for p elements within elements with class="marked":

Example

```
p
{
color:blue;
text-align:center;
}
.marked
{
background-color:red;
}
.marked p
{
color:white;
}
```

Positioning

The CSS positioning properties allow you to position an element. It can also place an element behind another, and specify what should happen when an element's content is too big.

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.

Static Positioning

HTML elements are positioned static by default. A static positioned element is always positioned according to the normal flow of the page.

Static positioned elements are not affected by the top, bottom, left, and right properties.

Fixed Positioning

An element with fixed position is positioned relative to the browser window.

It will not move even if the window is scrolled:

Example

```
p.pos_fixed
{
  position:fixed;
  top:30px;
  right:5px;
}
```

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

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

Fixed positioned elements can overlap other elements.

Relative Positioning

A relative positioned element is positioned relative to its normal position.

Example

```
h2.pos_left
{
  position:relative;
  left:-20px;
}
h2.pos_right
{
  position:relative;
  left:20px;
}
```

The content of relatively positioned elements can be moved and overlap other elements, but the reserved space for the element is still preserved in the normal flow.

Example

```
h2.pos_top
{
  position:relative;
  top:-50px;
}
```

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

Absolute Positioning

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>:

Example

```
h2
{
```

```
position:absolute;
left:100px;
top:150px;
}
```

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

Absolutely positioned elements can overlap other elements.

Overlapping Elements

When elements are positioned outside the normal flow, they can overlap other elements.

The z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others).

An element can have a positive or negative stack order:

Example

```
img
{
position:absolute;
left:0px;
top:0px;
z-index:-1
}
```

An element with greater stack order is always in front of an element with a lower stack order.

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

All CSS Positioning Properties

The number in the "CSS" column indicates in which CSS version the property is defined (CSS1 or CSS2).

Property	Description	Values	CSS
bottom	Sets the bottom margin edge for a positioned box	auto <i>length</i> % inherit	2
clip	Clips an absolutely positioned element	<i>shape</i> auto inherit	2
cursor	Specifies the type of cursor to be displayed	<i>url</i> auto crosshair default pointer move e-resize ne-resize nw-resize n-resize se-resize sw-resize s-resize w-resize text wait help	2

<u>left</u>	Sets the left margin edge for a positioned box	auto <i>length</i> % inherit	2
<u>overflow</u>	Specifies what happens if content overflows an element's box	auto hidden scroll visible inherit	2
<u>position</u>	Specifies the type of positioning for an element	absolute fixed relative static inherit	2
<u>right</u>	Sets the right margin edge for a positioned box	auto <i>length</i> % inherit	2
<u>top</u>	Sets the top margin edge for a positioned box	auto <i>length</i> % inherit	2
<u>z-index</u>	Sets the stack order of an element	<i>number</i> auto inherit	2

What is CSS Float?

With CSS float, an element can be pushed to the left or right, allowing other elements to wrap around it.

Float is very often used for images, but it is also useful when working with layouts.

How Elements Float

Elements are floated horizontally, this means that an element can only be floated left or right, not up or down.

A floated element will move as far to the left or right as it can. Usually this means all the way to the left or right of the containing element.

The elements after the floating element will flow around it.

The elements before the floating element will not be affected.

If an image is floated to the right, a following text flows around it, to the left:

Example

```
img
{
float:right;
}
```

Floating Elements Next to Each Other

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

Here we have made an image gallery using the float property:

Example

```
.thumbnail
{
float:left;
width:110px;
height:90px;
margin:5px;
}
```

Turning off Float - Using Clear

Elements after the floating element will flow around it. To avoid this, use the clear property.

The clear property specifies which sides of an element other floating elements are not allowed.

Add a text line into the image gallery, using the clear property:

Example

```
.text_line
{
clear:both;
}
```

All CSS Float Properties

The number in the "CSS" column indicates in which CSS version the property is defined (CSS1 or CSS2).

Property	Description	Values	CSS
clear	Specifies which sides of an element where other floating elements are not allowed	left right both none inherit	1
float	Specifies whether or not a box should float	left right none inherit	1

Navigation Bars

Having easy-to-use navigation is important for any web site.

With CSS you can transform boring HTML menus into good-looking navigation bars.

Navigation Bar = List of Links

A navigation bar needs standard HTML as a base.

In our examples we will build the navigation bar from a standard HTML list.

A navigation bar is basically a list of links, so using the and elements makes perfect sense:

Example

```
<ul>
<li><a href="default.asp">Home</a></li>
<li><a href="news.asp">News</a></li>
<li><a href="contact.asp">Contact</a></li>
<li><a href="about.asp">About</a></li>
</ul>
```

Now let's remove the bullets and the margins and padding from the list:

```
ul
{
list-style-type:none;
margin:0;
padding:0;
}
```

Example explained:

- list-style-type:none - Removes the bullets. A navigation bar does not need list markers
- Setting margins and padding to 0 to remove browser default settings

The code in the example above is the standard code used in both vertical, and horizontal navigation bars.

Vertical Navigation Bar

To build a vertical navigation bar we only need to style the <a> elements, in addition to the code above:

Example

```
a
{
display:block;
width:60px;
}
```

Example explained:

- display:block - Displaying the links as block elements makes the whole link area clickable (not just the text), and it allows us to specify the width
- width:60px - Block elements take up the full width available by default. We want to specify a 60 px width

Tip: Also take a look at our [fully styled vertical navigation bar example](#).

Note: Always specify the width for <a> elements in a vertical navigation bar. If you omit the width, IE6 can produce unexpected results.

Horizontal Navigation Bar

There are two ways to create a horizontal navigation bar. Using **inline** or **floating** list items.

Both methods work fine, but if you want the links to be the same size, you have to use the floating method.

Inline List Items

One way to build a horizontal navigation bar is to specify the elements as inline, in addition to the "standard" code above:

Example

```
li
{
display:inline;
}
```

Example explained:

- `display:inline;` - By default, `` elements are block elements. Here, we remove the line breaks before and after each list item, to display them on one line

Floating List Items

In the example above the links have different widths.

For all the links to have an equal width, float the `` elements and specify a width for the `<a>` elements:

Example

```
li
{
float:left;
}
a
{
display:block;
width:60px;
}
```

Example explained:

- `float:left` - use float to get block elements to slide next to each other
- `display:block` - Displaying the links as block elements makes the whole link area clickable (not just the text), and it allows us to specify the width
- `width:60px` - Since block elements take up the full width available, they cannot float next to each other. We specify the width of the links to 60px