**What is CSS?**

CSS is short for Cascading Style Sheets and is the primary language used to describe look and formatting for webpages across the Internet and documents of markup (e.g. HTML and XML) in general.

A markup language like HTML was initially designed to provide information about formatting and looks itself, but it soon became clear that it would make much more sense to split this into two layers: Document Content and Document Presentation, with CSS fulfilling the task of the latter. Historically that is why HTML has tags like font, which sole purpose is to adjust font family, color and size locally, a job that is today handled by CSS. This allows the developer to re-use formatting rules across several places in the same document and even across multiple documents. Here's an example to prove my point, and don't worry if it's not entirely clear to you what it does - all aspects will be explained throughout this tutorial:

CSS is used to control the style of a web document in a simple and easy way.

CSS is the acronym for "Cascading Style Sheet". This tutorial covers both the versions CSS1,CSS2 and CSS3, and gives a complete understanding of CSS, starting from its basics to advanced concepts.

# Audience

This tutorial will help both students as well as professionals who want to make their websites or personal blogs more attractive.

# Prerequisites

You should be familiar with:

* Basic word processing using any text editor.
* How to create directories and files.
* How to navigate through different directories.
* Internet browsing using popular browsers like Internet Explorer or Firefox.
* Developing simple Web Pages using HTML or XHTML.

# What is CSS?

**C**ascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

## **Advantages of CSS**

* **CSS saves time** − You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
* **Pages load faster** − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
* **Easy maintenance** − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
* **Superior styles to HTML** − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
* **Multiple Device Compatibility** − Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
* **Global web standards** − Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
* **Offline Browsing** − CSS can store web applications locally with the help of an offline catche.Using of this, we can view offline websites.The cache also ensures faster loading and better overall performance of the website.
* **Platform Independence** − The Script offer consistent platform independence and can support latest browsers as well.

## **Who Creates and Maintains CSS?**

CSS was invented by **Håkon Wium Lie** on October 10, 1994 and maintained through a group of people within the W3C called the CSS Working Group. The CSS Working Group creates documents called **specifications**. When a specification has been discussed and officially ratified by W3C members, it becomes a recommendation.

These ratified specifications are called recommendations because the W3C has no control over the actual implementation of the language. Independent companies and organizations create that software.

**NOTE** − The World Wide Web Consortium, or W3C is a group that makes recommendations about how the Internet works and how it should evolve.

## **CSS Versions**

Cascading Style Sheets, level 1 (CSS1) was came out of W3C as a recommendation in December 1996. This version describes the CSS language as well as a simple visual formatting model for all the HTML tags.

CSS2 was became a W3C recommendation in May 1998 and builds on CSS1. This version adds support for media-specific style sheets e.g. printers and aural devices, downloadable fonts, element positioning and tables.

CSS3 was became a W3C recommendation in June 1999 and builds on older versions CSS. it has divided into documentations is called as Modules and here each module having new extension features defined in CSS2.

### **CSS3 Modules**

CSS3 Modules are having old CSS specifications as well as extension features.

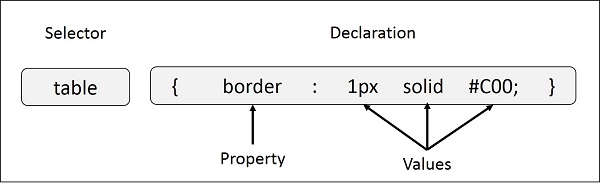
* Selectors
* Box Model
* Backgrounds and Borders
* Image Values and Replaced Content
* Text Effects
* 2D/3D Transformations
* Animations
* Multiple Column Layout
* User Interface

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts −

* **Selector** − A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.
* **Property** - A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be *color*, *border* etc.
* **Value** - Values are assigned to properties. For example, *color* property can have value either *red* or *#F1F1F1* etc.

You can put CSS Style Rule Syntax as follows −

selector { property: value }



**Example:** You can define a table border as follows −

table{ border :1px solid #C00; }

Here table is a selector and border is a property and given value *1px solid #C00* is the value of that property.

You can define selectors in various simple ways based on your comfort. Let me put these selectors one by one.

## **The Type Selectors**

This is the same selector we have seen above. Again, one more example to give a color to all level 1 headings:

h1 {

color: #36CFFF;

}

## **The Universal Selectors**

Rather than selecting elements of a specific type, the universal selector quite simply matches the name of any element type −

\* {

color: #000000;

}

This rule renders the content of every element in our document in black.

## **The Descendant Selectors**

Suppose you want to apply a style rule to a particular element only when it lies inside a particular element. As given in the following example, style rule will apply to <em> element only when it lies inside <ul> tag.

ul em {

color: #000000;

}

## **The Class Selectors**

You can define style rules based on the class attribute of the elements. All the elements having that class will be formatted according to the defined rule.

.black {

color: #000000;

}

This rule renders the content in black for every element with class attribute set to *black* in our document. You can make it a bit more particular. For example:

h1.black {

color: #000000;

}

This rule renders the content in black for only <h1> elements with class attribute set to *black*.

You can apply more than one class selectors to given element. Consider the following example:

<p class="center bold">

This para will be styled by the classes *center* and *bold*.

</p>

## **The ID Selectors**

You can define style rules based on the *id* attribute of the elements. All the elements having that *id* will be formatted according to the defined rule.

#black {

color: #000000;

}

This rule renders the content in black for every element with *id* attribute set to *black* in our document. You can make it a bit more particular. For example −

h1#black {

color: #000000;

}

This rule renders the content in black for only <h1> elements with *id* attribute set to *black*.

The true power of *id* selectors is when they are used as the foundation for descendant selectors, For example:

#black h2 {

color: #000000;

}

In this example all level 2 headings will be displayed in black color when those headings will lie with in tags having *id* attribute set to *black*.

## **The Child Selectors**

You have seen the descendant selectors. There is one more type of selector, which is very similar to descendants but have different functionality. Consider the following example −

body > p {

color: #000000;

}

This rule will render all the paragraphs in black if they are direct child of <body> element. Other paragraphs put inside other elements like <div> or <td> would not have any effect of this rule.

## **The Attribute Selectors**

You can also apply styles to HTML elements with particular attributes. The style rule below will match all the input elements having a type attribute with a value of *text* −

input[type = "text"]{

color: #000000;

}

The advantage to this method is that the <input type = "submit" /> element is unaffected, and the color applied only to the desired text fields.

There are following rules applied to attribute selector.

* **p[lang]** - Selects all paragraph elements with a *lang* attribute.
* **p[lang="fr"]** - Selects all paragraph elements whose *lang* attribute has a value of exactly "fr".
* **p[lang~="fr"]** - Selects all paragraph elements whose *lang* attribute contains the word "fr".
* **p[lang|="en"]** - Selects all paragraph elements whose *lang* attribute contains values that are exactly "en", or begin with "en-".

## **Multiple Style Rules**

You may need to define multiple style rules for a single element. You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example −

h1 {

color: #36C;

font-weight: normal;

letter-spacing: .4em;

margin-bottom: 1em;

text-transform: lowercase;

}

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

For a while, don't bother about the properties mentioned in the above block. These properties will be explained in the coming chapters and you can find complete detail about properties in [CSS References](https://www.tutorialspoint.com/css/css_references.htm).

## **Grouping Selectors**

You can apply a style to many selectors if you like. Just separate the selectors with a comma, as given in the following example −

h1, h2, h3 {

color: #36C;

font-weight: normal;

letter-spacing: .4em;

margin-bottom: 1em;

text-transform: lowercase;

}

This define style rule will be applicable to h1, h2 and h3 element as well. The order of the list is irrelevant. All the elements in the selector will have the corresponding declarations applied to them.

You can combine the various *id* selectors together as shown below −

#content, #footer, #supplement {

position: absolute;

left: 510px;

width: 200px;

}

There are four ways to associate styles with your HTML document. Most commonly used methods are inline CSS and External CSS.

## **Embedded CSS - The <style> Element**

You can put your CSS rules into an HTML document using the <style> element. This tag is placed inside <head>...</head> tags. Rules defined using this syntax will be applied to all the elements available in the document. Here is the generic syntax −

Following is the example of embed CSS based on the above syntax −

<!DOCTYPE html>

<html>

<head>

<style type = "text/css" media = "all">

body {

background-color: linen;

}

h1 {

color: maroon;

margin-left: 40px;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

It will produce the following result −

### **Attributes**

Attributes associated with <style> elements are −

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| type | text/css | Specifies the style sheet language as a content-type (MIME type). This is required attribute. |
| media | screen  tty  tv  projection  handheld  print  braille  aural  all | Specifies the device the document will be displayed on. Default value is *all*. This is an optional attribute. |

## **Inline CSS - The *style* Attribute**

You can use *style* attribute of any HTML element to define style rules. These rules will be applied to that element only. Here is the generic syntax −

<element style = "...style rules....">

### **Attributes**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| style | style rules | The value of *style* attribute is a combination of style declarations separated by semicolon (;). |

### **Example**

Following is the example of inline CSS based on the above syntax −

<html>

<head>

</head>

<body>

<h1 style = "color:#36C;"> This is inline CSS </h1>

</body>

</html>

It will produce the following result −

## **External CSS - The <link> Element**

The <link> element can be used to include an external stylesheet file in your HTML document.

An external style sheet is a separate text file with **.css** extension. You define all the Style rules within this text file and then you can include this file in any HTML document using <link> element.

Here is the generic syntax of including external CSS file −

<head>

<link type = "text/css" href = "..." media = "..." />

</head>

### **Attributes**

Attributes associated with <style> elements are −

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| type | text/css | Specifies the style sheet language as a content-type (MIME type). This attribute is required. |
| href | URL | Specifies the style sheet file having Style rules. This attribute is a required. |
| media | screen  tty  tv  projection  handheld  print  braille  aural  all | Specifies the device the document will be displayed on. Default value is *all*. This is optional attribute. |

### **Example**

Consider a simple style sheet file with a name *mystyle.css* having the following rules −

h1, h2, h3 {

color: #36C;

font-weight: normal;

letter-spacing: .4em;

margin-bottom: 1em;

text-transform: lowercase;

}

Now you can include this file *mystyle.css* in any HTML document as follows −

<head>

<link type = "text/css" href = "mystyle.css" media = " all" />

</head>

## **Imported CSS - @import Rule**

@import is used to import an external stylesheet in a manner similar to the <link> element. Here is the generic syntax of @import rule.

<head>

<@import "URL";

</head>

Here URL is the URL of the style sheet file having style rules. You can use another syntax as well −

<head>

<@import url("URL");

</head>

### **Example**

Following is the example showing you how to import a style sheet file into HTML document −

<head>

@import "mystyle.css";

</head>

## **CSS Rules Overriding**

We have discussed four ways to include style sheet rules in a an HTML document. Here is the rule to override any Style Sheet Rule.

* Any inline style sheet takes highest priority. So, it will override any rule defined in <style>...</style> tags or rules defined in any external style sheet file.
* Any rule defined in <style>...</style> tags will override rules defined in any external style sheet file.
* Any rule defined in external style sheet file takes lowest priority, and rules defined in this file will be applied only when above two rules are not applicable.

## **Handling old Browsers**

There are still many old browsers who do not support CSS. So, we should take care while writing our Embedded CSS in an HTML document. The following snippet shows how you can use comment tags to hide CSS from older browsers −

<style type="text/css">

<!--

body, td {

color: blue;

}

-->

</style>

## **CSS Comments**

Many times, you may need to put additional comments in your style sheet blocks. So, it is very easy to comment any part in style sheet. You can simple put your comments inside /\*.....this is a comment in style sheet.....\*/.

You can use /\* ....\*/ to comment multi-line blocks in similar way you do in C and C++ programming languages.

### **Example**

<!DOCTYPE html>

<html>

<head>

<style>

p {

color: red;

/\* This is a single-line comment \*/

text-align: center;

}

/\* This is a multi-line comment \*/

</style>

</head>

<body>

<p>Hello World!</p>

</body>

</html>

It will produce the following result −

**Old style text formatting, using only HTML:**

This is a piece of  
<font face="Tahoma,Verdana,Arial" color="Blue" size="3"><i><b>text</b></i></font> with  
<font face="Tahoma,Verdana,Arial" color="Blue" size="3"><i><b>highlighted</b></i></font> elements in  
<font face="Tahoma,Verdana,Arial" color="Blue" size="3"><i><b>it</b></i></font>.

**A more modern approach with CSS:**

<style type="text/css">  
.highlight {  
        color: Blue;  
        font-style: italic;  
        font-weight: bold;  
        font-size: 120%;  
        font-family: Tahoma, Verdana, Arial;  
}  
</style>  
  
This is a piece of  
<span class="highlight">text</span> with  
<span class="highlight">highlighted</span> elements in  
<span class="highlight">it</span>.

Notice how I simply re-use the same set of rules across several HTML tags. This is already an advantage when using it three times, like we do in the example, but it doesn't end there - put the CSS in an external stylesheet file (more on that later) and you can use the same rules across your ENTIRE website. And how about when you decide that highlighted text should be red instead of blue? With the first approach, you would have to manually edit the tags everywhere you used it - with CSS, just change the single ".highlight" rule!