

Quick Reference Guide to Cascading Style Sheets

– or *how to add stylesheets to your webpage. Quickly.*

Using Stylesheets

Stylesheets can be added to a page in three ways, and here's how you do it:

1 – Link your webpage to an external CSS file (*The Best Way!*)

Insert the following into your head section (ie. somewhere between <head> and </head>):

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

Note that “mystylesheet.css” should be replaced with the name of your own stylesheet file. Also, remember that **a style sheet file is just a plain text file saved with the extension, “.css”**

If you're going to use more advanced stylesheet features, such as positioning, it can be a good idea to hide your stylesheet from older browsers who's bugged CSS rendering engines may break your page (Netscape Navigator 4.7 being the prime example. Shudder!)

This is one way to do it:

```
<style type="text/css"> @import url("mystylesheet.css"); </style>
```

Older browsers don't understand the @import rule, so they ignore it and cannot see the linked stylesheet.

2 – Embed the stylesheet in an HTML document

Insert some style tags...

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

...into the head section of your document, and write your entire stylesheet between them:

```
<style type="text/css">
<!--
    body { font-size: 1.2em; }
    h1 { color: #099; }
    etc...
-->
</style>
```

Some people enclose the style declarations within HTML comment tags (<!-- -->, as above) to protect older browsers.

3 – Embed CSS styles with HTML elements (*can be messy if used extensively*)

Add a style declaration to individual HTML tags:

```
<p style="color: green">This is green</p>
```

How to write CSS

CSS has a straightforward syntax which is easy to learn.

A stylesheet file is a plain text file which is saved with the extension, “.css”. It contains a series of statements, or **rules**, which tell how the HTML element they refer to should be displayed.

Each rule has just two parts; a selector and a declaration. The declaration is enclosed within curly brackets:

```
selector    declaration  
  
p           { color: blue }
```

You'll notice that there are also two parts to the declaration – a **property** (color:), and a **value** (blue).

The selector relates the rule to a particular element of the webpage (in this example, the paragraph tag <p>), and the declaration states how that element should be displayed (in this example, coloured blue).

That's all there is to it... almost!

You can make lots of declarations for each rule, ie:

```
p {  
    color: blue;  
    font-weight: bold;  
    line-height: 1.5em;  
}
```

From this example, you'll also notice some other things:

- every property **must** be followed by a colon, “:”
- if you have multiple declaration for a single selector, each **has to be** finished with a semi-colon, “;”
- You can layout the stylesheet rule to make it easier to read – as with HTML, extra spaces don't matter.
- There **must** be a space between the selector and the declaration.
- There **must** be a space between the property and its value, **after** the separating colon.

Classes and IDs

As well as being able to select HTML elements with our stylesheet rules, we can use **Classes** and **IDs** for even finer control.

HTML elements can be given a **Class** like this:

```
<p class="highlighted">This is a paragraph</p>
```

and they can be given an **ID** like this:

```
<h1 id="page-title">Welcome to my website</h1>
```

The name of the Class or ID can be anything you like.

When you get to writing the stylesheet, you'd reference a Class like this:

```
.highlighted { background-color: yellow }
```

and you'd reference an ID like this:

```
#page-title { text-align: center; }
```

What's the difference between an ID and a Class?

A Class should be used to select recurring elements. In the example above, many paragraphs might be highlighted.

An ID should be used to select a unique element. In the example above, there is only one H1 element serving as the title to each page.

These aren't hard and fast rules – just Good Technique!

The Document Tree (why CASCADING Style Sheets?)

To fully understand how CSS works, we need to understand the concept of the “Document Tree”.

“Document Tree” = the tree of elements encoded in the source document

Every HTML element in a webpage has exactly one **parent**, with the example of the root element, `<html>`, which has none.

Example:

```
<body>
  <p>hello</p>
</body>
```

In this example, `<body>` is the **parent** of `<p>`, and `<p>` is the **child** of `<body>`.

This means something very important in practical terms. A style applied to the body element will be **“inherited”** by all elements within it.

In the above example, if my stylesheet applies the style, `body { font-family: Arial, sans-serif; }` then this formatting will be inherited by the paragraph, `<p>hello</p>`. Indeed, it will be inherited by all other paragraphs, headings, lists, etc. which I write in the body of my webpage.

Referencing specific areas of a webpage with DIV and SPAN

It's sometimes useful to be able to reference a specific area of your page that may contain many HTML elements. Alternatively, you might want to apply an effect to a particular part of a paragraph or heading instead of the whole lot. In the former case, we use DIVs. In the latter, we use SPANS.

DIV example:

```
HTML    <div id="navigation">
        <ul>
        <li><a href="index.html">Home page</a></li>
        <li><a href="about.html">About me</a></li>
        <li><a href="contact.html">Contact details</a></li>
        </ul>
    </div>
```

```
CSS    #navigation {
        width: 20%;
        float: right;
        border: 1px solid #000;
        padding: 1em;
    }
```

In the above example, the DIV tag is used in the webpage to identify part of the page as a navigation area. This makes no difference to what is seen in a browser, until we start to add some styling with CSS. The DIV is given the ID, "navigation", which is then used as a Selector in the CSS. (I used ID rather than CLASS, because there will only be one navigation area like this on my page).

The CSS rule written above will enclose the navigation area (everything between the DIV tags) in a box which is 20% of the screen width. The box is "floated" to the right of the page (i.e. text will now flow round it to the left and underneath), given a 1 pixel thick, black border, and given 1em of padding (padding is the space inside the box, between its edges and its content).

SPAN example:

```
HTML    <p>Welcome to the <span class="access-space">Access
        Space</span> website</p>
```

```
CSS    .access-space {
        font-weight: bold;
        color: #900;
        font-family: "Courier New", Courier, monospace;
    }
```

This example is a bit simpler. The SPAN tag is used to identify the name of an organisation to which I'd like to apply some specific styling. It is given the CLASS, "access-space". (I use CLASS rather than ID, because I will be using the name of this organisation many times on each webpage).

The CSS rule I've written causes the text to be rendered red, bold and in a monospace font.

CSS Properties

This list is by no means comprehensive, but it gives some basic stylesheet properties which should get you started.

For the full Monty, get yourself a copy of the CSS2 Spec. Available in a number of formats at: <http://www.w3.org/TR/REC-CSS2/>

	<i>Property</i>	<i>Example Values</i>
Font	color:	red; #990000; #900;
	font-family:	Arial, Helvetica, sans-serif; "Courier New", Courier, monospace; "Times New Roman", Times, serif;
	font-weight:	bold; normal;
	font-style:	italic; normal;
	font-size:	2em; 150%;
	text-align:	center; left; right; justify;
Background	background-color:	red; #990000; #900;
	background-image:	url(myimage.png);
	background-repeat:	repeat; no-repeat; repeat-x; repeat-y;
	background-position:	top; center; bottom; left; right; 20px 20px; /* x,y values */
Spacing	margin-left: margin-bottom: padding-right: padding-top: etc.	1.5em;
	padding: margin:	3em 1.5em 3em 1.5em /* this sets individual values for top, right, bottom and left, in that strict order */

	<i>Property</i>	<i>Example Values</i>
Borders	border-width: border-top-width: etc.	1px; 2%;
	border-style: border-bottom-style: etc.	solid; dotted; dashed; none;
	border-color: border-right-color: etc.	blue; #000099; #009;
	border: border-right: etc.	2px solid #009; /* this includes width, style and colour in the same property. Use width, style and colour values in that strict order, with a space after each. */

Recommendation: create a webpage with a variety of content, link a stylesheet to it, and start playing!