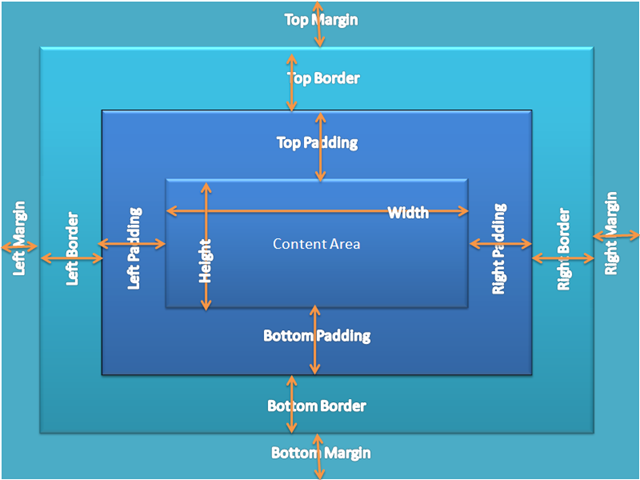
**The Box Model, CSS and Identifiers**

**The Box Model**

Every tag and object inside an HTML page can be thought of as box. These boxes have their own attributes and can be thought of as a simple diagram…



(Image from <http://codingarchitect.wordpress.com/author/rsendhil/page/2/> )

There’s a nice demonstration of this concept at CSS Box Model Exploratorium - <http://greystate.dk/resources/css-boxmodel/>

**Cascading Style Sheets**

The attributes of these boxes can be controlled through cascading style sheets. They are called cascading for good reason. The styles applied to box can be applied from an external file, internal styles and inline styles.

**External CSS**

Ideally, the styles should be applied from an external style sheet. The external style sheet and be applied to several pages and it makes updating the styles easier if they are all in one place. To specify which style sheet to use each page using it must specify the location of the file in the head section of the page…

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

It is possible to link to several style sheets using multiple links. In this case, if the style sheet contains different styles for the same box or object then the second style sheet will take precedence – another case of cascading. Doing this is not usually as good idea as it complicates matters. One good use for doing this is to specify a different style when printing a document using the media type property. For example

<head>  
<link rel="stylesheet" type="text/css" media="all" href="mystyle.css" />  
<link rel="stylesheet" type="text/css" media="print" href="print.css" />  
</head>

There are other uses for the media property such as braille, embossed, handheld, projection, screen, speech, tty and tv. See <http://www.w3.org/TR/CSS21/media.html> for a discussion of these.

There is another way of using external style sheets and that’s by using @import, for example

<head>  
<style type="text/css" media="all">  
@import "mystyle.css";  
@import "print.css";  
</style>  
</head>

The link method is preferred as @import was introduced to hide the style sheets from older browsers and the link method is faster, resulting in a shorter page loading time. For a discussion on this see <http://www.stevesouders.com/blog/2009/04/09/dont-use-import/>

**Internal CSS**

This is where the CSS is defined within the head section of a page. A page may need formatting unique to itself so it would make sense to create the CSS inside the page. As with the external CSS the styles are defined inside the head section of the page…

<head>  
<style type="text/css">  
. . . . .  
. . . . .  
. . . . .  
</style>  
</head>

**Inline CSS**

This is where the CSS is defined inside of a tag itself. Unless there is a particular reason for this to be done, it is not recommended. The point of CSS is to separate the content from the presentation of that content but there are times when this may be the best method to use.

**CSS Precedence**

Think of HTML tags as working for you and like any good worker, they respond to the last command you give them. With this in mind the order of precedence (from lowest to highest) is

Browser default style sheet  
External style sheet  
Internal style sheet  
Inline style sheet

Note: If you specify the external style sheet after the internal style sheet in the head section of the website then the external style sheet will take precedence, which negates the use of the internal style sheet in the first place.

If you are using an external and an internal style sheet always put the link to the external style sheet before you define the internal one.

Here’s an example of precedence…

The external style sheet has this structure…

h3  
{  
color:red;  
text-align:right;  
font-size:8pt;  
}

The internal style sheet has this structure…

h3  
{  
font-size:20pt;  
}

The inline style sheet has this structure…

<h3 style=”color:green;”>

The resulting text will be green (from the inline style), have a font size of 20pt (from the internal style) and be right aligned (from the external style sheet).

**Identities and Classes**

A class is can be reused for several elements by putting class=”xxxxx” inside the opening tag of that element. A class is referenced inside the CSS by using a “.”

An ID is unique and can only be used once. To give an element a unique ID use id=”xxxxx” inside the opening tag of the element. An id is referenced inside the CSS by using a “#”

It is possible to give an element both a class and an id.

It is possible to make an element a member of two or more classes, e.g.

<p class=”first second”

In the following examples this HTML code has been used…

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
<link rel="stylesheet" type="text/css" media="all" href="style.css" />  
<title>CSS Test</title>  
</head>  
  
<body>  
<div class="container" id="first">  
<p class="para1" id="fourth">Many a mickle makes a muckle</p>  
<p class="para2" id="fifth">A bird in the hand</p>  
<div class="content" id="second">  
<h1 class="para1">I'm a header</h1>  
<p class="para1" id="sixth">You can lead</p>  
<p class="para2" id="seventh">Too many cooks</p>  
</div>  
<div class="extras" id="third">  
<p class="para1" id="eighth">A Stitch in time</p>  
<p class="para2" id="ninth">All that glitters</p>  
<p class="para1">Least said</p>  
</div>  
</div>  
</body>  
</html>

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  **What’s Happening**  Because no identifiers have been used then all P tags within the page use the CSS.  **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  .content p h1{  color:green;  }  **What’s Happening**  This is actually user error. You cannot specify multiple tags within CSS like this unless the tags themselves are actually nested in this way. The browser does not understand what you are trying to do and simply skips that instruction. |

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  .container div p{  color:green;  }  **What’s Happening**  All P elements that are inside DIV elements inside the element with the class of “container” are green  **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  #first div p{  color:green;  }  **What’s Happening**  All P elements that are inside DIV elements inside the element with the id of “first” are green |

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  .para1{  color:green;  }  **What’s Happening**  The “.” means a class, so all elements of the class para1 are now green |

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  p.para1{  color:green;  }  **What’s Happening**  Only P tags with the class of Para1 are green. Although the header has the class of Para1 it is not a P tag |

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  .content{  color:green;  }  **What’s Happening**  Everything inside the element with the class of “content” is green BUT those tags that are specifically targeted elsewhere in the CSS are not affected  **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  #second{  color:green;  }  **What’s Happening**  Everything inside the element with the id of “second” is green BUT those tags that are specifically targeted elsewhere in the CSS are not affected |

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  #second p{  color:green;  }  **What’s Happening**  All the P elements inside the element with the id of “second” are green  **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  .content p{  color:green;  }  **What’s Happening**  All the P elements inside the element with the class of “content” are green |

|  |  |
| --- | --- |
|  | **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  .container p{  color:green;  }  **What’s Happening**  All the P elements inside the element with the class of “container” are green. It doesn’t matter the nesting of the box model – all targeted elements obey the CSS  **CSS**  p {  color:red;  font-size:20pt;  font-weight:bold;  }  #first p{  color:green;  }  **What’s Happening**  All the P elements inside the element with the id of “first” are green. It doesn’t matter the nesting of the box model – all targeted elements obey the CSS |

**Useful web pages**

<http://www.tizag.com/cssT/cssid.php>

<http://www.w3schools.com/css/css_pseudo_classes.asp> - About pseudo-classes

<http://www.htmlgoodies.com/beyond/css/article.php/3470231/So-You-Want-CSS-Classes-and-IDs-Huh.htm>

<http://www.1keydata.com/css-tutorial/class-id.php>

Ray Thomas  
October 2011