Introduction to CSS

Table of Contents

What is CSS?

Cascading Style Sheets (CSS) are used to define how to display HTML elements. Style sheets are usually stored in separate files with the .css extension but can also be embedded into the head of an HTML document.

Why is this Used?

The original specification of HTML was to define a document's content. Formatting of the document was supposed to be the job of the browser, not using formatting tags (i.e. <font>). Browsers continued to add new HTML tags and attributes to the HTML language to make formatting more versatile. This caused the big problem with not being able to recognize what part of the document was actually content and what part of it was formatting. A lot of websites today are still written like this and it makes it very complicated to read or edit the source code with all the extra formatting tags.

This problem has now been solved by the World Wide Web Consortium (W3C). W3C has added styles for the new version of HTML. All major browsers now support the use of CSS.

Style sheets can also save a huge amount of work. Instead of having to add in all the extra formatting tags inside every HTML document, a single style sheet can be used for all pages for a website. A style sheet defines styles for individual HTML elements. This allows developers to become more consistent with the layout and styles that are used on a website. To make a formatting change to an entire site you only need to change the CSS document. Changing the CSS document automatically updates all HTML documents that use the same CSS file.

Adding CSS to an HTML Document

There are multiple ways to add CSS to an HTML document. The first way you should add CSS to an HTML document is to create a separate CSS file and link to it within the head of the HTML document. Anything defined in an external style sheet will override the browser’s default style.

Example:

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

If you have a large website and use a basic external style sheet for the entire site you may also want to define an internal style sheet within an HTML document. This is done by writing the styles within the head of the document. Any styles defined in an internal style sheet will override both the external style sheet and the browser’s default style.

Example:

```
<head>
        …
        <style type="text/css">
                body {
                        background-color:blue;
                }
                …
        </style>
        …
</head>
```

The last way to define a style is to add it to the tag itself using the *style* attribute. This option is not generally used unless absolutely needed. It will make the source code of an HTML document less readable and very ugly. This can be avoided at all times if you use class declarations (explained later). Any styles defined inline are given the highest precedence and will override all other styles.

Example:

```
<p style=”padding:100px;”>A paragraph with a padding of 100px</p>
```

Cascading Order

The three different ways to define custom styles have now been introduced. Each way is given an order of precedence, meaning if you define styles in multiple places, the one defined with the highest precedence will override the others. Order of precedence from highest to lowest is as follows:

1) Inline style
2) Internal Style Sheet
3) External Style Sheet
4) Browser Default

If there is no style defined for an element it will be set as the browser’s default value.

Tips on Adding Styles

Start with an external style sheet and define all styles there first. If you need to override a style use an internal style sheet. You should never need to use an inline style but it is not wrong to do so.

Writing Styles

We now know what CSS is, why it was developed and how it is used. Now let’s actually learn CSS. Learning CSS is similar to how you learn HTML. First you learn the basic concepts of how styles are written, then you learn all the styles and experiment.

Format

Defining a style for any element of HTML is done in the same format. First you write the HTML element followed by an enclosing set of curly braces. Inside the curly braces you can then add the styles. This is done by writing the property you wish to change, followed by a : (colon) then the value of the property, and finally ended with a ; (semi-colon). You can add as many property/value pairs to get the desired style.

Example:

```
        element {
                property:value;
                property:value;
                …
        }
```

You can also group multiple elements by separating them with a comma and then define the style for the group of elements

Example:

```
        element1, element2, element3, element4 {
                property:value;
                …
        }
```

Notes and Tips

Always separate multiple values by putting them on a new line. It makes it much more readable.

Classes

An element may also have more than one style. This is done by defining different classes for the element. A class is defined by appending *.class_name* to the element when defining the style for the element.

Example:

```
element.class1 {
        …
}
element.class2 {
        …
}
```

When you define classes you must add this attribute inside your HTML for it to be able to determine which style to use.

Example:

```
<h2 class=”center”>The center class makes this header center-aligned</h2>
```

Another similar way to defining multiple styles to the same element is to give the element an *id* attribute and define a style for the *id*. The id is defined by using *#class_name* as the element name.

Example:

```
#id_name {
        …
}
```

When you define a style like this you must give the elements in the HTML that you want to use the style the *id* attribute.

Exampel:

```
<h2 id=”center”>The center id makes this header center-aligned</h2>
```

Naming conventions

The first rule when creating names for classes and ids is that you cannot start a name with a number. Second rule is to make the names short but descriptive. Third rule is to keep the formatting of the names consistent. Here are two ways you could go about formatting names.

1. Use all lower case letters separating words with a _ (underscore)
   i.e. main_navigation
2. Separate words by starting new words with capital letters
   i.e. mainNavigation

Use any format you like best but remember to stay consistent throughout.

Comments

When writing styles it is very important to add in comments to explain what the styles actually do. Sometimes it may not be clear what is actually being done when reading the style sheet. It makes it very helpful to have comments to explain what more complicated styles are actually doing.

Example:

```
/* This is a comment in CSS */
```

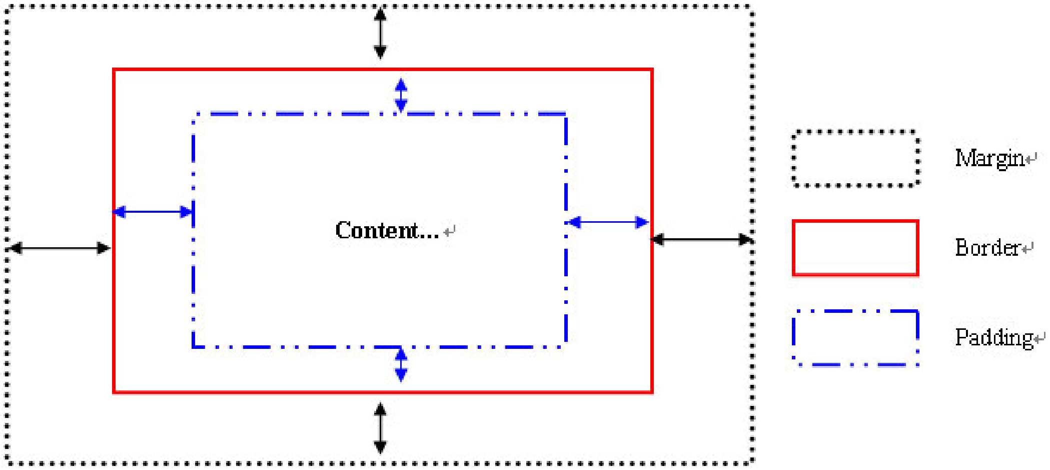
Anything between the two * is a comment and will not be interpreted by a browser.

CSS Properties

The rest of learning CSS is to familiarize yourself with all the properties and experimenting with them. A list of all the properties can be found at: http://www.w3schools.com/css/

Box Model

Trying to visualize how exactly margins, padding and borders are laid out can be confusing at the best of times. Here is a simple diagram that you can refer to whenever you need to clear things up.



Margins are not visible and thus make it hard to actually see where they are. Borders are very useful because it is the point where the margin extends from as well as the padding. If you have a large amount of content that fills up the content space you may be able to see the padding a little more clearly.

Tips on Using the Box Model

When first starting to format your webpage using the box model make a visible border and justify your text. This will make it easier to visualize where the padding and the margins are.

Remember that some elements have default style settings in browsers. They may not be the same from browser to browser. Always check that your web pages look the same in at least two different browsers (i.e. Internet Explorer and Mozilla FireFox).

Examples

You can download all the examples from;

http://aco.uwaterloo.ca/css.zip

Other Resources:

http://www.w3.org/TR/css3-roadmap/

http://www.w3.org/MarkUp/Guide/Style

http://www.w3schools.com/css/css_intro.asp

http://www.csszengarden.com/