Adding a touch of style

Dave Raggett, 8th April 2002.

This is a short guide to styling your Web pages. It will show you how to use W3C's Cascading Style Sheets language (CSS) as well as alternatives using HTML itself. The route will steer you clear of most of the problems caused by differences between different brands and versions of browsers.

For style sheets to work, it is important that your markup be free of errors. A convenient way to automatically fix markup errors is to use the HTML Tidy utility. This also tidies the markup making it easier to read and easier to edit. I recommend you regularly run Tidy over any markup you are editing. Tidy is very effective at cleaning up markup created by authoring tools with sloppy habits.

The following will teach you how to:

- use the style element
- link to separate style sheets
- set page margins
- set left and right and first-line indents
- set the amount of whitespace above and below
- set the font type, style and size
- add borders and backgrounds
- set colors with named or numeric values
- add style for browsers that don't understand CSS

Getting started

Let's start with setting the color of the text and the background. You can do this by using the STYLE element to set style properties for the document's tags:

```
<style type="text/css">
  body { color: black; background: white; }
</style>
```

The style element is placed within the document head. Here is a template HTML file showing where the above style element fits. You can copy and paste this into your editor as a convenient way to experiment with CSS style sheets:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title> replace with your document's title </title>
<style type="text/css">
  body { color: black; background: white; }
</style>
</head>
<body>

replace with your document's content

</body>
</html>
```

The stuff between the <style> and </style> is written in special notation for style rules. Each rule starts with a tag name followed by a list of style properties bracketed by { and }. In this example, the rule matches the *body* tag. As you will see, the body tag provides the basis for setting the overall look and feel of your Web page.

Each style property starts with the property's name, then a colon and lastly the value for this property. When there is more than one style property in the list, you need to use a semicolon between each of them to delimit one property from the next. In this example, there are two properties - "color" which sets the color of the text, and "background" which sets the color of the page background. I recommend always adding the semicolon even after the last property.

Colors can be given as names or as numerical values, for instance `rgb(255, 204, 204)` which is a fleshy pink. The 3 numbers correspond to red, green and blue respectively in the range 0 to 255. You can also use a hexadecimal notation, the same color can also be written as `#FFCCCC`. More details on color are given in a later section.

Note that the style element must be placed in the document's head along with the title element. It shouldn't be placed within the body.

Linking to a separate style sheet

If you are likely to want to use the same styles for several Web pages it is worth considering using a separate style sheet which you then link from each page. You can do this as follows:

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

The *link* tag should be placed within the <head> ... </head> element. Here is an HTML file with a link to an external style sheet:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title> replace with your document's title </title>
<link type="text/css" rel="stylesheet" href="style.css">
</head>
<body>

replace with your document's content

</body>
</html>
```

The link element's *rel* attribute must be set to the value "stylesheet" to allow the browser to recognize that the *href* attribute gives the Web address (URL) for your style sheet. A simple stylesheet file might look like the following:

```
/* style.css - a simple style sheet */
body {
  margin-left: 10%; margin-right: 10%;
  color: black; background: white;
}
```

Note that the same HTML file can link to an external style sheet and also include a style element for additional style settings specific to this page. If you place the link element before the style element, you can use the latter to override the style settings in the linked style sheet.

Setting the page margins

Web pages look a lot nicer with bigger margins. You can set the left and right margins with the "margin-left" and "margin-right" properties, e.g.

```
<style type="text/css">
  body { margin-left: 10%; margin-right: 10%; }
</style>
```

This sets both margins to 10% of the window width, and the margins will scale when you resize the browser window.

Setting left and right indents

To make headings a little more distinctive, you can make them start within the margin set for the body, e.g.

```
<style type="text/css">
  body { margin-left: 10%; margin-right: 10%; }
  h1 { margin-left: -8%;}
  h2,h3,h4,h5,h6 { margin-left: -4%; }
</style>
```

This example has three style rules. One for the body, one for h1 (used for the most important headings) and one for the rest of the headings (h2, h3, h4, h5 and h6). The margins for the headings are additive to the margins for the body. Negative values are used to move the start of the headings to the left of the margin set for the body.

In the following sections, the examples of particular style rules will need to be placed within the style element in the document's head (if present) or in a linked style sheet.

Controlling the white space above and below

Browsers do a pretty good job for the white space above and below headings and paragraphs etc. Two reasons for taking control of this yourself are: when you want a lot of white space before a particular heading or paragraph, or when you need precise control for the general spacings.

The "margin-top" property specifies the space above and the "margin-bottom" specifies the space below. To set these for all h2 headings you can write:

```
h2 { margin-top: 8em; margin-bottom: 3em; }
```

The em is a very useful unit as it scales with the size of the font. One em is the height of the font. By using em's you can preserve the general look of the Web page independently of the font size. This is much safer than alternatives such as pixels or points, which can cause problems for users who need large fonts to read the text.

Points are commonly used in word processing packages, e.g. 10pt text. Unfortunately the same point size is rendered differently on different browsers. What works fine for one browser will be illegible on another! Sticking with em's avoids these problems.

To specify the space above a particular heading, you should create a named style for the heading. You do this with the *class* attribute in the markup, e.g.

```
<h2 class="subsection">Getting started</h2>
```

The style rule is then written as:

```
h2.subsection { margin-top: 8em; margin-bottom: 3em; }
```

The rule starts with the tag name, a dot and then the value of the class attribute. Be careful to avoid placing a space before or after the dot. If you do the rule won't work. There are other ways to set the styles for a particular element but the class attribute is the most flexible.

When a heading is followed by a paragraph, the value for margin-bottom for the heading isn't added to the value for margin-top for the paragraph. Instead, the maximum of the two values is used for the spacing between the heading and paragraph. This subtlety applies to margin-top and margin-bottom regardless of which tags are involved.

First-line indent

Sometimes you may want to indent the first line of each paragraph. The following style rule emulates the traditional way paragraphs are rendered in novels:

```
p { text-indent: 2em; margin-top: 0; margin-bottom: 0; }
```

It indents the first line of each paragraph by 2 em's and suppresses the inter-paragraph spacing.

Controlling the font

This section explains how to set the font and size, and how to add italic, bold and other styles.

Font styles

The most common styles are to place text in italic or bold. Most browsers render the *em* tag in italic and the *strong* tag in bold. Let's assume you instead want em to appear in *bold italic* and strong in **bold uppercase**:

```
em { font-style: italic; font-weight: bold; }
strong { text-transform: uppercase;  font-weight: bold; }
```

If you feel so inclined, you can fold headings to lower case as follows:

```
h2 { text-transform: lowercase; }
```

Setting the font size

Most browsers use a larger font size for more important headings. If you override the default size, you run the risk of making the text too small to be legible, particularly if you use points. You are therefore recommended to specify font sizes in relative terms.

This example sets heading sizes in percentages relative to the size used for normal text:

```
h1 { font-size: 200%; }
h2 { font-size: 150%; }
h3 { font-size: 100%; }
```

Setting the font family

It is likely that your favorite font won't be available on all browsers. To get around this, you are allowed to list several fonts in preference order. There is a short list of generic font names which are guaranteed to be available, so you are recommended to end your list with one of these: serif, sans-serif, cursive, fantasy, or monospace, for instance:

```
body { font-family: Verdana, sans-serif; }
h1,h2 { font-family: Garamond, "Times New Roman", serif; }
```

In this example, important headings would preferably be shown in Garamond, failing that in Times New Roman, and if that is unavailable in the browsers default serif font. Paragraph text would appear in Verdana or if that is unavailable in the browser's default sans-serif font.

The legibility of different fonts generally depends more on the height of lower case letters than on the font size itself. Fonts like Verdana are much more legible than ones like "Times New Roman" and are therefore recommended for paragraph text.

Avoid problems with fonts and margins

My first rule is to avoid text at the body level that isn't wrapped in a block level element such as *p*. For instance:

```
<h2>Spring in Wiltshire</h2>

Blossom on the trees, bird song and the sound of lambs
bleating in the fields.
```

The text following the heading runs the risk on some browsers of being rendered with the wrong font and margins. You are therefore advised to enclose all such text in a paragraph, e.g.

```
<h2>Spring in Wiltshire</h2>

<p>Blossom on the trees, bird song and the sound of lambs
bleating in the fields.</p>
```

My second rule is to set the font family for *pre* elements, as some browsers forget to use a fixed pitch font when you set the font size or other properties for pre.

```
pre { font-family: monospace; }
```

My third rule is to set the font family on headings, p and ul elements if you intend to set borders or backgrounds on elements such as div. This is a work-around for a bug where the browser forgets to use the inherited font family, instead switching to the default font as set by the browser preferences.

```
h1,h2,h3,h4,h5,p,ul { font-family: sans-serif; }
```

Adding borders and backgrounds

You can easily add a border around a heading, list, paragraph or a group of these enclosed with a *div* element. For instance:

```
div.box { border: solid; border-width: thin; width: 100%; }
```

Note that without the "width" property some browsers will place the right margin too far to the right. This can then be used with markup such as:

```
<div class="box">
The content within this DIV element will be enclosed
in a box with a thin line around it.
</div>
```

There are a limited choice of border types: dotted, dashed, solid, double, groove, ridge, inset and outset. The border-width property sets the width. Its values include thin, medium and thick as well as a specified width e.g. 0.1em. The border-color property allows you to set the color.

A nice effect is to paint the background of the box with a solid color or with a tiled image. To do this you use the background property. You can fill the box enclosing a div as follows:

```
  div.color {
    background: rgb(204,204,255);
    padding: 0.5em;
    border: none;
  }
```

Without an explicit definition for border property some browsers will only paint the background color under each character. The padding property introduces some space between the edges of the colored region and the text it contains.

You can set different values for padding on the left, top, right and bottom sides with the padding-left, padding-top, padding-right and padding-bottom properties, e.g. `padding-left: 1em`.

Suppose you only want borders on some of the sides. You can control the border properties for each of the sides independently using the border-left, border-top, border-right and border-bottom family of properties together with the appropriate suffix: style, width or color, e.g.

```
  p.changed {
    padding-left: 0.2em;
    border-left: solid;
    border-right: none;
    border-top: none;
    border-bottom: none;
    border-left-width: thin;
    border-color: red;
  }
```

which sets a red border down the left hand side only of any paragraph with the class "changed".

Setting Colors

Some examples for setting colors appeared in earlier sections. Here is a reminder:

```
  body {
    color: black;
    background: white;
  }
  strong { color: red; }
```

This sets the default to black text on a white background, but renders strong elements in red. There are 16 standard color name, which are explained just below. You can also use decimal values for red, green and blue, where each value appears in the range 0 to 255, e.g. *rgb(255, 0, 0)* is the same as *red.* You can also used hex color values which start with the '#' characted followed by six hexadecimal digits. A two-way converter is included below which allows you to convert from RGB to hex color values.

Setting Link Colors

You can use CSS to set the color for hypertext links, with a different color for links that you have yet to follow, ones you have followed, and the active color for when the link is being clicked. You can even set the color for when the mouse pointer is hovering over the link.

```
  :link { color: rgb(0, 0, 153); }  /* for unvisited links */
  :visited { color: rgb(153, 0, 153); } /* for visited links */
  a:active { color: rgb(255, 0, 102); } /* when link is clicked */
  a:hover { color: rgb(0, 96, 255); } /* when mouse is over link */
```

Sometimes you may want to show hypertext links without them being underlined. You can do this by setting the *text-decoration* property to *none*, for example:

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

Which would suppress underlining for a link such as:

```
This is <a class="plain" href="what.html">not underlined</a>
```

Most people when they see underlined text on a Web page, will expect it to be part of a hypertext link. As a result, you are advised to leave underlining on for hypertext links. A similar argument applies to the link colors, most people will interpret underlined blue text as hypertext links. You are advised to leave link colors alone, except when the color of the background would otherwise make the text hard to read.

Color Blindness

When using color, remember that 5 to 10% of men have some form of color blindness. This can cause difficulties distinguishing between red and green, or between yellow and blue. In rare cases, there is an inability to perceive any colors at all. You are recommended to avoid foreground/background color combinations that would make the text hard to read for people with color blindness.

Named colors

The standard set of color names is: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow. These 16 colors are defined in HTML 3.2 and 4.01 and correspond to the basic VGA set on PCs. Most browsers accept a wider set of color names but use of these is not recommended.

Color names and sRGB values

⬛	black = "#000000"	🟩	green = "#008000"
⬜	silver = "#C0C0C0"	🟩	lime = "#00FF00"
⬛	gray = "#808080"	🟫	olive = "#808000"
⬜	white = "#FFFFFF"	🟨	yellow = "#FFFF00"
🟫	maroon = "#800000"	🟦	navy = "#000080"
🟥	red = "#FF0000"	🟦	blue = "#0000FF"
🟪	purple = "#800080"	🟦	teal = "#008080"
🟪	fuchsia = "#FF00FF"	🟦	aqua = "#00FFFF"

Thus, the color value "#800080" is the same as "purple".

Hexadecimal color values

Values like "#FF9999" represent colors as hexadecimal numbers for red, green and blue. The first two characters following the # give the number for red, the next two for green and the last two for blue. These numbers are always in the range 0 to 255 decimal. If you know the values in decimal, you can convert to hexadecimal using a calculator, like the one that comes as part of Microsoft Windows.

What about browsers that don't support CSS?

Older browsers, that is to say before Netscape 4.0 and Internet Explorer 4.0, either don't support

CSS at all or do so inconsistently. For these browsers you can still control the style by using HTML itself.

Setting the color and background

You can set the color using the BODY tag. The following example sets the background color to white and the text color to black:

```
<body bgcolor="white" text="black">
```

The BODY element should be placed before the visible content of the Web page, e.g. before the first heading. You can also control the color of hypertext links. There are three attributes for this:

- **link** for unvisited links
- **vlink** for visited links
- **alink** for the color used when you click the link

Here is an example that sets all three:

```
<body bgcolor="white" text="black"
  link="navy" vlink="maroon" alink="red">
```

You can also get the browser to tile the page background with an image using the background attribute to specify the Web address for the image, e.g.

```
<body bgcolor="white" background="texture.jpeg" text="black"
  link="navy" vlink="maroon" alink="red">
```

It is a good idea to specify a background color using the bgcolor attribute in case the browser is unable to render the image. You should check that the colors you have chosen don't cause legibility problems. As an extreme case consider the following:

```
<body bgcolor="black">
```

Most browsers will render text in black by default. The end result is that the page will be shown with black text on a black background! Lots of people suffer from one form of color blindness or another, for example olive green may appear brown to some people.

A separate problem appears when you try to print the Web page. Many browsers will ignore the background color, but will obey the text color. Setting the text to white will often result in a blank page when printed, so the following is not recommended:

```
<body bgcolor="black" text="white">
```

You can also use the bgcolor attribute on table cells, e.g.

```
<table border="0" cellpadding="5">
 <tr>
  <td bgcolor="yellow">colored table cell</td>
 </tr>
</table>
```

Tables can be used for a variety of layout effects and have been widely exploited for this. In the future this role is likely to be supplanted by style sheets, which make it practical to achieve precise layout with less effort.

Setting the font, its size and color

The FONT tag can be used to select the font, to set its size and the color. This example just sets the color:

```
This sentence has a <font color="yellow">word</font> in yellow.
```

The *face* attribute is used to set the font. It takes a list of fonts in preference order, e.g.

```
<font face="Garamond, Times New Roman">some text ...</font>
```

The *size* attribute can be used to select the font size as a number from 1 to 6. If you place a - or + sign before the number it is interpreted as a relative value. Use size="+1" when you want to use the next larger font size and size="-1" when you want to use the next smaller font size, e.g.

```
<font size="+1" color="maroon"
  face="Garamond, Times New Roman">some text ...</font>
```

There are a couple of things you should avoid: Don't choose color combinations that make text hard to read for people who are color blind. Don't use font to make regular text into headings, which should always be marked up using the h1 to h6 tags as appropriate to the importance of the heading.

Getting Further Information

For further information on CSS and tools that support it, you should look at the W3C home page for CSS. This includes pointers to books on HTML and CSS, for example, "Raggett on HTML 4", published 1998 by Addison Wesley. For a more detailed explanation of CSS, "Cascading Style Sheets" by Håkon Wium Lie and Bert Bos, pub. 2005 by Addison Wesley, which provides an in-depth look at CSS as seen by the architects of CSS themselves. Some errata are also available for this book.

I plan to extend this guide with additional pages explaining CSS positioning, printing and aural style sheets.

Best of luck and get writing!