CSS Formatting

Background

When HTML became popular among users who were not scientists, the limited formatting offered by the built-in tags was not enough for users who wanted a more artistic layout.

Netscape, whose browser was the industry leader at the time, created a number of extensions which eventually found their way into the HTML standards. The mostly relevant ones were:

body attributes

The body element was given a number of optional attributes which could affect the colour of the background and text.

The body element would look like this:

```
<body bgcolor="…" fgcolor="…" link="…" vlink="…" alink="…">
   …
</body>
```

These body attributes have now been deprecated, and should not be used in XHTML.

font tag

The font element was created as a container for other text. The font attributes would then describe the typeface, size and colour of the text.

The font tag would look like this:

```
<font face="…" size="…" color="…"> … </font>
```

The font tag has also been deprecated and should not be used in XHTML. (In fact it was already deprecated in HTML 4, but browser support for the alternative was not universally adequate at the time).

CSS Styles

Style Sheets

Style sheets allow you to specify styling information which affects all of the tags. So, for example, you can style all of the paragraphs, all of the headings etc.

Where are the Style Sheets?

Style sheets can be part of the HTML document, or in a separate document.

If they are in a separate document, they have the advantage of being available for all other HTML documents. To use an external style sheet, it will have to be linked with the HTML document. (This has nothing to do with the links we have been putting on web pages).

If a web page linked to an external style sheet also has its own styles, the local style information will override the external style information. For example, if the external style sheet indicates that all headings will be blue, but the HTML document style sheet indicates that they will be green, then they will be green. This allows the developer to customise individual pages.

Later, we will be using an external style sheet to allow us to use it for all of the pages.

Why use style sheets?

Style sheets allow you to make simple changes which affect the formatting of the entire page or even the whole site. With style sheets:

- You can apply the format easily to many different parts.
- Each part will have exactly the same format, instead of possible subtle differences.
- One single change will automatically be updated to the whole site.

Style sheets also allow you to implement some special effects you will have seen on other web sites, such as with links.

Embedded Style Sheets

Style sheets can be embedded in the head section of your page. They have the following form:

```
<head>
   <title> … </title>
   <style type="text/css">
      Style information here
   </style>
   …
</head>
```

The style element can appear anywhere in the head. You can even have multiple style elements if that suits you.

It is important to include the type attribute. It informs the browser that the data will be text, and that it is written using the CSS syntax. If you don't include the information, the browser will assume this anyway, but it is better to be in control.

Writing a Style

CSS styles have a language of their own. It is not difficult to learn, but there are many attributes.

The syntax for a style is:

```
tag {
   characteristic: value;
   ...
}
```

Note curly brackets (`{ ... }`), the colon after each characteristic (:) and the semicolon at the end (;). Technically, the semicolon is optional for the last attribute in the list, but it is always a good idea to include it anyway. This way you won't get caught out if you add another attribute afterwards.

The spacing for a style is unimportant. You could just as readily have written the following:

```
tag { characteristic: value; ... }
```

However, the first layout will be much more readable, especially if there are many characteristics for a tag. We will use this layout, generally with one characteristic per line.

Common Tags

Below are some possible settings for some common tags.

In general, the main tags you would change are:

`body` The appearance of the page.

You might set your main font here, and possibly your background colour or image.

`p` The appearance of paragraphs on the page.

`a` Links. :visited and :hover affect the links in particualr states.

Commonly, you might remove the underline, and change colour for the hover. You might also set the colour for visited links to the same as for unvisited.

`h1 ... h6` Headings

Here you can set font sizes and colours.

`td` Normal table cells.

Commonly you might set the alignment, especially the vertical alignment.

`th` Table heading cells.

Again you might set the alignment, especailly if you want to remove centering.

Sizes

Some of the attributes you wish to set may include the size of the text. This is written:

```
font-size: size;
```

The size may be expressed in a number of ways:

16pt This means 16 points (16/72 of an inch).

16px This means 16 pixels (dots).

150% This means 150% of the assumed size.

You can also measure in inches (`in`) centimetres (`cm`) and other units such as `ems`.

Whatever the unit you choose, do not put a space between the number and the unit. Spaces are interpreted as additional data, which, in this case, would be meaningless.

Colour

Colour can be defined for the text as well as the background. It is, unfortunately, spelt the American way, so you need to type it in carefully.

You can write your colours in a number of ways:

Using a colour name

You can choose from a number of common names. However, you should be aware that only 16 colour names are officially recognised:

> aqua, black, blue, fuchsia, gray (note American spelling), green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow

Using the RGB value

All colours on the screen are really a mixture of Red, Green and Blue (RGB). You can specify any of 16,777,216 colours by specifying the mix yourself.

Those new to RGB may find some of the mixtures non-intuitive. You can experiment if you like, but most graphics programs include colour mixers which will give you the RGB values of the colour you have chosen.

You can specify the RGB in one of two ways:

`rgb(…,…,…)`

This allows you to enter values between 0 and 255 for each of the Red, Green and Blue components of the number, in that order.

`#rrggbb`

You can enter the hexadecimal equivalents of the three values in one string. If you don't know what hexadecimal means, then you will probably use the former method.

The following examples are all equivalent:

```
color: pink;
color: rgb(255,192,203);
color: #ffc0cb;
```

More on Colour Names

The fact that over a hundred other names are recognised by most browsers is convenient. However there is no guarantee that the will be interpreted exactly the same way, or at all.

A good source of colour names is on the w3schools web site:

```
http://www.w3schools.com/css/css_colornames.asp
```

Common Colour Attributes

The two most common colour attributes are:

```
color: …;
```

This is, in fact, normally the colour of the text.

```
background-color: …;
```

This specifies the background colour for the element.

Fonts

You can specify the font for elements as follows:

```
font-family: …,…,…;
```

The font-family is the only CSS attribute which can take multiple values, separated by commas.

When choosing a font, you should always bear in mind that the font that the user sees will depend on the fonts installed on their system, not yours. Therefore, you should always specify alternatives, so that the browser isn't forced to choose a font you don't want.

The list of alternatives is comma-separated, and will always be searched from the beginning. If the browser cannot find the first is not available, then the browser will look further down the list.

Some fonts names have space. You must enclose these in double quotes ("…"). You can also do so for single-name fonts, but there is no need.

Generic Font Families

CSS defines certain generic font families:

`serif` These fonts have little strokes (serifs) on them.
The default serif font is something like "Times New Roman".

`sans-serif` These fonts are without (French: *sans*) serifs.
The default sans-serif font is something like "Helvetica" or "Arial".

`monospace` These fonts are all the same width, as they would be on a typewriter. They are sometimes used, as in these notes, to indicate code to be typed.
The default monospace font is something like "`Courier`".

There is also cursive and fantasy, which, by and large, should be avoided.

Linked Style Sheets

Although you can include CSS styles on the web page, it is usually better to have these styles in a separate document. This will allow you to use the same styles on several pages.

The advantages to having external style sheets include:

- Having your pages attached to the same style sheet will ensure consistent styles for all of your pages.
- A single change to the style sheet will be applied to all of your pages.
- It is better to separate the style information from the content of the page. This allows different developers to work on different areas without interfering with each other.

Using external style sheets also allows you, in some circumstances, to switch between alternative style sheets. This is useful, for example, to allow users with different needs to view your pages appropriately. The appendix contains one method on how you might accomplish this.

To use linked style sheets:

1. Create a separate document to contain the styles.

   It normally has the extension `.css` and is often called something like "`styles.css`".

2. Put all of the style information you would normally include between the `<style>` tags.

   Don't include the `<style>` tags themselves.

3. Put the following element in the head section of your page (somewhere after the title):

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

   The `link` element indicates that there is another document which is somehow related to this one. In this case the relationship is that of a stylesheet.

   The `type` attribute, as before, describes how the text is to be interpreted. The `href` locates the stylesheet to be used.

Cascading Styles

You can have more than on such link on your page. You can also have a style element. If you do, then the styles will cascade:

- Styles will be combined.
- If two styles involve the same attribute, then the most recently loaded style will override the former versions.

It is therefore very important to get the order of your stylesheets right.

Table Cells

You may have noticed that table data has a default behaviour regarding centring:

- All cells will be centre vertically.
- Head cells will also be centred horizontally.

To change centring, we use two types of alignment:

text-align

This is the horizontal alignment. The values can be left, center or right. Note the American spelling of center.

vertical-align

This is the vertical alignment. It can be top, middle or bottom.

We can fix both of these in one go with the following syntax:

```
tag, tag {
   …
}
```

If you have more than one tag, separated by commas, then the attributes will be applied to all of them. Don’t forget the comma: a tag list without commas implies a hierarchy of tags.

Link (Anchor) Styles

One popular change is to the link style, technically known as an anchor. For example you can remove the underline, change the colour, and change the colour again when the mouse is moved over the link.

The link tag is actually the `<a>` tag. However, we will be changing some of the states of the tag:

`a:link` This is the normal state of the link

`a:hover` This is when the mouse is above the link.

`a:active` This is when the mouse button is being pressed.

`a:visited` This is the link after it has already been used.

You can apply the attributes to the `<a>` tag. If you want to be more specific about one of the states, then you need to specify it as above.

Many designers use this to remove the underline of the links, make the visited link the same as unvisited links, and change appearance when the mouse is moved over the link.

Underlining

To remove the underline:

```
text-decoration: none;
```

To restore the underline:

```
text-decoration: underline;
```

Boldness

The boldness of an element is referred to as its weight. Although CSS supports many weights, most normal people only have two: bold and normal.

To make the element bold:

```
font-weight: bold;
```

To make the element normal:

```
font-weight: normal;
```

Image Borders

When using images inside an anchor, your will normally see a border around the image, corresponding to the underline you normally see underneath text.

To remove border, change border-width of the image to 0.

```
img {
   border-width: 0px;
}
```

Summary

Embedded Style Sheets

```
<head>
   <title> … </title>
   <style type="text/css">
      Style information here
   </style>
   …
</head>
```

Writing a Style

```
tag {  characteristic: value;
   ...
}
```

External Style Sheets

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

Common CSS Styles

Common Tags

`body` The appearance of the page.

You might set your main font here, and possibly your background colour or image.

`p` The appearance of paragraphs on the page.

`a` Links. :visited and :hover affect the links in particular states.

Commonly, you might remove the underline, and change colour for the hover. You might also set the colour for visited links to the same as for unvisited.

`h1 ... h6` Headings

Here you can set font sizes and colours.

`td` Normal table cells.

Commonly you might set the alignment, especially the vertical alignment.

`th` Table heading cells.

Again you might set the alignment, especially if you want to remove centring.

Fonts

```
font-size: size;
```

Sizes can be measured in various units. Commonly they are measured in points (pt) or pixels (px). Note that there must be no space between the size and the unit.

```
color: …;
background-color: …;
```

The colour can be specified by name or by RGB mix. For example:

```
color: pink;
color: rgb(255,192,203);
color: #ffc0cb;
```

```
font-family: …, …, …;
```

Specify a list of alternative fonts in preferred order (or only one font if you prefer). Font names with spaces must appear in quotes.

The last font in the list is normally one of the following:

```
font-family: … serif;
font-family: … sans-serif;
font-family: … monospace;
```

Styles

```
font-style: italic;
font-style: normal;
font-weight: bold;
font-weight: normal;
text-decoration: underline;
text-decoration: none;
```

Borders, Margins & Paddings

Borders

```
border-style: solid;
border-style: none;
border-width: [all];
border-width: [top-bottom] [right-left];
border-width: [top] [right] [bottom] [left];
```

Margins & Padding

```
margin: [all];
margin: [top-bottom] [right-left];
margin: [top] [right] [bottom] [left]
padding: [all];
padding: [top-bottom] [right-left];
padding: [top] [right] [bottom] [left]
```

Individual Values

The properties above specify all four border, margin and padding values for an element. If you want to specify a value for one side only (without affecting settings for other sides), you can specify it as follows:

```
...-top: ...;
...-right: ...;
...-bottom: ...;
...-left: ...;
```

Useful Table Styles

```
td, th {
      vertical-align: top;
      text-align: left;
   }
```

Anchor Styles

`a:link { … }`	The normal state of the link.
`a:hover { … }`	When the mouse is above the link.
`a:active { … }`	When the mouse button is being pressed.
`a:visited { … }`	After the link has already been used.

Images

```
… { background: url([imageName]); }
div { float: left|right }
```