Week 1 Exercises

The aim of this week's exercises is to gain some experience in creating a simply styled web page, displaying the page in a browser and managing printing of the page.

Time Required

These exercises should take you 1½ hours at the most, depending on your experience, and they are intended to be completed in class time.

Exercise 1: Hello World!

It is quite a tradition when learning a new programming language to start by programming 'Hello World!' and it would be a good way of demonstrating how CSS separates formatting and style from structure and content.

1. Start by creating a new HTML5 document in Dreamweaver and save it as `helloWorld.html` in your week one folder.

2. Insert a paragraph in the body of the document and type **`Hello World!`**

3. Save the document and view it in a browser.

Depending on the defaults that have been set for the browser, you should see the text of the paragraph in the top left hand corner of the viewport, showing in the default text size which is usually 16 pixels. Not very interesting at all!

4. Returning to Dreamweaver, create a new style sheet.

   Note that Dreamweaver has already inserted the at-rule which tells the browser that you will be using the UTF-8 character set. It has also inserted a comment that says this is a style sheet – this is not a requirement, just something that Dreamweaver does (in production I use this to start off giving the style a name and adding my copyright statement).

5. In the new stylesheet insert the paragraph selector:

   a. start the declaration by typing the opening curly bracket
   b. insert the property **`color`**
   c. set the value to `#00F`. My code looks like this:

```
@charset "utf-8";
/* CSS Document */

p {
   color:#00F;
}
```

Creating new documents in Dreamweaver

To create a new document either select Ctrl + N, or click on File in the application bar at the top of the Dreamweaver workspace and then select New. Choose whichever document type you require from the New Document dialogue that presents.

Spelling?

Note that HTML and CSS are programmed in American English and thus the colour property is spelt **`color`**. If you misspell the property it will be ignored by the browser.

6. Save the style sheet as world.css in your week one folder.

7. You now need to link the style sheet to the HTML document you created previously. Re-open **`helloWorld.html`** and insert a link to the stylesheet in the head of the document, thus:

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

8. Save both documents and view **`helloWorld.html`** in a browser.

   You should see that without changing the HTML (other than the link to the style sheet) the text is now showing in a blue colour.

9. This may be interesting but not particularly dramatic so reopen the style sheet and enlarge the font by adding to the paragraph declaration the property **`font-size`** setting the value to **`100px`**.

10. Save the style sheet and return to the browser; press the F5 key to update the display. That makes quite a difference, with the text much larger.

11. Now take control of the font that the display uses. Return to the style sheet and add a further property to the paragraph declaration;
    a. insert the property **`font-family`** and set the value to **`Verdana, Geneva, sans-serif;`**

    b. Save the style sheet and return to the browser.

       When you refresh the display you will notice the different font (assuming that the browser's default has not already been set to Verdana, Geneva or sans-serif).

Notice that you have changed the presentation on the screen by changing only the style sheet; the HTML has stayed the same throughout the exercise.

12. Now you will position the text differently by adding a margin to the paragraph declaration.
    a. Add the property **`margin`** and set the value to **`250px;`**
    b. save the style sheet but before you view the effect take a moment to think about what the impact will be.

       Now refresh the view in the browser – did it respond the way you expected?

Exercise 2: Decorating the text

Now you can add a bit of decoration to the text.

1. Return to the style sheet and add another property to the paragraph declaration,
    a. Insert `text-shadow` with a value of **`6px 6px 6px #FF6;`**
    b. Save the style sheet and take a look at the effect in Firefox, then take a look using Internet Explorer.

Firefox and Explorer should be displaying a yellow dropshadow. However, older versions of Explorer (IE9 or earlier) will be showing nothing new, i.e. no dropshadow.

This is one of those occasions where you would add a Microsoft proprietary extension for Explorer as it does not support all CSS properties. This is not required but best practice for users unable, or unwilling, to install newer versions of Explorer.

2. The property in this case is **`filter:`** (the colon is required) and the value is **`dropshadow(color=#FFFFFF66, offx=6, offy=6);`**

    a. Add this to the style sheet, in addition to the text-shadow
    b. save the style sheet and view the web page in an old version of Explorer if you can, you should see a blurred dropshadow with the text in both old and new versions of Explorer.
    c. [If Explorer sends up a warning about blocked content select 'allow' blocked content. The message is because the extension you have inserted is seen as an external program.].

Explaining the code

I am hoping that you are wondering what those metrics were in this exercise. In the text-shadow property the first value is the horizontal distance that the shadow is placed away from the text, e.g. 6 pixels to the right. The second value is the vertical shift of the shadow from the text, i.e. 6 pixels below. The third value is the degree of blur of the shadow, in this case 6 pixels. The final value is the colour of the dropshadow expressed in this case as the shorthand version of colour **`#FFFF66`** – yellow.

3. To understand the effects better try altering some of the values and looking at the results;
    a. I suggest that you first look at the effect of **`0 0 6px;`** You should see that this just creates a blurred shadow around the word, but with no offset.
    b. If you now try **`6px 0 0;`** you will see the effect of the offset to the right with no blur. This is not attractive but it is very clear in demonstrating the offset.
    c. Try doing the same with the vertical shift. Now try putting negative values in place for the horizontal and/or the vertical shifts – can you guess what the effect will be?

Take a look now at the Microsoft filter; can you see what the components of the value are? You should be able to deduce what the **`offx=`** and **`offy=`** are (offset on the x [horizontal] axis and offset on the y [vertical] axis), however the colour value is new. This is Microsoft's way of including the Alpha channel (transparency) value in the overall expression, i.e. the first two characters are the hexadecimal value of the alpha channel; the second two are the value of the red channel; the third of the green channel and finally the fourth is the value of the blue channel. There is no blur setting available with the Microsoft filter.

4. Restore the value of text shadow and save the stylesheet.

Exercise 3: Adding a twist

Continuing to explore how you can change the appearance of a web page without changing the HTML you will modify the presentation even further.

1. Re-open the style sheet if it is not already open
    a. In the paragraph declaration add a new property **`transform`** with the values **`scale(1) rotate(-15deg) translate(10px, 10px);`**
    b. When you save the style sheet and view the web page there is probably no difference because the browsers have not all caught up yet with the standard.

You can get the browsers to make the transformation by using the vendor specific prefixes for each of the main browsers. The full set for this property are:

```
-moz-transform: scale(1) rotate(-15deg) translate(10px, 10px);
-webkit-transform: scale(1) rotate(-15deg) translate(10px, 10px);
-o-transform: scale(1) rotate(-15deg) translate(10px, 10px);
-ms-transform: scale(1) rotate(-15deg);
```

2. Insert these into your style sheet ahead of the standard compliant property then view the result in Firefox.

You will find that the text has been rotated in an anti-clockwise direction from the end of the text. If you could view the result in Explorer 9 (IE9) you would find the rotation has worked but you now have a nasty black background to the text. This is one of those occasions where you would decide not to include the Explorer rotation and leave the text in its landscape format by simply excluding the **`-ms-transform`**

If you were able to view the result in Safari (the **`-webkit`** prefix) and Opera (the **`-o`** prefix) you would find that the transformation has worked perfectly there as it has in Firefox (the **`-moz`** prefix). Explorer V10 resolves this property as per the standard.

Explaining the code

Once again I am hoping that you are wondering about what the code is actually doing. Actually I suspect that you will be able to see that the scale is 100% of the original, i.e. not any bigger or smaller than its size should be. The rotate value is 15 degrees in the anticlockwise direction; this is defined by the negative value. The translate value moves the element along the x (horizontal) axis by the amount specified and along the y (vertical) axis by the amount specified after the comma; these values can be negative.

Exercise 4: Viewing the World

Having made some quite significant changes to the presentation of the Hello World! Text, we should now look at another effect that will transform how you view the page, but without changing the HTML.

1. Re-open the style sheet if it is not already open
    a. Between the row with the comment on and the paragraph declaration insert four new (blank) lines.
    b. On the second and third of these lines type the (empty) body declaration thus:

```
body {
}
```

> *All on one line!!*
>
> A convention I shall use throughout the course is the ↵ symbol.
>
> This denotes where the code on display has to be on multiple lines due to the restricted space available to display it in these printed documents.
>
> When typing the code in Dreamweaver continue the code along a single line, leaving out the ↵ symbols.

    c. Insert into the **`body{}`** declaration the **`background`** property followed by the value

```
#000 url(http://grin.hq.nasa.gov/IMAGES/MEDIUM/GPN-2000-001097.jpg)↵
center center fixed no-repeat;
```

    d. Next, add (inside the **`body{}`** declaration) the vendor specific **`-moz-background-size`** property and the value **`cover`**
    e. Finally add the standards compliant **`background-size`** property and the value **`cover`**
    f. Save the file and view the result – it may take a moment to appear.

Explaining the code

The values of this **`background`** property demonstrate compound shorthand. The first value **`#000`** is the value for the background-color property, in this case black. The second value **`url()`** is for the background-image property, and is calling an image that is on the server at NASA. The next value **`center center`** is for the background-position property; then comes the value **`fixed`** for the background-attachment property. Finally, the value **`no-repeat`** for the background-repeat property.

You also inserted a vendor specific prefix for the background size property before finally placing the standards compliant **`background-size`** property and the value **`cover`** This value is one of two possible, the other is **`contain`**

What is the difference? Using **`contain`** will scale the image, while preserving its intrinsic aspect ratio (if any), to the *largest* size such that both its width and its height can fit inside the background positioning area; **`cover`** will scale the image, while preserving its intrinsic aspect ratio (if any), to the *smallest* size such that both its width and its height can completely cover the background positioning area. So in practice **`cover`** always fills the browser window, whereas **`contain`** always fits the entire image within your viewport, leaving opaque borders on either the **`top-bottom`** or the **`left-right`** whenever the ratio of the background image and browser window are not the same.

Try changing the **`background-size`** property value to **`contain`** and saving the style sheet then see what the effect is in the browser. To see the effect magnified try resizing the browser using first one value and then the other.

A different view

This exercise might lead you to think that the image used for a background can only be a remote resource, but that is not the case. You can use a local image.

1. To see what I mean open the browser and navigate to **`http://www.headroomgate.co.uk/style/downloads/DSC_3393.jpg`** and when the image opens right-mouse click on it and choose the option to Save Image as… and save it to your stick in the same folder as your stylesheet.

2. Go back to your stylesheet and replace the URL of the NASA image with **`DSC_3393.jpg`** so that the body background declaration is:

   ```
   background: #000 url(DSC_3393.jpg) center center fixed no-repeat;
   ```

3. Save your stylesheet and then view **`helloWorld.html`** again in a browser and you should now see a background image of a seaside sunset which is being supplied by a local file and not across the Internet as it was with the NASA image.

Exercise 5: @media print

Now that you have Hello World! styled for the screen you should investigate what would happen if you were to print the page. You don't need to send the web page to the printer. With the webpage open you simply select Print Preview from the drop down list available from File in Firefox or the Tools cog wheel in Explorer.

You should see a reproduction of the webpage as it would appear on paper if sent it to the printer; this currently should include the background image – if it doesn't then that could mean that someone has turned off print background in the browser. If you need to restore background printing in Firefox first select Print Preview and when the preview window opens select the Page Setup… button and in the dialogue that opens select the Format & Options tab and check (select) the Options, Print Background (colors & images) check box and then click OK, the background image should re-appear.

If you need to restore background printing in Explorer first select Print Preview and when the preview window opens select the cog wheel icon. When the Page Setup dialogue opens check (select) the Paper Options, Print Background Colors & Images) check box and then click OK, the background image should re-appear.

Now that you know that the browser(s) is showing you what the web page would look like when printed with the background image in place you can re-open the style sheet and insert the at-rule to prevent it from printing.

1. In the style sheet locate the point where the body declaration has been completed and the paragraph declaration has not yet started.
    a. Insert four new (blank) lines
    b. On the second line type the **`@media print`** selector followed by an opening curly bracket
    c. On the third line type the closing curly bracket.
    d. Insert a new blank line between these two opening and closing lines
    e. Type the **`body`** selector with an opening curly bracket.
    f. Insert another new line
    g. Close the body selector with the appropriate curly bracket.
    h. Inside these brackets insert a new line with the **`background`** property and a value of **none**
    i. Save the style sheet and view the effect in a browser.

The display on screen should not have changed, however when you select Print Preview you should see no background image as the at-rule has told the browser that there should be no background. My style sheet code now looks like this:

```
@charset "utf-8";
/* CSS Document */

body {
	background: #000 url(http://grin.hq.nasa.gov/IMAGES/MEDIUM/GPN-2000-↵
001097.jpg) center center fixed no-repeat;
	-moz-background-size: cover;
	background-size: cover;
}

@media print {
	body {
		background:none;
	}
}

p {
	font-family: Verdana, Geneva, sans-serif;
	font-size:100px;
	color:#00F;
	margin:250px;
	text-shadow: 6px 6px 6px #FF6;
	filter: dropshadow(color=#FFFFFF66, offx=6, offy=6);
	-moz-transform: scale(1) rotate(-15deg) translate(10px, 10px);
	-webkit-transform: scale(1) rotate(-15deg) translate(10px, 10px);
	-o-transform: scale(1) rotate(-15deg) translate(10px, 10px);
	transform: scale(1) rotate(-15deg) translate(10px, 10px);
}
```

Explaining the code

The **`@media`** rule in this example is telling the browser that when the media of presentation is print then everything inside its declaration should override any previous declarations for the selectors. In this case you told the browser that when printing this page the **`background`** property of the **`body`** element should have a value of **none**

One further, and extremely important, point to note is that the rules declared in a style sheet and in the document itself are executed in the order in which they are encountered by the browser. Therefore (in this case) if a further rule for the body element was encountered later in the execution order then it would be that rule which prevailed.

The order of execution for styles is external style sheet first, next comes any embedded styles (in the head of the document) and finally any inline styles coded with the element on the page. You will cover this in more detail as you go through the course.