Week 2 Exercises

The aim of this week's exercises is to gain some experience in using various selector options to manage styling

Time Required

These exercises should take you 1½ - 2 hours, depending on your experience.

Exercise 1: Installing the sample web page

I have created an html file for you with a blank stylesheet linked to it.

1. Open a browser and locate the file at

   `http://www.headroomgate.co.uk/style/downloads/week2.zip`

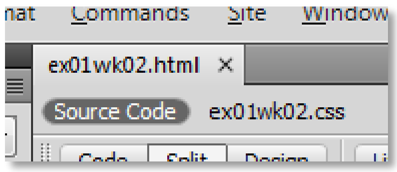
2. Download and unzip this file into your week 2 folder.

3. In the zip file you will find ex1wk2.html and ex1wk2.css.

4. Return to the browser and look at the html file and you will find a page with the heading 'Oxford, England'. The page is completely without any style declarations at the moment and therefore being rendered by the browser's default style. The rest of this evening's exercises will be involved with progressively adding style to the document in order to experience some of the possibilities offered by different style sheet selectors.

Exercise 2: Styling the Body and Headings

The first selector you will style is the **body** tag.



1. Open the html file in Dreamweaver and you will see that the stylesheet is listed in the Related Files Toolbar below the title of the main document.

2. You can access the stylesheet directly by clicking on the filename in the Related Files Toolbar.
   a. Try doing that now to open the stylesheet.
   b. When the stylesheet opens you will see that it is a 'blank' stylesheet presented by Dreamweaver as standard. It has the character set already declared as 'utf-8' and a comment that simply says CSS Document. The comment is of no value or consequence; you can leave it in or delete it. In production I usually put details of the version of the style sheet and also my copyright statements.

3. At the end of the stylesheet insert the selector **body** followed by the opening curly brace.
   a. As a matter of best practice I always hit return and put in the closing curly brace at this point too; this means that I know the rule is closed, even though I haven't made any declarations yet.
   b. Now insert the first declaration, you want the body text to be **`Arial`**, but you also need to declare some fall-back fonts too. I always include **`Helvetica`** and the default **`sans-serif`**.
   c. Next comes the overall font size of 0.9 em

d. Background colour of a pale green which is hex number CF9.
e. When you have completed this your body style should look like:

```
body {
      font-family:Arial, Helvetica, sans-serif;
      font-size:0.9em;
      background-color:#CF9;
}
```

4. Note the spelling of colour – it is in American English. Save the stylesheet by using Ctrl + s, or by going to File on the Application bar and selecting Save (**NB** do not use Saves As… because the file has already been created and named correctly).

5. Return to the browser and locate the file ex1wk2.html, or refresh it (F5) if it is still open. You should find that the presentation is in Arial and that it has a light green background. This is not a very interesting presentation but a slight improvement on the default styling.

6. Return to the stylesheet
   a. Add a rule set for the **`div`** with the ID of **`container`**
   b. Give it a width of 700pixels,
   c. A margin of 20px top and bottom and 'auto' left and right – you can use shorthand notation to enter those values once only.
   d. Add 0 padding top and bottom with 10 pixels left and right (use the shorthand again) and finally a background colour of #FFC.
   e. The rule set should be like this:

```
#container {
      width:700px;
      margin:20px auto;
      padding:0 10px;
      background-color:#FFC;
}
```

To explain about the margin; because the div is a block level element and has a width the auto margin left and right will cause the div to appear in the horizontal centre of the body tag in which it is positioned. Note that sadly this does not work in the vertical direction.

1. Save the stylesheet and view the page in the browser, after you have refreshed the view you should see the text sitting on a lightly coloured page in the centre of the screen.

2. Return to the stylesheet in Dreamweaver
   a. Style the three levels of headings in the document by a new style rule with multiple selectors: `h1, h2, h3`.
   b. Place this just after the body style, (by moving the container div down the sheet) then add the declaration that the headings are to be styled as "Trebuchet MS" (**NB** the quotes are required as there is a space in the font name, which is also required) with the fall-back fonts Arial, Helvetica and the default sans-serif. This results in the style rule:

```
h1, h2, h3 {
      font-family:"Trebuchet MS", Arial, Helvetica, sans-serif;
}
```

   c. Once again save the stylesheet then refresh the display of file ex1wk2.html in the browser and you will see that the headings are all now in Trebuchet MS.

3. These headings each need their sizing adjusted so you will need to declare a style rule for each one in addition to the rule set with the multiple selectors.
    a. Go ahead and do that and make the
        i. h1 a font-size of 1.5em,
        ii. h2 1.4em and the
        iii. h3 1.2em.

    You should end up with the following:

    ```
    h1 {
          font-size:1.6em;
    }

    h2 {
          font-size:1.4em;
    }

    h3 {
          font-size:1.2em;
    }
    ```

    b. Save the stylesheet again, refresh the display of file ex1wk2.html in the browser and you will see that the headings are showing as different sizes.

4. Continue by giving each level of heading its own distinctive colour along with the paragraph.
    a. Insert the colour #C96 into the h3 rule
    b. #C63 into the h2.
    c. Insert a paragraph rule and give it a colour of #630.
    d. Save the file and view the result in the browser.

5. The level 1 heading is to be treated very differently
    a. Give it a colour of #FFC and it will be hardly visible if you view it in the browser.
    b. Give the h1 a brown background colour; this is achieved by the declaration:

    ```
    background-color: #630;
    ```

    c. If you save the stylesheet and view the page the text will be cream on a brown background but the brown does not extend the full width of the white background because of the padding declaration in the container div.
    d. To overcome this you can add a margin declaration of zero top and bottom and -10 pixels left and right.
    e. In addition if you add a padding declaration of 6px top and bottom and 10px left and right the whole heading should look very much smarter.

    The level one heading style rule should now be:

    ```
    h1 {
          font-size:1.6em;
          background-color:#630;
          color:#FFC;
          padding:6px 10px;
          margin:0 -10px;
    }
    ```

That gives you the full styling for the main headings for now.

Exercise 3: Formatting the table

Turning now to the table;

1. Insert the table selector in your stylesheet just after the h1 rule you just completed, i.e. between the h1 and the container div.
   a. For now just declare the table as width of 100%.
   b. Save the stylesheet and view the webpage you will see that this has shuffled the content a little but overall has not had a great deal of visual impact.
   c. Create a rule for the multiple selectors `td` and `th` with a declaration of font size as 0.8em and align the text centrally. This rule looks like:

```
td, th {
      font-size:0.8em;
      text-align:center;
}
```

   d. Save the stylesheet and take a look at the page again.
   e. That should have made the table much clearer with space around the text and data items centred in each cell.
   f. Notice that the text alignment was achieved by a declaration of text-align center, this is because the **`td`** and **`th`** elements are in-line elements, whereas the container div is a block level element.

2. To continue styling the table give the table row (**`tr`**) a background colour of #930. You should be able to work out how to do that, just place the rule after the td, th you just created.

3. Save the stylesheet and take a look at the result in the browser. This result is rather ugly I think, but you can put some of that right straight away.
   a. To get rid of the white borders inside the table return to the table rule and add the declaration **`border-collapse:collapse;`**
   b. Whilst you are there set the colour of the text to cream.
   c. Save the stylesheet and view the result in the browser you should see that the borders have been collapsed and the text is now cream.

4. The table is looking better but you could still add more style to make the content easier to read. This is where you start to use pseudo-classes.
   a. First you should colour the even columns in the table with a background of #F60;
   b. To achieve this you need to use an nth-child pseudo-class for the `td`:

```
td:nth-child(even) {
      background-color:#F60;
}
```

5. This will do the trick, but you might be surprised by the result when you view it in the browser; only part of the column has been coloured. If you look for a moment at the HTML you will see that the second row is all **`th`** elements and not `td` elements as you might have expected. Each subsequent row has a single **`th`** element in it. The nth-child pseudo-class for the `td` has styled only the `td` elements on the even numbered columns as counted from the start of the table.

6. To colour the odd numbered columns you should use the same nth-child pseudo-class but with the attribute of odd and a background colour of #F90. When you view the result in the browser you will see that the first row and all the other odd numbered columns have now been coloured differently.

7. The table is looking better but the row containing the months looks a bit odd as it is the same colour as the left hand column. To change that you should use a different attribute for the pseudo-class .
   a. To target the second row in the table use an attribute of 2 for the nth-child pseudo-class for **`tr`** and give it a background colour of #630.
   b. That does it; that colours the second row.

8. Looking back at the table the title is still the same size and weight as the rest of the table which doesn't help with quickly understanding what the table is about.
   a. To style this you need to use another pseudo-class : first-of-class along with a descendent selector of td.
   b. You want to target the first row (first of type) and then style the data cell within it and reading from left to right in the selector you do just that, declaring a font size of 1.4ems, a weight of bold and padding of 6 pixels:

```
tr:first-of-type td {
      font-size:1.4em;
      font-weight:bold;
      padding:6px;
}
```

When you view the result you should now see a proper table title in place.

9. The final bit of styling for the table is the source credit on the final row of the table.
   a. To target this you use a variation of the pseudo-class you just used: last-of-type.
   b. You should style the td within the last row of the table and by now you should be able to see how to do that.
   c. Set the declaration to a font size of 0.8em with the style of italic aligned right with a padding of 4 pixels and coloured #FFC.

Exercise 4: Styling the blockquote

The blockquote is styled by default with just some ‘white’ space around it but no other styling.

1. To change that add the blockquote selector after those for the table and declare a solid, border of 1 pixel, with a colour of #603, and side padding of 10px:

```
blockquote {
	border: 1px solid #603;
	padding:0 10px;
}
```

2. This works on the whole blockquote, now you should target the level two heading inside it using another descendent selector.
   a. Declare the rule as:
      i. font variant of small caps
      ii. size 1.1em, cream in colour
      iii. background colour of mid-brown
      iv. margin of 0 top and bottom and -10px left and right
      v. padding of 6px and 10px.

   The rule looks like this:

```
blockquote h2 {
	font-variant:small-caps;
	font-size:1.1em;
	color:#FFC;
	background-color:#630;
	margin:0 -10px;
	padding:6px 10px;
}
```

This is fine in itself but the whole blockquote is still looking a little too strong given the content. Add one final descendent selector, the paragraph, and size it to 0.9em. When you view the result you should now see a small blockquote with a heading on a brown coloured background.

Exercise 5: Taming the trademark

Whilst you are working at the bottom of the page it would be a good time to style the trademark statement about Wikipedia.

a. Insert the trademark div selector after the container div
b. Give it a mid-brown background colour (#630)
c. text of the colour #FC6
   a. sized 0.8em
d. padding of 6 pixels 10 pixels
e. margin of 0 and -10px

Exercise 6: Moving the citation

Up at the top of the document you have the citation "*From Wikipedia, the free encyclopedia*".

1. You should move that into the dark brown background of the level one heading and colour it appropriately.
    a. To do this insert the cite selector in the stylesheet after the h3 rule
    b. Colour it #FFC.
    c. Make the font 0.83em
    d. Float it to the right
    e. Move the citation into the dark brown background by giving it a top margin of minus twenty pixels.

Exercise 7: More pseudo-classes

Probably the most common pseudo-classes are associated with the anchor tag; in this exercise you will style the anchor tag element without a pseudo-class and then add a pseudo-class for the 'hover' state. In addition you will set rules for how the anchor tag is styled when the document is sent to print.

Styling the anchor element

The default style for an anchor tag is blue and underlined which then changes to purple and underlined when the link has been visited. I always think this looks ugly and the BBC seem to share that view as they do not underline the anchor items, they remove the underline until the visitor hovers over the link and then it shows the underline.

1. To do this insert an anchor element selector in the stylesheet, in my case I have placed it after the blockquote rules.
    a. Set the text decoration to none
    b. Give it an orange colour of #F60
    c. After this rule insert another anchor tag selector with the pseudo-class of hover and declare that it should be underlined. These two rules look so:

```
a {
      text-decoration:none;
      color:#F60;
}

a:hover {
      text-decoration:underline;
}
```

Styling for print

Turning now to how links are dealt with when the page is printed.

2. For this you use an at-rule, @media with a descendent selector of print.
    a. Inside the at-rule you set rules for as many selectors as you need
    b. In this case you are going to make the colour of the anchor tag black
    c. Then you introduce an attribute selector combined with an after selector

The whole code stack is printed over the page but try coding it yourself and viewing the result in a browser before you look at my code.

My code stack looks like this:

```
@media print {
      a {
            color:#000;
      }

      a[href^="http://"]:after {
            content: " ["attr(href)"]";
      }
}
```

To explain: The at-rule @media allows you to select a set of rules to be implemented when the defined media is in use, in this case print. Inside the rule you place as many style rules as you need for the selected media. In this example you are targeting the anchor tag, first you set the colour of the tag.

The next rule is a little more complex, the attribute selector is placed within the square brackets immediately following the tag selector, in this case the ‘a’. The attribute selector in this case is targeting the href attribute of the anchor tag and the caret symbol means that the rule applies when the attribute href is equal to the characters that follow inside the double quotes, in this case `http://` . Concatenated with the attribute selector is the after selector `:after` . The result of the rule is then to include the value of the content declaration which in this case is to be the value of the href attribute which is placed inside square brackets when printed.

To see the result of this rule set place it in the stylesheet after the anchor rule and view the result in the browser. There will be no apparent change on the displayed page beyond the anchor styling you did previously. However if you select Print Preview in the browser you should see that the Wiley-Blackwell link has the content of the link displayed after it in square brackets, a very helpful feature when visitors print web pages containing useful links.

Exercise 8: Specific targeting with element + element selectors

This exercise shows how CSS can target certain selector combinations for styling, based on their relationship within the document. These can be combined with pseudo-classes as you will see.

1. Locate the cite rule and below it create a selector which combines h2 with the first line of the paragraph and declare it as displaying as dark red like this:

```
h2 + p::first-line {
      color:#900;
}
```

When you view the result you should see that the first line of every paragraph following a level 2 heading is now coloured red. This may not be something you want to do in reality but it does demonstrate the possibility.

2. To further demonstrate the possibilities try adding the cite, h2 and paragraph selectors together and
   a. Style the result as font family "Palatino Linotype", "Book Antiqua", Palatino, serif
   b. Font size of 1.1em
   c. Top margin of -10px.

This should make the first paragraph on the page move up from its current position, grow in size a little and become Palatino Linotype. Try coding this yourself before you turn the page for my code.

My code for this example is:

```
cite + h2 + p {
      font-family: "Palatino Linotype", "Book Antiqua", Palatino, serif;
      font-size:1.1em;
      margin-top:-10px;
}
```

3. One final rule combines three selectors, one with the first letter selector in place also. The syntax for the selectors is:

```
cite + h2 + p::first-letter
```

This says in effect select the first letter in the paragraphs that follow h2 headings which follow cite tags.

Try this out:

```
cite + h2 + p::first-letter {
      font-size:3.5em;
      font-weight:bold;
      float:left;
      margin-top:-4px;
      margin-bottom:-10px;
      padding-right:4px;
}
```

What you will get is a drop capital at the beginning of the first paragraph on the page of this site. The effect is somewhat imprecise, there are better ways of achieving a drop capital, but this does show the potential for combining selectors.