Table of Contents

Web Design & Development - Tutorial 04

CSS Positioning and Layout

During this tutorial, you will gain experience with layout and positioning using CSS and XHTML. You will learn about the different types of positioning available when rendering CSS and how to manipulate elements on a page. You shall understand the differences between fixed and liquid layouts. You shall also examine some of the common layouts used with web sites in operation today.

Conventions

> *This type of print indicates an exercise. They are usually present at the end of a major section of information*

```
This type of text indicates a code example / listing. You may wish
to type this code out or copy & paste.
```

Italic Text indicates code elements inline with a sentence. E.g.

The *<h1>* element.

What you need for this tutorial

- Access to the Internet (validation engine)
- A suitable web page / CSS editor such as Microsoft® Notepad® or Macromedia® Dreamweaver® MX 2004 / 8 (Refer to week 1 tutorial notes for Macromedia® Dreamweaver® MX 2004 assistance).
- A selection of web browsers for testing. E.g. Microsoft® Internet Explorer 6.x, Opera 7.54+, Mozilla Firefox 1.x or Safari
- An understanding of the topics presented in the previous tutorials. Please ensure that you are comfortable and grasp the concepts from all prior tutorials before proceeding or some material in this tutorial may be inaccessible

Common Terminology

You should be familiar with the following words/concepts/phrases:

- CSS, XHTML, Elements, Attributes and Mark-up
- Selectors, Declarations and Rules
- Block-level and inline elements

- Parent and child elements
- The box model
- CSS units of measurement

If you are not familiar with these topics, it is recommended that you review the tutorial material prior to this tutorial and/or discuss these topics with your tutor. For your convenience, the box model is reviewed on page 3 - Review the Box Model.

Layout with CSS

Traditionally, web designers and developers have used tables (with no border, padding or margin) to control layout. Invisible tables essentially create a grid structure for the page. This technique is in wide spread use still and some developers even promote the technique. You have already covered the basics of XHTML and the separation between presentation and meaning and might have realised already that the use of tables for anything other than the displaying of tabular structures is semantically incorrect. Layout is possible using CSS and allows the designers/developer to maintain strict XHTML which confirms to web standards and still achieve the desired visual effects.

Review the Box Model

You have seen how CSS implements the box-model (Figure 1) when formatting elements. Everything in XHTML is rendered as a box. You have also encountered inline and block-level elements and the differences between them (you might want to review those topics if you are not certain).

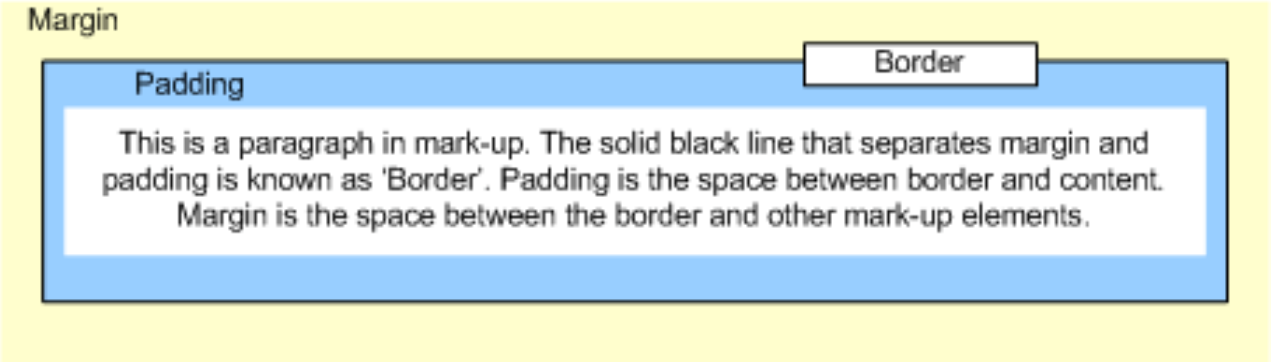

Margin
Border
Padding
This is a paragraph in mark-up. The solid black line that separates margin and padding is known as 'Border'. Padding is the space between border and content. Margin is the space between the border and other mark-up elements.

Figure 1 The Box Model

CSS enables you to position these boxes on a rendered page using several positioning schemes. One such scheme is 'normal flow' which is the one you have encountered up to now. This takes the elements in the order they appear in the mark-up and renders them appropriate as inline or block-level elements in the browser. The techniques shown in this tutorial explain how boxes can be displaced from normal flow under the control of the developer.

CSS Positioning

This section introduces three schemes for controlling the position of elements on a page. These are:

- Normal Flow – The default positioning scheme: http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_normal_flow.html
- Float – A box is taken out of normal flow and floated at the far edge of the window (the side is determined by CSS rule). http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_float.html
- Absolute – a box is removed from normal flow and placed according the co-ordinates specified in the CSS rules it has. http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_absolute.html

The main CSS property affect layout is the *position* property, the possible of which are listed in Table 1.

Table 1 Position property values (Ducket 2005)

Value	Description
static	A normal box laid out according to normal flow (because normal flow the default positioning scheme, you will rarely have to specify this value).
relative	The box is positioned according to the rules of normal flow and is then offset relative to the normal position (any subsequent boxes are calculated as if the box were not offset).
absolute	The box's position is determined using offsets from the top, left, bottom or right of the elements will be positioned within this containing box, there is no impact on the layout of sibling elements.
fixed	The box's position follows the same model as absolute positioning, but it will be fixed to some point within the page or screen, and will not move when the user scrolls the page. We shall not cover fixed as it is not supported well in popular browsers.

Exercise 1 – Investigate the CSS rules

Use the examples given in the section 'CSS Positioning' and examine the CSS files that are associated with the HTML documents. You should find links available to download the CSS file. Do the CSS rules make sense? Can you match the rules to the elements in the mark-up document? Explain your thinking and/or ask questions of your tutor regarding the relationships.

Normal Flow

Normal flow should be familiar to you as it is the default way that a browser renders XHTML mark-up. You need no styling rules to achieve normal flow. When a document uses normal flow, block-level elements are stacked vertically. You can separate boxes and their content using the margin and padding properties in CSS rules.

Relative Positioning

Relative positioning lays out a page according to normal flow, but then offsets any element whose *position* property has a value of *relative* from its position in normal flow according to the offset.

The offset is specified using four CSS properties: *top, bottom, right* and *left* (explanation in Table 2).

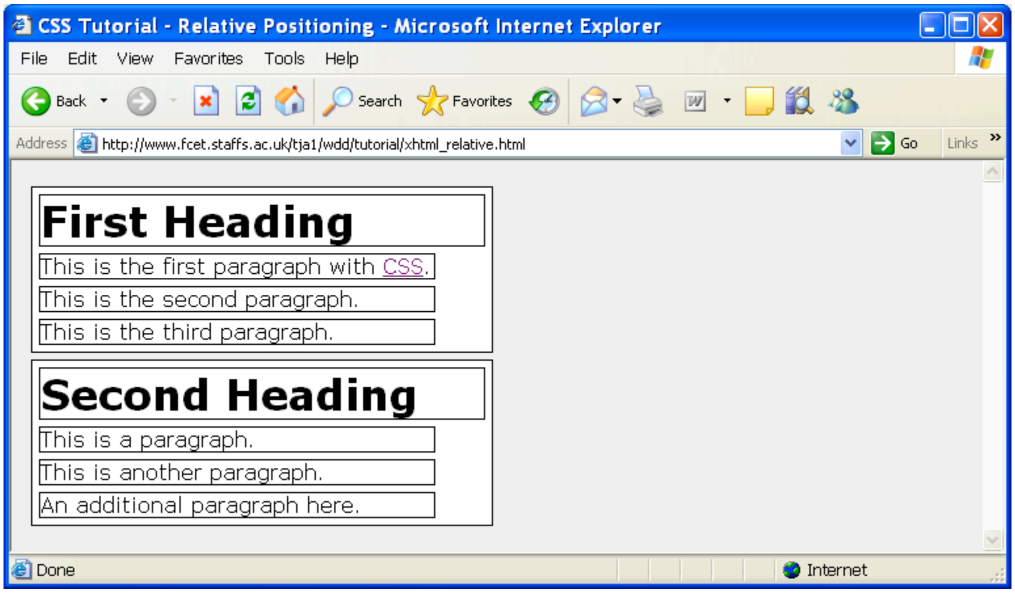
Table 2 Box Offset Properties (Ducket 2005)

Property	Purpose
Top	How far the top of a box is offset below the top of the containing block
Right	How far the right-hand side of the box is offset from the right edge of the containing block
Bottom	How far the bottom of a box is offset above the bottom of the containing block
Left	How far the left-hand side of the box is offset from the left edge of the containing block

Exercise 2 – Using relative positioning

Download the web page: http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_relative.html and accompanying CSS file (linked within the document – first paragraph). The document should look like Figure 2.

Examine the XHTML and ascertain the document structure. Examine the CSS and determine how the current style sheet is affecting the presentation of the document. Add styling rules to the second 'page' so the output resembles that of Figure 3. ***Hint:*** *It should only require three additional rules, use the tables above for assistance.*


CSS Tutorial - Relative Positioning - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_relative.html
Go Links
Done
Internet

First Heading

This is the first paragraph with CSS.

This is the second paragraph.

This is the third paragraph.

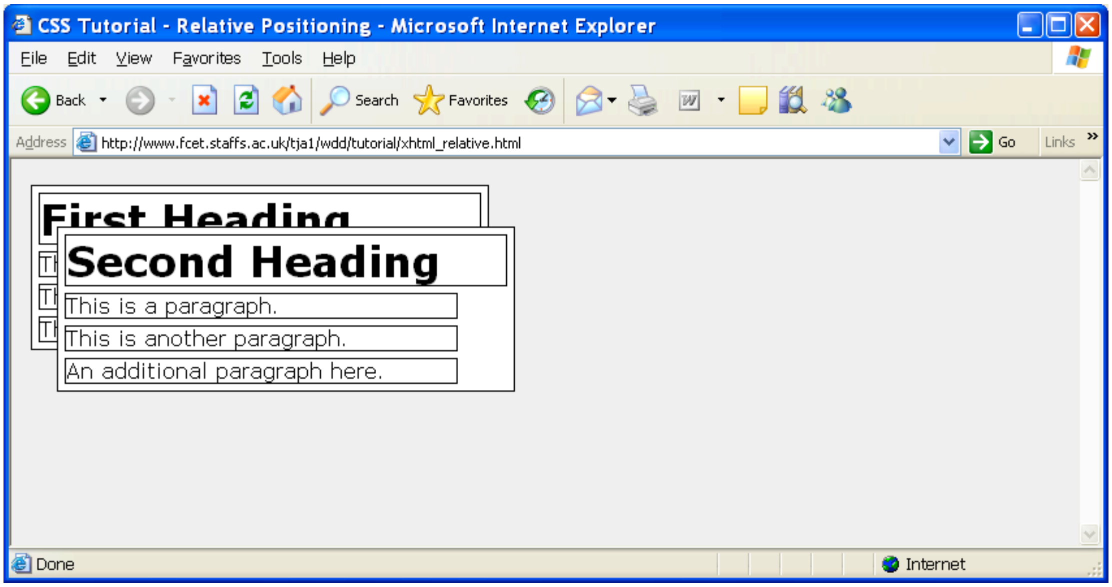
Second Heading

This is a paragraph.

This is another paragraph.

An additional paragraph here.

Figure 2 Sample File default view (Exercise 2)


CSS Tutorial - Relative Positioning - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
Go
Links
Done
Internet

Address: http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_relative.html

First Heading

Second Heading

This is a paragraph.

This is another paragraph.

An additional paragraph here.

Figure 3 Sample file desire outcome (Exercise 2)

Absolute Positioning

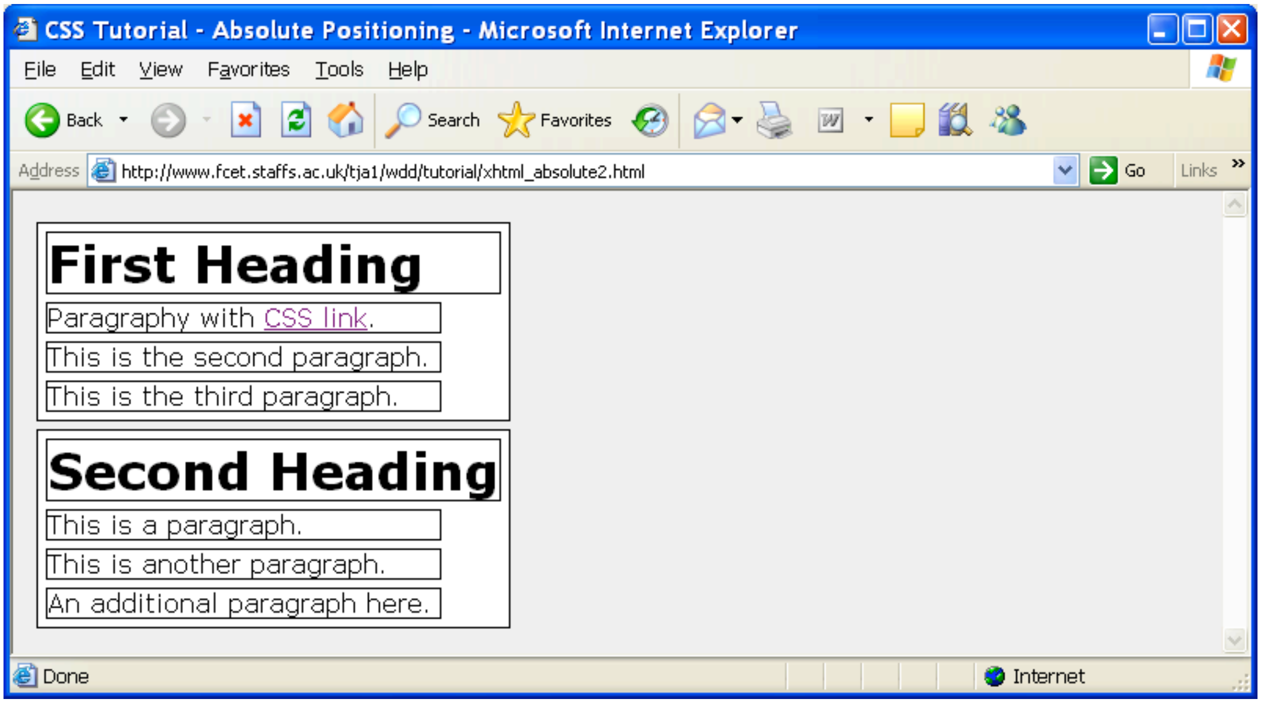
Absolute positioning enabled you to position a box using offsets from the top, right, bottom and left of the containing block (Ducket 2005). Absolutely positioned blocks are taken from normal flow and form a new containing block for any child elements.

Elements that have been positioned absolutely do not flow around other elements. It is possible to obscure other elements with an absolutely positioned element. To remedy this circumstance, the CSS property *z-order* is available. The *z-order* property can be assigned an integer (including negative numbers). The order of depth of an element is relative to those directly above and below it. An element with a *z-order* value of -1 will be behind an element with a *z-order* value of 0.

When specifying offsets and co-ordinates, the usual CSS units of measurements apply. It is common to specify layout boxes using pixels (*px*).

Exercise 3 – Using absolute positioning

Save the web page: http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_absolute2.html and save it to your personal storage area ensuring that you retain all the mark-up as shown in the source file. Using absolute positioning techniques, alter the file you downloaded from the start sample (Figure 4) so that it displays two columns, one page in each (Figure 5).


CSS Tutorial - Absolute Positioning - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_absolute2.html
Go Links
Done
Internet

First Heading

Paragraphy with CSS link.

This is the second paragraph.

This is the third paragraph.

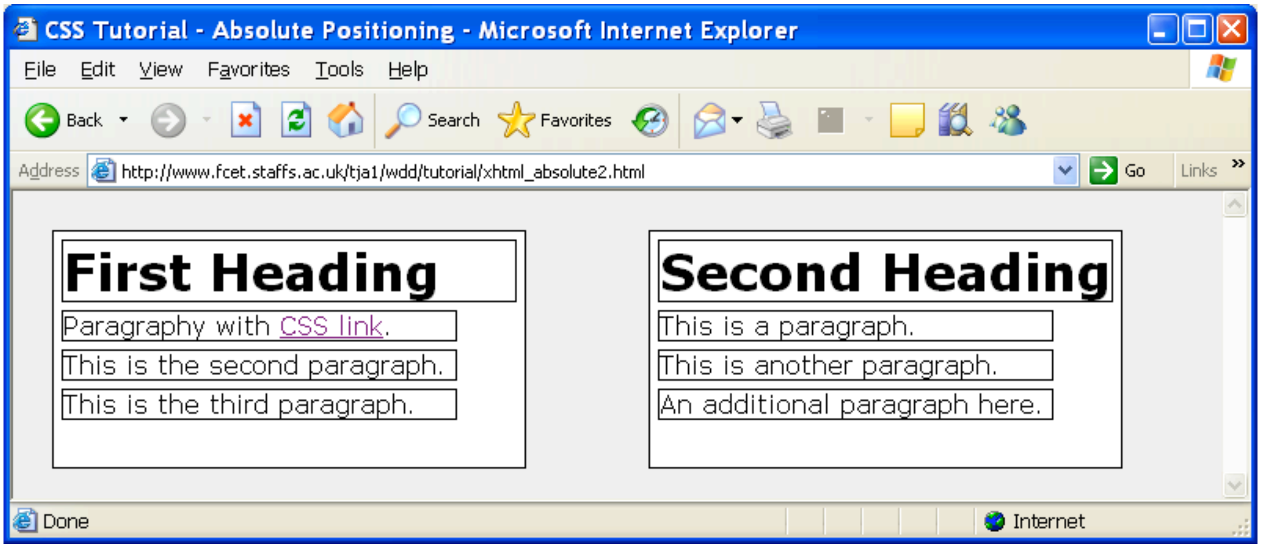
Second Heading

This is a paragraph.

This is another paragraph.

An additional paragraph here.

Figure 4 Starting page (exercise 3)


CSS Tutorial - Absolute Positioning - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_absolute2.html
Go Links
Done
Internet

First Heading

Paragraphy with CSS link.

This is the second paragraph.

This is the third paragraph.

Second Heading

This is a paragraph.

This is another paragraph.

An additional paragraph here.

Figure 5 Desired results for exercise 3

Further Selectors

This section will briefly introduce the concept of inherited element selectors, or child selectors. It is often necessary when performing layout operations with CSS to wrap multiple elements within a <*div*> element or inline elements within <*span*>. It is much more efficient to affect elements within a named region than it is to add class attributes to each and every element that needs to be altered. It is in these circumstances where child element selectors are used.

Child Selectors

To create child element selectors, simply insert white space (space, tab etc.) in between parent and child selectors. The code example in would create a style rule for all <*p*> elements inside a <*div*> with an *id* attribute of ‘content’.

```
div#content p {
       font-family:arial;
       font-size:0.9em;
       color:Navy;
}
```

Figure 6 Child selectors, <p> elements inside <div> element with ID = content

This technique can be used at multiple levels, for example, it is possible style all <*a*> elements that are contained within <*ul*>, that are in turn located within a <*div*> element with an *id* attribute of ‘navigation’ (Figure 7).

```
div#navigation ul a {
       /* style rules here */
}
```

Figure 7 Three depth child selector rule

It is also possible to specify many of these rules at once by separating them with commas. An example is shown in Figure 8.

```
div#content h1, div#pullpanel h2 {
       /* This rule will be applied to all <h1> elements inside
       <div id="content"> and all <h2> elements inside
       <div id="pullpanel"> */
}
```

Figure 8 Multiple child selectors separated with commas

Exercise 4 – Using Child Selectors

Download: http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/xhtml_child_selector.html and the accompanying CSS file (linked on the first content line of the document). Using child selectors and without modifying the XHTML mark-up, apply a style rule that effects only the <ul> element contained within the navigation block and not the other <ul> elements on the page. ***Hint:*** *You should try to use child selectors.*

Further Exercises

- Browse sample layouts located at: http://www.fcet.staffs.ac.uk/tja1/wdd/tutorial/layouts/ and download the ZIP resource file and expand it into a folder of your choice.
- Examine the sample layouts and try to ascertain how the style sheets are affecting the mark-up, try to explain how each section is created.
- Use these examples to create a five page web site using your own content / theme.
- It is possible to link more than one style sheet to a page. Try to create your site so the layout is contained within one file and presentation is stored in another.

References

Ducket, J. (2005) Accessible XHTML and CSS Web Sites: Problem-Design-Solution, Wrox Press (J. Wiley Publishing, Inc.), Indianapolis, IN, USA

Schafer, S.M. (2005) Web Standards, Wrox Press (J. Wiley Publishing, Inc), Indianapolis, IN, USA

Schmitt, C. *et al*. (2005) Professional CSS: Cascading Style Sheets for Web Design, Wrox Press (J. Wiley Publishing, Inc), Indianapolis, IN, USA

Acknowledgements

Portions of the tutorial exercises were taken from Duckett (2005)