**Infinite Skills Course**

**Learning CSS3**

**By**

**Geoff Blake**

**Introduction:**

CSS stands for cascading style sheets. It is a mechanism that provides the textual formatting (font, colors, sizes, etc...) and layout (positioning) control for the structural HTML that defines the content for web pages.

CSS is based on styles also called rules. declaration

Anatomy of a CSS rule:

p { font-family: verdana; }

selector property value

**Working with selectors:**

Class selectors start with a dot and can be used many times.

ID selectors start with a '#' and can only be used once on a web page.

Class selectors can be mixed with HTML selector types

Ex:

li.first { font-family: Verdana; }

such a class can only be used on or applied to list items on a web page.

When deciding or what type of selector to use, for example a class selector as opposed to an id selector. One should think about whether the rule that will be associated with that selector will be applied to just one element, or more than one. For one element rules, id selectors are ideal. For multi element rules (eg. You want the rule to apply to all lists on the page), class selectors are the choicest.

**Understanding the CSS box model:**

The CSS box model basically states that each block level element (an element that takes up its own 'line' on the page is surrounded by an invisible box. Three main properties of this box are of interest to us:

1. Padding (the space between the box border and its content, the inner space)
2. Border (the actual outline or edge of the box)
3. Margin (the space between the edge of the box and the edges of it's parent box)

Shorthand notation for boder:

Border: black solid 3px; (color, style, width) as opposed to defining attributes for each side of the border individually as:

border-top-color: blue;

border-top-style: solid;

border-top-width: 3px;

Difference between the different positioning modes:

1. Static: The static positioning attributes is the default attribute and confroms to the regular top down page flow layout scheme.

2. Absolute: This positioning causes an element to be placed relative to it's parent element or container. If there is no explicit parent to it, then the <body> container is assumed to be the parent because that is the natural parent of all the page elements which don't have an explicit parent. Also note that this positioning attribute may not behave as expected if the element to which it is being applied is nested in an element which doesn't also have the absolute positioning attribute because the parent is forced to follow the regular page flow due to not being declared absolutely positioned.

3. Relative: With this positioning attribute, like in the static case, the element is not pulled out of regular page flow. It is merely placed relative to where it normally would be.

4. Fixed: This is another positioning attribute that pulls the element to which it is applied out of the normal page flow. You have to declare at least on of top, left, bottom, right attributes for the element to appear and then it positions itself relative to the view port of the visible browser windows. It stays put even when the rest of the page scrolls 'behind' it.

Z-index

The z-index controls which element appears closer to the top (top meaning view point of user viewing the screen). For example if four rectangular divs are stacked on top of each other and we wish to bring 'up' the bottom most one to the top, we give it a z-index higher than any of the other three.

Overflow

Overflow controls how the content 'spills over' when it is bigger than it's parent container. The default value is 'visible' but content can appear to spill out of it's container. We can also user the 'hidden' value but that only clips the content. Of particular interest is the value 'scroll' which will provide a scroll bar but since it provide a vertical as well as a horizontal scrollbar (which is often no necessary), we are better off using the value 'auto' where in scroll bars are only provided when their need is detected in both the x and the y dimension.

Position using floating

When using floating, it is advisable to use explicit width values for the divs unless we are floating images.

Optimal web page width: Usually a width of 960 pixels is used as a safe bet for devices of different screen sizes.

Setting fonts: Setting a CSS font-family or font fallback or font stack is done so that a particular browser can show the first font that it finds in the list that it can display, if not it moves to the next one and so on. The idea is to specify the desired font first and then at the end of the list specify a generic fall back font. What's usually recommended is that we set a font family for our main body of all the paragraph elements(p) in the page and then set another font-family for our heading levels (h1, h2…).

Fonts are usually sized using pixels, percentages, points, and ems. This flexibility allows us some scalability when considering multiple devices of different screen sizes and browsers. Pixels and points are known as fixed or absolution measures or sizes. Ems and percents are relative or flexible units of measure.

The default font size of a browser is 12pt, which is 16 pixels. So these two will lead to the same font size. Using relative measures, if we use 100%, we are telling the browser to set the font size to one hundred percent of the default (which is most likely 12pt or 16 pixels). Using ems, works like percentages 1em = 100%. Relative font sizing works best when fonts on inner containers (paragraphs, headings, etc.) are set in ems or percentages and their parent containers (ex. body) set their font in a fixed measure, this way increasing the parent's font size automatically scales the children containers' fonts. This is illustrated in the diagram below.

Body font 13px;

P font 0.5em or 50%;

If we were to increase the body font from 13px to say 20px, since the paragraph element is inside and is using a relative font measure, it would scale it's font automatically.