Positioning (w3c)

The CSS positioning properties allow you to position an element a certain distance from either top, bottom, the left, or the right. It can also place one element behind another, and specify what should happen when an element's content is too big to fit.

Elements can be positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the positioning method.

There are four different positioning methods.
position: (absolute | fixed | relative | static | inherit) ;

Static Positioning

HTML elements are positioned *static* by default. A *static* positioned element is always positioned according to the normal flow of the page.

Static positioned elements are not affected by the top, bottom, left, and right properties.

Fixed Positioning

An element with *fixed* position is positioned relative to the browser window. It will not move even if the window is scrolled. The position can be specified using one or more of the properties top, right, bottom, and left.

```
p.pos_fixed    {
position:fixed;
top:30px;
right:5px;      }
```

Note: IE7 and IE8 support the fixed value only if a !DOCTYPE is specified.

Fixed positioned elements are removed from the normal flow. The document and other elements behave like the fixed positioned element does not exist.

Fixed positioned elements can overlap other elements.

Relative Positioning

A *relative* positioned element is positioned relative to its normal position.

```
h2.pos_left    {
position:relative;
left:-20px;      }
```

```
h2.pos_right   {
position:relative;
left:20px;       }
```

The content of *relative* positioned elements can be moved and overlap other elements, but the reserved space for the element is still preserved in the normal flow.

```
h2.pos_top     {
position:relative;
top:-50px;       }
```

Relative positioned elements are often used as container blocks for absolutely positioned elements. This value cannot be used for table cells, columns, column groups, rows, row groups, or captions.

Absolute Positioning

An *absolute* position element is positioned relative to the first parent element that has a position other than static. If no such element is found, the containing block is <html>.
The position can be specified using one or more of the properties top, right, bottom, and left.

```
h2             {
position:absolute;
left:100px;
top:150px;      }
```

Absolute positioned elements are removed from the normal flow. The document and other elements behave like the absolutely positioned element does not exist.

Absolute positioned elements can overlap other elements.

Overlapping Elements
When elements are positioned outside the normal flow, they can overlap other elements.

Z-Index Positioning
The *z-index* property specifies the stack order of an element (which element should be placed in front of, or behind, the others).

An element can have a positive or negative stack order:

```
img             {
position:absolute;
left:0px;
top:0px;
z-index:-1      }
```

An element with greater stack order is always in front of an element with a lower stack order.

Note: If two positioned elements overlap, without a *z-index* specified, the element positioned last in the HTML code will be shown on top.

Background Positioning Properties
Setting which image
background-image: url(imageName.jpg);

Should it not fill the space with copies?
background-repeat: no-repeat;
or repeat-x (make row) or repeat-y (column)

Where should it be generally?
background-position: center;
or left center, center center, right center or center bottom, center top

Where should it be specifically to the container?
background-position: 20px 50px;
(measured from the top and from the left)

Where should it be relatively to the page?
background-position: 10% 50%;
(% from the top and from the left)

Where should it stay visible even if page is scrolled?
background-attachment: fixed;

How can you make an element disappear?
visibility: hidden;
(or **visible**; or **inherit**;)

Reading: McFarland – Ch 13
How does absolute positioning work? Show an example.

How does relative positioning work? Show an example.

How does fixed positioning work? Show an example.

What is the real difference between fixed and absolute?

Which positioning method leaves a hole in the regular flow of the page?

How does the z-index property work? Show an example.

Is the element with the higher number in front of or behind the element with a lower number?

How does the visibility property work? Show an example.