Notes - Class 2

****

SEO

Writing semantically descriptive markup and having relevant, well-written and concise content and file structure really is the best SEO. But Google's SEO starter guide is definitely worth a read, as well as their Webmaster Guidelines on the subject. They are, after all, who you're trying to impress.
try to use correct html elements, ie correct markup to describe your content appropriately, this helps search engines to understand the type of content you are including on your site and rank you for specific searches.
its hard to fool search engines, so don't try, for example, wrapping a lot of content in heading tags (<h1>, <h2>...), as web crawlers will pick up on this and this will have a negative impact

We covered <header> <article> <section> and <footer>, and how they're all preferable to using plain <div> elements; but make sure you're using them correctly. Avoid common mistakes; HTML5Doctor.com has a fairly comprehensive post about the pitfalls.

very important to optimise image names for SEO:
<img src="images/logos/radiohead.jpg" alt="Radiohead logo">

always use "alt" attribute, but if you can't think of anything meaningful and relelvant to put into "alt" attribute this probably means that image is not contextual and this image should probably be a background image

*

<meta> tags are used to describe the information on your page, as well as Facebook's OpenGraph meta tags for making social sharing of your site prettier. There's more on OpenGraph on the Facebook Developer site. Also remember that you can test what information Facebook can gather from your web site using the Facebook URL Debugger, when your site is online.

<meta> tags are necessary, however they don't guarantee the high ranking by google.

*
Open graph tags (or other social sites' tags)

It is possible to choose what goes into a snippet about your page if the link is posted on social sites. one of the examples
<meta content="http://generalassemb.ly" property="**og**:url"> - this would display the indicated url to the site when you post something from the pages of that site on fb
twitter has their tags too, as well as some other sites.

*
Microformats
Have a read of Google's intro to microformats if this interests you. Be warned; the documentation is unnecessarily complicated.

*
Sitemaps
Google has a thorough microsite on creating sitemaps **(the maps of how Google crawls and displays your pages) on their Webmaster tools site.**

*
a tip re page titles (this is one of the practices, which you may or may not adapt):
when giving titles to pages of your site (other than you home page) use the page context as a first word for your title rather than site's name:
not: "GA - About",
but instead "About - GA"
this is to ensure that if user has a lot of tabs open (the tab has a little space to display the page title) and while browsing your site can still tell which pages they are visiting.

*
File structure
for your file names it's best to keep them lowercase, avoid spaces, make use of hyphens (There's a debate as to whether Google punishes you if you use underscores for file names)

*
For navigation try using text as much as possible, a lot of images/iconry, some commonly used, don't actually make real sense and therefore can be non-user-friendly;
additionally, don't use text which was stored as image, this prevents web crawlers from reading the text contained on those images and you can use images as a background for your navigation, there's also an image replacement technique, which you will cover later in a course

*

Another SEO-friendly practice for links:

"title" attribute for links is what browser will display on hovering that link over, below is an example of how the markup would look:

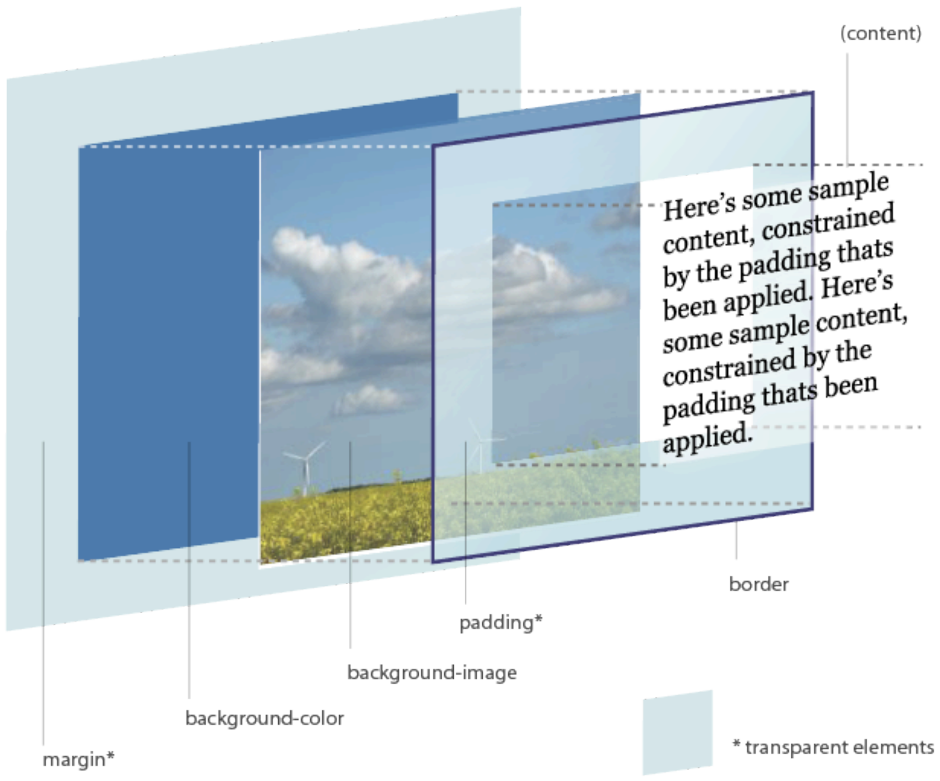
<a href="blah.com" title="Blah HQ Website">click here</a>

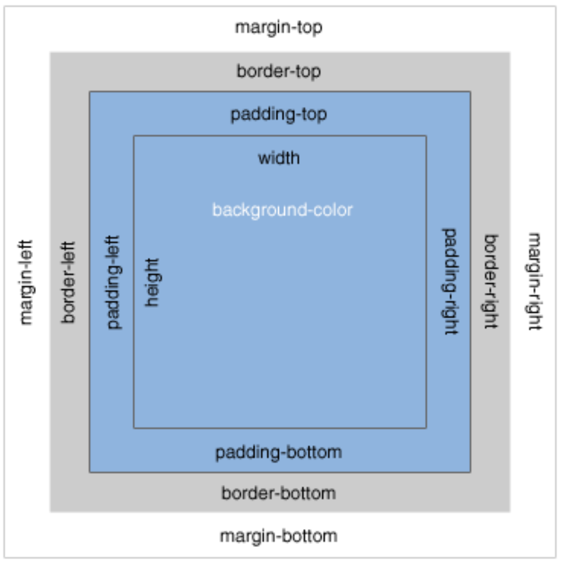
"click here" is what the hypertext will be and if you hover over it "Blah HQ Website" should display just above, and if you click you will be taken to blah.com. A lot of developers avoid using "click here", however some research does show that clickability is higher if you do.

****

The CSS Box Model

How CSS applies width, height, padding, borders and margin to shape and size elements. Here are a couple of diagrams illustrating those effects:


(content)
Here's some sample content, constrained by the padding thats been applied. Here's some sample content, constrained by the padding thats been applied.
border
padding*
background-image
background-color
margin*
* transparent elements



http://www.hicksdesign.co.uk/boxmodel/ - gives a good visualisation and some description of how padding, border and margin work together and the http://www.nicksdesign.co.uk/boxmodel/ - gives a good visualisation and some description of how padding, border and margin work together and the order in which elements are stacked.

Also take a look at the boxmodel-completed.zip file in among the files you would have downloaded previously

****

CSS Floats and Clearing

Floats and clearing are used to position elements of page layout; they're fairly unusual concepts outside of CSS development, so getting your head into some further reading will help to reinforce what we covered. There's the CSS Tricks floats primer, which covers all the basics very well, and then there's Smashing Magazine's piece on float theory, which is a little more in depth, but it's also six years old. Some of the 'bugs' mentioned are irrelevant now, but, at the time, it was **awful**.

There are some great visual examples of float combinations in this Design Shack article, too. Remember that if you float an element or elements, then somewhere along the line, you're almost certainly going to want to clear it. If you have an element in your layout already that you can use, give that a clear: property in CSS. If not, use the clearfix CSS (either as a .clearfix class or as part of the parent of the floats' styling).

Float is a CSS positioning property. To understand its purpose and origin, we can look to print design. In a print layout, images may be set into the page such that text wraps around them as needed. This is commonly and appropriately called "text wrap".

In print page-layout programs, the boxes that hold the text can be told to honor the text wrap, or to ignore it. Ignoring the text wrap will allow the words to flow right over the image like it wasn't even there. This is the difference between that image being part of the flow of the page (or not). Web design is very similar.

In web design, page elements with the CSS float property applied to them are just like the images in the print layout where the text flows around them. Floated elements remain a part of the flow of the web page. This is distinctly different than page elements that use absolute positioning. Absolutely positioned page elements are **removed** from the flow of the webpage, like when the text box in the print layout was told to ignore the page wrap. Absolutely positioned page elements will not affect the position of other elements and other elements will not affect them, whether they touch each other or not.

this is how the property is usually applied:

```
img {

    float: left;

}
```

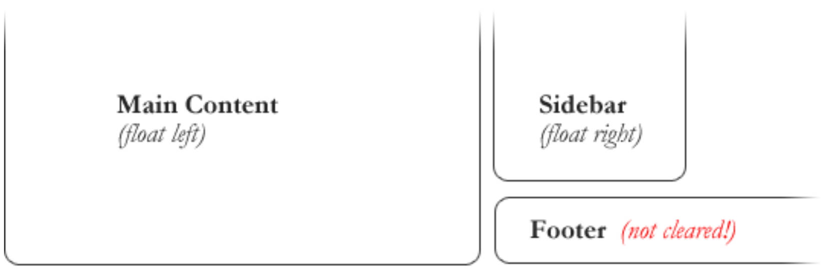
There are four valid values for the float property. **Left** and **Right** float elements those directions respectively. **None** (the default) ensures the element will not float and **Inherit** which will assume the float value from that elements parent element.

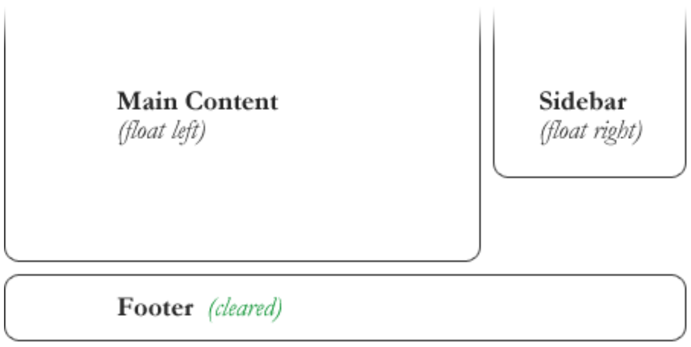
Aside from the simple example of wrapping text around images, floats can be used to create **entire web layouts**.

Clearing

Float's sister property is clear. An element that has the clear property set on it will not move up adjacent to the float like the float desires, but will move itself down past the float.

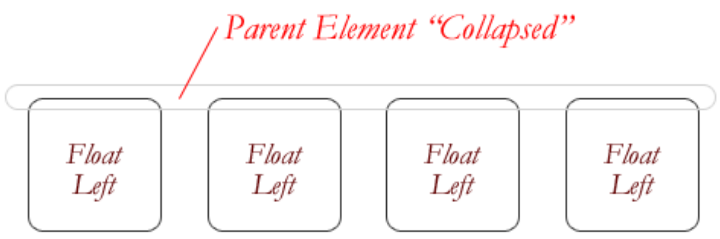
not cleared (first image below) vs cleared (second image below)





If you happen to have an unfloated element that follows your floated element then you can just use CSS "clear" property on that element (this is used in "Floating text" exercise)

One of the strange things to get to terms with about working with floats is how they can affect the element that contains them (their "parent" element). If this parent element contained nothing but floated elements, the height of it would literally collapse to nothing. This isn't always obvious if the parent doesn't contain any visually noticeable background, but it is important to be aware of.



Techniques for Clearing Floats

If you are in a situation where you always know what the succeeding element is going to be, you can apply the clear: both; value to that element and go about your business. This is ideal as it requires no fancy hacks and no additional elements making it perfectly semantic. Of course things don't typically work out that way and we need to have more float-clearing tools in our toolbox.

- **The Empty Div Method** is, quite literally, an empty div. <div style="clear: both;"></div>. Sometimes you'll see a <br /> element or some other random element used, but div is the most common because it has no brower default styling, doesn't have any special function, and is unlikely to be generically styled with CSS. This method is scorned by semantic purists since its presence has no contexual meaning at all to the page and is there purely for presentation. Of course in the strictest sense they are right, but it gets the job done right and doesn't hurt anybody.
- **The Overflow Method** relies on setting the overflow CSS property on a parent element. If this property is set to auto or hidden on the parent element, the parent will expand to contain the floats, effectively clearing it for succeeding elements. This method can be beautifully semantic as it may not require an additional elements. However if you find yourself adding a new div just to apply this, it is equally as unsemantic as the empty div method and less adaptable. Also bear in mind that the overflow property isn't specifically for clearing floats. Be careful not to hide content or trigger unwanted scrollbars.
- **Clearfix** or Easy Clearing Method uses a clever CSS pseudo selector (:after) to clear floats. Rather than setting the overflow on the parent, you apply an additional class "clearfix" to the parent of the floated elements in your HTML.

Description of what clearfix hack does:

```
.clearfix:after {              /*perform the following after element with class of "clearfix" */

    content: ".";              /* create content of a dot at the end of the element with class of "clearfix" (fake element in css) */

    display: block;            /* display it block */

    clear: both;               /* CSS property "clear" with value "both", ie clear on both left and right

    visibility: hidden;        /* to hide the dot */

    line-height: 0;            /* so that to line containing the dot has no height */

    height: 0;                 /* that created element (the dot) has no height */
```

****

CSS TECHNIQUES USED IN THE CLASS:

BOX-MODEL hack:
This was explained briefly, if you want to read up here's a good link:

This was explained briefly, if you want to read up here's a good link:

http://tantek.com/CSS/Examples/boxmodelhack.html

Essentially it overrides Box model which is described above

***

Clearfix:

refer to above section on Clearing

***

CSS image replacement is a technique of replacing a **text element** (usually a header tag) with an **image**. An example of this would be including a logo on a page. You may want to use a <h1> tag and text for this for the accessibility and SEO benefits, but ideally you'd like to show your logo, not text.

```
.your-class-name {
  text-indent: 100%;   /*kicks off the text off its usual place by 100% of its own width */
  white-space: nowrap; /* keep the text as one line */
  overflow: hidden; /* doesn't show any of that overflown text */
}
```

you can read a bit about it here: http://www.zeldman.com/2012/03/01/replacing-the-9999px-hack-new-image-replacement/

****

MISCELLANEOUS:

***

Example on use of background property in CSS:

```
background-color: #d6d6d6;
background-image: url(../images/background.jpg);
background-repeat: repeat-x; background-position: 0 0;
 /* ...OR... */
background: #d6d6d6 url(../images/background.jpg) repeat-x 0 0;  /* here you are combining the above 3 properties into one line of code
*/
```

there's another CSS property you can use in CSS3 and its a background-size

***

Useful:

image by default is an inline element and CSS property text-align can be used to align it horizontally, this would work for images placed via html (doesn't work for background images) once you are floating an image or displaying it as "block" then text-align will no longer work

***

Sublime shortcuts:

https://gist.github.com/lucasfais/1207002