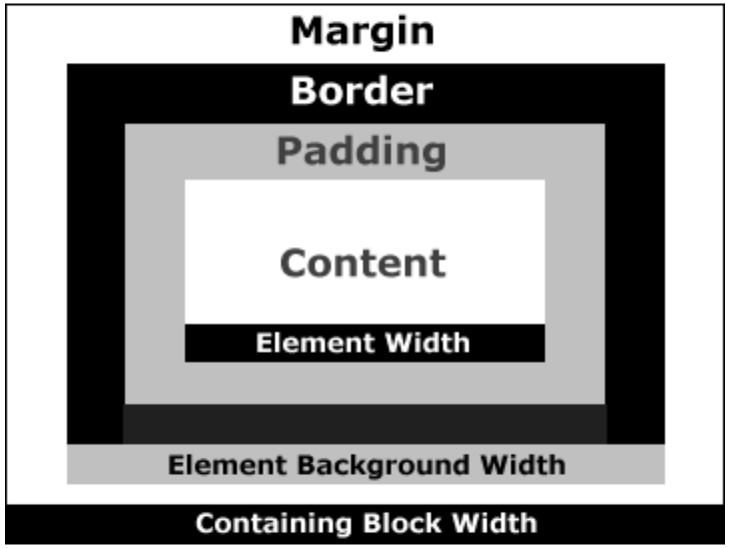
IWDD100

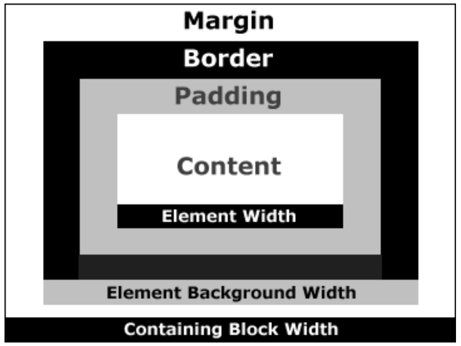
Lesson 3
CSS Layout

The Box Model

- In order to layout pages properly with modern CSS, you'll need to understand the box model!



Margin and Padding



```
margin: 20px;
```

Put a 20 pixel margin around the element or on each side of the element if the element is inline

```
padding: 50px;
```

Put 50 pixels of padding around the element or on each side if the element is inline

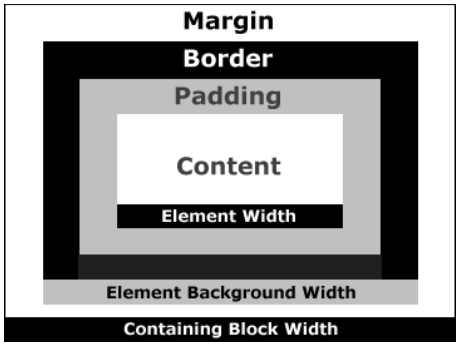
```
padding-right: 10px; margin-left: 40px;
```

Append -right, -left, -top, -bottom to be more specific

```
padding: 20px 10px; margin: 100px 30px;
```

Shorthand to specific padding on top and bottom then padding on left and right

Margin and Padding



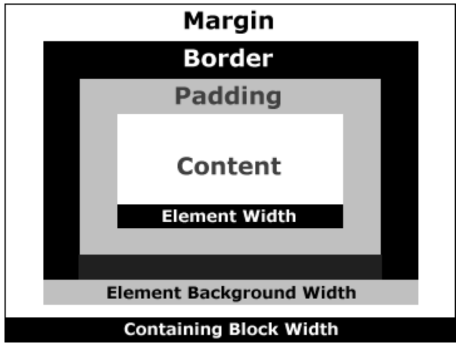
```
width: 900px;
margin: 0 auto;
```

- A popular “trick” to create a website with a centered block of content and a gutter on either side of the content.

- Generally, these attributes are set on a div with the id `wrapper`

- **Setting the left and right margin to auto will only work if a width is set**

Border



`border-style: dashed;`

Gives the border a dashed style. Other possible values include `dotted`, `solid`, `double`, `groove`, `ridge`, `inset`, and `outset`

`border-width: 10px;`

The border should be 10px wide

`border-color: #111111;`

Gives the border a gray-ish color specified using a hex code

`border: 1px solid red;`

Puts a 1 pixel solid red border around an element, shorthand for all of the above

`inline` vs `block`

`display: block;`

The element is displayed as a block, just like a <p> tag is, tolerates no elements aligned next to it unless float is used

`display: inline;`

The element is displayed inline, inside the current block on the same line as other elements it is near

`display: inline-block;`

A combination of the two which allows a block to be next to another block, but still have vertical padding and margins

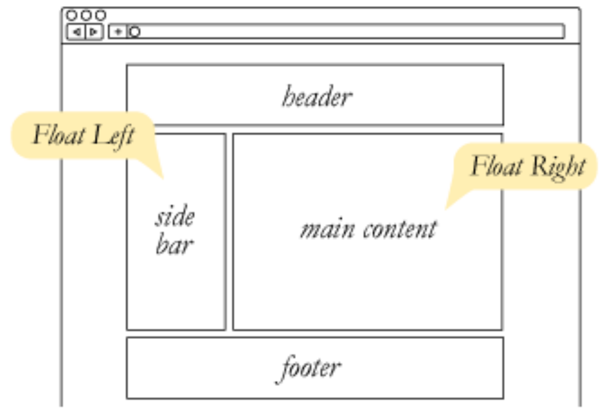
`float`

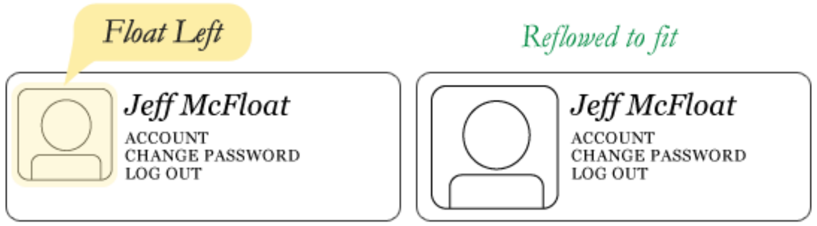
`float: left;`

Move an element all the way to the left side of its container. Have all other elements flow around it.

`float: right;`

Move an element all the way to the right side of its container. Have all other elements flow around it.





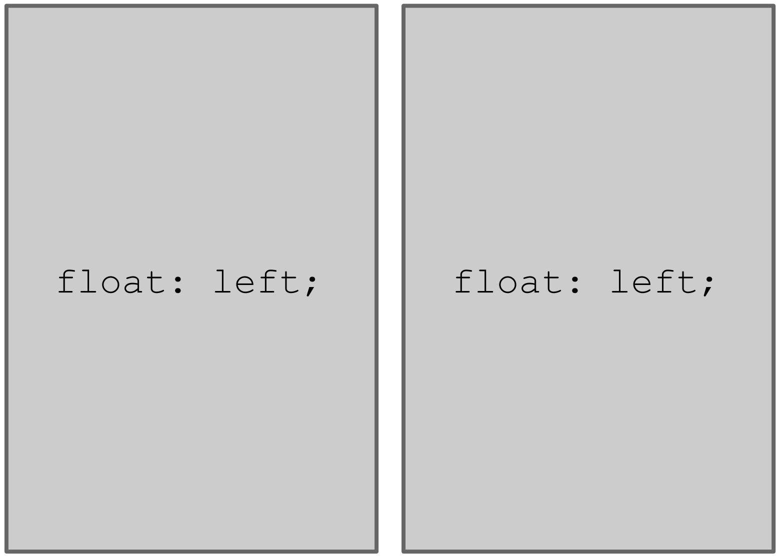
`clear: both;`

Remove the effects of a floated container, applied to an element after a floated container to clear the effects of using `float`

img sources at css-tricks

Using `float` for page layout

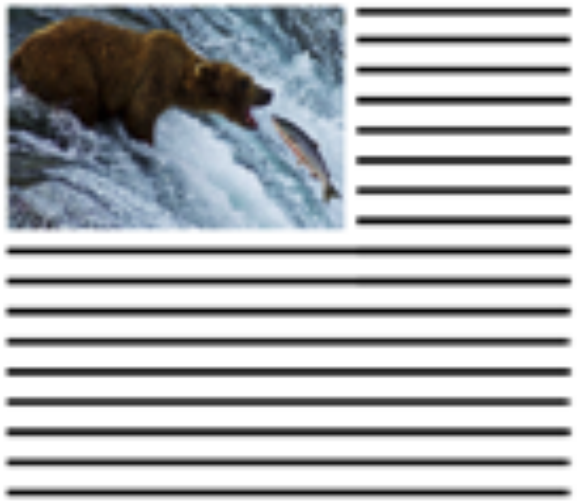
- To create a two-column page layout, float the first and second column left
- Try resizing the page to a smaller width - the second column will collapse under the first (left) column



Using `float` for page layout

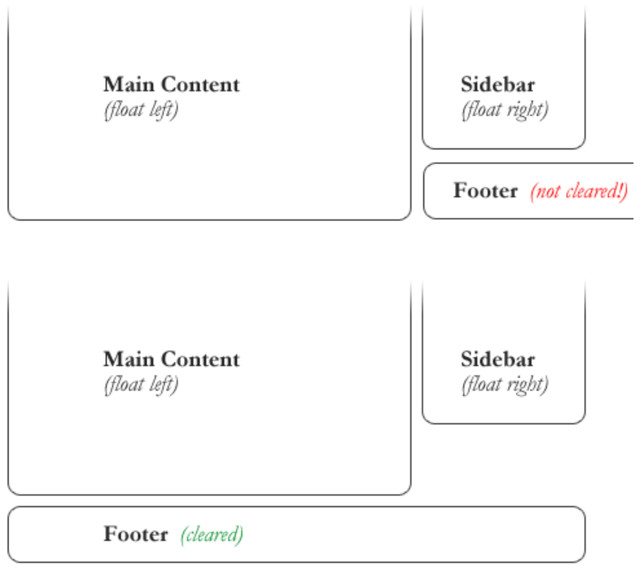
- To add an image to the page with text flowing around it, float it left

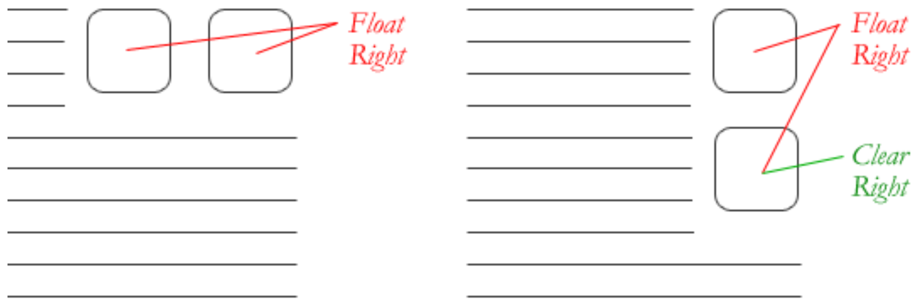
- Try adding some `margin-right` and `margin-bottom` to the image



Using `clear` for page layout

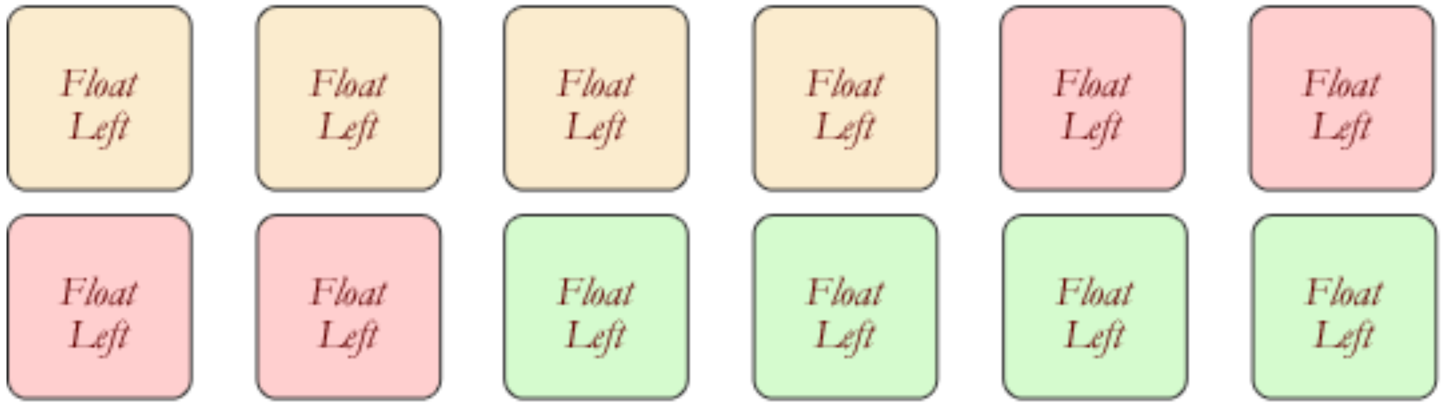
- The `clear` property will move the set element down past surrounding `float` property elements
- `clear` has 4 values: `both, left, right and none(default)`





Using `float` for page layout

- You could use `float: left;` to create a dynamic image gallery
- `float` each image left and give it a `margin-right`



Positioning with CSS

- Using the CSS `position` attribute, you can use X and Y values to move elements around the screen based on different frames of reference

- There are four possible values for `position`:

```
static /*This is the default value*/
fixed
absolute
relative
```

`left`, `right`, `top`, & `bottom`

- `left`, `right`, `top`, & `bottom` are used to specify an offset, the reference point of which is determined by the type of positioning (`fixed`, `absolute`, `relative`, etc.)
- For `left`, offset values that are positive will move your element to the right while negative values will move it to the left.
- `right` does the opposite from the other side of the screen
- For `top`, offset values that are positive will move your element down while negative values will move it up
- `bottom` does the opposite from the bottom of the screen

`fixed`

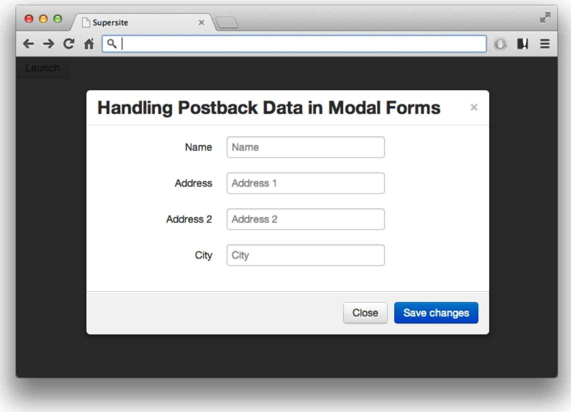
- Using `position: fixed;` your element will be positioned relative to the browser window
- The element will always be in the same place on the screen regardless of if the user scrolls
- `position: fixed;` is often used for “social media sharing bars” so the user has the option to scroll regardless of where they are on the page

`relative`

- Using `position: relative;` you can position your element relative to where it would normally be on screen
- The element is still in the normal “flow” of the page, it still takes up space
- Works well to make small changes in position unachievable through margin or padding

`absolute`

- Using `position: absolute;` you can position your element relative to the entire page
- The element will not be in the normal "flow" of the page, it will not take up space
- This is how "modal" elements are created



Putting it all together

Create a fake website for a newspaper,

“The New York Code + Design Academy Times”

- There should be two pages:
  - Home page, where 10 fake articles are listed in a two-column layout - they should all link to:
  - An example article with a link to Facebook that stays on the page no matter how much the page is scrolled
  - The example article should have a photo with text that wraps around the photo (hint: use a float!)

- Use the margin: 0 auto; width: 900px; “trick” to make the pages look nice

- If you finish the above, have fun with CSS making the newspaper look as professional as possible!

- Don’t worry about any of the actual text - just use lorem ipsum