Girl Develop It

GDI

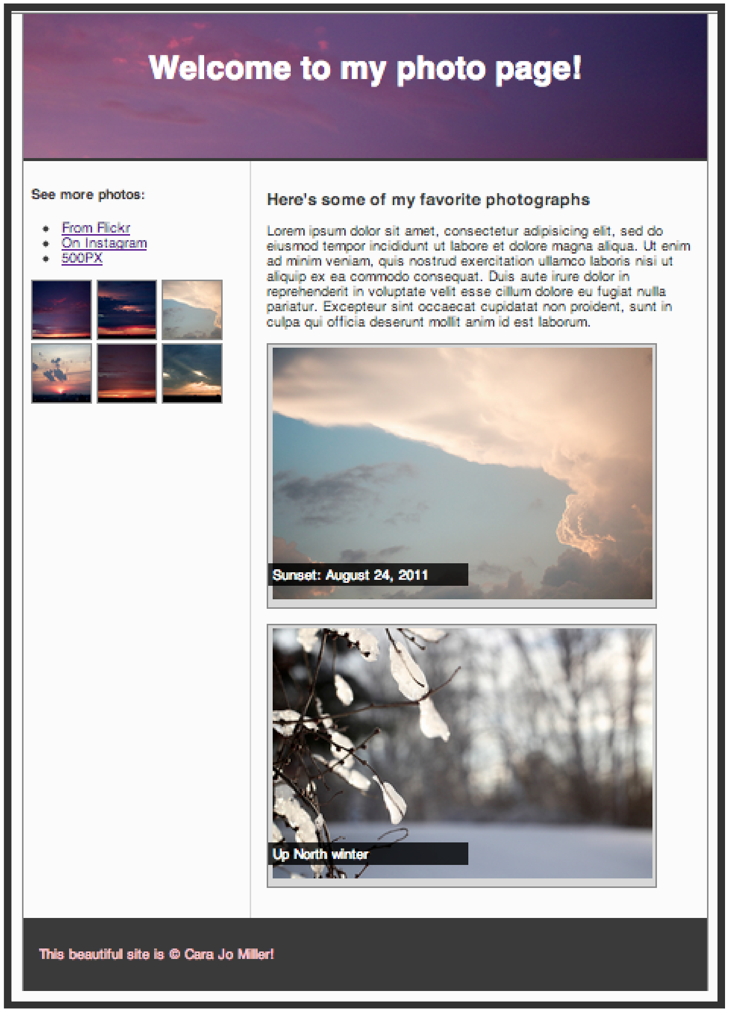
don't be shy. develop it.

Beginning HTML and CSS

Class 4

What we'll be making today

Today we will learn how to position our divs to create a page that has a 2-column layout.



Welcome to my photo page!

See more photos:

- From Flickr
- On Instagram
- 500PX

Here's some of my favorite photographs

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sunset: August 24, 2011

Up North winter

This beautiful site is © Cara Jo Miller!

A Common Practice

#wrapper

A really common code pattern on the web is using a *wrapper div* to help center a non-responsive website.

```
<body>
  <div id="wrapper">
    <div id="header">…</div>
    … the rest of your content …
  </div>
</body>
```

Develop It! Add a `#wrapper` to your `index.html` file from last class.

Positioning

Static Positioning

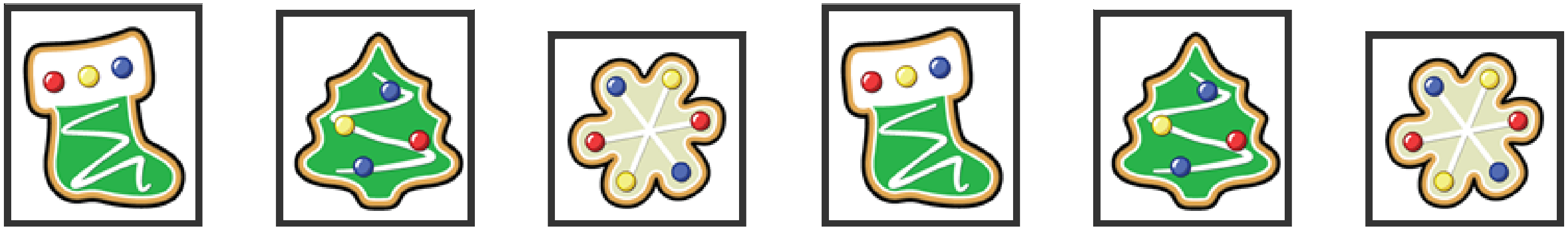
`position: static;`

- HTML elements are positioned **`static`** by default; you don't need a CSS declaration.
- Static elements are in the normal flow of the page.
- They ignore **`top`**, **`bottom`**, **`right`**, and **`left`** CSS declarations.

Static Positioning

In normal flow, inline boxes flow from left to right, wrapping to next line when needed.

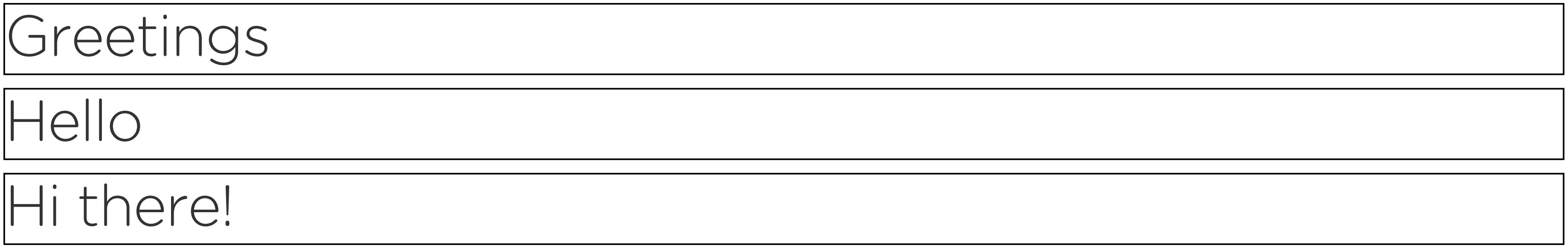
```
<img src="images/cookie1.png"/>
<img src="images/cookie2.png"/>
...
<img src="images/cookie3.png"/>
```



Static Positioning

In normal flow, block boxes flow from top to bottom, making a new line after every box.

```
<p>Greetings</p>
<p>Hello</p>
<p>Hi there!</p>
```



Greetings

Hello

Hi there!

Relative Positioning

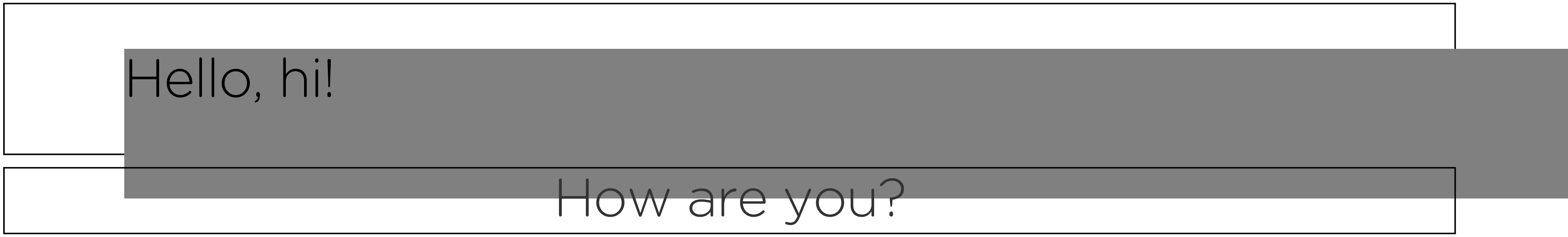
```
position: relative;
```

- Allows the element to be moved to the top, left, right or bottom.
- This does not affect the elements surrounding it.
- Makes an element a "positioning context" in which to position other elements relative to it.
- Relative positioning and absolute positioning are used together.

Relative Positioning

Relative positioning keeps the element in the normal flow, but offsets it according to **`top/left/right/bottom`** CSS declarations.

```
.relative{
  position: relative;
  left: 80px;
  top: 30px;
  height: 100px;
  background-color: gray;
}
```



Hello, hi!

How are you?

Absolute Positioning

`position: absolute;`

- Positions the element outside of the normal flow, meaning other elements act as if it's not there.
- It is offset from its container block, the first parent element that has a position other than static.
- If no such element is found, the container block is **`<html>`**.
- Offset using the properties **`top`**, **`left`**, **`right`**, & **`bottom`**.

Absolute Positioning

`position: absolute;` will take the element out of the normal flow and position it in relation to the window (or the closest non-static element).

Down here

Up here

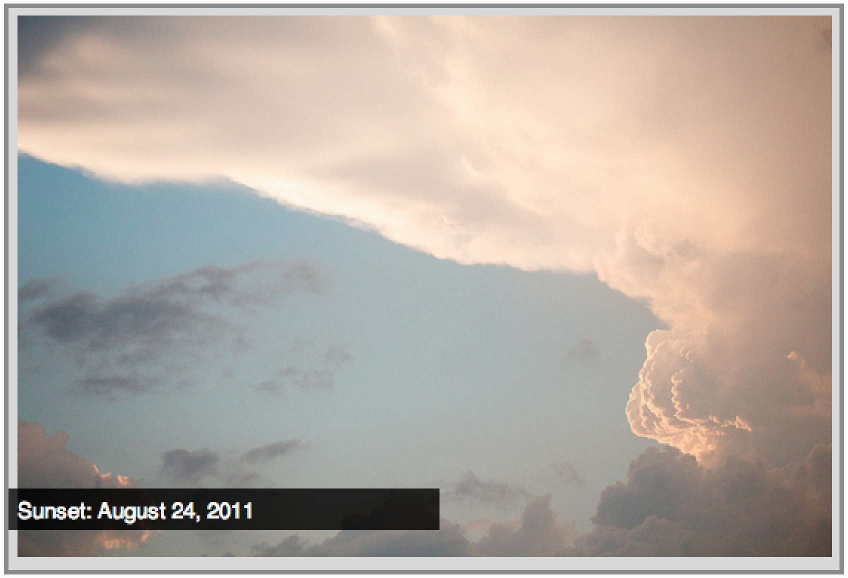
```
.top{
  position: absolute;
  top: -40px;
  right: 10px;
  background-color: yellow;
}

.bottom{
  position: absolute;
  bottom: 0;
  left:-10px;
  background-color: green;
}
```

Example: Absolute Positioning

Here's an example of an image with a caption absolutely positioned over top of it.

The containing **`div`** has a position of **`relative`**, and the caption has a position of **`absolute`**.



Stacking

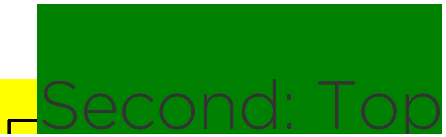
Z-index

`z-index: value;`

When you use positioning to move elements out of the normal flow of content, elements can overlap. You can change the order of overlapping with **`z-index`**.

The element with highest **`z-index`** goes on top.

```
.top{
  position: absolute;
  bottom: -10px;
  left:60px;
  background-color: green;
  z-index: 2;
}
.bottom{
  position: absolute;
  bottom: -30px;
  left:40px;
  background-color: yellow;
}
```



Changing the Flow

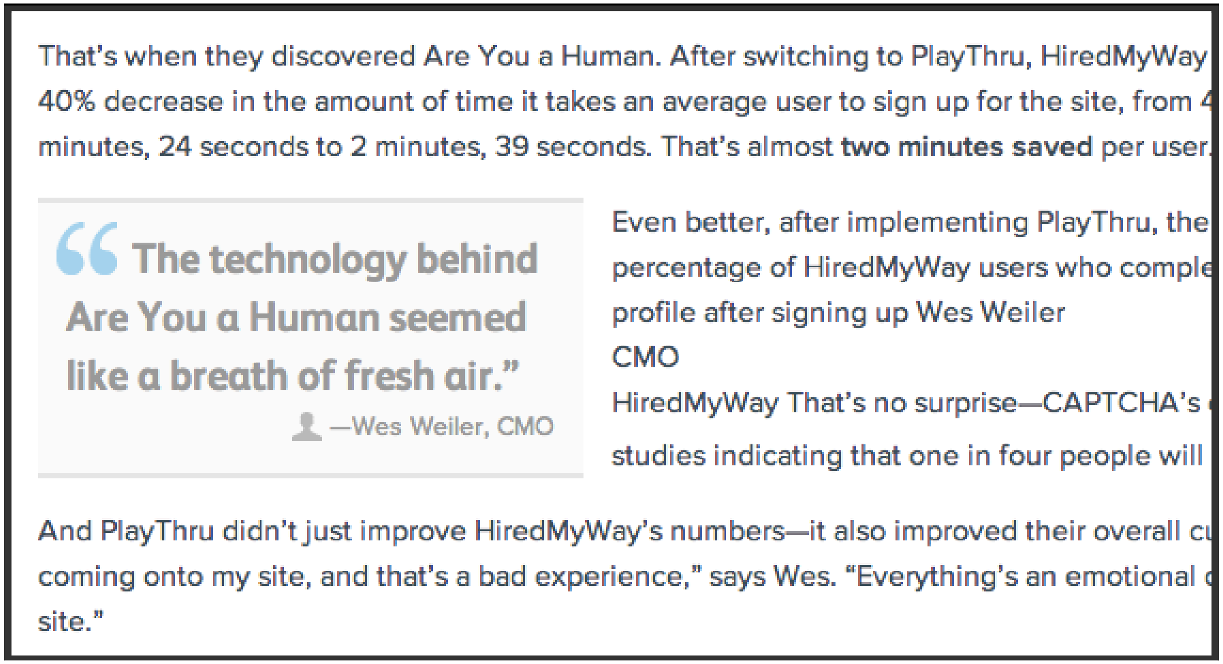
Float

```
float: left | right | none;
```

- "Floating" an element moves it in the flow, as far to the left or right of its containing element as possible.
- Any other elements will wrap around the floated element.
- You must specify a **`width`**, or it is likely to take up the whole page and not appear floated.

Float: Example

Below a `<blockquote>` is floated to the left, allowing text to wrap around it on the right



That's when they discovered Are You a Human. After switching to PlayThru, HiredMyWay 40% decrease in the amount of time it takes an average user to sign up for the site, from 4 minutes, 24 seconds to 2 minutes, 39 seconds. That's almost **two minutes saved** per user.

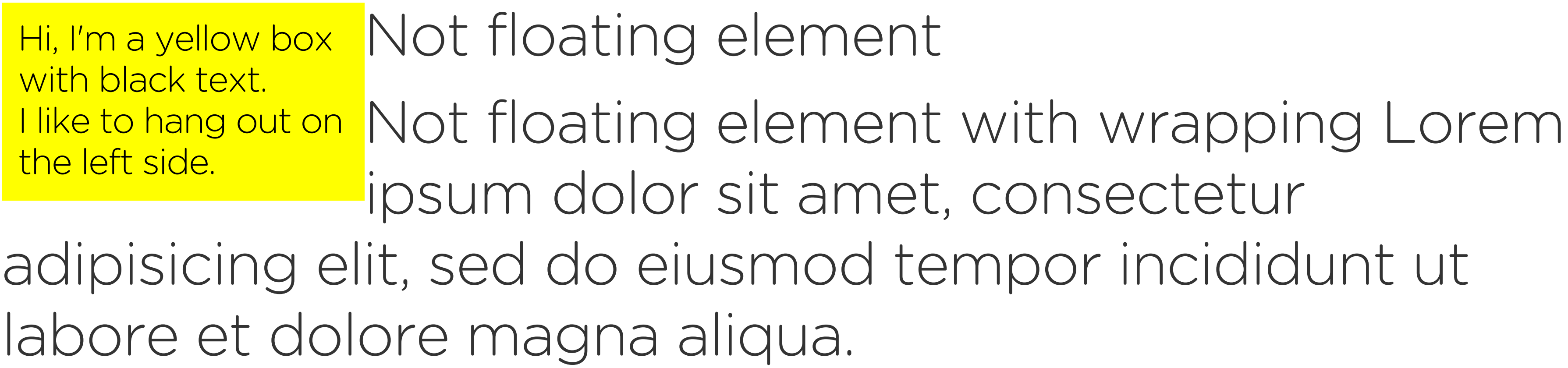
> "The technology behind Are You a Human seemed like a breath of fresh air."
>
> —Wes Weiler, CMO

Even better, after implementing PlayThru, the percentage of HiredMyWay users who comple profile after signing up Wes Weiler
CMO
HiredMyWay That's no surprise—CAPTCHA's
studies indicating that one in four people will

And PlayThru didn't just improve HiredMyWay's numbers—it also improved their overall cu coming onto my site, and that's a bad experience," says Wes. "Everything's an emotional c site."

Float

```
.pull-quote{
  float:left;
  width:200px;
  background:yellow;
}
```



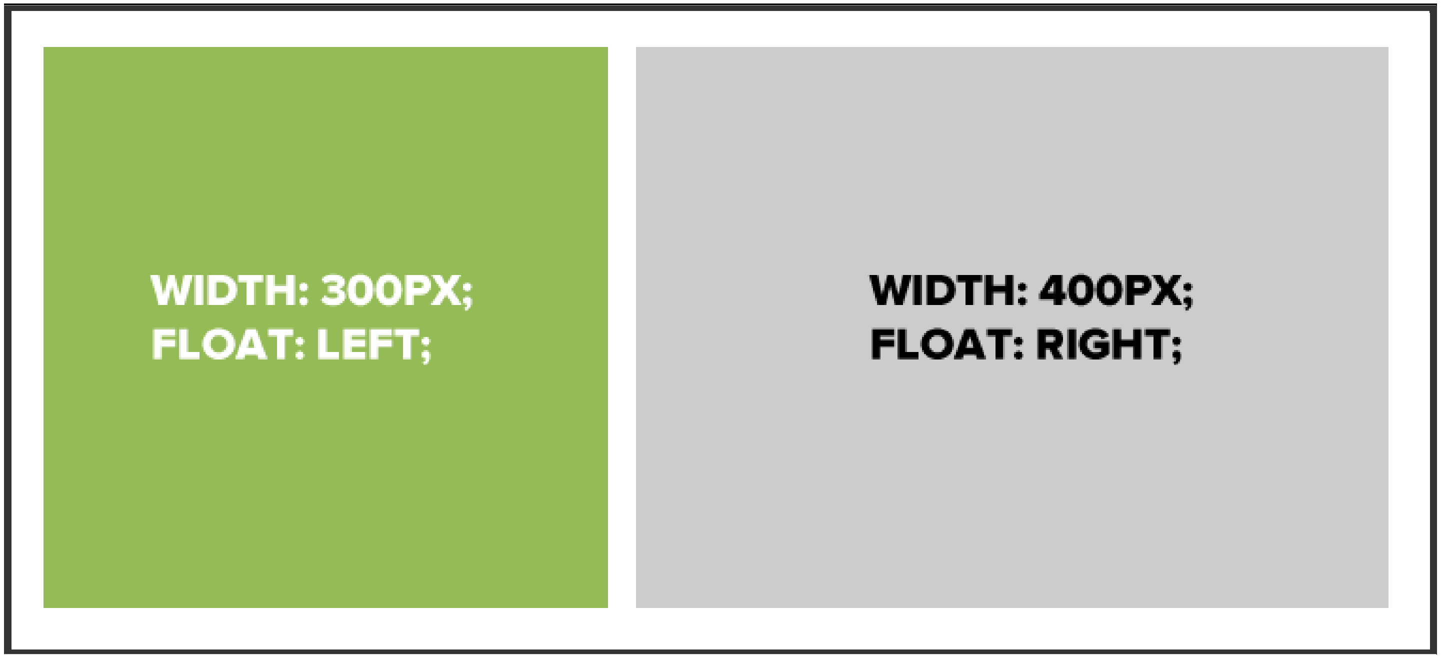
Hi, I'm a yellow box
with black text.
I like to hang out on
the left side.

Not floating element

Not floating element with wrapping Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Placing Elements Side by Side

If you want two block-level elements to be side by side, you need to float both elements. One left, and one right.



Clear

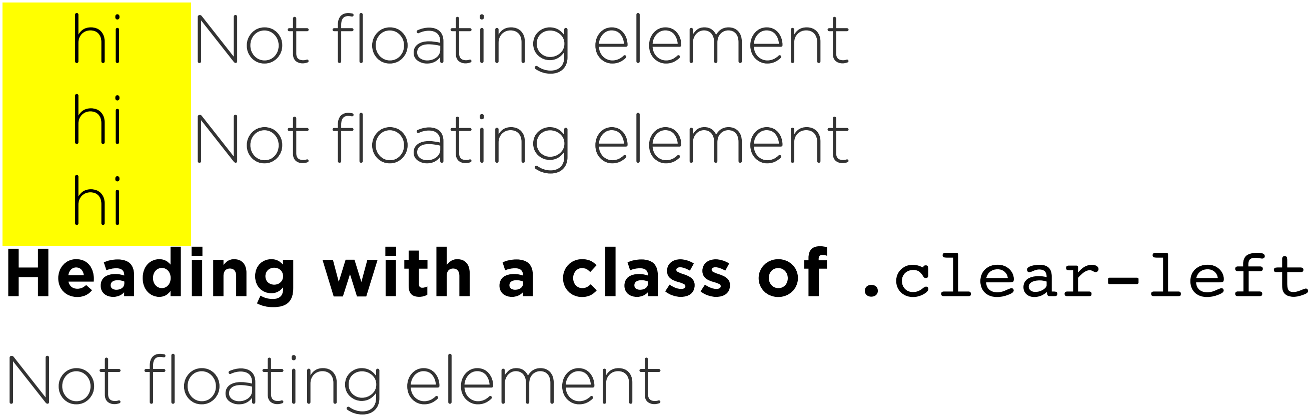
`clear: left | right | both;`

- Clearing tells the element on which side (right, left, both) other elements **cannot** appear.
- Clearing both sides makes sure floats don’t flow past the clear element.
- `clear` is applied to the non-floating element.

  Ex: If you had an image floated left, and you did not want the heading to appear next to it, you would add `clear: left;` to the **heading**.

Clear

```
.pull-quote{
  float:left;
  width:100px;
  background:yellow;
}
.clear-left{
  clear:left;
}
```



hi Not floating element
hi Not floating element
hi

Heading with a class of `.clear-left`

Not floating element

Changing Display

Display

What if your inline element needs a **`margin`**
(Ex: a floating image)?

What if your block-level list items need to be in a row
(Ex: a menu)?

You can change the way an element is displayed.

```
display:
block | inline | inline-block;
```

Display: Navigation Using Lists

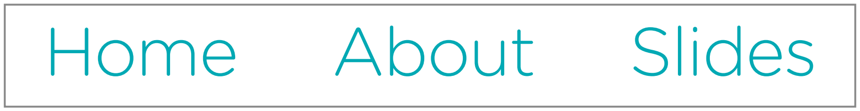
```html
<ul id="menu">
  <li><a href="#">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Slides</a></li>
</ul>
```

Would normally look like:

- Home
- About
- Slides

```css
#menu {
  border: 1px solid #888888;
}
#menu li {
  display: inline-block;
}
#menu li:hover {
  background-color: #aaaaaa;
}
#menu a {
  display: block;
  padding: 5px 20px;
}
```

Now looks like:



Home About Slides

Questions?

?