INTRODUCTION TO HTML & CSS

Instructor: Beck Johnson

Week 5

SESSION OVERVIEW

- Review float, flex, media queries
- CSS positioning
- Fun CSS tricks
- Introduction to JavaScript
- Evaluations

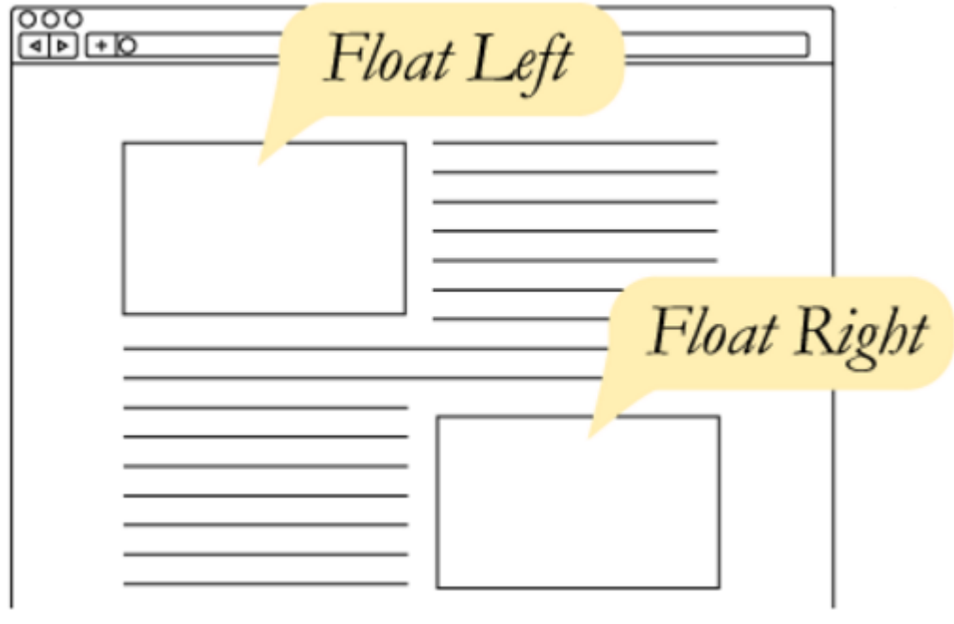
REVIEW!

CSS FLOATS

The `float` property takes an element out of the normal flow and "floats" it to the left or right side of its container.

- This allows other content
  to flow around it

```
img { float: left; }
```

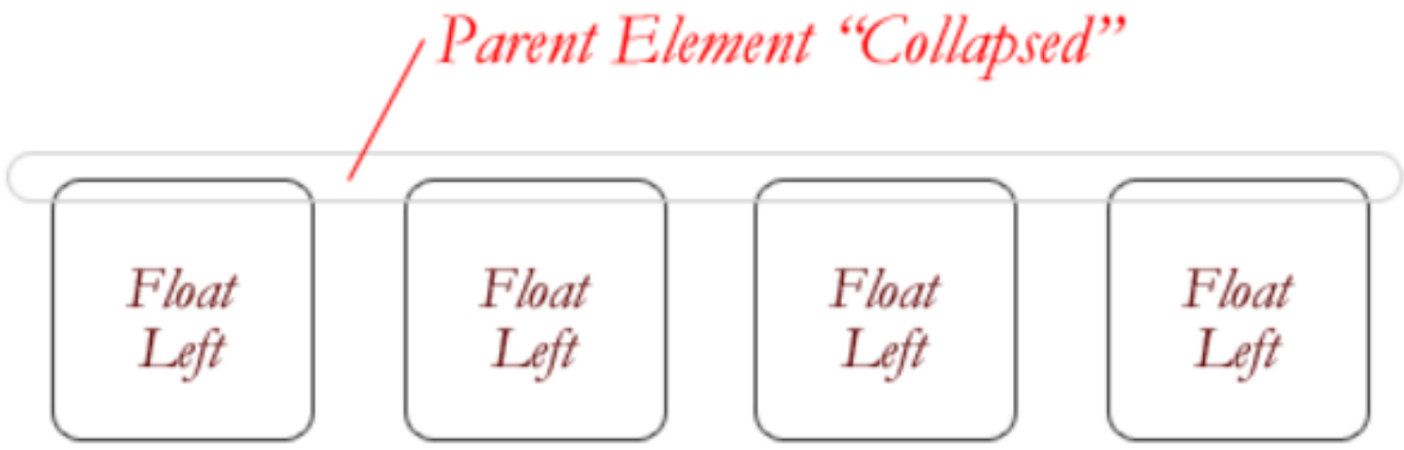


CSS FLOATS

The three values for `float` are:

- `left`
- `right`
- `none`

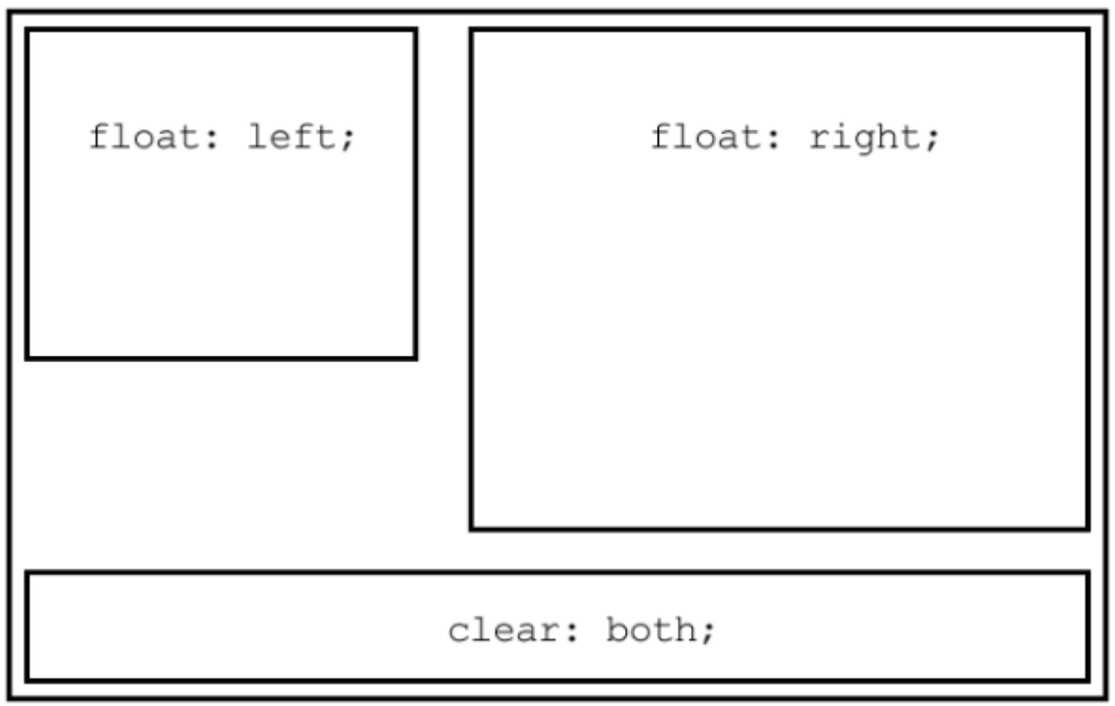
If everything in a container is floated, then the container thinks it's empty.



THE CLEAR PROPERTY

The `clear` property is the sister property to `float`

- It doesn't do much until there are floated elements on the page
- An element with `clear` applied to it will force itself **below** the floated element
- Everything after that will be back in the normal flow
- This "stretches" out the container and keeps it from collapsing



THE OVERFLOW PROPERTY

`overflow` is a CSS property that governs how content looks when it breaks out of its container.

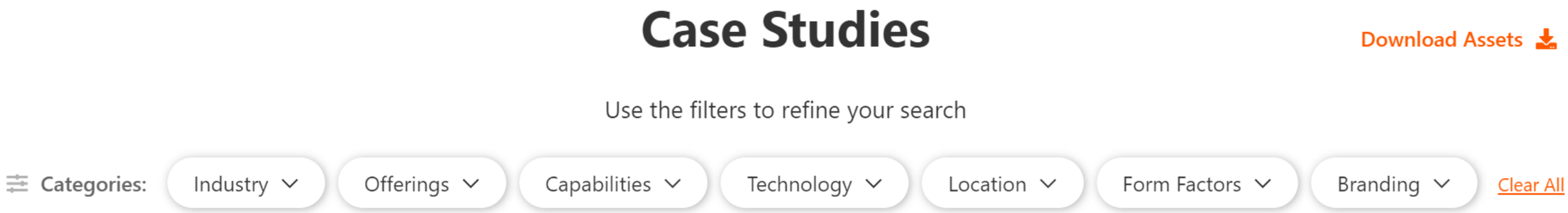
By default, elements have `overflow: visible`, which means all content is fully visible.

`overflow: auto` adds scrollbars when the content is bigger than its container.

LAYOUT WITH FLEXBOX

Instead of using `float` to change the layout of your page, you can use `display: flex`

Adding `display: flex` to a container puts every direct child inside it one line



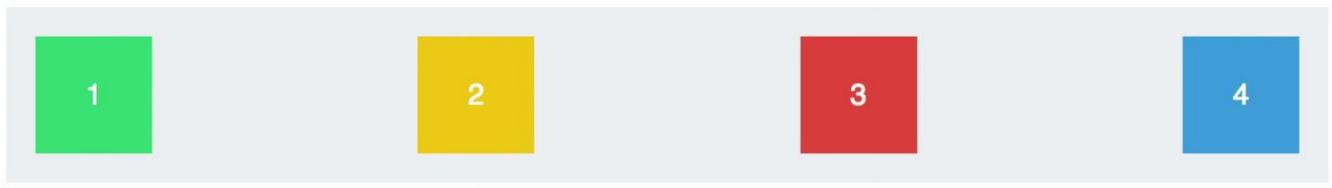
JUSTIFY-CONTENT

`justify-content` controls how horizontal space is distributed between flex children

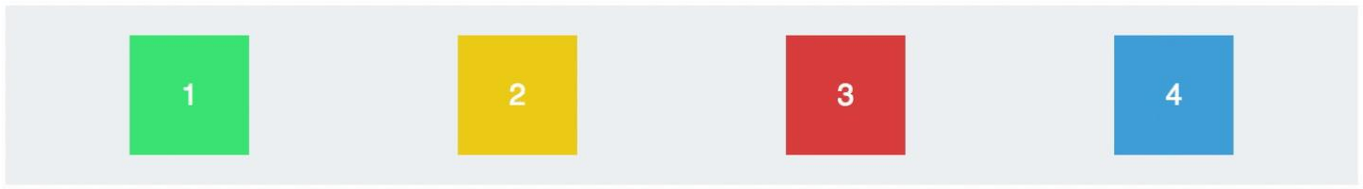
`center` puts everything together in the middle

`flex-start` and `flex-end` move everything to either the beginning or end of the row

`space-between` spreads all the items out evenly, with the first and last items flush against the ends of the container



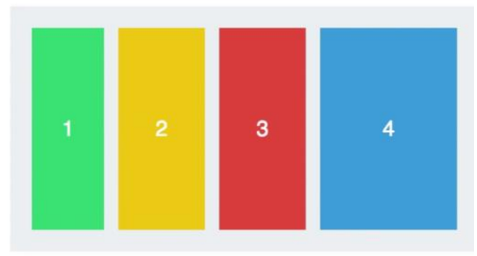
`space-around` distributes all the items with equal space around them



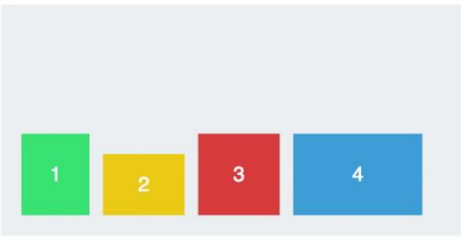
ALIGN-ITEMS

`align-items:` controls how items are aligned relative to one another vertically

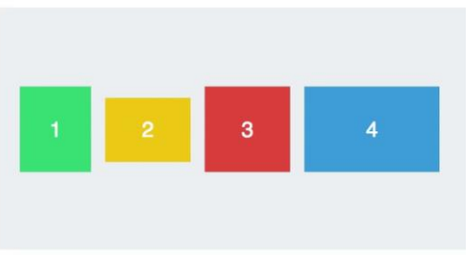
`stretch` makes all children stretch to the same height



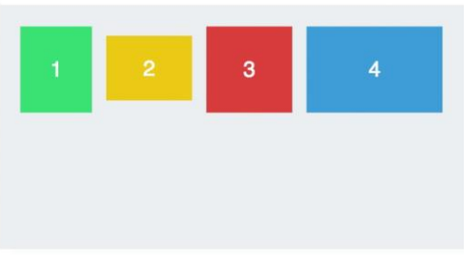
`flex-start` and `flex-end` align all children to either the top or bottom of their container



`center` makes all children vertically centered, with equal space on top and bottom



`baseline` vertically centers children so that their text lines up



WEB FONTS

```
<link href="https://fonts.googleapis.com/css?family=Roboto"
rel="stylesheet">
```

Once you include a font stylesheet from a CDN, you can refer to the font in a `font-family` rule, just like you would a web-safe font:

```
p { font-family: Roboto, sans-serif; }
```

MEDIA QUERIES

Media queries start with `@media` and have curly braces that contain all the CSS that applies for that case

```
@media (min-width: 768px) {
    /* Only when bigger than tablet */
    h1 { font-size: 20px; }
}
```

QUESTIONS?

ASSIGNMENT

Based on our "match the comp" html file, use `display: flex` to give the cards a gallery layout

Starting point:
https://beckjohnson.com/demos/classes.html

CSS POSITIONING

WEB LAYOUT PROPERTIES

- `display:` dictates how elements behave within the box model

- `float:` moves elements around within the page flow

- `position:` takes elements entirely out of the page flow

CSS POSITIONING

The `position` property specifies how an element is positioned on the page. Possible values are:

- `static`
- `fixed`
- `absolute`
- `relative`

The default `position` is `static`, which just means that the element obeys whatever its box model rules tell it to do.

POSITION: FIXED

`position: fixed` makes content "stick" to the browser window, regardless of where the user scrolls.

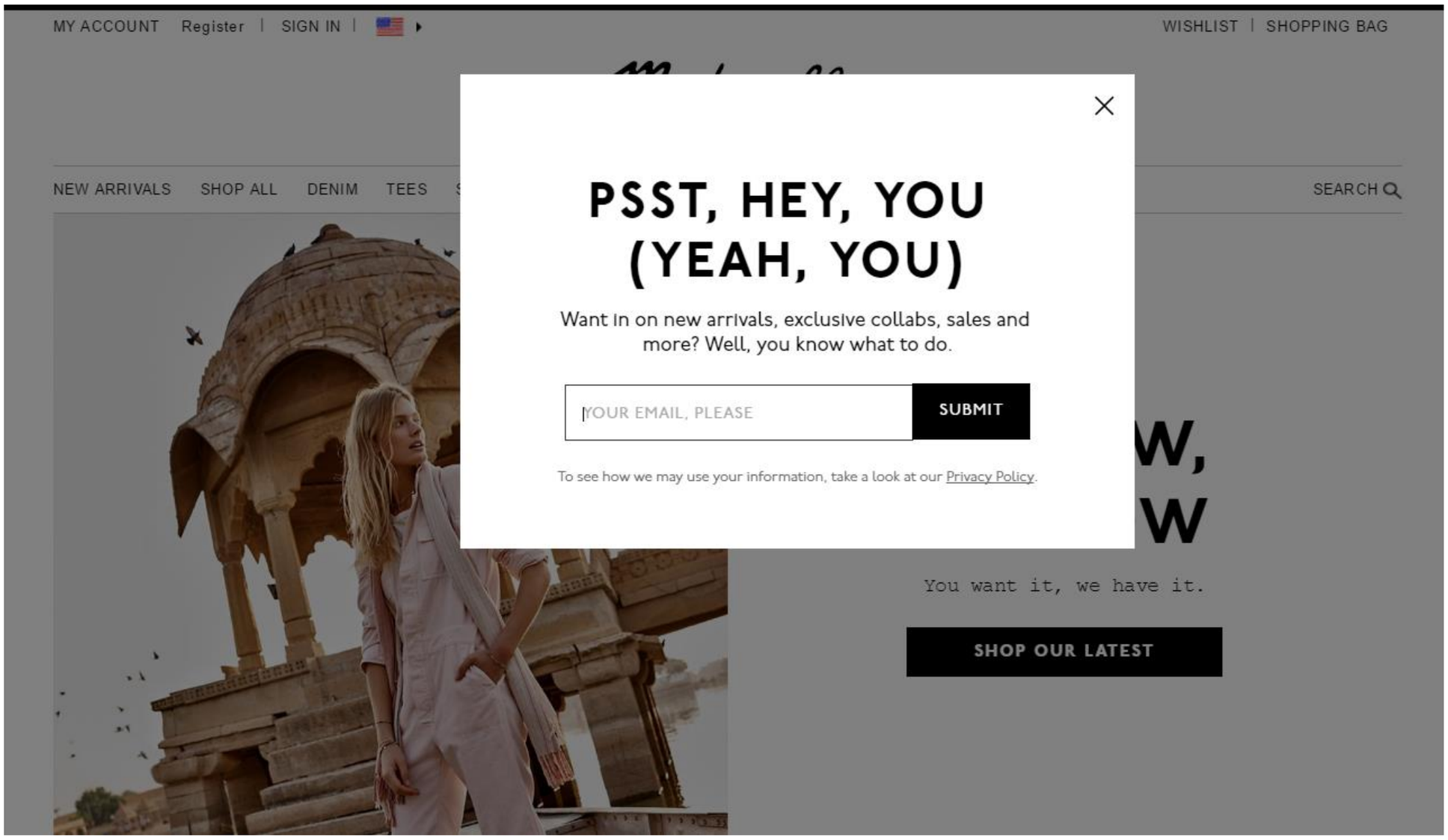
Commonly used to make headers, navigation menus, or sidebars that follow the page as it scrolls.

```
nav {
    position: fixed;
    left: 0px;
    top: 0px;
}
```

Hard to describe, see a live demo

FTFY

This popup background uses `position: fixed` to grey-out the entire page behind the lightbox, even if the user scrolls.



ABSOLUTE

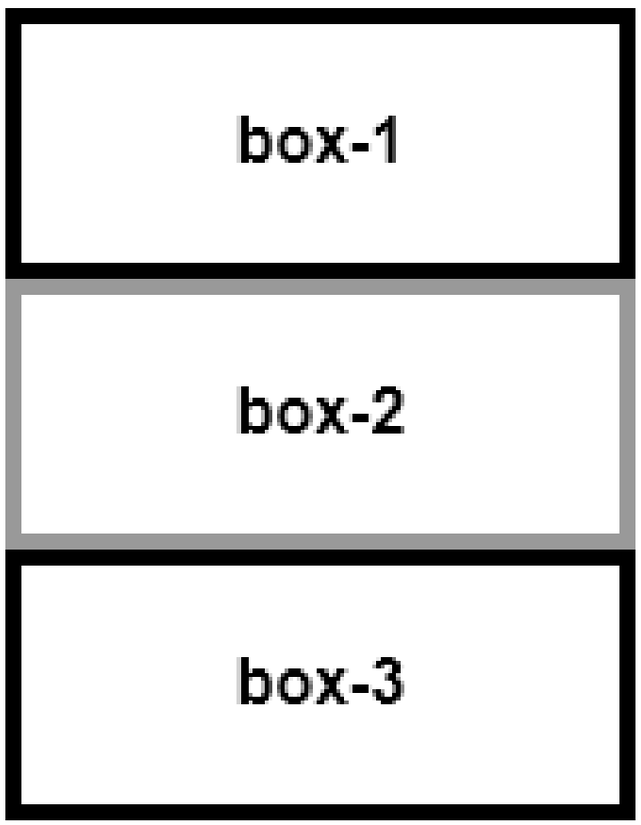
`position: absolute` is a powerful tool that allows you to place any page element exactly where you want it, down to the pixel

When an element has `position: absolute`, it is entirely removed from the normal flow of the page

- That means its padding, margins, and borders no longer affect the elements around it

DEFAULT POSITIONING

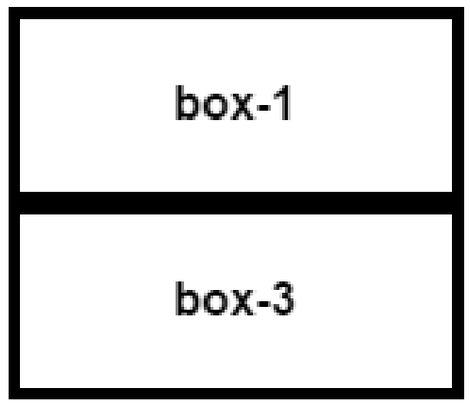
Remember that by default, block elements full up their entire row, and push any content to the next line, like this:



ABSOLUTE

If we give box-2 absolute positioning:

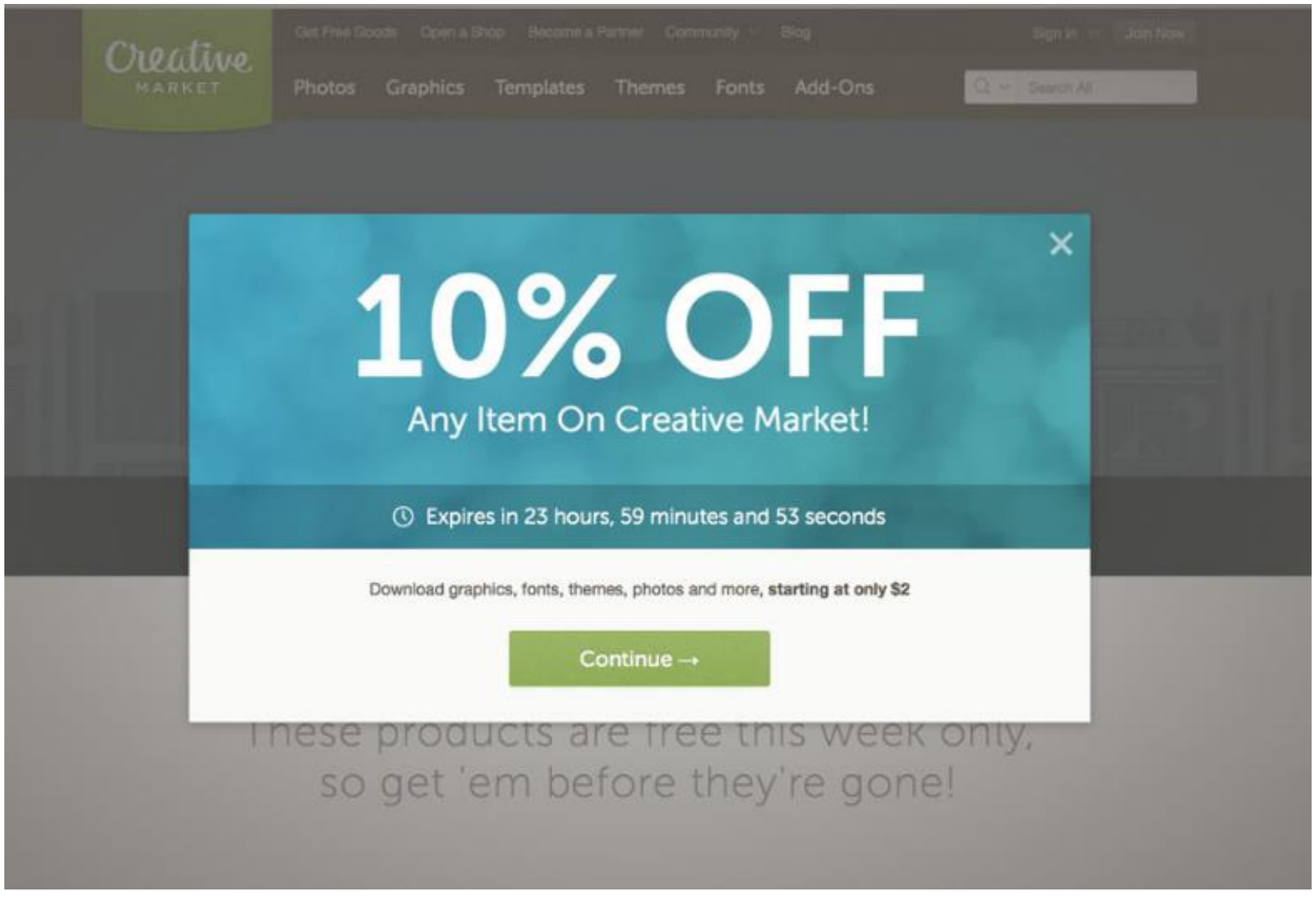
```
#box-2 {
    position: absolute;
    right: 0px;
    bottom: 0px;
}
```



- `box-2` is moved to the bottom right of the page
- `box-3` moves up to occupy the space vacated by `box-2`

ABSOLUTE POWER

`position: absolute` can be used to create lightboxes (also known as modals) that pop up over other content



ABSOLUTE POWER

Add to list

Death Note

+ Add to list

Fullmetal Alchemist

+ Add to list

Naruto

Spirited Away

Spirited Away

Alt title: Sen to Chihiro no Kamikakushi

Movie (1 ep) | Studio Ghibli | 2001 | 4.6

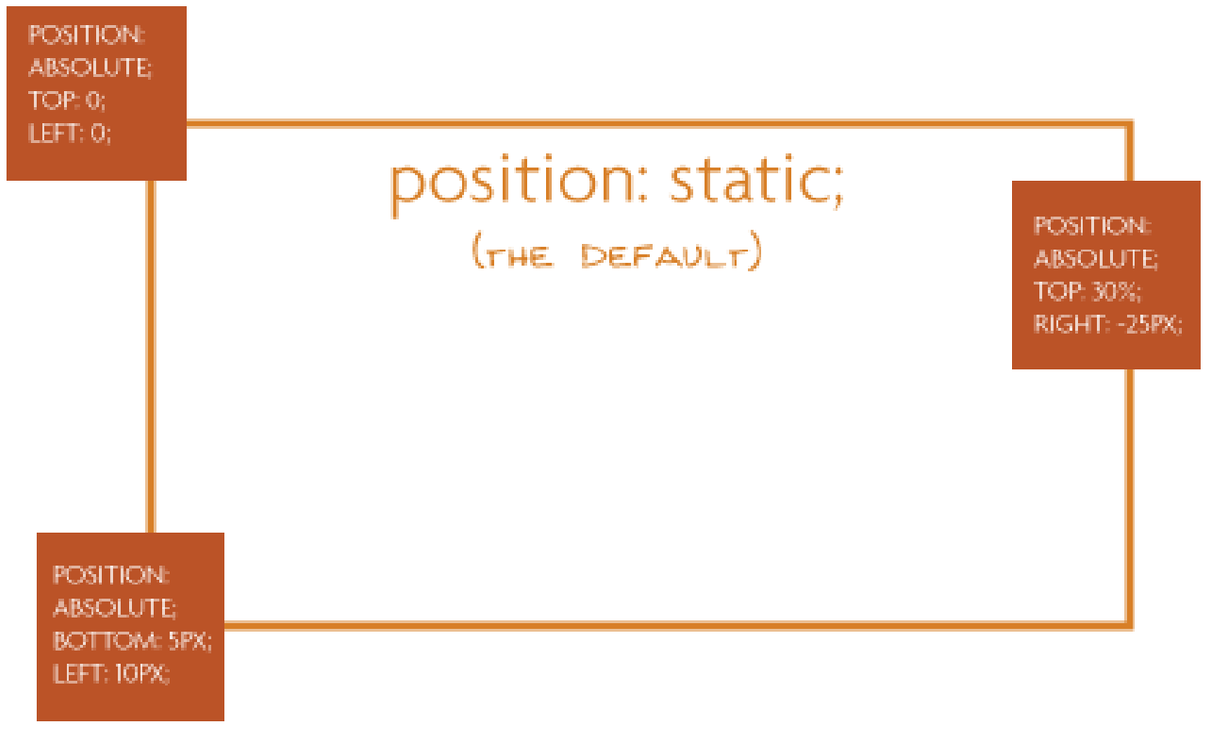
Chihiro and her family are on their way to their new home, when they discover an abandoned amusement park. After Chihiro's family mysteriously turn into pigs, she is thrown into a surreal world of magic and fantasy. Join her as she struggles to survive in the bathhouse of the gods, ruled by an evil witch who has stolen not only her name, but her way back to the real world.

TAGS Adventure | Fantasy | Curse | Family Friendly | Japanese Mythology | Magic | Person in a Strange World | Youkai | Original Work

ABSOLUTE

An element with `position: absolute` is absolutely positioned:

- To its closest parent with positioning
- Or if it has no parents with positioning, to the body of the page



ABSOLUTE

POSITION:
ABSOLUTE;
TOP: 0;
LEFT: 0;

position: static;
(THE DEFAULT)

POSITION:
ABSOLUTE;
TOP: 30%;
RIGHT: -25PX;

POSITION:
ABSOLUTE;
BOTTOM: 5PX;
LEFT: 10PX;

CSS POSITIONING

There are 4 directional properties that affect positioning:

- `left`
- `right`
- `top`
- `bottom`

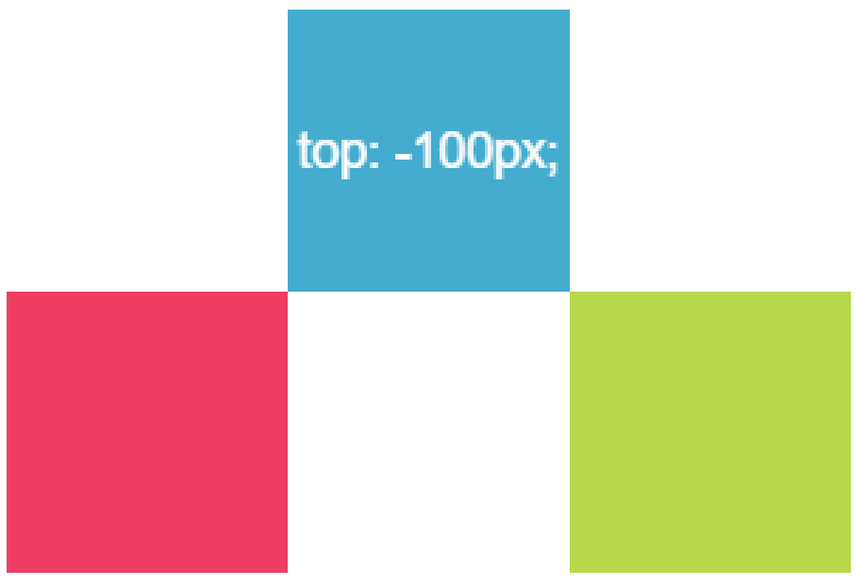
The value is a number (positive or negative) followed by a unit.

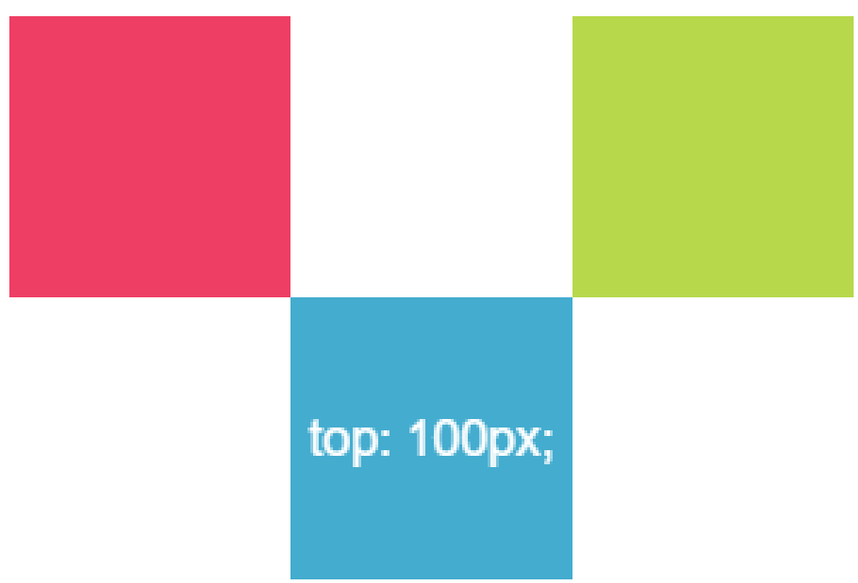
They define how far an element is offset that direction.

CSS POSITIONING

`top` defines how far an element is offset from its original top edge.

- Positive `top` values push an element **down**
- Negative `top` values push an element **up**

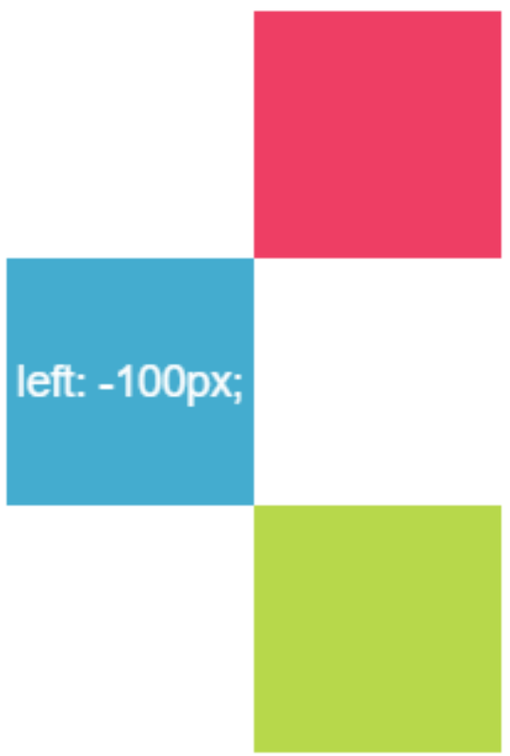




CSS POSITIONING

Similarly, `left` defines how far an element is offset from its original left edge.

- Positive `left` values push an element **right**
- Negative `left` values push an element **left**



GET YOUR Z'S

When you start changing how elements are positioned, you may need to specify which ones should be "on top", if they begin to overlap

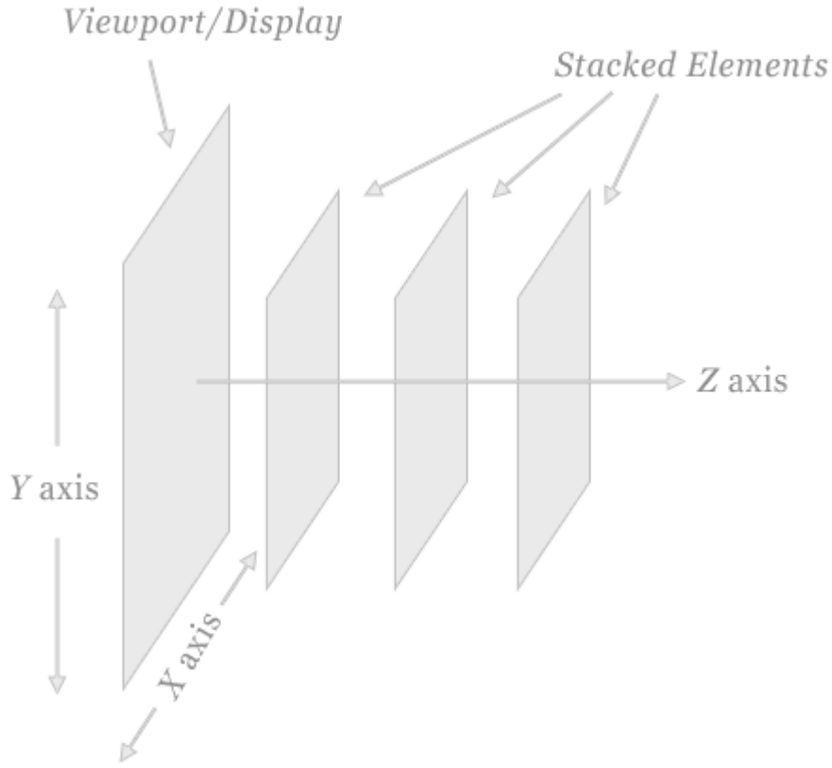
- Elements with no `position` in CSS will appear **under** elements that have `absolute`, `fixed`, or `relative` positioning

- By default, the *last* item on the webpage will appear on top of earlier elements that have the same type of `position`

GET YOUR Z'S

The CSS property `z-index` can be applied to force an element with position on top of any other by giving it a ***higher*** number

- By default all elements have `z-index: 0`
- Can assign negative or positive values to `z-index`



RELATIVE

`position: relative` allows you to move an element using directional attributes (`top`, `bottom`, etc).

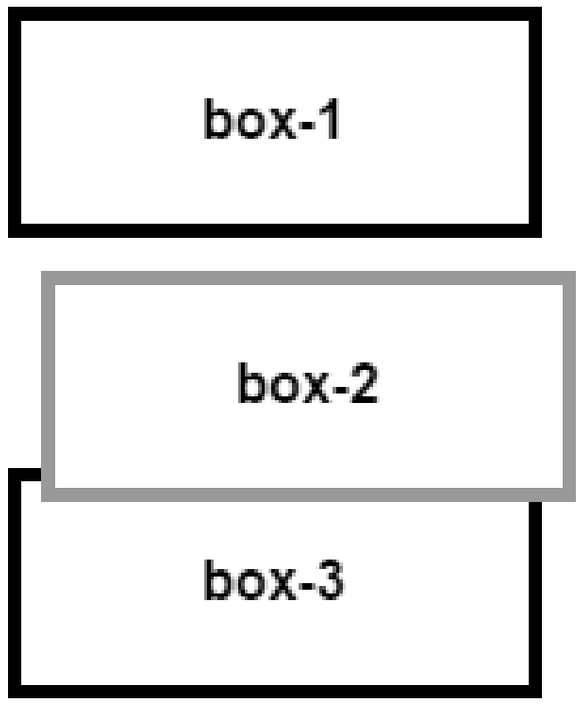
- If you set `position: relative` but no directional attributes, the element won't change at all

- If you set `top: 10px;` then it will be shifted 10 pixels down from where it would normally be

RELATIVE

box-2 has the following style, which moves it down and right:

```
#box-2 {
    position: relative;
    left: 10px;
    top: 10px;
}
```



The other boxes behave like box-2 was in its original position (unlike using negative margins).

RELATIVE

Unlike `position: absolute`, moving an element using `relative` means there will be space on the page where it would have been

RELATIVE

More commonly, `position: relative` is used on a parent container to position absolutely-positioned elements inside itself (as opposed to moving itself)

PRACTICE TIME!

ASSIGNMENT

Create a new page using this page as a template:

http://beckjohnson.com/dev-101/demos/modal.html

Write CSS so that the modal (the `div` with the class `modal`) appears over the boilerplate text

- Apply box model properties to the content until it looks nice

Hint: Recall that you target a class in CSS like this:

```
.modal
```

FUN CSS TRICKS

OPACITY

The `opacity` property changes an element's transparency

- By default, all elements have `opacity: 1;` meaning fully opaque
- To make an element transparent, specify a decimal value between 0 and 1

```
h2 { opacity: .5; }
```

Keep in mind that opacity value will apply to all children, and they can never become more opaque than their parent!

FILTER

The `filter` property can be used to apply graphical effects similar to Photoshop filters on images, backgrounds, or borders.

Some options include blur, grayscale, brightness, saturate, and sepia.

Its value is one of the above options, followed by parenthesis indicating how much to apply the effect, like this:

```
img { filter: grayscale(1); }
```

FILTER

Original image

```
filter: grayscale(1);
```

Converts the image to gray. 1 is fully gray, 0 is original. Any value between 0 and 1 is allowed

```
filter: sepia(1);
```

Converts the image to sepia. 1 is fully sepia, 0 is original. Any value between 0 and 1 is allowed

FILTER

```
filter: saturate(8);
```

Larger values are more saturated. Anything under 1 will make the element less saturated than the original

```
filter: hue-rotate(90deg);
```

Must be an angle of rotation in degrees, that will affect how far around the color circle the input is adjusted

```
filter: invert(.8);
```

Must be a percentage. Flips RGB values by that amount

FILTER

```
filter: brightness(3);
```

Larger values are brighter. Anything under 1 will make the element darker

```
filter: contrast(4);
```

Larger values apply greater contrast. Anything under 1 will apply less contrast than the original

```
filter: blur(5px);
```

Applies a Gaussian blur. The px value is how many pixels will blend into each other

ANIMATE

To allow elements on your page to animate using CSS, use the `transition` property

- By default, an element that transforms will change abruptly – for example, when you changed the text color of links using `:hover`

- `transition` makes those changes occur smoothly over time, instead of suddenly

ANIMATE

```
nav { transition: all 1s; }
```

This means, animate all CSS properties that happen to `nav` for 1 second

- The first value: *which* CSS properties can be animated
- The second value: *how long* the animation should take to finish
- Separate the values with spaces

ANIMATE

Until there is a CSS property that changes, `transition` won't have any noticeable effect

Remember that you can give any element a different style when the user hovers on it

- Use the CSS rule `:hover`
- This will allow us to see the animation effect as the style changes from one value to another

FILTER ON HOVER

```
img {
   filter: grayscale(0);
   transition: all 1s;
}

img:hover {
   filter: grayscale(1);
}
```

Note that a `grayscale` of `0` (ie, none) is applied to the "normal" image. This allows the `transition` to run both when the user hovers *and* when they move their mouse away

TRANSFORMATION

The `transform` property lets you manipulate an element by skewing, rotating, moving, or scaling

Like `filter`, the value is the type of transformation you want to apply, with the degree of transformation inside parentheses

```
.bigger {
   transform: scale(20);
}
```

TRANSFORMATION

`transform` options include:

`scale` – changes the size of an element

`skew` – tilts the element

`rotate` – rotates the element a specified number of degrees

`translate` – moves an element

TRANSFORMATION

Potential uses for `transform`:

- Flip an arrow when sorting or expanding a menu

```
.toggle {
	transform: rotateZ(0deg);
	transition: all .11s;
}

.toggle:hover {
	transform: rotateZ(180deg);
}
```

For a live example, see the Starbucks footer at mobile

TRANSFORMATION

- “Lift” a card when the user interacts with it

```
.card {
      transform: scale(1);
      transition: all .15s ease-in;
}

. card:hover {
      transform: scale(1.01);
}
```

For a live example, see the Dignity Memorial website

PRACTICE TIME!

PLAY WITH ANIMATION

Apply `transition` to at least one element on your page

- Give that element a different style on hover/focus so that you can see the animation occur

Apply `filter` or `transform` to at least one element on your page

- Play with all the possibilities!

Reference https://robots.thoughtbot.com/transitions-and-transforms to see more options for `transform`

OVERVIEW OF JAVASCRIPT

WHAT IS JAVASCRIPT?

JavaScript is a programming language that runs in your web browser

- Can manipulate any element on the page
- Can listen to user interaction, such as button clicks or scrolling the browser window

WHERE TO PUT JAVASCRIPT?

JavaScript goes inside a new tag called `<script>`

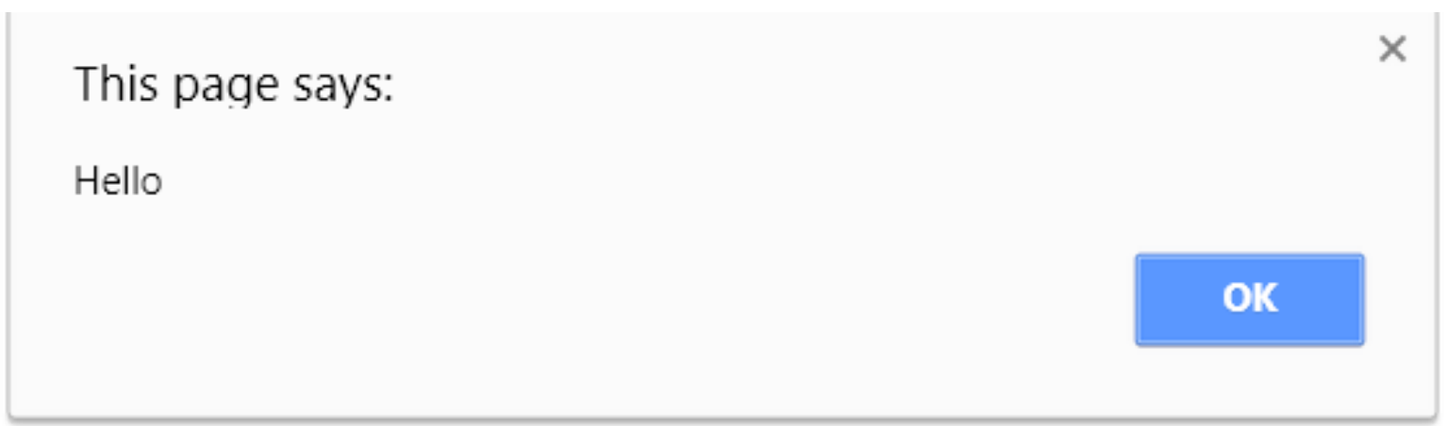
- Just like how anything inside a `<style>` tag is expected to be a different "language" than your HTML document (CSS), everything inside a `<script>` tag is expected to be in the Javascript language

- Unlike malformed CSS, you'll see red errors in the console if the browser can't understand how to read things inside `<script>`

JAVASCRIPT EXAMPLE

```
<button id="signup">Sign up</button>

<script>
   var signupButton = document.getElementById("signup");
   signupButton.addEventListener("click", function(){
      alert("Hello");
   });
</script>
```



See a live demo

WHAT IS JQUERY?

jQuery is a popular Javascript library that makes it easy to locate and manipulate elements

- Uses CSS syntax to locate DOM elements
- Handles browser inconsistencies (so that you don't have to)
- Simplifies many common tasks

USING JQUERY

To be able to use jQuery, you have to include a link to the Javascript file that contains the jQuery library

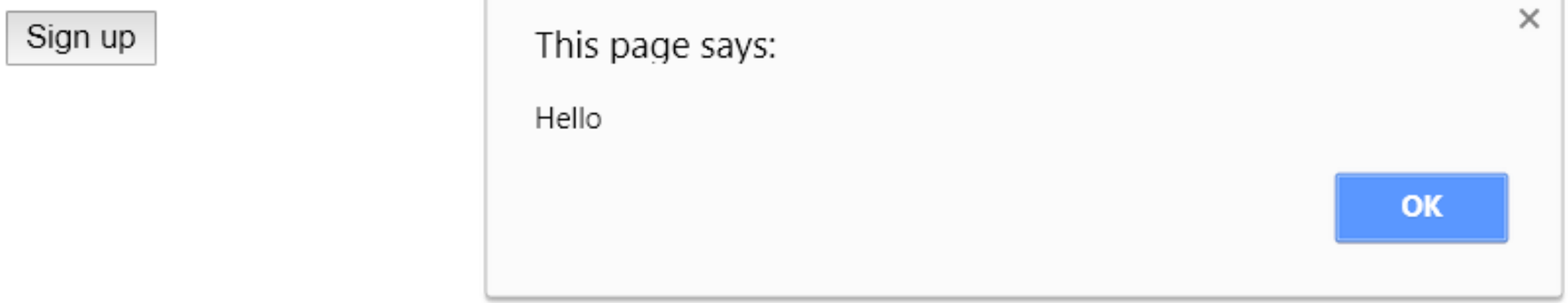
- Like web fonts, the easiest way to get started is to link to a CDN:

```
<script src="https://code.jquery.com/jquery-
1.12.4.js"></script>
```

JQUERY EXAMPLE

```
<button id="signup">Sign up</button>


<script>
   var signupButton = $("#signup");
   signupButton.click(function(){
      alert("Hello");
   });
</script>
```



BETTER JQUERY EXAMPLE

```
<a class="menu-toggle">Top level menu</a>
<ul class="menu">
        <li>Menu item</li>
        <li>Menu item</li>
        <li>Menu item</li>
</ul>

<script>
   $(".menu-toggle").click(function(){
           $(".menu").slideUp();
           $(this).next(".menu").slideDown();
   });
</script>
```

See a live demo

JQUERY PROVIDES EASY ANIMATIONS

jQuery makes showing or hiding elements very easy, and provides a bunch of animation options as well

- Show/hide
- Fade in / fade out
- Slide down / slide up

Each of these options can be given a duration (how long the animation lasts) and a speed (called “easing”, which can vary over time)

See some jQuery animation examples

“THE END”

END TIMES

I will be keeping all slides and demos up on the class site indefinitely

You can continue to ask me questions anytime at beckjohnson@gmail.com

THAT'S ALL FOLKS!

Please provide feedback for this class!

http://svcseattle.com/evaluation