FEWD

Week 6 · Class 13

Objects

Review

Add the error message to the msg div.

```
<div id="msg"></div>
```

```
var errorMsg = "<p>Please accept the terms</p>"

$(/* what goes here */).html(errorMsg);
```

Review

Make the Add To Do button work

```
<div id="todo-list"></div>
<button id="add-todo">Add To Do</button>
```

```
var template =
`<div>
  <input type="text" class="todo-input">
  <button>Save</button>
</div>`;

$(/* what goes here */).click(function(){
  $(/* what goes here */).append(template);
});
```

Review

Place the greeting in the paragraph

```
<input name="name" id="name" value="" placeholder="Enter your name">
<button>Submit</button>
<p id="greeting"></p>
```

```
$('button').click(function(){
  var inputValue = $(/* what goes here */).val();
  var greetMsg = "Hello " + /* what goes here */;

  $(/* what goes here */).text(/* what goes here */);
});
```

What We'll Cover

- What are objects?
- How to create objects
- How to get the stuff that's inside of them
- How to add stuff to them

“Understand objects and you will understand JavaScript.”

Cody Lindley, JavaScript Enlightenment

Objects

If you thought arrays were cool (*and I know you did*), you'll love objects. Objects are array on steroids. In fact, arrays are a form of object, as are ***almost everything*** else in Javascript.

What are objects?

Ummmm, you know, they're objects. 🙂

```
var table = {
  legs: 4,
  color: "white",
  style: "contemporary",
  materials: ["metal", "formica", "particleboard"]
};
```

Creating Objects

The simpliest way to create an object is with {}

```
var person = {

  firstName: "Jane", /* property: value */
  lastName: "Smith",
  age: 30,
  eyes: "blue",
  hair: "brown" /* no comma after the last property */

};
```

Object Properties

Whereas arrays have elements you access with an index, objects have properties you access by name.

```
var pet = {
  name: "Cosmo",
  age: 11,
  cute: true,
  color: ["black", "white"]
};

pet.age = 12; /* dot notation */
pet["age"] = 12; /* square bracket notation */
```

Object Properties

Properties can be added to objects any time.

```
var pet = {
  name: "Cosmo",
  age: 12,
  cute: true,
  color: ["black", "white"]
};

pet.birthday = new Date("September 12, 2005"); /* JS Date Object */
pet.image = "img/cosmo.jpg";
```

Object Methods

When we add functions to an object, we refer to them as ***methods***. We use the keyword `this` when we want to access other properties inside the same object.

```
var attendee = {
  firstName: "Jane",
  lastName: "Smith",
  company:  "Apple",
  fullName: function() {
    return this.firstName + " " + this.lastName;
  }
}
```

Object Methods

Like all functions, when we call the method, we need to add the () after its name to tell it to run.

```
var attendee = {
  firstName: "Jane",
  lastName: "Smith",
  fullName: function() {
    return this.firstName + " " + this.lastName;
  }
}

console.log( attendee.fullName() ); /* outputs: Jane Smith */
```

Object Methods

Naturally, we can also pass data into our method from outside our object too.

```
var attendee = {
  firstName: "Jane",
  lastName: "Smith",
  greeting: function(salutation) {
    return salutation + " " + this.firstName;
  }
}

console.log( attendee.greeting("Hello!") ); /* outputs: Hello! Jane */
```

Does any of this look familiar?

It should...

You already know how to use objects and methods. You've been doing it for weeks now.

`$( ... )` is a method that creates an object —specifically, a jQuery object. It takes the string value you pass to it as a selector and finds all of the matches in the document and puts them in an object. That object has properties and methods that we have been accessing with dot notation, like `html()`, `text()`, etc.!

Working with Objects

Arrays & Objects

Go Build Awesome Things!