(More) Advanced JS Concepts

Learning Goals

- Create and interact with JS objects
- Utilize functions as methods on JS objects
- Understand the scope of `this` in JS
- Explain the difference between JS and jQuery
- Bind events to jQuery selectors & manipulate the DOM
- Research Array protoype methods

Warm Up

- How do you invoke a function in JavaScript?
- What is the equivalent of `document.querySelector('#important-information')` in jQuery? (take an educated guess if you are unsure)
- What does $ stand for when writing jQuery?
- In your own words, describe an `event` in JavaScript.

Overview

We've briefly touched on quite a few JavaScript concepts, now is our time to dive deeper and ensure we're all on the same page moving forward.

JavaScript Objects

What do we mean when we say "JavaScript object"?

An object in JavaScript is a collection of properties, and a property is an association between a name (or key) and a value.

A property's value can be a function, in which case the property is known as a method.

How do we create new objects in JavaScript?

First of all, you can define an object literal by simply creating what may look like a Ruby hash:

```
var penelope = { firstName: "Penelope", age: 88 }

penelope
// Object {firstName: "Penelope", age: 88}
```

You could also use the JavaScript keyword new.

```
var penelope = new Object()
penelope.firstName = "Penelope"
penelope.age = 88

penelope
// Object {firstName: "Penelope", age: 88}
```

How do we interact with our newly created objects?

```
var penelope = { firstName: "Penelope", age: 88 }
```

We can access an object's properties/values by using dot or bracket notation.

```
penelope.firstName
// "Penelope"

penelope["firstName"]
// "Penelope"
```

Reflect: Why would we want to encapsulate properties/functions in objects?

What else can we add to an object's properties?

FUNCTIONS!

```
var penelope = {
                 firstName: "Penelope",
                 age: 88,
                 sayHi: function() {
                   return "Hello!"
                 }
               }

 penelope.sayHi()
 // "Hello!"
```

What if I want to return a value dynamically depending on Penelope's properties?

```
var penelope = {
                firstName: "Penelope",
                age: 88,
                sayHi: function() {
                  return "Hi! I'm " + this.firstName
                }
               }

 penelope.sayHi()
// "Hi! I'm Penelope"
```

Within the scope of a function set as a property on an object, `this` refers to the object itself. More on this to come!

You Do

Define a variable called "pizza" in your console. Set it's value equal to an object that contains a property for "type" and "size." Give each property whatever value you'd like. Next, add a property that returns the object's details like so: "This is a [type] pizza that is [size] inches long."

Here's an example...

```
var pizza = ?!?

pizza.type
// Cheese

pizza.size
// 10

pizza.details()
// This is a Cheese pizza that is 10 inches long.
```

What is `this`?

At a high level, `this` is a special property in JavaScript.

Note: `this` is not only very hard to talk about in English, it's also a confusing to many new (and experienced) JavaScript developers.

The short version is that `this` refers to the context in which a function was invoked in JavaScript. Keep in mind that this is different from where it was defined.

Thankfully, we have a few rules to follow.

this refers to the global object in all global code

```
function logThis() {
  console.log(this)
}

logThis()
// global object
```

`this` refers to the parent object inside function code, if the function is called as a property of the parent.

```
var penelope = {
  name: "Penelope",
  whatIsThis: function() {
    console.log(this)
  }
}

penelope.whatIsThis()
// Object { name: 'Penelope', whatIsThis: [Function: whatIsThis] }
```

Lastly, `this` in function code invoked using the `new` operator refers to the newly created object. For example:

```
function Car (make, name) {
  this.make = make
  this.name = name
}

var jeepWrangler = new Car('Jeep Wrangler', ‘DangerZone’)

console.log(jeepWrangler.name)
// DangerZone
```

We’ll talk more about using constructor functions and classes to create objects in a future lesson!

A quick note: as you can see, `this` can be confusing and changes depending on the context we're in.

`this` can also be explicitly set using other JS methods like `apply`, `call`, and `bind`.

These methods are outside the scope of this lesson, but we encourage you to dive deeper if you're interested.

Small Group Discussion

- When do we want to leverage `this` in our code?
- What are some situations where having access to `this` may be helpful?
- In your own words, how would you describe `this`?

jQuery vs JavaScript

Recap: what's the difference between JavaScript and jQuery?

jQuery makes this..

```
var elements = document.getElementsByTagName("img")

for (var i = 0; i< elements.length; i++) {
  elements[i].style.display = "none"
 }
```

turn into...

```
$('img').hide()
```

What is jQuery?

- javaScript library (most popular)
- open-source
- easy DOM manipulation
- simple methods for reading and interacting with your HTML

What are some popular jQuery methods?

- `find()`
- `hide()` & `show()`
- `html()`
- `prepend()` & `append()`
- `on()`
- `css()`

Small Group Discuss

- Why would we want to use jQuery over raw JavaScript?
- When might we want to use JavaScript instead of jQuery?

Practice

In your notebooks, translate the following from JS to jQuery. If you're stuck, ask whomever you're sitting next to or the interwebs!

```
var clickMeButton = document.getElementById('click-me');

clickMeButton.addEventListener('click', function () {
  console.log('You clicked me!');
});
```

Array prototype methods

As you know, Ruby has many enumerable methods available to make your life as a developer more convenient.

Thankfully, JavaScript has some similar methods available for Arrays.

Array prototype methods

Research popular Array prototype methods and answer the following questions:

- What do you notice about these methods?
- How are they similar/different from Ruby enumerables?
- Describe what "prototype" means (take an educated guess or research!)

Recap: DOM manipulation

- What's an event?
- What's the DOM?
- How do we select an DOM element using jQuery?

DOM manipulation

1. Select an element
2. Bind an event to the selector
3. Execute some code that manipulates the DOM accordingly

Let's play around with some jQuery!

TuringSchool Examples - jQuery Playground - Basic jQuery

Spend 5 minutes refreshing on basic jQuery using this playground. Can you add a few events that trigger DOM manipulation? Can you change the font with the click of a button?

Additional Resources

- Calling functions in JS
- Event Basics
- Intro to OOJS
- Array Prototype Methods
- What is this?