GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/jsd6` repo to your computer
2. Open the `starter-code` folder in your code editor

ASYNCHRONOUS JAVASCRIPT AND CALLBACKS

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Store and use anonymous functions in variables.
- Pass functions as arguments to functions that expect them.
- Write functions that take other functions as arguments.
- Return functions from functions.

AGENDA

- Callbacks
- Immediately invoked function expressions (IIFEs)

Checkin and questions

- The **most significant thing I learned** about Ajax and APIs is _____.
- My **biggest outstanding question** about Ajax and APIs is _____.

Brainstorm: What are the different ways we can create and store functions?

Functions and callbacks

ASYNCHRONOUS PROGRAMMING

- Code that relies on input or behavior that might not be instantly available
- We use asynchronous programming to run code at different times

ANONYMOUS FUNCTIONS

```
var $yesButton = $('#yes-button');

$yesButton.on('click', function(event) {
  ...
});
```

FUNCTIONS ARE FIRST-CLASS OBJECTS

- Functions can be used in any part of the code that strings, arrays, or data of any other type can be used
- We can store functions as variables
- We can pass them as arguments to other functions
- We can return them from other functions
- We can run them without otherwise assigning them

HIGHER-ORDER FUNCTION

- A function that takes another function as an argument, or that returns a function

FUNCTION AS PARAMETER IN VANILLA JS

setTimeout()

```
setTimeout(function, delay);
```

`setTimeout()`

- A JavaScript function that lets you specify a function to run after a delay (in milliseconds)

setTimeout()

syntax

```
setTimeout(function, delay);
```

example

```
setTimeout(switchPage, 1000);
```

FUNCTION AS PARAMETER IN VANILLA JS

setTimeout() with anonymous function as argument

```
setTimeout(function(){
  console.log("Hello world");
}, 1000);
```

FUNCTION AS PARAMETER IN VANILLA JS

setTimeout() with named function as argument

```
function helloWorld() {
  console.log("Hello world");
}

setTimeout(helloWorld, 1000);
```

CALLBACK

- A function that is passed to another function as an argument, and that is then called from within the other function
- A callback function can be anonymous (as with `setTimeout()` or `forEach()`) or it can be a reference to a function defined elsewhere

EVENT LISTENERS

- A way of specifying a function that should run in response to an event
- Performs the same function as specifying a value for a property like `onclick`
- The difference:
  - An event property (like `onclick`) can take only a single value (you can do only one thing in response) for any given element
  - But you can set multiple values (functions) to run in response to a single event listener

EVENT LISTENER SYNTAX

```
element.createEventListener("event", function, false);
```

EVENT LISTENER EXAMPLE

```
var nextButton = document.getElementById(“#next-button”);
nextButton.createEventListener(“click”, switchPage, false);
```

equivalent to

```
var nextButton = document.getElementById(“#next-button”);
nextButton.onclick = switchPage;
```

Immediately-invoked function expressions

Immediately-invoked function expression (IIFE)

- A function expression that is executed as soon as it is declared
- Pronounced “iffy”
- Make a function expression into an IIFE by adding `()` to the end (before the semicolon)
- Make a function declaration into an IIFE by adding
  `(` at the start and
  `)()` to the end

IIFE based on a function expression

```
var countDown = function() {
  var counter;
  for(counter = 3; counter > 0; counter--) {
    console.log(counter);
  }
}();
```

IIFE based on a function declaration

```
(function countDown() {
  var counter;
  for(counter = 3; counter > 0; counter--) {
    console.log(counter);
  }
})();
```

Callbacks and IIFEs in practice

- Callbacks are a best practice for handling interface updates based on user interactions and/or data from web services
- Callbacks and IIFEs let us better organize our code
  - module pattern
  - we'll learn about this in a couple weeks

LEARNING OBJECTIVES - REVIEW

- Store and use anonymous functions in variables.
- Pass functions as arguments to functions that expect them.
- Write functions that take other functions as arguments.
- Return functions from functions.

NEXT CLASS PREVIEW

Advanced APIs

- Generate API specific events and request data from a web service.
- Implement a geolocation API to request a location.
- Process a third-party API response and share location data on your website.
- Make a request and ask another program or script to do something.
- Search documentation needed to make and customize third-party API requests.

Exit Tickets!

Q&A