Advanced jQuery

GENERAL ASSEMBLY

Learning Objectives

- **Explain** what a callback is
- **Describe** a JavaScript event
- **Talk about** what makes up a JavaScript event handler
- **Identify** some of the common JavaScript events
- **Explain** the parameter
- **Experiment** with timers in JavaScript
- **Create** basic animations with jQuery

Agenda

- Review
- Callbacks in JavaScript
- Timers in JavaScript
- Events with jQuery
  - The parameter
- Animations with jQuery

Review

GENERAL ASSEMBLY

Callbacks

GENERAL ASSEMBLY

What are they?

- Just a fancy name for JavaScript functions
  - Only difference is that you don't decide when these functions run
  - They are functions that act as a response
  - When ... happens, call this callback

It's all about callbacks

Hey Girl,
I wish I were an Event Listener...
...so you would give me a callback

Timers

GENERAL ASSEMBLY

Timers in JavaScript

- There are two main ways to work with time in JavaScript
  - You can set a **delay** with
  - You can set an **interval** with

What is the *window*?

- In JavaScript, you always have access to a special, predefined variable called
  - It defines what the browser has built in, and details about the browser itself
    - JavaScript APIs
    - Global Variables
    - Browser Dimensions
    - Way to access browser-related events

window.setTimeout

```
window.setTimeout( CALLBACK_FUNCTION, TIME_IN_MS );
```

```
function delayedFunction () {
  console.log( "I was delayed by a second" );
}

window.setTimeout( delayedFunction, 1000 );

// OR...

window.setTimeout( function () {
  console.log( "I was also delayed by a second" );
}, 1000 ); // Anonymous Function!
```

window.setTimeout

```
window.setInterval( CALLBACK_FUNCTION, TIME_IN_MS );
```

```
function intervalFunction () {
  console.log( "I am called every second" );
}

window.setInterval( intervalFunction, 1000 );

// OR...

window.setInterval( function () {
  console.log( "I am also called every second" );
}, 1000 ); // Anonymous Function!
```

Events

GENERAL ASSEMBLY

What are they?

- Every browser has events built-in
- Events are important moments that take place on a webpage
- We can attach functions (or callbacks) to these moments, and the browser will call them for us
- There are lots of events
    - Mouse events, window events, keyboard events, form events etc.

What is an *event handler*?

- An event handler is the way that we a callback function to an event in JavaScript
- They are made up of three important things
    - A target element
    - An event type
    - A callback function
    - e.g. When you a , call

How do they look?

```
$( ELEMENT ).on( EVENT_TYPE, CALLBACK_FUNCTION );
```

```
var $btn = $( "#btn" );

var eventType = "click";

function myButtonCallback () {
  console.log( "Button clicked" );
}

$btn.on( eventType, myButtonCallback );
```

How do they look?

```
$( ELEMENT ).on( EVENT_TYPE, CALLBACK_FUNCTION );
```

```
var $btn = $( "#btn" );

function myButtonCallback () {
  console.log( "Button clicked" );
}

$btn.on( "click", myButtonCallback );
```

How do they look?

```
$( ELEMENT ).on( EVENT_TYPE, CALLBACK_FUNCTION );
```

```
$( "#btn" ).on( "click", function () {
  console.log( "Button clicked" );
} );
```

So many ways!

- The more variables you have, the easier it will be to debug - I would start off defining everything
- Once you get more comfortable, you can start storing less
    - But I much prefer using named functions rather than anonymous functions (for debugging purposes)

The *event* parameter

- When JavaScript runs an event handler, it provides us with a little bit of information as a parameter
  - How long we have been on the page
  - Where the mouse was
  - What key was pressed
  - The target of the event
  - etc.
- We can call it whatever we would like, but the names and are very common

The *event* parameter

```
var $btn = $( "#btn" );

var eventType = "click";

function myButtonCallback ( event ) {
  console.log( "Button clicked", event );
}

$btn.on( eventType, myButtonCallback );
```

The *event* parameter

```
$( "#btn" ).on( "click", function ( e ) {
  console.log( "Button clicked", e );
} );
```

What types of events?

- We always create them in the same way, but there are:
  - Mouse Events
  - Keyboard Events
  - Browser Events
  - Form Events

Mouse Events

```
function myCallback () {}
function mySecondCallback () {}

$("p").on( "click", myCallback );

$("p").on( "dblclick", myCallback );

$("p").on( "hover", myCallback, mySecondCallback );

$("p").on( "mousemove", myCallback );

$("p").on( "contextmenu", myCallback );
```

Keyboard Events

```
function myCallback () {}

$("p").on( "keydown", myCallback );

$("p").on( "keypress", myCallback );

$("p").on( "keyup", myCallback );
```

Browser Events

```
function myCallback () {}

$(window).on( "resize", myCallback );

$(window).on( "scroll", myCallback );

$(window).on( "resize", myCallback );
```

Form Events

```
function myCallback () {}

$("input").on( "focus", myCallback );

$("input").on( "blur", myCallback );

$("input").on( "change", myCallback );

$("input").on( "select", myCallback );

$("form").on( "submit", myCallback );
```

Exercise!

Add some events!

- click
- resize
- keypress (to the body tag || an input)
    -
- mousemove (to the entire browser)
    -

Animations

GENERAL ASSEMBLY

Available Methods

- fadeIn
- fadeOut
- fadeToggle
- slideDown
- slideUp
- slideToggle
- animate

Animations

```
$( ELEMENT ).animate( CSS_OBJECT, TIME, ON_COMPLETE_CALLBACK );
```

```
$("img").animate({
    width: "500px",
    height: "500px"
}, 1000, function () {
    console.log( "Animation complete" );
});
```

Animations

```
$( ELEMENT ).animate( CSS_OBJECT, TIME, ON_COMPLETE_CALLBACK );
```

```
$("nav").animate({
    left: "0px",
}, 1000, function () {
    console.log( "Animation complete" );
});


$("nav").animate({
    left: "-200px",
}, 1000, function () {
    console.log( "Animation complete" );
});
```

Homework

- Your project!
- Try jQuery
- Code Academy: jQuery
- Learn jQuery
- jQuery Fundamentals

Next lesson

- More jQuery
  - Plugins

Q & A

GENERAL ASSEMBLY

Feedback Time

Lesson 14: **Advanced jQuery**

https://ga.co/fewd32syd

Thanks!

GENERAL ASSEMBLY