1. Course Overview

Contents

[1. Introduction 1](#_Toc27447)

[2. What Is an Event? 1](#_Toc260)

[3. Events on a Web Page 1](#_Toc24245)

[4. Demo: document ready 2](#_Toc26567)

[5. Demo: on and off 2](#_Toc10196)

[6. Demo: one 2](#_Toc20694)

[7. Demo: click 2](#_Toc27009)

[8. Demo: blur 2](#_Toc14042)

[9.Summary 2](#_Toc20849)

# 1. Introduction

=>slides: Pg. 1

When events happen – well, you want to be able to do something interesting in your applications. Let’s talk for just a moment about what events are so you have a clear picture as we get into the jQuery examples.

# What Is an Event?

=>slides: Pg. 2

An event is something that happens. When we think of events in the real world, you might have in mind events on your calendar

=>slides: Pg. 3

Between birthdays, anniversaries, meetings, conferences and much more, events are “things” that happen. Now it’s fairly easy to grasp the concept of a birthday, but how do events relate to web applications? Well let’s listen in on a conversation between a rockstar developer and his boss.

=>slides: Pg. 4

The developer is working on an application critical to the business and the boss wants to know as soon as possible when progress is being made on the project.

=>slides: Pg. 5

So the boss begins by asking the developer for updates on the project.

=>slides: Pg. 6

And the developer is happy to comply…

=>slides: Pg. 7

In fact he’s so enthusiastic about the project he begins to share every detail

=>slides: Pg. 8

Probably too much detail..

=>slides: Pg. 9

In fact the messages to the boss are starting to get to be a bit too much

=>slides: Pg. 11

To the point that the boss makes it clear that he only wants to hear from the rock star developer in the event that something interesting happens

=>slides: Pg. 12

Now the developer gets it and just notifies the boss when the application is ready for review.

=>slides: Pg. 13

And this is optimal because now the boss is only notified when something happens that is truly valuable to him.

This is a metaphor for how events work in software. There hundreds of thousands of “things” happening all the time in a software application, but as developers we just want to know about certain events. Common events in a web application include

# Events on a Web Page

=>slides: Pg. 14

Being able to know when [click] the page loads – or

[click] when a key is pressed on the keyboard – or

[click] (and this is a common one) when a button is clicked – or [click] when a HTML form is submitted

[click] when the window is scrolling [click] or when an element gets focus

Now this is just a list of 6 events you might care to handle in a web application, but there are many more available – this is just a sample.

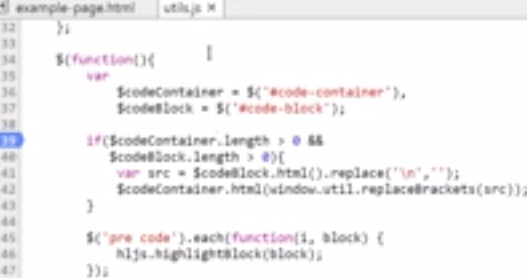
# Demo: document ready

=>slides: Pg. 15

The first event that I'd like to show you is the document ready event. Now, this is also referred to as the jQuery ready event, and I showed you this earlier in the course. I'll review it one more time here, just because it's such an important event to use. This is the event that you want to handle once the page loads, once the DOM is ready, so once all the elements on the page are ready to be interacted with, and so you'll use this over and over again in jQuery.



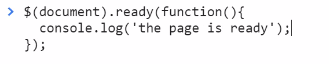
Here's what it looks like in code. So this is what the syntax looks like. Now, I can't just execute this statement here to show you the result, because this event fires when the page is loaded, but I did want you to have an opportunity to see the syntax just in its basic form one more time. Now to give you an idea of what this looks like in real life, let's take a look at the script that executes on this page as it loads. So let's take the console window, and move it down to the bottom of the page, and we'll just remove what's here because we won't be executing that.



And then take a look at the sources of this page. So here you can come in, and you can see that there's a JavaScript file that's referenced in this page called utils.js. Within utils.js, here you can see that same jQuery ready function being opened up, and then a number of different statements that are executed as the page is loaded.



I've set a breakpoint here on line 39, and so now as I refresh the page, I'll press F5 to refresh the page, and you'll see here that the breakpoint is hit. So again, once the page is loaded, then all of the code that I need in order to run on the page is found within the anonymous method that's inside the $ and parentheses. So I'll let this continue to execute, and now the page is done with its initial run.



Another way that you can write the exact same functionality is using this syntax. So here you'll notice that it's selecting the document, and then running the ready function, and then an anonymous function is being executed as a result of the ready function being run. So it's exactly the same thing as I showed you before, it's just this approach is a little more verbose. You'll notice that most people tend to go with a shorthand version of the jQuery ready function, and I think you'll probably find it works well in most situations as well. Next, let's take a look at handling specific events on the page.

# Demo: on and off

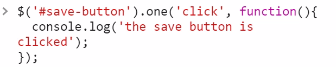
=>slides: Pg. 16

The first event that I'd like to show you is the document ready event. Now, this is also referred to as the jQuery ready event, and I showed you this earlier in the course. I'll review it one more time here just because it's such an important event to use. This is the event that you want to handle once the page loads, once the DOM is ready, so once all the elements on the page are ready to be interacted with, and so you'll use this over and over again in jQuery. Here's what it looks like in code. So this is what the syntax looks like. Now, I can't just execute this statement here to show you the result, because this event fires when the page is loaded, but I did want you to have an opportunity to see the syntax just in its basic form one more time. Now to give you an idea of what this looks like in real life, let's take a look at the script that executes on this page as it loads. So let's take the console window and move it down to the bottom of the page, and we'll just remove what's here, because we won't be executing that, and then take a look at the sources of this page. So here you can come in, and you can see that there's a JavaScript file that's referenced on this page called utils.js. Within utils.js, here you can see that same jQuery ready function being opened up, and then a number of different statements that are executed as the page is loaded. I've set a breakpoint here on line 39, and so now as I refresh the page, I'll press F5 to refresh the page, and you'll see here that the breakpoint is hit. So again, once the page is loaded, then all of the code that I need in order to run on the page is found within the anonymous method that's inside the $ and parentheses. So I'll let this continue to execute, and now the page is done with its initial run. Another way that you can write the exact same functionality is using this syntax. So here you'll notice that it's selecting the document and then running the ready function, and then an anonymous function is being executed as a result of the ready function being run. So it's exactly the same thing as I showed you before. It's just this approach is a little more verbose. You'll notice that most people tend to go with the shorthand version of the jQuery ready function, and I think you'll probably find it works well in most situations as well. Next, let's take a look at handling specific events on the page.

# Demo: one

=>slides: Pg. 17

Now in the previous clip, I showed you how you can handle an event using on, and remove that handling by using off, but what if you want to only deal with a certain event the first time it happens?



So let's do that and we'll start off with the same selector that we used before, and here what we'll do is use the one function. Now you have to be careful when you're typing this because it's only one character way from on. So here what we're doing is we're saying we want to handle the click event, but only one time. So as I execute that statement, I'll go ahead and click on the Save button, and notice that it logs it the first time that I click it, but as I continue to click on the Save button, nothing else is happening. So if you need to only have an event happen one time and one time only, then the one function is your friend. Now dealing with click events can get a whole lot easier by using a specialized function specifically for clicks. So let's look at that next.

# Demo: click

=>slides: Pg. 18

Now, when you're dealing with specific events, jQuery makes life a little bit easier by simplifying the API for you. So let's take a look at our button once again. And now, instead of using on or one or anything like that that's more general, what we can use is a specific function just for click. So here's what I can do is



type click, and now all I need to provide is the anonymous function that's run once the click happens. So you can see how the API is much more simplified here, because now I have a function called click, rather than passing it in as a string. So now when I go and click on the Save button, you can see that I'm logging out as many times as I'm clicking on that button. And click isn't the only event that gets that special type of treatment.

Over on jQuery's API documentation, you can see the list of all the events that it has special functions for. So if we scroll down the list a little bit, you can see that .click() is right there, but you can also use .dblclick(), .change(), .blur(), on and on. So all you need to know is the event that you need to handle, and you can look and see if jQuery has that specialized event handler already prepared for you. So let's take a look at just using one more of those items, the .blur() event method.

# Demo: blur

=>slides: Pg. 19

1. So as you can see that working with these event methods is actually quite easy.
2.   
   So if we set up the blur event method for the save‑button, now when I click on the button and then tab off of it, that fires the blur event from that element, and then it runs the function where it logs out no more love for the save‑button. So now that I've familiarized you with events in jQuery, let's wrap up this module and then I can talk to you about making your code even more expressive and more efficient as well.

# Summary

=>slides: Pg. 20

In this module I’ve shared with you the nature of events and how they relate to jQuery.

=>slides: Pg. 21

I covered the nature of events

=>slides: Pg. 22

I reviewed a number of different types of events in jQuery and how you can use them in your web applications.

=>slides: Pg. 23

The next module I’ll show you how to clean up your jQuery code and make it more efficient with chaining.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*