7. Working with User Input

Contents

[1. Course Overview 1](#_Toc18888)

### 1. Introduction

=>slides: Pg. 1

Closing the circle for an interactive website is allowing your web pages to accept user input. In this module I show you strategies for reading user input using jQuery. Please note as this is a “getting started” course, I don’t cover forms in the context of dealing with posting data to the server – for that information you’ll want to dive in deeper with one of the other jQuery courses here on Pluralsight. For now let’s start by working with textboxes and textareas

### Demo: Textboxes and Textareas

=>slides: Pg. 2

So the first input element that we'll work with is text areas and text boxes. I'm grouping them together because they're virtually the same when you work in jQuery with them. Let's take a look at the HTML, 

and as we look down here, you can see that the text area has an ID of notes‑box, so we'll use that in our selector. So I'll come over here and select that text area. Now the function that I'm using is val, which stands for value, this is a getter or a setter function, so I can provide a value by passing in an argument or I could call it just like this and get the value out.



So let's go ahead and add something in to begin with. So as I execute this, you'll notice down in the text area, it now has a value of these are my notes,



and so if I run the same function again without an argument, what we get back is the value out of the text area, and again, this works exactly the same for text boxes. Now there is some input elements that have some specialized rules so let's take a look at working with a date input type next.

### Demo: Date Picker

=>slides: Pg. 3

The next element that we'll work with is an input type of date. So once again, let's return to the HTML and look at the markup for this item.



So here, you can see that I have a standard input type, the type is set to date, and then it has an ID of showing date, so we'll use that within the selector. So let's go back over to the page itself, and then we can set up the selector like this. Now, once again, we'll be using the val function, but this one has some validation rules built into it because we're working with an input type of date. So first, let's try passing in just the string of test.



Now notice what happens here. We get an error bubbled up by jQuery that says the specified value of test does not conform to the required format of year, month, and day.



So let's try passing in a valid date now. We still have to pass it in as a string, so we'll pass in January 1 of 2015. And so as I evaluate that, when we look over at the input element, you can see that it shows that date. Now you can tell also that this is an input type of date because in Chrome this renders a nice little datepicker for us, it's got the spinner buttons, the X to clear it out, and everything that we'd expect from an input type that equals date



So since this is a getter and a setter function, once again, we can call it without any arguments. And when this is evaluated, it returns back a string of the date. So jQuery makes it a little bit easier for you when you're working with specific input types. The next one we'll take a look at is checkboxes.

### Demo: Checkboxes

=>slides: Pg. 4

The next input element that we'll look at is the checkbox.



So we have a checkbox down here for whether or not the owner is present as the person is filling out this real estate form. Let's, for a moment, take a look at the HTML.



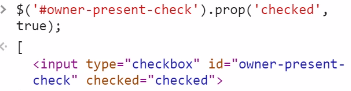
So here you can see is the input element, the type is checkbox, and the id is owner‑present‑check. Now, let's go back to the page for a moment, and let's inspect the element, come down here, and then what we can do is come to the input element,



and what we'll do is add an attribute of checked. So when we do that, you'll notice over on the running page the checkbox is now checked. So I've shown you the attribute function before, the attr function, so it seems to follow that that's the function that we would use in order to check a checkbox. So let's go ahead and give that a shot.



So I'll paste in the selector, and then we'll run attr for the attribute on the element, and we'll say checked and pass in a value of true. And just to make sure this test works correctly, I'll make sure to uncheck this checkbox. Now, when we run this, it should check the checkbox, right? You'll notice nothing happened. So if that command doesn't work, well, which one will? In fact, it's the property function, and that looks like this.



So here we're running p‑r‑o‑p for the prop function, and now you'll notice that the checkbox is checked. Now the reason this works like this is a bit out of scope of a getting started course. But basically, there's a representation in JavaScript of that checkbox, and that representation is in memory as a JavaScript object, and that object has a property of checked, which eventually influences how the HTML element is rendered on the page. But all you really need to know is that if you want to check a checkbox, use the prop function in order to make that happen. Now of course, this is a getter and setter function.



So in the same way, we can ask for the value of checked by calling prop as checked as the argument, and we get back true.



So we can turn it off by passing in a false. You'll see that it's not checked anymore.



And if we want to get its current value, we just call prop again and get back that value. So now let's take a look at radio buttons, which have its own little nuance as well.

### Demo: Radio Buttons

=>slides: Pg. 5



Now let's talk about radio buttons for a moment. We've got a couple of buttons down here at the bottom of this form. You'll notice that I can click on Immediate or Delayed, and I can only choose one at a time. And the reason it works that way is because in HTML there's a special way that you set up radio buttons so that they're grouped together. Let's take a look at the HTML to investigate this a little bit further.



This is the group here that has my radio buttons. So you notice, I have an input with a type of radio, and they each have a name and a value. You'll notice that the name is the same for each one of them. They're both set as available. They have a different value, so the first value is immediate, the second one is delayed, and they both have a unique ID value. So, the grouping that happens between these two radio buttons is by the value that's given for its name. So when we're using jQuery, what we conceptually want to do is to take a look at the group of radio buttons, and we'll access them by its name, and we want to find out which one of them is checked. Once we know which item is checked, then we want to get the value out of that checkbox. Let's take a look at what that looks like in jQuery. So what we're looking for is the availability status of this piece of real estate, and I'll take the value and set it aside in a variable. Then we'll create the selector for the checkbox that we're looking for. Now, we're going to look at the group and then ask for the item that's checked.

  
And to do this, we'll use a pseudo class. So first, we're looking for an input element with a specific attribute value. So we're looking for that input where the name is equal to available. And beyond that, we're looking for the item within the available input items that is checked. Now, once that's been selected, what we want to do is get the value from that item.

  
So as we evaluate this statement, we'll get the status back, and when we take a look at it, you can see that it's set to immediate.

  
So I can come down here, and I can change the status to delayed and then run my command once again, and you'll see it returns back to me delayed. So this is a little bit different than what we might be used to doing in jQuery where normally you just make basically a plain selection against the DOM and then call property or value or attribute or something like that. But in this case, it's different. Really, the selector itself is giving us the power that we need in order to find the selected radio button. And the pseudo class of checked is key in making that happen, in this case. Well, this gets you started with working with forms with jQuery and showing you some of the specific nuances that are involved in different input types. Let's go ahead and wrap up this module.

### Summary

=>slides: Pg. 6

In this module I showed you how to work with a number of different input elements in jQuery

=>slides: Pg. 7

You saw how the jQuery API differs among different input types

=>slides: Pg. 8

In the next module we’ll take a step back from the implementation details and look at jQuery at a more general level. Here you learn where do download jQuery, the differences among the different available versions and more

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