**Solution: Create an event listener**

Selecting transcript lines in this section will navigate to timestamp in the video

(upbeat music) - [Instructor] All right. Let's look at how I solve the Create an Event Listener challenge. When I take on a tricky challenge like this the very first thing I think about is what I want the HTML to look like in the end. So first, I went into the browser and messed around to build out the form that I wanted to inject. So here you can see that form. I placed it inside the list item with a class backpack strap. Then I set up a form, that has a class that indicates what side we're on, so left length and then the other class would be right length. Then I create an input inside, it's an input with a type number so it only receives numbers and nothing else. I give it a name, Left Length so that I can point at it if I need to, and I set up a place holder for this input so that it says new left length and new right length when nothing else is input. It's a placeholder so if I start typing, you can see the value goes away and if I take the value away then the placeholder comes back. Then there's a button, and it's just a button, it does nothing, it has no name or anything it just says Update 'cause I can target the button easily using standard JavaScript. All right, so this is what I need to build, a form for each of these items, and there's two items in each backpack. In the exercise files for this movie I've extensively commented everything that's going on to make it easier for you to see how all of this works. So if we go from the top and scroll all the way down past the huge template literal we've been working with before, you'll find here on line 116, I've set up a new item, that says, let strapLengths backpackArticle.querySelectorAll Backpack\_strap. So this will create a node list for me inside strapLengths, that has the two items, so both of these list items, that have the backpack strap lengths. Right now, they're just out putting strap length and nothing else. So what I want to do is grab the objects, so that I can inject more content into each of those objects and return them back. Then I've created a new function called newStrapLengths, and I pass this strapLength node list to that function. So that way I now have the two objects in question, the two list items, and I can work on them independently. This function, newStrapLength sits up here on line 46 and it captures the strap arrays, it's, the node list is still effectively an array, it works the same way. Inside the constant I've created an arrow function, and in the arrow function I start off by saying strapArray.forEach listElement so I'm immediately going into each of the items in the strap array. So now from here on forward, I'm working on listElement which is the element for each of the list items. Then for each of the list items I first use this list element getAttribute to get the data side attribute. Now remember data side is populated with either left or right telling me which side I'm on right now and I'll use that later on. So that's captured in the let called site. Then we create a new constant called lengthForm, this is the constant level will create the form. And I use document.createElement to create the form element. Then I immediately give lengthForm a class, and here I use side and the string length to create either left length or right length as the class. That will make it easier for me later to target just that elements if I need to. So it's always a good idea to just append classes whenever you can. And you can see here I'm using a template literal to mix the side value with some text. So now we have the form element, then we need to put something inside the form element. And here I'm just using innerHTML as we've done before with the template literal. In the template literal I set up an input with the type name, then I set the name to again, side, that's the value from up here. Length and the placeholder is new, and that I use side again, length. And that's what you see in the browser here, you see it says new left lengths and new right length. So left and right here are this side let that sits up here. So now the form element has an input and a button, then I can append an event listener to each of these forms. So now I can grab length form and add an event listener. And what I'm listening for is to submit event on the form itself. So when you click a button inside a form, it always submits that form. Now the tricky part is a form when you submit it, will try to reload the page. So the first thing I do is capture the event, then inside my arrow function I say e.preventDefault. All this does is prevent the browser from reloading the page immediately, so we're preventing the default behavior of the form. As a result of the form does nothing, so now we need to make the form do something. What I'll do is get the value from the form input so we set up a new let with the name, new value. Then we grab lengthForm, use the querySelector to find the input, that's the input up here. And then we grab the value of the input, so that will be whatever is inside that form element, whatever's inside this form element here. Now we have the value set us newValue. Then we use listElement, that's the current list item we were looking at. We use querySelector span to find the span. You'll remember span is what contains 53 inches in this case. hen we set span in our HTML equal to the newValue, so that is the newValue we got out of the form and inches. Then finally because we now no longer need the form input itself, I grab lengthForm.querySelector input again, so that's the form input. And then I set its value to nothing. That means when I type in a value here, 82 or 84 click Update, you'll see it clears out and then the value sits up here. Now that we've used the form, we've reset the value we exit our event listener, and the final thing we need to do is just append this new lengthForm, so the entire form to the listElement. Now remember the listElement is the list item, and by appending the form, we are adding it to the end of the item after all the other children. And that's why in the HTML here you see we have the list item, then we have the text, we have the span and then the form at the end. As I said in the setup to this challenge, there's nothing new here. And as you can see now, there really is nothing new, we're using the same methods as before, and the same methodology as before, is just that we're working with a form which looks a little bit different than other things, but works exactly the same way because as you now know, everything in JavaScript is an object, the document as a giant object, all the elements inside the documents are objects, and we can treat objects the same way no matter where we are and what we're doing.