**Advanced event listeners and "this"**

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

- [Instructor] Now that our event listener is working, we need to expand its functionality a bit. For one thing, the button text needs to be updated to reflect what the button actually does. Right now, it says open lid when the lid is closed. And when I click on it, it also says open lid when the lid is open. So it should really say close lid when the lid is open and vice versa. Secondly, we need to update the property for lid status in the object itself. So if that object was living in a database, and we want to pass that object back into the database to change the status, we can do that. This serves as a good example to highlight an important issue when choosing whether to use an arrow function or a function declaration as the event listener callback. Let me show you what I mean. In our event listener here, we are targeting the button element and then adding an event listener to it. So if I want to change the text inside the button element, I can do so by simply calling the button element inside the event listener saying, innerText and set it equal to something else like changed. And save that. Go back in the browser and click the button once. And now it says changed. All right. So it works, that means we can copy this ternary operator below here and just apply it. So let me say, button instead. Copy that three times. So that we're targeting the button and then we'll say, and then instead of just saying open, we say open lid and close lid and open lid. And the funny part is it doesn't actually matter what order you do this in inside the listener operator. You can say close lid, open lid, close lid and it will work the same way. 'Cause all we're doing is looking at the existing string and then swapping it for something else. And effectively just toggling it back and forth. So if we go here and look at this now, if I click open lid when it's closed, it'll now say close lid when it's open and vice versa. Okay, so the logic here works. Now I mentioned this whole arrow function versus function declaration. Right now, we're using an arrow function. You see it right here. And you remember when we talked about arrow functions previously, I said arrow functions don't have their own this. That comes into play here because ideally, we should be able to say instead of button, just say this. So this innerText, meaning this object that we are currently interacting with, and then just do that across here. This innerText, this innerText and this innerText. And that should work in theory, right? We're inside an object. But if you look closely, you'll see when I do this, we get this grayed out text behind it because the browser is saying, probably actually working but we can try it. So let's open the console and click the button. And when I do so it says, "Cannot read property innerText of undefined." What's happening here is this arrow function doesn't have its own this. So it doesn't know what object we're pointing at when we're using this. So if we want to use this inside an event listener, we need to change it to a function declaration. So we say function and then take away the arrow. Save. And now, everything works again because now we are working inside a proper object that has this. So we are able to point at this. So why am I showing you this? Well, once we get more advanced, we need to have the ability to control whether we want to use this or not. So if you scroll up to the top of this file, I've also added a new function up here called lid toggle. And that's because when we start working with a lot of different things that are happening, it doesn't make any sense to have the entire function sitting inside the article generator. You want to separate the function out. So here we can see we have a function declaration called lid toggle. Inside, we first set up a new lead called backpack objects. Then we grab the backpack object array. That's the array that has all the backpack information. And we use the find method to find a specific element. Here, we want to find the ID inside all of those elements. And we want to see if that ID matches the ID of the parent element of the button we clicked. So the article down here will have the ID for the current element. And then we're using that ID here. So we say the button, the parent elements of the button, which is the article. Find the idea of that article then put that in this ID and use that idea to see if you can find an element in the array that matches. That will give us the object in question. Then we can say backpack object. That's the array object we just found then set lid open to true. And then we're using a ternary operator. Then we do the same thing down here. Here's the, this innerText that applies to the button itself. And then finally here, we have this parent element query selector and then the backpack lid span. So that is the text inside the backpack lid status. And this also is just the ternary operator we've been working with before. So we can use this function instead as a callback function. To do that, we'll go down to our event listener and just wipe out all the content in site. And in place of the anonymous function, I'll just put in the name of the function we want to call, which is lid toggle. Now, like I said in the previous movie, this is a callback function. So we're not putting the parenthesis at the end here. We are calling the function back into the event listener and then running it in the event listener. And that's why this statement up here still works because this function is now effectively part of the event listener down here thanks to the callback. Save this. Run it in the browser, click a button. Everything is working properly. So now we've effectively created an advanced event listener separated the callback function out into a separate function. And we are using a function declaration to be able to access the this keyword and point at the button element.