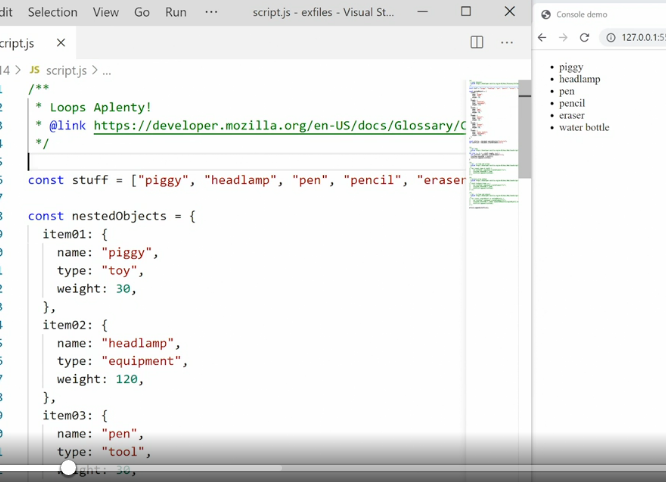
* - [Instructor] JavaScript gives us several different statements and methods for repeating a process over and over or looping through arrays and nested objects.
* Common to all of them is we first specify under what conditions the loop or iteration should run in the statement declaration.
* And then we specify what should happen at each iteration in the body of the statement.
* Let's look at some of the most common and useful iteration statements for arrays and objects.
* In the exercise files for this movie I've set up an array called stuff that has some items in it and a nested object called nested objects.



* That is an object with a bunch of nested objects inside it named item one, two, three, and so on.
* Now I want to loop through each of them and display that in the browser over on the right hand side, scrolling down, all we're doing is grabbing the article in the page then creating a URL and then using different loops to populate that URL.
* And then finally we append the list of list items to that URL so that it displays in the page.
* Off the top we use a **for loop** to loop through the array and displayed in the browser.



* The **for loop** is the classic and default way of looping through an array and it also allows us to loop through anything else.
* The for-loop always contains a statement like this inside that says, we first set up a let's name I.
* It could be named anything but it's typically I just increment and we set it to zero.
* Then we say for as long as I is smaller than some specified length, then run through the loop and then iterates I.
* What happens now is we'll keep iterating through this content again and again and again and I will keep increasing in size until we hit stuff length.
* Now to get this to work with an array what we do is use I inside the statement here.
* So we're saying stuff I, so we're starting with item number zero.
* That would be the first lot in the array.
* Then we iterate through that once.
* Then I is incremented up one.
* So now we have item number one it's still smaller than stuff length.
* So then we pass it down here, use it again and we keep doing that until we hit stuff length.
* So if you want to iterate through an array you use stuff length, that's the length property of the array to say when the iteration should stop.
* This for loop allows you to repeat a process as many times as you'd like and you just specify when you want it to stop.
* It is for anything.
* It's not just for arrays but you can see when you use it for arrays, it's a little bit clunky because you have to keep passing this I value down into the body of the loop to get everything to work properly, but it works.
* However, in modern JavaScript we have better tools.
* Down here below, we have a for of a loop.
* In the **for of a loop** we specify which array we're working with and then we say for as long as we have new items in this array, keep iterating through them.



* So we say for, then we create a const or a let's name the item or something similar of the array you want it to work with.
* So for long as we have an item in the array then do this and then we can keep looping through.
* And here we're just working with a single item.
* So you see, we don't need to say stuff I anymore, we're directly working with the item and we'll keep looping through until there are no more items available in the array.
* If you want to be even more direct you can also use the for each method.
* And this is the preferred way of dealing with arrays in modern JavaScript.
* The **for each method** applies directly to an array.
* It grabs each of the items and then it performs an action on those items in the form of a function.



* So here we pass an actual function using the item as the parameter and then we can deal with the item inside that function.
* **The difference between using a for each method and using the for of statement is in the for each method we are actively working on the item itself in functional programming.**
* We're using a function to do something with the item instead of just naming the item and then performing some actions around it.
* *Both the for each method and the for of statement, apply to arrays.*
* So what about objects? Down here at the bottom, we have a for in statement.
* The **for in** statement works exactly the same as the for of statements, except we are now working with objects.
* We're saying, for single object, in the objectig question in this case, nested object.



* So we are saying for as long as nested objects has more items, take the next item put it into this constant called single object and then do something with that constant.
* So single object becomes the item we're working with inside the loop then we'll do some action on it go up and say, are there more objects? If so, put them in single object, do a thing with them.
* Go up again.
* Are there more items? No.
* Okay.
* Stop the for loop.
* For objects we don't have a for each method or anything like it.
* That's because objects generally are structured differently than arrays, but using for in we can still loop through them.
* So now you have four different ways of iterating through arrays and objects, and you can experiment with all of them to see which one fits your work process the best.
* Right now the popular standard is to use for each methods for arrays and for in statements for objects.
* But these are just opinions.
* You can use the other approaches as well.