CS2350

Learning Activity 20

Arrays, Listeners, and Timers

Redo LA20 to put the index test right after index change:

index++

if(index > array.length) {

index = 0;

}

### Getting Started

To get started on this activity, do the following:

Download the following file and extract it.

[pa10files.zip](https://weber.instructure.com/courses/463779/files/87361845/download?wrap=1)

It contains the files needed for this activity.

### Add an Event Listener When the Page Loads

Create a .js file and store it in the folder with the .html file.

In the.html file add a script tag that links to the .js file. Replace name.js with the actual name of the file you just created.

<script src="name.js"></script>

In this new file, add a function called addListeners.

function addListeners(){

}

At the very top of the .js file add the code that will call this function when the page loads.

window.addEventListener("load", addListeners);

### Add an Event Listener When a Button is Clicked

Notice there is an empty 'p' element.

* Inside this 'p' element add a button.
* Set the id attribute to “next”.
* Put the word Next between the 'button' tags.

Note: You do NOT need the onclick attribute

After the addListeners function, add another function called changeImage.

function changeImage() {

}

Inside the addListeners function, use the addEventListener function to wire up the button in the .html file to the changeImage function. Have this event listener respond to the click event.

document.getElementById("next").addEventListener("click", changeImage);

### Changing Attributes With JavaScript

When the Next button is clicked, we want the image to change. Use JavaScript to change the value of the 'src' attribute. To do this, we have to give the 'img' tag an 'id' attribute.

Add an 'id' attribute to the 'img' tag. Set the 'id' attribute to "picture".

Remember when you use id it is to be unique on the entire web page so always choose a value for id that you have not used on any other element.

Once you have the id attribute set, you can access any HTML element by using the document.getElementById function. This function expects one argument of type string. This string is the value of the 'id' attribute of the element you want.

document.getElementById("picture")

Once you have the element there are several things you can set with it. One of the things you can set is the value of the 'src' attribute.

Inside the changeImage function add the following line:

function changeImage() {

document.getElementById("picture").src = "fatBoy.jpg";

}

### Using an Array

Currently this has limited functionality.

* Click the Next button again. Nothing changes.
* Refresh.
* Click the Next button again. The image changes.

Every time we call the changeImage function, it sets the 'src' attribute to fatBoy.jpg. If that is what it is currently showing, we do not see a change.

We don't just want to change the image once. We want to change it over and over again. To do this we need many image files. There are six image files in your folder. These will do.

We need to get the filenames so JavaScript can use them. To do this we are going to use an array. When defining a variable (even an array) in JavaScript use the keyword var followed by the variable name.

We will be writing several functions. Many of the functions will be using this array. To make it available to all the functions, put the array definition outside of any function. Towards the top of the .js file just under the line: window.addEventListener("load", addListeners); begin the definition of this array:

var images;

To make this an array you add = new Array( ). Inside the ( )s you can put the values you want in the array. We want the filenames of all our image files. These will be strings so put "s around them. Separate each filename with a comma:

var images = new Array("alligator.jpg", "beware.jpg", "fatBoy.jpg", "gatorEyes.jpg", "gatorGoing.jpg", "gatorRain.jpg");

We can use this array to move through each of the images. This function is to change the 'src' attribute of the picture just one time so a loop won't do. We just need to create a variable to hold the current index and increment each time the function is called.

Outside any function and just after the array definition, add a variable named index. Set it to 0. Like images this variable is defined outside of any function so it is a global variable and can be accessed by any function.

Inside the changeImage function increment index:

function changeImage() {

index++;

document.getElementById("picture").src = images[index];

}

Now each time we call this function, it will change the value of index and then it will change the 'src' attribute of our 'img' element.

Click the Next button several times and watch.

Each time we click the button, the changeImage function is called the value of index is incremented by 1.

If you keep clicking, index will become 6 and it will set the 'src' attribute to images[6] which does not exist. That is when you get the broken image sign.

Fix this inside the changeImage function by resetting index back to 0 once it gets to 6. You can do this with an if statement.

Click the Next button several times and watch.

* After you go through the first six pictures what happens? If you still get a broken image, make corrections until it works as expected.

We do not want to use the number 6 because that keeps us limited to always using an array with six elements. Instead we want it to work with however many filenames are in our array. JavaScript arrays have a value called length that reports how many elements are currently in the array.

Replace the 6 with images.length. Notice how this is a value, not a function call so you don't use ( )s.

Test to make sure your slideshow works as expected.

### Changing Content With JavaScript

We can change the following in HTML with JavaScript:

* attributes
* content
* css style
* elements

To change content we use document.getElementById("caption").innerHTML. This gets the element that has the id attribute set to caption. This is the paragraph at the end of the page. Then innerHTML will set what is between the opening and closing tags. Add a line inside the changeImage function that will change the content of this paragraph to "Hello World!".

document.getElementById("caption").innerHTML="Hello World!";

Click on the Next button to make sure the caption under the picture changes.

Outside any function create an array that contains six different strings. Each string will be used as a caption to one of the images.

Change the code so it uses this new array and index to set the content of the paragraph to a different caption each time it loads a new picture.

Click on the Next button several times to make sure the image and caption change together.

### Using a Timer & an Anonymous Function

Our slideshow works by clicking a button. Another approach is to have it change images after a certain amount of time.

To do this we will use a JavaScript function, setInterval. This function lets us tell it which function to call and how often to call it.

First create another button. Put this button in the same paragraph as the Next button.

* Set the id attribute to “play”.
* Put the word Play between the 'button' tags.

Now we need an event listener for this button that will respond to the click event. Use the addEventListener function again but this time we are going to use an anonymous function. An anonymous function is a function with no name. Instead of using a function name, we will put the entire function where the name would go:

document.getElementById("play").addEventListener("click", function(){ });

Inside the addListeners function, add the above line.

We will also need a variable that will keep track of a timer. In the .js file, create a global variable called timer.

Inside the anonymous function put a call to the setInterval function. This function requires 2 arguments. The first argument is a string that contains a function call. We want it to call our changeImage function so the first argument will be "changeImage()". The second argument is the number of milliseconds to wait between function calls.

document.getElementById("play").addEventListener("click", function(){

timer = setInterval("changeImage()",1000);

});

Notice how we create the variable outside the function and set the value of the variable inside the function. This is because we are going to use this variable in another function.

Click the Play button. Make sure the images and captions are changing automatically.

### Adding a Pause

Once we get a slideshow running, it is nice to be able to stop it.

Add one more button in the same paragraph as the other two.

* Set the id attribute to “pause”.
* Put the word Pause between the 'button' tags.

In the .js file, add another event listener that will respond to the click event on the Pause button. Have this event listener call a function called pauseSlideShow.

Outside of any other function, add a new function called pauseSlideShow.

Inside this function call the JavaScript clearInterval function. This function expects one argument. The argument is the variable used when calling setInterval. For us that is the variable named timer:

function pauseSlideShow(){

clearInterval(timer);

}

Click Play.

Click Pause.

Try these buttons a few times to make sure you can start and stop the slideshow when you want.

Now the play button is working almost correctly. There is still one problem.

* Refresh your page.
* Click the Play button.
* Click the Play button again. Notice how you have two timers changing the picture.
* Now click the Pause button.

Did the slideshow pause?

There is no way to pause the first slideshow. The first time you clicked the Play button a timed event was started and the information about the timed event was stored in the variable named timer.

When you clicked Play again, a new timed event was started and the information to this second timed event was stored in the same variable named timer.

The information for the first timed event is not stored anywhere, we do not have access to it.

When you clicked the Pause button, it stopped the timed event stored in the variable timer (the second timed event.)

Since no variable contains the information of the first timed event, there is no way to stop it.

Fix this problem. There are a couple ways to do this.

* You can test to make sure timer==null before you start a new timed event (in the anonymous function.) Then set timer=null after you call the clearInterval function in the pauseSlideShow function.
* Or you can always call the clearInterval function before you start a new timed event (in the anonymous function.) This will clear any existing timer before starting a new one.

Choose one of these approaches to fix the multiple timer problem.

### Adding a Previous Button

Now the only thing we are missing is a previous button.

Add another button, give it an id and put Previous between the opening and closing tags.

Now we need an event listener for this button that will respond to the click event. Inside the addListeners function, use a call to the addEventListener function to add an **anonymous** function to this new button.

Inside the anonymous function write code that will decrement index and change the image and the caption to the preceding image/caption in the array.

Be sure to handle the slide transition when the Previous button is clicked at the first image. At this point, the function is to display the last image and matching caption. Use the .length value so it doesn't matter how many values are in the array. Remember that the .length value is not a valid index value. It is one too many so you will need to -1.

Click on the Prev button several times to make sure the image and the caption change as expected. If not, make changes until it does.

In this activity we have used the load event and the click event to call a JavaScript function. There are many other events that can be handled. For a complete list of HTML DOM events go to https://www.w3schools.com/jsref/dom\_obj\_event.asp.