JSTutorial 3

The length of an Array: Finding out how many things are in an array…

Look at the following code. Can you tell what it does?

```
<!DOCTYPE html>
<html>
<head><meta charset= "utf-8" >
   <script >
       var picArray = new Array()
       picArray[0]= "safari1.jpg"
       picArray[1]="safari2.png"
       picArray[2]="safari3.jpg"
       picArray[3]="safari4.jpg"
       picArray[4]="safari5.jpg"

       function displaypic()
       {   var num = Math.floor(Math.random()*5)
           document.getElementById("pic1").src = picArray[num]
       }
   </script>
</head>
<body>
     <h1> Vacation Pics </h1>
     <p><img src = "Leopard.jpg" height = "300" width = "390"
          alt = "vacation pics" id = "pic1" > </p>
     <input type = "button" value = "Click for more pics" onClick = "displaypic()">
</body>
</html>
```

The code above is a web page with an image on it with the id 'pic1' and a button on it. When the button is clicked on, the function displaypic() is called and executed. In the function, a random number between 0 and 5 (not including 5) is generated. That random number is used to change the element on the web page with the id 'pic1' (the image) src (or picture) to whatever picture is stored in the array picArray at that random number.

How would we add a picture? Well, if you remember from previous tutorials, we can add a picture fairly easily by saying,

picArray[5]='safari6.jpg'

I could even write a simple function to add a picture:

```
function addpic()
{
     var x = prompt('Enter the name of a picture to be added to the array')
     picArray[5] = x
}
```

However, there is a problem with this. What if we want to add more than one picture to the array? What if we call this function more than once? Each time it is called, the user will be asked to enter the name of a picture they want to add to the array. But then, each time the new picture will be placed in the picArray at location 5, overwriting the picture that was there before. So with this function, at most the user can add only one new picture to the array. Yet we can call this function again and again. Most users would assume that each time they enter a new picture, it is added to the array as opposed to replacing the previous picture they added.

So how can we always add a picture to the end of the array, regardless of the number of items in the array? We need a way to find out the current number of items in the array. Luckily javaScript gives us a way to do that easily:

picArray.length

the .length method tells us the current length (the number of elements) in any array. To use it, you must say the name of the array, and add .length to it.

For example:

```
var myArray = new Array()
myArray[0]= "safari1.jpg"
myArray[1]="safari2.png"
myArray[2]="safari3.jpg"

var num = myArray.length
```

In the above code**, Num** now holds **3** because there are 3 elements in myArray.

Now let's write a function that adds a picture to the array:

```
<!DOCTYPE html>
<html>
<head>
      <meta charset= "utf-8" >
      <script>
            var picArray = new Array()
            picArray[0]= "Images/safari1.jpg"
            picArray[1]="Images/safari2.png"
            picArray[2]="Images/safari3.jpg"
            picArray[3]="Images/safari4.jpg"
            picArray[4]="Images/safari5.jpg"

            function addpic()
            {
                  var newpic = prompt("Enter new picture")
                  var num = picArray.length
                  picArray[num] = newpic
            }
      </script>
</head>
<body>
      <h1> Vacation Pics </h1>
      <p><img src = "Images/Leopard.jpg" height = "300" width = "390"
            alt = "vacation pics" id = "pic1" > </p>
      <input type = "button" value = "Click here to add a pic" onClick = "addpic()">
</body>
</html>
```

Now in the above code, whenever we click on the button that calls the function addpic(), the code prompts the user to enter a new picture. Then we find out the latest number of pictures in the picArray using picArray.length. Whatever that number is goes into the variable num. Because the number of elements in an array is always one more than the last location in the array (because we placed the first element at location 0), we can put the newest picture into the array at the location of num. For example, in the array picArray, above, the first time we use picArray.length, num will hold 5 because there are 5 pictures in the array picArray. Yet the last location address in the array is picArray[4]. So if we want to add a new picture to the end of the array, we would want to add it at picArray[5]. 5 is the number of elements in the array, or picArray.length.

Remember this code?

```
   <script >
        var picArray = new Array()
        picArray[0]= "safari1.jpg"
        picArray[1]="safari2.png"
        picArray[2]="safari3.jpg"
        picArray[3]="safari4.jpg"
        picArray[4]="safari5.jpg"

       function displaypic()
       {    var x = Math.floor(Math.random()*5)
            document.getElementById("pic1").src = picArray[x]
       }
   </script>
```

In this script, the function generates a random number between 0 and 5, not including 5. Then the picture in picArray at that random number's location will be displayed. This works fine, as long as the picArray only has 5 pictures in it. But if we start adding pictures to the array, the code will only display the pictures between locations 0 and 4. It will never display the new pictures we added to the end of the array.

To fix that, we again use picArray.length. This time we'll use it to get the length of the array before we generate a random number, and, instead of generating a random number between 0 and 5, we'll generate a random number between 0 and the length of the array picArray.

```
<!DOCTYPE html>
<html>
<head>
      <meta charset= "utf-8" >
      <script >
            var picArray = new Array()
            picArray[0]="Images/safari1.jpg"
            picArray[1]="Images/safari2.png"
            picArray[2]="Images/safari3.jpg"
            picArray[3]="Images/safari4.jpg"
            picArray[4]="Images/safari5.jpg“

            function addpic()
            {     var newpic = prompt("Enter new picture")
                  var num = picArray.length
                  picArray[num] = newpic
            }

            function displaypic()
            {     var num = Math.floor(Math.random()*picArray.length)
                  document.getElementById("pic1").src = picArray[num]

            }
      </script>
</head>
<body>
   <h1> Vacation Pics </h1>
        <p><img src = "Leopard.jpg" height="300" width="390" alt="vacation pics" id="pic1" >
              </p>
        <input type="button" value="Click here for more vacation pics" onClick="displaypic()">
        <input type="button" value="Click here to add a pic" onClick="addpic()">
</body>
</html>
```

Now we can add as many pictures as we want to the array using the addpic() function. Each time a new picture will be added to the end of the array, and the length of the array will increase by 1. Then, when we call the function displaypic(), it will always first determine the current number of elements in the array, and use that number to generate a random number between 0 and that current length, and that random number will be used to choose the picture from picArray that is to be displayed in the src of the image with the id of ‘pic1’.

Going through an array in order:

In the example above, we are able to click on a button and randomly see one of the pictures in the array of pictures. But what if we want to see the images in the array in the order in which they occur in the array, e.g., we want to see safari1.jpg first, then safari2.png second, then safari3.jpg third, etc.? Think of your vacation pictures. Probably you would want to see them in order – otherwise it might be hard to figure out where you were and what you were doing in the picture. The order of pictures can matter, and there are times when we want to go through things in order.

To do that, we need another variable. We’ll start this variable at 0. Then, each time we call the function and make the code inside of it run, we will increase the variable’s value by 1 and display the image in the array at that variable. This way we will see the pictures in the array in order.

```
<!DOCTYPE html>
<html>
<head>
      <meta charset= "utf-8">
      <script >
            var picArray = new Array()
            picArray[0]="Images/safari1.jpg"
            picArray[1]="Images/safari2.png"
            picArray[2]="Images/safari3.jpg"
            picArray[3]="Images/safari4.jpg"
            picArray[4]="Images/safari5.jpg"

            var count = 0    /* We’ll change this value later*/
            function displaypic()
            {     count = count + 1
                  document.getElementById("pic1").src = picArray[count]
            }

      </script>
</head>
<body>
   <h1> Vacation Pics </h1>
   <p><img src="Leopard.jpg" height="300" width="390" alt="vacation pics" id="pic1" > </p>
   <input type="button" value="Click here for more vacation pics" onClick="displaypic()">
</body>
</html>
```

A couple of things should be pointed out. First, the variable count is created outside and above the function it is used in. Why? Because if I put it inside the function, e.g.,:

```
            function displaypic()
            {     var count = 0
                  count = count + 1
                  document.getElementById("pic1").src = picArray[count]
            }
```
Every time the function displaypic() was called and the code was executed, the first thing that would happen is that the variable count would be set to hold 0. We don't want that. We want the function displaypic() to increase the count variable by 1 only, and then display the picture in picArray at the new count value. We don't want it to be set back to 0 each time we call it and the code runs. By placing var count=0 outside of the function, it will only happen one time, and not every time the function is called (because only code between the opening and closing { } happens when the function is called.)

The other thing that might look confusing is:

    count = count + 1

This is really the same thing as saying:

```
      var x = count + 1
      count = x
```

Or, our new variable x holds the value inside of count + 1. So if count is 0, x will hold 1. If count is 1, x will hold 2, etc. Then I'm setting the count variable to hold whatever is inside of x. So if x holds 1, count will now hold 1. If x holds 2, count will now hold 2, etc.

Another way to look at it is that we do the addition on the right side first. So count+1 gives us a number. Whatever that number is, it goes into the variable on the left side (regardless of the name of the variable. So given the following code:

```
            var count = 0
            function displaypic()
            {     count = count + 1
                  document.getElementById("pic1").src = picArray[count]
            }
```
The first time displaypic is called, the count variable has already been set to hold the value 0. So when we get to the line, count = count + 1, the right side of the equation, or count + 1, can be replaced with 0+1, or the number 1. That is the number that goes into the variable on the left. So now count will hold 1.

What have we created in total? Well, before anything happens, a variable count is set to 0. Then, every time the user clicks on the button on the web page, the function displaypic() is called, at which point the value inside of count goes up by 1. Then the src of the image with the id of 'pic1' on the web page is changed to the picture in picArray at the count value. Thus, each time the user clicks on the button, s/he sees the next image in the picArray.

Going back to the beginning…

So far, the code we've written will show the pictures in the array from first to last. But there's a problem. What happens when the count variable's value gets to 5? There's no picture at picArray[5]. What do we probably want to happen now? In most cases, we probably want to go back to the picture at the beginning of the array and start over. That will mean resetting the count variable to 0 when it gets to the location of the last element in the array. To do this, we can add an if condition to our code:

```
<!DOCTYPE html>
<html>
<head><meta charset= "utf-8">
   <script >
```

```
        var picArray = new Array()
        picArray[0]="safari1.jpg"
        picArray[1]="safari2.png"
        picArray[2]="safari3.jpg"
        picArray[3]="safari4.jpg"
        picArray[4]="safari5.jpg"

        var count = -1
        function displaypic()
        {   count = count + 1
            if (count >= picArray.length)
            {   count = 0
            }
            document.getElementById("pic1").src = picArray[count]
        }

   </script>
</head>
<body>
   <h1> Vacation Pics </h1>
   <p><img src="Leopard.jpg" height="300" width="390" alt="vacation pics" id="pic1"> </p>
   <input type="button" value="Click here for more vacation pics" onClick="displaypic()">
</body>
</html>
```

Now when we get to the end of the array of pictures, the count variable's value is set back to 0, and the picture displayed is the picture in the picArray[0]. Thus when we get to the end of the picArray, we loop back to the beginning.

Notice that outside the function I've set var count = -1, as opposed to how I had it previously with var count = 0. The reason for this is that when the function displaypic() is called, the first line of code that is executed increases the count variable's value by 1. That means if count = 0, then the very first time displaypic() is called, the count value changes to 1. After it has been changed to 1, the picArray[count] picture is displayed. That means that the first picture to be displayed and seen will be the picture in picArray[1]. But most likely we'd want to see the picture in picArray[0] first. To make that happen, I needed to start the count variable's value outside of the function at -1. Then the very first time the function displaypic() is called, and the first line in the function is executed, the count variable's value is increased by 1, which will make it 0. Then when the picture at picArray[count] is displayed, it will be the picture at location 0 in the array.

Adding pictures:

```
<!DOCTYPE html>
<html>
<head><meta charset= "utf-8">
   <script >
        var picArray = new Array()
        picArray[0]="safari1.jpg"
        picArray[1]="safari2.png"
        picArray[2]="safari3.jpg"
        picArray[3]="safari4.jpg"
        picArray[4]="safari5.jpg"

        var count = -1
        function displaypic()
        {   count = count + 1
            if (count >= picArray.length)
            {     count = 0
            }
            document.getElementById("pic1").src = picArray[count]
        }

        function addpic()
        {   var newpic = prompt("Enter new picture")
            var num = picArray.length
            picArray[num] = newpic
            document.getElementById("pic1").src = picArray[num]
        }
   </script>
</head>
<body>
   <h1> Vacation Pics </h1>
        <p><img src="Leopard.jpg" height="300" width="390" alt="vacation pics" id="pic1"></p>
        <input type="button" value="Click here for more vacation pics" onClick="displaypic()">
        <input type="button" value="Click here to add a pic" onClick="addpic()">
</body>
</html>
```

Can you see in the above code that using picArray.length everywhere instead of using the number 5 to represent the number of pictures in picArray allows this code to work properly and allows all the images to be displayed, regardless of how many pictures we add to the array?

Going Backwards

Finally, when you're going through your gallery of vacation pictures, you often want the ability to go backwards as well as forwards. You'd first need another button, representing the ability to go backwards through your pictures. You'd also need another function, that went backwards instead of forwards.

To go backwards, instead of increasing the value inside of the count variable, you'd want to decrease it by 1. So you'd have:

```
            count = count -1
```

Now when we were going forward, when we got to the last picture in the array, we went back to the beginning by setting the count variable to 0. But now we're going backwards. So when the count value gets lower than 0 (because there is a picture at picArray[0]), we want to set it to the last picture in the array's location, which is at picArray.length – 1 (remember, picArray.length gives us the number of pictures in the array, which is always 1 larger than the last location in the array). So the if condition would be:

```
              if (count < 0)
              {   count = picArray.length-1
              }
```

That's it. Now we have a function that will take us through your vacation slides backwards as well as forwards.

```
<!DOCTYPE html><html>
<head><meta charset= "utf-8">
   <script >
        var picArray = new Array()
        picArray[0]="safari1.jpg"
        picArray[1]="safari2.png"
        picArray[2]="safari3.jpg"
        picArray[3]="safari4.jpg"
        picArray[4]="safari5.jpg"

        var count = -1
        function displaypic()
        {   count = count + 1
            if (count >= picArray.length)
            {   count = 0
            }
            document.getElementById("pic1").src = picArray[count]
            document.getElementById("p1").innerHTML = count
        }

        function displaybak()
        {   count = count - 1
            if (count < 0)
            {   count = picArray.length-1
            }
            document.getElementById("pic1").src = picArray[count]
            document.getElementById("p1").innerHTML = count
        }

        function addpic()
        {   var newpic = prompt("Enter new picture")
            var num = picArray.length
            picArray[num] = newpic
            document.getElementById("pic1").src = picArray[num]
        }

   </script>
</head>
<body>
   <h1> Vacation Pics </h1>
        <p><img src="Leopard.jpg" height="300" width="390" alt="vacation pics" id="pic1"></p>
        <input type="button" value="Go Forward" onClick="displaypic()">
        <input type="button" value="Go Back" onClick="displaybak()">
        <input type="button" value="Click here to add a pic" onClick="addpic()">
        <p id="p1">Image number </p>
</body>
</html>
```