# Week 9: JS Part II

**Agenda**

* Introduction to JavaScript
  + More JS Concepts
    - Variable Scope
    - Arrays
    - Objects
  + Class Example
  + JS Challenges 2
* Next Week

## More JS Concepts

**Variable Scope**

In JavaScript, there are two major types of scopes for variables, local and global. Global variables are ones declared outside of functions while local variables are declared inside of functions. The difference is that global variables can be accessed anywhere in the document. Local variables can only be manipulated inside of their function. In the example below, x is a global variable and y is local to “myFunction()”…this means we can use the x value in and outside of the function. However, we can only refer to y inside of the function it was created in.

var x = 5; // Globally declared x variable

var myFunction = function() {

var y = 4; // Locally declared y variable

console.log(x); // X can be accessed inside this function

}

myFunction(); // Produces output

console.log(y); // Will cause an error because y is only defined inside of myFunction

There is also **block scope** which is when a variable is declared inside of a block of code. If that same variable name exists as a global variable, changing its value inside of a block will change the value of the global variable. In the example below, we declare x to be 5 and log it to the console. After that, we check to see if x is greater than 0, if it is, we redeclare it to be 4.

var x = 5;

console.log(x); // Will log 5

if (x > 0) {

var x = 4; // Redeclare our variable

}

console.log(x); // Will log 4 since the value has been changed

Sometimes this is what we want, however, there are cases where we want to use the same variable name inside of a block of code without changing the value of the global variable. We can do that by using the **let** keyboard. Instead of using var x = 4, we can use let x = 4.

var x = 5;

console.log(x); // Will log 5

if (x > 0) {

let x = 4; // Declare our variable using **let** instead of **var**

}

console.log(x); // Will log 5 this time since using let doesn’t affect our global variable.

In the example above, I used let instead of var to declare a variable. As such, the global variable isn’t affected even though I used the same variable name. For the most part, we will deal with function/global variables and won’t need let but it’s a good thing to understand.

**Arrays**

These are essentially lists of any kinds of data (which can include other arrays). These data pieces are separated by commas and placed inside of square brackets. Every item inside of an array has an index number which is used to refer to that item. For example, we might have an array of house pets:

var housePets = [];

The one above is empty and has nothing in it.

var housePets = [‘cat’, ‘dog’, ‘bird’, ‘chinchilla’]

The above array has **four** items, but the index number only goes up to **three.** This is because the first item in any array has an index number of 0. The values are listed below:

housePets[0]; // Refers to cat

housePets[1]; // dog

housePets[2]; // bird

housePets[3]; // chinchilla

You can change values at any point by referring to the item and then assigning it a new value:

housePet[3] = “fish”;

The array would now be housePets[‘cat’, ‘dog’, ‘bird’, ‘fish’]

Like strings, arrays have their own set of methods/functions which can be used on them.

Using housePets.length; would produce 4 for our case.

We can also use things like pop() or push() to remove or add items to our array.

housePets.pop(); // Removes last item in our array – becomes cat, dog, bird

housePets.push(‘cow’); // Adds cow to the end of our array – becomes cat, dog, bird, cow

**Objects**

These are things in JS that have their own properties and methods which can be called upon using something called **dot syntax**.

For example, if you have a car, that car has multiple characteristics and things it can perform. We can create an object for a car with the following code:

var myCar = {

name: “Herbie”;

owner: “none”;

age: 57;

honk: function() {alert(“HONK HONK!”);}

};

All you need to do is say var variableName = {} and place all methods/properties inside of the curly brackets. Name, owner, and age are all properties while honk is a method. We separate these with a colon but refer to them as key/value pairs. To get these values and use them, we use **dot syntax** which utilizes a period and the property/method name.

console.log(myCar.name); // Refers to the name key but logs the value “Herbie” to the console

myCar.honk(); // Would cause an alert to appear with “HONK HONK!”

myCar.age; // Refers to the age key and the value of 57

Objects in JS are powerful because almost everything is an object of some sort. Arrays, strings, and other types all have their own properties or methods (length, concat, push, pop, splice, etc.).

Additional Reading

* <https://www.w3schools.com/js/js_let.asp> (Let vs var)
* <https://css-tricks.com/javascript-scope-closures/> (Only up to “function hoisting”, this goes very in-depth)
* <https://www.htmldog.com/guides/javascript/beginner/arrays/> (Arrays)
* <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array> (Array Methods - ignore let for now)
* <https://www.htmldog.com/guides/javascript/beginner/objects/> (Objects)
* <https://www.geeksforgeeks.org/objects-in-javascript/> (More Objects)

**Class Examples:**

<https://iinf362.000webhostapp.com/examples/> (All examples for the course)

<http://iinf362.000webhostapp.com/examples/javascript-part-2/> (This week’s example)

Right-click this page in order to view the page source. I’ve left comments throughout the JavaScript to help you understand what is going on. These examples will help you with the challenges for this week’s assignment.

**If you haven’t done so already, please read the “Creating and Viewing our Web Pages.docx” file on Blackboard. You will not be able to submit anything for the assignment without completing that portion first.**

**JS Challenges 2**

Download the “JS-Challenges-2.zip” folder from Blackboard under the Lecture Notes for this week or in the Assignments folder. Inside of this zip folder, you’ll find an HTML file, a js folder, and a script.js file. Using these files, your task is to add JS to complete the challenges as they are described in the HTML file. In total, there are 4 main challenges. Feel free to reference the class example code as it has items that will directly help you with this assignment. Also, the very last slide of the JavaScript PowerPoint slides on Blackboard has a video which depicts how the output/results should be for this assignment. Feel free to watch it for assistance. If the download is too big (roughly 70MB), please let me know.

To receive credit for these challenges, submit a link to your JS Challenges 2 page by Wednesday, April 1st at midnight. The assignment with be called JS Challenges 2 in the assignments folder. It will also be accessible through the Lecture Notes for this week.

## Next Week (April 1st)

Next week, we will look at JQuery and the Initial Draft of the Final Project. We also have the CSS quiz due later in the week as well. The JQuery assignment will consist of a few short tasks. Other than that, your job will be to work on your Initial Draft and ask questions throughout the week. Make sure to look at the content provided in the Final Project Materials folder and think of ways to incorporate it into the page. The reading below isn’t mandatory but will give you a glimpse into JQuery.

* <https://api.jquery.com/>
* <https://www.w3schools.com/jquery/jquery_examples.asp>