**Interactive Development with JavaScript – Module 7**

**Arrays**

What are arrays?

Arrays are a type of variable designed to hold multiple values at the same time. An example might be an array of presidents such as George Washington, John Adams, Thomas Jefferson, James Madison, and James Monroe. To iterate through these to find a specific president would be difficult and require a lot of additional code when using String variables holding their names. Arrays are variable types designed to provide a simpler solution for accessing specific values.

JavaScript arrays are single dimensional, a linear type of variable value collection or list with a single location finding value. Unlike some other languages, JavaScript does not allow built-in multidimensional arrays. However, JavaScript does allow a multidimensional array type of a collection or list by defining an array of elements with each element being another array, in other words, an array of arrays. Because of this some will say JavaScript does support multidimensional arrays.

* Creating:
  + There are two basic procedures for creating arrays, an array literal and the “new” operator.
  + When creating arrays they may contain values or be empty.
  + Array Literal
  + Empty Array

var presidents = [];​

* + Array with Values, a more common approach
  + var presidents = ["George Washington", "John Adams",

"Thomas Jefferson", "James Madison", "James Monroe"];​

* “new” Operator
  + Arrays created using the new operator.
  + var presidents = new Array();
  + var presidents = new Array("George Washington", "John
  + Adams", "Thomas Jefferson", "James Madison" , "James

Monroe ");​

* Creating arrays using an array literal or the new operator do basically the same, so it is often suggested you use the array literal in place of the new operator. This is considered a simpler syntax that is more readable and will actually have a better execution speed.

**Elements**

Values held in arrays are referred to as elements. These elements may be accessed using the array name followed by square brackets "[ ]" containing an element’s position.

* Examples:
* presidents[0]
* presidents[1]
* Accessing the first array element
  + document.write(presidents[0]);
* Accessing the last array element
  + document.write(presidents[presidents.length - 1]);
* Accessing specific array elements
  + document.write(presidents[1]);
* Array index numbers start with 0, the first element, and ends with a value of the array’s length – 1. Example of an array of 10 elements will have an ending index value of 9. Incorrectly accessing an array element resulting from the index starting with 0 and ending with 1- length is commonly referred to as an off-by-one error.
* Altering element values
  + var presidents = ["George Washington", "John Adams",
  + "Thomas Jefferson"];
  + document.write(presidents);
  + document.write("<br />");
  + presidents[1] = "Abraham Lincoln";
  + document.write(presidents);
* Adding elements
  + There are basically three ways to add elements to an array, the first two examples are preferred.
  + “push”
    - var presidents = ["George Washington", "John Adams",
    - "Thomas Jefferson"];
    - document.write(presidents + "<br />");
    - presidents.push("James Madison");
    - document.write(presidents + "<br />");
    - Output
      * George Washington,John Adams,Thomas
      * Jefferson
      * George Washington,John Adams,Thomas
      * Jefferson,James Madison
* Length
  + var presidents = ["George Washington", "John Adams",
  + "Thomas Jefferson"];
  + document.write(presidents + "<br />");
  + presidents[presidents.length] = "James Madison";
  + document.write(presidents + "<br />");​
  + George Washington,John Adams,Thomas
  + Jefferson
  + George Washington,John Adams,Thomas
  + Jefferson,James Madison
* Assignment elements with index numbers, this approach should be avoided.
* var presidents = ["George Washington", "John Adams",
* "Thomas Jefferson"];
* document.write(presidents + "<br />");
* presidents[5] = "James Madison";
* document.write(presidents + "<br />");
* Output
  + George Washington,John Adams,Thomas
  + Jefferson
  + • George Washington,John Adams,Thomas
  + Jefferson,,,James Madison
* Removing elements
  + Elements contained in an array may be removed using the delete operator. The process does not shorten the array, it simply removes the elements as if it had never been assigned to the array as the length of the array will remain the same.
  + Example:
  + var presidents = ["George Washington", "John Adams",
  + "Thomas Jefferson"];
  + // Output 3
  + document.write(presidents.length + "<br />");
  + document.write(presidents + "<br />");
  + delete presidents[1];
  + // Output is still 3
  + document.write(presidents.length + "<br />");
  + document.write(presidents + "<br />");

**Warnings**

* JavaScript arrays allow elements to be added to an array using indexes beyond an arrays length leaving potentially empty elements (holes in an array).
  + Example:
  + var presidents = ["George Washington", "John Adams"];
  + presidents[30] = "Grover Cleveland";
  + document.write(presidents.length + "<br />");
  + document.write(presidents + "<br />");
  + Output:
  + 81
  + George Washington,John
  + Adams,,,,,,,,,,,,,,,,,,,,,,,,,,,,,Grover Cleveland
* You should never have a comma after the last element in an array declaration, as with some older versions of browsers this will throw an error.
  + Example:
  + var presidents = ["John Adams", "John Adams", ];

**Arrays and Objects**

In JavaScript arrays are best described simply as arrays, however in the JavaScript language arrays are a special type of an object. The one major difference is this gives arrays the ability to hold different data types in the same array.

Properties:

* length
  + Holds the number of elements in an array
  + Example:

presidents.length​

* + Sets/extends the number of elements in an array
  + Example:

presidents.length = 2​

* The following two properties are beyond the scope of this class.
  + Prototype
  + Constructor

**Methods**

We will discuss JavaScript methods/functions in more detail in the next module. Listed here are some of the Array class methods. Notice that length is not listed, as it is a property, not a method/function.

concat(), copyWithin(), entries(), every(), fill(), filter(), find(),

findIndex(), forEach(), from(), includes(), indexOf(), isArray(),

join(), keys(), lastIndexOf(), map(), pop(), push(), reduce(),

reduceRight(), reverse(), shift(), slice(), some(), sort(), splice(),

toString(), unshift(), valueOf()

**Arrays and Iteration**

Arrays may be accessed using all loops types, however the one considered the safest is the for loop. Here are two examples using both a for and while loop.

* The Array Code:
* var presidents = ["George Washington", "John Adams",
* "Thomas Jefferson", "James Madison", "James Monroe", "John
* Quincy Adams", "Andrew Jackson", "Martin Van Buren",
* "William Henry Harrison", "John Tyler", "James K. Polk",
* "Zachary Taylor", "Millard Fillmore"];
* Example 1 – for:
* document.write("<ul>");
* for(var i = 0; i < presidents.length; ++i){
* document.write("<li>" + presidents[i] + "</li>");
* }
* document.write("</ul>");
* Example 2 – while
* var i = 0;
* document.write("<ul>");
* while(i < presidents.length){
* document.write("<li>" + presidents[i] + "</li>");
* ++i;
* }
* document.write("</ul>");