**Interactive Development with JavaScript – Module 9**

**Creating/Writing Blocks of Code**

As a review, we remember functions as blocks of code designed to perform a specific action. Methods are functions designed to perform actions on objects. Let's see how to write both methods and functions.

**Functions**

Functions can be thought of as programs inside other programs, or we can think of them as subprograms. Similar to a program, a function is composed of statements. These statements are inside a block with a beginning and end. Functions can receive values, arguments, and return a value.

* Syntax:
  + function functionName(parameter list){
  + function body
  + };
* Example:
  + function returnHelloWorld(){
  + return "Hello World";
  + }

Using the previous example, we will now define the structure. The function definition starts with the key word “function”, that is then followed with a new, unique name, then a list of parameters from 0 to many, and lastly a block of code enclosed in curly braces. Inside the curly braces are the statements in the function to be executed. Statements inside a function may include all other types of statements such as selection, iteration, declaration, and the invoking (calling) of other functions. As we saw in the previous example, the function created took no parameters.

* Example that takes parameters:
  + function addTwoNumbers(value\_01, value\_02){
  + return value\_01 + value\_02;
* Invoking (calling) the addTwoNumbers function:
  + document.write(addTwoNumbers(10, 6));

// Output 8

* + document.write(addTwoNumbers(10, 6));

// Output 16

**Return Statement**

In the previous example there was a single return statement. Functions are not required to have a return statement unless the function is to return a value. Generally, we will see functions with a single return statement, however you may have multiple return statements. This logic gets into the questions:

* Is it a better practice to have a single return statement?
* Or, is it a good practice to return from a function as soon as possible?

To keep things simple, the answer I will give is yes. On a serious note, you will need to think about the logic you are using to get the most efficient functionality as well as the easiest function to maintain.

**Function Literal**

In a function literal expression, you will define an unnamed function.

* Syntax:
  + var funName = function(param\_1 , param\_2, …){
  + return result;
  + }
  + document.write(funName (argument1, argument2));
* Example:
  + var addTwoNumbers = function(value\_1, value\_2){
  + return value\_1 + value\_2;
  + }
  + document.write(addTwoNumbers(4, 5));
  + document.write(addTwoNumbers(22, 33));

**Functions Printing Arrays**

Functions can print all data types including arrays.

* + Example:
    - 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"];
    - function printPresidentsUL(names){
    - var list = "<ul>";
    - for(var i = 0; i < names.length; ++i){
    - list += "<li>" + names[i] + "</li>";
    - }
    - list += "</ul>";
    - return list;
    - }
    - document.write(printPresidentsUL(presidents));

**Methods: Basic Class Syntax**

We can create our own classes in JavaScript as the more current versions allow an advanced approach for constructing objects. Objects are constructed with both methods (functions) and properties (variables) with two syntax formats. Basic Class Syntax This type of class declaration is newer to the JavaScript programming language and is more of an ObjectOriented (OO) approach.

* Syntax:
  + class ClassName{
  + constructor(){ ... }
  + methodName1(){ ... }
  + methodName2(){ ... }
  + }
  + var instance = new ClassName(arguments …);
  + instance.methodName1()
  + instance.methodName2()
* Example:
  + class MyNewClass{
  + constructor(initParam){
  + this.classMember = initParam;
  + }
  + getClassMember(){
  + return this.classMember;
  + }
  + }
  + var instance = new MyNewClass("Hello World");
  + document.write(instance.getClassMember());

**Methods: Class Expression**

Another way to define a class is with a class expression. With this approach the class may or may not be named. When the class is named, the name is in the class body only.

* Example
  + var NewClass = class{
  + constructor(parameter){
  + this.parameter = parameter;
  + }
  + getClassMember(){
  + return this.parameter;
  + }
  + };
  + document.write(new NewClass("Hello
  + World").getClassMember());

**Methods: Properties**

Properties are values inside an object. These properties are for reading (getting), altering (setting), adding, and deleting.

* Syntax:
  + Basic syntax for accessing an object property is:
  + objectName.localProperty
* Example:
  + class myFirstClass{
  + constructor(value){
  + this.value = value;
  + }
  + }
  + var inst = new myFirstClass(777);
  + document.write(inst.value);

**Methods: Getters and Setters**

JavaScript, as do other languages, will allow shortcut notation such as getters and setters. These methods are designed to set and/or get a specific properties value.

* Syntax:
  + class NewClass {
  + get property(){
  + // Return value
  + }
  + set property(value){
  + // Set value
  + }
  + };
  + var instance = new NewClass();​
    - // Invoke setter
      * instance. property = 123;
    - // Invoke getter
      * instance. property
* Example:
  + class MyClass {
  + constructor(value){
  + this.value = value;
  + }
  + get localProperty(){
  + return this.value;
  + }
  + set localProperty(value){
  + this.value = value;
  + }
  + };
  + var inst = new MyClass(352);
  + document.write(inst.localProperty);
  + // Setter
    - inst.localProperty = 123;
  + // Getter
    - document.write(inst.localProperty);

[**Module 9 Programming Assignment (Click Here To Submit Your Assignment)**](https://cyberactive.bellevue.edu/webapps/assignment/uploadAssignment?content_id=_13970465_1&course_id=_513834_1&group_id=&mode=view)

**Assignment:**

For this assignment, create a function titled counter that takes an array of numbers as the parameter. The function is then to return an array holding the numbers of negative elements, zeros, and values greater than zero in the given array parameter.