Lab 11 – Code Testing

For today's lab we will be working with Jasmine, a JavaScript framework for testing your code. The scope of today's lab is fairly small and the code we will be testing rather simple, but this will hopefully provide you with a basic understanding for how you can test your own JavaScript code in future web projects. If you want to learn more, head over to the Jasmine Documentation.

Starter Files

You have been provided a zip folder (Jasmine Setup.zip) which contains:

```
Jasmine Setup/
        -lib/   (Jasmine Source Code – Don't Modify)
        -spec/
                -CalculatorSpec.js (Contains our Test Cases)
        -src/
                -calculator.html (Website)
                -Calculator.js
        -SpecRunner.html (Used to view & run test cases)
```

Quick Instructions

For this lab you will need to complete the JavaScript code to handle the math operations in Calculator.js and test the methods you created in CalculatorSpec.js.

1. Modify the Calculator object in Calculator.js to include a sub(), multiply(), and divide() method.

2. Create two test cases for sub(), multiply(), and divide() in CalculatorSpec.js.

Prototyping Methods in Javascript

```
function Calculator() {//Calculator Object

}

Calculator.prototype.add = function(value1, value2) {//Attach a function to our Calculator

  return value1 + value2;

};
```

In order for Jasmine to test our code, we need to work with Javascript Objects. In the above example, we are using the prototype property to add additional methods to our Calculator Class. In this way we can add methods for each of our math operations.

To access/use the method we created, we'll need to create a Calculator object:

```
var calc = new Calculator();//Line 14
```

With the calc object, we can now access the add method:

```
total = calc.add(total,value2);//Line 22
```

From this example, you'll now need to attach a method for subtraction, sub(), multiplication, multiply(), and division, divide().

For more information on Javascript Objects : https://www.w3schools.com/js/js_object_definition.asp

For more information on Javascript Prototype Property: https://www.w3schools.com/js/js_object_prototypes.asp

Creating Test Cases in Jasmine

In order to test our add method, we need to create a spec file (a javascript file within our spec folder). The spec file we will be working with is called CalculatorSpec.js which is located inside the spec folder.

Inside CalculatorSpec.js you will find a completed Test Suite and a Test Case for testing the add() function.

```
describe("Test add() method", function() {
   var calc;
   //This will be called before running each spec
   beforeEach(function() {
      calc = new Calculator(); //Create a calculator object
   });
   /*
   *  Example Test Case for Addition Operation
   */
  it("Check addition, two positive values", function() {
      expect( calc.add(1,2) ).toEqual(3);
   });
  it("Check addition, two negative values", function() {
      expect( calc.add(-7,-5) ).toEqual(-12);
   });
});
```

The **describe** key word defines a test suite. In Jasmine a test suite is a way to organize a series of test cases together. The String we pass will name the Test Suite.

beforeEach() will run before each of our test cases. We will use it to create a new/fresh Calculator object to test in each test case.

The **it** key word defines a test case. The String passed will name the Test Case.

Inside the test case, we have an assertion created with the **expect** keyword. Here we have the simplest assertion we can with work which simply verifies that two values are equal. He we test to ensure that 1 + 2 = 3, using the add() function.

From this example, you’ll need to create a new Test Suite for each of the remaining math functions. Inside each Test Suite you’ll need to create two test cases to test the math function.

Run Test Cases

To run your test cases, all you need to do is open the SpecRunner.html file. SpecRunner will run your test cases and inform you which ones have passed or failed.

Submission

For this lab you will need to submit a Zipped folder of your completed Jasmine Setup. When viewed, your SpecRunner.html file should show 8 test cases in total, that all pass!