**Hands-On Lab**

# JavaScript Unit Testing

Contents

[JavaScript Unit Testing 1](#_Toc410735071)

[Lab Introduction 3](#_Toc410735072)

[Creating JavaScript Unit Tests 4](#_Toc410735073)

[Task 1 – Download and Install Chutzpah 4](#_Toc410735074)

[Task 2 – Create and Run Simple Test 5](#_Toc410735075)

[Task 3 – Test External .js files 8](#_Toc410735076)

# Lab Introduction

In this lab you will learn the basics of doing unit testing for JavaScript. This lab assumes you are familiar with unit testing concepts using the VS 2012 testing tools. It also assumes you have a level of competency with JavaScript and JQuery.

**Objectives**

In this Hands-On Lab, you will learn:

* Create unit tests for JavaScript code
* Organize your JavaScript unit tests
* Manipulate DOM objects in unit tests

**Duration**

Suggested time to complete this Hands-On Lab is approximately 30 minutes.

**Setup – Starting Materials**

This Hands-On Lab includes the starting materials that are available under the root courseware folder on your machine. Open the folder **C:\Unit Testing\Labs\9\_JavaScript**

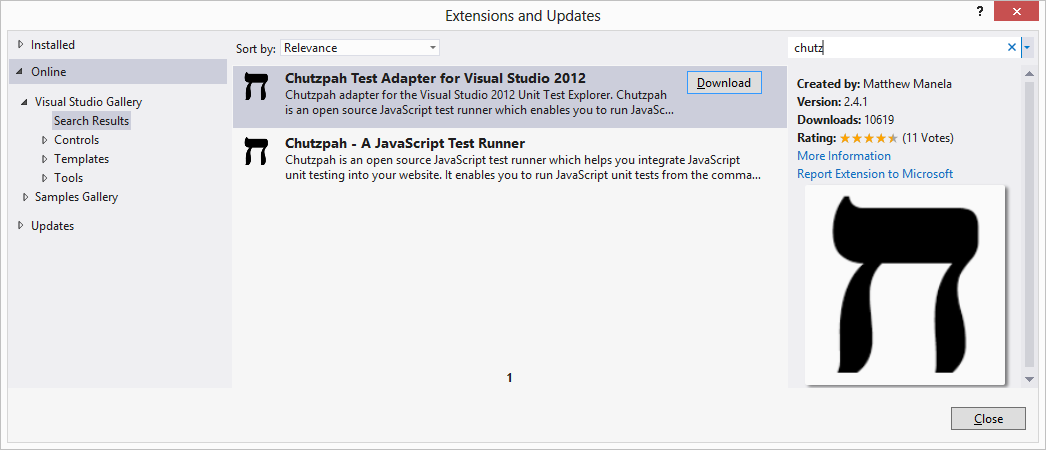
**Note**: This lab requires Visual Studio 2012 and an internet connection

# Creating JavaScript Unit Tests

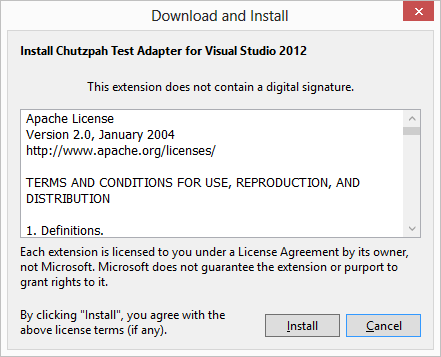
JavaScript unit testing is not natively supported by VS 2012. You will need to use NuGet to download and install the testing framework and runner.

## Task 1 – Download and Install Chutzpah

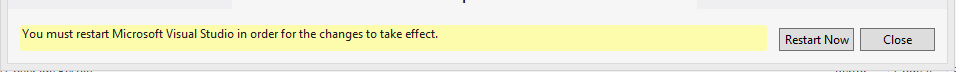
1. Open Visual Studio 2012
2. Open solution file in the Root > Begin folder. You should see two projects in the solution, SampleWebFormApplication & SampleWebFormApplication.UnitTests
3. Click on Tools > Extensions and Updates
4. Select “Online” in the left navigation and then enter “Chutzpah” in the search box



1. Click download on “Chutzpah Test Adapter for Visual Studio 2012”. When prompted click the install button on the Download and Install dialog box.



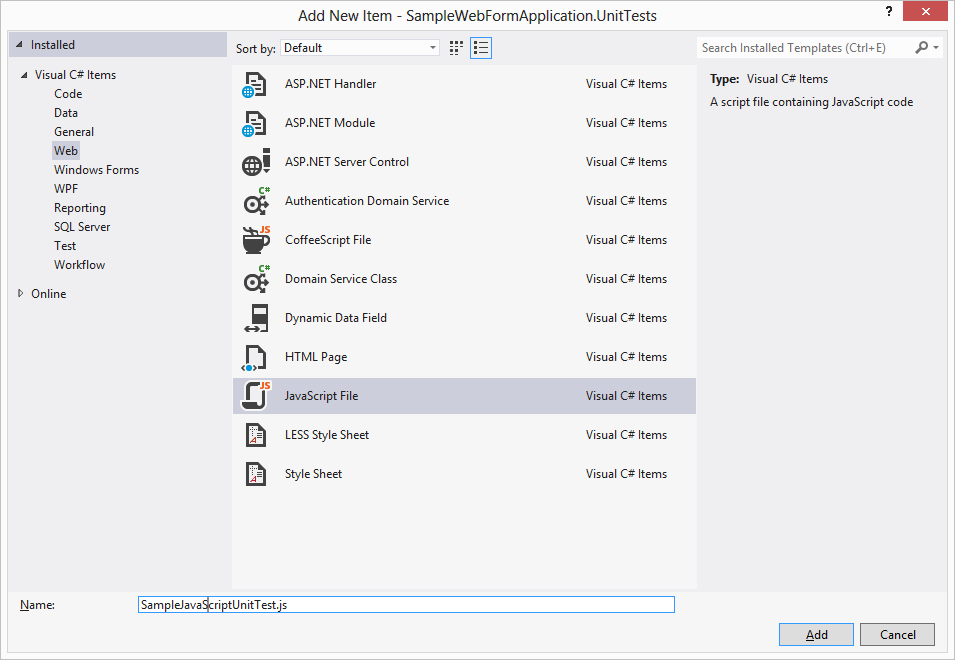
1. Once the install completes Visual Studio will prompt you to restart



## Task 2 – Create and Run Simple Test

Now that the Chutzpah Test Adapter is installed we can start writing unit tests for JavaScript.

1. Right click on the **SampleWebFormApplication.UnitTests** project > Add > New Item. Select the JavaScript file and name the file “SampleJavaScriptUnitTest.js” and click the Add button.

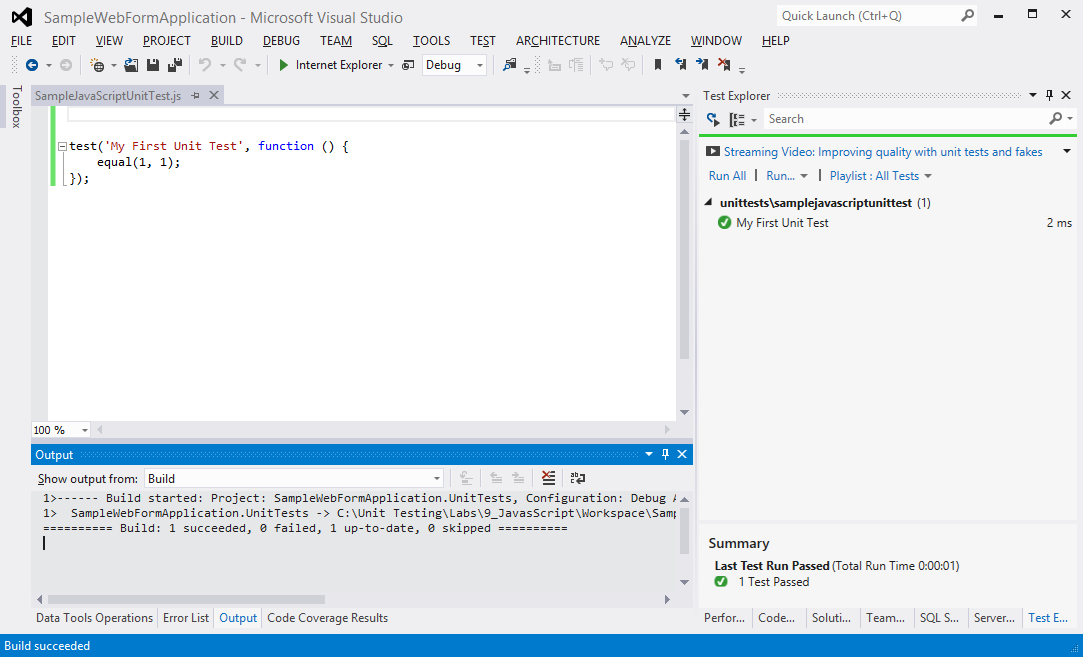


Now we can add code and create a unit test.

1. In the code window enter the following code:

|  |
| --- |
| test('My First Unit Test', function () {  equal(1, 1);  }); |

1. Save the file and open your Test Explorer to see the test in your list of active unit tests. You can now run the test by click on the Run All link.



1. Now let’s refactor the test code to include a module so we can better organize our unit tests. Modify your test so that it looks like the following code:

|  |
| --- |
| module('Module Name');  test('My First Unit Test', function () {  equal(1, 1);  }); |

Notice how the VS Test Explorer now displays the test. The string in the module is used as a way to organize your different tests.

|  |  |
| --- | --- |
| Before | After |

1. Finally, let’s modify the code to include our setup and teardown routines as well as safe guard it against any potential scope problems.

|  |
| --- |
| (function () {  module('Module Name', {  setup: function () {  //setup code  },  teardown: function () {  //teardown code  }  });  test('My First Unit Test', function () {  //arrange  //act  //assert  equal(1, 1);  });  })(); |

## Task 3 – Test External .js files

In the SampleWebFormApplication project you will notice a ShirtsAndPants.aspx and ShirtsAndPants.js file in the Samples/ColorCordinator directory. The .js contains the logic needed by the .aspx file. The .js file is what we will focus on creating tests for to ensure it is working correctly.

The.js file contains the logic that takes a pants color and recommends a shirt color. The .aspx page then displays that recommendation on the screen.

1. Open the “SampleJavaScriptUnitTest.js” file in the SampleWebFormApplication.UnitTests project and copy the following code so it looks like the following:

|  |
| --- |
| /// <reference path="../samplewebformapplication/samples/colorcordinator/shirtsandpants.js" />  (function () {  module('RunColorCordinatorForPants', {  setup: function () {  // setup code  },  teardown: function () {  // teardown code  }  });  test("Pink Expected Red", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("red");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "pink");  });  test("Blue Expected Green", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("blue");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "green");  });  test("Green Expected Blue", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("green");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "blue");  });  test("Pink Expected Yellow", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("pink");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "yellow");  });  test("Black Expected White", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("black");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "white");  });  test("White Expected Error", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("white");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "error");  });  test("Empty Expected Error", function () {  //arrange  var objColorCordinator = new ColorCordinator();  //assert  objColorCordinator.RunColorCordinatorForPants("");  var result = objColorCordinator.shirtcolor;  //act  strictEqual(result, "error");  });  })(); |

Note that at the beginning of the code you have path references to the ShirtsAndPants.js file.

|  |
| --- |
| /// <reference path="../samplewebformapplication/samples/colorcordinator/shirtsandpants.js" /> |

This tells the test runner to reference the correct .js file(s) that you are writing your test for.

1. Save your file and now you can successfully execute the tests.

