**Demo #1 – ECMAScript 5 Array Demo**

Purpose is to open the link below and show the ECMAScript5 array performance features.

Open in each browser and compare.

* Short Link: <http://on.csell.net/es5demo>
* Long Link: <http://ie.microsoft.com/testdrive/HTML5/ECMAScript5Breakout/Default.html>

**Demo #2 – Fundamental’s**

Fun with Variables:

Steps:

1. Create a var without var

2. Create a var with a var

3. Create a function that takes two args.

3a. In the function set one of the arguments to the global var created earlier

3b. Return the addition of the two arguments

4. Write out each of the vars from their inital state

5. Call the add function and write the result

6. Write out the var's again

Demo Code

a = 1;

var b = 2;

var add = function (num1, num2) {

a = num1;

return num1 + num2;

}

write(a);

write(b);

write( add(3, 4) );

write(a);

write(b);

**Fun with Functions**

Steps:

1. Create a variable foo as a global with some string.

2. Create a function that gets involved immediatly. this function should have the name foo.

3. Call Alert with the argument and with the var foo

4. Change the function name such that it doesn't collid

5. Take the function name out.

6. OverWrite Alert.

Sample Code

window.alert = function () {

console.log(arguments[0]);

};

var foo = "my message";

(function foo (someMessage) {

alert("Function Argument: " + someMessage)

alert("In function: " + foo);

})(foo);

alert("At End: " + foo);

**Demo #3 – Closure**

Simple Closure: Purpose is to create a set of nested var's and functions where the inner most function prints the values from the outer functions.

Finished Code:

var a = "a";

( function () {

var b = "b";

( function () {

var c = "c";

( function () {

write("a: " + a);

write("b: " + b);

write("c: " + c);

} ) ();

})();

})();

Real World Closure: Purpose is to create an array of items of which one of the objects in the array is a function. Loop through the array and use jQuery to bind the items in the array to the buttons on the page. Use the button, therefor executing the function that was originally in the array.

var arrayOfStuff = {

items: [{

id: '#getData',

click: function () {

alert('the GetData button');

}

},

{

id: '#doSomething',

click: function () {

alert('the doSomething button');

}

}]

};

(function (stuff) {

$.each(stuff.items, function ( key, value ) {

$(value.id).bind('click', value.click);

//$(value.id)[0].attachEvent('onclick', value.click);

});

})(arrayOfStuff);

**Demo #4 – Automated Testing**

Purpose is to just show how the test runner works.

1. How do you include it?
2. What does the page body look like
3. How do you write a test

Sample Test

test('testName', function() {

ok(true);

equal( 1, 1, "some message" );

notEqual( 2, 1, "some message" );

});

**IE9ify Script**

Purpose is to go from nothing to pinned in less than 10 minutes. This demo will

* Add Jumplists
* Add a custom color
* Add the appropriate icons
* Add the ability to pin a piece of content on the site
* Add a pin teaser to the site
* Check to see if the site is pinned
* Create ThumbBar buttons
* Show ICON Overlays

**Steps**

1. Run the default PhotoGallery showing off the basic functionality of pinning.
2. Install jQuery.ie9ify from the NuGet Package Manager in Visual Studio
3. Show the ie9ify file being added to the script folder
4. Add the ie9ify script file to the head
5. Add the default ie9ify block
6. ***Run*** 
   1. ***showing the basic setup of JumpLists***
   2. ***Color***
7. Pin the an image - ie the logo
8. Add the Pin Teaser
9. Add the block checking to see if things are pinned
10. Use AJAX to get a dynamic set of items for a jumplist
11. Add the ThumbBar buttons for the hover mode
12. Add the comment overlay
13. Run the site.
    1. Login and add a comment.
    2. See if get overlayed.