

Professional JavaScript

Lee Brandt

Who **is** I?

- Drinker of Beer
- Eater of Villagers
- Star of Finding Bigfoot
- Keeper of Keys
- Very Serious
- Very Professional
- Lee Brandt
- Director, R&D at PaigeLabs
- Coder
- .Net for 15 Years
- JavaScript (hardcore) 3 Years
- iOS (once upon a time)

iDoSomethingAwesomeJS

Agenda

- Some Basics
- The Compiler
- Scope
- Closure
- Prototypes
- Design Patterns
- Asynchronous
- Strict Mode

Editors

Object

```
var myObject = {  
  key: '1',  
  newKey: ['1', '2', '3', '4'],  
  nutherKey: {  
    key: 2  
  },  
  yak: function() {  
    console.log('Ralph!');  
  },  
  barf: function() {  
    return 'Puke!';  
  }  
};
```

```
console.log(myObject.key);  
console.log(myObject.newKey[3]);  
console.log(myObject.nutherKey.key)  
;  
myObject.yak();  
console.log(myObject.barf());
```

Function (objects)


```
function myFunction() {  
    var myVal = 0;  
    var mySum = 2 + 2;  
}
```

JavaScript Compiler

```
var john = 'Daniels';
```

```
function Sheldon(){  
  var spot = 'mine';  
}
```

```
function Leonard(girlfriend){  
  girlfriend = 'Penny';  
  inhaler = true;  
}
```

hoisting

```
console.log(foo);  
console.log(bar);
```

```
var foo = 'baz';  
var bar = 'bam';
```

```
console.log(foo);  
console.log(bar);
```

```
var foo;  
var bar;
```

```
console.log(foo);  
console.log(bar);
```

```
foo = 'baz';  
bar = 'bam';
```

```
console.log(foo);  
console.log(bar);
```

```
console.log(foo);  
console.log(bar);
```

```
function foo(){  
  console.log('I am in foo');  
}
```

```
var bar = function bar(){  
  console.log('I am in bar');  
}
```

```
function girl(){  
  console.log('No, you hang up.');
```

boy();

```
}
```

```
function boy(){  
  console.log('No, you hang up.');
```

girl();

```
}
```



```
function girl(){  
  if(dad.cantTakeItAnyMore){  
    return dad.hangUp();  
  }  
  console.log('No, you hang up.');
```

boy();

```
}
```

```
function boy(){  
  console.log('No, you hang up.');
```

girl();

```
}
```

||FE

```
var SimpleMan = (function(me) {  
  
    this.logMe = function() {  
        console.log(me + ' is a simple man. ');  
    };  
  
    return this;  
  
})('Ronnie Van Zant');  
  
SimpleMan.logMe();
```

The Face
(function(){})();

```
(function(){  
    'use strict';  
  
    var myController = function(){  
        var ctrl = this;  
  
        ctrl.loadMeSomething = function(){  
            ctrl.something = 'SHOTZ!!';  
        };  
  
        return ctrl;  
    };  
  
    angular.module('myModule')  
        .controller('MyCtrl', [myController]);  
})();
```

Prototype

Inheritance

Inheritance Delegation

proto

this.isCrazy()

call(me)(maybe)

```
function clickHandler(e) {  
    console.log(this === e.currentTarget);  
}
```

bind()

```
function bindDemo() {  
    function f() {  
        return this.a;  
    }  
    var g = f.bind({a: "azerty"});  
    var h = f.call({a: "qwerty"});  
  
    main.log(g());  
    main.log(h);  
}
```

Scope

Closure


```
function add(number) {  
    counter = 0;  
    console.log(counter += number);  
}
```

```
for (var i = 0; i < 10; i++) {  
    add(1);  
}
```

```
var add = (function() {  
    var counter = 0;  
    return function(number) {  
        main.log(counter += number);  
    };  
})();
```

```
for (var i = 0; i < 10; i++) {  
    add(1);  
}
```

Patterns

```
var Vehicle = function(){
    this.drive = function(){
        console.info('I\'m driving');
    };
    this.stop = function(){
        console.info('I\'m stopped');
    };
};

function DodgeTruck(){
    this.make = 'Dodge';
    this.model = 'Ram 1500';
    this.numberOfDoors = 4;
};

DodgeTruck.prototype = new Vehicle();

var myTruck = new DodgeTruck();

myTruck.drive() // I'm driving
myTruck.stop() // I'm stopped
myTruck.make // Dodge
myTruck.model // Ram 1500
```

Module

```
var CarService = function(){  
  return {  
    pickUp: function(){  
      console.log('Picking you up.');    },  
    dropOff: function(){  
      console.log('Dropping you off.');    }  
  };  
};
```

```
var myCarService = new CarService();  
myCarService.pickUp();  
myCarService.dropOff();
```

Revealing Module

```
var Killer = function(weapon){
    var stabEm = function(){
        console.info('Stab. Stab. Stab.');
```

 },
 shootEm = function(){
 console.info('Pew. Pew. Pew.');
 }
 killEm = function(){
 if(weapon==='knife'){
 return stabEm();
 }
 if(weapon==='gun'){
 return shootEm();
 }
 console.info('Sorry, I can\'t kill ya today.');
 };

 return {
 KillEm: killEm
 }
};

```
var Psycho = new Killer('knife');
Psycho.KillEm();
```

```
var Sniper = new Killer('gun');
Sniper.KillEm();
var NormalPerson = new Killer();
NormalPerson.KillEm();
```


Always use

Strict Mode

```
function strictDemo(){
    'use strict';

    var obj = {firstName: 'Jim', firstName: 'Davis'};

    var x = 17;
    var obj = { x: 10 };
    with(obj){
        x = 5;
        y = 10;
    }

    eval('Object.prototype = {}');
}
```

```
'use strict';

function strictDemo(){

    var obj = {firstName: 'Jim', firstName: 'Davis'};

    var x = 17;
    var obj = { x: 10 };
    with(obj){
        x = 5;
        y = 10;
    }

    eval('Object.prototype = {}');
}
```

Asynchronicity



Why So **Asynchronous**?

```
function getMeSomething() {  
    var returnValue = someAsyncCall();  
    return returnValue;  
};
```

```
function whenYouFinish(result) {  
    return result;  
}
```

```
function getMeSomething() {  
    return someAsyncCall(whenYouFinish);  
}
```

```
function getMeSomething() {  
    return someAsyncCall()  
        .then(function(result) {  
            return result;  
        });  
}
```



```
function getMeSomething() {  
    return someAsyncCall()  
        .then(  
            // success  
            function(result) {  
                return result;  
            },  
            //error  
            function(err) {  
                console.log(err.statusText);  
            })  
        );  
}
```

```
function someAsyncCall(){
    var deferred = $q.defer();

    if(iHaveSomething){
        deferred.resolve(someResult);
    }else{
        deferred.reject({statusText: 'I got nothing.'});
    }

    return deferred.promise();
}

function getMeSomething(){
    return someAsyncCall()
        .success(function(result){
            return result;
        })
        .error(function(err){
            console.log(err.statusText);
        });
}
```

```
function someAsyncCall(){
    var deferred = $q.defer();

    if(iHaveSomething){
        deferred.resolve(someResult);
    }else{
        deferred.reject({statusText: 'I got nothing.'});
    }

    return deferred.promise();
}

function getMeSomething(){
    return someAsyncCall()
        .success(function(result){
            return result;
        })
        .error(function(err){
            console.log(err.statusText);
        })
        .finally(function(){
            //something I ALWAYS want to do
        })
        .catch(function(err){
            // I had an error in one of the success/error/finally blocks
        });
}
```

Useful Links

Mozilla Developer Network

- <https://developer.mozilla.org>

ECMA Website

- <http://www.ecma-international.org/ecma-262/5.1/>

jsbin.com

jsfiddle.com

plnkr.co

codepen.io