```
(function() {
    console.log('JavaScript');
})();
```

RYAN ALBRECHT - FRESHBOOKS

NOVEMBER 2013

HISTORY

Developed by Brandon Eich at Netscape in 10 days

Has many names:

LiveScript, JavaScript, JScript, ActionScript, ECMAScript

Combines parts and ideas from many other functional and object-oriented languages:

SmallTalk, LISP, Self, Java, Hypercard

---

10 of the top 25 projects on **http://github.com/trending** are primarily JavaScript (plus one JS style guide)

WHO HAS A JAVASCRIPT ENGINE INSTALLED?

GOODSIDES

- Object based
- Object and array literals
- First class functions
- Classical OOP inheritance
- Functional patterns
- Light-weight and expressive
- C-Style language

DOWNSIDES

- Single threaded
- Stigma from The Old Days
  - Internet Explorer 6
  - DOM api
  - Internet Explorer 7
  - Slow interpreters
  - Internet Explorer 8
- Type comparisons

JS SYNTAX

REFRESHER

example-1.js

```
"use strict";

var name = '';

if (!name) {
    name = window.prompt('What is your name?');
}

console.log('Hello ' + name);

printOdd(10);

function printOdd(max) {
    for (var i = 0; i < max; i++) {
        if (i % 2 === 1) {
            console.log(i + ' is Odd');
        }
    }
}
```

```
"use strict"; // ?
```

MAKES JS SANE

- Subset of JS
- Complains about undefined variables
- Removes slow features, like `with { ... }`
- Removes magic around `arguments`
- Prepares for future versions

USE STRICT IN YOUR SCRIPTS

vars-typeof.js

```
"use strict";

var isHappy = true;             // typeof isHappy   === 'boolean'

var count = 0,                  // typeof count     === 'number'
    quotient = NaN;             // typeof quotient  === 'number'

var appName = 'freshapp',       // typeof appName   === 'string'
    mascot = "Billingsworth"; // typeof mascot    === 'string'

var something;                  // typeof something === 'undefined'

var myThing = {key: 'value'}, // typeof myThing   === 'object'
    myList = [1, 2, 3],         // typeof myList    === 'object'
    dinner = null;              // typeof dinner    === 'object'
```

vars-truthy.js

```
"use strict";

var isTruthy = true;
isTruthy = 'ryan';
isTruthy = 1;
isTruthy = {key: 'value'};
isTruthy = [1, 2, 3];
isTruthy = '0';

var isFalsy;
isFalsy = false;
isFalsy = null;
isFalsy = 0;
```

function

THE OTHER TYPE

add-sub.js

```
"use strict";

var add = function(left, right) {
    return left + right;
};

function getSubFn() {
    return function(left, right) {
        return left - right;
    };
}

var subtract = getSubFn(); // returns a function

add(1, 2);                 // returns 3
subtract(5, 3);            // returns 2
```

RECAP

- Single-threaded execution
- C-Style with curlies and semi-colons;
- `'use strict';` will make your life better
- Loose types checks are unintuitive *sometimes*
- Functions are types too!
  - They can be passed around just like any other value

JS IN THE BROWSER

WEBPAGE LIFECYCLE

1. Context Creation Stage
    - Identify New Identifiers
    - Resolve Variable Scope
    - Bind Context
2. Code Execution Stage
    - Run the code

index.html

```
<html>
    <head>
        <title>example.com</title>
    </head>
    <body>

        <script type="text/javascript" src="example-1.js"></script>

    </body>
</html>
```

WEBPAGE LIFECYCLE

1. Context Creation Stage
    - **Identify New Identifiers**
    - Resolve Variable Scope
    - Bind Context
2. Code Execution Stage
    - Run the code

IDENTIFY NEW IDENTIFIERS

A LITTLE PICK-ME-UP

Variable Hoisting

example-1_weird_vars.js

```
"use strict";

if (!name) {
    var name = window.prompt('What is your name?');
}

console.log('Hello ' + name);

printOdd(10);

function printOdd(max) {
    for (i = 0; i < max; i++) {
        if (i % 2 === 1) {
            var i;
            console.log(i + ' is Odd');
        }
    }
}
```

```
--- talks/src/javascript/example-1_weird-vars.js
+++ talks/src/javascript/example-1_interpreted.js
@@ -1,17 +1,20 @@
- "use strict";
-
+function printOdd(max) {
+    var i;
+    for (i = 0; i < max; i++) {
+        if (i % 2 === 1) {
+            console.log(i + ' is Odd');
+        }
+    }
+}
+
+var name;
+
 if (!name) {
-    var name = window.prompt('What is your name?');
+    name = window.prompt('What is your name?');
 }

 console.log('Hello ' + name);

 printOdd(10);
-
-function printOdd(max) {
-    for (i = 0; i < max; i++) {
-        if (var i % 2 === 1) {
-            var i;
-            console.log(i + ' is Odd');
-        }
-    }
-}
```

```
--- talks/src/javascript/add-sub.js
+++ talks/src/javascript/add-sub_interpreted.js
@@ -1,16 +1,16 @@
-"use strict";
-
-var add = function(left, right) {
-    return left + right;
-};
-
 function getSubFn() {
     return function(left, right) {
         return left - right;
     };
 }

-var subtract = getSubFn(); // returns a function
+var add, subtract;
+
+add = function(left, right) {
+    return left + right;
+};
+
+subtract = getSubFn(); // returns a function

 add(1, 2);                  // returns 3
 subtract(5, 3);             // returns 2
```

name-same.js

```
"use strict";

var countDownOddNums = function(from) {
    for (i = from; i >= 0; i--) {
        if (i % 2 === 1) {
            var i;
            console.log(i);
        }
    }
};

for (var i = 0; i < 10; i++) {
    countDownOddNums(i);
}

// Output:
// 1
// 1
// 3
// 1
// 3
// 1
// 5
// 3
// ...
```

WEBPAGE LIFECYCLE

1. Context Creation Stage
    - Identify New Identifiers
    - **Resolve Variable Scope**
    - Bind Context
2. Code Execution Stage
    - Run the code

RESOLVE VARIABLE SCOPE

OPEN GUIDE TO CLOSURES

Lexical Scope / Closure

global-scope.js

```
"use strict";

var name = 'Alice';

var saySomething = function(phrase) {
    console.log(name + ' says ' + phrase);
};

saySomething('Hello World'); // prints "Alice says Hello World"



name = 'Bob';
saySomething('Knock Knock.');   // prints "Bob says Knock Knock."

name = 'Carol';
saySomething("Who's there?");     // prints "Carol says Who's there?"
```

```
--- talks/src/javascript/global-scope.js
+++ talks/src/javascript/global-scope_refactor.js
@@ -1,8 +1,10 @@
 "use strict";

+var whoSaysSomething = function(name) {
-    var name = 'Alice';

     var saySomething = function(phrase) {
         console.log(name + ' says ' + phrase);
     };

+    return saySomething;
+};
```

function-scope.js

```
"use strict";

var whoSaysSomething = function(name) {

    var saySomething = function(phrase) {
        console.log(name + ' says ' + phrase);
    };

    return saySomething;
};

var bobSays = whoSaysSomething('Bob'),
    carolSays = whoSaysSomething('Carol');

bobSays('Knock Knock.'); // prints "Bob says Knock Knock."
carolSays("Who's there?");   // prints "Carol says Who's there?"
```

window-scope.js

```
"use strict";

var whoSaysSomething = function(name) {

    if (!name) {
        name = window.name;
    }

    var saySomething = function(phrase) {
        console.log(name + ' says ' + phrase);
    };

    return saySomething;
};

var bobSays = whoSaysSomething('Bob'),
    carolSays = whoSaysSomething('Carol');

bobSays('Knock Knock.'); // prints "Bob says Knock Knock."
carolSays("Who's there?");   // prints "Carol says Who's there?"

var name = 'Dan',
    danSays = whoSaysSomething();

danSays('A Fish!');       // prints "Dan says A Fish!"
```

counter.js

```
"use strict";

var counter = function(start, amount) {
    var count = start;
    return function() {
        count += amount;
        console.log('Started at', start, 'Now at', count);
        return count;
    };
};

var next = counter(0, 1);

next();      // returns 1, prints "Started at 0 Now at 1"
next();      // returns 2
next();      // returns 3

var countDown = counter(100, -1);
countDown(); // returns 99
countDown(); // returns 98

next();      // returns 4
next();      // returns 5
countDown(); // returns 97
```

the-javascript-question.js

```
"use strict";

var i, j, a;

a = [];

for (i = 1; i <= 5; i++) {
    a.push(function() {
        console.log(i);
    });
}

for (j = 0; j < 5; j++) {
    a[j]();
}
```

the-javascript-question-fixed.js

```
"use strict";

var i, j, a;

a = [];

for (i = 1; i <= 5; i++) {
    a.push(
        (function(n) {
                return function() {
                console.log(n);
        }
        })(i)
    );
}

for (j = 0; j < 5; j++) {
    a[j]();
}

// prints:
// 1
// 2
// 3
// 4
// 5
```

WEBPAGE LIFECYCLE

1. Context Creation Stage
    - Identify New Identifiers
    - Resolve Variable Scope
    - **Bind Context**
2. Code Execution Stage
    - Run the code

BIND CONTEXT

`THIS` THIS IS NOT THAT `THIS`

say-hi.js

```
"use strict";

var sayHiTo = function(name) {
    alert("Hi " + name + ", it's me, " + this.name);
};

var alice = {
    name: 'Alice'
};
sayHiTo.call(alice, 'Bob'); // "Hi Bob, it's me Alice"

alice.sayHiTo = sayHiTo;
alice.sayHiTo('Carol');     // Hi Carol, it's me Alice"
```

say-hi.py

```
class Person:
    def __init__(self, name):
        self.name = name

def sayHiTo(me, name):
    print "Hi %s, it's me, %s" % (name, me.name)

alice = Person('Alice')

sayHiTo(alice, 'Bob')  # prints Hi Bob, it's me, Alice

setattr(Person, 'sayHiTo', sayHiTo)
alice.sayHiTo('Carol') # prints Hi Carol, it's me, Alice
```

fn-apply.py

```
"use strict";

var alice = {
    name: 'Alice'
};

var saySomething = function(phrase) {
    console.log(this.name + ' says ' + phrase);
}

window.name = 'Zack';
saySomething('Hi Yari');                // prints "Zack says Hi Yari"

var name = 'Yari';
saySomething('Hi Zack');                // prints "Yari says Hi Zack"

saySomething.call(alice, 'Hello World'); // prints "Alice says Hello World"

saySomething.apply(alice, ['Foo Bar']);  // prints "Alice says Foo Bar"
```

object-says.py

```
"use strict";

var alice = {
    name: 'Alice',

    saySomething: function(phrase) {
        console.log(this.name + ' says ' + phrase);
    }
};

alice.saySomething('Hello World');           // prints "Alice says Hello Worl

var bob = {
    name: 'Bob'
}

alice.saySomething.call(bob, 'Knock Knock'); // prints "Bob says Knock Knock"
alice.saySomething.call(bob, 'Ring Ring');   // prints "Bob says Ring Ring"

bob.saySomething = alice.saySomething;
bob.saySomething("I'm cold outside!");       // prints "Bob says I'm cold out
```

sharing-methods.js

```
"use strict";

var methods = {
    sendEmail: function(letter) {
        console.log('Sending mail to', this.emai);
        console.log('Put this in the envelope', letter);
    },
    placeCall: function(message) {
        console.log('Calling by phone', this.phone);
        console.log('The message is', message);
    }
};

var alice = {
    name: 'Alice',
    phone: '416-555-1234',
    email: 'alice@freshbooks.com'
};

for (var name in methods) {
    alice[name] = methods[name];
}

alice.sendEmail(alice, 'Happy Birthday');
alice.placeCall('That email is hilarious');
```

```
(function() {
    return {
        name: 'Presentation Ended',
        status: 'Success',
        next: 'Questions?'
    };
})();
```