JavaScript by Immersion

Brazos Valley Makers

May 31, 2015

1 About JavaScript

JavaScript is a prototypal, weakly typed, dynamically typed scripting language created for Netscape by Brendan Eich in 10 days in 1995[1]. All of these characteristics have specific implications for JavaScript, and we will explore each of them and others as we go along.

2 When in Doubt, Type it Out

This primer will take a kinesthetic, learning by doing, approach to JavaScript. As such, we will need some things, most notably a JavaScript REPL (Read Eval Print LoopâĂŤin our case Node) and a browser with decent console capabilities (Chrome or Firefox will suffice).

2.1 I Seriously Hope You Have a Browser Installed Already

If not, there is no hope for you, but I heard they are serving refreshments somewhere.

2.2 Installing Node

2.2.1 On a Mac

```
$ ruby -e "$(curl -fsSL\
 https://raw.githubusercontent.com/Homebrew/install/master/install)
     "
```

[1] https://www.w3.org/community/webed/wiki/A_Short_History_of_JavaScript

```
$brew install node
```

When installing node with homebrew, you will see the following message: "If you update npm itself, do NOT use the npm update command. The upstream-recommended way to update npm is: npm install -g npm@latest "

This is nothing to worry about. It simply means, that when you want to update the node package manager, do it like so:

```
$npm install -g npm@latest
```

2.2.2 On Windows

Go to https://nodejs.org/download/ and download the installer. Run the installer.

3 Functions

Simply put, a function performs one or more operations. Functions can exist on their own. Functions can be use to define objects. Functions can be used to give objects behavior.

3.1 Functions Can Produce a Value

At the command prompt in a terminal window (Type "Terminal" in Spotlight in OS X or Press Windows + R to bring up the Run box and type cmd.exe to open a terminal), type in node and hit Enter. You should be greeted with the Node prompt (a ">"). Type in the following:

```
function addOne (n)  { return n+1; }
```

And press the *Enter* key.

Then type the following pressing enter at the end of the line:

```
addOne(1);
```

You should see:

```
2
>
```

From here on out, it is implied that you need to press the *Enter* key after typing in code to the Node prompt.

3.2 Functions Don't Need a Name

Type:

```
(function () { return 'You never even call me by my name.'; })();
```

3.3 Functions Can be Function Parameters

Type:

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

function before (callback) {
    console.log('before');
    callback();
}

before(after);
```

This feature is taken advantage of in Continuation Passing Style.

4 Numbers

Type:

```
typeof 1;
typeof 1.1;
```

JavaScript is very egalitarian when it comes to numbers. It treats them all as floats.[2] It will print numbers with nothing after the decimal place as integer numbers though.

Side note: *typeof* is a unary operator, not a function, so you don't need parentheses to use it.

4.1 Arithmetic

Type:

```
1 + 2 * 6;
(1 + 2) * 6;
```

Arithmetic operations are performed from left to right in precedence order. Expressions in parentheses are evaluated first.[3]

[2] http://speakingjs.com/es5/ch11.html

[3] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Operator_Precedence

5 Strings

Type the following at the Node prompt:

```
'Hola';
"Mundo";
```

Strings are basically just text enclosed in quotation marks.[4]

Type:

```
'one' + 'two' + 'three';
```

As you can see, strings can be composed from smaller strings. This is called string concatenation.

Try:

```
'hello world'.replace('world', 'college station');
```

JavaScript gives us all kinds of neat things we can do to string primitives courtesy of type coercion and the String constructor.[5]

Try:

```
"Hook 'Em".toLowerCase();
'whoop'.toUpperCase();
```

A *string primitive* is a value that is a string. A *string literal* is literally "a string".

[4] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String

[5] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/var