Javascript 101

Finally!

Introducing.... Javascript!

- Netscape decides HTML needs to be interactive
- 1995, Netscape created Mocha
- Renamed to Javascript
- Microsoft created JScript
- 1997, Javascript made into an ECMA standard
- Javascript takes over!

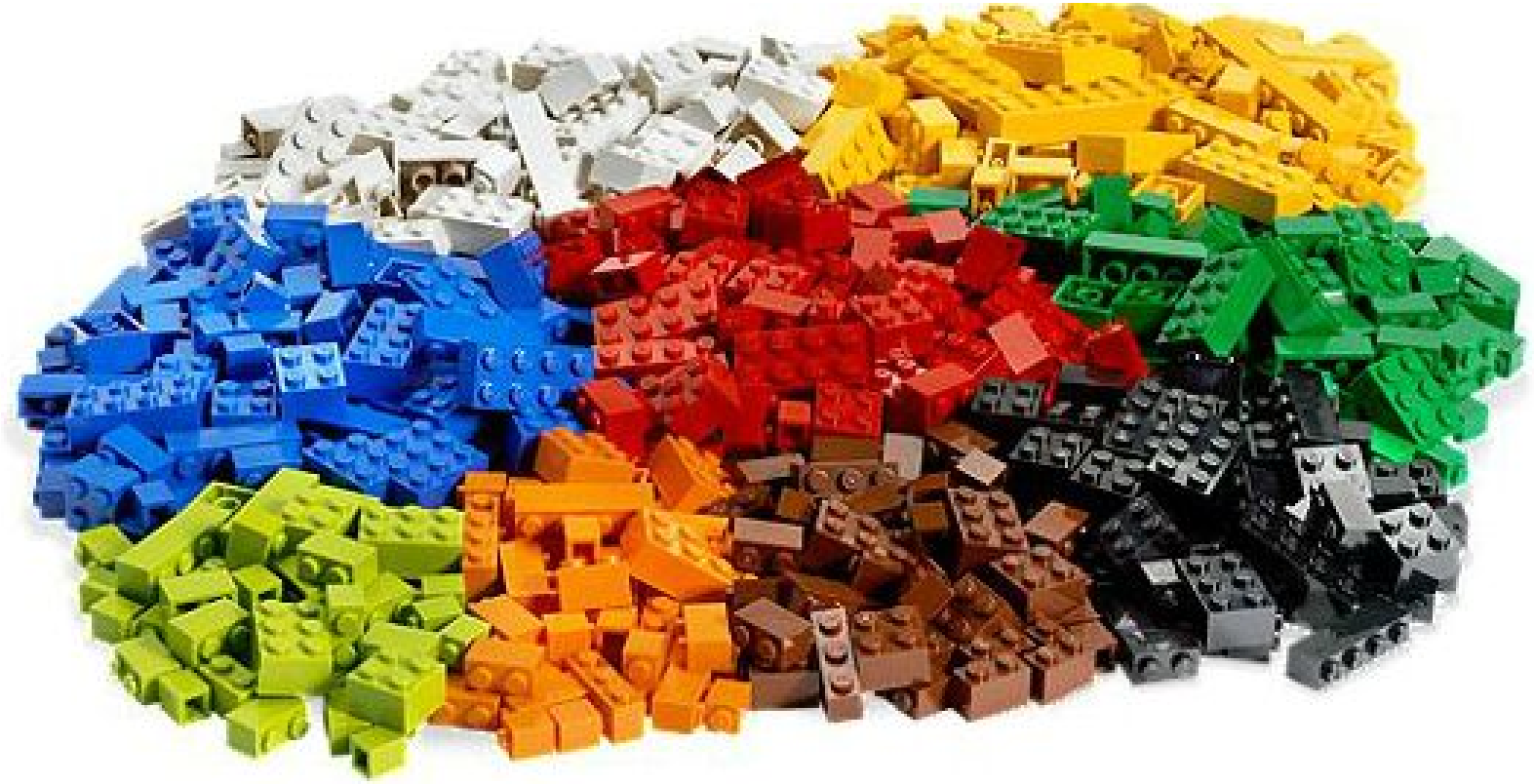
Javascript is similar to Python, but it is *not* the same!

But most things you learned in Python can be applied to Javascript!

Python vs Javascript

Lives in the OS	Lives in a host (like your browser)
Lines terminate with newline character	Lines terminate with a semicolon
“print” will print to your terminal	“console.log()” will print to your console
Variables must be defined with initial value	Variables can be defined without a value
Code blocks denoted by indentation	Code blocks denoted by { }
list	array
dict	object

LEGOs!



Variables

- Variables are *declared* with the keyword “var”
- Variables are like labels - they point to any kind of value
- Variables are initialized with “=”

```
var undefined; // Undefined!

var number = 4;

var float = 4.1;

var  string = “hello world!”
```

Numbers

- Javascript numbers can be written with or without a decimal point
- Numbers with decimal points are called floats

```
var number = 4;

var float = 4.1;
```

Strings

- Strings are a literal type used to define text
- Unlike python, there is no concept of a “character”
- A single character is just a string with one character;
- A string literal is denoted by “” or ‘’

```
var emptyString = “”;

var string = ‘This is a string!’;

var string = “a”;
```

Objects

- Objects are similar to Dicts from Python
- They allow you to map “keys” to values
- Objects are denoted with { }
- Object keys can be accessed with either .keyname or [“keyname”]

```
var emptyObject = { };

var object = {

  firstKey: “hello!”,

  secondKey: 4

}

var object.firstKey  == “hello!”; // True

var object[“secondKey”}  == 4; // True
```

Arrays

- Arrays are just “special” objects
- They automatically create keys in numerical order: 1, 2, 3, etc.
- You can access elements in an array by their index with [index]

```
var emptyArray = [];

var array = [ “hello!”, 4 ]

var array[0]  == “hello!”; // True

var array[1]  == 4; // True
```

Number Operators

- +, -, /, and * will add, subtract, divide, and multiply (respectively)
- % calculates “modulo”, or the remainder after division
- +=, -+, /=, *=, %= combines operators with assignment

```
var add = 2 + 2; // equals 4

var subtract = add -  2; // equals 2

var multiply = 2 * 2; // equals 4

var divide = 6 / 4; // equals 1

var mod = 6 % 2; // equals 0


var add += 2; // add = add + 2;
```

String Operators

- + will concatenate two strings (or a string and a non-string)
- .length will return the number of characters in the string
- .indexOf(sub) will return the index of the substring

```
var concat = “hello” + “world”; // helloworld

var concat2 = 4 + “world”; // 4world

var length = concat2.length; // 6

var indexWorld = concat.indexOf(‘world’);

 // 5
```

Boolean Operators

- ! will return the opposite of the boolean

- && will apply “and” to two boolean values

- || will apply “or” to two boolean values
- == will check for equality between two values

```
var boolean = true;

var not = !boolean; // false

var and = boolean && false; // false

var or = boolean || false; // true

var equals = true == false; // false
```

Array Functions

- .push(element) will append a value to the end of the array
- .pop() will remove the last element of the array
- .length will tell you the number of items in the array

```
var array = [‘adam’, ‘szaruga’];

array.push(4); // [‘adam’, ‘szaruga’, 4]

array.pop(); // [‘adam’, ‘szaruga’]

array.length; // 2
```

Conditionals

- Exactly like Python conditionals, except ‘elif’ is now ‘else if’

```
if ( adam == “awesome”) {
        // Code here will execute if adam is
            Awesome
} else if (adam == “just ok”) {
        // Code here will execute if adam is
            Just ok
} else {
        // Code here will execute if adam
          Isnt awesome or just ok
}
```

For loops

- For loops need three statements: for (s1; s2; s3)
- Statement1 is run before the for loop starts
- Statement2 is checked to decide whether the code should loop again
- Statement3 is run after each loop

```
for ( var i=0; i < 10; i++) {
        console.log('hi');
        // this for loop will run 10 times
}
```

While loops

- while loops need one statement: while (s1)
- Statement1 is run before every loop to see if the code should be run

```
var i=0;
while ( i < 10) {
        i++;
        // this while loop will run 10 times
}
```

Functions

- JS functions are *almost exactly* like Python functions
- Function names are *optional*
- Functions can be saved to a variable
- Functions take many inputs and can return *one* output

```
function myFunc() {
        return “hi!”;
        // this function has no inputs, but outputs the string “hi!”
}
function myFunc2(arg1, arg2) {
        return arg1 + arg2;
        // this function returns the sum of its two arguments
}
var anon = function () {
                // this function has no name, but is saved to a variable
} ____
anon(); // we can call the function with ()
```

Here’s a Javascript cheat sheet

https://codepen.io/aszaruga6/pen/jwMYmY