INF5750 - JS and node.js demo

Values and types

Weak/dynamic typing

```
var a = 1; // declare a and assign it
typeof a //number

a = "abc"; //
typeof a //string
```

Built-in types

Other built-in types

```
var typeBool = true;
typeof typeBool;

var typeNull = null;
typeof null;
//returns Object - JS bug that is unlikely to be fixed

var typeUndefined = undefined;
typeof typeUndefined;

var typeObject = {}
typeof typeObject;
```

Will look at symbol next week as part of new features in ES6

Functions and properties of built-in types

```
var string = "abc";
string.toUpperCase();

var array = [1, 2, 3];
array.length;
```

Objects

```
var obj = {
  a: 1,
  b: "abc"
}

//Accessing properties
obj.a;
obj["b"];

var arr = [1, 2, 3];
arr[1]; //index 1 = 2;
typeof arr; //Also an object
```

Comparisons

Conversion

```
//Explicit
var str = "1";
var newNum = Number(str);
typeof newNum;

//Implicit
```

```
var newNum = str * 1;
typeof newNum;
```

Equality with and without coercion

```
var str = "1";
var num = 1;

str == num;
//true

str === num;
//false
```

Equality on arrays

arrays are joined wiht ,

```
var arr = [1, 2, 3]

arr == "1,2,3";
true //comparing after arA has been converted to string

arr === "1,2,3";
false //comparing with reference

var newArr = [1, 2, 3];
arr == newArr
false
//Because objects are compared by reference (unless they are coerced)
```

Conditionals

```
var a = 1, b = 2;

if (a > b) {
  console.log("a > b");
}
else if (a == b) {
  console.log("a == b");
}
else {
  console.log("a < b");
}
//a < b
```

Functions

Defining functions

```
function add1(a, b) {
  return a + b;
}

var add2 = function(a, b) {
  return a + b;
}


add1(2, 2);
add2(2, 2);

//Immediately invoked functions
(function() { console.log(2 + 2); })();
```

Scope

```
var a = "global scope";
function scope1() {
  var b = "scope of scope1()";

  function scope2() {
      var c = "scope of scope2()";

  }
  console.log(c); //ReferenceError - not in scope of scope1()
}

scope1();
```

Variables

Hoisting

```
c = 2;
console.log(c);
var c; //hoisted


foo();
function foo() { //hoisted
  console.log("Hello world");
}
```

Prototypes

Object.create() sets the prototype link

```
var myPrototype = {
  a: "Hello"
}

var myObject = Object.create(myPrototype);
myObject.b = "World";

console.log(myObject.b);
console.log(myObject.a); //found in the prototype
```

Node.js

Hello world

Add this in helloWorld.js

```
console.log("Hello world");
```

Using npm

In a project folder

```
npm install moment
```

In a .js file

```
var moment = require("moment");

var thisWeek = moment();
console.log(thisWeek.format());

var nextWeek = thisWeek.add(1, 'week');
console.log(nextWeek.format());
```

Web server

Write a web server that responds to request with a message

```
var http = require("http");

var server = http.createServer();
server.on('request', function(request, response) {

  response.end(request.method + " request was received!");

});

server.listen(8090);
console.log("Listening on port 8090");
```

Chrome debugging

```
node --inspect server.js

//In Chrome:
chrome://inspect
```

Curl

-X => method

-H => header

-vv => verbose

```
curl -vv -X GET "http://localhost:8090"

curl -vv -H "Content-Type: application/json" -X GET "http://localhost:8081"

curl -vv -X POST -H "Content-Type: application/json" "http://localhost:8081"
```

Strict mode

```
function test() {
  unDeclared1 = 123;
  console.log(unDeclared1);
};
test();

function test() {
  "use strict";
  unDeclared2 = 123;
  console.log(unDeclared2); //Reference error - implicit assignment to global not allowed
};
test();
```

Code checker

```
npm install -g eslint //-g install globally rather than for one particular project
eslint --init
eslint server.js
```