<!DOCTYPE html>

<html>

<body>

<h2>My First Web Page</h2>

<p>My First Paragraph.</p>

<p id="demo"></p>

<script>

document.getElementById("demo").innerHTML = 5 + 6;

</script>

</body>

</html>

getElementById(“demo”) used to find element id of an HTML element by the name in “”

.innerHTML makes changes to HTML content

Node.js

var http = require('http');  
  
http.createServer(function (req, res) {  
 res.writeHead(200, {'Content-Type': 'text/html'});  
 res.end('Hello World!');  
}).listen(8080);

Print “Hello World!” when someone pings your server

var http = require('http'); = looks for a module (aka method/function) in a specific location

Ndownload node.js, npm install

Npm install express

Also lookup babel for newer javascript syntax

//var http = require('http');

var express = require('express');

var dt = require('./myfirstmodule.js')

var app = express();

app.get('/', function(req, res){

console.log(req);

res.send('yo');

})

app.get('/test', function (req, res){

res.send("the date and time are " + dt.myDateTime());

})

// http.createServer(express, function(){

// console.log('hi');

// }).listen(8080);

app.listen(8080);

Webpak is used to serve up javascript code

var http = require('http');

var express = require('express');

var fs = require('fs')

var app = express();

app.get('/', function(req, res){

res.send('yo');

});

app.get('/test', function (req, res){

fs.readFile('test.html', readTest.bind(null, res));

console.log('2')

});

//read

function readTest(res, err, data){

res.write(data);

res.end();

console.log('1')

};

app.listen(8080)

// http.createServer(function (req, res) {

// fs.readFile('demofile1.html', function(err, data) {

// res.writeHead(200, {'Content-Type': 'text/html'});

// res.write(data);

// res.end();

// });

app.listen(8080);

Express static files

bind passes in reference to readTest function and passes is res to given readTest scope of res

var http = require('http');

var express = require('express');

var fs = require('fs')

var app = express();

app.get('/', function(req, res){

res.send('yo');

});

app.get('/test', function (req, res){

//bind passes in reference to readTest function and passes is res to given readTest scope of res

fs.readFile('test.html', readTest.bind(null, res));

console.log('2')

});

//read

function readTest(res, err, data){

res.write(data);

res.end();

console.log('1')

};

// http.createServer(function (req, res) {

// fs.readFile('demofile1.html', function(err, data) {

// res.writeHead(200, {'Content-Type': 'text/html'});

// res.write(data);

// res.end();

// });

app.listen(8080);

Nodemon = hotpushing filechanges

npm install nodemon --save-dev

# URL PARSING:

var url = require('url');

var adr = 'http://localhost:8080/default.htm?year=2017&month=february&test=3141592';

var q = url.parse(adr, true);

? marks end of host + pathname

Search returns part of path after pathname

.host = pre /

.pathname / up to ?

.search / after ?

Q.query = returns

'http://localhost:8080/test?today=tuesday&tomorrow=wednesday'

### **console(req.originalUrl) = parse url after /**

app.post('/test', function (req, res){

//bind passes in reference to readTest function and passes is res to given readTest scope of res

//fs.readFile('test.html', readTest.bind(null, res));

//var demo = JSON.parse();

try{

undefined.test

console.log(req.body);

}catch(error){

console.log(error);

res.send("Internal server failure");

}

res.send()

});

var http = require('http');

var url = require('url');

var bodyParser = require('body-parser');

var express = require('express');

var fs = require('fs');

var app = express();

app.use(bodyParser.json()); // for parsing application/json

app.get('/', function(req, res){

res.send('yo');

});

app.get('/test', function(req, res){

jsonFileName = req.query.filename;

//fs.readFile(jsonFileName, readTest.bind((null, jsonFileName)))

//demo = fs.readFile(jsonFileName, readTest.bind((null, res)))

var data=fs.readFileSync(jsonFileName, 'utf8');

var demo = JSON.stringify(JSON.parse(data), null, 1);

res.send(demo);

// app.get('/test', function(req, res){

// jsonFileName = req.query.filename;

// //fs.readFile(jsonFileName, readTest.bind((null, jsonFileName)))

// //demo = fs.readFile(jsonFileName, readTest.bind((null, res)))

// var data=fs.readFileSync(jsonFileName, 'utf8');

// res.send(data)

//res.send(tempURL.query.filename)

//readJsonFileName = .query.filename

//res.send(readJsonFileName)

});

// app.post('/test', function (req, res){

// //bind passes in reference to readTest function and passes is res to given readTest scope of res

// //fs.readFile('test.html', readTest.bind(null, res));

// //var demo = JSON.parse();

// try{

// console.log(req.body);

// res.send(req.body);

// }catch(error){

// console.log(error);

// res.send("Internal server failure");

// }

// res.send()

// });

//read

function readTest(res, err, data){

res.write(data);

res.end();

};

// http.createServer(function (req, res) {

// fs.readFile('demofile1.html', function(err, data) {

// res.writeHead(200, {'Content-Type': 'text/html'});

// res.write(data);

// res.end();

// });

app.listen(8080);

console.log('listening on port 8080')

# Working get and post

var http = require('http');

var url = require('url');

var bodyParser = require('body-parser');

var express = require('express');

var fs = require('fs');

var app = express();

var jsonFileName = null

app.use(bodyParser.json()); // for parsing application/json

app.get('/', function(req, res){

res.send('yo');

});

app.get('/test', function(req, res){

jsonFileName = req.query.filename;

//fs.readFile(jsonFileName, readTest.bind((null, jsonFileName)))

//demo = fs.readFile(jsonFileName, readTest.bind((null, res)))

var data=fs.readFileSync(jsonFileName, 'utf8');

var demo = JSON.stringify(JSON.parse(data), null, 1);

res.send(demo);

});

app.post('/test', function (req, res){

// tempData = ""

// temp = JSON.stringify(req.body);

// for (x = 1; x<tempData.length; i++){

// tempData += temp.charAt(x);

// }

// data = "{\n" + tempData + "\n}"

try{

fs.writeFile('./test.json', JSON.stringify(req.body, null, 1), function(err) {

if (err) throw err;

console.log('Replaced!');

});

res.send(req.body);

}catch(error){

console.log(error);

res.send("Internal server failure");

}

res.send()

});

// app.post('/', function (req, res) {

// res.send('POST request to homepage');

// });

// app.get('/test', function(req, res){

// jsonFileName = req.query.filename;

// //fs.readFile(jsonFileName, readTest.bind((null, jsonFileName)))

// //demo = fs.readFile(jsonFileName, readTest.bind((null, res)))

// var data=fs.readFileSync(jsonFileName, 'utf8');

// res.send(data)

//res.send(tempURL.query.filename)

//readJsonFileName = .query.filename

//res.send(readJsonFileName)

// });

// app.post('/test', function (req, res){

// //bind passes in reference to readTest function and passes is res to given readTest scope of res

// //fs.readFile('test.html', readTest.bind(null, res));

// //var demo = JSON.parse();

// try{

// console.log(req.body);

// res.send(req.body);

// }catch(error){

// console.log(error);

// res.send("Internal server failure");

// }

// res.send()

// });

//read

function readTest(res, err, data){

res.write(data);

res.end();

};

// http.createServer(function (req, res) {

// fs.readFile('demofile1.html', function(err, data) {

// res.writeHead(200, {'Content-Type': 'text/html'});

// res.write(data);

// res.end();

// });

app.listen(8080);

console.log('listening on port 8080')

# Example of Class

class Mortgage {  
   
 constructor(principal, years, rate) {  
 this.principal = principal;  
 this.years = years;  
 this.rate = rate;  
 }  
   
 get monthlyPayment() {  
 let monthlyRate = this.rate / 100 / 12;  
 return this.principal \* monthlyRate / (1 - (Math.pow(1/(1 + monthlyRate),  
 this.years \* 12)));  
 }  
   
 get amortization() {  
 let monthlyPayment = this.monthlyPayment;  
 let monthlyRate = this.rate / 100 / 12;  
 let balance = this.principal;  
 let amortization = [];  
 for (let y=0; y<this.years; y++) {  
 let interestY = 0;  
 let principalY = 0;  
 for (let m=0; m<12; m++) {  
 let interestM = balance \* monthlyRate;  
 let principalM = monthlyPayment - interestM;  
 interestY = interestY + interestM;  
 principalY = principalY + principalM;  
 balance = balance - principalM;  
 }  
 amortization.push({principalY, interestY, balance});  
 }  
 return amortization;  
 }  
   
}

# To do

Babel, react, webpack

Es6/2015 es7

Array and object manipulation and notation, typecasting

This command is to be put in json to make it define what npm babel does

babel --presets es2015 js/main.js -o build/main.bundle.js

Install babel using:

npm install babel-cli babel-core --save-dev

npm install babel-preset-es2015 --save-dev

**For webpack:**

npm install babel-loader webpack --save-dev

Change any packages inside dependencies and then run npm install

var path = require('path');

var webpack = require('webpack');

module.exports = {

entry: './js/main.js',

mode: 'development',

output: {

path: path.resolve(\_\_dirname, 'build'),

filename: 'main.bundle.js'

},

module: {

rules: [

{

test: /\.js$/,

loader: 'babel-loader',

query: {

presets: ['es2015']

}

}

]

},

stats: {

colors: true

},

devtool: 'source-map'

};

# Starting new project

Npm init - creates package.json

npm install nodemon --save-dev

npm install babel-cli babel-core --save-dev

npm install babel-preset-es2015 --save-dev

npm install babel-loader webpack --save-dev

Npm install express

Npm install = updates dependencies

var http = require('http');

var url = require('url');

var bodyParser = require('body-parser');

var express = require('express');

var fs = require('fs');

var app = express();

If webpack needed, will need to set up webpack-config.js

Set mode to production

Replaces loaders with rules

# Arrays

var fruits = [ 'apples', 'banana', 'cheese']; - declaration

console.log(fruits.length);

> 3

Display all contents:

fruits.forEach((item, index, array) => {

console.log(item, index);

})

Reference specific index

var first = fruits[0];

Methods:

.length

.push(‘object’) = adds to end of original array, .push returns length of array

.pop(‘object’) = remove from end of array and changes original array to match

.shift() = removes from beginning of array

.unshift() = add to front of array

.indexOf(‘string’) = returns index

.splice(pos, n) = remove from index pos to index n inclusive

newArray=array.slice()

numbers.map(Math.sqrt)

var newSet = [1,2,3];

var mappedSet = newSet.map((x) => x \* x); // performs operation x^2 to newSet. Does not change newSet

Map and reduce

var reducer = (acumulator, currentVal) => accumulator + currentVal;  
 console.log(fruits.reduce(reducer)); // returns number total of set fruits

# Strings:

.split(‘ ‘) sets ‘ ‘ as delimiter and returns an array

# For loops:

for(let x = 0; x<5; x++){

console.log(x);

fruits.push(x);

}

Objects:

var testObject = { fruits: fruits }

console.log(testObject.fruits);