# workshop intro to node

Max Brosnahan @ Code Camp 2014

# the plan

- Say ‘hello world’
- Create api for geojson data file
- Display geojson on a map
- Filtering via streams
- Misc extras

# Getting started

Install node (unless you have done so)

Create a directory for this workshop

run `npm init` and fill in the relevant details

# Hello world

To make sure everything is working create index.js file in your directory with the contents

console.log(‘Hello Code Camp 2014’);

Execute this file by running `node index.js`

# npm

npm is the package manager for node.

It allows you to install modules from a central repository into the node_modules folder for your project

# npm

To install a package run

npm install --save [package name]

You can install packages globally with -g but this is not a good idea for most packages

# npm

Note: the --save flag installs the module into the node_modules folder for your project and saves version information to the project package.json file

# Install express

Install the package

express

from npm

# Express

Express is a lightweight framework that helps with building applications in node.

It adds a simplified interface to register handlers for endpoints.

It also adds the concept of middleware

# Middleware

In express middleware are helpers that sit between the main request handler inside express and your endpoint handler.

Middleware allow for things like authentication, session setup etc to happen before it reaches your business logic code

# Hello over http

Replace the contents of index.js with

```
var express = require('express');
var app = express();

app.get('/', function(req, res){
    res.send('Hello Code Camp');
}
app.listen(8080);
```

# Hello over http

Execute the file and visit localhost:8080 to check the result

# Static file serving

Create a client folder with a index.html file saying hi. To index.js add

```
var path = require('path')
var clientDir = path.resolve(__dirname, 'client')
app.use(express.static(clientDir))
```

# Static file serving

Restart the server and visit

localhost:8080/index.html

to see your new website taking shape

# Api fun

Create an api folder

# Routing

We are going to create an endpoint with the url

/api/tmps

and it is going to return the contents of the tmp-chch.json file

# Routing

Everything in the api folder should be mounted with the base path /api

To do this we mount a router on the /api path and add sub routers or handlers as needed

# Routing api/

Create a router.js file in the api folder with contents

```
var router = require('express').Router()

router.get('/', function(req, res){
    res.json({message: 'hello from api'});
}
module.exports = router;
```

# Routing /api

Now that we have a router to handle api requests we need to add it to the main app

In index.js add

app.use(‘/api’, require(‘./api/router));

Restart the server and check that it works!

# Routing /tmps

Now that we have /api we are going to add /tmps to it

Create a tmps.js file in the api folder.

Using a similar approach to router.js say ‘hi’ from /tmps

# Stream tmps

Now that we have routing setup we can add some more interesting data to the /api/tmps end point

# Stream tmps

Because the tmp data file we have is quite small (~1mb) we are going to stream it directly to the browser

# Stream tmps

To create a stream from a file we need the ‘fs’ module

```
var fs = require(‘fs’);

var fileStream = fs.createReadStream(pathToTmpData)
```

Note: See index.js for how to get the path to a file

# Stream tmps

Now that we have a stream for the file we need to send it to the browser

Change the /tmps handler to

```
function(req, res){
  fs.createReadStream(pathToFile)
     .pipe(res);
}
```

# Tmps

Now when you visit /api/tmps you should see the contents of the json file

Time to show it on a map

# Browserify

Browserify is a node module that allows us to treat the browser as though it was compatible with the node module system (common js)

Install the browserify module from npm

# Browserify

Add an index.js to the client folder with hello console.log

The client index.js is going to be the entry point for the browser javascript code

Add a script tag to index.html to load index.js

# Browserify

To add browserify in we are going to override the route to the client index.js file with the browserified version

# Browserify

Add the following to the server index.js

```
app.get('/index.js', function(req, res){
  res.setHeader('Content-type', 'text/javascript');
  browserify()
    .add('./client/index.js')
    .bundle({debug: true})
    .pipe(res)
});
```

# Browserify

Now that browserify is setup we can load the tmps from the server as though we were in node land

# Load tmps on the client

```
var http = require(‘http’);

http.get('/api/tmps', function(res){
    var body = ''
    res.on('data', function(data){ body += data; });
    res.on('end', function(){
      var result = JSON.parse(body);
      console.log(result);
    });
  });
```

# Display on tmps on a map

Install google-maps from npm

Add div with an id of google-map-container to index.html

# Display tmps on a map

Setup in client js file

```
var http = require('http');
var GoogleMapsLoader = require('google-maps');
var mapElement = document.getElementById('google-map-container');
```

# Display tmps on a map

```
GoogleMapsLoader.load(function(google){
  var map = new google.maps.Map(mapElement, {
      center: new google.maps.LatLng(-43.525650, 172.639847),  });
  function dislayResultsOnMap(results) {map.data.addGeoJson(results); }
  http.get('/api/tmps', function(res){
    var body = ''
    res.on('data', function(data){body += data; })
    res.on('end', function(){ var result = JSON.parse(body);
dislayResultsOnMap(result);
    });  }); });
```

# Misc 0

Add Authentication middleware

Hard coded username and password is a good start

See http://expressjs.com/ for more info

# Misc 1

Using modules

through, json-array-stream, JSONStream

Create a /api/citylots endpoint and send a small subset of data from the massive citylots json file

# Misc 2

Using streams in the browser add features to the map one at a time

# Misc 3

Add your favourite templating language to express

# Misc 4

Look at various browserify transforms

brfs, envify, es6ify