NODE JS

1 What and Why?

- Server-side scripting in JavaScript: alternative to php, asp, RoR
- Single thread execution: non-blocking
- Not designed for compute-heavy applications
- Package manager npm claims to be largest ecosystem of open source libraries in the world.

2 History

- Developed from 2009
- Based on Chrome V8 Javascript engine: compiles to machine code
- Originally by Joyent, now has its own foundation
- Major fork in 2014 to io.js, since merged
- MIT-style licence
- Package manager npm (and yarn). See left pad incident

3 Hosting

- Cross-platform installation available for local hosting
- Available on many PaaS installations e.g. openshift, IBM bluemix
- Not as widely available as php but more than most others (my opinion/experience)

4 Hello world

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

- Save in hello.js
- run with `node hello.js`
- uses terminal as console instead of browser tools

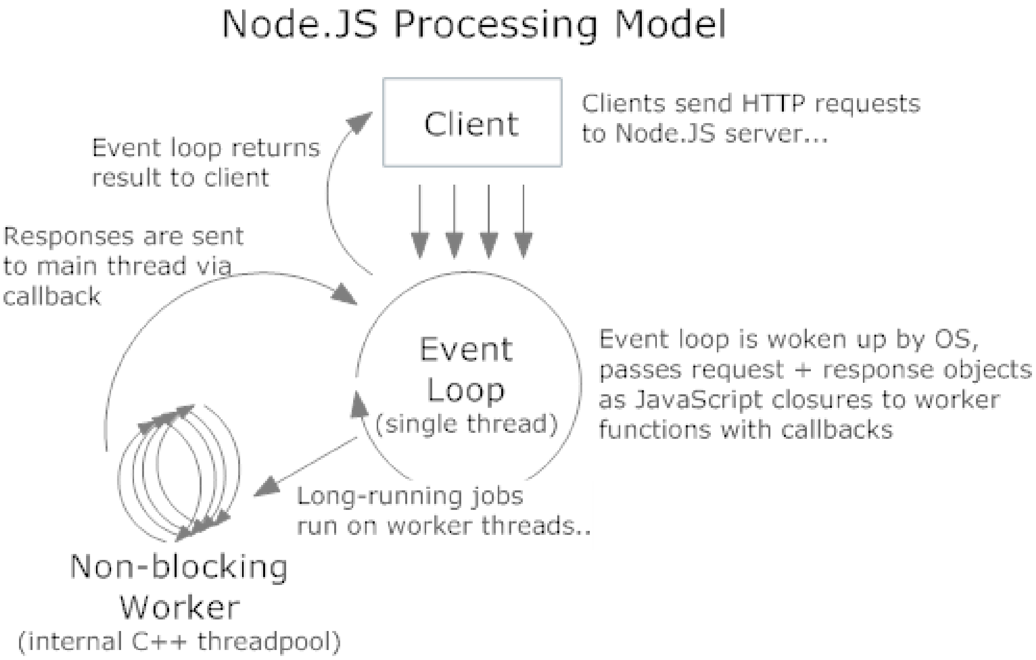
5 Web server

```
http = require("http")

http.createServer(function (request, response) {
        response.writeHead(200, {'Content-Type': 'text/plain'});
        response.end('Hello World\n');
}).listen(8080);

console.log('Server running at http://127.0.0.1:8080/');
```

- Use module http, based on CommonJS
- Module = file
- require returns a js object that contains the module components
- modules are installed with npm if necessary (plus dependencies)
- node starts its own web server rather than running as an apache module (like php often is)

6 Event loop

- node runs on an event loop
- callbacks are associated with events
- programs should be non-blocking so that callbacks are provided for things like
  - block of data arrives from file
  - block of data arrives from REST request
  - results data arrives from database request
- callbacks can themselves trigger new events
- next instruction is executed once all callbacks are complete


Node.JS Processing Model
Client
Clients send HTTP requests to Node.JS server...
Event loop returns result to client
Responses are sent to main thread via callback
Event Loop (single thread)
Event loop is woken up by OS, passes request + response objects as JavaScript closures to worker functions with callbacks
Long-running jobs run on worker threads..
Non-blocking Worker
(internal C++ threadpool)

Thanks to SO user568109

7 Hello events

```
var events = require('events');
var e = new events.EventEmitter();

function say(data){
        console.log("Hello " + data)
        e.emit('done')
}

e.on('say', say);

e.on('done', function(){
        console.log('Message done');
})

e.emit('say', 'world');

console.log('Program done');
```

---

8 Routing requests

- Variables can be GET-encoded
- In REST APIs they are often included in the URL directly (without question marks)
- E.g. use twitter API
- We want to use routing to pick bits out of the URL for this in nodejs
- There is a commonly used package called express

---

9 npm packages

- npm is your friend
- Dependencies are stored in package.json
- Uses semantic versioning
  - Code should have public API by version
  - Version is X.Y.Z
  - X = major version, Y = minor version, Z = patch
  - Never change code within version once released

- version semantics
  - X = 0 for pre-release versions
  - Increase Z for bug-fixes, no new functionality
  - Increase Y, set Z=0 for backwards-compatibile new functionality
  - Increase X, set Y=Z=0 for backwards-incompatible new functionality
- In package.json can use tilde to match minor version, caret to match major version etc

---

10 Install express

```
npm init
```

creates package.json

```
npm install express
```

installs express package and its dependencies

```
npm install express --save
```

installs and puts dependency in package.json

---

11 Express routing

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

app.get('/', function(req, resp){
resp.send('Hello world')
})

app.listen(8090)
```

This just starts an express app handling GET requests via port 8090

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

app.get('/random/:max', function(req, resp){
max = parseInt(req.params.max)
rand = Math.floor(Math.random()*max) +1
console.log('Max via url is ' + max + ' rand is ' + rand)
resp.send('' + rand)
})

app.get('/r', function(req, resp){
max = parseInt(req.query.max)
rand = Math.floor(Math.random()*max) +1
console.log('Max via query is ' + max + ' rand is ' + rand)
resp.send('' + rand)
})

app.listen(8090)
```

- Adds two separate routes
- Extract max value in two different ways
- Via value in URL (:max)
- Via value in GET-encoded variable

---

12 Popular tools for node

- See list at `https://www.npmjs.com/`
- Killer app: chat system