# NodeJS for dummies

Ievgen Svietikov

https://github.com/forlan-ua/nodejs-for-dummies

# NodeJS

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient.

# CommonJS

CommonJS is a project with the goal of specifying an ecosystem for JavaScript outside the browser (for example, on the server or for native desktop applications).

Specifications: Modules, Packages, Promises

# NodeJS Modules

- Buffer, Stream
- Child Process
- Crypto
- Debugger, Utils
- Events
- FS
- HTTP/HTTPS, Net

```
module = {exports:{}, require:function(){}}
function(module, exports, require) {}
```

http://wiki.commonjs.org/wiki/Modules/1.1.1

# package.json

```
{
    “name”: …
    “version”: …
    “main”: …

    “description”: …
    “repository”: …
    “license”: …
}
```

http://wiki.commonjs.org/wiki/Packages/1.1

# npm

npm is the default package manager for the JavaScript runtime environment Node.js.

https://www.npmjs.com/

# Asynchronous JavaScript

## Task

- Get your ip from ipify.org
- Fetch geoip data from the server freegeoip.net
- Save data to a file geoip.json

## Asynchronous JavaScript

1. Callback Hell (lesson1/step1.js)
2. Queue (lesson1/step2.js)
3. Promise (lesson1/step3.js)
4. Generators (lesson1/step4.js)
5. Async/await (lesson1/step5.js)

# HTTP module

- The simplest http server (lesson1/step6.js)

```
const http = require("http");
const PORT = 9092;

const server = http.createServer(function(req, res) {
  res.end("Hello World!");
});

server.listen(PORT, function() {
  console.log(`LISTEN ${PORT}`);
});
```

# REST

REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations.

## CRUD, BREAD, MADS, DAVE

- Create (PUT, POST)
- Read (GET)
- Update (PUT, POST, PATCH)
- Delete (DELETE)

# REST

- GET https://zeo.tv/api/1/screen/
- POST https://zeo.tv/api/1/screen/
- GET https://zeo.tv/api/1/screen/1/
- PUT https://zeo.tv/api/1/screen/1/
- PATCH https://zeo.tv/api/1/screen/1/
- DELETE https://zeo.tv/api/1/screen/1/
- GET https://zeo.tv/api/1/screen/1/widget/?l=10&p=2

# Testing

- Unit testing
- Integration testing
- Functional testing
- System testing

# Test-Driven Development

1. Add a test
2. Run all tests and see if the new test fails
3. Write the code
4. Run tests
5. Refactor code
6. ...
7. PROFIT

# Jasmine

https://jasmine.github.io/edge/introduction

- The first runner (lesson1/step7/index.js)
- The first test suite and test case (lesson1/step7/spec/01. true.js)
- The first setup/teardown (lesson1/step7/spec/02. setup.js)
- The first spy (lesson1/step7/spec/03. spy.js)

# Development tools

1. console.log
2. IDE debugger
3. Module debugger (https://nodejs.org/dist/latest-v7.x/docs/api/debugger.html)
4. V8 profiler (https://nodejs.org/en/docs/guides/simple-profiling/)
5. Insomnia (https://insomnia.rest)
6. JSDoc (http://usejsdoc.org/)

# Thank you for attention!