Kicking it with Node, starter edition.

APRIL 2016 MEETING

Agenda

Introduction

Node.js Heritage

Install Options

npm

Our first and second projects

Use 3rd party modules

Event loop and asynchronicity

Debugging

Next Meeting

May 11th, 2016: Node.JS Framework Shootout

- Ray Bayly will present

Meeting place:

- Advent Coworking

Who wants to help?

- Co-organizer, Event host, food?

Hello

Technology Enthusiast
Standup Philosopher
Relentless Startupper

@BmanClt

bsoylu@xcoobee.com

~bilalsoylu

Bilal Soylu
Chief Worker Bee @ XcooBee

Node.js history

Created in 2009 by Ryan Dahl as experiment.

In essence Google's V8 JavaScript engine (open sourced in 2008) + event loop + low level I/O.

In 2011 NPM (node package manager) was introduced.

Ownership went through a little struggle, but now managed under the Node.js foundation (nodejs.org)

Node.js allows full stack development in JavaScript syntax.

- Client (browser) and server (node) can share code.

The Install

Overall straightforward

Download install package from nodejs.org

Run installer

- This will add path and also add NPM (Node Package Manager)

If you need to run it as windows service you need to use a wrapper service such as NSSM (http://nssm.cc/)

- Node-windows: https://github.com/coreybutler/node-windows

Let's try to run it and type in code

npm

Pre-installed package manager for Node.js platform

It has two components: the package manager and the repository

The package manager

- The package manager makes it easier for the community to publish and share open-source Node.js libraries and is designed to simplify installation, updating and uninstallation of libraries.
- Responsible for explosion of productivity
- Command line client: to become successful with node you need to get familiar with the command line

The Repository (npmjs.com)

- Last check on (npmjs.com) there were 259K packages
- Companies can create private repositories

Watch out for the syntax

JavaScript is baseline but many syntax variants can be supported by Node.js through a “transpile” or “harmony” concept. Simply load a module via npm that can talk that dialect and node will handle the syntax variant

Native support through –harmony flag:

- ES6 (ECMAScript 2015 (ES6) in Node.js) : https://nodejs.org/en/docs/es6/

Some populate languages transpile into JS:

- CoffeeScript, Coco, LiveScript, Uberscript, TypeScript, LispyScript, LiteScript, imba, Dart, Elm
- https://github.com/jashkenas/coffeescript/wiki/list-of-languages-that-compile-to-js
- Babel project is popular transpiler

Let’s create a project

```
npm -init
```

Anatomy of Node.js app

Node.js code is organized in packages

- You create a project by placing a package.json file in directory
  - Will keep track of all your libraries and dependencies
  - npm can use the package.json file to install everything for you: `npm install`
- npm can help create the file

/node_modules

- All modules go into this subdirectory
- This maybe nested in that projects my bundle their own dependencies

/node_modules/.bin

- executables

Let’s Write Some Code

Create own module(s)

The require system (AMD modules)

Properties

Functions

Let's use a 3rd party module

```
npm install --save imagemin
```

```
var Imagemin = require('imagemin');
new Imagemin()
        .src('images/ted.jpg')
        .dest('build')
        .use(Imagemin.jpegtran({progressive: true}))
        .run(function (err, files) {
                console.log(files[0]);
        });
```

Manage Project and Structure

index.html

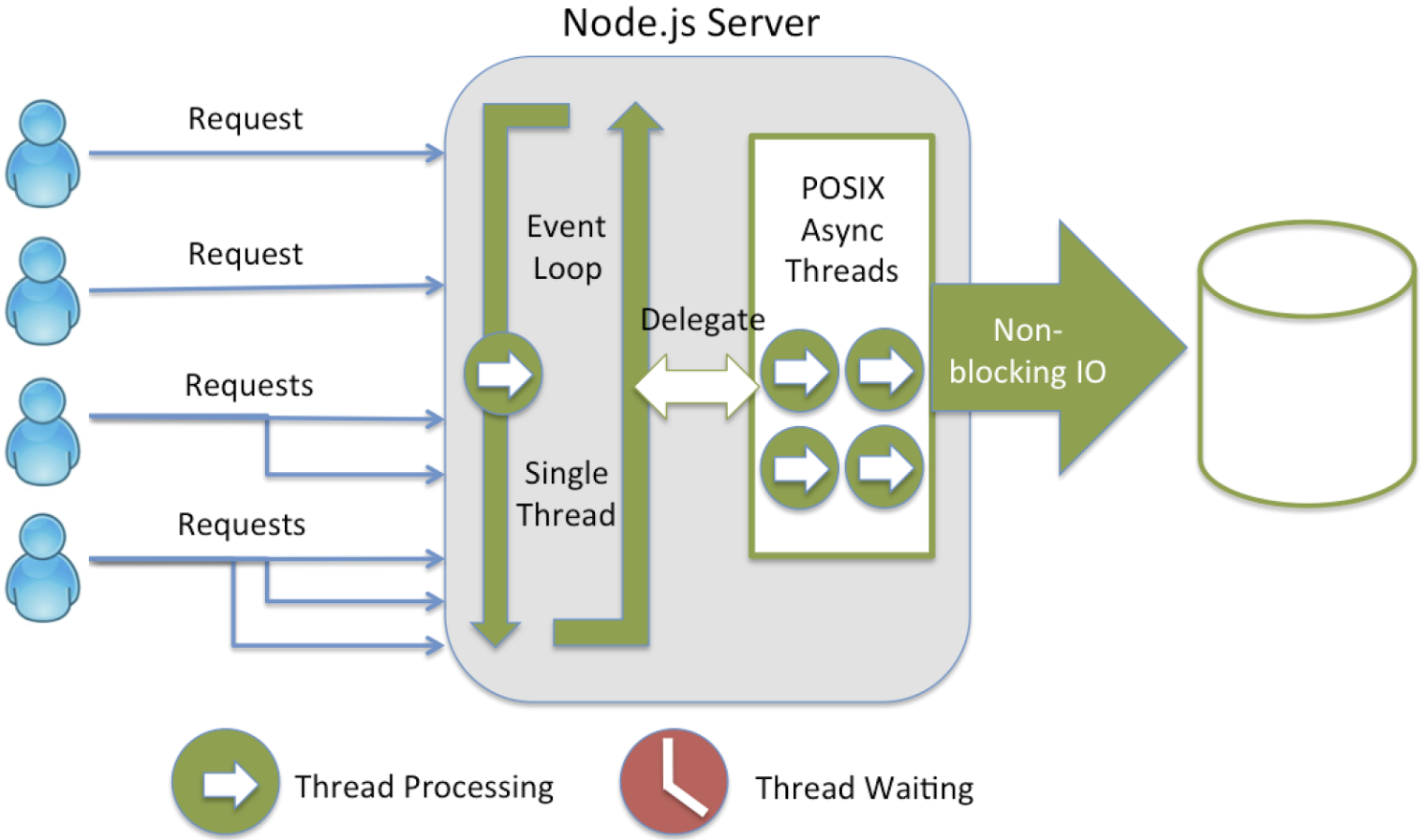
- js/
  - main.js
  - models/
  - views/
  - collections/
  - templates/
  - libs/
    - backbone/
    - underscore/
    - ...
- css/
- ...

In one there are many

Node is single threaded

Let me repeat: Node is single threaded!

The Event Loop



*

*illustration curtesy of StrongLoop

Node loves Asynchronisity

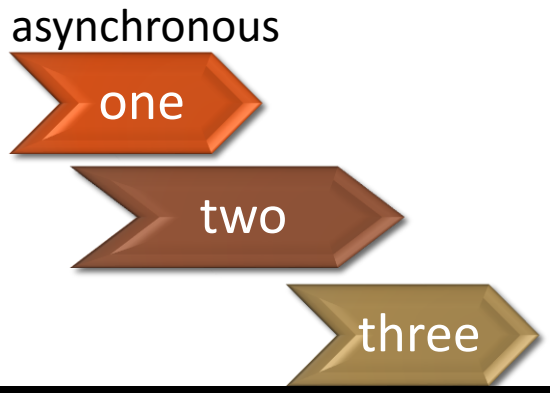
In order for event loop to work and handle high concurrency processing we will need to work as asynchronously as possible.

In asynchronous execution node does not wait on a task to finish before starting a new one. If execution order is important use patterns that ensure processing order.

- Horrible to read code because of Continuous Callback Patterns (CCB)
- To the rescue:
  - Promises
  - Generator functions / Yields

Let's look at some examples

- Node API (https://nodejs.org/api)
- fs Sync
- fs async



Spotting Asynchronicity

Callbacks
- Look for parameters taking functions
- Look for missing return

```
setTimeout(function(){ alert("Hello"); }, 3000);
```

Promises / promise chains
- promise.then(function())

Listeners
- finder.on('done', function (event, records) {})

Generator functions
- function* gen() { }
- Yield

Flow control libraries (e.g. Async js)
- async.series([function(){ ... }, function(){ ... }]);

Let’s add more packages

Global Install (-g)

- Moves packages and code to global area making it available to all node running
  - Most command line extensions work this way
  - `npm install -g node-inspector`

Local Install

- Install package in current working directory (./node_modules)

[Search and add a package node-inspector]

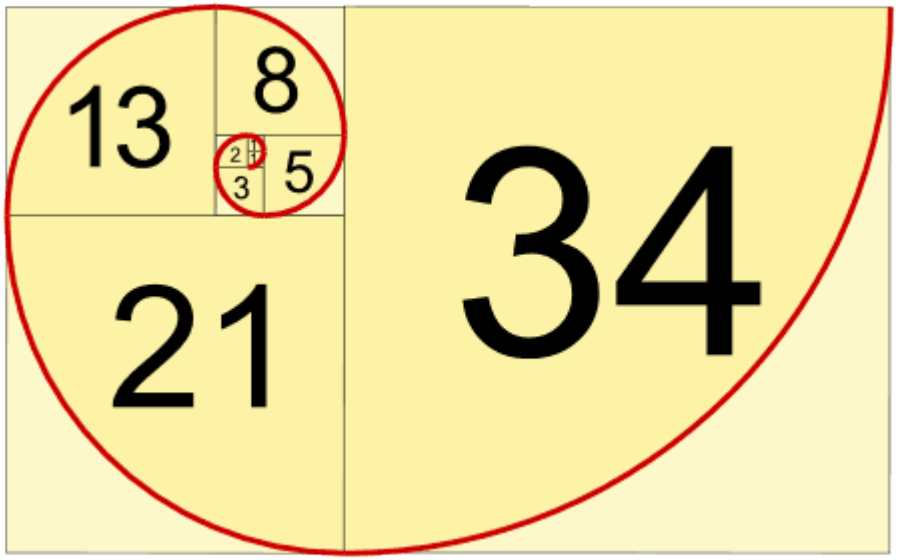
Basic debugging

Using Console class

- console.log, console.assert, console.info, console.time
- `console.log('Error catch:', JSON.stringify(err, null, 2));`
- https://nodejs.org/api/console.html

Node Inspector

- `node-inspector`
- `node-debug fin.js`



Thank You !

Next Meeting

May 11th, 2016: Node.js Framework Shootout

- Ray Bayly will present

Meeting place:

- Advent Coworking

Who wants to help?

- Co-organizer, Event host, food?