Node.js Programming

Agus Kurniawan
Lecturer, Faculty of Computer Science, Universitas Indonesia
PhD student, Freie Universität Berlin, Germany
Microsoft MVP
http://blog.aguskurniawan.net
@agusk2010

Some materials are taken from http://www.slideshare.net/vikasing/introduction-to-nodejs-11730771

Agenda

- Introduction to Node.js
- Basic Node.js Programming Language
- Building Web Application Based Node.js
- Database Programming using Node.js
- Node.js for Embedded System
- Q&A

Introduction to Node.js

Background

- **V8** is an open source JavaScript engine developed by Google, https://code.google.com/p/v8/ . Its written in C++ and is used in Google Chrome Browser.
- **Node.js** runs on V8.
- It was created by **Ryan Dahl** in 2009.
- Latest version is **0.10.33**
- Is **Open Source**. It runs well on Linux and Mac systems, can also run on Windows systems.
- If you have worked on EventMachine (Ruby) or Python's Twisted or Perl's AnyEvent framework then following presentation is going to be very easy.

Introduction: Basic

- In simple words Node.js is **‘server-side JavaScript’**.
- In *not-so-simple* words Node.js is a high-performance **network applications framework**, well optimized for high concurrent environments.
- It’s a **command line** tool.
- In ‘Node.js’ , **‘.js’** doesn’t mean that its solely written JavaScript. It is 40% JS and 60% C++.
- From the official site:

  ***‘Node's goal is to provide an easy way to build scalable network programs’* - (from nodejs.org!)**

Introduction: Advanced (& Confusing)

- Node.js uses an **event-driven, non-blocking I/O** model, which makes it lightweight. (from nodejs.org!)
- Node.js is built upon libuv (https://github.com/libuv/libuv), a cross-platform library that abstracts apis/syscalls for asynchronous (non-blocking) input/output provided by the supported OSes (Unix, OS X and Windows)
- It makes use of **event-loops** via JavaScript's **callback** functionality to implement the non-blocking I/O.
- Programs for Node.js are written in JavaScript but not in the same JavaScript we are use to. There is no DOM implementation provided by Node.js, i.e. you **can not** do this:

```
var element = document.getElementById(“elementId”);
```

- Everything inside Node.js runs in a **single-thread**.

Case Study

- Projects, Applications, and Companies Using Node
  - https://github.com/joyent/node/wiki/projects,-applications,-and-companies-using-node
- Industry
  - http://nodejs.org/industry/

Example-1: Getting Started & Hello World

- Install/build Node.js.
  - https://github.com/joyent/node/wiki/Installing-Node.js-via-package-manager

```
$ curl -sL https://deb.nodesource.com/setup | sudo bash -
$ sudo apt-get install -y nodejs
```

- Open your favorite editor and start typing JavaScript.
- When you are done, open cmd/terminal and type this:

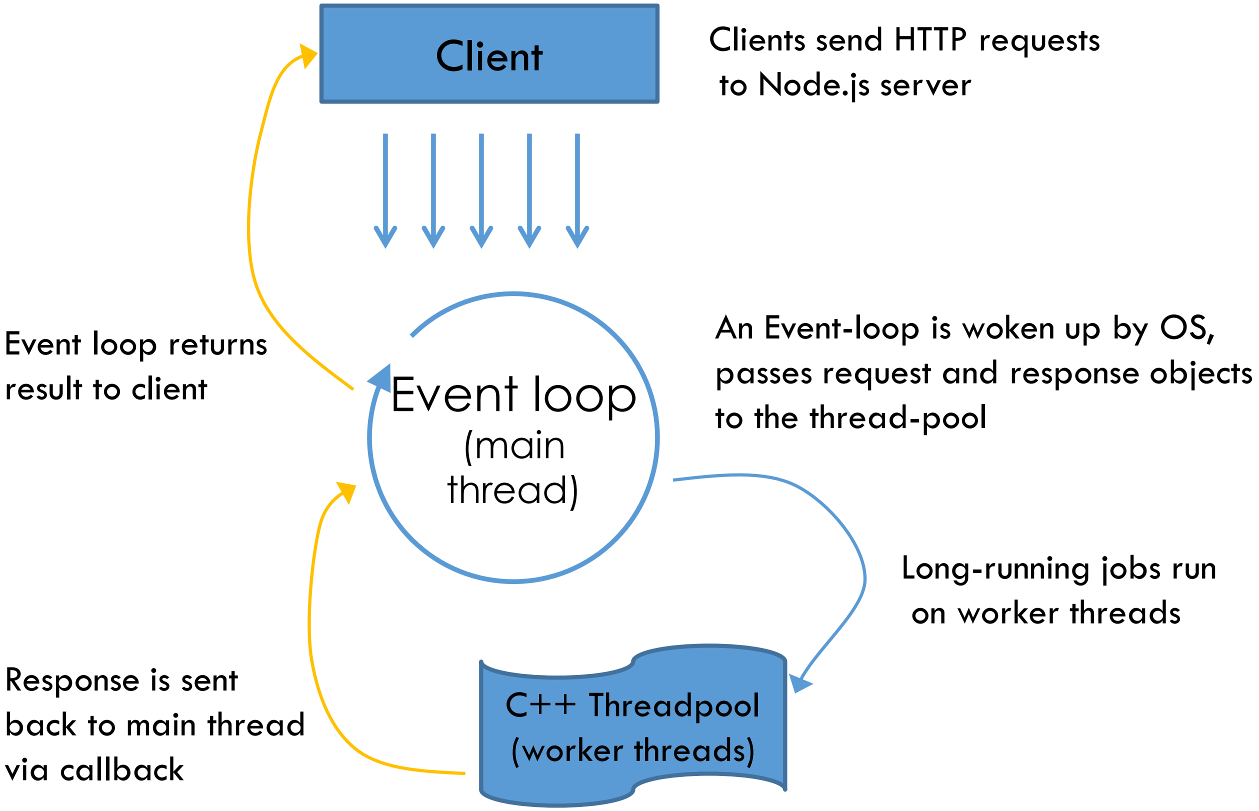
  ```
  'node YOUR_FILE.js'
  ```

- Here is a simple example, which prints '*hello world*'

  ```
  var sys = require("sys");
  setTimeout(function(){
  sys.puts("world");},3000);
  sys.puts("hello");
  //it prints 'hello' first and waits for 3 seconds and then
    prints 'world'
  ```

Some Theory: Event-loops

- Event-loops are the core of event-driven programming, almost all the UI programs use event-loops to track the user event, for example: Clicks, Ajax Requests etc.



Client

Clients send HTTP requests to Node.js server

Event loop returns result to client

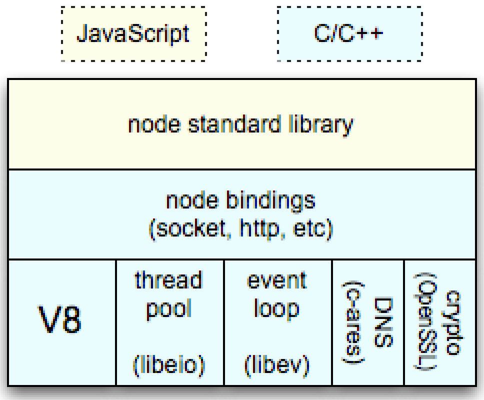
Event loop (main thread)

An Event-loop is woken up by OS, passes request and response objects to the thread-pool

Long-running jobs run on worker threads

Response is sent back to main thread via callback

C++ Threadpool (worker threads)



Some Theory: Non-Blocking I/O

- Traditional I/O

```
var result = db.query(“select x from table_Y”);
doSomethingWith(result); //wait for result!
doSomethingWithOutResult(); //execution is blocked!
```

- Non-traditional, Non-blocking I/O

```
db.query(“select x from table_Y”,function (result){
  doSomethingWith(result); //wait for result!
});
doSomethingWithOutResult(); //executes without any delay!
```

What can you do with Node.js ?

- You can create an **HTTP server** and print ‘*hello world*’ on the browser in just 4 lines of JavaScript.
- You can create a **TCP server** similar to HTTP server, in just 4 lines of JavaScript.
- You can create a **DNS server**.
- You can create a **Static File Server**.
- You can create a **Web Chat Application** like GTalk in the browser.
- Node.js can also be used for creating online games, collaboration tools or anything which sends updates to the user in real-time.

Basic Node.js Programming Language

Node.js Programming Language

- Node.js uses JavaScript as programming language
- JavaScript language is a part of C language family
- Resources
  - http://www.w3schools.com/js/
  - https://developer.mozilla.org/en-US/docs/Web/JavaScript

```
var x = 5;
var y = 6;
var z = x + y;
```

```
if (condition) {
    block of code to be executed if the condition is true
} else {
    block of code to be executed if the condition is false
}
```

```
for (statement 1; statement 2; statement 3) {
    code block to be executed
}
```

Lab…

- Basic Node.js programming language

Building Web Application Based Node.js

Lab…

- Simple web application based Node.js
- Demo: Jakarta.js
  - Node.js, Socket.io, and flot.js
  - https://github.com/agusk/jakarta.js

Web Framework

- Web framework is an abstraction in which common code providing generic functionality can be selectively overridden or specialized by user code providing specific functionality
- Web framework for Node.js
  - Express, http://expressjs.com/
  - Sail.js, http://sailsjs.org/
  - Locomotive, http://locomotivejs.org/
  - Koa, http://koajs.com/
  - Geddy, http://geddyjs.org/

Express

- Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications
  - http://expressjs.com/

```
$ npm install express
```

- Express middleware
  - https://github.com/senchalabs/connect#middleware
- Express generator

```
$ npm install express-generator -g
```

```
$ express myapp
$ cd myapp && npm install
```

```
$ DEBUG=myapp ./bin/www
> set DEBUG=myapp & node .\bin\www
```

Database Programming using Node.js

Database Programming

- Node.js doesn’t provide built-in library to access database server
- Need database driver + node.js module
- Database driver samples:
  - MySQL
    - Node-mysql, https://github.com/felixge/node-mysql/
  - MongoDB
    - Mongodb, http://docs.mongodb.org/ecosystem/drivers/node-js/
  - SQL Server
    - Node-sqlserver, https://github.com/Azure/node-sqlserver
  - Oracle
    - Node-oracle, https://github.com/joeferner/node-oracle

Lab..

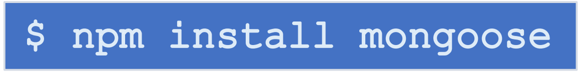
- Node.js and MongoDB

Object-Relational Mapping

- Object-relational mapping (ORM, O/RM, and O/R mapping) is a programming technique for converting data between incompatible type systems in object-oriented programming languages
- ORM for Node.js
  - mongoose, http://mongoosejs.com/
  - Sequelize, http://sequelizejs.com/
  - Caminte, http://www.camintejs.com/
  - Bookshelf.js, http://bookshelfjs.org/
  - backbone-orm, http://vidigami.github.io/backbone-orm/

Mongoose

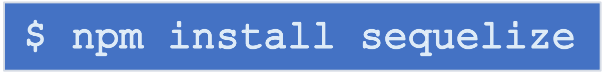
- Mongoose provides a straight-forward, schema-based solution to modeling your application data and includes built-in type casting, validation, query building, business logic hooks and more, out of the box
- http://mongoosejs.com/
- Getting started: http://mongoosejs.com/docs/index.html



```
$ npm install mongoose
```

Sequelize

- The Sequelize library provides easy access to MySQL, MariaDB, SQLite or PostgreSQL databases by mapping database entries to objects and vice versa. To put it in a nutshell, it's an ORM (Object-Relational-Mapper). The library is written entirely in JavaScript and can be used in the Node.JS environment
- http://sequelizejs.com/



```
$ npm install sequelize
```

Lab

- Node.js + Sequelize + MySQL

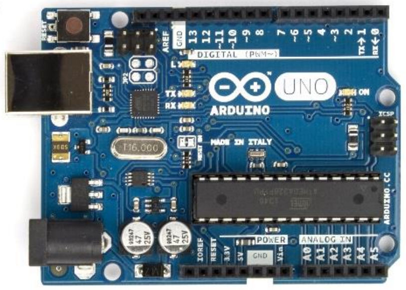
Node.js for Embedded System

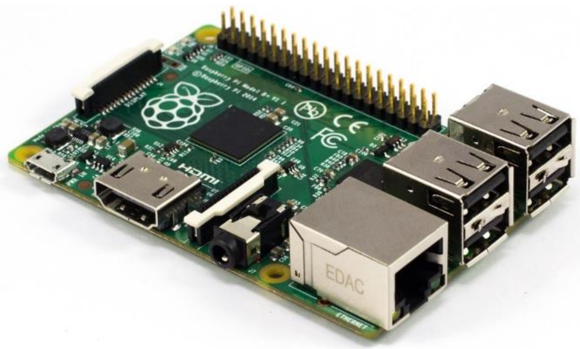
Embedded System

- Microcontroller
- Embedded Hardware
- Internet of Things
- Wearable devices

Node.js Library for Embedded System

- **Johnny-Five** is an Open Source, IoT and Robotics programming framework, developed at Bocoup. Johnny-Five programs can be written for Arduino (all models), Electric Imp, Beagle Bone, Intel Galileo & Edison, Linino One, Pinoccio, Raspberry Pi, Spark Core, TI Launchpad and more!
  - https://github.com/rwaldron/johnny-five





Lab

- Arduino + Firmata protocol
- Node.js

Course Material:
https://github.com/agusk/tutor-nodejs

Q&A

Email: agusk2007@gmail.com
http://blog.aguskurniawan.net
https://github.com/agusk
@agusk2010