# Node.js MariaDB and REST

Contents

[Node.js MariaDB and REST 1](#_Toc530818677)

[1. Node.js 2](#_Toc530818678)

[1.1. Introduction 2](#_Toc530818679)

[1.2. Installation 4](#_Toc530818680)

[1.3. Getting started 5](#_Toc530818681)

[1.4. Express.js 6](#_Toc530818682)

[1.4.1. Installation 6](#_Toc530818683)

[1.5. Nodemon 7](#_Toc530818684)

[1.5.1. Install 7](#_Toc530818685)

[2. REST 7](#_Toc530818686)

[2.1. Postman 9](#_Toc530818687)

[3. Node.js with MariaDB 9](#_Toc530818688)

[3.1. Sequelize JS 10](#_Toc530818689)

What is MariaDB?

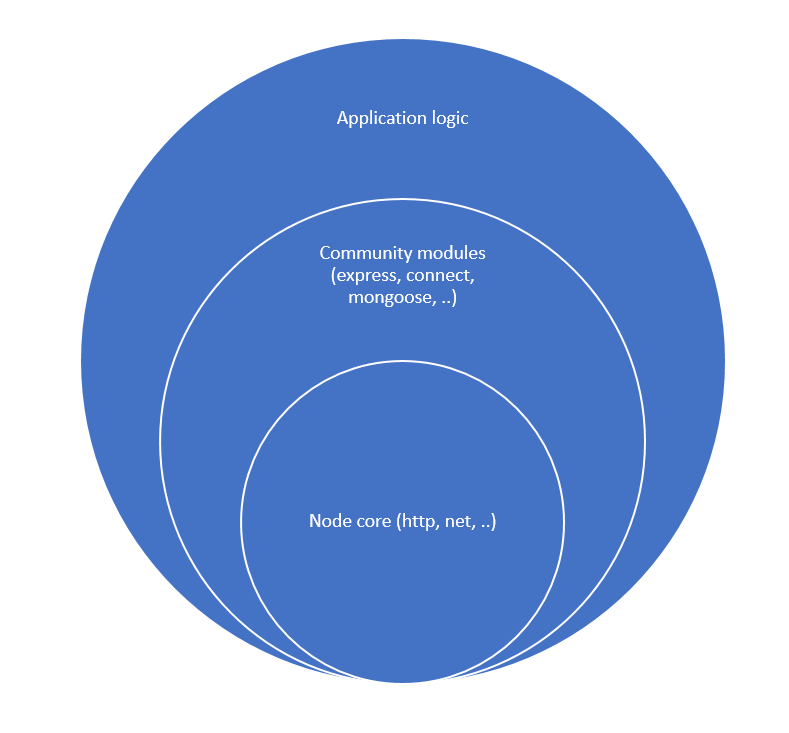
* Replaces Mysql
* Drop in replacement

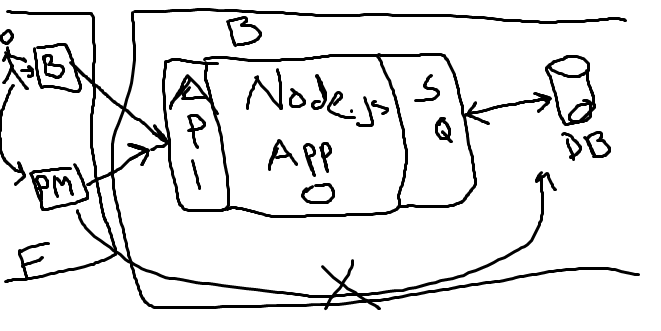
## Node.js

## Introduction

1. What Node.js?
   1. A platform built on Chrome's JavaScript runtime (VM)
   2. For building fast and scalable network applications
   3. Server-side of JavaScript technology stack
   4. Event-driven
      1. Core philosophy of Node
      2. event loop / main loop (Async,non-blocking I/O)
         1. triggers callback when event is detected
         2. Without event loop single-thread will block
      3. I/O performed outside of event loop
   5. Node.js is single-threaded
2. History
   1. Debuted in 2009
3. People
   1. Created by Ryan Dahl
4. Benefits
   1. Fast and efficient
   2. Very easy to setup and configure
      1. No complex IDEs or build system
   3. Single language on client and server
   4. JSON is popular and native to JS
   5. NoSQL databases use JavaScript (same language to talk to the db)
   6. Platform-independent
5. Application areas
   1. DIRTy (data intensive real-time)
   2. Streams and streaming
   3. System scripting
   4. Backend web apps
   5. Embedded applications
   6. Networking
6. Storing Data
   1. NoSQL
   2. SQL
   3. Serverless
7. Web applications
   1. Express.js
   2. Koajs
8. Latest versions of Node implements ES6 (partially)
9. Custom modules and a lot of built-in modules

Application “layers”





## Installation

Download and install

<https://nodejs.org/dist/v10.13.0/node-v10.13.0-x64.msi>

Or use NVM

Download and install NVM

* <https://github.com/coreybutler/nvm-windows/releases/download/1.1.7/nvm-setup.zip>

Get the latest Node.js

* nvm list available
* nvm install 10.9.0

Once installed test that Node.js is working!

* node --version

## Getting started

New project

1. mkdir <dirname>
2. cd <dirname>
3. npm init
4. Add start script (e.g. index.js)
5. Run with: node index.js

**Hello World application**

**Hello world server from :** [**https://www.w3schools.com/nodejs/nodejs\_get\_started.asp**](https://www.w3schools.com/nodejs/nodejs_get_started.asp)

var http = require('http');  
  
http.createServer(function (req, res) {  
    res.writeHead(200, {'Content-Type': 'text/plain'});  
    res.end('Hello World!');  
}).listen(8080);

**RawJSON application**

var http = require('http');

const students = [

{

id: 1,

name: "Gordon Fallin",

phoneNumber: "1234"

},

{

id: 2,

name: "Altha Neisler",

phoneNumber: "5678"

},

{

id: 3,

name: "Oda Highfill",

phoneNumber: "90123"

},

]

http.createServer(function (req, res) {

res.writeHead(200, {'Content-Type': 'application/json'});

res.end(JSON.stringify(students));

}).listen(8080);

## Express.js

It’s laborious to build a web application, specially a bigger one with Node’s built-in http module -> Community has built special libraries to make it easier to build web applications with Node. Express is the mostly popular of these.

## Installation

**npm install express --save**

Prereq: you need to have a project.

Now check package.json dependencies.

"dependencies": {

"express": "^4.16.4"

}

Express and other Node modules get installed to **node\_modules** directory.

You start developing on a new machine, you need to install modules with following command:

**npm install**

To update modules to a newer versions use:

**npm update**

## Nodemon

This module allows you to concentrate on development without needing to restart the server again when you make changes to the source code. Nodemon will automatically restart your Node.js application when a file gets changed

## Install

**npm install nodemon –save-dev**

--save-dev will add the module dependency to the devDependencies section.

"dependencies": {

"express": "^4.16.4"

},

"**devDependencies**": {

"nodemon": "^1.18.6"

}

## REST

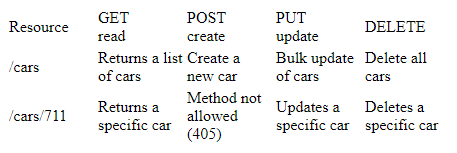
REST = Representational State Transfer

Introduced and defined in 2000 by Roy Fielding in doctoral dissertation.

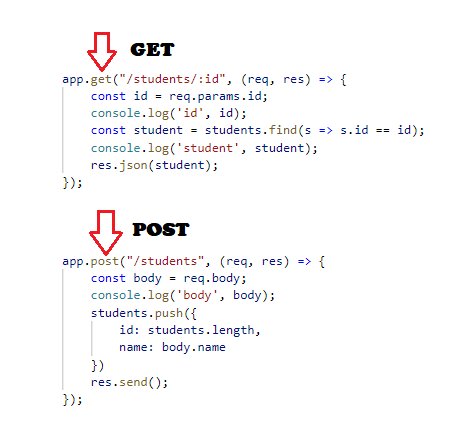
A software architectural style for building web applications

Mostly followed when defining web APIs

HTTP methods are used to define the operation that gets executed on a resource



In Express example.

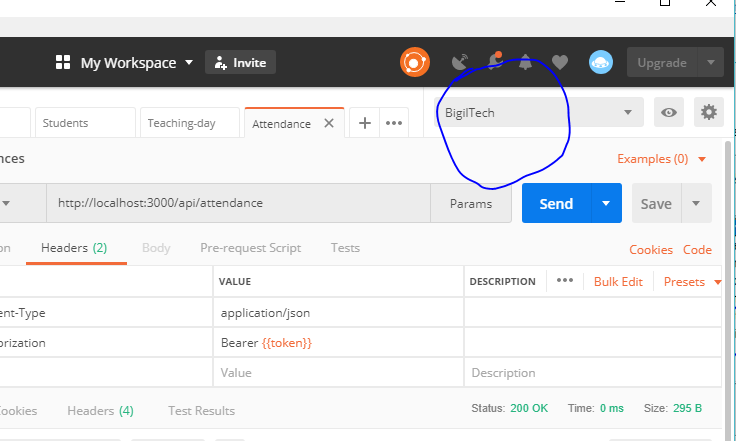


## Postman

Postman is the Only Complete API Development Environment

Download and Docs

1. <https://www.getpostman.com/apps>
2. <https://www.getpostman.com/docs/v6/>



## Node.js with MariaDB

We will use Sequlize.js library with MariaDB.

## Sequelize JS

Sequelize a promise based ORM for Node.js

Supports different dialects sqlite, mysql, postgres etc

Supports a connection pooling

Getting started

1. Setup a connection

const Sequelize = require('sequelize');

const sequelize = new Sequelize('database', 'user', 'password', {

host: 'localhost',

dialect: 'mysql',

pool: {

max: 5,

min: 0,

acquire: 30000,

idle: 10000

},

operatorsAliases: false

});

1. Test the connection

sequelize

.authenticate()

.then(() => {

console.log('Connection has been established successfully.');

})

.catch(err => {

console.error('Unable to connect to the database:', err);

});

1. Define your first model

// sequelize.define('name', {attributes}, {options})

const Student = sequelize.define('student', {

firstName: Sequelize.STRING,

lastName: Sequelize.STRING,

phoneNumber: Sequelize.STRING,

regDate: Sequelize.DATE

});

// force: true will drop the table if it already exists

Student.sync({force: true}).then(() => {

// Table created

return Student.create({

firstName: 'Hussein',

lastName: 'Hussein',

phoneNumber: '12345',

regDate: new Date(2018, 11, 21)

});

});

1. Your first query

Student.findAll().then(users => {

console.log(users)

process.exit()

})

1. Your first query

Promises read more from here: <http://bluebirdjs.com/docs/why-promises.html>