Adon Moskal

adonm@op.ac.nz

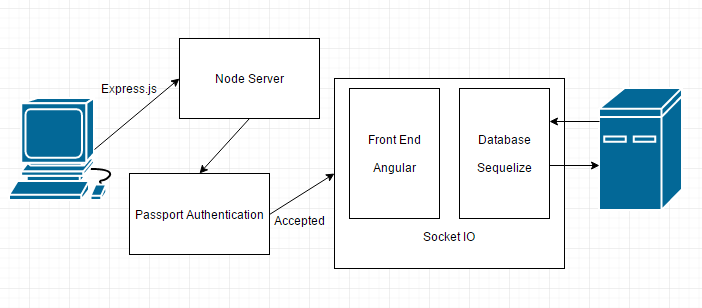
Software Engineering

Technical Documentation

Meals on Wheels

# Introduction

## Summary



# Sections

## Express

*“Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications. Following are some of the core features of Express framework”*

* Allows to set up middleware to respond to HTTP Requests.
* Defines a routing table which is used to perform different actions based on HTTP Method and URL.
* Allows to dynamically render HTML Pages based on passing arguments to templates.

#### Express Steps

Assuming you’ve already installed Node.js, create a directory to hold your application, and make that your working directory.

$ mkdir myapp

$ cd myapp

Use the npm init command to create a package.json file for your application.

$ npm init

This command prompts you for a number of things, such as the name and version of your application. For now, you can simply hit RETURN to accept the defaults for most of them, with the following exception:

entry point: (index.js)

Enter app.js, or whatever you want the name of the main file to be. If you want it to be index.js, hit RETURN to accept the suggested default file name.

Now install Express in the myapp directory and save it in the dependencies list. For example:

$ npm install express --save

To install Express temporarily and not add it to the dependencies list, omit the --save option:

$ npm install express

Node modules installed with the --save option are added to the dependencies list in the package.json file. Afterwards, running npm install in the app directory will automatically install modules in the dependencies list.

IMPORTANT!

<https://www.digitalocean.com/community/tutorials/how-to-install-express-a-node-js-framework-and-set-up-socket-io-on-a-vps>

Setting up NodeJS

Node Version Manager (NVM) is a tool to help install various versions of NodeJS on your linux machine. In order to use NVM ensure you have git and curl installed.

Setting up Express

Express is a web application framework for Node. It is minimal and flexible. In order to start using Express, you need to use NPM to install the module. Simple type:

npm install -g express

This will install the Express command line tool, which will aid in creating a basic web application. Once you have Express installed, follow these steps to create an empty Express project:

<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-node-js-application-for-production-on-ubuntu-16-04>

<https://www.vultr.com/docs/installing-node-js-and-express>

<https://gist.github.com/anotheruiguy/10674846>

<https://code.tutsplus.com/tutorials/authenticating-nodejs-applications-with-passport--cms-21619>

<https://github.com/jaredhanson/passport>

<http://mherman.org/blog/2015/01/31/local-authentication-with-passport-and-express-4/#.WSeoZWiGO70>

<https://www.sitepoint.com/local-authentication-using-passport-node-js/>

**Laiken and Patrick To Do’s:**

Start of lesson:

**Server Prep**

Download Node onto the machine

Open cmd in the server file location

Type: node server.js

Npm install for package

**Sass prep:**

Install ruby

Using the cmd line:

If we cant install sass with certificate error run gem source -a http://rubygems.org/

Run the gem install sass command with ruby cmd line

Bootstrap:

Bootstrap 4 not 3 from now on

**Download and run MySQL database**

Password: 1234

Add mow to the schema as a new schema

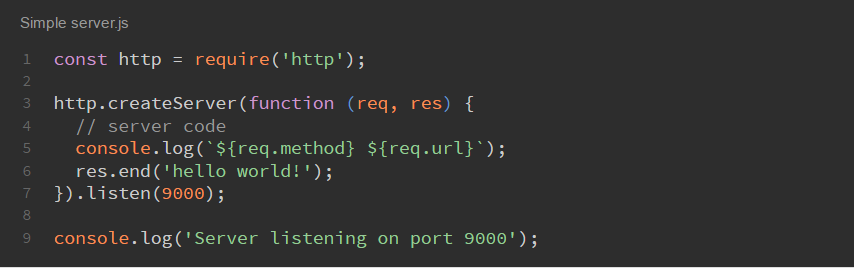
## Node.js

*“Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”*

#### Node.js Steps

<http://www.hongkiat.com/blog/node-js-server-side-javascript/>

Simple Node Server



Then open the browser on http://localhost:9000 and you will see the “hello world!” message.

Let’s explain what’s going on in the code. We are using the function http.createServer with a callback. This callback function is going to be called every time a client connects to the server. You can see that it takes two parameters: request and response.

The request contains the client’s information. For instance: requested URL, path, headers, HTTP method, and so forth.

The response object is used to reply to the client. You can set what you want to send back to the client. For instance, data, headers, etc.

Finally, the listening part. It allows you to set the port that you want your server to run on. In this case, we are using 9000.

## Package.json

Step 1: Make sure your package.json file is an actual JSON, not just a JavaScript object literal.

Step 2: name and version fields required (identifier)

Step 3:

Some rules:

* The name must be less than or equal to 214 characters. This includes the scope for scoped packages.
* The name can't start with a dot or an underscore.
* New packages must not have uppercase letters in the name.
* The name ends up being part of a URL, an argument on the command line, and a folder name. Therefore, the name can't contain any non-URL-safe characters.

A package.json file affords you a lot of great things:

It serves as documentation for what packages your project depends on.

It allows you to specify the versions of a package that your project can use using semantic versioning rules.

Makes your build reproducible which means that its way easier to share with other developers.

Requirements

As a bare minimum, a package.json must have:

"name"

all lowercase

one word, no spaces

dashes and underscores allowed

"version"

in the form of x.x.x

follows semver spec

For example:

{

"name": "my-awesome-package",

"version": "1.0.0"

}

Creating a package.json

To create a package.json run:

> npm init

This will initiate a command line questionnaire that will conclude with the creation of a package.json in the directory you initiated the command.

The --yes init flag

The extended CLI Q&A experience is not for everyone, and often if you are comfortable with using a package.json you'd like a more expedited experience.

You can get a default package.json by running npm init with the --yes or -y flag:

> npm init --yes

This will ask no questions, and instead will generate a default package.json using information extracted from the current directory.

> npm init --yes

Wrote to /home/ag\_dubs/my\_package/package.json:

{

"name": "my\_package",

"description": "",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"repository": {

"type": "git",

"url": "https://github.com/ashleygwilliams/my\_package.git"

},

"keywords": [],

"author": "",

"license": "ISC",

"bugs": {

"url": "https://github.com/ashleygwilliams/my\_package/issues"

},

"homepage": "https://github.com/ashleygwilliams/my\_package"

}

name: the current directory name

version: always 1.0.0

description: info from the readme, else an empty string ""

main: always index.js

scripts: by default creates a empty test script

keywords: empty

author: empty

license: ISC

bugs: info from the current directory, if present

homepage: info from the current directory, if present

You can also set several config options for the init command. Some useful ones:

> npm set init.author.email "wombat@npmjs.com"

> npm set init.author.name "ag\_dubs"

> npm set init.license "MIT"

NOTE:

If there is no description field in the package.json, npm uses the first line of the README.md or README instead. The description helps people find your package on npm search, so it's definitely useful to make a custom description in the package.json to make your package more discoverable.

## Node Passport

*“Passport is authentication middleware for Node. It is designed to serve a singular purpose: authenticate requests. When writing modules, encapsulation is a virtue, so Passport delegates all other functionality to the application. This separation of concerns keeps code clean and maintainable, and makes Passport extremely easy to integrate into an application.”*

#### Passport Steps

## Sequelize – MySQL, DAO (Data access object)

*“Sequelize is a promise-based ORM for Node.js v4 and up. It supports the dialects PostgreSQL, MySQL, SQLite and MSSQL and features solid transaction support, relations, read replication and more.”*

#### Sequelize Steps

## Angular

*“AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. AngularJS's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.”*

#### Angular Steps

## Socket.IO

*“Socket.IO is a JavaScript library for realtime web applications. It enables realtime, bi-directional communication between web clients and servers. It has two parts: a client-side library that runs in the browser, and a server-side library for Node.js. Both components have a nearly identical API. Like Node.js, it is event-driven.”*

*“Socket.IO primarily uses the WebSocket protocol with polling as a fallback option, while providing the same interface. Although it can be used as simply a wrapper for WebSocket, it provides many more features, including broadcasting to multiple sockets, storing data associated with each client, and asynchronous I/O.”*

#### Socket.IO Steps

# The Appendix

The appendix is used to place more information or detailed instructions on how to deal with specific issues. For instance in the above example if lost packets were detected when you ping the remote station you would refer to Appendix 3. Appendix 3 would detail what to steps to take when lost packets are detected.

By using this approach you keep your steps clean and easy to follow. You do not want to clutter them with detailed instructions on how to handle each problem or issue that may arise. Use the Appendix for that.