Node Js

Nikhil Jaunjal

INTRODUCTION

What is Node Js?

- Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine.
- Node.js was developed by Ryan Dahl in 2009.
- Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications.
- Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js uses an **event-driven**, **non-blocking I/O** model that makes it lightweight and efficient.

Node Js = Runtime JavaScript + JavaScript Library

Features

1. **Asynchronous and Event Driven :** All APIs of Node.js library are asynchronous, i.e non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

2. **Super-Fast:** Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.

3. **Single Threaded** but highly scalable Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

4. **No Buffering:** Node.js applications never buffer any data. These applications simply output the data in chunks.

5. **License:** Node.js is released under the MIT license.

Who are using

- Some of the well-known companies are: Netflix, PayPal, Uber, Go Daddy, Microsoft, IBM, Yahoo!, and Yammer, and the list go on...
- To check who are using Node JS, please click here (https://www.linkedin.com/pulse/top-5-companies-using-nodejs-production-anthony-delgado)

  Recommended: Watch the videos embedded on the page.

Where to use

Node.js is proving itself as a perfect technology partner in these areas:

- JSON APIs based Applications
- Single Page Applications
- I/O bound Applications
- Data Streaming Applications

Where not to use

It is not advisable to use Node.js for CPU intensive applications.

Installation

You can download the latest version of Node.js installable archive file from Node.js download page https://nodejs.org/en/download

Mac OSX

- Step 1: Open the Terminal app and type brew update. This updates Homebrew with a list of the latest version of Node.
  Type '**brew install node**' .

- Step 2: Sit back and wait, Homebrew has to download some files and install them. But that's it.

Ubuntu OS (Linux Machine)

- Step 1: Open the Terminal app and type **sudo apt-get install nodejs**.

- Step 2: Sit back and wait, Terminal has to download some files and install them. That's all.

Windows OS

- Step 1: Download the Windows installer from the https://nodejs.org/en/download

- Step 2: Run the installer(the .msi file you downloaded in the previous step).

- Step 3: Follow the prompts in the installer (Accept the license agreement, click the NEXT button a bunch of times and accept the default installation settings)

- Step 4: Restart any open command prompt for the change to take effect.

Verify installation:

- Step1: Open terminal and type **node –v** which results some version.

- Step 2: Type **npm -v** which results some version.

First Application

Let's run the basic JavaScript using Node.js runtime.

create a file eg. index.js

```
console.log("Hello Node World!");
```

Now open the Command Prompt and execute the index.js using Node.js interpreter to see the result:
node index.js

If everything is fine with your installation, it should produce the result as follows:

```
Hello Node World!
```

Second Application

Creating Server

```
var http = require('http');

http.createServer(function (req, res, next) {
    res.write('Hello World! running at port 1995');
    res.end();
}).listen(1995);
```

- The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client.
- The function passed into the http.createServer() method, will be executed when someone tries to access the computer on port 1995.
- run it with **node yourFile.js**
- now open url **localhost:1995** in browser.

Callback

- Callback is an asynchronous equivalent for a function.
- Callback function is called at the completion of a given task.
- Node makes heavy use of callbacks. All the APIs of Node are written in such a way that they support callbacks.

Example:

Consider a function whose task is read a file content. This function starts reading a file asynchronously and return the control to the execution environment immediately. Hence, next instruction can be executed.

Once file I/O is complete, it will call the callback function with the file content as a input parameter. So there is no blocking or wait for File I/O. This makes Node.js highly scalable, as it can process a high number of requests without waiting for any function to return results.

eg.

```
function fun1 (variable1, variable2, next) {

    var x = variable1 + 5;
    var y = variable2 + 5;

    console.log('calling callback..');
    next(x, y);
    console.log('after callback...');
}

function callback1(x, y){
 console.log('x = ' + x);
 console.log('y = '+ y);
}


fun1(1,2, callback1);
```

Note :
https://medium.com/javascript-in-plain-english/callbacks-in-node-js-how-why-when-ac293f0403ca

NPM

Node Package Manager (NPM) provides two main functionalities:

- Online repositories for node.js packages/modules which are searchable on https://www.npmjs.com/

- Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.

Note : Now, We don't have to separate install npm. NPM comes bundled with Node.js.

Let's create a project with some packages in it-

npm init

Package.json

Using package.json - package.json is present in the root directory of any Node application/module and is used to define the properties of a package.

Attributes of package.json :

name - name of the package
version - version of the package
description - description of the package
homepage - homepage of the package
author - author of the package
contributors - name of the contributors to the package
dependencies - list of dependencies. NPM automatically installs all the dependencies mentioned here in the node_modules folder of the package.
repository - repository type and URL of the package
main - entry point of the package
keywords – keywords
For example: You can check the package.json under express directory of node_modules.

Installing Modules

There is a simple syntax to install any Node.js module through npm:

npm install <module name> | npm install --save <module name>

Example: Command to install a famous Node.js web framework module called express is as follows:

npm install express

Now you can use this module in your js file as following:

const express = require('express');

Global v/s Local Installation:

- By default, NPM installs any dependency in the local mode.
- Local mode refers to the package installation in **node_modules** directory lying in the folder where Node application is present.
- All the installed packages are accessible via **require()** method.

Example:

When we installed express module, it created **node_modules** directory in the current directory which consist express module.
Try this command, and check the current directory for the result:

npm install express

After completing the installation of express, In the current directory, **node_modules** directory will be created and under that you can find the express directory.

Globally installed packages/dependencies are stored in system directory.

- Such dependencies can be used in CLI (Command Line Interface) function of any node.js but cannot be imported using require() in Node application directly.
- Now let's try installing the express module using global installation.

npm install -g express

This will produce similar result but the modules will be installed globally.