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.