**NODE.JS**

****

Node.js is a powerful open-source and cross-platform server-side javascript framework that was built on Google Chrome’s Javascript Engine V8 by Ryan Dahl in the year 2009. Originally, javascript was used mainly for client-side scripting but with Node.js it allowed developers to create javascripts via server-side.Thus, helping developers to produce a dynamic website content.

The filename extension “.js” is an indication that the file is written in javascript code but in Node.js the “.js” does not refer to any file in this manner because “Node.js” is just the name of the product.

Node.js processes request in a loop, called the “event loop” and has a rich library of Javascript modules that allows developers to create server-side tools and applications in Javascript.

**HISTORY**

In the year 2009, Node.js was initially released and introduced by Ryan Dahl, creator of node.js. The supported operating system of the node.js during this year was only Mac OS X and Linux.

In January 2010, node package manager (npm), a package manager, was introduced for the Node.js environment. In the same yea, Node.js 0.2.0 was also released.

In the year 2011, Node.js build finally supported windows operating system which was released in July 2011. In the same year, npm 1.0 and node.js guide was released.

Ryan Dahl left the company and promoted his coworker and the creator of npm which is Isaac Schuelter to manage the project in January 2012. Also in this year that node.js version 0.8.0 was released.

Schuelter announced the promotion of Timothy Fontaine making him the project leader in January 2014. Also in December 2014, a fork of node.js which is the io.js was created by Fedor Indutny.

In the year 2015, Joyent planned to establish a neutral and open governance Foundation of Node.js. By the month of june of the same year, Node.js and IO.js communities decided to work together as a team under the Node.js foundation.

In the year 2016, Express, one of the framework of node.js, was announced as it’s most popular web server framework and will become it’s new incubation project of the Node Foundation. The 2017 will serve as the year of adoption and mainstream for the Node.js.

**FEATURES OF NODE.JS**

1. Asynchronous - Node.js APIs library are asynchronous, it operates and runs process independently from other processes. Moreover, asynchronous operation is non-blocking that start an operation and immediately returns before the operation is complete.
2. Fast - Node,js library is fast in executing codes since it is built on Google Chrome’s V8 Javascript Engine.
3. Single Threaded Event Driven - Single thread model allows all request in the server to run on the same thread. This is efficient in terms of the server resources and speed of the application.
4. Buffer free - Node.js applications outputs data in chunks.
5. License - Node.js is under the license of MIT ([Massachusetts Institute of Technology](https://en.wikipedia.org/wiki/Massachusetts_Institute_of_Technology)).

**NODE PACKAGE MANAGER**

Node Package Manager (npm), as the name says is a package manager for node.js environment. Npm provides an online repository where Node.js projects is published. With the use of the repository it helps developers to connect with their team in sharing, updating, and solving particular problems in their project. Also, npm is a command-line utility for the developers to interact with the npm repository which will help them in package installation, dependency management, and version management.

Upon installing Node.js, npm is already included in it. You can verify it by issuing the command **“npm -v”** in the command prompt. If you want to have the latest version of the npm, you can update it by entering the command **“npm install npm -g”** in the command prompt. If you want to know more about npm, check their official website: https://www.npmjs.com

**NODE.JS MODULES**

Node.js modules can be considered as JavaScript libraries, set or collection of functions that you can integrate in your application. Modules in node.js can be separated in different .js file. Node.js has three types of modules namely core modules, local modules, and third party modules.

1. **Core Modules**

These modules are compiled into the binary and will load automatically upon starting Node.js process, but before that, you need first to import the core modules in order for you to use it in your project . Core modules are located and defined in Node.js source under the /lib folder.

The lists below are some of the core modules of Node.js:

|  |  |
| --- | --- |
| **Module** | **Description** |
| http | This module includes events, classes, and methods to make node.js act an HTTP server. |
| url | Url module is used for url parsing. |
| quertstring | A module that includes methods for handling query string. |
| path | A module that deals with file paths. |
| fs | Fs module is used in handling file systems. |
| util | This use of this module is for accessing the utility functions. |

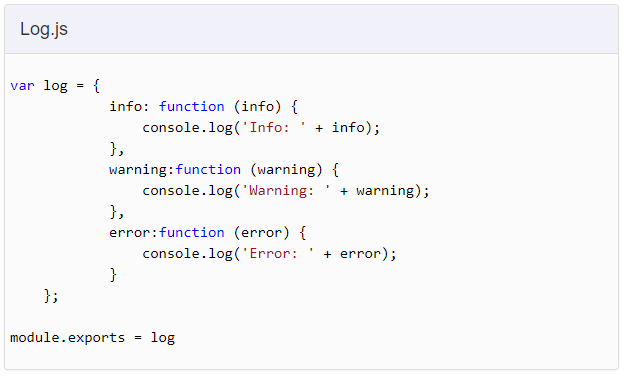
In order for you to run and use the core modules of node.js, import the module using the require() function.

Example: var module = require(‘moduleName’);

**2) Local Modules**

These modules are created locally in the Node.js application. Similar with the core modules local modules has different functionalities that you can integrate in your application in separate files and folders. Local modules can also be packaged and distributed via NPM (Node Package Manager) so that it can be used by the Node.js community. If you want to know about npm, see the section above about “What is an npm”.

The example below shows how to write a simple local module showing a login module that records information, error or warning to the console:



In the example shown above, we have created three functions of an object namely info(), warning(), and error(). And at the end, we assigned the object “log” to module.exports, a special object which is already included by default in Node.js application that is used to expose or export a function, variable or object in Node.js as a module.

In order for you to load and use the created local module in your application, similar to core module you need to import the modules using the require() function. But in this case you need to specify location or path of the created module.

Example: var logModule = require(‘./Log.js’);

logModule.info(‘Node.js started’);

And to run the created module

above using windows command prompt (for windows OS) shown below:



**3) External Modules**

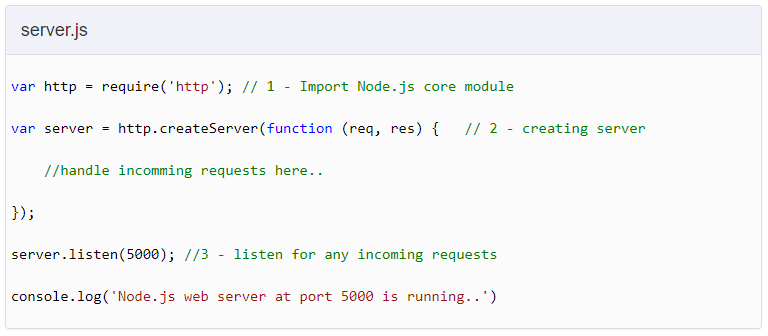
Node.js has a very big collection of open source modules that is provided by the Node.js community. These modules can be referred to as external modules. External modules can be integrated in our codebase but since it is not built-in modules, requiring it is not enough to integrate them in our project. But with the use of npm, we can install the codebase that contains the external module locally and with that we can now integrate and use those external modules in to our codebase.

**NODE.JS WEB SERVER**

In every website, you need a web server to access the web application. Web server focuses in handling all http request coming from a client or user of the website. In Node.js, it allows you to create a web server that can handle HTTP requests asynchronously.

**Creating Node.js Web Server**

The example below shows how to create a simple web server in Node.js that can process incoming requests asynchronously.

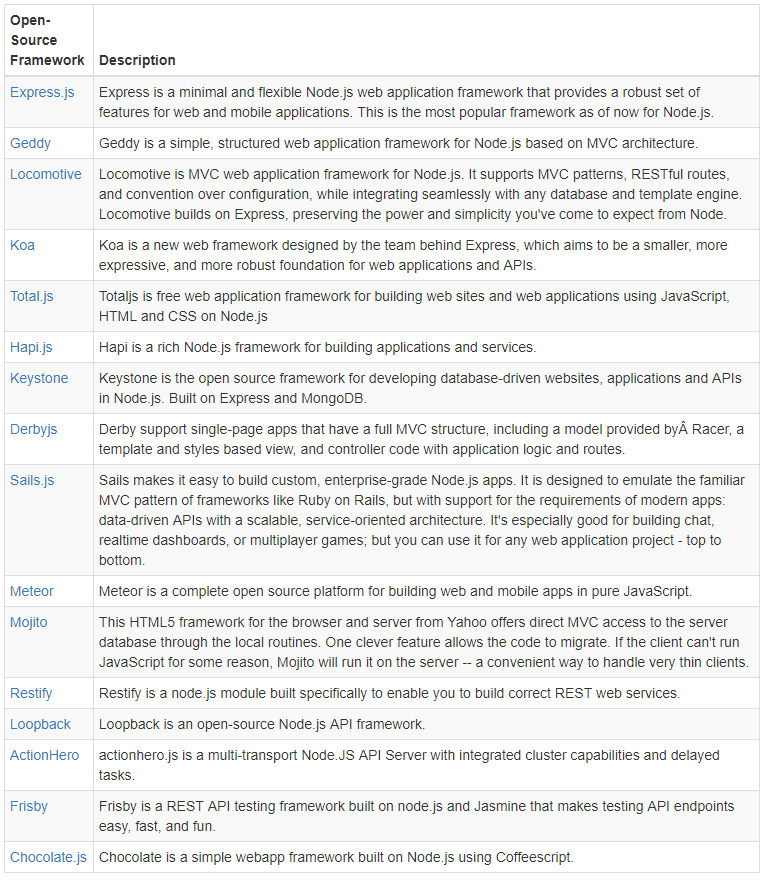


In the example shown above, we imported the module http, one of the core modules in Node.js, by using the require() function. The next thing that we did is we called createServer(), a method of the http, and we specified a callback function with the request and response parameter. The last step is we called listen() method and assigned a port number where the the program will start listening.

To check if your code is working, run the program by issuing the command “node server.js” in the command prompt and it will display the message that you wrote in console.log which is “Node.js web server at port 5000 is running..”.

**NODE.JS FRAMEWORKS**

In Node.js, there are a lot of framework that you can use and integrate into your web application which is available in NPM. With the use of these frameworks the development of your Node.js application is made faster and easier. You can choose on which framework to use in your application in the list below depending on what is the requirement or needed in your website:



**Express.js**

The Express.js is one of the most used and popular web application framework of Node.js. This framework has a lot of features that lessen the time production of your web application. The Express.js framework was based on one of the modules of Node.js which is the connect module.

**Installing Express.js**

Express.js framework can be installed using npm. To do that, issue the command **”npm install -g express”** in the command prompt. The command will install the latest express.js version globally, for it to be used by every Node.js application on your computer.

To install the latest version of express.js locally into your project folder, issue the command **“npm install express --save”** in the command prompt.