Node.js Tutorial – Learn basics of Node.js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js helps developers write JavaScript code to run on server side, to generate dynamic content and deliver to web client.

NodeJS Tutorial

Welcome to **NodeJS Tutorial**. Following these lucid tutorials helps you learn basic understanding of Node.js. With the progression of your learning, more advanced concepts to program and build server side applications and networking applications will be provided.

Introduction to Node.js

Node.js if free to use and is an open source server framework. Node.js runs on various platforms and some of them are Linux, Unix, Windows, Mac OS X, etc., which are widely used operating systems.

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. The two features that make Node.js stand-out are :

1. Event-driven
2. Non-blocking I/O model

To appreciate the features of Node.js and how it revolutionized the server side applications in terms of resource usage and event handling, we might need to take a sneak peak into the traditional model of web servers; and know how Node.js fills the performance voids.

In a traditional web server, when a web-client (browser or an application) makes a request, server takes the request and waits for the response to be prepared by file system/database/any other resource. Consider that your web service has become popular and number of web-client requests hitting the server is increasing day by day. But if you observe, the server is serving one request at a time and while the file-system/database is preparing the response, the server is being idle, instead of taking requests from other customers. Node.js

exploits this idle time with its event-driven framework. That is after taking a request, the server tries to prepare the response. And if there is any resource (file-system/database/etc) involved to prepare the response, server is let free and is informed through events when the response is ready. Mean while, instead of being idle during this time, server takes requests from other clients. When a response for a client is ready, an event is generated to let the server know the response is ready, which is in turn served to clients.

Programming Node.js Applications

Node.js is a platform built on JavaScript engine. Node.js applications could be developed using Java Scripting and most of the modern web applications are being built using Node.js. And this is also one of the important factor for Node.js to become popular as there is a wide community of JavaScript developers out there. They adopted Node.js, thus industry adopted Node.js quite easily.

Some of the capabilities of Node.js are :

- Node.js can dynamically generate content and present it to web clients
- Node.js can do file operations like read, write, update, delete, etc.
- Node.js can collect data from clients through forms.
- Node.js can connect to various databases like MySQL, MongoDB, etc., and perform respective DB operations.
- Node.js has builtin JSON module to work with JSON files.

Node.js Production Ready

Currently Node.js v8.x is the latest Long Term Supported version. So, most of the examples you come across in this tutorial are tried out on Node.js v8.x.

Node.js Use Cases

Following are some of the notable use cases that can take advantage of the Node.js Event driven model.

- Low latency provided by Node.js makes it suitable for Real-time applications like instant messaging, online-gaming and collaborative applications like Google Docs etc.
- Doing tasks asynchronously makes Node.js scale for a much higher traffic.
- Node.js can scale horizontally using its cluster mode. So, if you are looking for a scalable web application, Node.js is there.

Node.js Limitations

Following are some of the limitations of Node.js

- Node.js is not a good choice for applications that do heavy computation. This is because incoming requests are blocked during computation tasks that take relatively long time.
- Security provided by many of npm packages are not promising, even though core Node.js is quite stable. You may have to depend on third party tools to evaluate a package for any known vulnerabilities before using it in your application.

NodeJS Tutorial Index

Node.js Installation

Node.js can be installed in your local machine and you can get started with Node.js Application development.

- Tutorial – Install Node.js in PC

Node.js Modules

Modules are reusable parts of code that usually export specific objects to be used in your Node.js programs. Node.js provides many builtin modules. You may also create your own module or extend a builtin module, or override some of the functionalities of the module. We shall learn in detail.

- Tutorial – About Node.js Modules
- Tutorial – Create Node.js Module
- Tutorial – Publish Node.js Module
- Tutorial – Extend Node.js Module
- Tutorial – Manage Node.js Module

Node.js Buffers

Node.js Buffer is a class that helps to handle and work with octet streams; that come into picture when dealing with TCP data streams and file system operations.

- Tutorial – Create, Write to and Read Buffers in Node.js
- Tutorial – Find Node.js Buffer Length
- Tutorial – Convert JSON data to Buffer in Node.js
- Tutorial – Convert Array data to Buffer in Node.js

Node.js HTTP Module

- Tutorial – Create HTTP Web Server in Node.js
- Tutorial – Redirect URL in Node Server

Node.js FS (File System)

To handle file operations like creating, reading, deleting, etc., Node.js provides inbuilt module called FS (File System).

- Tutorial – About Node.js FS
- Tutorial – Node.js FS – Create a File
- Tutorial – Node.js FS – Read a File
- Tutorial – Node.js FS – Write to File
- Tutorial – Node.js FS – Delete a File
- Tutorial – Node.js FS Extra – Copy a Folder

Node.js Request module

Requests from different clients to the Node.js server can be handled using this request module. The request module contains routines to handle get requests, post requests, etc.

- Tutorial – Handle Get Requests using Request Node.js module

Node.js MySQL module

In most often cases, Node.js Server may require connection to a DB to store data received from clients or retrieve the data from database and serve the clients. And MySQL is popular among RDBMS. Following Node.js Tutorials help to get connected to MySQL database and do CRUD operations.

- Tutorial – About Node.js MySQL
- Tutorial – Node.js MySQL Connect DATABASE
- Tutorial – Node.js MySQL SELECT FROM query
- Tutorial – Node.js MySQL WHERE query
- Tutorial – Node.js MySQL ORDER BY query
- Tutorial – Node.js MySQL INSERT INTO query
- Tutorial – Node.js MySQL DELETE FROM
- Tutorial – Node.js MySQL Result Object

Node.js MongoDB

MongoDB is popular among NoSQL databases and it supports high volumes of data. If you plan to build a service or some sort of that, where you can expect high throughput of data, following tutorials help you in integrating MongoDB with your Node.js application.

- Tutorial – Node.js MongoDB
- Tutorial – Node.js MongoDB – Connection
- Tutorial – Node.js MongoDB – Create Database
- Tutorial – Node.js MongoDB – Drop Database
- Tutorial – Node.js MongoDB – Create Collection
- Tutorial – Node.js MongoDB – Delete Collection
- Tutorial – Node.js MongoDB – Insert Documents

Node.js JSON

Nowadays, JSON is the format of data that is mostly being exchanged between client and server, and also JavaScript has in built JSON tools. Following Node.js tutorials help you with the example to parse a JSON file or object and write the JSON object to a file (if required).

- Tutorial – Node.js Parse JSON File
- Tutorial – Node.js Write JSON Object to File

Node.js Helpful Examples

Following are some of the example Node.js scripts that help as reference for some of the server-client tasks.

- Tutorial – Upload files to Node.js Server from web-client

Node.js Examples

- Node.js Examples

Node.js Interview Questions

We have gathered some important interview questions for basic, intermediate and advanced levels. Each question has detailed answer and examples whenever necessary.

- Node.js Interview Questions

Conclusion

Concluding this **Node.js Tutorial**, we got introduced to Node.js, its capabilities, and different modules that are popular with Node.js. Follow these series of Node.js Tutorials to get hands on working with Node.js basic and advanced concepts.

- Node.js - Convert JSON to Buffer
- Node.js - Array to Buffer

Node.js HTTP

- Node.js - Create HTTP Web Server
- Node.js - Redirect URL

Node.js MySQL

- Node.js MySQL
- Node.js MySQL - Connect to MySQL Database
- Node.js MySQL - SELECT FROM
- Node.js MySQL - SELECT WHERE
- Node.js MySQL - ORDER BY
- Node.js MySQL - INSERT INTO
- Node.js MySQL - UPDATE
- Node.js MySQL - DELETE
- Node.js MySQL - Result Object

Node.js MongoDB

- Node.js MongoDB
- Node.js - Connect to MongoDB
- Node.js - Create Database in MongoDB
- Node.js - Drop Database in MongoDB
- Node.js - Create Collection in MongoDB
- Node.js - Delete Collection in MongoDB
- Node.js - Insert Documents to MongoDB Collection
- MongoError: failed to connect to server

Node.js Mongoose

- Node.js Mongoose Tutorial
- Node.js Mongoose - Installation
- Node.js Mongoose - Connect to MongoDB
- Node.js Mongoose - Define a Model
- Node.js Mongoose - Insert Single Document to MongoDB
- Node.js Mongoose - Insert Multiple Documents to MongoDB

Node.js URL

- Node.js - Parse URL parameters

Node.js FS (File System)

- Node FS
- Node FS - Read a File
- Node FS - Create a File
- Node FS - Write to a File
- Node FS - Append to a File
- Node FS - Rename a File
- Node FS - Delete a File
- Node FS Extra - Copy a Folder

Node.js JSON

- Node.js Parse JSON
- Node.js Write JSON Object to File

Node.js Error Handling

- Node.js Try Catch

Node.js Examples

- Node.js Examples
- Node.js - Handle Get Requests
- Node.js Example - Upload files to Node.js server

Useful Resources

- Node.js Interview Questions
- How to Learn Programming