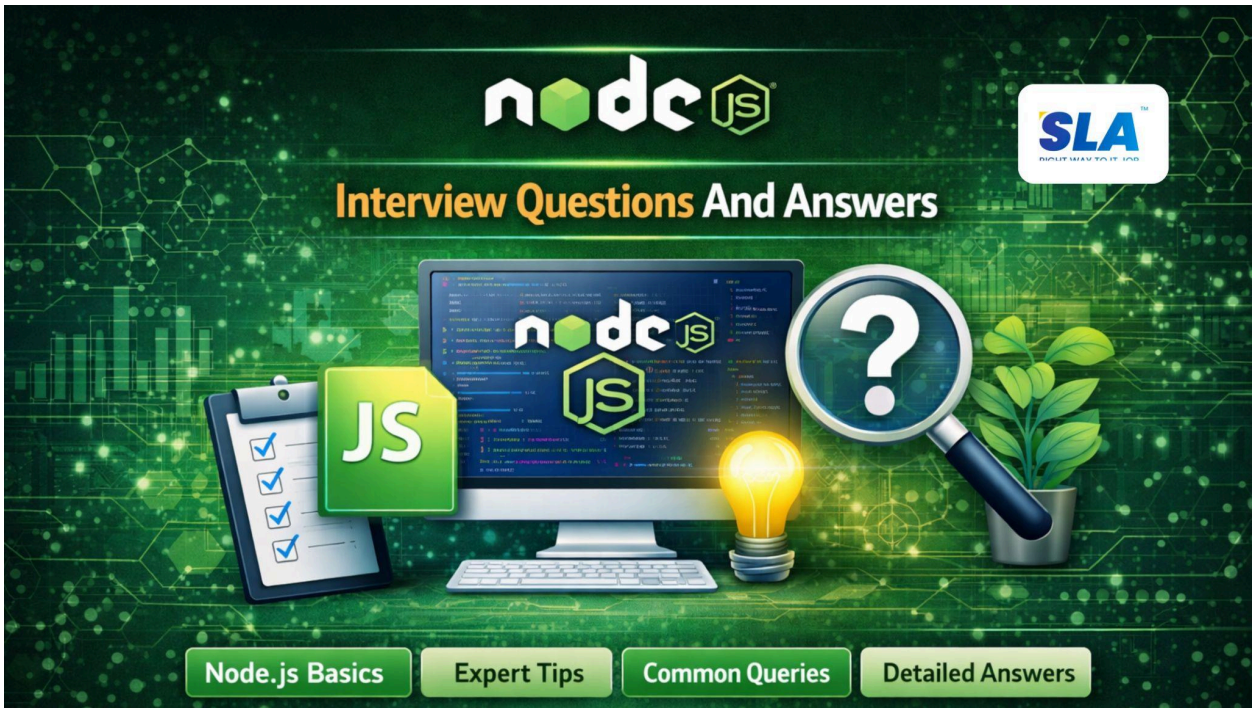

NodeJS Interview Questions and Answers

Introduction

Being proficient in Node.js is essential in building fast, scalable, and data-intensive network applications. This article, which contains Node.js interview questions and answers, will walk you through the core concepts of Node.js, which include the Event Loop, Non-blocking I/O, Middlewares, and Package Management with NPM. Being proficient in Node.js will allow you to build high-performance backend applications, making you an essential resource in today's fast-paced full-stack or backend development teams.
Want to become a certified backend developer? Want to master server-side JavaScript? Enroll in our professional Nodejs Certification Course in Chennai today and start building high-speed, scalable web applications.

List of NodeJS Interview Questions for Freshers

1. What is Node.js?
2. What does a JavaScript first-class function mean?
3. How do you handle packages in Node.js?
4. Why is Node.js single-threaded?

5. What is a callback in Node.js?
6. In what sense would you define I/O?
7. What is the use of the module ".Exports"?
8. Which database is used with Node.js more frequently?
9. Which Node.js command is used to import external libraries?
10. What does a Node.js event loop do?
11. How may a basic Express.js application be made?
12. What does Node.js' REPL mean?

Get started with our NodeJS course syllabus.

NodeJS Interview Questions and Answers for Freshers

1. What is Node.js?

A JavaScript engine called Node.js runs JavaScript code outside of a browser. It is scalable and typically used to construct the application's backend.

2. What does a JavaScript first-class function mean?

First-class functions are those that can be utilized just like any other variable. Numerous other programming languages are similar to this, such as JavaScript, Scala, Haskell, etc.

Due to this function, a function can now return a higher-order function or be supplied as a parameter to a callback function. Commonly used higher-order functions are map() and filter().

3. How do you handle packages in Node.js?

Several package installers and their corresponding configuration files can handle it. Use yarn or npm for the most part.

Both offer enhanced capabilities for managing environment-specific configurations that are available to practically all JavaScript libraries.

We utilize package.json and package-lock.json to maintain versions of the libraries installed in a project so that migrating that application to a different environment won't be a problem.

4. Why is Node.js single-threaded?

Node.js supports async processing using a single thread. Compared to the usual thread-based design, additional performance and scalability can be obtained by performing async processing on a single thread under typical web loads.

5. What is a callback in Node.js?

Once a task is completed, a callback function is triggered. It avoids blocking and permits the execution of other programs in the interim. Because Node.js is an asynchronous platform, callbacks are very important. Node's APIs are all designed to support callbacks.

6. In what sense would you define I/O?

Any program, procedure, or apparatus that transports data to or from one medium to another is referred to as I/O.

A transfer is always an input into one medium and an output into another. A physical device, a network, or files inside a system can all serve as the medium.

7. What is the use of the module ".Exports"?

When transferring all pertinent functions into a single file, a module in Node.js can be used to parse all connected code into a single unit of code. A module can be exported together with its function, allowing it to be imported into another project with the necessary keyword.

8. Which database is used with Node.js more frequently?

The most popular database used with Node.js is MongoDB. It is a document-oriented, cross-platform, NoSQL database with simple scalability, high availability, and excellent performance.

9. Which Node.js command is used to import external libraries?

Utilizing the “require” command allows you to import external libraries. Using “var http=require (“HTTP”)” as an example, by using the HTTP variable, this will load the HTTP library and the single exported object.

10. What does a Node.js event loop do?

Asynchronous callbacks in Node.js are controlled via event loops. It is one of the most crucial environmental elements since it forms the basis of Node.js’s non-blocking input/output.

11. How may a basic Express.js application be made?

The request object, which has attributes for the request query text, parameters, body, HTTP headers, and more, represents the HTTP request.

When an Express application receives an HTTP request, it responds with an HTTP object.

12. What does Node.js’ REPL mean?

“*Read Eval Print Loop,”* or REPL, is a symbol representing a computer environment. It is comparable to entering commands in a Unix/Linux shell or Windows console. The system then produces an output in response.

Develop your web development skills with our NodeJS tutorial for beginners.

List of NodeJS Interview Questions for Experienced

1. What kinds of HTTP requests are there?
2. What distinguishes synchronous from asynchronous functions?
3. How can you use Node.js to write Hello World?
4. Describe a few clustering techniques for Node.js.
5. Explain the various types of Node.js streams.

6. How can you use Node.js to read command-line arguments?
7. In Node.js, how are environment variables handled?
8. Describe the Redis module for Node.js.

Explore our NodeJS project ideas to learn more.

NodeJS Technical Interview Questions and Answers for Experienced

1. What kinds of HTTP requests are there?

A collection of request methods defined by HTTP is used to carry out specific tasks. Among the requested techniques are:

- **GET:** Data retrieval method
- **POST:** Typically used to modify the server's status or responses
- **HEAD:** Requests a response without including the response body, akin to the GET method.
- **DELETE:** This command removes the specified resource.

2. What distinguishes synchronous from asynchronous functions?

Writing scalable Node.js applications requires the use of asynchronous functions, which permit other code to run concurrently with their own execution, while synchronous functions stop it until they are finished.

3. How can you use Node.js to write Hello World?

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

// Create a server object

http.createServer(function (req, res) {

  res.write('Hello World!');

  res.end();
```

}).listen(3000);

Use the command line to launch this application, and the browser window will display the results. When a browser sends a request through http://localhost:3000/, this application prints Hello World in the browser.

4. Describe a few clustering techniques for Node.js.

- **Fork():** It splits the master process into a new child process. If the current process is master, the isMaster function returns true; otherwise, it returns false.
- **isWorker:** It returns true if the process that is now running is a worker; otherwise, it returns false.
- **process:** It gives back the global kid process.
- **send ():** It transmits a message from the worker to the master or the other way around.
- **kill():** This function terminates the active worker.

5. Explain the various types of Node.js streams.

- **Readable Stream:** A stream that allows you to receive and read data in an organized manner is known as a readable stream. You are not permitted to send anything, though. For instance, we can read a file's contents using fs.createReadStream().
- **Writable Stream:** A writeable stream allows you to transfer data in an ordered manner, but it prevents you from receiving it back. For instance, we can write data to a file using fs.createWriteStream().
- **Duplex Stream:** A duplex stream may be written to and read from simultaneously. You can therefore send and receive data at the same time. For instance.TCP is used by this socket.
- **Transform stream:** As data is read, this stream is utilized to change or transform it. In essence, the converted stream is a duplex. For instance, gzip is used to compress the data using zlib.createGzip stream.

6. How can you use Node.js to read command-line arguments?

When an application is executing through the operating system's command line interface, it can receive additional information from a program via strings of text known as command-line arguments (CLI).

The global object in the node, or the process object, makes it simple for us to read these arguments. Here's how it works:

Step 1: Save a file as index.js, then open it and insert the code below.

```
let arguments = process.argv ;

console.log(arguments) ;
```

Step 2: Use the following command to launch the index.js file:

```
node index.js
```

7. In Node.js, how are environment variables handled?

Environment variables in Node.js are controlled by process.env. Keys and environment configurations that we can supply are contained in the.env file.

The syntax "process.env.VARIABLE_NAME" is used to access the variable within the application. To utilize it, use the following command to install the dotenv package:

```
npm install dotenv
```


8. Describe the Redis module for Node.js.

- An open-source data structure storage system is called Redis. It can be used in numerous ways. It performs the roles of message broker, database, and cache.

- Data structures like strings, hashes, sets, sorted sets, bitmaps, indexes, and streams can all be stored in it.
- Redis is a great tool for Node.js developers since it minimizes cache size, which improves application performance. Still, Redis integration with Node.js apps is rather simple.

Conclusion

To pass the Node.js interview, you will be expected to show off your expertise at a sophisticated level of the Event Loop, Asynchronous programming, and handling of streams. The employer will want to interview someone who not only knows how to use Express.js to manage routes but also to manage Worker Threads, Buffer operations, and Middleware Optimization. Are you ready to become a certified backend development expert? Learn the art of scalable backend development with our software training institute in Chennai.