What is Node.js?

* Node.js is an open source server environment
* Node.js is free
* Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
* Node.js uses JavaScript on the server

Why Node.js?

**Node.js uses asynchronous programming!**

A common task for a web server can be to open a file on the server and return the content to the client.

Here is how PHP or ASP handles a file request:

1. Sends the task to the computer's file system.
2. Waits while the file system opens and reads the file.
3. Returns the content to the client.
4. Ready to handle the next request.
5. It is generally fast
6. It rarely blocks
7. It offers a unified programming language and data type
8. Everything is asynchronous
9. It yields great concurrency

Here is how Node.js handles a file request:

1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting, and simply continues with the next request.

Node.js runs single-threaded, non-blocking, asynchronous programming, which is very memory efficient.

What Can Node.js Do?

* Node.js can generate dynamic page content
* Node.js can create, open, read, write, delete, and close files on the server
* Node.js can collect form data
* Node.js can add, delete, modify data in your database

What is a Node.js File?

* Node.js files contain tasks that will be executed on certain events
* A typical event is someone trying to access a port on the server
* Node.js files must be initiated on the server before having any effect
* Node.js files have extension ".js"

## What is NPM?

NPM is a package manager for Node.js packages, or modules if you like.

[www.npmjs.com](https://www.npmjs.com/) hosts thousands of free packages to download and use.

The NPM program is installed on your computer when you install Node.js

NPM is already ready to run on your computer!

## What is a Package?

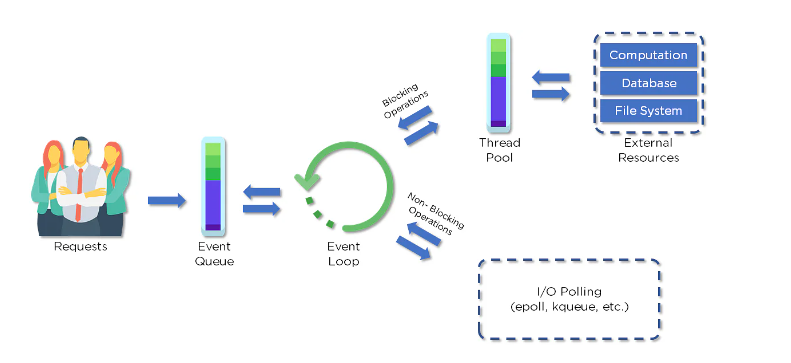
A package in Node.js contains all the files you need for a module.

Modules are JavaScript libraries you can include in your project.

|  |  |  |
| --- | --- | --- |
| Package.json | The package. json file is the heart of any Node project. It records important metadata about a project which is required before publishing to NPM, and also defines functional attributes of a project that npm uses to install dependencies, run scripts, and identify the entry point to our package. | npm init  npm init -y |
| .env file |  |  |
|  |  |  |

### **3. How does Node.js work?**

A web server using Node.js typically has a workflow that is quite similar to the diagram illustrated below. Let’s explore this flow of operations in detail.



* Clients send requests to the webserver to interact with the web application. Requests can be non-blocking or blocking:
* Querying for data
* Deleting data
* Updating the data
* Node.js retrieves the incoming requests and adds those to the Event Queue
* The requests are then passed one-by-one through the Event Loop. It checks if the requests are simple enough not to require any external resources
* The Event Loop processes simple requests (non-blocking operations), such as I/O Polling, and returns the responses to the corresponding clients
* A single thread from the Thread Pool is assigned to a single complex request. This thread is responsible for completing a particular blocking request by accessing external resources, such as computation, database, file system, etc.
* Once the task is carried out completely, the response is sent to the Event Loop that sends that response back to the client.

### **4. Why is Node.js Single-threaded?**

Node.js is single-threaded for async processing. By doing async processing on a single-thread under typical web loads, more performance and scalability can be achieved instead of the typical thread-based implementation.

### **5.If Node.js is single-threaded, then how does it handle concurrency?**

The Multi-Threaded Request/Response Stateless Model is not followed by the Node JS Platform, and it adheres to the Single-Threaded Event Loop Model. The Node JS Processing paradigm is heavily influenced by the JavaScript Event-based model and the JavaScript callback system. As a result, Node.js can easily manage more concurrent client requests. The event loop is the processing model's beating heart in Node.js.

### **6. Explain callback in Node.js.**

A callback function is called after a given task. It allows other code to be run in the meantime and prevents any blocking.  Being an asynchronous platform, Node.js heavily relies on callback. All APIs of Node are written to support callbacks.

### **7. What are the advantages of using promises instead of callbacks?**

* The control flow of asynchronous logic is more specified and structured.
* The coupling is low.
* We've built-in error handling.
* Improved readability.

### **9. How is Node.js most frequently used?**

Node.js is widely used in the following applications:

1. Real-time chats
2. Internet of Things
3. Complex SPAs (Single-Page Applications)
4. Real-time collaboration tools
5. Streaming applications
6. Microservices architecture

### **12. What are the modules in Node.js?**

Modules are like JavaScript libraries that can be used in a Node.js application to include a set of functions. To include a module in a Node.js application, use the **require()** function with the parentheses containing the module's name.

Node.js has many modules to provide the basic functionality needed for a web application. Some of them include:

|  |  |
| --- | --- |
| Core Modules | Description |
| HTTP | Includes classes, methods, and events to create a Node.js HTTP server |
| util | Includes utility functions useful for developers |
| fs | Includes events, classes, and methods to deal with file I/O operations |
| url | Includes methods for URL parsing |
| query string | Includes methods to work with query string |
| stream | Includes methods to handle streaming data |
| zlib | Includes methods to compress or decompress files |

### **13. What is the purpose of the module .Exports?**

In Node.js, a module encapsulates all related codes into a single unit of code that can be parsed by moving all relevant functions into a single file. You may export a module with the module and export the function, which lets it be imported into another file with a needed keyword.

### **17. What are some of the most commonly used libraries in Node.js?**

There are two commonly used libraries in Node.js:

* [**ExpressJS**](https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-express-js)- Express is a flexible Node.js web application framework that provides a wide set of features to develop web and mobile applications.
* **Mongoose** - [Mongoose](https://www.simplilearn.com/tutorials/nodejs-tutorial/nodejs-mongodb) is also a Node.js web application framework that makes it easy to connect an application to a database.

### **21. What is an Event Loop in Node.js?**

Event loops handle asynchronous callbacks in Node.js. It is the foundation of the non-blocking input/output in Node.js, making it one of the most important environmental features.

### **24. What are the two types of API functions in Node.js?**

The two types of API functions in Node.js are:

* Asynchronous, non-blocking functions
* Synchronous, blocking functions

### **Explain asynchronous and non-blocking APIs in Node.js.**

* All Node.js library APIs are asynchronous, which means they are also non-blocking
* A Node.js-based server never waits for an API to return data. Instead, it moves to the next API after calling it, and a notification mechanism from a Node.js event responds to the server for the previous API call

### **33. How do we implement async in Node.js?**

As shown below, the async code asks the JavaScript engine running the code to wait for the request.get() function to complete before moving on to the next line for execution.



### **34. What is a callback function in Node.js?**

A callback is a function called after a given task. This prevents any blocking and enables other code to run in the meantime.

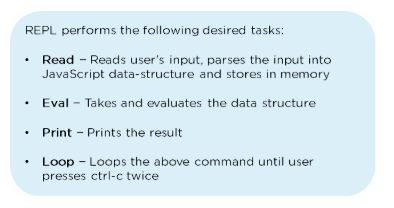
In the last section, we will now cover some of the advanced-level Node.js interview questions.

## Node.js Interview Questions and Answers For Experienced Professionals

This section will provide you with the Advanced Node.js interview questions which will primarily help experienced professionals.

### **35. What is REPL in Node.js?**

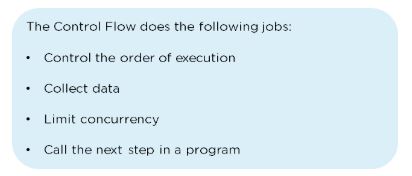
REPL stands for Read Eval Print Loop, and it represents a computer environment. It’s similar to a Windows console or Unix/Linux shell in which a command is entered. Then, the system responds with an output



### **36. What is the control flow function?**

The control flow function is a piece of code that runs in between several asynchronous function calls.

### **37. How does control flow manage the function calls?**



### **39. What is the buffer class in Node.js?**

Buffer class stores raw data similar to an array of integers but corresponds to a raw memory allocation outside the V8 heap. Buffer class is used because pure JavaScript is not compatible with binary data

### **40. What is piping in Node.js?**

Piping is a mechanism used to connect the output of one stream to another stream. It is normally used to retrieve data from one stream and pass output to another stream

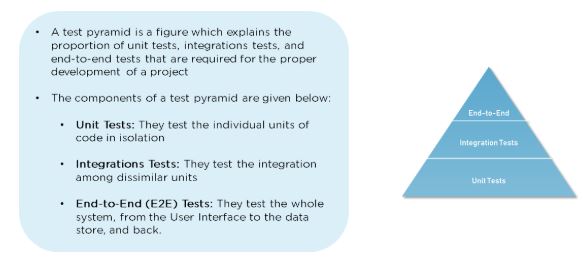
### **43. What is callback hell?**

* Callback hell, also known as the pyramid of doom, is the result of intensively nested, unreadable, and unmanageable callbacks, which in turn makes the code harder to read and debug
* improper implementation of the asynchronous logic causes callback hell

### **44. What is a reactor pattern in Node.js?**

A reactor pattern is a concept of non-blocking I/O operations. This pattern provides a handler that is associated with each I/O operation. As soon as an I/O request is generated, it is then submitted to a demultiplexer

### **45. What is a test pyramid in Node.js?**



### **48. Explain the concept of middleware in Node.js.**

Middleware is a function that receives the request and response objects. Most tasks that the middleware functions perform are:

* Execute any code
* Update or modify the request and the response objects
* Finish the request-response cycle
* Invoke the next middleware in the stack

### **49. What are the different types of HTTP requests?**

HTTP defines a set of request methods used to perform desired actions. The request methods include:

**GET:**Used to retrieve the data

**POST:**Generally used to make a change in state or reactions on the server

**HEAD:**Similar to the GET method, but asks for the response without the response body

**DELETE:** Used to delete the predetermined resource

### **50. How would you connect a MongoDB database to Node.js?**

To create a database in MongoDB:

* Start by creating a MongoClient object
* Specify a connection URL with the correct IP address and the name of the database you want to create

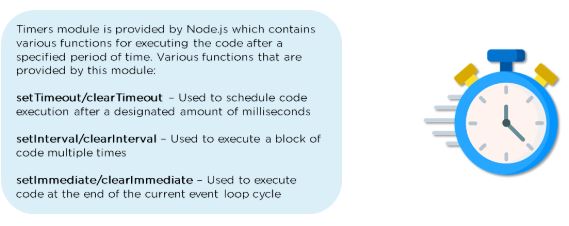


### **51. What is the purpose of NODE\_ENV?**

### **node-env**

### **52. List the various Node.js timing features.**

As you prepare for your upcoming job interview, we hope that this comprehensive guide has provided more insight into what types of questions you’ll be asked.



### 1. What is a first class function in Javascript?

When functions can be treated like any other variable then those functions are first-class functions. There are many other programming languages, for example, scala, Haskell, etc which follow this including JS. Now because of this function can be passed as a param to another function(callback) or a function can return another function(higher-order function). map() and filter() are higher-order functions that are popularly used.

### 3. How do you manage packages in your node.js project?

It can be managed by a number of package installers and their configuration file accordingly. Out of them mostly use npm or yarn. Both provide almost all libraries of javascript with extended features of controlling environment-specific configurations. To maintain versions of libs being installed in a project we use package.json and package-lock.json so that there is no issue in porting that app to a different environment.

### 7. What are the advantages of using promises instead of callbacks?

The main advantage of using promise is you get an object to decide the action that needs to be taken after the async task completes. This gives more manageable code and avoids callback hell.

### 8. What is fork in node JS?

A fork in general is used to spawn child processes. In node it is used to create a new instance of v8 engine to run multiple workers to execute the code

### 11. How many types of API functions are there in Node.js?

There are two types of API functions:

* **Asynchronous, non-blocking functions** - mostly I/O operations which can be fork out of the main loop.
* **Synchronous, blocking functions** - mostly operations that influence the process running in the main loop.

### 10. How do you create a simple server in Node.js that returns Hello World?

**var** http = require("http");

http.createServer(**function** (request, response) {

response.writeHead(200, {'Content-Type': 'text/plain'});

response.end('Hello World\n');

}).listen(3000);

### 13. List down the two arguments that async.queue takes as input?

* Task Function
* Concurrency Value

### 14. What is the purpose of module.exports?

This is used to expose functions of a particular module or file to be used elsewhere in the project. This can be used to encapsulate all similar functions in a file which further improves the project structure.  
  
For example, you have a file for all utils functions with util to get solutions in a different programming language of a problem statement.

const getSolutionInJavaScript = async ({

problem\_id

}) => {

...

};

const getSolutionInPython = async ({

problem\_id

}) => {

...

};

module.exports = { getSolutionInJavaScript, getSolutionInPython }

Thus using module.exports we can use these functions in some other file:

const { getSolutionInJavaScript, getSolutionInPython} = require("./utils")

### 15. What tools can be used to assure consistent code style?

ESLint can be used with any IDE to ensure a consistent coding style which further helps in maintaining the codebase.

## Intermediate Node.js Interview Questions

### 16. What do you understand by callback hell?

async\_A(function(){

async\_B(function(){

async\_C(function(){

async\_D(function(){

....

});

});

});

});

For the above example, we are passing callback functions and it makes the code unreadable and not maintainable, thus we should change the async logic to avoid this