**1) What is Node.js?**

Node.js is a very powerful JavaScript based platform or framework which is built on Google Chrome's JavaScript V8 Engine.

**2) Why would you want to use Node.js?**

It is used to develop I/O intensive web applications like video streaming sites, single page applications (SPA) and other web applications. Node.js is open source and used by thousands of developers around the world.

**3. How do you update NPM to a new version in Node.js?**

You use the following commands to update NPM to a new version:

$ sudo npm install npm -g

/usr/bin/npm -> /usr/lib/node\_modules/npm/bin/npm-cli.js

npm@2.7.1 /usr/lib/node\_modules/npm

**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 as opposed to the typical thread-based implementation.

**5. Explain callback in Node.js.**

A callback function is called at the completion of a given task. This 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.

**6. How do you prevent/fix callback hell?**

The three ways to prevent/fix callback hell are:

• Handle every single error

• Keep your code shallow

• Modularize – split the callbacks into smaller, independent functions that can be called with some parameters then joining them to achieve desired results.

The first level of improving the code above might be:

var logError = function(error){

if(!error){

console.log("success!!");

}else{

console.log("error");

}

},

updateTransaction = function(t){

query("UPDATE transactions SET value = " + (t.value\*0.1) + " WHERE id=" + t.id, logError);

},

handleTransactions = function(transactions){

transactions.each(updateTransaction);

},

handleClient = function(id){

query("SELECT \* FROM transactions WHERE clientId=" + id, handleTransactions);

};

query("SELECT clientId FROM clients WHERE clientName='picanteverde';",handleClient);

You can also use Promises, Generators and Async functions to fix callback hell.

7. **Name the types of API functions in Node.js.**

There are two types of functions in Node.js.:

• Blocking functions - In a blocking operation, all other code is blocked from executing until an I/O event that is being waited on occurs. Blocking functions execute synchronously

For example:

const fs = require('fs');

const data = fs.readFileSync('/file.md'); // blocks here until file is read

console.log(data);

// moreWork(); will run after console.log

The second line of code blocks the execution of additional JavaScript until the entire file is read. moreWork () will only be called after Console.log

• Non-blocking functions - In a non-blocking operation, multiple I/O calls can be performed without the execution of the program being halted. Non-blocking functions execute asynchronously.

• For example:

const fs = require('fs');

fs.readFile('/file.md', (err, data) => {

if (err) throw err;

console.log(data);

});

// moreWork(); will run before console.log

Since fs.readFile () is non-blocking, moreWork () does not have to wait for the file read to complete before being called. This allows for higher throughput.

**8. Which is the first argument typically passed to a Node.js callback handler?**

Typically, the first argument to any callback handler is an optional error object. The argument is null or undefined if there is no error.

Error handling by a typical callback handler could be as follows:

function callback(err, results) {

// usually we'll check for the error before handling results

if(err) {

// handle error somehow and return

}

// no error, perform standard callback handling

}

**9. What are the functionalities of NPM in Node.js?**

NPM (Node package Manager) provides two functionalities:

• Online repository for Node.js packages

• Command line utility for installing packages, version management and dependency management of Node.js packages

**10. What is the difference between Node.js and Ajax?**

Node.js and Ajax (Asynchronous JavaScript and XML) are the advanced implementation of JavaScript. They all serve completely different purposes.

Ajax is primarily designed for dynamically updating a particular section of a page’s content, without having to update the entire page.

Node.js is used for developing client-server applications.

**11) Explain REPL in Node.js?**

REPL stands for Read Eval Print Loop. Node.js comes with bundled REPL environment which performs the following desired tasks –

• Eval

• Print

• Loop

• Read

**12) What is the latest version of Node.js available?**

Latest version of Node.js is - v0.10.36.

**13) List out some REPL commands in Node.js?**

Below are the list of REPL commands –

• Ctrl + c - For terminating the current command.

• Ctrl + c twice – For terminating REPL.

• Ctrl + d - For terminating REPL.

• Tab Keys - list of all the current commands.

• .break - exit from multiline expression.

• .save with filename - save REPL session to a file.

**14) Explain NPM in Node.js?**

NPM stands for Node Package Manager (npm) and there are two functionalities which NPM takes care of mainly and they are –

• Online repositories for node.js modules or packages, which can be searched on search.nodejs.org

• Dependency Management, Version Management and command line utility for installing Node.js packages.

**15) Explain global installation of dependencies?**

Globally installed dependencies or packages are stored in <user-directory>/npm directory and these dependencies can be used in Command Line Interface function of any node.js.

**16) Explain local installation of dependencies?**

By default npm will install the dependency in the local mode. Here local mode refers to the package installation in node\_modules directory lying in the folder where Node application is present. “require ()” is used to access the locally deployed packages.

**17) Explain Package.JSON?**

This will be present in the root directory of any Node module/application and will be used to define the properties of a package.

**18) Explain “Callback hell”?**

“Callback hell” will be referred to heavily nested callbacks which has become unreadable or unwieldly.

**19) What you mean by chaining in Node.JS?**

It’s a mechanism in which output of one stream will be connected to another stream and thus creating a chain of multiple stream operations.

**20) Explain “Path” module in Node.JS?**

“Path” module will be used for transforming and handling file paths. Below is the syntax of path module –

var mypath = require("path")

**21) Explain “Path” module in Node.JS?**

“Path” module will be used for transforming and handling file paths. Below is the syntax of path module –

var mypath = require("path")

**22) What are the advantages of NodeJS?**

Below are the list of advantages of NodeJS –

• Javascript – It’s a javascript which can be used on frontend and backend.

• Community Driven - NodeJS has great open source community which has developed many excellent modules for NodeJS to add additional capabilities to NodeJS applications.

**23) What you mean by JSON?**

JavaScript Object Notation (JSON) is a practical, compound, widely popular data exchange format. This will enable JavaScript developers to quickly construct APIs.

**24) Explain “Buffer class” in Node.JS?**

It is a global class which can be accessed in an application without importing buffer modules.

**25) Why to use “SetTimeout” in Node.JS?**

This is the global function and it is used to run the callback after some milliseconds.

Syntax of this method –

setTimeout(callbackmethod, millisecs)

sdfsdf

**26) Explain “Path” module in Node.JS?**

“Path” module will be used for transforming and handling file paths. Below is the syntax of path module –

var mypath = require("path")