Top 50 Node.js Interview Questions & Answers

1. What is Node.js?

Node.js is a JavaScript runtime built on Chrome's V8 engine. It allows execution of JavaScript outside the browser.

2. What are the main features of Node.js?

Asynchronous and event-driven, single-threaded but scalable, fast execution with V8, non-blocking I/O, cross-platform.

3. What is NPM?

NPM (Node Package Manager) is the default package manager for Node.js that helps manage libraries and dependencies.

4. Explain the difference between Node.js and JavaScript.

JavaScript is a programming language; Node.js is a runtime environment to execute JavaScript outside the browser.

5. What are advantages of Node.js?

High performance, scalability, large community, full-stack JavaScript, event-driven architecture.

6. What is the event loop in Node.js?

It's the mechanism that handles asynchronous operations, allowing Node.js to perform non-blocking I/O operations.

7. What is REPL in Node.js?

REPL stands for Read-Eval-Print-Loop. It allows developers to test code snippets interactively.

8. What are streams in Node.js?

Streams are used to handle continuous data flow efficiently. Types: Readable, Writable, Duplex, Transform.

9. What is the difference between require() and import?

require() is CommonJS, import is ES6 module system. import is more modern and preferred.

10. What is middleware in Node.js?

Middleware are functions used in Express.js to process requests and responses.

11. What is callback hell?

Callback hell refers to deeply nested callbacks, making code unreadable. It can be solved using Promises or async/await.

12. What are Promises in Node.js?

Promises represent the eventual completion or failure of an asynchronous operation.

13. What is async/await?

async/await are syntactic sugar over Promises that make asynchronous code look synchronous.

14. Explain process.nextTick().

process.nextTick() schedules a callback to execute after the current operation completes.

15. What are microtasks and macrotasks in Node.js?

Microtasks (Promises, process.nextTick) run before macrotasks (setTimeout, setImmediate, I/O).

16. What is setImmediate() in Node.js?

setImmediate() executes a callback after the current event loop phase is completed.

17. What is the difference between setTimeout() and setImmediate()?

setTimeout() executes after a given delay, setImmediate() executes immediately after the current phase.

18. What is the purpose of package.json?

package.json stores project metadata including dependencies, scripts, and version info.

19. What is package-lock.json?

package-lock.json locks the versions of dependencies to ensure consistent installs.

20. What is Express.js?

Express.js is a minimal and flexible Node.js framework used for building web applications and APIs.

21. What are the types of streams in Node.js?

Readable, Writable, Duplex, Transform.

22. What is cluster module in Node.js?

The cluster module allows running multiple Node.js processes to take advantage of multi-core systems.

23. What is the difference between fork() and spawn() in Node.js?

spawn() launches a new process, fork() creates a new Node.js instance that can communicate via IPC.

24. What are child processes in Node.js?

Child processes allow executing commands in a separate process, enabling parallelism.

25. How does Node.js handle concurrency?

Using the event loop and non-blocking I/O operations.

26. What is the difference between synchronous and asynchronous functions in Node.js?

Synchronous functions block execution; asynchronous functions run in background without blocking.

27. What is the difference between process and thread in Node.js?

A process is an independent execution unit, while a thread is a lightweight unit within a process. Node.js runs single-threaded event loop but uses worker threads internally.

28. What is CORS in Node.js?

CORS (Cross-Origin Resource Sharing) allows servers to specify domains that can access resources.

29. What is JWT in Node.js?

JWT (JSON Web Token) is used for authentication and secure information exchange.

30. What is the difference between PUT and POST?

POST creates resources, PUT updates or replaces existing resources.

31. What are REST APIs?

REST APIs follow REST architecture using HTTP methods like GET, POST, PUT, DELETE.

32. What are WebSockets in Node.js?

WebSockets enable two-way communication between client and server in real time.

33. What is the difference between WebSockets and HTTP?

HTTP is request-response based, WebSockets allow persistent two-way communication.

34. What is buffer in Node.js?

Buffers are temporary storage areas for raw binary data.

35. What is the difference between process.nextTick() and setImmediate()?

process.nextTick() executes before the next event loop phase, setImmediate() executes at the end of the current phase.

36. What is an event emitter in Node.js?

EventEmitter is a class that facilitates communication between objects via events.

37. What is error-first callback in Node.js?

Error-first callbacks have the first argument reserved for error, followed by data.

38. How to handle exceptions in Node.js?

Using try/catch, error events, or promises with .catch().

39. What is the difference between spawn and exec in Node.js?

spawn() streams data, exec() buffers entire output.

40. What are worker threads in Node.js?

Worker threads provide a way to run JavaScript code in parallel on multiple threads.

41. What is the difference between process.env and dotenv package?

process.env accesses environment variables, dotenv loads them from a .env file.

42. What is the difference between fs.readFile and fs.createReadStream?

fs.readFile reads the entire file into memory, fs.createReadStream reads in chunks.

43. What is the difference between synchronous and asynchronous file system methods?

Synchronous blocks execution, asynchronous allows other operations to continue.

44. What is the difference between Node.js and PHP?

Node.js is non-blocking and event-driven, PHP is synchronous and blocking by default.

45. What are advantages of using Express.js with Node.js?

Simplifies routing, middleware support, easy API building, better code structure.

46. What is middleware chaining in Express.js?

Multiple middleware functions can be executed sequentially in request-response cycle.

47. What are HTTP modules in Node.js?

The 'http' module allows Node.js to transfer data over HTTP.

48. What is the difference between require() and import() in Node.js?

require() is static, import() (dynamic import) is asynchronous and returns a promise.

49. What is the difference between Node.js and Deno?

Node.js uses CommonJS, npm, and requires config; Deno uses ES modules, has built-in TypeScript, and no package.json.

50. What are common use cases of Node.js?

Building REST APIs, real-time chat apps, streaming apps, microservices, IoT applications.