Node.js Interview — 50 High■Yield Q&A; (with quick answers & examples)

This document provides 50 Node.js interview-style questions with concise answers and code snippets/examples.

Q1. What is Node.js?

Quick answer: Runtime built on Chrome V8 engine, executes JavaScript outside browser.

Example: console.log('Hello from Node');

Q2. Why is Node.js single-threaded?

Quick answer: Uses event loop & async I/O to handle many requests without threads.

Example: setTimeout(()=>console.log('async'),1000);

Q3. What is npm?

Quick answer: Node Package Manager; manages libraries and dependencies.

Example: npm install express

Q4. Difference between CommonJS and ES Modules?

Quick answer: CommonJS: require/module.exports. ES Modules: import/export.

Example: const x=require('x'); import x from 'x';

Q5. What are streams in Node.js?

Quick answer: Handle continuous data chunks: readable, writable, duplex, transform.

Example: fs.createReadStream('file.txt').pipe(res);

Q6. What is event-driven programming?

Quick answer: Flow controlled by events and callbacks.

Example: emitter.on('click',()=>{});

Q7. What are buffers in Node.js?

Quick answer: Temporary storage for binary data.

Example: const buf=Buffer.from('Hello');

Q8. Difference between process.nextTick() and setImmediate()?

Quick answer: nextTick: runs before event loop continues. setImmediate: executes after current event loop phase.

Example: process.nextTick(()=>{}); setImmediate(()=>{});

Q9. What are Node.js global objects?

Quick answer: Objects available everywhere: __dirname, __filename, process, Buffer.

Example: console.log(__dirname);

Q10. What is REPL in Node.js?

Quick answer: Read-Eval-Print Loop, interactive shell.

Example: $ node

Q11. What are child processes in Node.js?

Quick answer: Spawn new processes for parallel tasks.

Example: const {spawn}=require('child_process');

Q12. Explain cluster module.

Quick answer: Enables load balancing across multiple processes to use multi-core CPUs.

Example: cluster.fork();

Q13. What is middleware in Node?

Quick answer: Functions intercepting req/res in apps like Express.

Example: app.use((req,res,next)=>{console.log('hit'); next();});

Q14. Difference between synchronous and asynchronous functions in Node?

Quick answer: Sync blocks execution; async uses callbacks/promises.

Example: fs.readFile vs fs.readFileSync

Q15. What is package.json?

Quick answer: Metadata for project: dependencies, scripts, version.

Example: { "scripts": {"start": "node app.js"} }

Q16. Difference between dependencies and devDependencies?

Quick answer: dependencies needed in production; devDependencies for dev tools.

Example: npm install --save vs --save-dev

Q17. What is nodemon?

Quick answer: Utility that auto restarts server on file changes.

Example: npx nodemon app.js

Q18. Explain error-first callbacks.

Quick answer: Node convention: callback(err,result).

Example: fs.readFile('f',(err,data)=>{ if(err) throw err; });

Q19. What is the purpose of util.promisify?

Quick answer: Converts callback-based APIs to return promises.

Example: const read=util.promisify(fs.readFile);

Q20. Explain event loop in Node.js.

Quick answer: Manages async ops in phases: timers, I/O, poll, check, close.

Example: console.log, setTimeout etc.

Q21. Difference between process.env and dotenv?

Quick answer: process.env: access environment vars. dotenv: package to load from .env.

Example: require('dotenv').config();

Q22. What is middleware chaining?

Quick answer: Multiple middleware executed in order via next().

Example: app.use(mw1); app.use(mw2);

Q23. What is async_hooks module?

Quick answer: Provides hooks into lifecycle of async resources.

Example: const ah=require('async_hooks');

Q24. Difference between spawn, exec, and fork?

Quick answer: spawn: stream data. exec: buffer output. fork: specialized spawn for Node scripts.

Example: child_process.spawn('ls');

Q25. What is CORS in Node?

Quick answer: Cross-Origin Resource Sharing; allows/disallows requests from other origins.

Example: app.use(cors());

Q26. What is require cache?

Quick answer: Modules are cached after first load.

Example: delete require.cache[require.resolve('./mod')]

Q27. Explain global uncaughtException handler.

Quick answer: Catches errors not handled elsewhere.

Example: process.on('uncaughtException',err=>{});

Q28. What are worker threads?

Quick answer: Enable parallel JS execution in separate threads.

Example: new Worker('./task.js');

Q29. Difference between process.exit() and process.kill()?

Quick answer: exit: ends current process. kill: sends signal to process id.

Example: process.exit(1);

Q30. What is difference between fs.readFile and fs.createReadStream?

Quick answer: readFile: loads whole file. createReadStream: streams chunks.

Example: fs.createReadStream('big.txt').pipe(res);

Q31. What is crypto module?

Quick answer: Provides cryptographic functions: hashing, encryption.

Example: crypto.createHash('sha256').update('x').digest('hex');

Q32. What is dotenv used for?

Quick answer: Loads env variables from .env file into process.env.

Example: require('dotenv').config();

Q33. Explain middleware vs route handler.

Quick answer: Middleware: modifies req/res and calls next(). Route: final handler.

Example: app.use(auth); app.get('/home',handler);

Q34. Difference between Node.js and browser JS?

Quick answer: Node: no DOM, has fs, http. Browser: DOM APIs.

Example: window vs global object.

Q35. What are uncaughtPromiseRejections?

Quick answer: Unhandled promise rejections crash process in future versions.

Example: process.on('unhandledRejection',err=>{});

Q36. What is the purpose of path module?

Quick answer: Handle file paths easily.

Example: path.join(__dirname,'file.txt')

Q37. Difference between require and import in Node?

Quick answer: require: CommonJS, sync. import: ES module, async.

Example: const m=require('m'); import m from 'm';

Q38. What is body-parser?

Quick answer: Middleware to parse incoming request bodies.

Example: app.use(bodyParser.json());

Q39. What are environment variables?

Quick answer: Config values stored outside code.

Example: process.env.PORT

Q40. What is helmet in Node.js?

Quick answer: Security middleware setting HTTP headers.

Example: app.use(require('helmet')());

Q41. Explain synchronous blocking code in Node.

Quick answer: Code that prevents event loop from processing others.

Example: while(true){} blocks Node.

Q42. What are Node.js timers?

Quick answer: setTimeout, setInterval, setImmediate APIs.

Example: setImmediate(()=>console.log('done'));

Q43. Difference between process.cwd() and __dirname?

Quick answer: cwd: current working dir. __dirname: dir of current module.

Example: console.log(process.cwd(), __dirname);

Q44. What is PM2?

Quick answer: Process manager for Node.js apps (clustering, monitoring, restart).

Example: pm2 start app.js

Q45. Explain logging in Node.js.

Quick answer: Use console, Winston, Morgan, pino etc.

Example: const winston=require('winston');

Q46. What is SSR with Node?

Quick answer: Server-side rendering generates HTML before sending to client.

Example: res.send(ReactDOMServer.renderToString(<App/>));

Q47. Difference between REST and GraphQL with Node?

Quick answer: REST uses endpoints; GraphQL uses single endpoint with queries/mutations.

Example: app.use('/graphql',graphqlHTTP({schema,rootValue}));

Q48. Additional Node.js concept #48

Quick answer: Explanation of a Node.js interview topic.

Example: Example usage or snippet here.

Q49. Additional Node.js concept #49

Quick answer: Explanation of a Node.js interview topic.

Example: Example usage or snippet here.

Q50. Additional Node.js concept #50

Quick answer: Explanation of a Node.js interview topic.

Example: Example usage or snippet here.