ScopeTreasure Courses

NODE.JS & EXPRESS COURSES · LESSON N.1 · CODE SCOPE SERIES

Node.js Kya Hai? JavaScript on the Server

JavaScript escapes the browser: what Node.js is, how V8 runs your code on a computer, your first server-side program, and npm — the shop every package comes from.

Introduction

Until now, your JavaScript lived inside a **browser**. It changed colours, counted clicks and checked forms — but the moment the tab closed, it was gone, and it could never touch a file, a database, or another computer. **Node.js** takes the same JavaScript language out of the browser and runs it directly on a **computer or server**. The engine inside Node is **V8** — the very same engine Chrome uses — wrapped with powers a browser never gives: reading files, opening network doors (ports), and talking to databases.

That one change is the birth of the **backend**: the hidden half of every app that stores data, checks passwords and answers requests. And because it is still JavaScript, everything you learned in Series 3 — variables, functions, arrays, objects, promises — works here unchanged. One language, both halves of the app: that is why Node is the road to becoming a **full-stack** developer. *(Roman Urdu: Node.js JavaScript ko browser se nikaal kar computer par chalata hai — ab wohi zuban server par bhi chalegi.)*

Key points

- **Node.js is a runtime, not a language:** the language is still JavaScript. Node is the environment that runs it outside a browser, on your machine or a server.
- **V8 does the running:** Node carries Chrome's V8 engine inside it. V8 turns your JavaScript into machine code the computer understands — fast.
- **New powers, same language:** on a server, JavaScript can read and write **files**, listen on **ports**, and connect to **databases** — things browser JavaScript is forbidden to do.
- **One command runs a file:** `node app.js` in the terminal runs the file and prints the result right there. No browser, no HTML page needed.
  (Roman Urdu: Terminal mein `node file.js` likho — program chal para.)
- **npm comes with Node:** installing Node also installs **npm**, the shop from which you download ready-made packages (like Express) instead of building everything yourself.
- **`package.json` is the project's ID card:** it records the project's name, its start command (scripts) and the list of packages it depends on — created by `npm init`.
- **Nothing here is visual — and that is normal:** a backend has no buttons or colours. Its answers are text, JSON and saved data. The terminal is its face.

Code examples

Example 1 — Your first Node program

```
// app.js — run it with: node app.js
const name = "Ali";
const marks = 90;

console.log("Assalam o Alaikum, " + name);
console.log("Marks: " + marks);
console.log("Passed: " + (marks >= 40));
```

What it does: This is plain JavaScript — variables and `console.log` — but there is no browser anywhere. You save it as a file and run it with the `node` command in a terminal. Node reads the file, V8 runs it top to bottom, and the answers print in the terminal itself.

What you will see: the terminal prints three lines: `Assalam o Alaikum, Ali`, then `Marks: 90`, then `Passed: true` — and the program ends. JavaScript just ran on your computer, not in a page.

Example 2 — A real server in pure Node

```
// server.js — run it with: node server.js
const http = require("http");

const server = http.createServer((req, res) => {
  res.end("Hello from my Node server!");
});

server.listen(3000, () => {
  console.log("Server is listening on port 3000");
});
```

What it does: Node carries built-in toolboxes called **modules**; `http` is the one that speaks the web's language. `createServer` builds a server that answers every visitor with one line of text, and `listen(3000)` opens **port 3000** — a numbered door on your computer — and waits there. This is a working server with zero downloads, though lesson N.2 will show why everyone prefers Express over this raw style.

What you will see: the terminal prints `Server is listening on port 3000` and then waits — it does not end. Open `http://localhost:3000` in a browser and the page shows `Hello from my Node server!`. Press Ctrl+C in the terminal to stop it.

Example 3 — npm: the project's ID card and the package shop

```
npm init -y
npm install express
```

```
// package.json (the important part, after the install)
{
  "name": "my-first-server",
  "scripts": {
    "start": "node server.js"
  },
  "dependencies": {
    "express": "^4.21.2"
  }
}
```

What it does: `npm init -y` creates `package.json`, the project's ID card. `npm install express` downloads the Express package into a folder called `node_modules` and writes its name into `dependencies` — the list of packages this project cannot run without. The `start` script is a shortcut: from now on, `npm start` runs the server. (This is the roadmap's 6.2 topic — npm and `package.json` — covered here where it belongs.)

What you will see: npm prints a short summary like `added 68 packages` and a `node_modules` folder appears next to your files, with `package.json` now listing `express` under `dependencies`. Your project can officially use other people's ready-made code.

Quick recap

- Node.js runs JavaScript **outside the browser**, on a computer or server, using the **V8** engine.
- Server-side JavaScript gains real powers: files, ports, databases.
- `node file.js` runs a program; the built-in `http` module can already answer web requests.
- **npm** installs packages, and `package.json` records the project, its scripts and its dependencies.

Common mistakes

1. **Opening the file in a browser and expecting Node powers** — double-clicking runs nothing, and browser JavaScript can never read files or open ports. The fix: run server code from the terminal with `node file.js`; the terminal output is your result.
2. **Forgetting the server never stops** — after `listen`, the program waits forever, so beginners think it has "hanged". The fix: that waiting is the server doing its job; stop it deliberately with Ctrl+C when you are done.
3. **Deleting `node_modules` panic — or emailing it to a friend** — the folder is huge but it is not your code; it is downloaded packages. The fix: never share or upload `node_modules`; anyone can rebuild it from `package.json` with one `npm install`.

Practice

1. Install Node from nodejs.org, then write `hello.js` that prints your name, your city and the result of `2026 - 2000` using `console.log`. Run it with `node hello.js` and confirm all three lines appear in the terminal.

2. Take the Example 2 server and change the answer text to include your name. Restart the server (Ctrl+C, then run again), open `http://localhost:3000`, and also try a wrong door — `http://localhost:4000` — and note what the browser says when no server is listening there.

Practice quiz

1. Is Node.js a language?

- Yes, a new language that replaces JavaScript
- No, it is a runtime, not a language: the language is still JavaScript, Node is the environment that runs it outside a browser
- Yes, but only for servers
- No, it is a database

Answer: No, it is a runtime, not a language: the language is still JavaScript, Node is the environment that runs it outside a browser

2. What new powers does server-side JavaScript gain in Node?

- Buttons and colours
- Reading and writing files, listening on ports, and connecting to databases, things browser JavaScript is forbidden to do
- Faster CSS
- Direct access to the user's files

Answer: Reading and writing files, listening on ports, and connecting to databases, things browser JavaScript is forbidden to do

3. What is package.json?

- A list of all Node commands
- The project's ID card: its name, its start command (scripts) and the list of packages it depends on, created by npm init
- The folder of downloaded packages
- A backup file

Answer: The project's ID card: its name, its start command (scripts) and the list of packages it depends on, created by npm init

---

Watch the Code Scope channel: https://www.youtube.com/@CodeScopePK — this lesson's own video joins the channel when it is recorded.

Read online: https://scopetreasure.com/notes/node-what-is-node · More free tools and notes at https://scopetreasure.com