

Table of Contents



- [1. Installation](#)
- [2. Running Node.js](#)
- [3. Creating a Basic Node.js App](#)
- [4. Core Modules](#)
- [5. Creating an HTTP Server](#)
- [6. File System \(fs\)](#)
 - [Reading and Writing Files](#)
- [7. Path Module](#)
- [8. Events and Event Emitters](#)
- [9. NPM \(Node Package Manager\)](#)
 - [Install Packages:](#)
 - [Install as Dev Dependency:](#)
 - [Uninstall Package:](#)
 - [Global Installation:](#)
- [10. Package.json Overview](#)
- [11. Express.js \(Basic Setup\)](#)
- [12. Middleware in Express](#)
- [13. API Routes \(CRUD Example\)](#)
- [14. Working with JSON](#)
- [15. Handling Promises](#)
- [16. Asynchronous/Await](#)
- [17. Environment Variables](#)
- [18. Useful Node.js Commands](#)
- [19. Error Handling](#)
- [20. Deployment \(PM2 Process Manager\)](#)

1. Installation

- Download and Install:
<https://nodejs.org/>
- Check Installation:

```
node -v      # Node.js version
npm -v       # npm (Node Package Manager) version
```

2. Running Node.js

1. Interactive Mode (REPL):

```
node
```

- Type JavaScript directly into the terminal.

```
console.log("Hello, Node!");
```

2. Run a Script:

```
node app.js
```

3. Creating a Basic Node.js App

1. Create Project Directory:

```
mkdir myapp  
cd myapp
```

2. Initialize Node Project:

```
npm init -y
```

3. Create app.js:

```
console.log("Node.js App Running!");
```

4. Run App:

```
node app.js
```

4. Core Modules

```
const fs = require('fs');           // File System
const http = require('http');       // HTTP Server
const path = require('path');       // Path Utilities
const os = require('os');           // OS Information
const events = require('events');   // Event Emitter
```

5. Creating an HTTP Server

```
const http = require('http');

const server = http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Hello, World!');
});

server.listen(3000, () => {
  console.log('Server running at http://localhost:3000/');
});
```

- Visit <http://localhost:3000> in your browser.
-

6. File System (fs)

Reading and Writing Files

```
const fs = require('fs');

// Read File
fs.readFile('text.txt', 'utf8', (err, data) => {
  if (err) throw err;
  console.log(data);
});
```

```
// Write to File
fs.writeFile('output.txt', 'Hello Node.js!', (err) => {
  if (err) throw err;
  console.log('File written!');
});
```

7. Path Module

```
const path = require('path');

console.log(__dirname); // Directory name
console.log(path.basename(__filename)); // File name
console.log(path.join(__dirname, 'src', 'index.html'));
```

8. Events and Event Emitters

```
const EventEmitter = require('events');
const eventEmitter = new EventEmitter();

eventEmitter.on('greet', () => {
  console.log('Hello, Event!');
});

eventEmitter.emit('greet');
```

9. NPM (Node Package Manager)

Install Packages:

```
npm install express
```

Install as Dev Dependency:

```
npm install nodemon --save-dev
```

Uninstall Package:

```
npm uninstall express
```

Global Installation:

```
npm install -g nodemon
```

10. Package.json Overview

- Dependencies: Packages required for production.
- devDependencies: Packages required during development.

```
{  
  "name": "myapp",  
  "version": "1.0.0",  
  "main": "app.js",  
  "dependencies": {  
    "express": "^4.18.1"  
  },  
  "scripts": {  
    "start": "node app.js",  
    "dev": "nodemon app.js"  
  }  
}
```

11. Express.js (Basic Setup)

```
const express = require('express');  
const app = express();
```

```
app.get('/', (req, res) => {
  res.send('Welcome to Express!');
});

app.listen(3000, () => {
  console.log('Server running at http://localhost:3000/');
});
```

12. Middleware in Express

```
app.use(express.json());

app.use((req, res, next) => {
  console.log(`${req.method} ${req.url}`);
  next();
});
```

13. API Routes (CRUD Example)

```
let users = [{ id: 1, name: 'John' }];

app.get('/users', (req, res) => {
  res.json(users);
});

app.post('/users', (req, res) => {
  const newUser = req.body;
  users.push(newUser);
  res.status(201).send('User added');
});
```

14. Working with JSON

```
const data = {
  name: "Node.js",
  version: "18.0"
};

const jsonData = JSON.stringify(data); // Convert to JSON
console.log(JSON.parse(jsonData));     // Parse JSON back to object
```

15. Handling Promises

```
const fetchData = () => {
  return new Promise((resolve, reject) => {
    setTimeout(() => resolve('Data Loaded'), 2000);
  });
};

fetchData()
  .then((data) => console.log(data))
  .catch((err) => console.error(err));
```

16. Asynchronous/Await

```
async function loadData() {
  const data = await fetchData();
  console.log(data);
}

loadData();
```

17. Environment Variables

```
PORT=3000
```

```
require('dotenv').config();  
console.log(process.env.PORT);
```

18. Useful Node.js Commands

```
node app.js          # Run app  
nodemon app.js       # Auto-reload app (install nodemon globally)  
npm start            # Run start script  
npm run dev          # Run dev script  
npm list             # List installed packages
```

19. Error Handling

```
app.use((err, req, res, next) => {  
  console.error(err.stack);  
  res.status(500).send('Something broke!');  
});
```

20. Deployment (PM2 Process Manager)

```
npm install pm2 -g  
pm2 start app.js  
pm2 list  
pm2 stop app  
pm2 logs
```