

RenderATL 2026



Code and Learn

Presented by



Ethan Arrowood

Head of Open Source Engineering at Harper



@Ethan-Arrowood



linkedin.com/in/ethan-arrowood



bsky.app/profile/arrowood.dev



ethan@arrowood.dev



Plan

Quick overview then diving in head first

1

Understand Node.js

What it is and why the project is shaped this way

2

Contributing

Community and contribution details

3

Work safely

Contribution workflow, prerequisites, AI policy

4

Start coding

Claim a task, get to work!

What is Node.js?

 [Learn](#) [About](#) [Download](#) [Blog](#) [Docs](#) [Beta Docs](#) [Contribute](#) [Courses](#)

⌕ K   

Run JavaScript Everywhere

Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts.

Get Node.js®

Get security support
for EOL Node.js versions

Node.js is proudly supported by the partners above and more.

Create an HTTP Server [Write Tests](#) [Read and Hash a File](#) [Streams Pipeline](#) [Work v](#)

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain' });
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with `node server.mjs`
```

JavaScript </> Copy to clipboard

Learn more what Node.js is able to offer with our Learning materials.

v24.19.0 Latest LTS v26.7.0 Latest Release

Node.js

[About this documentation](#)

[Usage and example](#)

[Assertion testing](#)

[Asynchronous context tracking](#)

[Async hooks](#)

[Buffer](#)

[C++ addons](#)

[C/C++ addons with Node-API](#)

[C++ embedder API](#)

[Child processes](#)

[Cluster](#)

[Command-line options](#)

[Console](#)

[Crypto](#)

[Debugger](#)

[Deprecated APIs](#)

[Diagnostics Channel](#)

[DNS](#)

[Domain](#)

[Environment Variables](#)

[Errors](#)

[Events](#)

[File system](#)

[FFI](#)

[Globals](#)

[HTTP](#)

[HTTP/2](#)

[HTTPS](#)

[Inspector](#)

[Internationalization](#)

Node.js v26.7.0 documentation

[► Table of contents](#) | [► Index](#) | [► Other versions](#) | [► Options](#)

▼ Table of contents

- Index

Index

- [About this documentation](#)
- [Usage and example](#)

- [Assertion testing](#)
- [Asynchronous context tracking](#)
- [Async hooks](#)
- [Buffer](#)
- [C++ addons](#)
- [C/C++ addons with Node-API](#)
- [C++ embedder API](#)
- [Child processes](#)
- [Cluster](#)
- [Command-line options](#)
- [Console](#)
- [Crypto](#)
- [Debugger](#)
- [Deprecated APIs](#)
- [Diagnostics Channel](#)
- [DNS](#)
- [Domain](#)
- [Environment Variables](#)
- [Errors](#)
- [Events](#)
- [File system](#)
- [FFI](#)
- [Globals](#)
- [HTTP](#)
- [HTTP/2](#)
- [HTTPS](#)
- [Inspector](#)
- [Internationalization](#)
- [Iterable Streams API](#)

Node.js

[About this documentation](#)

[Usage and example](#)

[Assertion testing](#)

[Asynchronous context tracking](#)

[Async hooks](#)

[Buffer](#)

[C++ addons](#)

[C/C++ addons with Node-API](#)

[C++ embedder API](#)

[Child processes](#)

[Cluster](#)

[Command-line options](#)

[Console](#)

[Crypto](#)

[Debugger](#)

[Deprecated APIs](#)

[Diagnostics Channel](#)

[DNS](#)

[Domain](#)

[Environment Variables](#)

[Errors](#)

[Events](#)

[File system](#)

[FFI](#)

[Globals](#)

[HTTP](#)

[HTTP/2](#)

[HTTPS](#)

[Inspector](#)

[Internationalization](#)

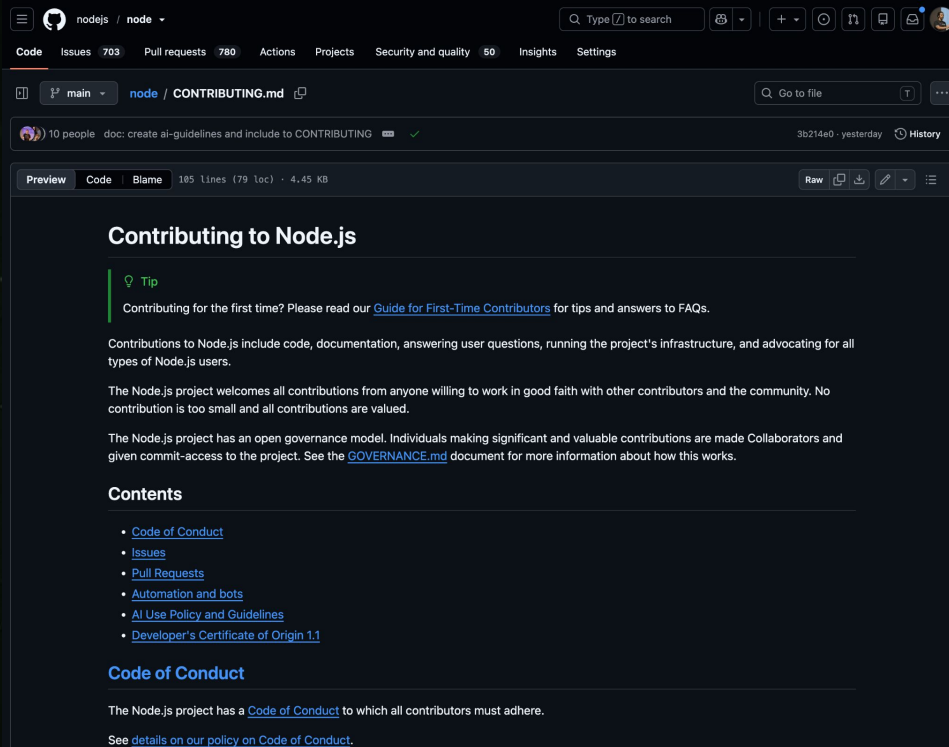
[Iterable Streams API](#)

Node.js v26.7.0 | [► Table of contents](#) | [► Index](#) | [► Other versions](#) | [► Options](#)

- [Internationalization](#)
- [Iterable Streams API](#)
- [Modules: CommonJS modules](#)
- [Modules: ECMAScript modules](#)
- [Modules: node:module API](#)
- [Modules: Packages](#)
- [Modules: TypeScript](#)
- [Net](#)
- [OS](#)
- [Path](#)
- [Performance hooks](#)
- [Permissions](#)
- [Process](#)
- [Punycode](#)
- [Query strings](#)
- [Readline](#)
- [REPL](#)
- [Report](#)
- [Single executable applications](#)
- [SQLite](#)
- [Stream](#)
- [String decoder](#)
- [Test runner](#)
- [Timers](#)
- [TLS/SSL](#)
- [Trace events](#)
- [TTY](#)
- [UDP/datagram](#)
- [URL](#)
- [Utilities](#)
- [V8](#)
- [Virtual File System](#)
- [VM](#)
- [WASI](#)
- [Web Crypto API](#)
- [Web Streams API](#)
- [Worker threads](#)
- [Zlib](#)

- [Code repository and issue tracker](#)

Contributing to Node.js



The screenshot shows the GitHub interface for the `nodejs/node` repository. The file `CONTRIBUTING.md` is selected, showing its commit history and a preview. The preview content includes a tip for first-time contributors, a list of contribution types, and a list of project governance documents.

nodejs / node

Code Issues 703 Pull requests 780 Actions Projects Security and quality 50 Insights Settings

node / CONTRIBUTING.md

10 people doc: create ai-guidelines and include to CONTRIBUTING 3b214e0 - yesterday History

Preview Code Blame 185 Lines (79 loc) · 4.45 KB Raw

Contributing to Node.js

Tip

Contributing for the first time? Please read our [Guide for First-Time Contributors](#) for tips and answers to FAQs.

Contributions to Node.js include code, documentation, answering user questions, running the project's infrastructure, and advocating for all types of Node.js users.

The Node.js project welcomes all contributions from anyone willing to work in good faith with other contributors and the community. No contribution is too small and all contributions are valued.

The Node.js project has an open governance model. Individuals making significant and valuable contributions are made Collaborators and given commit-access to the project. See the [GOVERNANCE.md](#) document for more information about how this works.

Contents

- [Code of Conduct](#)
- [Issues](#)
- [Pull Requests](#)
- [Automation and bots](#)
- [AI Use Policy and Guidelines](#)
- [Developer's Certificate of Origin 1.1](#)

Code of Conduct

The Node.js project has a [Code of Conduct](#) to which all contributors must adhere.

See [details on our policy on Code of Conduct](#).

<https://github.com/nodejs/node/blob/main/CONTRIBUTING.md>

AI Use in Node.js

<https://github.com/nodejs/node/blob/main/doc/contributing/ai-guidelines.md>

- Tools should never replace human judgment, regardless of whether they are powered by AI.

Node.js requires contributors to understand and take full responsibility for every change they propose. The answer to "Why is X an improvement?" can never be "I'm not sure. The AI did it."

Okay

- Read and understand the project
- Learn appropriate contributing processes
- Validate your work
- Translating messages

Not Okay

- Automate your contributions
- Opening too many PRs
- Interacting with other contributors
- Commenting on issues and PRs
- Reporting security vulnerabilities

AI policy overview

Use AI to learn faster, do not use it to bypass human accountability.

Appropriate

- Explain unfamiliar code
- Draft a test plan
- Check wording
- Suggest debugging paths

Inappropriate

- Unattended bots
- PRs you cannot explain
- Fake repro steps
- Large opaque rewrites

Human in the loop, human signs-off, human responsible.

How to sustainably source ideas

Good contributors learn where problems come from, not just where labels appear.

1

Good first issues

Start with curated entry points (like we are going to do today)

2

Recent PRs

Learn from recently accepted work.

3

Docs confusion

Fix where your own understanding got stuck.

4

Test gaps

Protect behavior that already exists.

5

Issue triage

Reproduce, reduce, clarify, label.

Sustainable rule: make the smallest useful change that you can explain.

Prerequisite recap

<https://gist.github.com/Ethan-Arrowood/b75daccca7af5889ea70eab586292a78>

Let's start coding

**WIFI: RenderOpenJS
Password: NodeJS26!**

<http://nodejs.org/code-and-learn-2026>

Before you open the PR

Make the reviewer's job easier, that is part of being helpful.

✓ Clear title

✓ Small scope

✓ Why this matters

✓ Tests or docs updated

✓ Signed-off commit

✓ Respectful follow-up

Reviewer question to answer:
What changed, why does it matter, and how did you check it?

Thank you!

