Node.js Next 10 Survey Results

Summary

We conducted the Node.js Next 10 Survey from **April 15-May 1** and amplified through Node.js social media channels. In total, there were **1,672 responses**. We also received feedback on our social posts, which are detailed below as well.

Surprisingly, the completion rate was 100% and took about 5 minutes to complete for folks. The most skipped question was if they had any recurring issues when using Node.js.

I've put the top findings in this document, but if you are interested in all of the data, see below:

- View the SurveyMonkey Powerpoint here.
- View the excel files with all of the data here.

Key Findings

- The majority of respondents were **application developers (58%)** followed by **back-end server authors (25%)**.
- Respondents largely work for **tech companies** or **startups**.
- The majority of folks **do not participate** in a Node.js community working group.
- Overwhelmingly, users leverage **npm** for their package and version managers.
- Top 3 technical priorities were:
  - Modern HTTP (75%)
  - Support for features from the latest ECMAScript spec (57%)
  - Documentation (51%)
- Top 3 important Node.js features were:
  - Consumable APIs and docs (67%)
  - Predictable and stable releases (53%)
  - A well maintained and secure standard library (51%)

Responses Received from Social

Social Post 1 (Twitter)

- Please provide built in array search and object search functions . And i think node js still need a mature orm so please work on that. And please work on worker thread give some mechanism of mature multi threading . Bcz node js only lack in multi threading
- You should definitely implement the following:
  First, #webgpu
  Then, a structure similar to virtual threads in Java (I know it's event-based, I know there's async/await 😄)
  More modern language features.
- Ts, esm and PERFORMANCES are the pains points

- Multithreading !

Social Post 2 (Twitter)

- Keep focus on stability and long term maintenance.
- More standard library
- Reduce bundle size. That's what really scares everyone out there.
- 1. A built-in 'nvm' tool could be nice. 2. At some point we have to acknowledge that TS has won the battle, and start supporting it natively just as @oven_sh does
- A more friendly API docs. Where every API will be described in a friendly manner rather than academic like writings.
- Make it easier to find memory leaks as now it is not easy to decipher heap profile snapshots and pinpoint to the problematic packages or code
- Normal TypeScript support
- let sharedMemory = 0; const runnable = () => {sharedMemory++;}; (new Thread(runnable)).start();
- Faster performance, less bloated. It is really slow.
- Reduce build size
- Make typescript first class. Simplify package.json and tsconfig.json working together for features like esm and top level await.
- I'd like to see a GUI much like how Docker does it to manage, to get insights, perhaps assist with debugging and integrations. Installation should be simpler and require less 3rd party stuff. And with a GUI you could still implement those settings for those that really need it
- 1. Pls improve tracing . For example pls provide trace I'd for entire transaction
  2. Pls improve error stack properties . Pls provide line no and file name in the message property itself
  3. Try to bring goto concept which is C language
- Make a standard and official nvm that works same on all OS, in Windows is a pain, it’s not read automatically the version in .nvmrc files

Social Post 3 (LinkedIn)

- 1. Less storage space and resources with more power;
  2. More OS features and better interaction with the OS like Java or C# for example;
  3. Native Artificial intelligence tools as we have today for example the http-server;
  4. DOM and easier http-requests and automation without external modules.
- 1.Better Support for Multi-Threading
  2.Improved Security
  3.Enhanced Debugging Capabilities
  4.More Comprehensive Documentation
  5.Better Integration with Other Technologies

Social Post 4 (LinkedIn)

- a suggestion: better navigation into Documentation. If you visit spring docs, it’s way clean and understandable.
- Looking for better multithreading support and Single Executable Application