neovim

Eduardo Vaz and João Pedro Sarti

1. What is the purpose of the project?

Neovim is a modern text editor that was created as a refactor and extension of Vim, which itself evolved from vi, the standard text editor that originally came with Unix system.

The purpose of Neovim’s development was to address long-standing design limitations in Vim while preserving its powerful modal editing philosophy. Just as Vim extended vi, Neovim reworked and improved upon Vim’s core, aiming for better maintainability, extensibility, and integration with modern tools.

2. Where is the source code located? Which technologies are used?

Neovim's source code is hosted on GitHub at **github.com/neovim/neovim,** and was developed using **C,** with heavy refactoring from Vim's code to increase maintainability**.**

Neovim also introduces **Lua** as a first-class language for configuration and plugin development, replacing most uses of Vimscript.

Another interesting feature of Neovim is it's implementation of an **LSP client** and integration with **Tree-sitter,** both for modern IDE-like features

3. What is the license? Are there public guidelines or any kind of bureaucracy for contributions?

Neovim is released under the Apache License 2.0. Contributions are openly welcomed through its GitHub repository, where the project maintains clear contribution guidelines and a code of conduct.

The process is relatively lightweight: contributors open pull requests or issues, and changes are discussed and reviewed by maintainers and the active community. There is no heavy bureaucracy, but contributors are expected to follow the guidelines on coding style, commit messages, and testing to keep the project consistent and maintainable.

4. Does Neovim have developer documenation?
Is it easy to compile/run? Is it easy to modify?

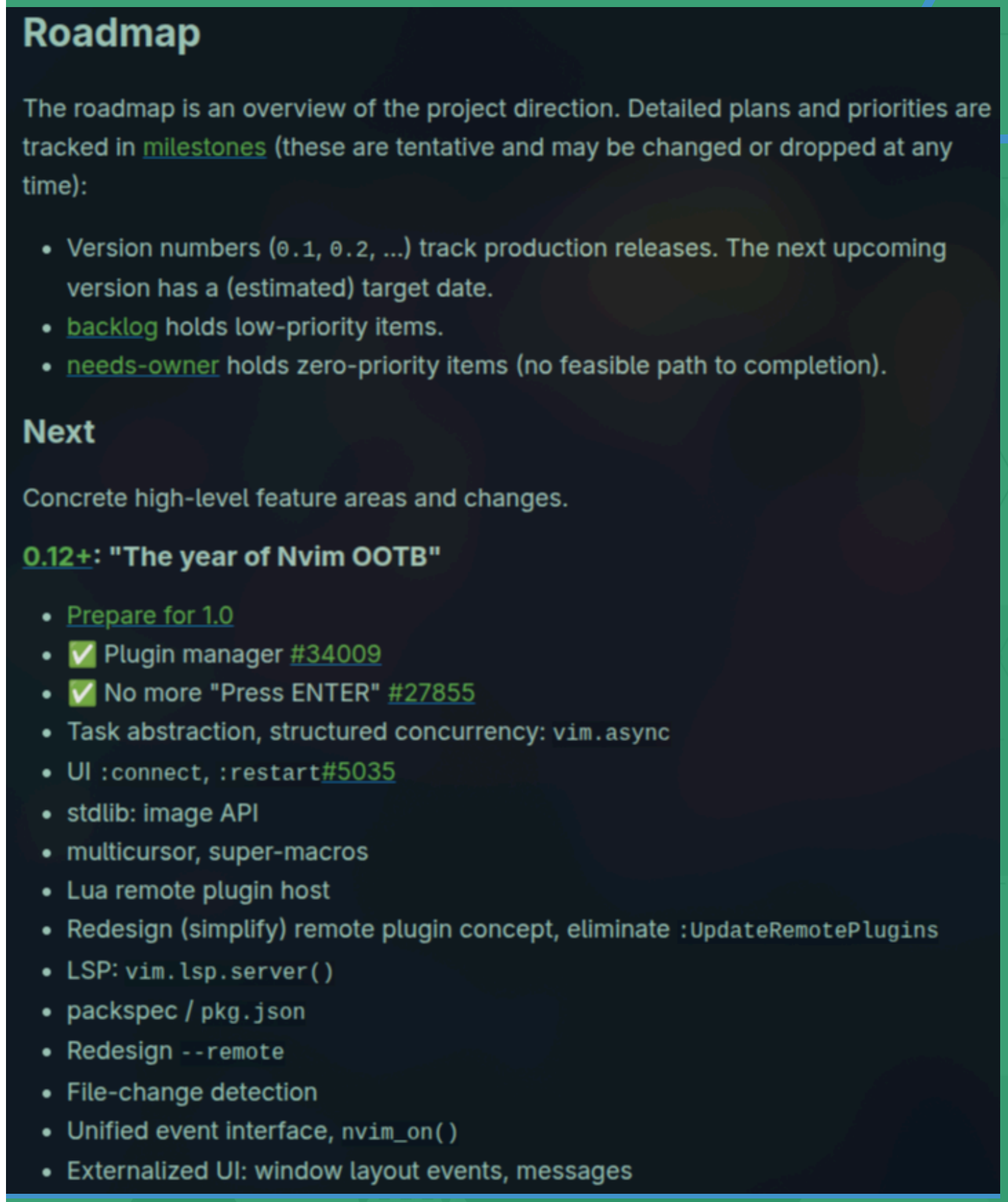
1. Neovim provides developer documentation in its GitHub repository, including a CONTRIBUTING.md, a BUILD.md with compilation instructions, and inline code documentation.

2. Neovim uses CMake for the build system, which makes compilation and setup straightforward on Linux, macOS, and Windows.

3. One of the main features of Neovim is it's modular architecture (with components like msgpack-RPC and Lua integration) making extending or adapting features relatively accessible, although it's core code is written in C, which requires more low-level knowledge.

5. How is the project governed?

NeoVim has a continuous maintenance team of 32 active developers, who also act as the project's internal code reviewers. The project is financed and supported by major companies such as Amazon, with CodeRabbit being one of its biggest supporters.
These same developers are also responsible for meeting and discussing the vital points of NeoVim and deciding how the project will move forward. From these discussions, they create an official NeoVim roadmap, which they follow when implementing new features.
The repository is open to contributions, mostly regarding bug fixes and general improvements, while the core team is responsible for implementing most of the new features. On the "Milestones" page on GitHub, we can see which bugs and enhancements were contributed by the community between two major releases from the roadmap.



Roadmap

The roadmap is an overview of the project direction. Detailed plans and priorities are tracked in milestones (these are tentative and may be changed or dropped at any time):

- Version numbers (`0.1`, `0.2`, ...) track production releases. The next upcoming version has a (estimated) target date.
- backlog holds low-priority items.
- needs-owner holds zero-priority items (no feasible path to completion).

Next

Concrete high-level feature areas and changes.

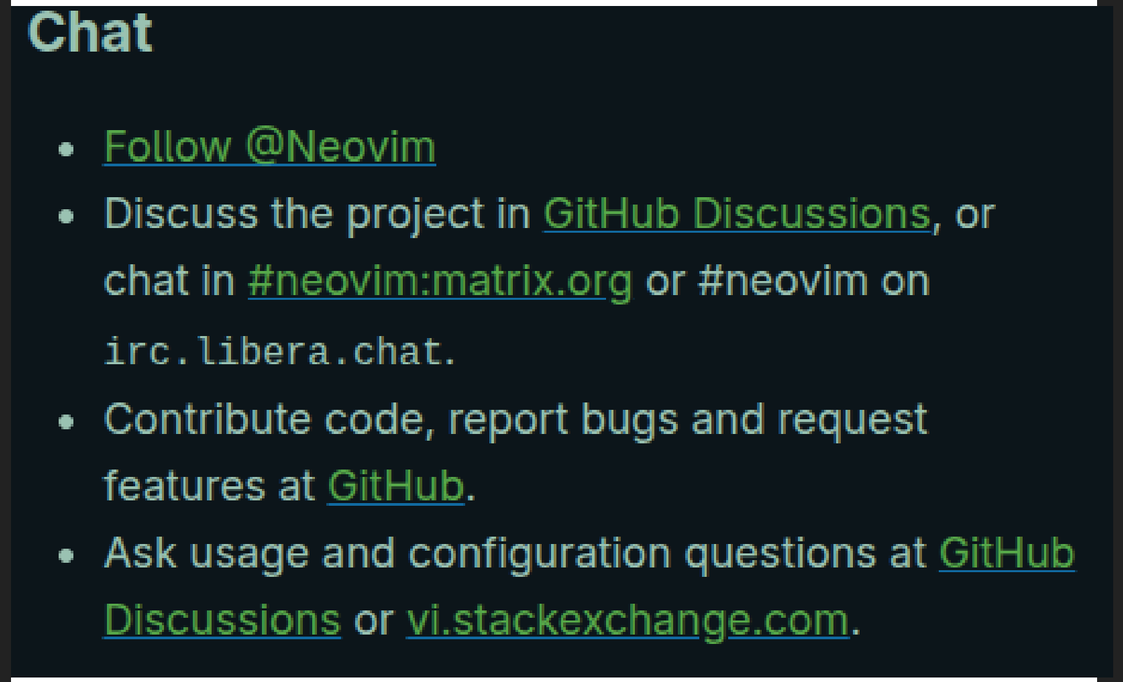
0.12+: "The year of Nvim OOTB"

- Prepare for 1.0
- ✅ Plugin manager #34009
- ✅ No more "Press ENTER" #27855
- Task abstraction, structured concurrency: `vim.async`
- UI `:connect`, `:restart` #5035
- stdlib: image API
- multicursor, super-macros
- Lua remote plugin host
- Redesign (simplify) remote plugin concept, eliminate `:UpdateRemotePlugins`
- LSP: `vim.lsp.server()`
- packspec / `pkg.json`
- Redesign `--remote`
- File-change detection
- Unified event interface, `nvim_on()`
- Externalized UI: window layout events, messages

6. Where can project members be found? Where can one seek help?

- Neovim community is largely organized in two main places, beign both a Discord and Github Projects.
- Through the use of a Kamban board, the developers organize themselfs together with the community through the Issues system.

Neovim Devs can also be found everywhere in the world, being one of the biggest Open Source projects in the world at the momment, it gathers devs from companies from all around the world, such as Google, Amazon, Netflix and the list goes on...



7. Does the project raise funds?

NeoVim have a fundraiser program based in external companies sponsorships and external community donations. The values of recurrent sponsors are not public, since GitHub does not reveal the subscribers tier and companies don’t disclose the information. But in their bitcoin donation wallet alone the value is already over $1.000.000.
All the value received is 100% re passed to the core developers in the team

Neovim open budget platform