GitterGraph: A TUI Tool for Git Repository Visualization

Tymofii Ivanov

ČVUT–FIT

ivanotym@fit.cvut.cz

January 4, 2026

1 Introduction

GitterGraph is a fast, interactive TUI (Terminal User Interface) tool for efficient git repository browsing directly in the terminal. The application provides clean visualization of linear repository history by following the first-parent chain,[1] significantly reducing visual clutter from feature branches while preserving the main development line.

2 Technologies

The project leverages modern Python libraries for performance and responsiveness:

- **Textual** [2] – Python TUI framework providing reactive component model and efficient terminal rendering.
- **pygit2** [3] – Python bindings for libgit2 [4], ensuring fast access to commit history, branches, and tags through native C implementation.

Figure 1: Textual framework for building TUI applications

Figure 2: libgit2 library for native Git operations

Development follows strict quality standards using mypy [5] for static type checking, pytest [6] for testing, and black [7] with isort [8] for code formatting.

3 Implemented Features

GitterGraph successfully implements:

- Fast loading and browsing of commit history with comprehensive metadata
- Organized display of all branches and tags
- Detailed commit inspection
- Fully keyboard-driven navigation
- Automatic repository discovery in parent directories
- Live reload without application restart

4 Extension and Improvement Possibilities

The project offers several paths for future development:

- **Partial nonlinear history visualization** – The current linear view could be enhanced with limited-depth branch display. Full nonlinear graph visualization remains challenging due to layouting complexity in terminal environments. Since git history forms a Directed Acyclic Graph (DAG) [9], sophisticated algorithms would be required for node placement within constrained character-grid space.
- **Git stash management** – Integration of stash stack display and operations.
- **Enhanced annotated tag support** – Display of tag messages, metadata, and GPG[2] signatures.

[1] The `-first-parent` option in git-log follows only the first parent commit of merge commits, creating a linear view of the main development line [1].

[2] GPG (GNU Privacy Guard) is a free implementation of the OpenPGP standard for encrypting and signing data.

- **Integrated diff viewer** – Side-by-side or unified diff display for commit comparisons.
- **Full-text search** – Query capabilities across commit messages and author information.

5 Conclusion

GitterGraph delivers an efficient terminal-based solution for git repository exploration. The combination of pygit2's native performance and Textual's modern framework enables a responsive, keyboard-driven interface. The application successfully fulfills its primary objective of providing clean, linear history visualization optimized for trunk-based development workflows.

References

[1] Git SCM. Git documentation: git-log. online, 2024. [cit. 2026–01–04] `https://git-scm.com/docs/git-log`.

[2] Textualize.io. Textual: Modern python tui framework. online, 2024. [cit. 2026–01–04] `https://www.textualize.io/`.

[3] libgit2 contributors. pygit2: Python bindings for libgit2. online, 2024. [cit. 2026–01–04] `https://www.pygit2.org/`.

[4] libgit2 contributors. libgit2: A portable, pure c implementation of the git core methods. online, 2024. [cit. 2026–01–04] `https://libgit2.org/`.

[5] Python Software Foundation. mypy: Optional static typing for python. online, 2024. [cit. 2026–01–04] `https://mypy-lang.org/`.

[6] pytest dev. pytest: helps you write better programs. online, 2024. [cit. 2026–01–04] `https://docs.pytest.org/`.

[7] Python Software Foundation. Black: The uncompromising code formatter. online, 2024. [cit. 2026–01–04] `https://github.com/psf/black`.

[8] PyCQA. isort: A python utility to sort imports. online, 2024. [cit. 2026–01–04] `https://pycqa.github.io/isort/`.

[9] Git SCM. Git internals - git objects. online, 2024. [cit. 2026–01–04] `https://git-scm.com/book/en/v2/Git-Internals-Git-Objects`.