GSOC'24 Final Report: Nix Internals: Use std::filesystem::path for Path

Siddhant Kumar

August 20, 2024

1 Summary

This document marks the completion of my participation in Google Summer of Code 2024 with the NixOS Foundation. I worked on replacing the usage of the internal 'Path' type (an alias for 'std::string') with C++17's 'std::filesystem::path' type, which is portable across Unix and Windows. My mentor was Ericson2314.

2 Introduction

Nix is a package manager available on Unix-based systems for writing build recipes for software, enabling the build process to be reproducible. It has many use cases and powers a very stable and reliable Linux distribution called NixOS.

My project is part of an ongoing port of Nix to Windows. The scope of my project was to use 'std::filesystem::path' and related types and functions throughout the codebase to make it portable across Unix and Windows.

3 Goal of the Project

- Aid in porting Nix to Windows.
- Encourage the use of modern C++ (C++17).
- Reduce the source code size by relying more on standard library features.

4 Outcomes of the Project

The migration to ‘std::filesystem::path‘ is complete in many parts of the codebase. One open PR requires attention, and some compiler errors need to be resolved in siddhantk232/more-std-filepath before it can be merged.

- List of related PRs: `https://github.com/NixOS/nix/pulls?q=author:siddhantk232`.

5 Future Prospects

Some work remains to complete this project, specifically:

- Fixing failing tests in `https://github.com/NixOS/nix/pull/10937`.
- Fixing compiler errors in `https://github.com/siddhantk232/nix/tree/more-std-filepath`.

For more information on the larger goal of porting Nix to Windows, please see `https://discourse.nixos.org/t/nix-on-windows/1113/109` and `https://github.com/NixOS/nix/labels/windows`.

6 Learnings

This project was a valuable opportunity to learn more about the internals of Nix—the project I use daily. It was also the first contribution I made to a C++ codebase of this scale. During the project, I learned many things:

- How changes are introduced in a large project so that it is easier for reviewers to merge your code.
- How to read a lot of C++ code and how its features are used throughout the codebase.
- The Meson build system and some of its features that help in building Nix for different platforms.

7 Acknowledgements

I want to extend my deepest thanks to John Ericson for all his support during this GSoC project. He helped me set up my development environment, explained the contribution process, and assisted me throughout the GSoC. Working with him was an enriching experience.

I am also grateful to Eelco Dolstra (edolstra) for reviewing some of my PRs.

My sincere thanks go to the NixOS Foundation and everyone who made this event possible, especially Janik, who helped during the initial proposal phase.

Lastly, I would like to express my gratitude to Google for providing this wonderful opportunity to be part of open-source development.