VSCode

Visual Studio Code is a popular open source code editor from Microsoft with extension support.

> **WIP**
>
> This page has not been completed.

Using in Nix based projects

If your project provides a flake.nix along with a development shell, it can be developed on VSCode using one of the two ways (prefer the 2nd way):

1. Open VSCode from a terminal, inside of a devshell (i.e., `nix develop -c code .`), **or**
2. Setup direnv and install the direnv VSCode extension.

> **The `.vscode` folder**
>
> You can persist Nix related extensions & settings for VSCode in the project root's `.vscode` folder (see example). This enables other people working on the project to inherit the same environment as you.

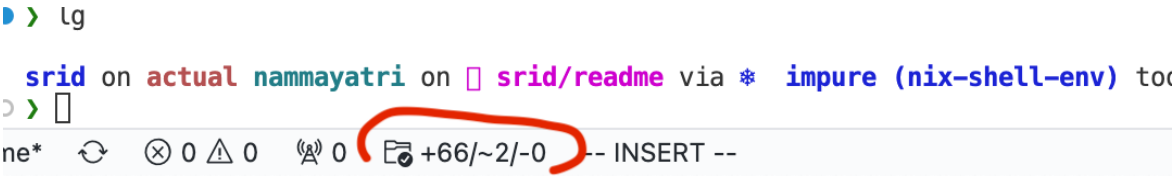
Working on **`direnv`**-activated projects

If you use direnv, it is rather simple to get setup with VSCode:

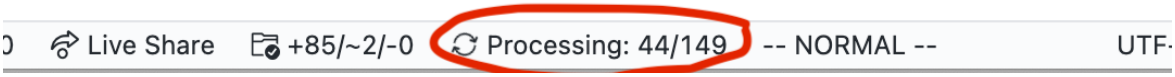
Once you have cloned your project repo and have activated the direnv environment (using `direnv allow), you can open it in VSCode to develop it:

- Launch VSCode, and open the **`git clone`**'ed project directory as single-folder workspace
  - NOTE: If you are on Windows, you must use the Remote - WSL extension to open the folder in WSL.
- When prompted by VSCode, install the workspace recommended extensions.

  - If it doesn't prompt, press Cmd+Shift+X and search for `@recommended` to install them all manually.
- Ensure that the direnv extension is fully activated. You should expect to see this in the footer of VSCode:



- For Haskell projects: Once direnv is activated (and only then) open a Haskell file (`.hs`). You should expect haskell-language-server to startup, as seen in the footer:



  - Once this processing is complete, all IDE features should work.
  - The experience is similar for other languages; for Rust, it will be rust-analyzer.

To give this a try, here are some sample repos:

- Haskell: https://github.com/srid/haskell-template
- Rust: https://github.com/srid/rust-nix-template

Links to this page

Rust FFI in Haskell

> You can find the template at https://github.com/shivaraj-bh/haskell-rust-ffi-template. This template also includes formatting setup with treefmt-nix and VSCode integration.

Getting Started with Nix for Haskell & Rust

> Sridhar Ratnakumar will demonstrate the delights of using Nix to develop Rust as well as Haskell projects without needing to do any manual global setup on your system. We'll start from a pristine macOS machine as well as a pristine Linux machine to get our development environment up and running in no time, all the way up to LSP support in VSCode.