About me: My name is *Solène Rapenne*, pronouns she/her. I like learning and sharing knowledge. Hobbies: '(NixOS BSD OpenBSD Lisp cmdline gaming internet-stuff). I **love** percent and lambda characters. OpenBSD developer solene@.

Contact me: *solene+www at dataswamp dot org* or *@solene@bsd.network* (mastodon). If for some reason you want to support my work, this is my paypal address: donate@perso.pw.

How to pin a nix-shell environment using niv

Written by *Solène*, on 12 January 2022.
Tags: #nix #nixos #shell

Comments on Fediverse/Mastodon

Introduction §

In the past I shared a bit about Nix nix-shell tool, allowing to have a "temporary" environment with a specific set of tools available. I'm using it on my blog to get all the dependencies required to rebuild it without having to remember what programs to install.

But while this method was practical, as I'm running NixOS development version (called unstable channel), I have to download the new versions of the dependencies every time I use the nix shell. This is long on my DSL line, and also a waste of bandwidth.

There is a way to pin the version of the packages, so I always use the exact same environment, whatever the version of my nix.

Use niv tool §

Let's introduce you to niv, a program to manage nix dependencies, for this how-to I will only use a fraction of its features. We just want it to init a directory with a default configuration pinning the nixpkgs repository to a branch / commit ID, and we will tell the shell to use this version.

niv project GitHub homepage

Let's start by running niv (you can get niv from nix package manager) in your directory:

```
niv init
```

It will create a nix/ directory with two files: sources.json and sources.nix, looking at the content is not fascinating here (you can take a look if you are curious though). The default is to use the latest nixpkgs release.

Create a shell.nix file §

My previous shell.nix file looked like this:

```
with (import <nixpkgs> {});
mkShell {
    buildInputs = [
        gnumake sbcl multimarkdown python3Full emacs-nox toot nawk mandoc libxml2
    ];
}
```

Yes, I need all of this for my blog to work because I have texts in org-mode/markdown/mandoc/gemtext/custom. The blog also requires toot (for mastodon), sbcl (for the generator), make (for building and publishing).

Now, I will make a few changes to use the nix/sources.nix file to tell it where to get the nixpkgs information, instead of which is the system global.

```
let
  sources = import ./nix/sources.nix;
  pkgs = import sources.nixpkgs {};
in
with pkgs;
pkgs.mkShell {
    buildInputs = [
        gnumake sbcl multimarkdown python3Full emacs-nox
        toot nawk mandoc libxml2
    ];
}
```

That's all! Now, when I run nix-shell in the directory, I always get the exact same shell and set of packages every day.

How to update? §

Because it's important to update from time to time, you can easily manage this using niv, it will bump the latest commit id of the branch of the nixpkgs repository:

```
niv update nixpkgs -b master
```

When a new release is out, you can switch to the new branch using:

```
niv modify nixpkgs -a branch=release-21.11
```

Using niv with configuration.nix §

It's possible to use niv to pin the git revision you want to use to build your system, it's very practical for many reasons like following the development version on multiple machines with the exact same revision. The snippet to use sources.nix for rebuilding the system is a bit different.

Replace "{ pkgs, config, ... }:" with:

```
{
  sources ? import ./nix/sources.nix,
  pkgs ? import sources.nixpkgs {},
  config, ...
}:
```

Of course, you need to run "niv init" in /etc/nixos/ before if you want to manage your system with niv.

Extra tip: automatically run nix-shell with direnv §

It's particularly comfortable to have your shell to automatically load the environment when you cd into a project requiring a nix-shell, this is doable with the direnv program.

nixos documentation about direnv usage

direnv project homepage

This can be done in 3 steps after you installed direnv in your profile:

1. create a file .envrc in the directory with the content "use nix" (without double quotes of course)
2. execute "direnv allow"
3. create the hook in your shell, so it knows how to do with direnv (do this only once)

How to hook direnv in your shell

Everytime you will cd into the directory, nix-shell will be automatically started.
