
Introduction to Nix

Functionally pure package management for deterministic artifacts and more

DevOps @ KTH

yannik@kth.se

What is this Nix?

It's difficult...

1. A programming language
2. A package manager
3. An operating system
4. A full ecosystem of tools

What the Nix?



Nix, the package manager

- packages/artifacts in nix: *Derivations*
- Defined as pure functions

Arguments: Dependencies

Function definition: Build steps

Return value: Build result

```
{ lib, fetchFromGitHub, stdenv, autoreconfHook
, ncurses, IOKit
}:

stdenv.mkDerivation rec {
  pname = "htop";
  version = "3.0.5";

  src = fetchFromGitHub {
    owner = "htop-dev";
    repo = pname;
    rev = version;
    sha256 = "sha256-9zecDd3oZ24Ry0LnKdJmR29Chx6S24Kvuf/F7RYz14I=";
  };

  nativeBuildInputs = [ autoreconfHook ];

  buildInputs = [ ncurses
  ] ++ lib.optionals stdenv.isDarwin [ IOKit ];

  meta = with lib; {
    description = "An interactive process viewer for Linux";
    homepage = "https://htop.dev";
    license = licenses.gpl20only;
    platforms = with platforms; linux ++ freebsd ++ openbsd ++ darwin;
    maintainers = with maintainers; [ rob relrod ];
  };
}
```

Did you say
Pure
functions?!



—

PURE FUNCTIONS



Yes :)

- Immutable inputs
- Immutable outputs
- Derivation instructions, dependencies and sources are hashed into a single identifier
- *Turing Complete*

How does that work

/nix/store/47kr83pgj3k2zgy7p46354mr42gdb0sl-tree-1.8.0

The `ro nix` store, contains
all built derivations
It is also the single thing
heavier than
`node_modules`

The aforementioned hash
value of **one specific**
instantiation of the `tree`
program

Information for *human*

Contents of a built result

Resembles /usr folder in Linux
Filesystem Hierarchy Standard
(FHS)

/nix/store/25hn ... ppn5-neuropil

```
├── include
│   ├── neuropil.h
│   ├── neuropil_attributes.h
│   └── neuropil_data.h
└── lib
    ├── libneuropil.a
    └── libneuropil.dylib
```

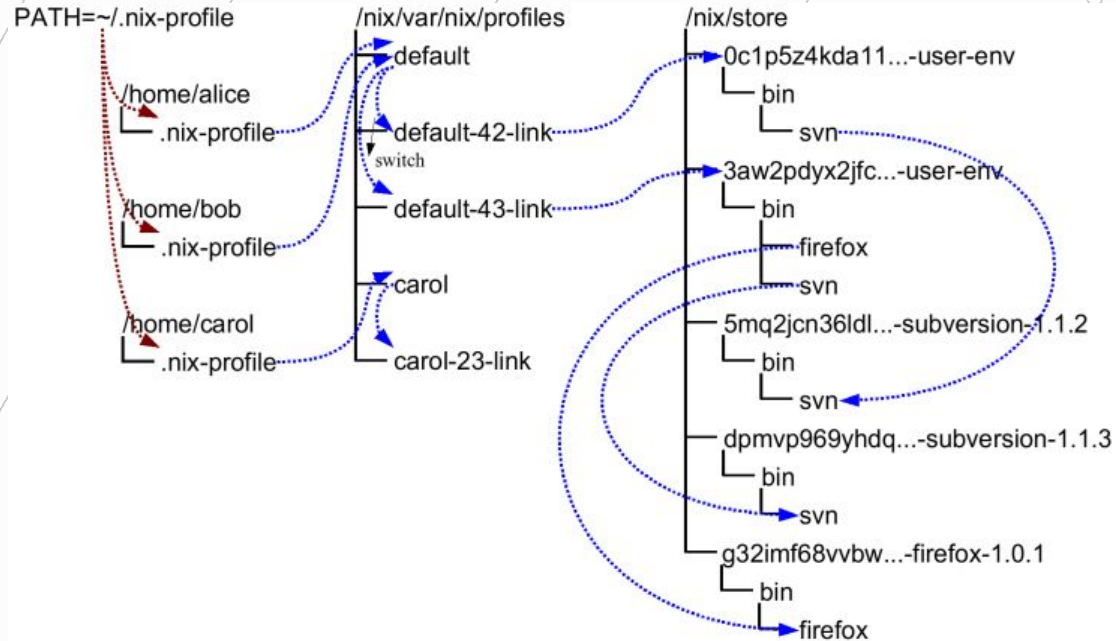

**ceci n'est pas un
package manager**

—

Package Management

Installing derivations creates a sophisticated graph of links

Binaries **linked** into a folder on the user's \$PATH



What Else?





Nixpkgs

- The [nixpkgs repo](#) contains all available packages
- 80.000 and counting
- [Automatized updates](#)



Development Shells

- **Virtual Environments**
- Define Programs, libraries, environment variables
- Can be loaded automatically ([direnv](#))



Declaratively installed Software

- Define what **should** be installed
- Nix takes care of installing providing programs
- [NixOS](#) and [home-manager](#)



Declaratively configured Services

- Configure Servers and Daemons in one single place
- Nix Implementation writes config files
- [NixOS](#) and [home-manager](#)



NixOS

- A complete operating system built entirely on Nix
- Configure almost anything declaratively
- Users, Services, Hardware, Network, etc
- Reversible configurations

DevOps?





Nix as a DevOps tool

- Pure builds, push proven packages to servers
- Minimally disruptive updates
- Integration with Testing pipeline [hydra](#) or [Hercules CI](#)
- [NixOps](#) to deploy and provision many servers at once

Nix combines: **Building**, *Testing*, **Deploying** Packages, and deployment of full reversible systems

SHUT UP AND



TAKE MY MONEY!

Thank you!

Useful links

