WIEERWILL.DEV

NIX KANN'S BESSER

DER NAECHSTE SCHRITT IN LINUX

Präsentiert von WieErWill

LINUX INFO TAG – 26. APRIL 2025

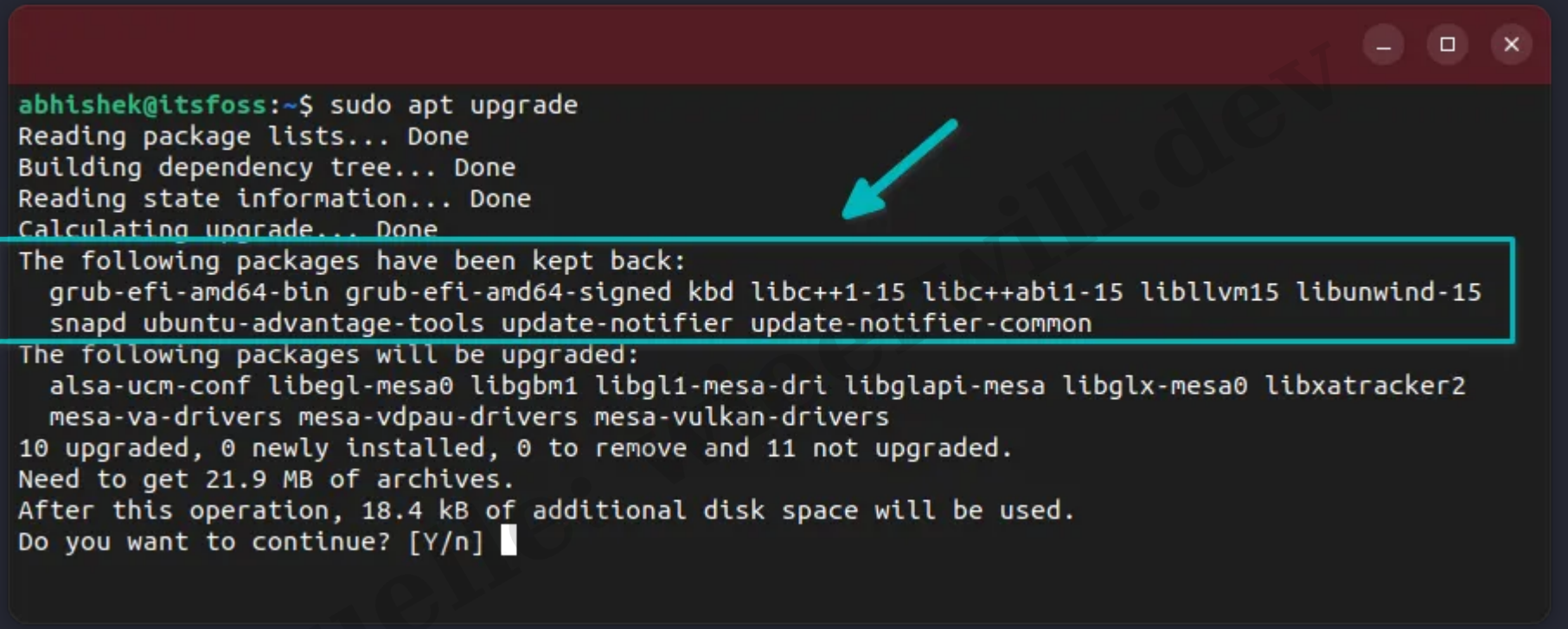
Quelle: wieerwill.dev

Your PC ran into a problem that it couldn't handle, and now it needs to restart.

You can search for the error online: HAL_INITIALIZATION_FAILED

Quelle: wieerwill.dev

VERALTETE PAKETE & DEPHASING



AUF ZU BESSEREN LOESUNGEN

Was wäre, wenn Software stabil und reproduzierbar wäre?

Quelle: will.dev

Was ist Nix?

Ein deklarativer Paketmanager,

der Software isoliert

und reproduzierbar im Nix-Store installiert

- plattformunabhängig und stabil.

Quelle: wiederwill.dev

KLASSISCHE PAKETMANAGER

APT, YUM und Co. installieren global Änderungen

```
usr
├── bin
│   └── curl
│   └── zsh
├── share
│   └── man
│       └── man1
│           └── curl.1.gz
├── lib
│   └── x86_64-linux-gnu
│       └── libcurl.so.4
```

führt zu Konflikten - bekannt als “Dependency Hell”

Quelle: wieerwill.dev

MAGIE DES NIX STORE

Nix installiert Pakete isoliert im /nix/store

/nix/store/<hash>-<name>-<version>/

```
/nix/store/
├── abc123-zsh-2.10
│   ├── bin/
│   │   └── zsh
│   └── share/
│       └── doc/
├── def456-zsh-2.11
│   ├── bin/
│   │   └── zsh
│   └── share/
│       └── doc/
│
```

verschiedene Versionen können gleichzeitig existieren

=> keine Versionskonflikte und Stabilität

=> Nix vermeidet Dependency Hell

Quelle: wieerwill.dev

ATOMARE UPDATES

Updates geschehen nicht in-place

neue Generationen, sodass System immer in einem intakten Zustand bleibt

Jeder Update-Schritt ist komplett atomar

einfache, nachvollziehbare Upgrades

Quelle: wiederwill.dev

ROLLBACKS - SICHERHEIT AUF ABRUF

mit `nix-env –rollback` jederzeit zurück

stabiles System garantiert schnell erreichbar

=> ein echter Sicherheitsmechanismus

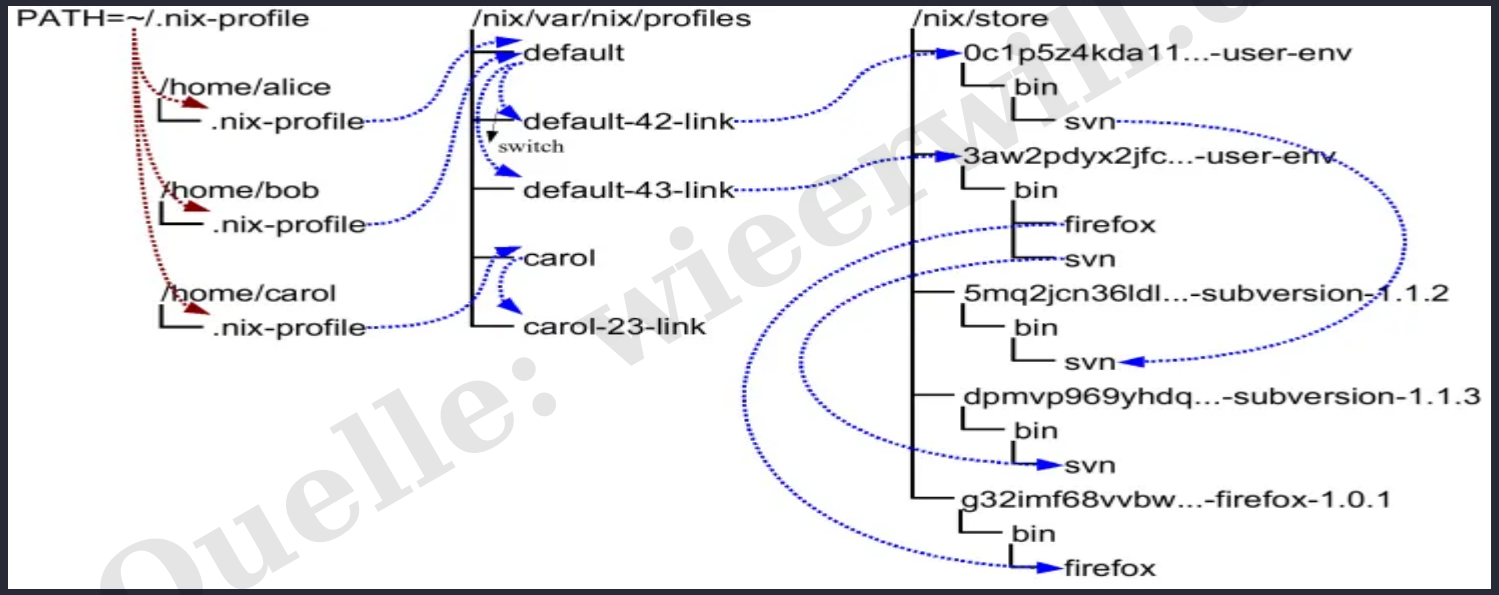
Garbage Collector: `nix-collect-garbage -d`

Quelle: wiederwinder

ISOLIERTE PROFILE

jeder Nutzer erhält eigenes Profil

keine Konflikte und isolierte Umgegungen



Quelle: wieerwill.de

VORTEILE VON NIX

- Reproduzierbare Builds
- Sichere, isolierte Installationen
- Atomare Updates und einfache Rollbacks
- Unabhängige Nutzerprofile

=> Zeitersparnis, Sicherheit und Stabilität

Quelle: wieeswill.dev

INSTALLATION VON NIX

Installiere Nix neben dem bestehenden System

ohne Eingriff in die Systemverwaltung

```
$ curl -L https://nixos.org/nix/install | sh -s -- --daemon
  ...
$ nix --version
  nix (Nix) 2.11.0
```

Nicht-invasiv und parallel zu bestehenden Systemen

Quelle: wieerwill.dev

ERSTE BEFEHLE

nix-env für permanente Installationen

Bsp: nix-env -i hello - Installiert Programm “hello”

oder

nix-shell für temporäre, isolierte Umgebungen

Bsp: nix-shell -p gcc - Startet temporäre Shell mit gcc

Quelle: wieerwill.dev

AD HOC SHELL ENV

Schnell, reproduzierbar, isoliert

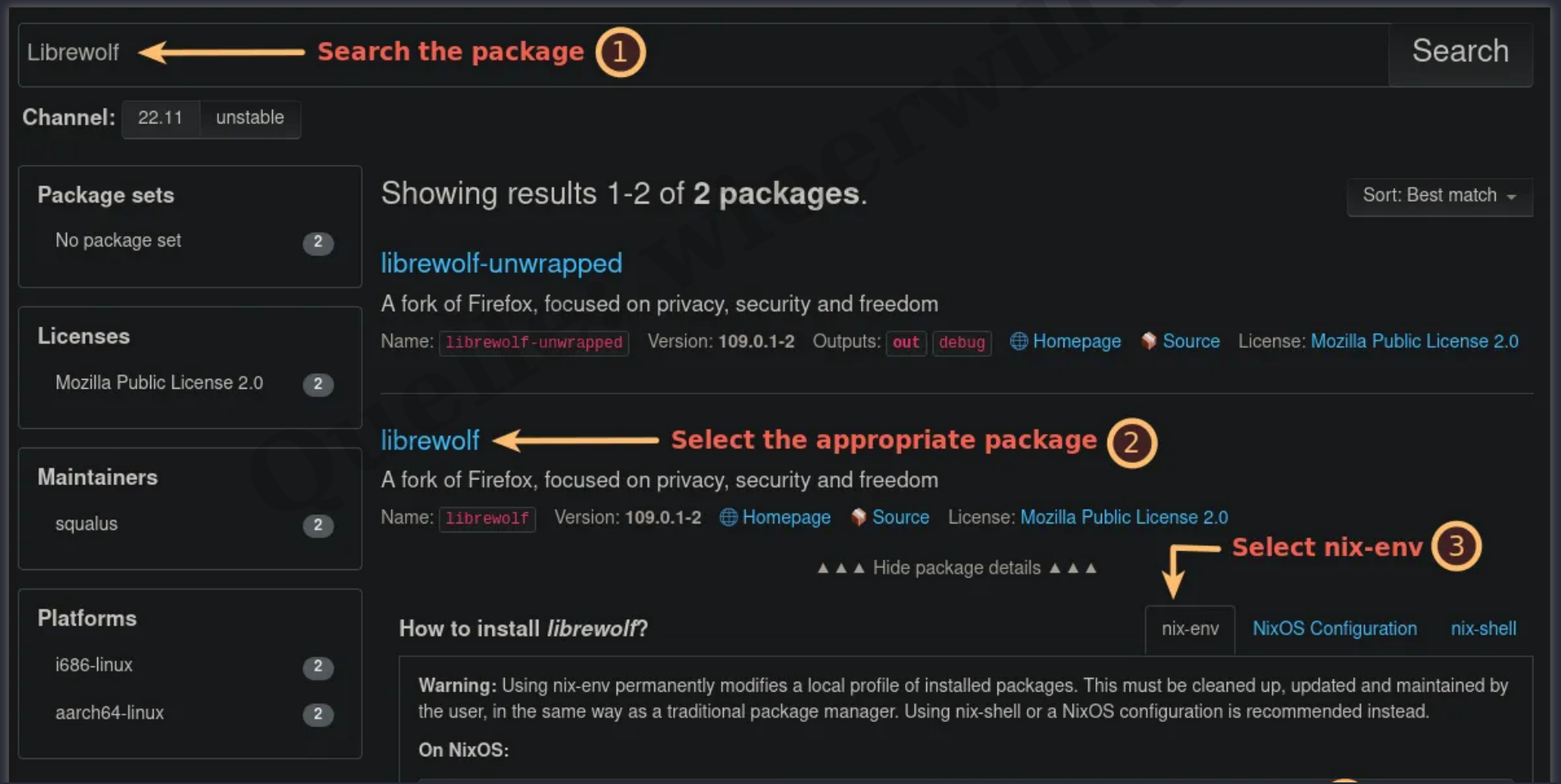
```
$ cowsay no can do
The program ‘cowsay’ is currently not installed.

$ echo no chance | lolcat
The program ‘lolcat’ is currently not installed.

$ nix-shell -p cowsay lolcat
these 3 derivations will be built:
  /nix/store/zx1j8gchgwzfjn7sr4r8yxb7a0afkjdg-builder.pl.drv
  /nix/store/h9sbaa2k8ivnihw2czhl5b58k0f7fsfh-lolcat-100.0.1.drv
  ...

(nix)$ cowsay Hello, Nix! | lolcat
Hello, Nix!

(nix)$ exit
exit
```

Quelle: wasserwill.de

NIX PACKAGES

über 120’000 Pakete verfügbar

Verschiedene Versionen und Konfigurationen



HOME MANAGER

verwalte die Umgebung als deklarativen Code

definiere Programme und Einstellungen zentral

ideal zur Reproduzierbarkeit und Migration

```
$ nix-channel --add \
  https://github.com/nix-community/home-manager/archive/release-24.11.tar.gz \
  home-manager
$ nix-channel --update
$ nix-shell '\<home-manager\>' -A install
```

Quelle: wiki.nixos.org

HOME MANAGER-CONFIG

einfache Konfiguration für Firefox, Vim, Git und ZSH

```
{ pkgs, ... }:
{
  home.packages = [ pkgs.firefox pkgs.vim ];
  programs.git = {
    enable = true;
    userName = "Max Mustermann";
    userEmail = "max@beispiel.de";
  };
  programs.zsh.enable = true;
}
```

Versionskontrolle und Reproduzierbarkeit

Quelle: wieerwill.dev

WAS IST NIXOS?

eine Linux-Distribution,

die den Nix-Paketmanager für das gesamte System nutzt.

Die deklarative Systemkonfiguration garantiert Stabilität

Quelle: wieerwill.de

NIXOS: DEKLARATIVE KONFIGURATION

Alle Systemkomponenten - von Diensten bis zu Paketen - werden in configuration.nix definiert (oder über Home-Manager)

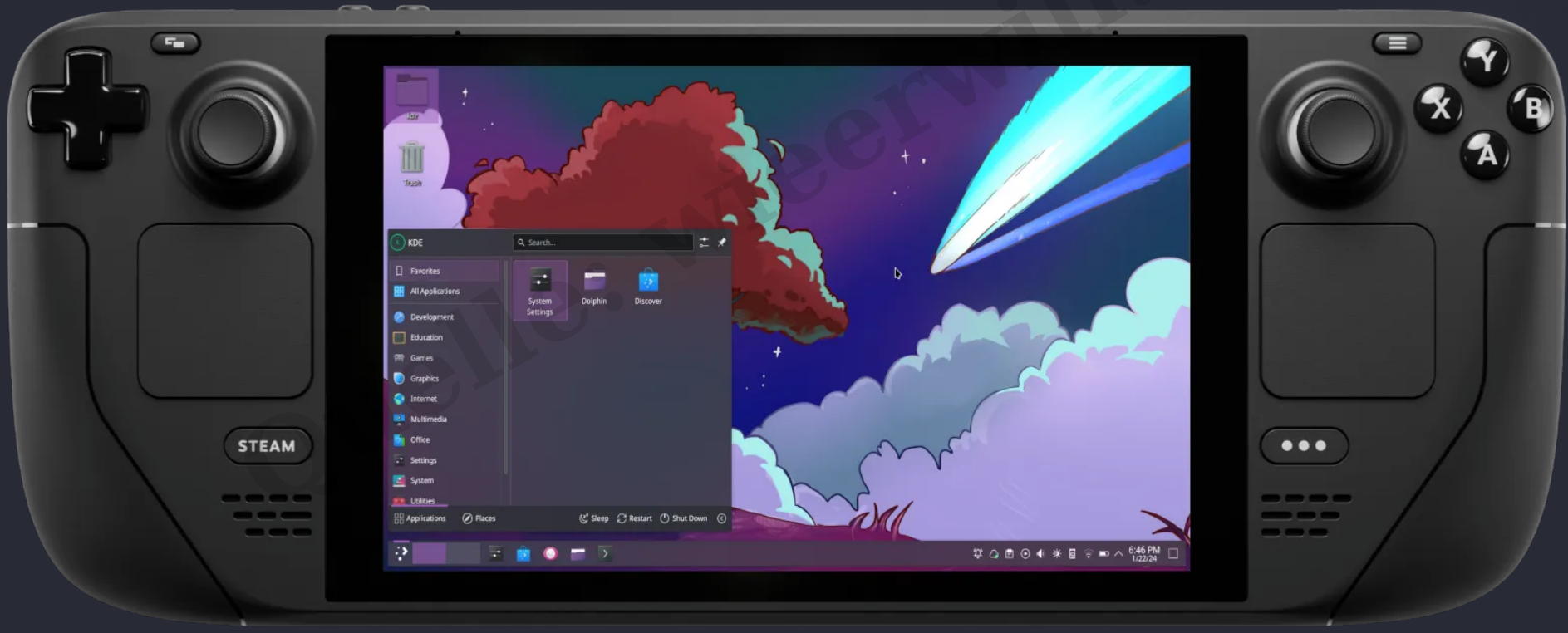
```
{
  services.openssh.enable = true;
  networking.firewall.enable = true;
  environment.systemPackages = [ pkgs.tmux pkgs.git ];
}
```

Nachvollziehbar und Wiederherstellbar

Quelle: wiki.nixos.org

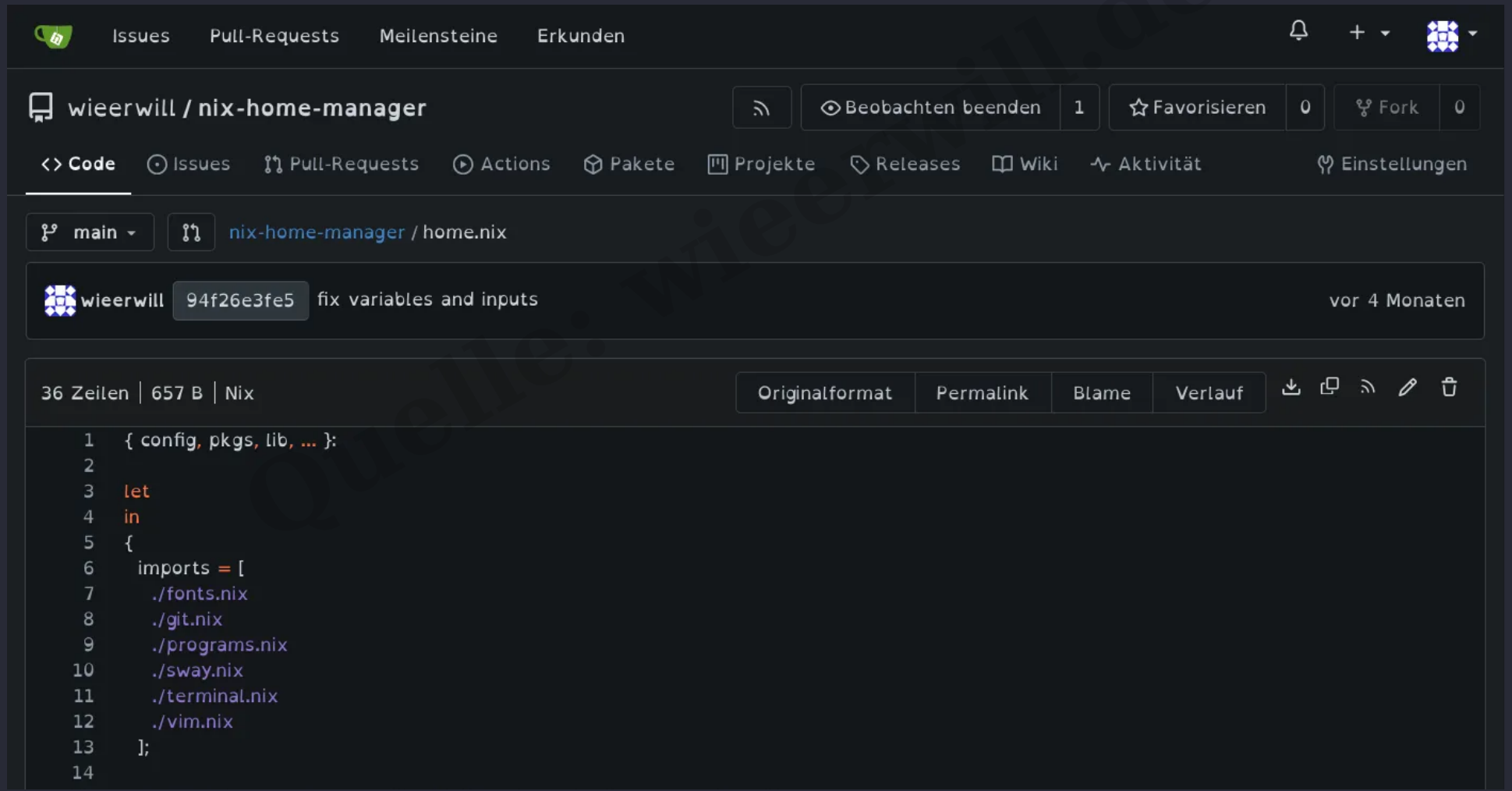
NIXOS AUF DEM STEAM DECK

Unser Experiment: NixOS auf einem Steam Deck zeigt, wie flexibel und robust das System auch auf mobilen Geräten ist.



ERKENNTNISSE AUS DEM EXPERIMENT

Robust, reproduzierbar und weniger stressig
Vorteile deklarativer Systemverwaltung



RESSOURCEN & DOKUMENTATION

- Offizielle NixOS-Dokumentation: nixos.org/manual
- NixOS Wiki: nixos.wiki
- Tutorials & Blogartikel (z.B. Nix Pills, c’t Artikel)
- Community: Eine engagierte Gemeinschaft unterstützt alle über Foren, Chats und regelmäßige Treffen - ideal, um schnell Hilfe zu finden.

Quelle: wieerwill.dev

AUSBLICK IN DIE ZUKUNFT

Nix entwickelt sich ständig weiter - mit innovativen Features wie Flakes, DevEnv und verbesserten Dev-Tools. Die Zukunft der Softwareverwaltung ist deklarativ, nachvollziehbar und sicher.

DIE ZUKUNFTS IST NIX

ZUSAMMENFASSUNG NIX(OS)

- Reproduzierbare, sichere Softwareumgebungen
- Isolierte Installationen und atomare Updates
- Home Manager und NixOS erweitern den Nutzen
- Community und Ressourcen unterstützen den Einstieg

Probier Nix selbst aus!
Wage den Schritt in eine neue Ära der Softwareverwaltung

Quelle: Wie ein Wahnsinn

ZEIT F R EURE FRAGEN & DISKUSSIONEN

WieErWill.dev/vcard.vcf

Quelle: WieErWill.dev