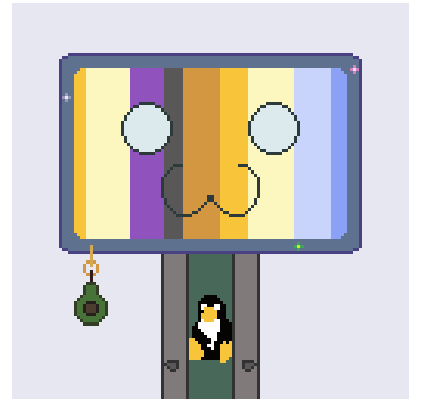


**Nix auf einem Server ist  
ganz schön viel!**

**Server-Verwaltung mit NixOS Containern**

# Wer bin ich?

- Mia/Motte (dey/es/sie/er)
- Studi und Programmierer\*in
- Nix, Rust, Go und alles was mir so unterkommt
- <https://www.motte.gay>



# Der Plan für heute

- Kleine Übersicht über Nix
- Ein wenig über NixOS-Container
- Demo :)

# Was ist Nix überhaupt?

- Programmiersprache



<https://github.com/NixOS/nixos-artwork/blob/master/logo/nix-snowflake-colours.svg>

# Was ist Nix überhaupt?

- Programmiersprache
- Package Manager



<https://github.com/NixOS/nixos-artwork/blob/master/logo/nix-snowflake-colours.svg>

# Was ist Nix überhaupt?

- Programmiersprache
- Package Manager
- Betriebssystem
- <https://www.nixos.org>



<https://github.com/NixOS/nixos-artwork/blob/master/logo/nix-snowflake-colours.svg>

# Die Sprache

- Funktionale Programmiersprache
- Speziell gemacht für Derivations
  - Prozess um von Datei A zu Datei B zu kommen

```
1 {stdenv, fetchurl, ...}: {}
```



```
1 {stdenv, fetchurl, ...}:
```



```
2
```

```
3 stdenv.mkDerivation {}
```

```
1  {stdenv, fetchurl, ...}:  
2  
3  stdenv.mkDerivation {  
4      pname = "foobar";  
5      version = "0.1.0";  
6      src = fetchurl {  
7          url = "...";  
8          sha256 = "...";  
9      };  
10 }
```



# NixOS

- Deklarative Konfiguration in der Nix-Sprache

# NixOS

- Deklarative Konfiguration in der Nix-Sprache
- Änderung der Konfiguration → Neue Generation

# NixOS

- Deklarative Konfiguration in der Nix-Sprache
- Änderung der Konfiguration → Neue Generation
- Das bedeutet:
  - Reproduzierbares Betriebssystem
  - Konfiguration einmal schreiben und überall nutzen
  - Was kaputt? Reise in der Zeit zurück >:)

```
1  {pkgs, ...}:  
2  {  
3    # ...  
4    users.users.motte.initialPassword = "supersecure";  
5    services.openssh.enable = true;  
6    # ...  
7  }
```

# NixOS container

- systemd-nspawn container
- Definiert in einem NixOS-System unter containers

# NixOS container

- systemd-nspawn container
- Definiert in einem NixOS-System unter containers
- Haben selbst NixOS am laufen
  - Voller Zugriff aufs Ökosystem!
  - Einfaches umschreiben von Konfigurationen

# systemd-nspawn?

# systemd-nspawn?

- “chroot auf Stereoiden”
- Kaum overhead
- Einfache Isolation von Systemen
- <https://wiki.archlinux.org/title/Systemd-nspawn>

```
1  {pkgs, ...}:  
2  {  
3    containers.hello = {  
4      config = {pkgs, ...}: {  
5        services.openssh.enable = true;  
6      };  
7    };  
8  }
```

# Weitere Informationen

- <https://noogle.dev>

# Weitere Informationen

- <https://noogle.dev>
- <https://wiki.nixos.org>

# Weitere Informationen

- <https://noogle.dev>
- <https://wiki.nixos.org>
- <https://nix.dev>

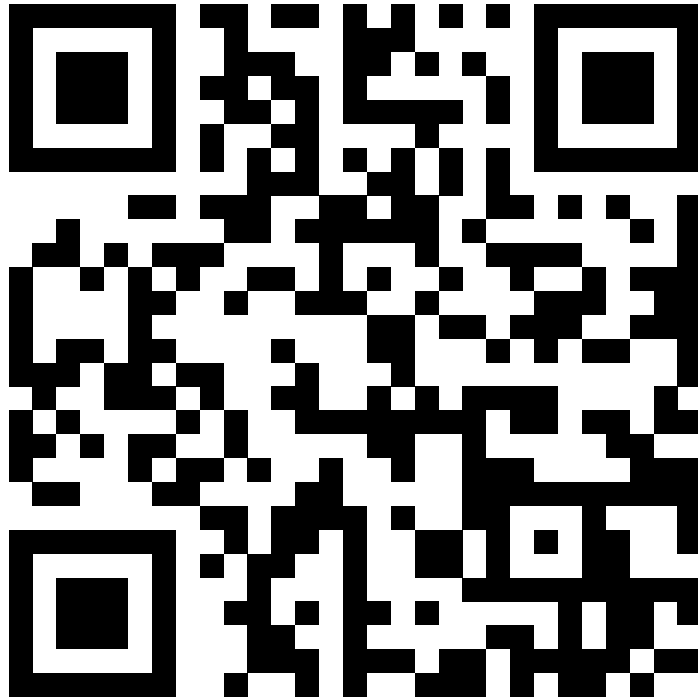
# Weitere Informationen

- <https://noogle.dev>
- <https://wiki.nixos.org>
- <https://nix.dev>
- <https://search.nixos.org/packages>

# Weitere Informationen

- <https://noogle.dev>
- <https://wiki.nixos.org>
- <https://nix.dev>
- <https://search.nixos.org/packages>
- <https://search.nixos.org/options>

# Präsentation und Code



<https://codeberg.org/motte/froscon-nix>

# Vor der Demo...

Language	files	blank	comment	code
Nix	17	205	132	1695
Lua	1	85	86	518
Bourne Shell	6	26	13	140
TOML	1	0	0	77
vim script	1	3	4	15
SUM:	26	319	235	2445

