What if GitHub Actions were local-first and built using Nix?

Domen Kožar @ 2025.nixcon.org

Travis CI: The Pioneer (2011)

```yaml
language: python
python:
  - "3.8"
  - "3.9"
script:
  - pytest
```

GitHub Actions: The Ecosystem Reuse (2018)

```yaml
on: [push, pull_request]
jobs:
  test:
    runs-on: ubuntu-latest
    strategy:
      matrix:
        python-version: ["3.8", "3.9"]
    steps:
      - uses: actions/checkout@v5
      - uses: actions/setup-python@v4
        with:
          python-version: ${{ matrix.python-version }}
      - run: pytest
```

Travis CI: The Pioneer (2011)

```yaml
language: python
python:
  - "3.8"
  - "3.9"
script:
  - pytest
```

GitHub Actions Example

```yaml
name: Demo Python Workflow
on: [push]
jobs:
  build:
    runs-on: ubuntu-latest
    strategy:
      matrix:
        python-version: ["3.8", "3.10"]
    steps:
      - uses: actions/checkout@v5
      - uses: actions/setup-python@v4
        with:
          python-version: ${{ matrix.python-version }}
      - name: Install dependencies for demo Python project
        run: |
          python -m pip install --upgrade pip
          pip install -r requirements.txt
      - name: Linting project with flake8
        run: flake8 . --count --exit-zero
      - name: Testing the project using pytest
        run: pytest tests.py --cov=com --cov-report=xml --cov-report=html
```

But

- completely ignores how to run locally
- expressions in yaml
- pre-baked OS

We’re constructing a developer environment!

Nix: Runs locally, runs in CI

Developer Experience

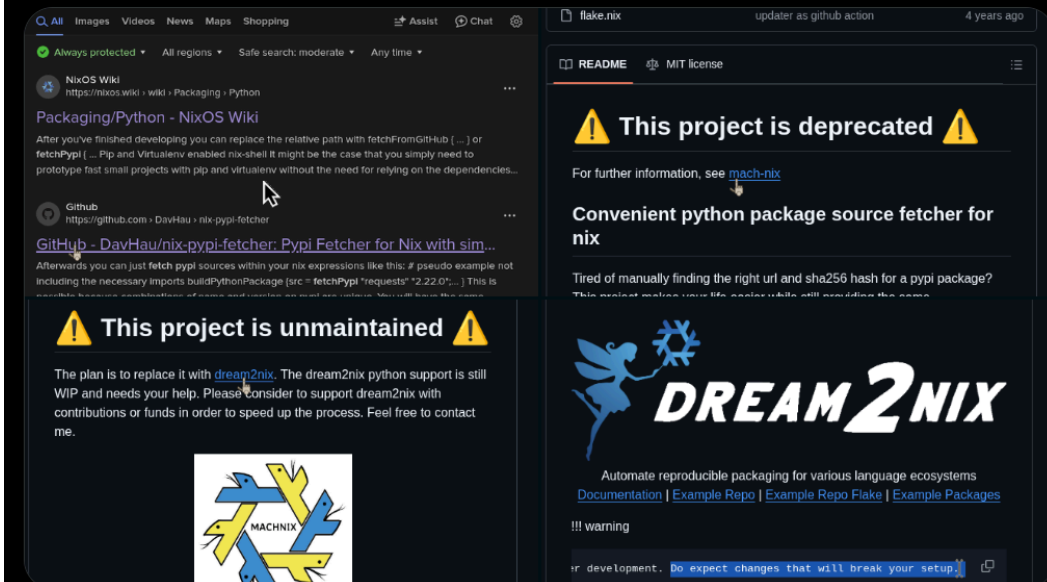
Empathy for newcomers

Cognitive load

1. Start doing a task
2. I have to make a decision

jason
@jvmncs

this is the Classic Nix experience. this is what it's all about



4:53 PM · May 26, 2025 · **35.8K** Views

16 53 951 100

Conventions over configuration

Uniform interface

```
languages.python.enable = true;
```

Optionally specify a version

```
languages.python.enable = true;
languages.python.version = "3.14";
```

low-level interface

```
packages = [ pkgs.python314 ];
```

~~flakes~~ devenv.sh

```
{ pkgs, ... }: {
  languages.python = {
    enable = true;
    version = "3.14";
    venv.enable = true;
    venv.requirements = ./requirements.txt;
  }

  enterTest = "pytest";

  git-hooks.hooks.flake8.enable = true;
}
```

```
$ devenv shell
```

Local … Docker Compose … Nixified

```
{ pkgs, ... }: {
  packages = [ pkgs.mkdocs ];

  processes.mything.exec = "mkdocs serve";

  services.postgresql.enable = true;
}
```

```
$ devenv up
• Building processes ...
• Starting processes ...
...
```

GitHub Actions … Nixified

```
{ pkgs, config, ... }: {
  languages.python = {
    enable = true;
    version = "3.14";
    venv.enable = true;
    venv.requirements = ./requirements.txt;
  }

  enterTest = "pytest";

  services.postgresql.enable = !config.cloud.enable;

  git-hooks = {
    hooks.flake8.enable = true;
    fromRef = config.cloud.ci.github.base_ref or null;
    toRef = config.cloud.ci.github.ref or null;
  };
}
```

GitHub Actions steps

```
- name: Linting project with flake8
  run: flake8 . --count --exit-zero
- name: Testing the project using pytest
  run: pytest tests.py --cov=com --cov-report=html
```

devenv tasks

```
{
  tasks = {
    "myapp:flake8" = {
      exec = "flake8 . --count --exit-zero";
      before = [ "devenv:enterTest" ];
    };

    "myapp:pytest" = {
      exec = "pytest tests.py --cov=com --cov-report=html";
      before = [ "devenv:enterTest" ];
    };
  };
}
```

```
$ devenv test
```

GitHub Actions matrix

```yaml
  strategy:
    matrix:
      python-version: ["3.14", "3.15"]
  steps:
    - uses: actions/checkout@v5
    - uses: actions/setup-python@v4
      with:
        python-version: ${{ matrix.python-version }}
```

devenv profiles (cachix/devenv#2137)

```nix
{ pkgs, config, ... }: {
  languages.python = {
    enable = true;
    version = "3.15";
  }

  profiles."python-3.14" = {
    languages.python.version = "3.14";
  };
}
```

```
$ devenv --profile python-3.14 shell
```

GitHub Actions artifacts

```yaml
- uses: actions/upload-artifact@v4
  with:
    name: my-artifact
    path: |
      path/output/bin/
```

devenv outputs

```nix
{ config, ... }: {
  # https://devenv.sh/languages
  languages = {
    rust.enable = true;
    python.enable = true;
  };

  # https://devenv.sh/outputs
  outputs = {
    rust-app = config.languages.rust.import ./rust-app {};
    python-app = config.languages.python.import ./python-app {};
  };
}
```

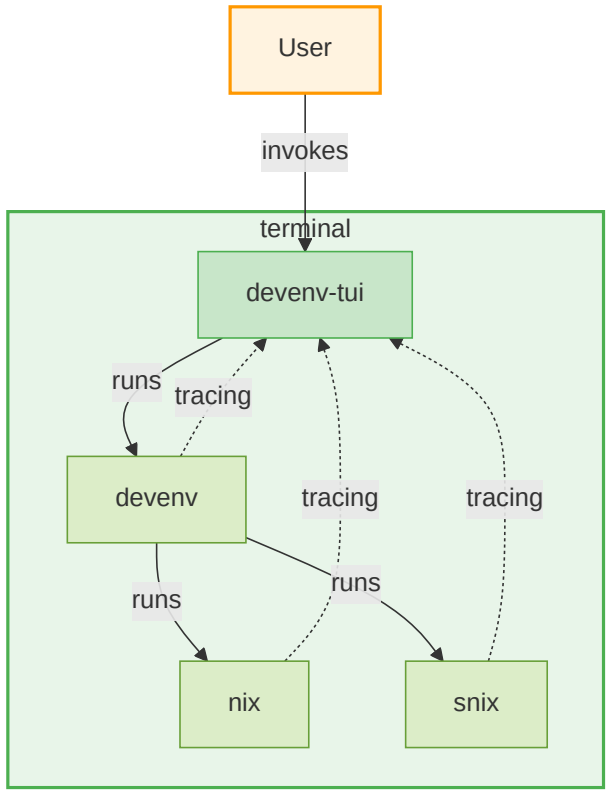
cachix/cloud.devenv.sh

GitHub Actions with Nix

tracing

devenv-tui: cachix/devenv#2060

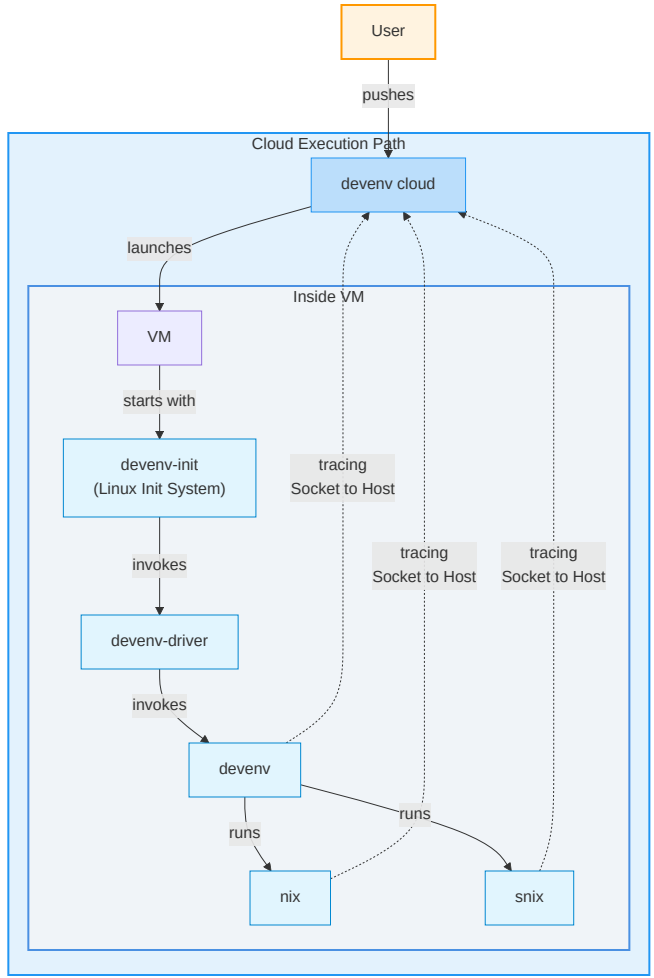
devenv-tui: architecture



From git push to Nix evaluation under a second on Linux

Linux workflow (macOS is similar)

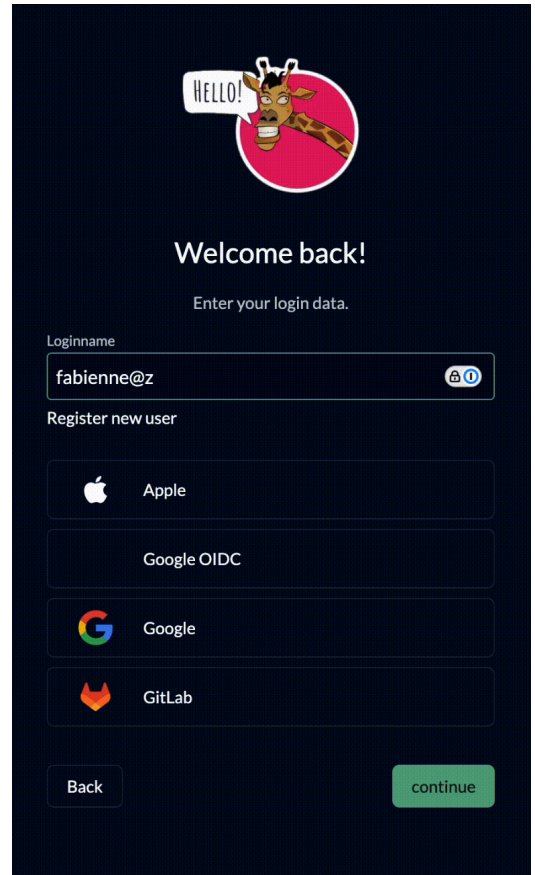
- Spawns VM for each CI job.
- Uses tracing to report status to backend.
- Launched VM is practically empty.
- Web interface for tracing (TBD).



zitadel/zitadel

Open-Source SSO using OIDC

- OAuth2.x
- SAML2
- LDAP
- Passkeys / FIDO2
- OTP



It’s that simple?!

devenv: systemd of developer environments

~~flakes~~
~~docker-compose~~
~~github actions~~
~~direnv~~ upcoming native support for background reloading
~~process-compose~~ upcoming native support for processes in TUI

OceanSprint.org April 2026

Thank you NixCon!

Private Beta

Sign up at **cloud.devenv.sh**

Community

cachix/cloud.devenv.sh
devenv.sh/discord
@devenv_nix