Python For Good

使用nix搭建可复现的python环境

黄毅
Senior blockchain developer

我是谁

- 一位老pythoner
- 不过最近用到python的主要是做区块链项目的集成测试

可复现是什么意思

- 在项目周期的不同阶段运行完全一致的代码：
  - 开发阶段，团队成员一致开发环境和本地测试环境
  - CI/CD的测试环境
  - 部署环境
- 尝试新的配置后能够安全的回滚到之前的状态
  - 安装新的python包后回滚
  - 升级c库后回滚到老版本
  - 生产环境升级回滚

可复现的挑战 - 具体问题

- 在所有环境使用完全一致的：
  - python实现
  - linter工具
- 在本地同时管理多个python版本
- 在本地切换一个python库的不同版本
- 在所有环境使用一致的c扩展库
- 切换不同版本的c扩展库

可复现的挑战

- Python生态本身
  - pre-virtualenv时期，一团乱麻
  - virtualenv+requirements.txt，能够管理纯python库，但是：
    - 不能管理扩展库依赖
    - 不能管理Python本身的编译
  - 其他，略
- 多语言混合开发，没有统一的包管理工具
- 系统包管理器本身就不可复现
- OS本身也不可复现

Nix是什么

- 想象一个由软件包组成的immutable hash graph
- 节点由所有软件包的所有版本组成，并以hash标识，hash包含在文件路径中
  - 不同内容的软件包hash不同
  - 同一个软件包的多个版本可以共存
- 节点之间按照绝对路径直接依赖，所以一个软件包的依赖发生变化，软件包本身也随之变化，比如：
  - 动态链接库引用绝对路径，LD_LIBRARY_PATH没用了
  - shebang引用绝对路径，不用 /usr/bin/env

Nix*

- nix store: hash graph存储
- nix-env: 包管理器CLI
- nix-shell: 终极virtualenv
- nix expression language
  - pure
  - lazy
  - functional
  - json-like

nix store

```
$ ls -1 /nix/store | grep "python3-3.8.5$"
71mscsq9qzsnwgyn2fw61qnmybss3pzr-python3-3.8.5

$ otool -L /nix/store/71ms...3pzr-python3-3.8.5/bin/python
...
   /nix/store/71ms...3pzr-python3-3.8.5/lib/libpython3.8.dylib
   /nix/store/6xv9...zbx3-ncurses-6.2/lib/libncursesw.6.dylib

$ head -n 1 /nix/store/9649...3i4l-python3.8-pytest-5.4.3/bin/pytest
#! /nix/store/58vs...rnik-bash-4.4-p23/bin/bash -e
```

nix-env

```
$ nix-env -iA nixpkgs.python38
  copy ... from cache.nixos.org
$ python --version
Python 3.8.6
$ nix-env -iA nixpkgs.python27
  copy ... from cache.nixos.org
$ python --version
Python 2.7.18
$ nix-env --rollback
switching from generation 45 to 44
$ python --version
Python 3.8.6
$ nix-env --switch-generation 45
switching from generation 44 to 45
$ python --version
Python 2.7.18
```

nix-shell

```
$ nix-shell -p python27
[nix-shell]$ python --version
Python 2.7.18
```

```
$ nix-shell -p python38
[nix-shell]$ python --version
Python 3.8.6
```

```
$ nix-shell -p python38 python3Packages.pytest
[nix-shell]$ pytest --version
This is pytest version 5.4.3, imported from /nix/store/9649...3i4l-python3.8-pytest-5.4.3/lib/python3.8/site-
packages/pytest/__init__.py
```

nix expression language

```
$ nix repl
nix-repl> 1 + 1
2

nix-repl> {a=1; b=2;}
{ a = 1; b = 2; }

nix-repl> rec {a=1; b=a+1;}
{ a = 1; b = 2; }

nix-repl> let a=1; in {a=a; b=a+1;}
{ a = 1; b = 2; }

nix-repl> with {a=1; b=2;}; a + b
3
nix-repl> with {a=1; b=2;}; "${toString a} + ${toString b} = ${toString (a + b)}"
"1 + 2 = 3"
```

nix expression language

```
nix-repl> fn = a: {a=a; b=a+1;}
nix-repl> fn 1
{ a = 1; b = 2; }

nix-repl> fn = {a, b ? a+1}: {inherit a b;}
nix-repl> fn {a=1;}
{ a = 1; b = 2; }

nix-repl> fix = f: let fixpoint = f fixpoint; in fixpoint
nix-repl> pkg = self: { a=1; b=self.a+1; }
nix-repl> fix pkg
{ a = 1; b = 2; }

nix-repl> withOverride = overrides: f: self: f (self // overrides)
nix-repl> fix (withOverride {a=2;} pkg)
{ a = 2; b = 3; }
```

derivation

derivation包含build一个包需要的所有输入

$$\begin{tikzcd}
\boxed{\text{Nix expression}} \arrow[r, "\text{求值}"] & \boxed{\text{derivatio}} \arrow[r, "\text{Build}"] & \boxed{\text{output}}
\end{tikzcd}$$

nixpkgs: a giant set of packages

```
nix-repl> pkgs = import <nixpkgs> {}
nix-repl> pkgs.python38
«derivation /nix/store/425v4rxmdqsi6wsbfsv5k3rxfb1yi5ir-python3-3.8.6.drv»
nix-repl> pkgs.python38.outPath
"/nix/store/5nwzb4f3bybv1ny1zn233smz6vfa8aq6-python3-3.8.6"

nix-repl> pkgs.python38.builder
"/nix/store/58vsldc9lzkrr0lwiinn84rfz7i5rnik-bash-4.4-p23/bin/bash"

nix-repl> pkgs.python38.src.outPath
"/nix/store/pfpaiyl1nx0g37lxrmw33krwjxh1dhds-Python-3.8.6.tar.xz"

nix-repl> :b pkgs.python38
...
this derivation produced the following outputs:
  out -> /nix/store/5nwzb4f3bybv1ny1zn233smz6vfa8aq6-python3-3.8.6
```

override in action

```
nix-repl> py38 = pkgs.python38.override {
  openssl = null;
}
nix-repl> :b py38
...
this derivation produced the following outputs:
  out -> /nix/store/4mvrxghhrlxcha825j8rcb3l8hdpi133-python3-3.8.6
```

nix-shell进阶

shell.nix

```
with (import <nixpkgs> {});
mkShell {
  buildInputs = with python3Packages; [
    poetry
    flake8
    black
    isort
    pytest
  ];
  shellHook = ''
    export PYTHONPATH=./src:$PYTHONPATH
  '';
}
```

```
$ nix-shell
[nix-shell] $
```

poetry2nix

shell.nix

```
with (import <nixpkgs> {});
poetry2nix.mkPoetryEnv {
  projectRoot = ./.;
  editablePackages = {
    hello = ./src;
  };
}
```

```
$ poetry add dep
$ poetry remove dep
$ nix-shell
```

closure

```
$ nix-store -qR /nix/store/pmgqxlxk6cmcm0b1hlqpv8pb8rnnb3l6-hello-2.10
...

$ nix-copy-closure user@remote /nix/store/pmgq...b3l6-hello-2.10
```

build docker image

closure is added automatically

```
with (import <nixpkgs> {});
let
  app = poetry2nix.mkPoetryApplication { projectRoot = ./.; };
in
dockerTools.buildLayeredImage {
  name = “awmsome image”;
  contents = [
    postgresql
    redis
    ...
  ];
  config.Entrypoint = [ "${app}/bin/run-app" ];
}
```

```
$ docker load $(nix-build -Q)
```

PyCon China 2020

Python For Good

PyConChina 2020 | PYTHON 中国开发者大会 2020

THANK YOU