Изграждане на пакети върху flake системата

Универсални конфигурации с Nix

Павел Атанасов Камен Младенов

20.05.2025

Преговор

- Запознахме се с flake-овете
- Запознахме се с V3 команди
- Разгледахме какво са overlay функции

```
{
  inputs = {
    a.url = "github:edolstra/dwarffs";
    b.url = "path:./stuff/myflake";
  };
  outputs = { self, ... }@inputs: {
    packages."<system>"."<name>" = derivation;
    legacyPackages."<system>"."<name>" = derivation;
    apps."<system>"."<name>" = derivation;
    devShells."<system>"."<name>" = derivation;

    nixosModules."<name>" = { config, ... }: { options = {}; config = {}; };
    nixosConfigurations."<hostname>" = {};

    overlays."<name>" = final: prev: { };
  };
}
```

Раздел 1

Задачи

Задачи

- Ако не е казано друго, може да използвате функции в `builtins` и `nixpkgs`
- **Не** може да използвате (други) външни библиотеки за пакетиране
- Използвайте интернет, особено https://search.nixos.org/packages и https://search.nixos.org/options

Тривиални flake-ове

Задача 1: *В тази задача НЕ можете да използвате* ***`nixpkgs`*** *(но може* ***`builtins`****)!*
Реализирайте flake без входове и който връща два атрибута:

- `separator` - низ, който съдържа една запетая
- `columns` - функция, която приема *три* аргумента: низ, начална колона (първата е номер едно) и крайна колона. Използвайки `separator`, низът трябва да се раздели на колони (с разделител `separator`), от тях да се премахнат тези между началната и крайната (включително) и колоните да се обединят обратно в един низ (с разделител `separator`). Върната стойност е този финален низ.

Пример

Извикване

```
columns "First,Second,Third,Fourth" 3 4
```

Върната стойност

```
"First,Second"
```

Задача 2: Реализирайте flake, който приема flake-а от предходната задача като вход и връща атрибут `columns`. Това е функция, подобна на тази от предходната задача, която обаче приема път към файл и премахва колоните от всеки **ред** на файла.

Пример

./example.csv

```
First,Second,Third,Fourth,Fifth
Apple,Bottom,Jeans,Looking,At
```

Извикване на `columns`

```
columns ./example.csv 2 3
```

Резултат

```
First,Fourth,Fifth
Apple,Looking,At
```

Задача 3: Реализирайте flake, който приема flake-овете от предходните две задачи и връща атрибут `columns`, който работи подобно на предходните, обаче първия аргумент може да бъде низ **или** път. Функцията трябва да избере коректната **вече направена** имплементация от зависимостите спрямо типа.

Семантични flake-ове

Задача 4: Реализирайте flake, който приема `nixpkgs` и връща три **пакета**: `english_text_normalization`, `ts` и `cloak`. Тези пакети бяха дефинирани в лекция 6 “Функции над `mkDerivation`” (може да намерите тогавашните реализации тук, но ще са нужни модификации).

Не може да използвате `<nixpkgs>`!!! Използвайте този `nixpkgs`, който сте дефинирали като вход.

Всеки пакет трябва да може да бъде компилиран чрез `nix build` и пуснат чрез `nix run`. Примерно: `nix run .#cloak`

Задача 5: Реализирайте flake, който приема `nixpkgs` и връща две NixOS конфигурации.

1. Първата е на име `smiths` и е модуляризираната конфигурация, която бе дефинирана от задача 5 на занятие (упражнение) 10 “Изграждане на системи върху NixOS” (може да намерите решение на задачата тук).
2. Втората е на име `johnny` и е немодуляризираната конфигурация, която бе разглеждана на лекция 9 “Конкетики в модулната система на NixOS” (може да я намерите тук).

Задача 6: Към flake-а от предходната задача добавете като изход модулите `international` и `rust-dev-improved`, които бяха дефинирани в задачи 7 и 8 от занятие (упражнение) 10 “Изграждане на системи върху NixOS” (може да намерите решения на задачите тук).

Задача 7: Реализирайте нов flake, който връща NixOS конфигурация на име `modo`, която е модуляризираната конфигурация от лекция 9 “Конкетики в модулната система на NixOS” (може да я намерите тук).

Допълнително нека да приема и използва модулите `international` и `rust-dev-improved`, които дефинирахме в предходната задача. Интернационалните локации трябва да бъдат `Sofia` и `London`. `rust-dev-improved` трябва да добавя програмите си **само** за потребител `john`.

Задача 8: Разширете flake-а от задача 4, така че да връща и две overlay функции:

1. Първата наименувана `mypkgs`, която би добавила трите пакета в `nixpkgs`
2. Втората наименувана `acepe`, която променя опциите `lexiconPath` и `pname` на пакета ape

Задача 9: Разширете flake-а от задача 7, така че overlay функциите, дефинирани в предходната задача, да бъдат приложени и използвани.