

Nix

Язык, пакетный менеджер и экосистема

Александр Бантьев



Section 1

Вступление

Что такое nix?

Nix - это чистый функциональный язык с динамическим типизированием и пакетный менеджер на основе этого языка.

Философия Nix как пакетного менеджера - атомарность, повторяемость и чистота сборки.

Как язык, Nix отличается упором на ленивые вычисления и ориентированностью на данные: большая часть кода на Nix представляет из себя напоминающие JSON множества атрибутов (attrsets).

Экосистема



Section 2

Nix как язык

Nix как язык

Рассмотрим основы синтаксиса Nix.

Краткая инструкция для желающих повторить

```
$ curl https://nixos.org/nix/install | sh  
<...>  
$ nix repl
```

Числа, арифметика и комментарии

```
nix-repl> :t 4 # Показать тип выражения
```

```
an integer
```

```
nix-repl> 1 + 2 - 3 * 2
```

```
-3
```

```
nix-repl> 1 / 2
```

```
0
```

```
nix-repl> builtins.add 1 2
```

```
3
```

```
nix-repl> # Однострочный комментарий
```

```
nix-repl> /* Многострочный комментарий */
```

Замечание

Арифметические операторы лучше обносить пробелами! У символов / и - есть особые значения в Nix

```
nix-repl> 6/2
```

```
/home/balsoft/6/2
```

Булевы, булева арифметика и if

```
nix-repl> :t true
a boolean
nix-repl> true && false
false
nix-repl> true || false
true
nix-repl> if true || false then 3 else 4
3
nix-repl> if false then 5 else 6
6
```


Строки (strings)

```
nix-repl> "foo"
"foo"
nix-repl> :t "foo"
a string
nix-repl> "Привет"
"Привет"
nix-repl> ``foo``
"foo"
nix-repl> ``
      foo
      bar
      ``
"foo\nbar\n"
```

Переменные и строковая интерполяция

```
nix-repl> foo = "hello"  
nix-repl> "${foo}, world!"  
"hello, world!"
```

Замечание

Названия переменных могут содержать знак -

```
nix-repl> foo-bar = 10  
nix-repl> foo-bar  
10
```

Но не могут содержать не-ASCII!

```
nix-repl> привет  
error: syntax error, unexpected $undefined, at (string):1:1
```

Как избежать интерполяции?

Довольно часто встречаются ситуации, когда мы хотим вставить в строку последовательность `${...}` (например, в `bash`-код).

```
nix-repl> "\${foo}"  
"${foo}"
```

```
nix-repl> 'test '${foo} test'  
"test ${foo} test"
```

Пути (paths) и URL

```
nix-repl> /bin/sh
/bin/sh
nix-repl> :t /bin/sh
a path
nix-repl> ./hello # Относительно текущего .nix файла или PWD для repl
/home/balsoft/hello
nix-repl> https://typeable.io
"https://typeable.io"
nix-repl> :t https://typeable.io
a string
```

Замечание

Пути, как и названия переменных, не могут содержать не-ASCII. Я бы рассматривал это, как баг.

```
nix-repl> /home/balsoft/Документы
error: path '/home/balsoft/' has a trailing slash
```

Списки (lists)

Списки в Nix иммутабельны и могут содержать значения любых типов вперемешку.

```
nix-repl> [ 2 "foo" true (2+3) ]  
[ 2 "foo" true 5 ]
```

Замечание

Обратите внимание: элементы списка не разделены запятыми!

```
nix-repl> [ 1 + 2 ]  
error: syntax error, unexpected '+', at (string):1:5  
nix-repl> [ (1 + 2) ]  
[ 3 ]  
nix-repl> [ import ./hello.nix {} ]  
[ «primop-app» /home/balsoft/hello.nix { ... } ]
```

Множества атрибутов (attribute sets, attrsets)

```
nix-repl> s = { foo = "bar"; a-b = "baz"; "123" = "num"; }
nix-repl> s
{ "123" = "num"; a-b = "baz"; foo = "bar"; }
nix-repl> s.a-b
"baz"
nix-repl> s."123"
"num"
nix-repl> s.${toString (122 + 1)}
"num"
nix-repl> s.abc or "not found" # Note that `or` is a keyword
"not found"
nix-repl> s // { a = 10; }
{ "123" = "num"; a = 10; a-b = "baz"; foo = "bar"; }
nix-repl> rec { a = 3; b = a + 4; } # Recursive attrset
{ a = 3; b = 7; }
```

Let n with

```
nix-repl> let a = 3; b = 4; in a + b
```

```
7
```

```
nix-repl> let a = 4; b = a + 5; in b # Let definitions are recursive
```

```
9
```

```
nix-repl> let a = 3; a = 8; in a
```

```
error: attribute `a' at (string):1:12 already defined at (string):1:5
```

```
nix-repl> longName = { a = 3; b = 4; }
```

```
nix-repl> longName.a + longName.b
```

```
7
```

```
nix-repl> with longName; a + b
```

```
7
```

Функции (functions)

```
nix-repl> x: x*2
«lambda»
nix-repl> double = x: x * 2
nix-repl> :t double
a function
nix-repl> double 3
6
nix-repl> greet = end: name: "Hello, ${name} ${end}"
nix-repl> greet "dredozubov" "!"
"Hello, dredozubov !"
nix-repl> greetExclamaition = greet "!"
nix-repl> greetExclamaition "s9gf4ult"
"Hello, s9gf4ult !"
```


Паттерн-матчинг

```
nix-repl> greet = { name, end }: "Hello, ${name} ${end}"
nix-repl> greet { name = "ak3n"; exc = "!"; }
"Hello, ak3n !"
nix-repl> greet = { name, end ? "!" }: "Hello, ${name} ${end}"
nix-repl> greet { name = "ak3n"; }
"Hello, ak3n !"
nix-repl> greet { name = "ak3n"; end = "."; }
"Hello, ak3n ."
nix-repl> greet = { name, end ? "!", ... }: "Hello, ${name} ${end}"
nix-repl> greet { name = "ak3n"; end = "."; extra = true; }
"Hello, ak3n ."
nix-repl> foo = {a, ...}@args: args // { a = a + 3; }
nix-repl> foo { a = 10; b = 11; }
{ a = 13; b = 11; }
```

Подключения файлов (import)

```
number.nix
```

```
3
```

```
function.nix
```

```
a: a + 3
```

Обратно в наш уютный repl

```
nix-repl> a = import ./number.nix  
nix-repl> f = import ./function.nix  
nix-repl> f a  
6
```

Деривации (derivation)

```
nix-repl> d = derivation {  
    name = "myname";  
    builder = "mybuilder";  
    system = "mysystem";  
}  
nix-repl> :t d  
a set  
nix-repl> d  
«derivation /nix/store/z3hhlxbckx4g3n9sw91nnvltkjvyw754p-myname.drv»
```

Лирическое отступление

На этом этапе нужно рассмотреть, что же представляет из себя `nix-the-package-manager`.

Section 3

Пакетный менеджер Nix

Хранилище (nix store)

In the beginning was the Store, and the Store was with Nix, and the Store was Nix.

– *Nix manual, 1:1*

```
$ ls /nix/store | wc -l
```

```
46251
```

```
$ nix add-to-store nix.conf
```

```
/nix/store/za34q8y0jcx3qsrnbrd005mh8zcnlr1r-nix.conf
```

Деривации

```
$ cat /nix/store/z3hhlxbckx4g3n9sw91nnvlkjvyw754p-myname.drv
Derive
([("out", "/nix/store/40s0qmrfb45vlh6610rk29ym318dswdr-myname", "", "")]
, []
, []
, "mysystem"
, "mybuilder"
, []
, [("builder", "mybuilder")
, ("name", "myname")
, ("out", "/nix/store/40s0qmrfb45vlh6610rk29ym318dswdr-myname")
, ("system", "mysystem") ] )
```

Принцип работы



На эту диаграмму ушло четыре с половиной часа. Ненавижу `diagrams`, `diagrams-pandoc` и `svg-fonts`

Section 4

Заключение

Ссылки

<https://nixos.org/> <https://imgur.com/0gPQ4zg>