Demo

...but let’s first talk about setups
State of Lean 3 Setup

- lean+leanpkg: via GitHub Releases, or elan
- elan: Lean version manager, via curl | sh
- leanproject: store & restore *.olean tars, via PyPI
- your favorite editor + Lean extension
IT HAS BEEN

00000 DAYS

SINCE THE LAST BROKEN LEAN SETUP
State of Lean 3 Setup

- lean+leanpkg: via GitHub Releases, or elan
- elan: Lean version manager, via curl | sh
- leanproject: store & restore .olean tars, via PyPI
- your favorite editor + Lean extension

Look forward to further dependency fun in Lean 4:

- C compiler chain
- external FFI libraries
- Lean dependencies compiled into plugins
...and I don’t want to write any of this

Can’t someone else do that for us?
Nix

Nix is a purely functional build system & package manager

- hermetic, reproducible builds against pinned dependencies
- built-in support for remote binary caches & distributed builds
- the world’s largest package repository

...works on Linux, macOS and WSL
Sebastian’s Experiment: Nix-Based Lean Setup

Perfect .olean/.c/.o local & remote caching
Perfectly reproducible Lean development setup

- change branches without losing progress
- build while changing branches, build commits without checkout

https://lean4.cachix.org/
Nix Setup: Unresolved Questions

- Does it scale?
- Is it actually simpler/more robust?
- Relationship with leanpkg.toml?
Demo

for real this time
partial def pown (a : Int) : Int → Int

| 0 => 1
| 1 => a
| n =>

let b := pown a (n / 2)

b * b * (if n % 2 = 0 then 1 else a)

partial def add : Expr → Expr → Expr

| Val n, Val m => Val (n + m)
| Val 0, f => f
| f, Val 0 => f
| f, Val n => add (Val n) fa
| Val n, Add (Val m) f => add (Val (n+m)) f

PROBLEMS 4 ...

Filter (e.g. text, **/*.ts, !**/node_modules/**)

unknown identifier 'fa' Lean 4 server [25, 47]