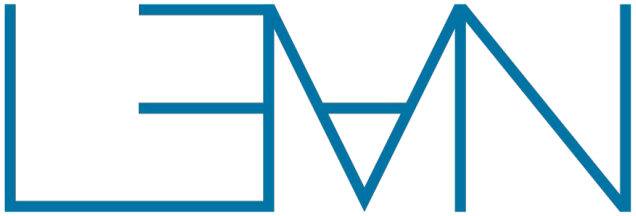
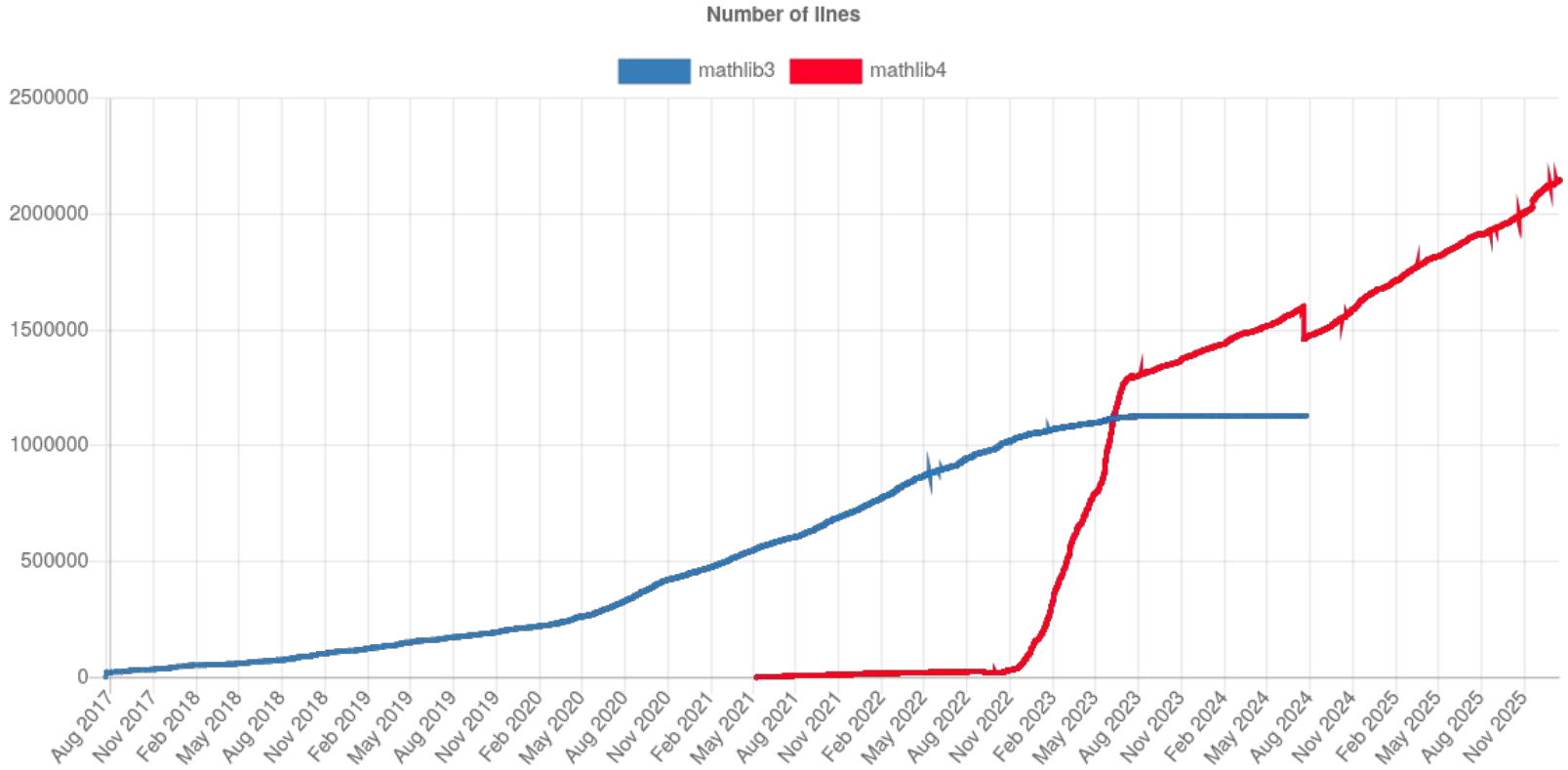


The Lean Module System

Sebastian Ullrich, Lean FRO

Lean Together, Jan 20, 2026

Mathlib growth, never-ending…?



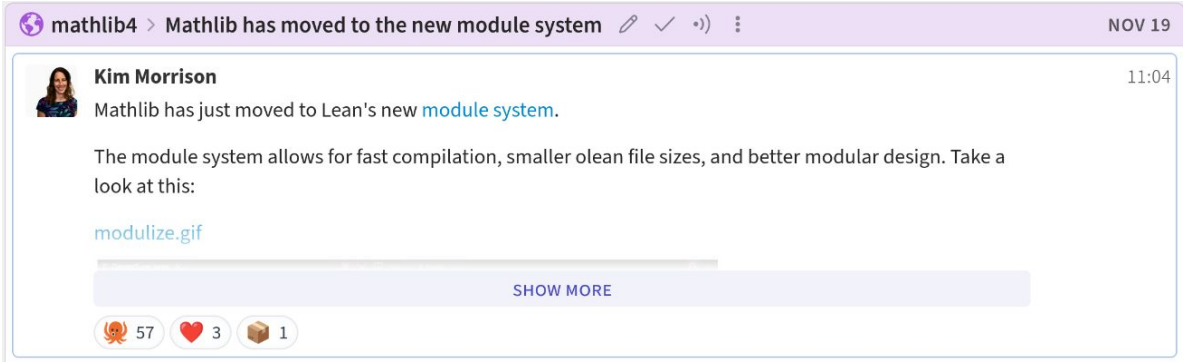
The module system

A system of language restrictions to address scalability and maintenance issues

Experimental version since Lean 4.22, stabilized in 4.27.0-rc1

Opt-in via `module` header keyword, to be introduced top-down

Fully adopted by `Init`, `Std`, `Lean`, `Cslib`, and `Mathlib` + dependencies



feat: move to the module system (cslib#243)

Large changes (1✅)

- ✅ `build//instructions`: -94.8G (-5.2%)

Module system basics

- Declarations *and imports* now private by default; adjust using `public`
- `def` bodies are also private to the module unless `@[expose]`d
- Proofs are always private
- All metaprograms need to be `meta` annotated/imported

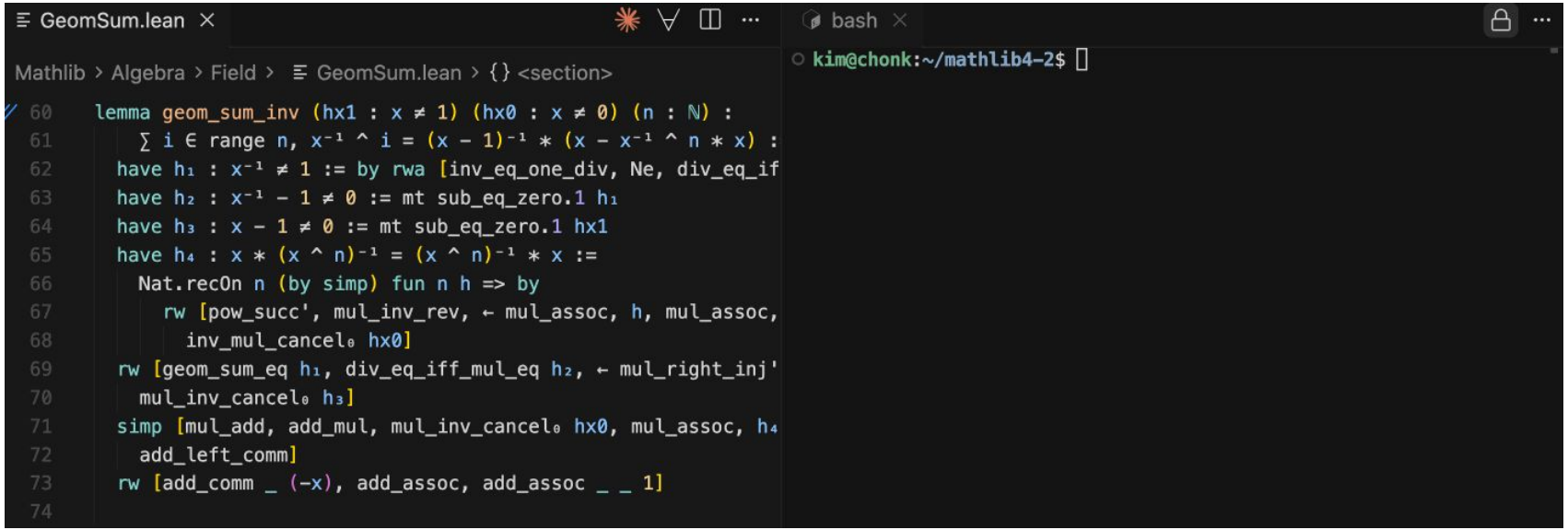
=> Much more precise control over what information a module exposes

More info: Reference Manual §5.4+, Lean Hackathon keynote #2 (cf. talk abstract)

Module system goals

Build times scalability

Changes limited to private data => no need to recompile anything else



Module system goals

Build times scalability

Changes limited to private data => no need to recompile anything else

Public changes propagate through `public imports` but not beyond!

Module system goals

Output size scalability

By default only public interface has to be (down)loaded [download **TBD** Q1'26]

`Mathlib` breakdown:

1.8GB public scope data
+ 0.2GB IR data for metaprogram execution
+ 0.1GB metadata for language server
+ 2.8GB private scope data

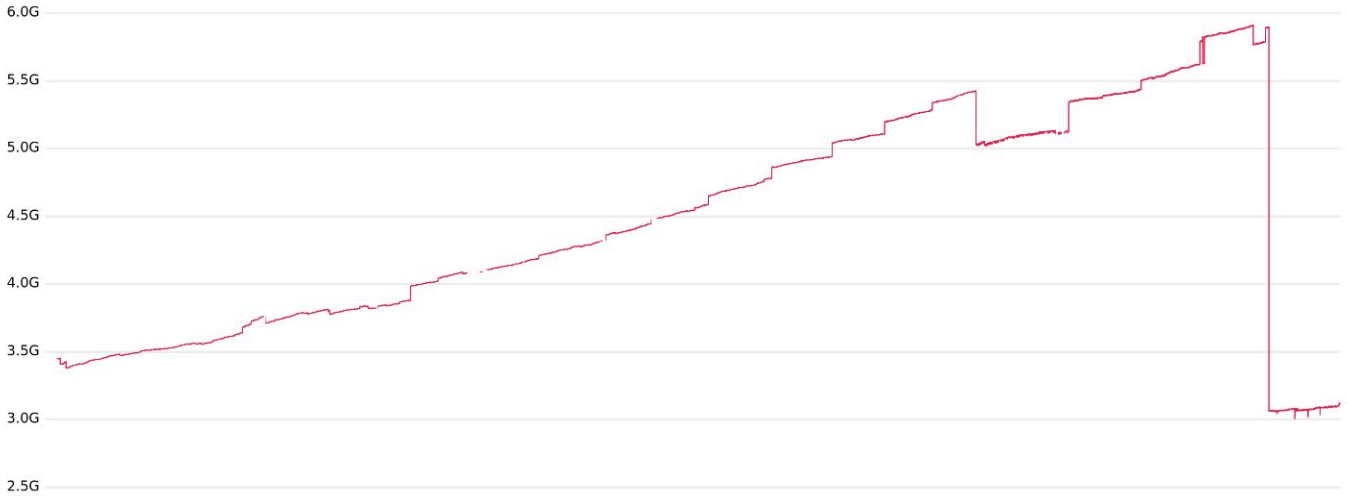
Private imports can further minimize the (down)load closure, drastically

Module system goals

Output size scalability

By default only public interface has to be (down)**loaded**

Pleasant surprise: RAM use of loading `Mathlib` fell by 48% from this



Module system goals

Unfolding scalability

Enforced through public interface barrier and enticed by new `def` defaults

People are already writing proofs that require the module system

▼ Derivative.lean:546:8

(kernel) deterministic timeout

▼ Derivative.lean:546:8

type checking took 89.1s

type checking took 3.69s

Lean 4

Polynomial.iterate_derivative_derivative_mul_X_sq

Module system goals

Library design scalability

Private changes cannot influence other code

chore(CStarAlgebra): only import Liouville's theorem privately (mathlib4#32561)

Major changes (1)

- ✅ `build/module/Mathlib.Analysis.CStarAlgebra.ApproximateUnit//instructions`: **-14.9G** (-7.9%)

How to port 2M Lines to the module system

`script/Modulize.lean` automates the header syntax changes

`public @[expose] section` to mostly preserve remaining file's semantics

Metaprograms need `meta` annotations, sometimes reorganizing and adjusting

Prior uses of `private` unlikely to conform to new scope checks

- `backward.privateInPublic` keeps them in the public scope

Cleaning up with `lake shake` (Lean 4.28)

New version of Mathlib's import minimizer: understands both module system imports and implicit metaprogramming dependencies

617 lines of broad `noshake.json` exclusion replaced by
77 in-source annotations

[Merged by Bors] - chore: run the new `shake` tool #32731

Closed **Kha** wants to merge 22 commits into `leanprover-community:master` from `Kha:push-nzxsuxolzzrk`

Conversation 31 | Commits 22 | Checks 11 | Files changed 300+

Further cleaning up import closures

`lake shake` as configured for Mathlib does not yet attempt to remove `public` from imports, which could result in many more downstream imports

We should concentrate on reducing those on the *rebuild critical path*

mathlib4 v | Overview Graph Queue About

Commit

commit f760e25ec70bae816086c65e47146be6c394a7bd (source, diff)
Author: Yury G. Kudryashov <188813+urkud@users.noreply.github.com>
Date: 2026-01-20 02:13:31 (7 hours ago)

- feat(NNReal): add lemmas about images of intervals (#34126)

Lakeprof report
Parent: *chore: tidy various files (#34081)*
Child: *> chore: golf using `grind` and `simp` (#33843)*

rebuild critical path

time [s]		[cum]		drv
0.9	0.1%	0.9	0.1%	Batteries.Data.Array.Merge
0.4	0.1%	1.3	0.2%	Aesop.Util.UnorderedArraySet
1.8	0.3%	3.1	0.5%	Aesop.Util.Basic
0.9	0.1%	4.0	0.6%	Aesop.Tracing
0.9	0.1%	4.9	0.8%	Aesop.Stats.Basic
3.5	0.6%	614.3	96.6%	Mathlib.NumberTheory.LSeries.HurwitzZetaEven
1.8	0.3%	616.1	96.9%	Mathlib.NumberTheory.LSeries.HurwitzZeta
1.9	0.3%	618.0	97.2%	Mathlib.NumberTheory.LSeries.RiemannZeta
2.3	0.4%	620.3	97.6%	Mathlib.NumberTheory.LSeries.Dirichlet
1.9	0.3%	622.2	97.9%	Mathlib.NumberTheory.EulerProduct.DirichletLSeries
2.1	0.3%	624.3	98.2%	Mathlib.NumberTheory.LSeries.DirichletContinuation
2.6	0.4%	626.9	98.6%	Mathlib.NumberTheory.LSeries.Nonvanishing
3.3	0.5%	630.2	99.1%	Mathlib.NumberTheory.LSeries.PrimesInAP
5.6	0.9%	635.8	100.0%	Mathlib [private]

Conclusion

The module system introduces enforced information hiding to Lean

Benefits to API design, compilation speed & size, and disk & memory use

Now stable in the latest RC!