Monthly Community Meeting

Lean FRO
May 10, 2024

Our team

Leo de Moura (AWS)
Chief Architect, Co-Founder

Sebastian Ullrich
Head of Engineering, Co-Founder

Corinna Calhoun
Chief Operating Officer

Henrik Böving
Research Software Engineer

Joachim Breitner
Senior Research Software Engineer

David Thrane Christiansen
Senior Research Software Engineer

Markus Himmel
Research Software Engineer

Marc Huisinga
Research Software Engineer

Mac Malone
Research Software Engineer

Kyle Miller
Research Software Engineer

Kim Morrison
Senior Research Software Engineer

Sofia Rodrigues
Research Software Engineer

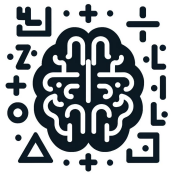
Lean FRO Mission

Our mission focuses on enhancing **Lean** in key areas: **scalability**, **usability**, **documentation**, and **proof automation**, while also broadening its application in various fields such as **education**, **research**, and **industry**. Over the next five years, we are dedicated to advancing Lean's capabilities and expanding its influence, ensuring it becomes an indispensable tool in these diverse domains. A pivotal aspect of our mission is to steer Lean towards **self-sustainability**, laying a strong foundation for its enduring growth and widespread utilization.

lean-fro.org/about/roadmap

Formal mathematics

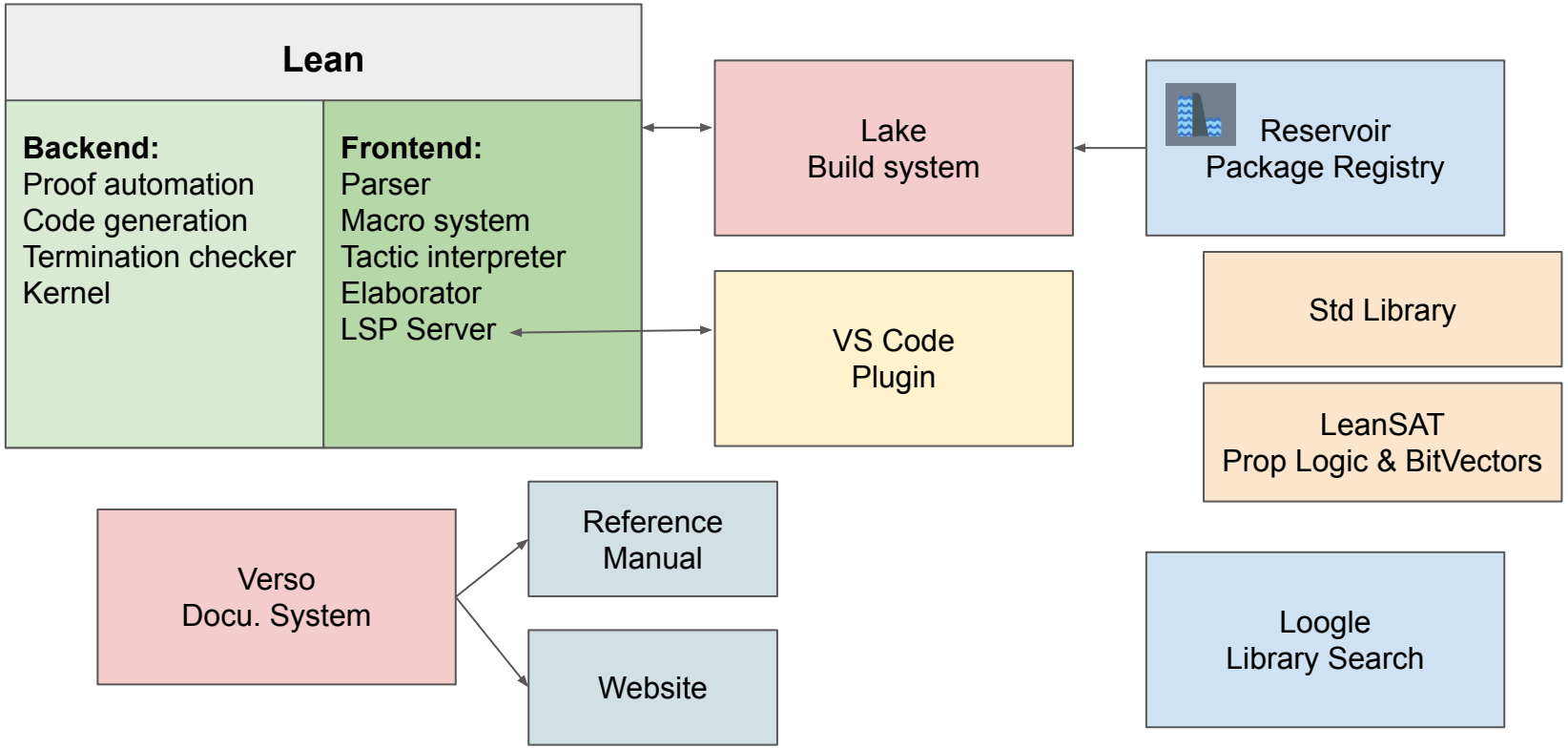
Software/Hardware verification



AI for math and code synthesis

Math and CS education

Lean


Lean
Backend:
Proof automation
Code generation
Termination checker
Kernel
Frontend:
Parser
Macro system
Tactic interpreter
Elaborator
LSP Server
Lake
Build system
Reservoir
Package Registry
VS Code
Plugin
Std Library
LeanSAT
Prop Logic & BitVectors
Verso
Docu. System
Reference
Manual
Website
Loogle
Library Search

News since last FRO community meeting

- Released v4.7.0: blog post
- Next release candidate is out v4.8.0-rc1, v4.8.0-rc2 coming soon.
- Markus Himmel joined the Lean FRO. Joe Hendrix moved to AWS.
- leanprover/std4 → leanprover-community/batteries

Lean @leanprover · Apr 10

AWS Clean Rooms Differential Privacy uses the SampCert sampler, a proven correct sampler implementation developed in Lean by AWS.
docs.aws.amazon.com/clean-rooms/la...

Lean @leanprover · Apr 8

This is a great blog post describing one of Lean's applications at AWS.

Lean 4 repo statistics since the last meeting

(March 8 to May 10)

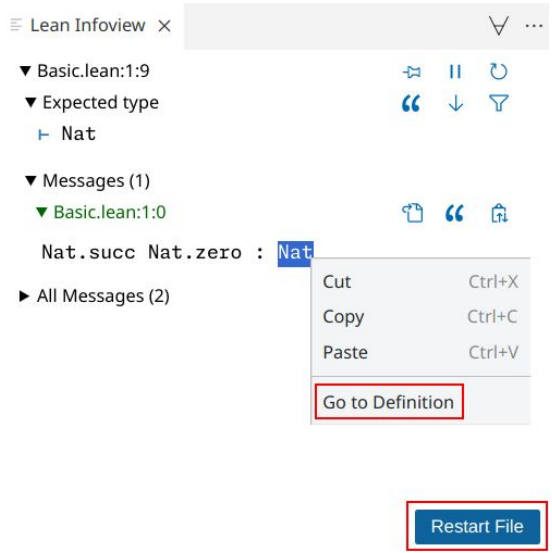
- 78 issues closed
- 67 new issues (48 of them were closed)
- 358 commits
- 374 PRs created (331 of them were closed)

leanprover/std4 → leanprover-community/batteries

- New Std is in the core repository, and will be distributed with Lean.
- Std is now using the Cathedral development model: managed by the FRO.
- Batteries is still using the Bazar development model: managed by the community.
- Scope of new Std
  - Basic types + theorems about them
  - Async IO
  - Networking
  - Internet protocols
  - Regular expressions
- Goals: coherence, no gaps, no core behavior overrides, similar to the Rust std.

VS Code Extension: New InfoView Features

- ‘Restart File’ button in InfoView to replace ‘Restart File’ notifications
- ‘Go to Definition’ button in InfoView context menu

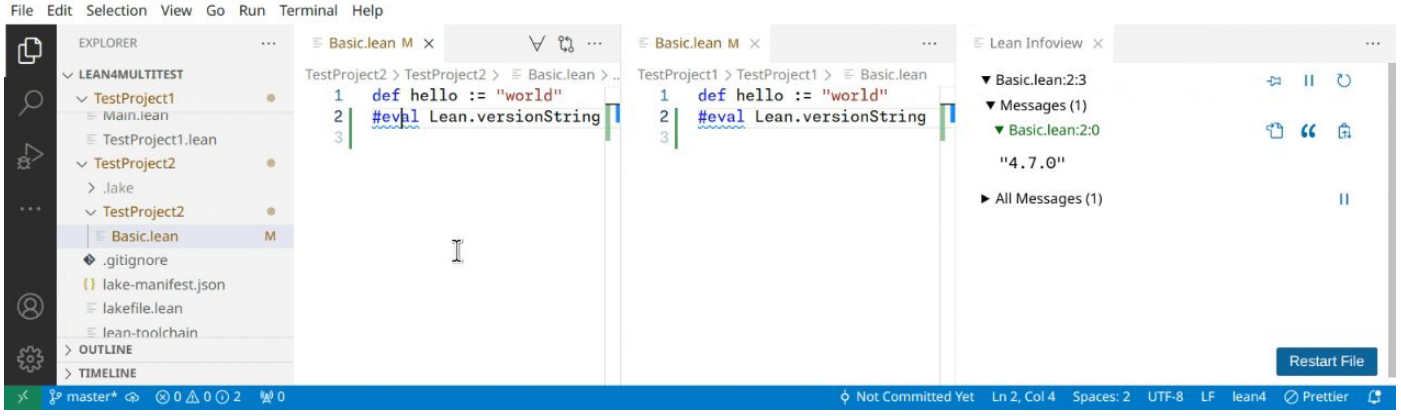


VS Code Extension: Multi-Toolchain Workspaces

The following now work correctly:

- Using multiple projects with different lean-toolchains at the same time in a single folder
- Using Lean files from a Lean project without first opening the project in VSC

Caveat: Nested Lean projects are not supported yet!

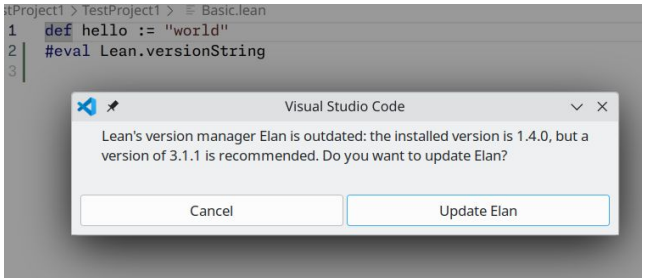


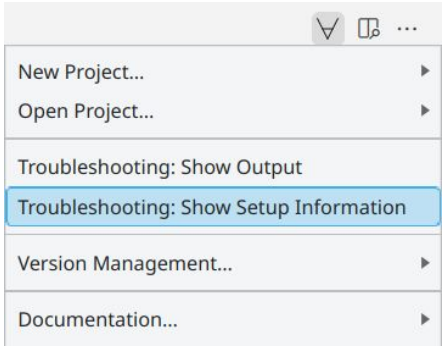
VS Code Extension: Better Setup Diagnostics (Pre-Release)

Problem: Users sometimes have outdated or subtly broken setups that are hard to debug, especially on Zulip

Solution 1: More warnings and error messages when setup issues are detected

Solution 2: New command to collect information about the user's setup





Operating system: Linux (release: 6.8.8-300.fc40.x86_64)
CPU architecture: x64
CPU model: 8 x Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz
Available RAM: 16.44 GB

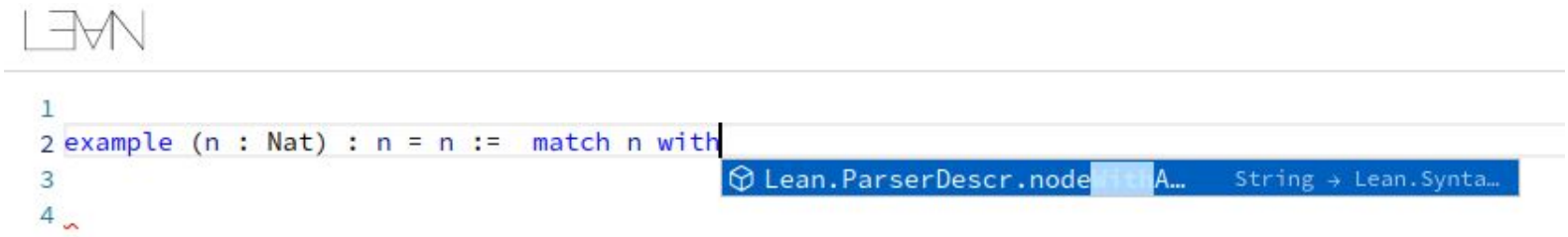
Curl installed: true
Git installed: true
Elan: Reasonably up-to-date (version: 3.1.1)
Lean: Reasonably up-to-date (version: 4.7.0)
Project: Valid Lean project (path: /home/mhuisi/Lean4Test16)

Elan toolchains:

```
installed toolchains
--------------------

leanprover/lean4:stable (default)
leanprover/lean4:v4.7.0
leanprover/lean4:v4.8.0-rc1

active toolchain
----------------

leanprover/lean4:v4.7.0 (overridden by '/home/mhuisi/Lean4Test16/lean-toolchain')

Lean (version 4.7.0, x86_64-unknown-linux-gnu, commit 6fce8f7d5cd1, Release)
```

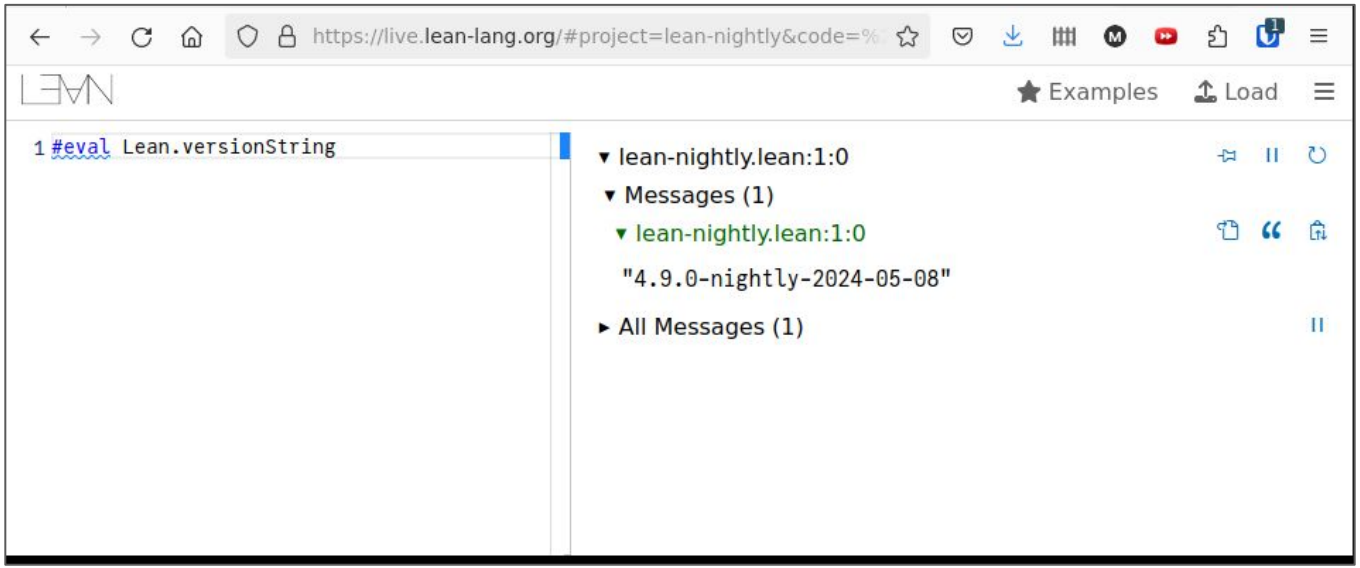
Language Server: Bug Fixes

- Semantic highlighting should not go stale anymore
- Go-to-definition jumps to correct definition when names clash globally
- No server-side auto-completion after keywords that messes with input



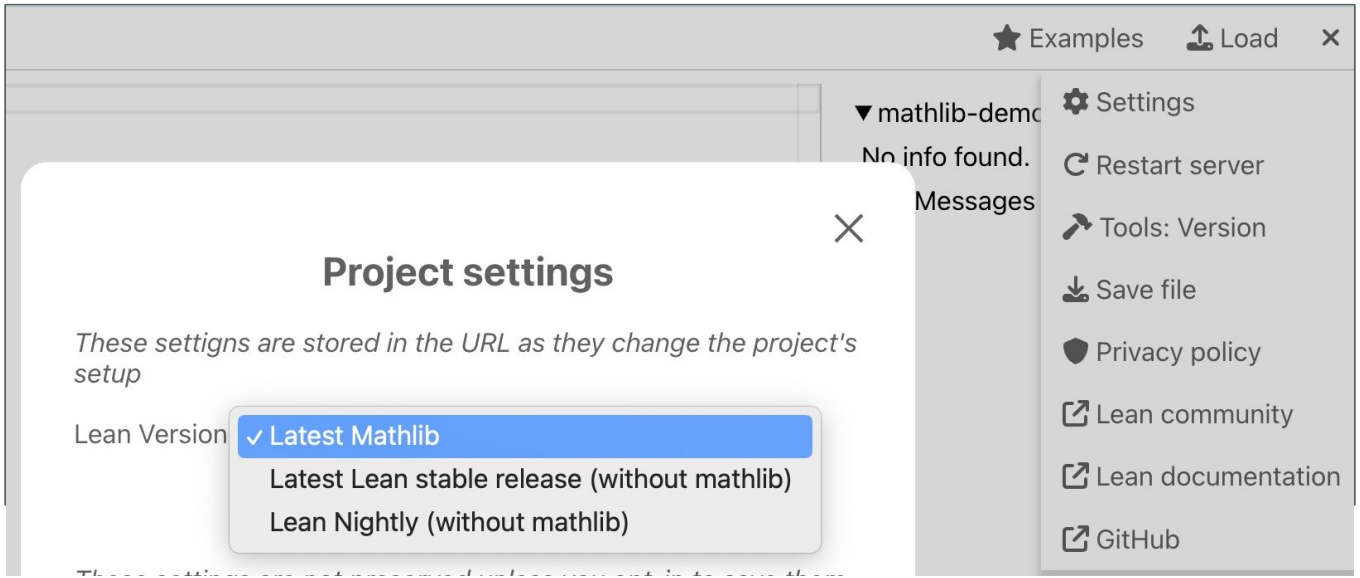
live.lean-lang.org: Now with lean nightly (and stable)

- See settings in top-right menu
- Useful when reporting bugs



live.lean-lang.org: Now with lean nightly (and stable)

- See settings in top-right menu
- Useful when reporting bugs



set_option diagnostics true

```
set_option diagnostics true in
theorem Submodule.toLocalizedQuotient'_mk (x : M) :
    M'.toLocalizedQuotient' S p f (Submodule.Quotient.mk x) = Submodule.Quotient.mk (f x) :=
      rfl
```

```
[reduction] unfolded declarations (max: 342322, num: 11): ▼
  toAddSubgroup ↦ 342322
  localized' ↦ 95382
  Set ↦ 67296
  setOf ↦ 61486
  SetLike.coe ↦ 13162
  Membership.mem ↦ 3884
  Set.Mem ↦ 3782
  AddSubgroup.op ↦ 3366
  Set.preimage ↦ 1836
  Set.range ↦ 24
  VAdd.orbit ↦ 24
[reduction] unfolded instances (max: 99144, num: 12): ▶
[reduction] unfolded reducible declarations (max: 357510, num: 5): ▶
[def_eq] heuristic for solving `f a =?= f b` (max: 42180, num: 11): ▶
use `set_option diagnostics.threshold <num>` to control threshold for reporting counters
```

set_option diagnostics true

```
def f (x : Nat) := x + 1
def g (x : Nat) := 1 + x

@[simp] theorem f_eq : f x = g x := by simp_arith [f, g]
@[simp] theorem g_eq : g x = f x := by simp_arith [f, g]

example : f (x + 1) = x + 2 := by
  set_option diagnostics true in
  simp
```

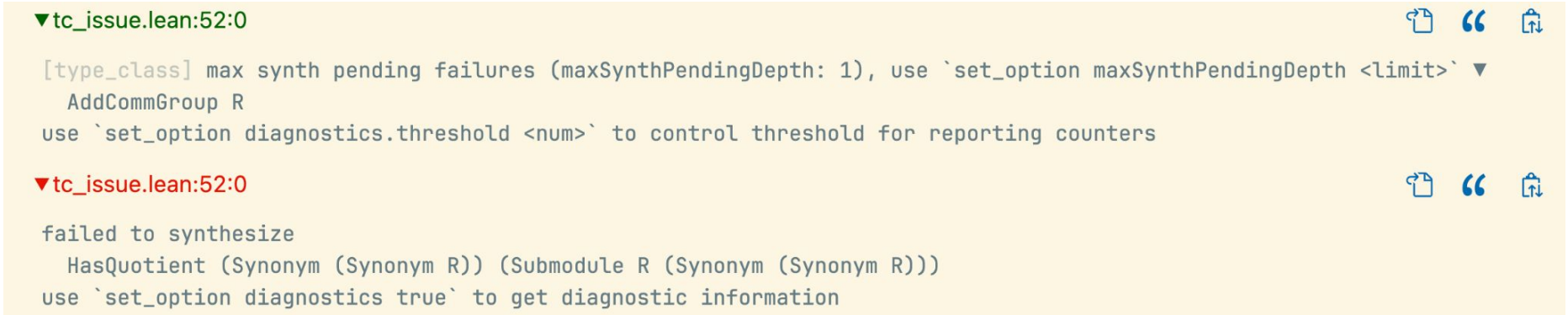
```
[simp] used theorems (max: 249, num: 2): ▼
  f_eq  @f_eq : ∀ {x : Nat}, f x = g x
  g_eq ↦ 249
[simp] tried theorems (max: 250, num: 2): ▶
use `set_option diagnostics.threshold <num>` to control threshold for reporting counters
```

▼simp_issue.lean:19:2

```
tactic 'simp' failed, nested error:
maximum recursion depth has been reached
use `set_option maxRecDepth <num>` to increase limit
use `set_option diagnostics true` to get diagnostic information
```

set_option diagnostics true

```
set_option maxSynthPendingDepth 1 in
set_option diagnostics true in
#synth HasQuotient (Synonym (Synonym R)) (Submodule R (Synonym (Synonym R)))
```



`seal` and `unseal` commands

```
set_option diagnostics true in
seal AddSubgroup.op Set in
theorem Submodule.toLocalizedQuotient'_mk (x : M) :
    M'.toLocalizedQuotient' S p f (Submodule.Quotient.mk x) = Submodule.Quotient.mk (f x) :=
      rfl
```

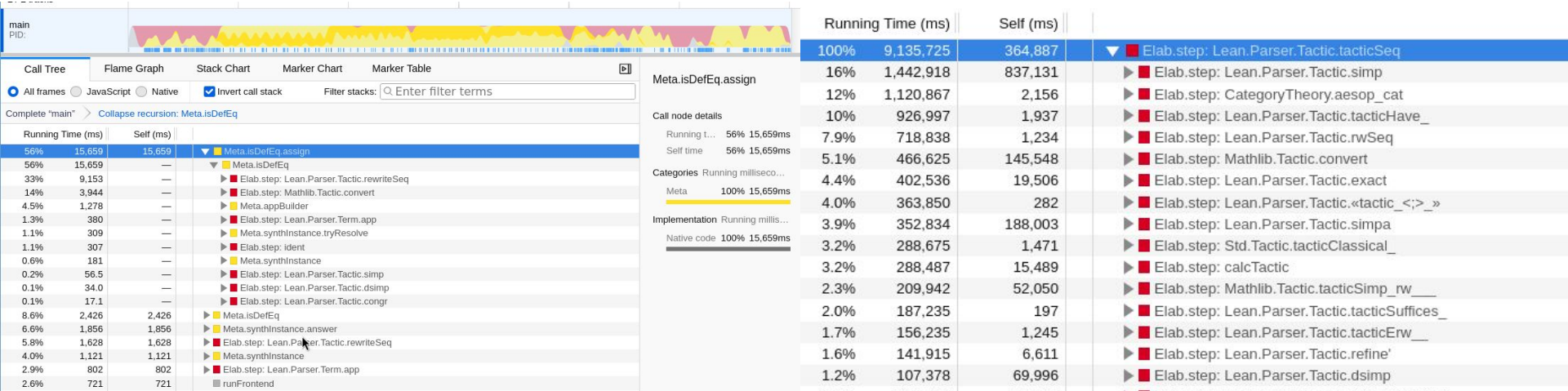
`trace.profiler.output`

A better way to inspect `trace.profiler` output visually and structurally:

```
lake env lean -Dtrace.profiler=true -Dtrace.profiler.output=out.json YourFile.lean
```

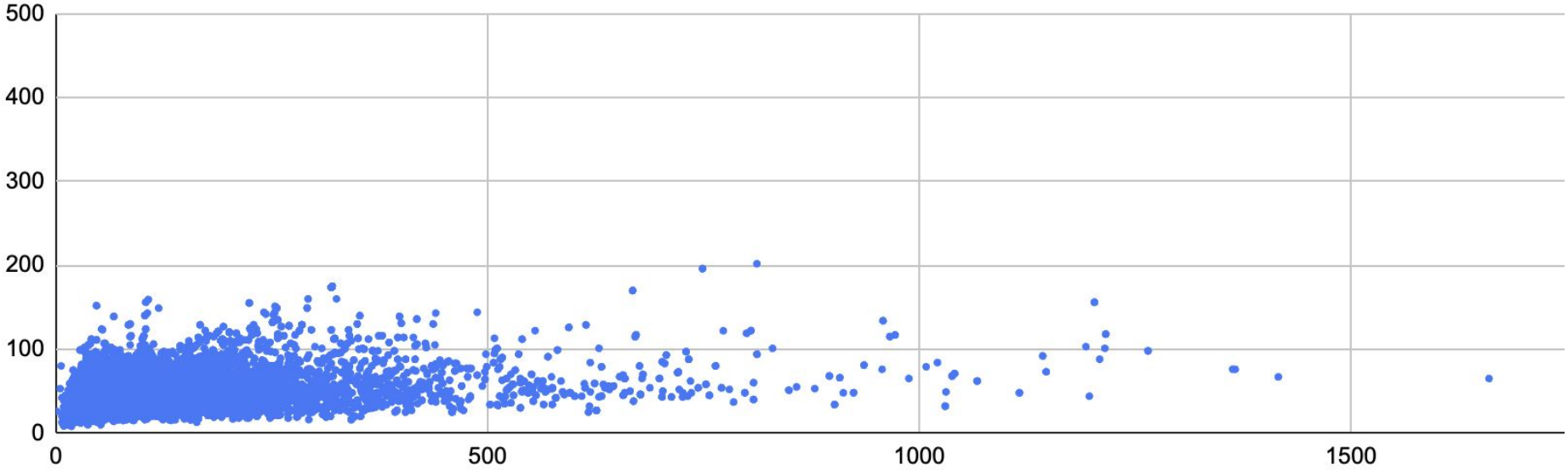
Then open `out.json` on https://profiler.firefox.com/ (diffs supported as well).

See also `trace.profiler.useHeartbeats`!



Shorter instances names

IsSelfAdjoint.instContinuousFunctionalCalculusRealIsSelfAdjointToStarToInvolutiveStarToAddMonoidToAddMonoidWithOneToAddGroupWithOneToRingToStarAddMonoidToNonUnitalNonAssocSemiringToNonUnitalNonAssocRingToNonAssocRingInstCommSemiringRealInstStarRingRealToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToNonUnitalNonAssocCommRingToNonUnitalCommRingCommRingMetricSpaceToTopologicalSemiringToTopologicalSpaceToUniformSpaceToPseudoMetricSpaceToNonUnitalNonAssocRingToNonUnitalNonAssocCommRingToNonUnitalCommRingToNonUnitalSeminormedCommRingToSeminormedCommRingNormedCommRingInstTopologicalRingRealToTopologicalSpaceToUniformSpacePseudoMetricSpaceToNonUnitalNonAssocRingToNonUnitalNonAssocCommRingToNonUnitalCommRingCommRingInstContinuousStarRealToTopologicalSpaceToUniformSpacePseudoMetricSpaceToStarToInvolutiveStarInstAddMonoidRealToStarAddMonoidToNonUnitalNonAssocSemiringToNonUnitalNonAsonAssocSemiringToSemiringToTopologicalSemiringToNonUnitalNonAssocRingToNonUnitalNonAssocCommRingToNonUnitalCommRingToNonUnitalSeminormedCommRingToTopologicalRingToDivisionRingToNormedDivisionRingTo_topologicalDivisionRingTo_continuousConstSMulToSMulIdUniformContinuousConstSMulInstRingComplexTo_uniformAddGroupToSeminormedAddCommGroupToNonUnitalSeminormedRingToContinuousMulToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToAddCommMonoidToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToNonUnitalNonAssocCommRingToNonUnitalCommRingToNonUnitalSeminormedCommRingToSeminormedCommRingToModuleToSeminormedAddCommGroupToNonUnitalSeminormedRingToNormedSpace'ToNormedRingToTopologicalSpaceToUniformSpaceToPseudoMetricSpaceToSeminormedRingCharacterSpace

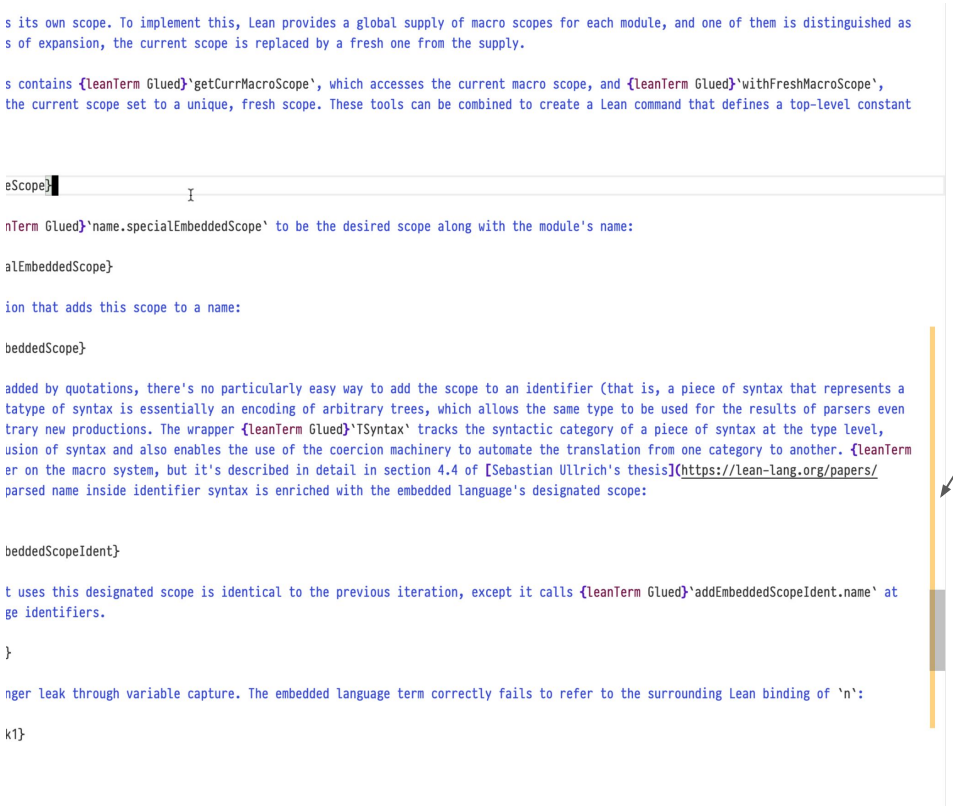


LeanSAT + verified bit-blaster

```
example {x y : BitVec 2048} (h : x = y) : (~~~x) &&& y = (~~~y) &&& x := by
  bv_decide
```

```
set_option trace.bv true in
theorem unit_6 {x y : BitVec 256} : x + y = y + x := by
  bv_decide
```

Verso: Incremental Elaboration



Yellow bar shows progress, just as in ordinary Lean

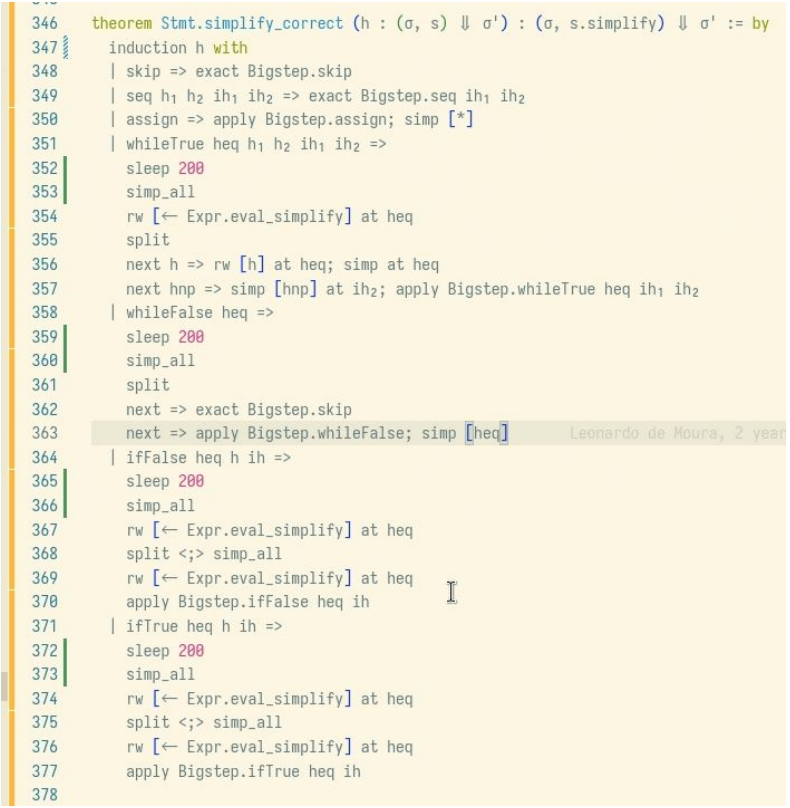
Demonstration that non-Lean DSLs can also support incrementality

Lean Incremental Elaboration: Call for Testing

Significantly reduces reprocessing and reporting delay inside tactic-mode theorems

Target: 4.9.0, help finding issues to make it a reality!

See #lean4 > Incrementality Call for Testing



`lakefile.toml`

- Lake now supports TOML as an alternate format for configuration files.
- Some packages within the Mathlib dependency chain have already moved to TOML (e.g., Aesop, ImportGraph).
- New Lake features to enabling moving Mathlib and all of its dependencies to TOML are in the works.
- Automatic migration is available via `lake translate-config toml`. It drops unsupported features but keeps the original configuration file.

```
name = "aesop"
defaultTargets = ["Aesop"]
precompileModules = false

[[require]]
name = "batteries"
git =
"https://github.com/leanprover-community/batteries"
rev = "main"

[[lean_lib]]
name = "Aesop"

[[lean_lib]]
name = "AesopTest"
globs = ["AesopTest.+"]
leanOptions = {linter.unusedVariables = false}
```

Lake Build Refactor

- Lake also had a substantial change to the way builds are managed.
- When running from a terminal, build progress is now displayed on a single line that updates in-place (using ANSI escape codes).
  - Colored output will be coming soon!
- Logs from jobs are now reliable grouped below their header (e.g., `[N/M] Building Foo`)
- A new `--wfail` option causes a Lake build to fail if a job logs warnings.
- We received a lot of feedback about this update and a number of fixes and improvements are expected to be part of `v4.8.0-rc2`.

Q2 2024 OKRs

Objective		Key results
(O1)	Package management	(K1) Reservoir, and Lake support
		(K2) Cloud cache
		(K3) Lake critical fixes and TOML
(O2)	Documentation	(K1) Verso final touches
		(K2) Reference manual
		(K3) Port existing documentation to Verso
(O3)	Language frontend	(K1) Elaboration critical fixes
		(K2) Incremental tactics
		(K3) Elaboration parallelism
		(K4) Module system design
(O4)	Language backend	(K1) SMT-like automation: congruence closure, e-matching, case analysis
		(K2) BitVector solver
		(K3) Mutual structural recursion; elimination principles + equation theorem consolidation
(O5)	User interface	(K1) Visual Code Plugin: usability, critical fixes, user-friendly installation
		(K2) LSP Server: critical missing features

Q2 2024 OKRs (cont'd)

Objective		Key results
(O6)	Standard library	(K1) Roadmap
		(K2) Hashmap, Red Black Trees, Array and List theorems
		(K3) Web server fundamentals
(O7)	AI/ ML	(K1) VS Code integration
		(K2) LLMs fine-tuned for Lean
		(K3) REPL improvements (including Python package)

Q & A