

# 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.

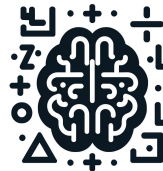
[lean-fro.org/about/roadmap](https://lean-fro.org/about/roadmap)



Formal mathematics



Software/Hardware  
verification



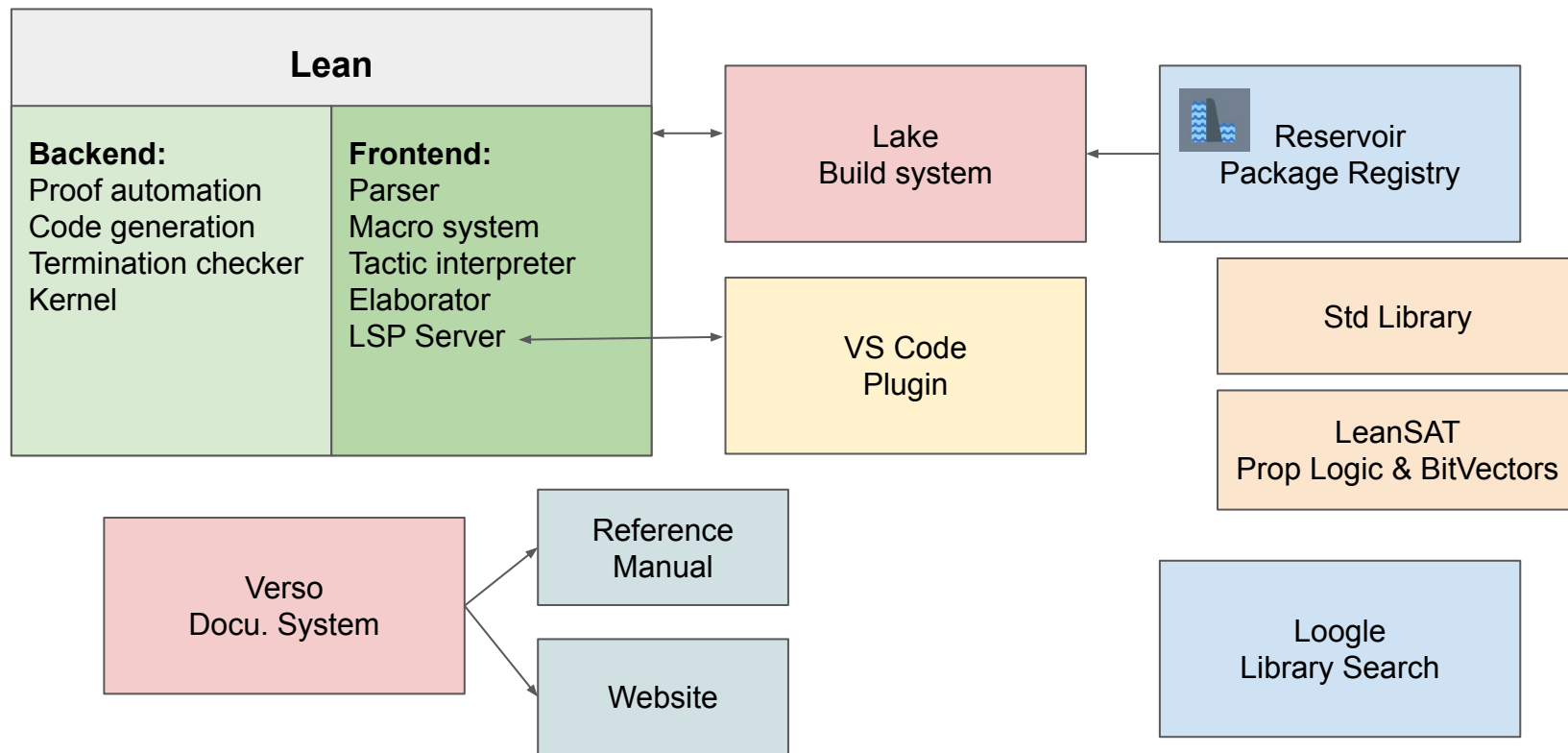
AI for math and  
code synthesis



Math and CS  
education

# Lean

---



# 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...](https://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 Into Verified Software Development

The image shows a dark blue background with a complex, geometric pattern of overlapping shapes in various shades of blue and purple. The text "Lean Into Verified Software Development" is prominently displayed in white, bold font in the upper left quadrant.

aws Open Source Blog

Lean Into Verified Software Development | Amazon Web Services

# Lean 4 repo statistics since the last meeting

(March 8 to May 10)

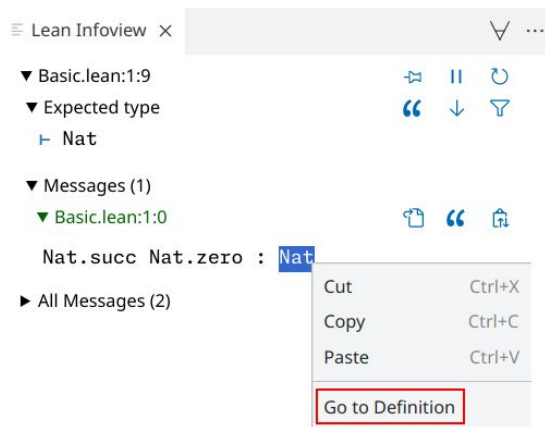
- [78 issues closed](#)
- [67 new issues](#) (48 of them were closed)
- 358 commits
- [374 PRs created](#) (331 of them were closed)

## leanprover/std4 → [leanprover-community/batteries](https://github.com/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



Restart File

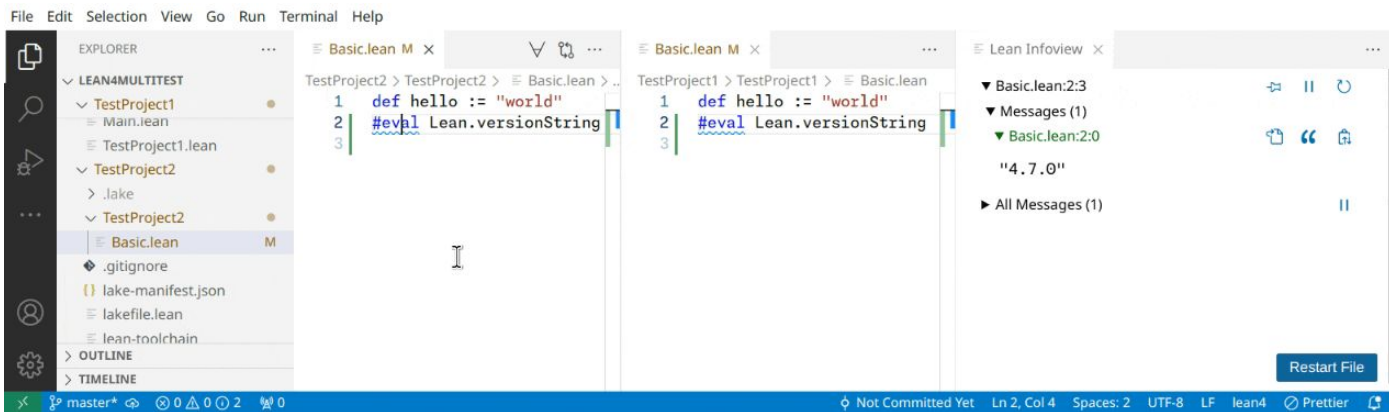


# VS Code Extension: Multi-Toolchain Workspaces

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!

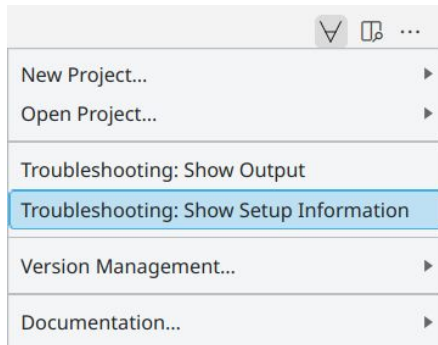
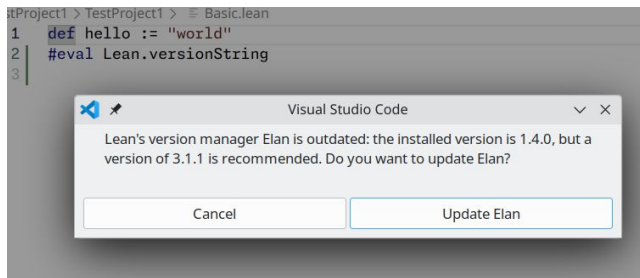


# VS Code Extension: Better Setup Diagnostics (Pre-Release)

**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

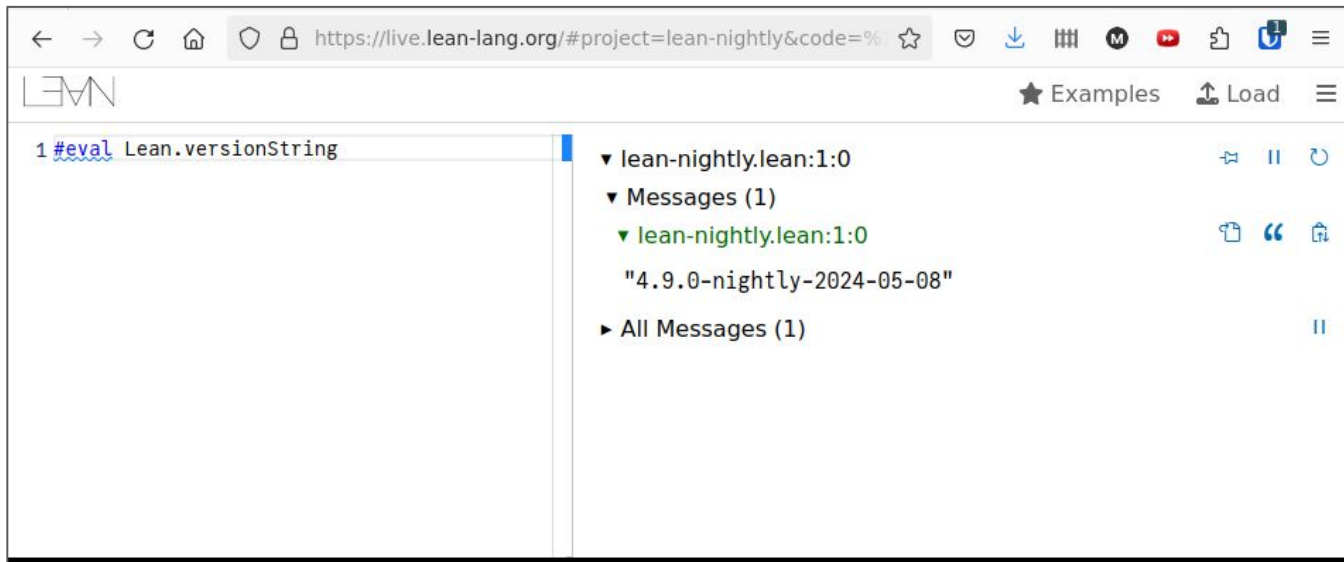
LEAN

```
1
2 example (n : Nat) : n = n := match n with
3
4
```

Lean.ParserDescr.nodeWithA... String → Lean.Synta...

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

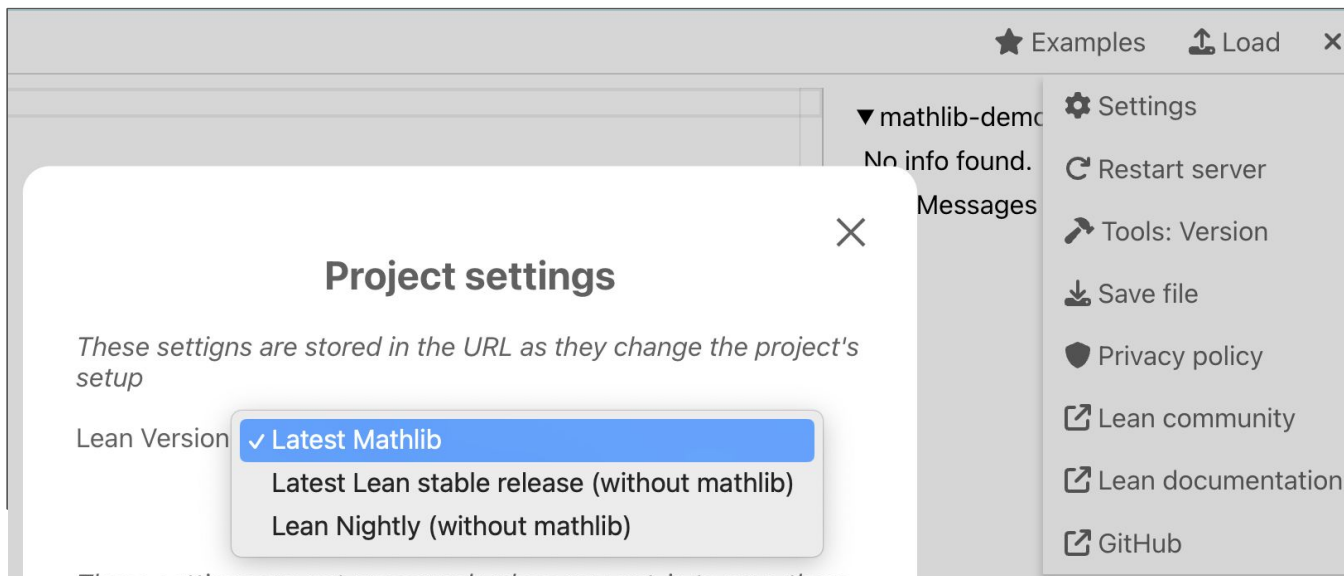
- See settings in top-right menu
- Useful when reporting bugs



The screenshot shows a web browser window with the URL `https://live.lean-lang.org/#project=lean-nightly&code=%`. The page features the LEAN logo and a top-right menu with 'Examples' and 'Load' options. The main content area displays a code editor with the line `1 #eval Lean.versionString`. To the right of the code editor, the output is shown in a tree view: `lean-nightly.lean:1:0` (with expand/collapse and refresh icons), `Messages (1)` (with expand/collapse icon), `lean-nightly.lean:1:0` (with expand/collapse, copy, quote, and share icons), and the string `"4.9.0-nightly-2024-05-08"`. Below this, there is an `All Messages (1)` section with an expand/collapse icon.

# 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)))
```

▼ tc\_issue.lean:52:0



[type\_class] max synth pending failures (maxSynthPendingDepth: 1), use `set\_option maxSynthPendingDepth <limit>`  
AddCommGroup R  
use `set\_option diagnostics.threshold <num>` to control threshold for reporting counters

▼ tc\_issue.lean:52:0



failed to synthesize  
HasQuotient (Synonym (Synonym R)) (Submodule R (Synonym (Synonym R)))  
use `set\_option diagnostics true` to get diagnostic information



## 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
```

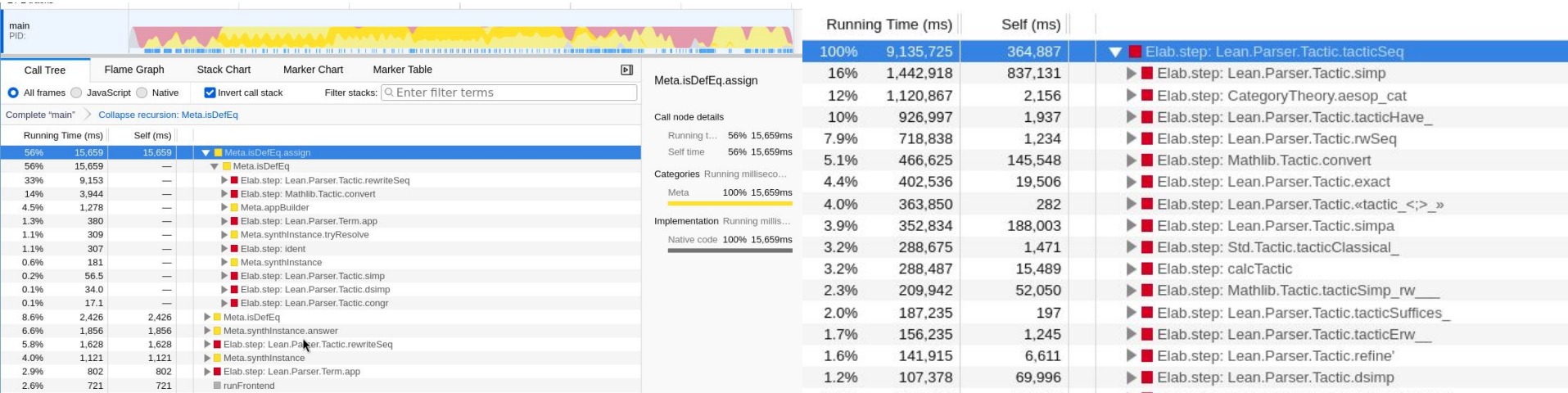
# trace.profiler.output

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

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

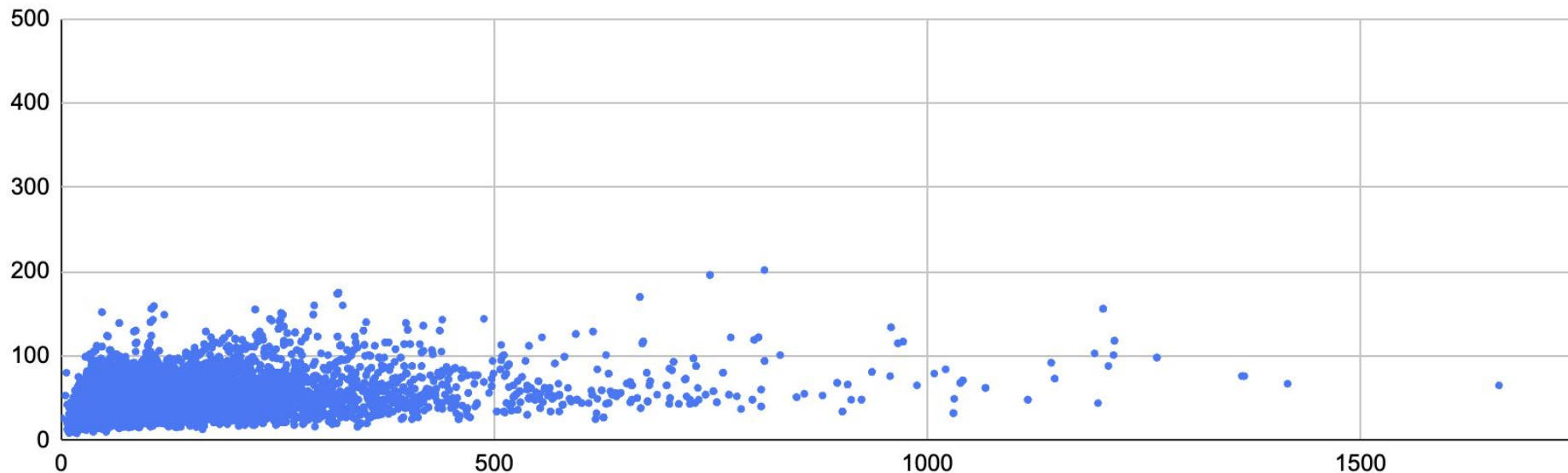
Then open `out.json` on <https://profiler.firefox.com/> (diffs supported as well).

See also `trace.profiler.useHeartbeats!`



# Shorter instances names

IsSelfAdjoint.InstContinuousFunctionalCalculusRealsSelfAdjointToStarToInvolutiveStarToAddMonoidToAddMonoidWithOneToAddGroupWithOneToRingToStarAddMonoidToNonUnitalNonAssocSemiringToNonUnitalNonAssocRingToNonAssocRingInstCommSemiringRealInstStarRingRealToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToNonUnitalNonAssocCommRingToNonUnitalCommRingCommRingMetricSpaceToTopologicalSemiringToTopologicalSpaceToUniformSpaceToPseudoMetricSpaceToNonUnitalNonAssocRingToNonUnitalNonAssocCommRingToNonUnitalCommRingToNonUnitalSeminormedCommRingToSeminormedCommRingNormedCommRingInstTopologicalRingRealToTopologicalSpaceToUniformSpacePseudoMetricSpaceToNonUnitalNonAssocRingToNonUnitalNonAssocCommRingToNonUnitalCommRingCommRingInstContinuousStarRealToTopologicalSpaceToUniformSpacePseudoMetricSpaceToStarToInvolutiveStarInstAddMonoidRealToStarAddMonoidToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToSemiringToTopologicalSemiringToNonUnitalNonAssocRingToNonUnitalNonAssocCommRingToNonUnitalCommRingToNonUnitalSeminormedCommRingToTopologicalRingToDivisionRingToNormedDivisionRingTo\_topologicalDivisionRingTo\_continuousConstSMulToSMulUniformContinuousConstSMulInstRingComplexTo\_uniformAddGroupToSeminormedAddCommGroupToNonUnitalSeminormedRingToContinuousMulToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToAddCommMonoidToNonUnitalNonAssocSemiringToNonUnitalNonAssocCommSemiringToNonUnitalNonAssocCommRingToNonUnitalCommRingToNonUnitalSeminormedCommRingToSeminormedCommRingToModuleToSeminormedAddCommGroupToNonUnitalSeminormedRingToNormedSpaceToNormedRingToTopologicalSpaceToUniformSpaceToPseudoMetricSpaceToSeminormedRingCharacterSpace



# 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

s its own scope. To implement this, Lean provides a global supply of macro scopes for each module, and one of them is distinguished as s of expansion, the current scope is replaced by a fresh one from the supply.

s contains `{LeanTerm Glued}`getCurrMacroScope``, which accesses the current macro scope, and `{LeanTerm Glued}`withFreshMacroScope``, the current scope set to a unique, fresh scope. These tools can be combined to create a Lean command that defines a top-level constant

```
eScope}
|
nTerm Glued}`name.specialEmbeddedScope` to be the desired scope along with the module's name:
alEmbeddedScope}
ion that adds this scope to a name:
beddedScope}
added by quotations, there's no particularly easy way to add the scope to an identifier (that is, a piece of syntax that represents a
tatype of syntax is essentially an encoding of arbitrary trees, which allows the same type to be used for the results of parsers even
trary new productions. The wrapper {LeanTerm Glued}`TSyntax` tracks the syntactic category of a piece of syntax at the type level,
usion of syntax and also enables the use of the coercion machinery to automate the translation from one category to another. {LeanTerm
er on the macro system, but it's described in detail in section 4.4 of [Sebastian Ullrich's thesis](https://lean-lang.org/papers/
parsed name inside identifier syntax is enriched with the embedded language's designated scope:
beddedScopeIdent}
t uses this designated scope is identical to the previous iteration, except it calls {LeanTerm Glued}`addEmbeddedScopeIdent.name` at
ge identifiers.
}
nger leak through variable capture. The embedded language term correctly fails to refer to the surrounding Lean binding of `n`:
k1}
```

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](#)

```
346 theorem Stmt.simplify_correct (h : (σ, s) ↓ σ') : (σ, s.simplify) ↓ σ' := by
347   induction h with
348   | skip => exact Bigstep.skip
349   | seq h1 h2 ih1 ih2 => exact Bigstep.seq ih1 ih2
350   | assign => apply Bigstep.assign; simp [*]
351   | whileTrue heq h1 h2 ih1 ih2 =>
352     sleep 200
353     simp_all
354     rw [← Expr.eval_simplify] at heq
355     split
356     next h => rw [h] at heq; simp at heq
357     next hnp => simp [hnp] at ih2; apply Bigstep.whileTrue heq ih1 ih2
358   | whileFalse heq =>
359     sleep 200
360     simp_all
361     split
362     next => exact Bigstep.skip
363     next => apply Bigstep.whileFalse; simp [heq]
364   | iffFalse heq h ih =>
365     sleep 200
366     simp_all
367     rw [← Expr.eval_simplify] at heq
368     split <;> simp_all
369     rw [← Expr.eval_simplify] at heq
370     apply Bigstep.iffFalse heq ih
371   | iffTrue heq h ih =>
372     sleep 200
373     simp_all
374     rw [← Expr.eval_simplify] at heq
375     split <;> simp_all
376     rw [← Expr.eval_simplify] at heq
377     apply Bigstep.iffTrue heq ih
378
```

# 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 reliably 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
(01)	Package management	(K1) Reservoir, and Lake support
		(K2) Cloud cache
		(K3) Lake critical fixes and TOML
(02)	Documentation	(K1) Verso final touches
		(K2) Reference manual
		(K3) Port existing documentation to Verso
(03)	Language frontend	(K1) Elaboration critical fixes
		(K2) Incremental tactics
		(K3) Elaboration parallelism
		(K4) Module system design
(04)	Language backend	(K1) SMT-like automation: congruence closure, e-matching, case analysis
		(K2) BitVector solver
		(K3) Mutual structural recursion; elimination principles + equation theorem consolidation
(05)	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