Lean で本を執筆する

井上亜星

Proxima Technology

2025-11-30

はじめに

自己紹介

- Proxima Technology 所属のソフトウェアエンジニア
  - 仕事では Python 等を書いている
  - 趣味で Lean を勉強している
- Lean by Example という日本語リファレンスを書いてます
  - おそらく日本語圏では最大規模
- 今年の 9 月に Lean の入門書をラムダノートから出版しました

Proxima Technology について

制御の会社で、数学科卒を積極採用する謎のスタートアップとして知られています

今回は業務として参加させていただいているので、宣伝です

今回の目的

- Lean の紹介
- Lean の書籍や Lean by Example の執筆の背景
  - どういう問題があったか
  - それをどういう風に解決したか
- みなさんも Lean の本や記事を書こう

Lean の紹介

そもそも Lean とは

- 依存型に基づく定理証明支援系
- 汎用プログラミング言語としても使うことが可能
- 既存の著名な定理証明支援系と比較すると...
  - Isabelle と違って依存型ベース
  - Agda と違ってタクティクフレームワークがある
  - Rocq (旧 Coq)と違って汎用プログラミング言語でもある

なんで Lean？

- プログラムも書けて、その証明も書けるのおもしろいよね
- do 構文が強力で、手続き的プログラムが書きやすい
- Mathlib という大規模な数学ライブラリがあって、
  たいていそれでなんとかなる
- 今年 `grind` という冗談みたいに強力なタクティクが登場。
  しかも高速。
- 手続き的プログラムについて証明を書くためのフレームワーク
  が近日登場予定。 登場したら Dafny の代替ができるかも？

内容のチェックの問題

markdown で書きたい

- 便利なので markdown で書きたい
  - web ページとして公開したければ mdbook 等が使える
  - ラムダノートでは markdown で納品する
- 当然 Lean コードはコードブロックとして書くことになる

````
```lean
def fib (n : Nat) : Nat :=
  match n with
  | 0 => 1
  | 1 => 1
  | n + 2 => fib (n + 1) + fib n
```
````

markdown で書きたい

- しかし、コードブロックに書いてしまうと Lean コードとして実行することができない
- Lean の場合毎月リリースがあるので、頻繁にバージョンを更新してみて「まだ動くかどうか」をチェックしたい
- 自動的にチェックできる仕組みがないと大変！

解決策

Lean のソースコードから markdown ファイルを生成するのはどうか? つまり左から右のような変換を行う：

```
/- コメントに地の文を書く -/

/-- ドキュメントコメント -/
def foo := "hogehoge"
```

コメントに地の文を書く

```lean
/-- ドキュメントコメント -/
def foo := "hogehoge"
```

これで、Lean のソースと markdown を自動的に同期させられる

演習問題の管理問題

演習問題の解答をどこに置くか？

- 本や記事には演習問題が必要
  - 演習問題には解答も付けておきたい
  - しかし解答は読者から隠したい
- 結果、「演習問題」と「演習問題の解答」をどう同期させるか？という問題が発生する
  - 「問題 A の解答」が「問題 A」の解答として正しく機能しているのかチェックする必要がある
  - それでいて解答は読者から隠して、本文の最後におきたい

解決策

解答ファイルから、演習問題を生成するのはどうか？ つまり左から右のような変換を行う:

```
example : 1 + 1 = 2 := by
  -- sorry
  decide
  -- sorry
```

```
example : 1 + 1 = 2 := by
  sorry
```

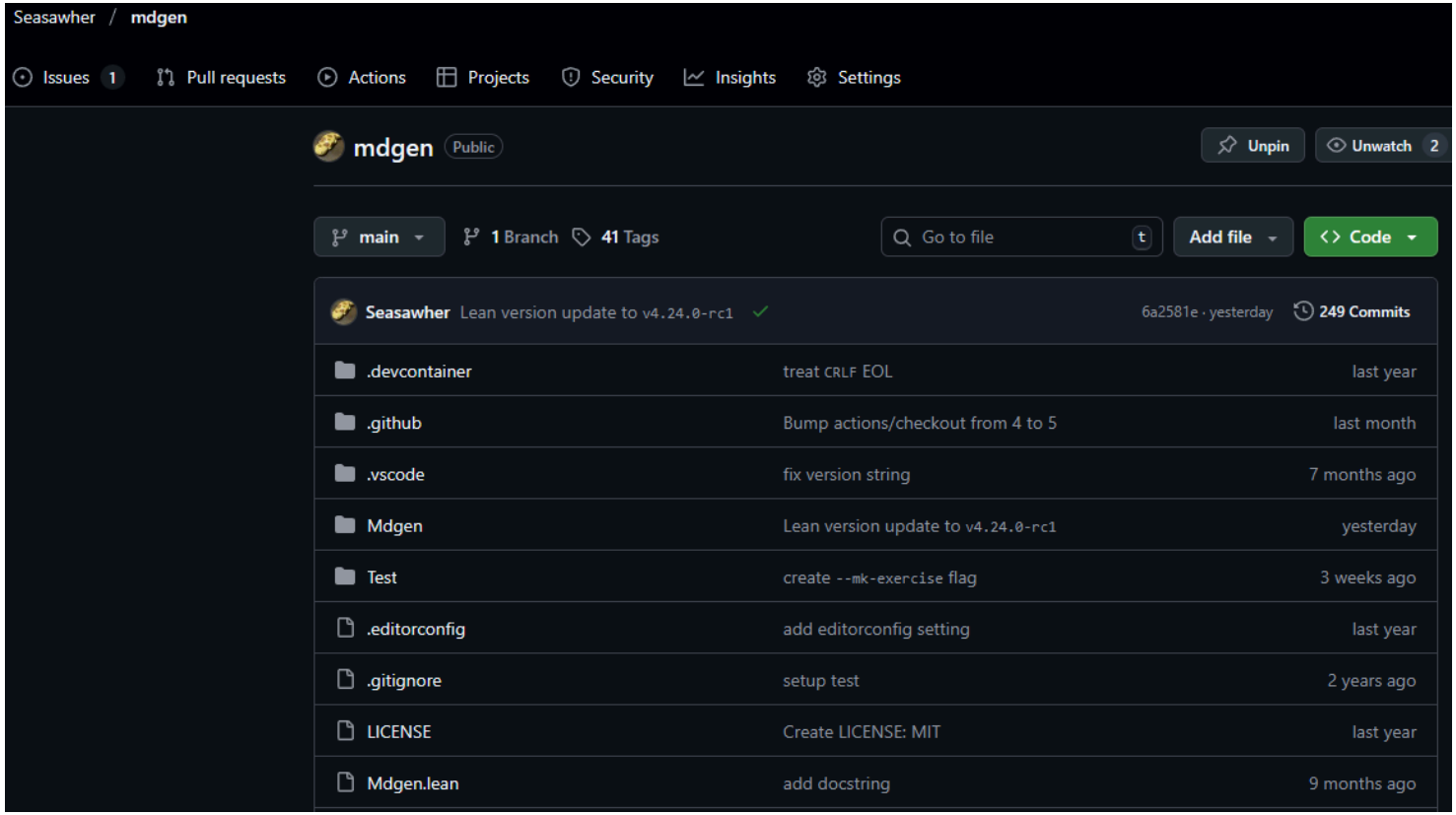
そして「sorry の部分を埋めてください」という形で演習問題を作る。

そうすれば、問題と解答が同期していることは自動的に保証されるし、解答が実行できることも自動的に保証される。

mdgen

ツールを作った

上記のようなことを実行する mdgen というツールを Lean で書き、
それを使って記事や本を書いた



裏話: 歴史的経緯

- lean2md というツールが先にあって、そのコンセプトを真似て作った
  - 「Lean ファイルを markdown に変える」という基本コンセプトはそこから借りた
  - lean2md はあまり実用的でなかったので機能追加して実用に耐えるようにした
- 演習問題を抜き出すのも、Patrick Massot さんの GlimpseOfLean から拝借したアイデア
- 私の貢献は「これは記事や本を書くのに便利だ」と気づいてツールとして完成させたこと、mdbook を Lean 用にセットアップする方法も見つけたこと

裏話: なんで Lean で書いたか❓

実際、もともとの lean2md は Python スクリプトだった。 また mdgen は依存型を一切使っておらず、他の言語でも同じものは作れるはず。

裏話: なんで Lean で書いたか？

- 「Lean はプログラムも書ける」ことを体感したかった
- プロジェクトに依存関係を追加しなくて済む
- 少なくとも Lean には CLI ツールを書くライブラリは既にあった
- Lean なら保守できる気がした
  - 実際、「互換性を維持しつつ新たな機能を追加する」というのを 2 年も続けられたのは Lean のおかげという側面がありそう。他の言語だったらダメだったかもしれない。

課題

- mdgen で解答から問題を生成するのはいいが、そのあと「解答から読者に見せるための解答説明を生成する」のはラムダノートの編集者にやっていただいた。つまり手動
- 書籍のコードをまとめてサポートサイトを作る作業は手動でやる必要がある

最後にひとこと

みなさんも、Lean の記事や本を書きましょう!

ご清聴ありがとうございました