大江戸Ruby会議11

本屋と AI と ベンチマークの話

笹田耕一 (ko1)

STORES, Inc.

自己紹介: 笹田耕一

Ruby インタプリタ開発

STORES, Inc. にて

VM　GC　Ractor　...

本屋

2024年から、イベント会場で本を売っています

今日の話

01

本屋の話

きっかけ・仕入れ・支出・管理・利益

02

AI の話

ASTro/koruby, rubvim, lumitrace, dopairb, rperf

03

ベンチマークの話

prperf と これからのお願い

SECTION

本屋の話

01

https://bs.atdot.net/

きっかけ

2024

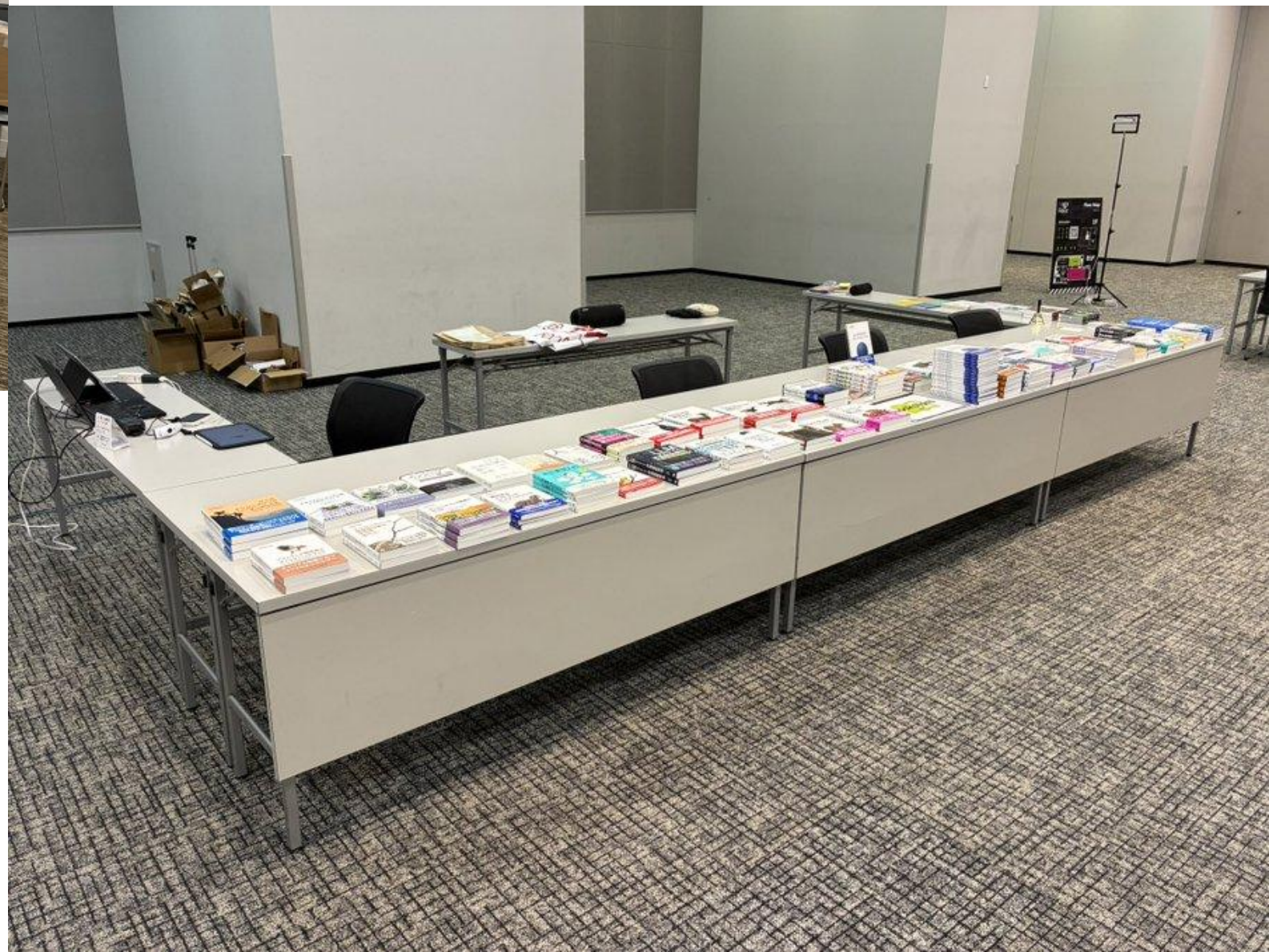
年から、なんとなく開始

「とくにきっかけはない」



Kaigi on Rails 2024

Kaigi on Rails 2026



動機

01　Ruby の普及発展

最後に本屋で Ruby の本を見たのはいつですか？

02　イベントに関わりたい

よく行くイベントを盛り上げたい
（手持無沙汰な人のために／コミュニケーション手段として）
知らないイベントに行くきっかけを作りたい

03　会社のサービスを使いたい

04　ビジネス（の真似事）をしてみたい

選書は？

最初は私の独断

- **私が好きな本**
- **気になってた本**
- **新刊**

やっぱり売れるのは新刊

イベント側の協力が欲しい

スピーカー・著者

サイン本は売れる

参加者

読みたい本を教えてほしい

本をどこから仕入れるの？

出版社にメールしてお願い

- **断られることも多い** — ぶっちゃけコネっぽい
- **70%〜90% の掛率** — 仕入値 ÷ 定価
- **返本可能（委託販売）か買い切り** — 在庫リスクを持つかどうか
- **条件は他にも色々** — 出版社ごとに違う

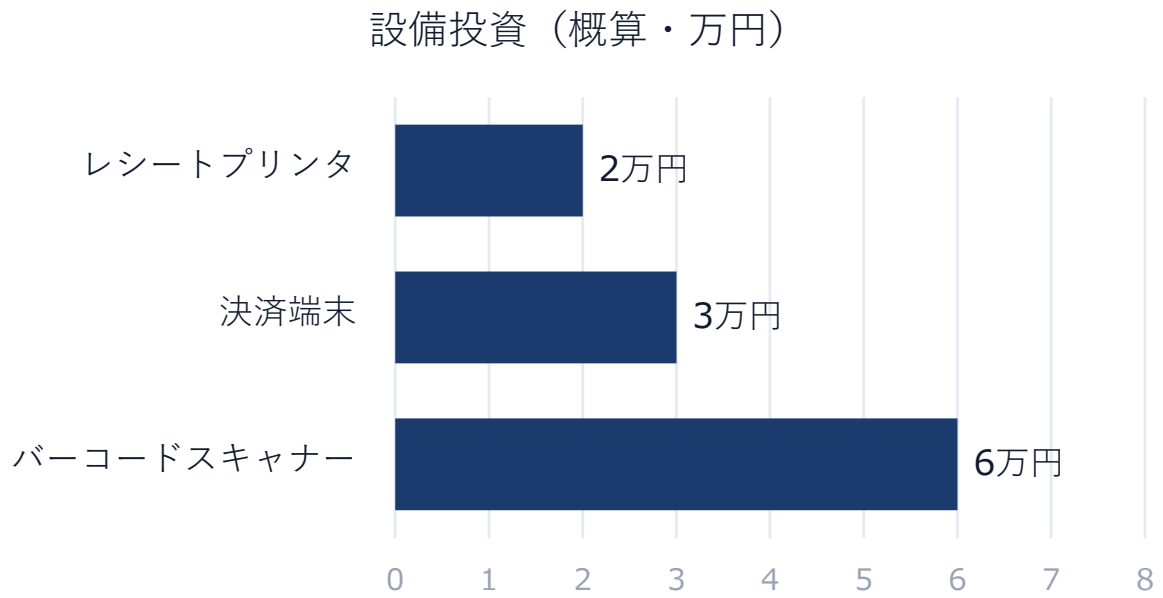
70–90%

掛率

取次

お願いするのは難しい

支出は？



決済端末は2台（最初のモデルはキャンペーンで無料）／iPad はたまたま手元にあったもの

ランニング・その他

STORES という会社が利益を……

STORES 利用料	4,000円くらい / 月
STORES 決済利用料	大雑把に 3% くらい
送料	返本など
飾り付け	

管理は？

本当に手作業

Google Spreadsheet

在庫と売上を手入力

→

コピペ

必要な行を抜き出す

→

出版社にメール

注文・返本の連絡

GAS が読めるようになった

利益は？

ビジネスとしては儲からない

33万円	15%	5万円
売上（概算）	利益率	手残り / 日

1,000人規模のイベント・1日あたり（3,000円の本が100冊＝10人に1人が1冊）

今年は 30万円くらいのプラス

イベント何本かの合計

コスト

在庫リスク（食品ほど腐らないからなんとか）
事故リスク

今後は？

暇がある限り、続けたい

誰かやってくれる人がいたら嬉しいな

SECTION

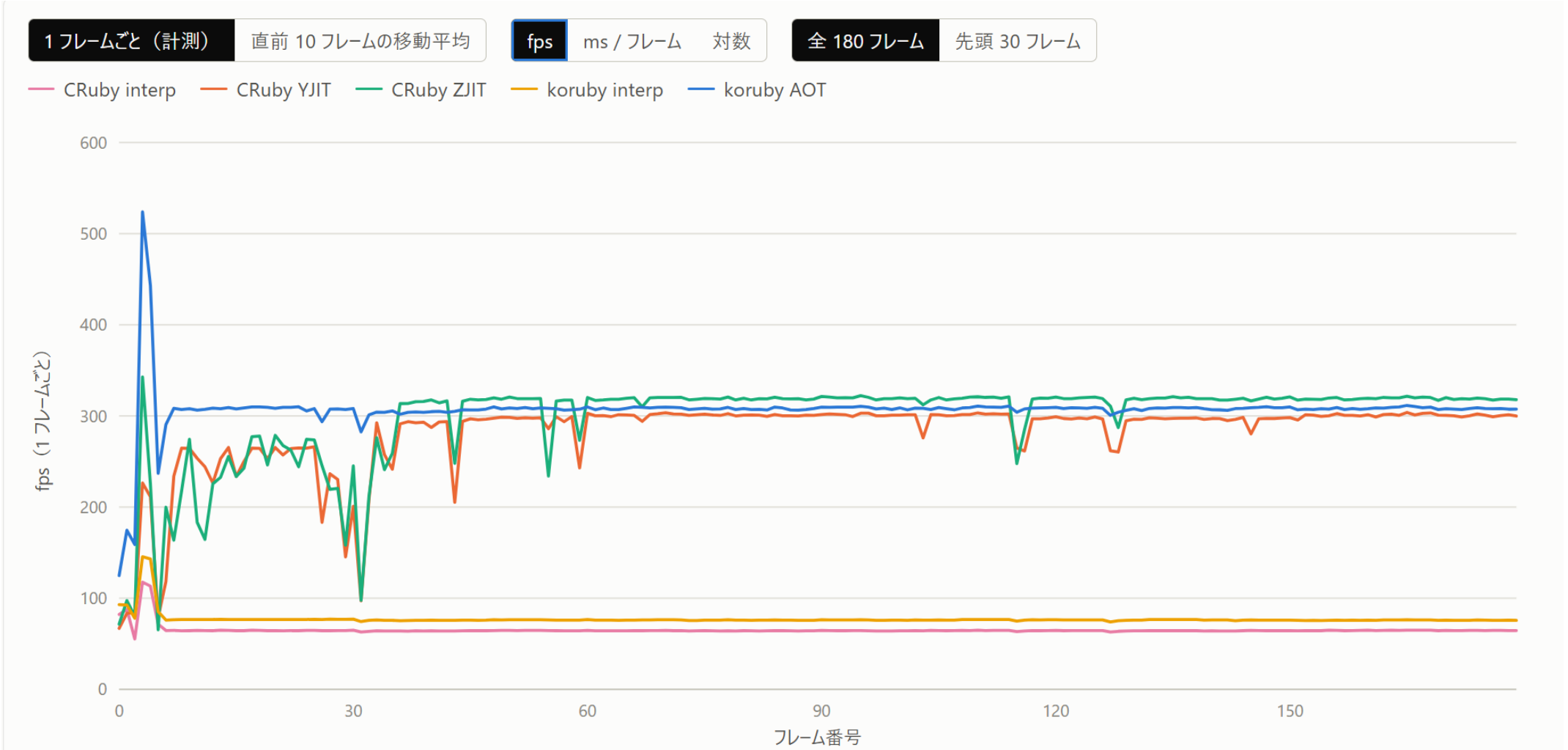
AI の話

02

作ってみたものを見てください

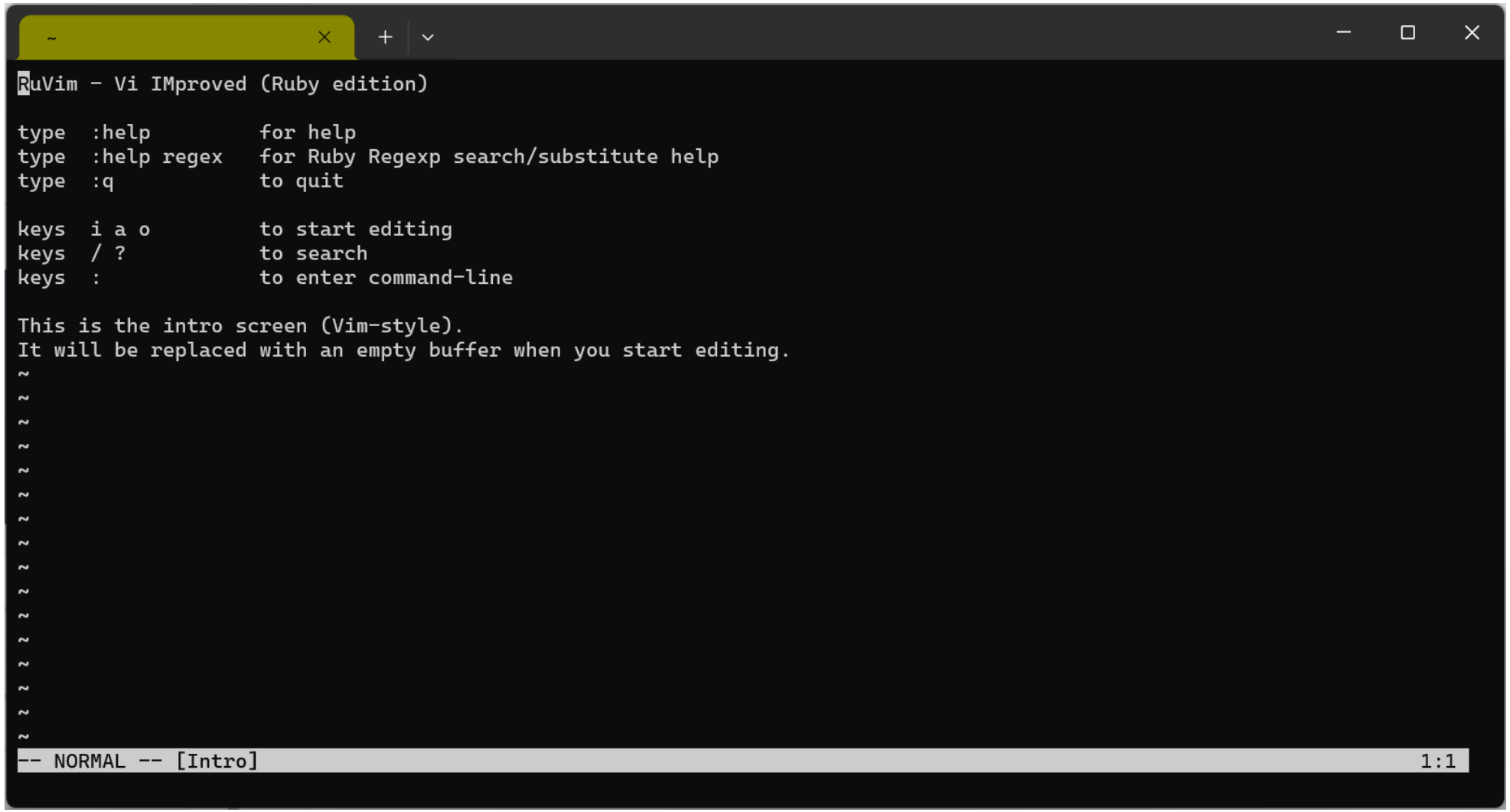
ASTro / koruby

RubyKaigi 2026 のその後
部分評価を用いた新しい Ruby 処理系



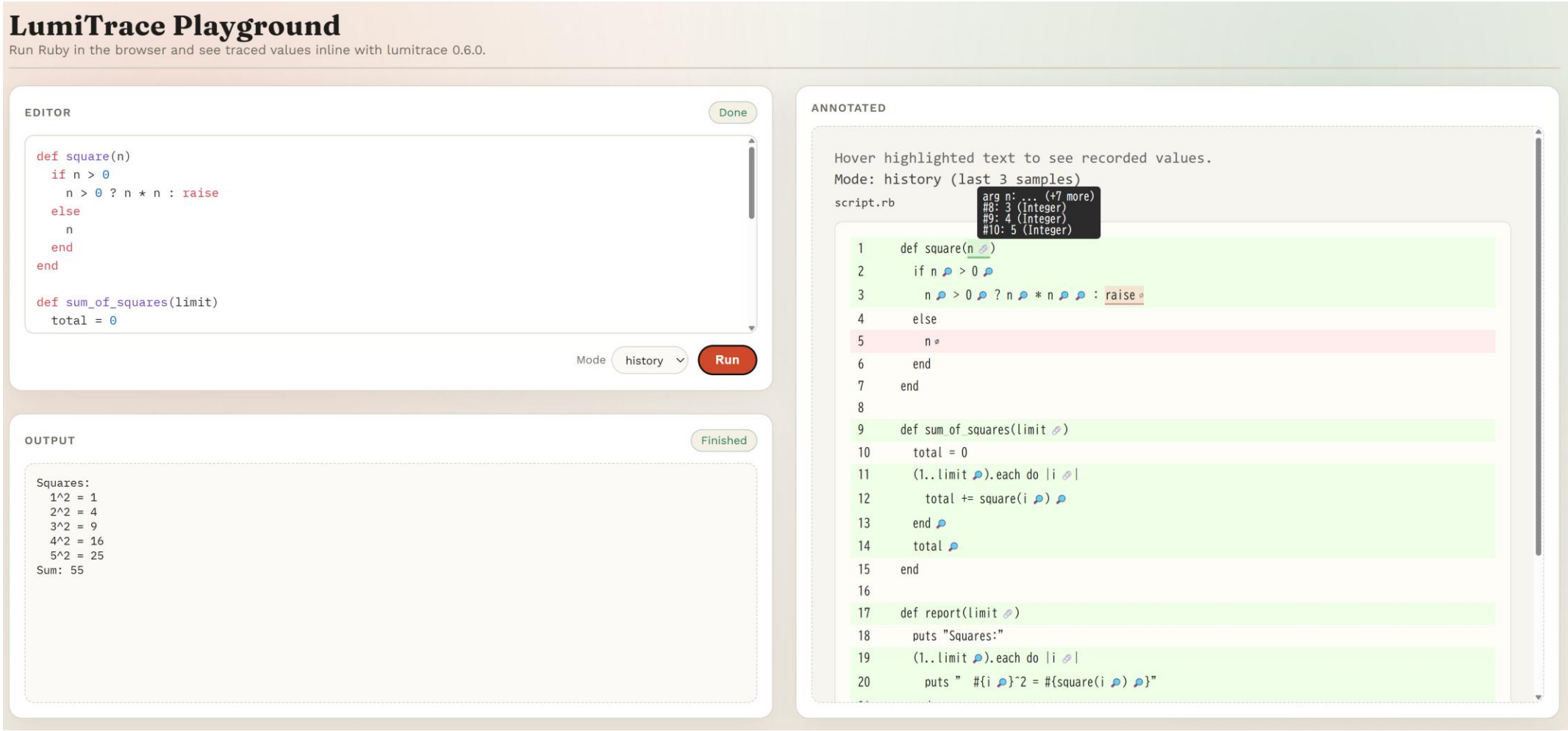
https://claude.ai/artifact/Hy4tT1emkERmmZ3wdrVhfY

rubvim



手元では vi コマンドにしてる

lumitrace



https://ko1.github.io/lumitrace/runv/

dopairb



rperf



1. セーフポイントベースのサンプリング性能プロファイラ
2. わたしが開発
   - コストがかかる部分は手書き
   - ほかは Claude Code

SECTION

ベンチマークの話

03

measure, then make it fast

prperf

`https://prperf.atdot.net/`

01

PR ごとに計測

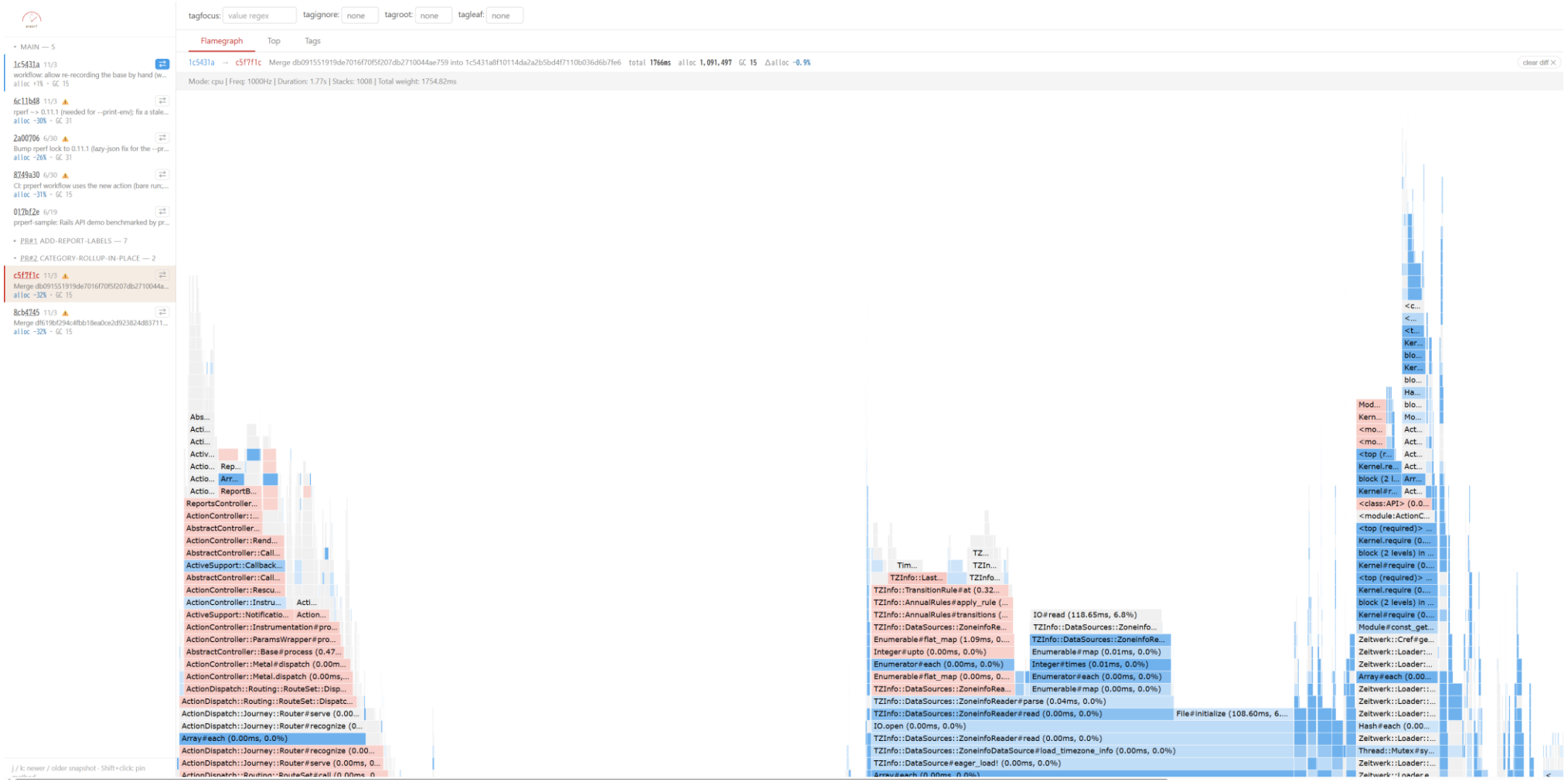
Pull Request ごとにベンチマークを測って、結果をアップロードする仕組み

02

Profile 結果 aggregator

集まったプロファイル結果をまとめて見られるようにする

prperf



https://prperf.atdot.net/view/1274261652/diff?base=1c5431a8f10114da2a2b5bd4f7110b036d6b7fe6&head=c5f7f1c5c36a194d5070e9b7a8069cdabf6a5366&bench=report

GH workflow

How you write it

One workflow, triggered on both `push` to the default branch (records the base) and `pull_request` (compared against it). You fill in only the highlighted command — the action measures it under rperf for you; everything else is copy-paste. A Bundler project also needs `gem "rperf", "~> 0.11.1"` in its `Gemfile`.

.github/workflows/prperf.yml

```yaml
name: prperf
on:
  push: { branches: [main, master] }   # records the base (default branch)
  pull_request:                        # compared against the base
jobs:
  bench:
    runs-on: ubuntu-latest
    permissions:
      contents: read
      id-token: write          # OIDC upload (no secrets)
      checks: write            # write the Check Run (public repos)
      pull-requests: write     # write the sticky comment (public repos)
    steps:
      - uses: actions/checkout@v6
      - uses: ruby/setup-ruby@v1
        with: { bundler-cache: true }
      - uses: rperf-dev/prperf-action@v1
        with:
          run: bin/rails runner ""   # ← the only line you fill in
```

真の目的: Ruby の性能改善に役立てたい

01

ベンチマーク結果が集まる

どの変更で速くなったか／遅くなったかを読む

02

ベンチマークが集まる

みなさんの「速くしたいコード」が集まる

ベンチマークそのものを送ってもらえると、いちばん嬉しい

今日のまとめ

01 本を買ってね

協力者募集

02 AI すごいね楽しいね

この資料も AI 生成

03 おらにベンチマークを分けてくれ！

prperf.atdot.net