Better Ruby

NaCl
OSS Vision
Ruby Association

まつもとゆきひろ
Yukihiro "Matz" Matsumoto
@yukihiro_matz

Rubyを**もっと**良くしたい

30年間ずっと考えてきたこと

文法・機能的にはほぼ満足

あと、欲しいのは「ネームスペース」

Namespace

独立した名前空間

バージョン依存からの解放(?)

しかし、実現できていないのには理由が

いまだ検討中

長期的な課題

それとは別にやりたいこと

言語を変えない変更

- パフォーマンス
- ツール

パフォーマンス

高速化

理論的限界よりはいつも遅い

速くて文句を言う人はいない

Ruby3x3

Ruby3.0はRuby2.0の3倍高速

(ある種のベンチマークで)

JITコンパイラによって実現

ただし、Railsアプリを除く

最初の一歩だが期待以下

理論的限界よりはいつも遅い

速くて文句を言う人はいない

RUby3.0以降もさらなる改善

YJIT

日々高速化

Ruby3.3ではRailsアプリも20%以上高速化

20%高速化≒20%コスト削減

Railsも正式採用(デフォルトでYJITオン)

素晴らしい

その他の改善

メモリ管理の改善

VWA
Variable Width Allocation

メモリ割当て効率化

GCの改善

日々高速化

素晴らしい

しかし、安泰ではない

JavaScriptの速度

PythonのJIT採用

「現状維持には全力疾走」

今後も継続して改善する

ツール

現代の高生産性は言語から来ない

開発環境

統合開発環境

VSCode

- 昔
  - 良い言語
  - 良いエディタ(Emacs, Vi)

- 今
  - そこそこの言語
  - 良い開発環境
  - Language Server Protocol

- 補完
- 動的エラーチェック
- リファクタリング支援

ツールの重要性向上

Prism / parse.y

Ruby3.3 / 大構文解析時代

ステッカー

開発支援ツールと構文解析

- RuboCop
- ruby-lsp
- Steep / Sorbet

構文解析が必要

- メンテナンス性
- 独立性
- エラー許容性

メンテナンス性

Bisonの限界

状態付き字句解析機

PrismとparseのY

それぞれ異なるアプローチ

Prism

Prism

- 手書き再起下降構文解析
- 実は最近の流行
- 高速・省メモリ

Prism

- CRubyから独立したAPI
  - JRubyなどで採用
- エラー許容性を実装
  - RubyCopなどで採用

parse.y

Bisonの限界

- バージョン問題
- 機能問題

「Bisonを捨てれば」

新規開発構文解析機ジェネレーター

Lrama

- Bison互換
- より高機能
- エラー許容性自動生成

メリット

Lramaのメリット

- BNFで文法が書ける
- Bisonよりも記述が簡単
- 状態付き字句解析機の問題回避(予定)
- エラー許容性自動生成

よりメンテナンス性が高い(予定)

競争関係

未来はどっちだ

ニコイチ案

API(独立性)はPrism

コア(構文解析機)はparse.y

サポートツール案

Prismを本採用

parse.yで文法サポートツール

２年後くらいに判断(か？)

いずれにしてもツールを支援

良いツールが充実

- irb
- AIプログラミングツール
- などなど

より高い生産性

より良いプログラミング体験

より良いRuby

提供

- NaCl
- OSS Vision
- GitHub Sponsors
- Ruby Community

Thank you