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

2000年代、言語ルネッサンス

言語「リスタート」の流行

- Perl6 (Perl5)
- Python3000 (Python2)
- PHP6 (PHP5)
- JavaScript4 (JS3)
- Ruby2 (Ruby1.8)

Ruby2アイディア

- Selector Namespace
- Keyword Arguments
- Method Combination
- Unicode Support
- Pattern Matching
- Packages
- JIT Compiler

Ruby2の発想は良くなかった

ゼロからの再スタートは死屍累々

OSSでの成功例はRails3くらいしか知らない

大事なのは漸近的な進歩

Ruby2アイディア実装

- Refinement: part of Selector Namespace (3.0)
- Real Keyword Arguments (3.0)
- Method Combination (Module#prepend: 2.0)
- Unicode Support (1.9)
- Pattern Matching (2.7)
- JIT Compiler (2.6)

文法・機能的にはほぼ満足

- Selector Namespace
- Packages

あと、欲しいのは「ネームスペース」

Namespace

Namespace = Better Refinement + Package

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

- 効率の良い実装
- わかりやすいAPI
- 誤解のない利用例
- C拡張のリンク (最大の難関)

最近、光明が

Ruby3.4には実験レベルで導入かも

どんどん実験する

実験機能は互換性が不要

- Refinementの反省
- 他言語の調査
- ユーザー期待の把握

その他のアイディア

アノテーション

C#の「属性」のようなもの

指定することでメタ情報を付加

一種のマクロとしても使える

記法が未定

既存言語の記法はRuby文法と衝突

記号が足りない

メソッド取り出し記法

「obj.method(:foobar)」の代わり

以前は「obj.:foobar」が候補

今いち納得できなかった

アイディア募集中

- ネームスペース
- アノテーション
- メソッド取得記法

これらが揃えばRuby4.0(かも?)

2028年、Ruby35周年とか？

2025年(Matz60歳)を目指していたのだが

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

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.y

それぞれ異なるアプローチ

3.4でPrismがデフォルト(か？)

いずれにしてもツールを支援

良いツールの充実を期待

- 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

提供

- Shopify
- JetBrains

Thank you