AI時代のOSS開発

NaCl
OSS Vision
Ruby Association

Yukihiro "Matz" Matsumoto
@yukihiro_matz

2025

2023

2024

2025

Gemini CLI / Claude Code / Codex

ゲームチェンジャー

Claude Opus 4.5

(さらなる)ゲームチェンジャー

Karatsuba

mruby

RSSリーダー

Ruby AOTコンパイラ

Emacsを封印

縛りプレイ

ソフトウェア開発の重心移動

Ruby

初期

構想・設計からコーディングまで

一人しかいないから

後期

コミッターが参加

自分より賢い

Rubyの方向性を決める

プロダクトオーナー

Rubyの未来を考える

ビジョナリー

Too Much To Do

I have many things I want to do

Many, many things

- mruby GC improvements
- bigint optimization
- Ruby Compiler
- Language design proposals
- ...

And more

Every week, more ideas

Every week, more issues

Every week, more proposals

The list grows

My time doesn't

Neither does my ability

Ideas pile up

Most never get done

This is frustrating

You probably know the feeling

AI Changed Things

AI getting better and better

I got better at using AI

Last year I said:

They (AI) are not your Master

I meant it

But they are not your slave either

They are something new

A partner, sometimes

A tool, sometimes

You still lead

You still decide

But now, things get done

What I Built

Let me show you

mruby Set class

A small thing

Implemented in C

(I didn't write the code myself)

It works

Karatsuba

For mruby bigint

Multiplication, faster than schoolbook

I tried this in September 2024

With ChatGPT

It didn't work

I gave up

Tried again in January 2026

With Claude Code

Same problem. Different time.

This time: Toom-3 + Karatsuba

Success

Why now?

AI got better

I got better at using AI

RSS Reader

I wanted a good RSS reader

I couldn't find one I liked

So I made my own

In October 2025

It works

I use it every day

Regexp for mruby

Mostly DFA

ReDOS aware

Small ideas, done

What about the big one?

The dream I had for years

Meet Spinel

Ruby AOT Compiler

Why AOT?

- Fast startup
- Small binary
- Single file deploy
- Embedded systems

JIT is great

YJIT, ZJIT — amazing work

But JIT is not AOT

Different tools for different jobs

mruby is small

But it need to carry whole runtime system

What if we could compile Ruby to the native code

For many years, this was just a dream

Why a dream, not a project?

Writing a Ruby compiler is hard

Because Ruby is dynamic

Simple compilation requires full runtime

Not Fast, Not Small

Requires Type Analysis

But We (I) hate Type Declaratons

TypeProf is one attempt

AOT had been too big for me

Alone, at least

But I was not alone anymore

March 2026

I made **Spinel**

How it works

- Input: Ruby source
- Output: C source
- Then: native binary

No Ruby at runtime

Just libc + libm

How?

- Parse with Prism
- Whole-program type inference
- Generate optimized C
- Compile with cc -O2

Written in Ruby

Self-hosting

spinel compiles spinel

spinel compiled by spinel compiles spinel

History

Version 1: C, 18,000 lines

In 2 weeks

Version 2: Ruby

In 2 days

Version 3: Self-hosting Ruby subset

Worked In a day; Improved since

Three rewrites in a month

AI made rewriting cheap

Supported Features

- Classes, inheritance, mixin
- Blocks, yield, lambda
- Exceptions
- Pattern matching
- Fiber
- Regexp (built-in engine)
- Bigint (arbitrary precision)

Optimizations

- Whole-program type inference
- Value-type promotion
- Constant propagation
- Loop-invariant hoisting
- Method inlining
- String concat flattening
- Bigint auto-promotion

Value-type promotion

Small classes become stack structs

1 million allocations

85 ms → 2 ms

Some programs emit no GC at all

Let's see the numbers

Benchmarks

vs CRuby
(Ruby 4.1.0dev, latest CRuby; no JIT)

Computation

- life: 86.7x
- ackermann: 74.8x
- mandelbrot: 58.1x
- fib: 34.2x
- nqueens: 30.4x
- tarai: 28.8x

Data structures

- rbtree: 22.6x
- splay tree: 13.9x
- huffman: 9.8x
- so_lists: 5.4x
- binary_trees: 3.6x
- gcbench: 2.0x

Real-world programs

- json_parse: 10.1x
- bigint_fib: 8.0x
- ao_render: 8.0x
- template engine: 6.2x
- csv_process: 3.7x
- io_wordcount: 2.9x

Geometric mean

~11.6x

Across 28 benchmarks

vs the fastest CRuby available

Numbers are nice

But let's see it run

Live Demo

Demo 1: fib(40)

Classic recursive Fibonacci

CRuby vs Spinel

Demo 2: Mandelbrot

Drawn in your terminal (Sixel)

Same code. Same image.

But wait...

YJIT is fast

YJIT is great work

AOT and JIT are different tools

JIT: long-running programs

AOT: startup, size, embedded

Complementary, not competing

Use Cases

- CLI tools
- Serverless functions
- Embedded systems
- Single-file distribution

Limitations

Not all of Ruby

AOT has trade-offs

No eval

- eval
- instance_eval
- class_eval

No dynamic metaprogramming

- send
- method_missing
- define_method (dynamic)

No Thread, No Mutex

(Fiber works)

No encoding

UTF-8 / ASCII only

No require (yet)

(require_relative works, resolved at compile time)

Not all Ruby

But useful Ruby

Status

Experimental

But working

74 tests pass

55 benchmarks pass

Open Source

MIT License

github.com/matz/spinel

★ Welcome!

Come try it

Come break it

Come help

Thank you!

One More Thing...

I have a cat

Meet Suppi

Where does "Suppi" come from?

Spinel Sun
スピネル・サン

Ruby Moon
ルビー・ムーン

Ruby and Spinel

It is Spinel,

Not a Ruby

My compiler

Takes Ruby

Produces native code

Looks like Ruby. Runs different.

That's why

Spinel

Thank you!

Sponsored by
NaCl

Sponsored by
OSS Vision

Sponsored by
GitHub Sponsors

- Buildkite
- JetBrains
- Shopify

Sponsored by
Ruby Community