Ruby after 25 years

Heroku
Ruby Association

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

25th Anniversary of Ruby

I created Ruby

Just for fun

Background

1993-02-24

The day I named it Ruby

Early 1990s

Bubble Economy crashed in Japan

My project was canceled

I assigned as a maintainer, not a developer

I had a time, a computer, and passion

Programming Languages

I wanted to create my own language

It was my high-school dream

A Problem

No Internet

10 years

I wanted a language like Perl but better

Perl+C+Smalltalk+Lisp

Combination of my favorite languages

Total Number of Users: 1

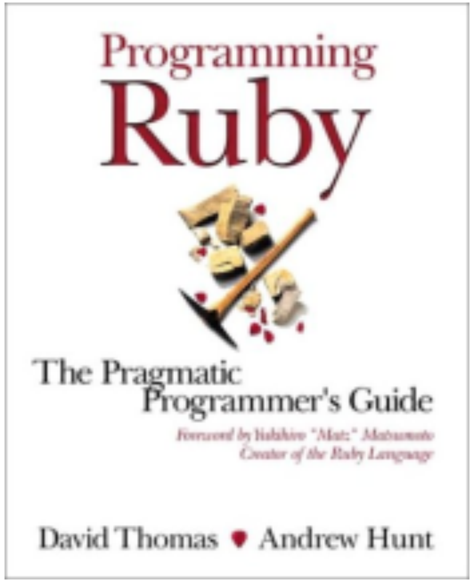
1995

I put it on the Internet

Total Number of Users: 200

2000

The Pickaxe Book



Programming Ruby

The Pragmatic Programmer's Guide

Foreword by Yukihiro "Matz" Matsumoto
Creator of the Ruby Language

David Thomas ♦ Andrew Hunt

Total Number of Users: 10000?

2001

The first RubyConf

In Tampa, Florida

Attendees: 30+

2017

RubyConf in New Orleans (USA)

Attendees: 700+

2018

Total Number of Users: 1M+?

1 to 1M+ User(s)

Ruby became Popular

Ruby is used everywhere

From Small to Big

Numerous applications written in Ruby

Ruby Should be Stable

Compatibility Matters

Bigger Community

Even The Slightest Change May Crash Code

Every Change Can Be Very Bad
(in some aspect)

Imagine your app crashes with upgrade

You would have some mixed feeling

You will be upset

Compatibility Is Important

At The Same Time,

OSS Community is a weird thing

Community

Can't expect strong loyalty

We need to attract community

Otherwise community members
will go away

We will eventually fade away

OSS Community is like a shark

We Have To Keep Moving Forward

OSS Cannot Stop

Otherwise it will die

Contradiction

Keeping Compatibility

Keeping Progressing

We Cannot Stop

We Have To Do Something To Survive

What Shall We Do?

Rebuilding the language?

- Perl6
- PHP6
- JavaScript (ES4)
- Python3
- Ruby1.9

Keeping the compatibility forever?

- FORTRAN
- COBOL

We Seek Something In Between

Ruby is not the fastest language

Ruby is not the most powerful language

We use Ruby because
Ruby is good enough

We use Ruby because
Ruby is comfortable

We use Ruby because
Ruby is nice

Human centric design

Ruby is a nice language

Being nice could bring power

Be nice

Consider Users

Talk With Potential Users

Our Prediction

- Compatibility
- Performance
- Productivity

Ruby Needs Compatibility

Compatibility

Compatibility is Important

Not because we are conservative

Not because we are afraid of changes

But because it makes us progress

Without compatibility, community keep using old versions

Don't pursue beauty of the software

It's a selfish trap for designers

- Ruby1.9
- Python3
- PHP6
- ECMAScript4

Ruby Needs Performance

No one complains faster Ruby

Ruby3x3

Ruby3 will be 3x Faster than Ruby2.0

GC

- Generational GC
- Incremental GC
- Sleepy GC
- Transitional Heaps

JIT

Just-In-Time Compiler

MJIT

Ruby2.6

Don't Expect Too Much

MIR

Other Improvements

Concurrency

Single Core May Not Become Faster

Multi Cores

Parallelism

Better Concurrency Abstraction

Guilds (or Whatever)

Utilize More Cores

Performance Solves Many Problems

- JRuby
- TruffleRuby

Besides Performance,

We Need Productivity

Types

Typing is Hot

Especially Static Typing

Languages of 2010s

- Swift
- Go
- Rust
- Crystal

All Statically Typed

- Scala
- Haskell
- Clojure (*)

These Are Statically Typed Too

Should We Introduce Static Types?

As Other Languages Do?

- Python
- JavaScript (TypeScript)
- PHP

The Answer is NO!

For Some Reasons:

It's Not DRY

Don't-Repeat-Yourself

Avoid Redundant

When Program Works Without Them

We Can Remove Them

Static Typing Has Its Benefits

- Performance
- Documentation
- Early (Static) Error Check

I Admit Dynamic Typing Lacks Them

But What If We Could Have Them?

Performance

V8

Pure JavaScript Does Not Have Static Types

V8 is the fastest VM

In Theory, VM Can Be Fast Without Static Typing

Early (Static) Type Checking

We Can Tell Types

Without Type Annotations

Type Inference

Runtime Type Aggregation

A.I.

Abstract Interpretation

Experimental Projects

- Steep
- mruby-mjit
- Sorbet

Documentation

Programming will be more conversational

IDE

Interactive Programming

Language Server Protocol

Future

Dynamic vs Static

Cycles Goes On

Foresee 10 Years Later

Type Annotation May Become Obsolete

I Believe It Will

I Will Regret Adding Type Annotation

So I Won't Add It

To Make The Better Future

Making The Better Future

Our Goal

No One Can See The Future

We Just Predict

So One Must Make His Own Steps

According To His Prediction

The Best Way to Predict The Future Is To Invent It

So We Are Creating The Future

We have a lot of things to be done

But I am sure they will benefit the community

Remember, We Try To Be Nice

We Are Trying Make Ruby Nicer

That Means More Powerful Ruby

More Comfortable Ruby

And Faster Ruby

Being Nice Is Our Way To Create The Future

Obviously Ruby Community Is Not Just Ruby

Gems Can Be Nicer

Your Apps Can Be Nicer

That Means:

We Need Your Cooperation
To Create The Future

Summary

Be Nice

Be Nice Each Other

We Can Be Stronger By Being Nice

Being nice includes:

- Moving Forward to Survive
- Keeping Compatibility
- Gradual Changes
- Providing Benefits to Users

Also includes:

- Form the community
- Help each other
- Share your effort (as OSS)

Let's Move Forward To Survive

Thank you