Rails症候群の研究

A Study on Rails Syndrome

前田 修吾

ネットワーク応用通信研究所

2008年6月22日

自己紹介(Self-intro)

名前(Name)

前田 修吾 (Shugo Maeda)

所属(Organization)

ネットワーク応用通信研究所(Network Applied Communication Laboratory)

Rubyアソシエーション(Ruby Association)

本日のテーマ(Theme)

- Rails症候群
  - Rails Syndrome

Rails症候群(Syndrome)

- Rubyに対する誤解
  - Misunderstanding against Ruby
- RailsからRubyを学んだ人々に多く見られる
  - Most common among people who learned Ruby from Rails

症例1(Case #1)

- やたらとprotectedを使う
  - Use protected profusely

例(Example)

```
class ApplicationController < ActionController::Base
  before_filter :check_access

  protected

  # check_access should be accessible
  # from subclasses, so make it protected.
  def check_access
    ...
  end
end
```

問題(Problem)

- レシーバの明示は必要ない
  - No need for explicit receiver
- だからprivateにすべき
  - So It should be private

よい例(Good example)

```
class BinaryCodedDecimal
  def +(other)
    figure1 = self.figure
    figure2 = other.figure
    ...
  end

  protected

  attr_accessor :figure
end
```

症例2(Case #2)

- やたらと既存クラスを再定義する
  - Redifine existing classes profusely

とある新人(A newbie)

- 「このRailsのクラスを変更したんいですけど…」が口癖
  - He always says "I'd like to redefine this class in Rails..."

例(Example)

```ruby
module ActiveRecordSelectOptions
  def search_option
    find(:all).map{|i| [i.name, i.id]}
  end
end

ActiveRecord::Base.class_eval do
  extend ActiveRecordSelectOptions
end
```

症例3(Case #3)

- Rubyが何かわかっていない
  - Don't know what Ruby is

とある質問(A question)

```
Subject: Areal noob question
From: XXXXXX <xxxxx@xxxxx.com>
Date: Mon, 27 Mar 2006 02:06:00 +0900

Can Ruby run without rails?
```

病因(Pathogenesis)

- Railsのせい?
  - Rails?
- ではなく、Ruby自体
  - No, Ruby itself

protected

- protectedは名前が悪い
  - protected is bad name
- 私のせいですが…
  - It's my fault...

Visibilities in Java

	private	protected	public
from client	NG	NG	OK
from subclass	NG	OK	OK
explicit receiver	OK	OK	OK

Visibilities in Ruby

	private	protected	public
from client	NG	NG	OK
from subclass	OK	OK	OK
explicit receiver	NG	OK	OK

Open class

- open classは強力
  - Open classes are powerful
- でも強力すぎる
  - But too powerful
- とくにアプリプログラマにとっては
  - Especially for app programmers

What's Ruby?

- 以前は、Ruby == MRI(Matzの実装)
  - Ruby == MRI in the past time
- 今は複数の実装がある
  - There're multiple impls now
- じゃ、Rubyって何?
  - So, what's Ruby?

予防(Prevention)

- Rubyそのものをどうにかする
  - Do with Ruby itself

protected

- protectedの意味をprivateのように変える
  - Alter the semantics of protected methods like private methods

例(Example)

```
class BinaryCodedDecimal
  def +(other)
    figure1 = self.figure    # error
    figure2 = other.figure   # error
    ...
  end

  protected

  attr_accessor :figure
end
```

Open class

- selector namespace
- or Classbox

例(Example)

```
class String
  def jcode$length
    return gsub(/[^\Wa-zA-Z_\d]/, ' ').length
  end
end

class Foo
  using jcode
  def bar
    p "あいうえお".length => 5
  end
end
```

Considerations

- 文法(syntax)
- 探索順序(lookup order)
- local rebinding
- open classを廃止するかどうか
  - Deprecate open classes?

What's Ruby?

- Rubyのように歩き、Rubyのように鳴くものは、Rubyに違いない
  - If it walks like Ruby and quacks like Ruby, I would call it Ruby

like Ruby?

- MRIのようにってこと?
  - like MRI?
- バグも含めて?
  - including bugs?

言語仕様(Lang spec)

- 一つのRuby
  - One Ruby
- 明確な仕様が多様性を保証する
  - The explicit specification ensures diversity

Keep

- RubySpec
  - 実行可能な仕様
    - Executable specification

Problem

- まつもとさんはテストが嫌い
  - Matz hates tests
- オープンスタンダードと認められる?
  - Approved as open standard?

Try

- ドキュメントとしての仕様
  - Specification as a document

結論(Conclusion)

- Rubyはもっとよくなる
  - Ruby will get better
- たぶんあなたのおかげで
  - Maybe thanks to you