

メタプログラミングRuby

LL Planets

2011-8-20

角 征典

kdmsnr@gmail.com

角 征典 - kdmsnr



MF's bliki

[新規作成](#) [FrontPage](#) [ページ一覧](#) [検索](#) [更新履歴](#) [RSS](#) [ログイン](#)

FrontPage

ここは、[Martin Fowler's Bliki](#) の翻訳Wikiです。

Martin Fowler氏本人の許可を得て公開しています。Wikiですので、どなたでも参加可能です。ご自由にページの追加、修正、変更を行ってください。

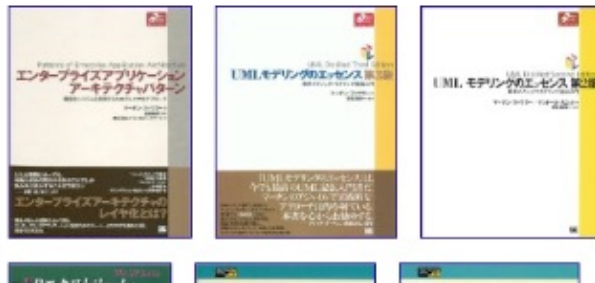
- まずは [およみください](#) をどうぞ。
- ご意見は [ご意見箱](#) までどうぞ。

まだ翻訳していないページは、[InHandOrNot](#)で確認できます。是非「新規作成」してください ;-)

13"りきし"か
Mo
—Martin

Martin Fowler's Books

- See Also [シグニチャシリーズの判断基準](#)



Links

- [Martin Fowler's Bliki](#)
- [スペイン語訳](#)
- [韓国語](#)
- [中国語](#)
- [タイ語](#)

Menu

- [お読みください](#)
- [ご意見箱](#)
- [InHandOrNot](#)

Search

Categories

- [agile](#)
- [design](#)
- [leisure](#)
- [refactoring](#)
- [thoughtWorks](#)
- [tools](#)
- [uml](#)
- [writing](#)

Access Rank 10

- [FrontPage \(274282\)](#)
- [朝会のパターン：立ってるだけじゃないよ \(65933\)](#)
- [ドメインロジックとSQL \(48477\)](#)

Rubyにおける メタプログラミングとは？

内部DSLを
書くこと
である

プログラミング言語Ruby

8章 リフレクションとメタプログラミング

- ❖ 豊富なリフレクションAPIを使い、
- ❖ メタプログラミングを実現する。
- ❖ それは、DSLを書くという発想と結び付いている。

DSLの種類

by Martin Fowler

- 外部DSL (言語外DSL)
- 内部DSL (言語内DSL)
- 言語ワークベンチ → ググって

外部DSLの例

Cucumber の Gherkin記法

フィーチャ：プロジェクトの閲覧
ユーザとして、
プロジェクトを閲覧したい。
それは、チケットを割り当てるためだ。

シナリオ：すべてのプロジェクトを一覧表示する
前提 "TextMateプロジェクト"がある
かつ "ホーム"ページを表示している
もし "TextMate"リンクをクリックする
ならば "TextMateプロジェクト"ページを表示する

内部DSLの例

tDiaryのテストコード

```
feature '日記の追記' do
  scenario '日付を指定して新しく日記を書く' do
    append_default_diary('2001-04-23')
    visit '/'
    click_link "2001年04月23日"
    within('div.day div.section') {
      page.should have content
        "とりあえず自前の環境では" +
        "ちゃんと動いているが、" +
        "きっと穴がいっぱいあるに違いない:-P"
    }
  end
end
```

DSLの再来

““ DSLに対する関心は急激に高まってきています。これは**Rubyコミュニティのおかげ**と言ってもよいでしょう。
— 『プログラミングScala』

おそらく**Rails**が発端

設定よりも規約 (CoC)

```
# usersテーブルとマッピング  
class User < ActiveRecord::Base  
end
```

DSLの目的

🔴 for プログラマ

- 生産性・理解度・表現力の向上

🔴 for プログラマ + **ドメイン専門家**


- **コミュニケーション**の円滑化

DSLにあると嬉しい

by Glenn Vanderburg

- 🔴 文脈が記述できる
- 🔴 文章のように記述できる
- 🔴 単位が記述できる
- 🔴 使える語彙が豊富である
- 🔴 階層データが記述できる

新しい**語彙**をうまく導入

 **語彙**の増加 → 新しい**視点**

- サピア=ウォーフの仮説
- 例) BDD (振る舞い駆動開発)

 「名前重要」文化

新しい**文法**は導入しない

“ 長年拒否している**機能**があります。
それはマクロ，特に**Lispスタイルのマ**
クロです。
— 『まつもとゆきひろコードの世界』

メタプログラミング技法の 分類・概要・応用例

RubyでDSLを作る方針

- 🔴 言葉を見つける
- 🔴 文章のように書いていく
- 🔴 Rubyの文法に合わせる

“ コレクションがあってシャッフルしたいと思ったら、shuffleと書いて、それで動かなきゃいけないんですよ。

— yugui

<http://www.atmarkit.co.jp/news/200907/24/ruby2.html>

Rubyの基本的な道具

- 🔴 オブジェクトと変数
- 🔴 メソッドとブロック
- 🔴 制御構文 (if, unless, while, ...)

メソッドチェーン

a.k.a. 流れるようなインタフェース

```
3.weeks.ago.saturday?
```

```
(0..9).map{|i| i + 1 }.reduce(:+)
```

```
User.where(:hobby => "Ruby").  
  order(:nickname).limit(10).offset(20)
```

カッコの省略

```
# メソッド呼び出し、引数のハッシュリテラル
establish_connection :adapter => "sqlite",
                    :database => "dbfile"

# ブロック (do ... end)
group :test do gem "shoulda" end

# if 文
if true then "\ (^o^) /" else ' / (^o^) \' end
```

オープンクラス

```
3.hours.from_now # 今から3時間後を求めたい  
  
# あらかじめ以下を定義  
class Fixnum  
  def hours  
    self * 3600  
  end  
  
  def from_now  
    Time.now + self  
  end  
end
```

クラスマクロ

単なるクラスメソッドの呼び出し

```
class Book < ActiveRecord::Base
  has_many      :releases
  belongs_to   :publisher
  validates_presence_of :title, :author
end
```

大クラス主義とMix-in

```
module ActiveRecord
  class Base
    # ...
    Base.class_eval do
      include ActiveRecord::Persistence
      extend ActiveRecord::Naming
      extend QueryCache::ClassMethods
      extend ActiveSupport::Benchmarkable
      extend ActiveSupport::DescendantsTracker

      include ActiveRecord::Conversion
      include Validations
    end
  end
end
```


クラスもオブジェクト

```
# オブジェクトIDを持っている
Fixnum.object_id # => 2156200840

# 任意のモジュールで機能拡張できる
target.extend ActiveSupport::Naming

# 変数に代入できる
fxn = Fixnum
fxn.name # => "Fixnum"
```

戻り値がクラスのメソッド

a.k.a. ミミックメソッド

```
# Camping by _why  
class Index < R "/" # <= Here  
  def get  
    render :index  
  end  
  # ...
```

演算子もメソッド

```
require 'open-uri'  
require 'hpricot'  
doc = Hpricot open "http://www.ruby-lang.org  
  
# 「/」で要素を検索している  
doc / :div.post / :h3 # => 実行結果
```

まとまったタスクの記述

Chefのレシピ

```
directory "/tmp/monkey" do
  owner "root"
  group "root"
  mode 0755
  action :create
end
```

ブロック付メソッド呼出し

```
recipe = Recipe.new(:TKG)
recipe.step "生卵に醤油を入れよくかき混ぜる。"
recipe.step "箸で茶碗の飯に適当な窪みを作る。"
recipe.step "卵を窪みに流し込み、飯と混ぜる。"

# 上のコードを意味的にまとめると
Recipe.create(:TKG) do
  step "生卵に醤油を入れよくかき混ぜる。"
  step "箸で茶碗の飯に適当な窪みを作る。"
  step "卵を窪みに流し込み、飯と混ぜる。"
end
```

ブロックの「文脈」で評価

instance_eval

```
class Recipe
  def self.create &block
    obj = self.new
    obj.instance_eval &block
    obj
  end
  # ...
end
```

eval 族

 instance_eval, class_eval

 instance_exec, class_exec

 eval

method_missing

a.k.a. ゴーストメソッド

```
require 'ostruct'  
ost = OpenStruct.new  
ost.iofjklffhdlsakf = 'めちゃくちゃなメソッド'  
  
# 呼び出し  
ost.iofjklffhdlsakf # => 実行結果
```


もっとメタプログラミング

ennnnnd

<http://redmine.ruby-lang.org/issues/5054>

```
module MyModule
  class MyClass
    def my_method
      10.times do
        if rand < 0.5
          p :small
          ennnnnd # <= Here
        end
      end
    end
  end
end
```

各種ライブラリ

📖 ParseTree (1.8 only)

📖 Rubinius (処理系), RubyParser

📖 Ripper (1.9 標準添付)

📖 Racc (パーザジェネレータ)

📖 rparsec (パーザコンビネータ)

Webフレームワークの メタプログラミング



<http://lokka.org/>

HTTPメソッドとメソッド

```
get '/admin/posts' do
  # ...
end

put '/admin/posts/:id' do |id|
  # ...
end

delete '/admin/posts/:id' do |id|
  # ...
end
```

HTTPヘッダとメソッド

```
get 'index.atom'  
  # ...  
  content_type 'application/atom+xml',  
               :charset => 'utf-8'  
  # ...  
end
```

Rubyのバージョンの調整

オープンクラス

```
unless String.public_method_defined?\  
    (:encoding)  
  class String  
    def encoding(encoding)  
      self  
    end  
  end  
end  
end
```


haml

HTML作成用の外部DSL

```
- if locale == settings.default_locale
  - page = Page('home') || Page.new
- else
  - page = Page("home-#{locale}")
.section
  .header
    %h2= page.title
  .body= page.body
```

RSpec

テスト用の内部DSL

```
describe "App" do
  context "Access pages" do
    it "should show index" do
      get '/'
      last_response.body.should \
        match('Test Site')
    end
  end
end
```

短縮メソッドの追加

特異メソッドの利用

```
def Page(id)
  Page.get_by_fuzzy_slug(id.to_s)
end
```

```
Page("home")
```

プラグインの追加

フックメソッドの利用

```
module Hoptoad
  def self.registered(app)
    app.use HoptoadNotifier::Rack
    # ...
  end
end
```

eval

```
@entry = Entry(id)
type = @entry.class.name.downcase.to_sym
eval "@#{type} = @entry"
```

肯定派？

否定派？

“**何も違わん**。違うと思うのはお主の
心の中だけじゃ。
—マスター・ヨーダ”

“ Rubyのコードは信頼できないって
？ それはコードじゃなくて、**コーダー**
を信頼してないからだろ？

— Dave Thomas

敢えて言うなら

- 🔴 DSL作るの難しい！
- 🔴 英語っぽく書けても日本人はあまり嬉しくない
- 🔴 DSLを作るならもっと厳密なチェックやエラーメッセージが欲しいかも



Paolo Perrotta (著), 角 征典 (翻訳)
<http://www.amazon.co.jp/dp/4048687158>