Ruby《を》教えてるんじゃない、 Ruby《で》教えてるんだってば

Teaching ***with*** Ruby (not "Teaching Ruby")

増原英彦

東京大学 教養学部
/ 大学院総合文化研究科

Hidehiko Masuhara

College of Arts and Sciences,
University of Tokyo

"You're teaching Ruby at Todai, aren't you?"

十月下旬、米デンバーに三百人を超えるIT（情報技
術）関係者が集まった。通称まつもとゆきひろ（４１）
をゲス　…　の「ル
…
…　る
はてな（東京　…　開発のドリ
コムなど新興企業はソフト開発の負荷が軽いル
ビーをこぞって採用する。東大でも来春から情報
科学の講座でルビーを扱う。(略)

Nikkei's article reported that
Todai picked Ruby up in
the course on information science

日本経済新聞「成長を考える: 第１部もっとできる（５）ルビーの奇跡―誰も
がITの受益者」2006/11/03より

No!

We teach with Ruby…

- …in a course on "Information Science" since 2006
  - ▸ Target:
    - all 1st and 2nd year students in College of Arts & Sciences
    - more than 400 registrants
  - ▸ Goal: fundamental concepts on computer science
  - ▸ Style: using Ruby ***as the assistance of understanding***
- ***not*** a course on Ruby programming

Talk organization

- Why Ruby?
- How we are using Ruby
- Our experience, or call-for-advice
  - ▸ Appreciation and unproven concerns
  - ▸ Pitfalls
  - ▸ Wishlist
- Summary

(CS enrollment graph)

Our observation:

We need to attract interests of non-hacker type students!

The language must be friendly to beginners

Jay Vegso, “Freshmen Interest in CS and Degree Production Trends”, Oct. 2007, http://www.cra.org/wp/index.php?cat=45

Why Ruby?

Selection criteria:

- interactive
- easy installation / pre-installation
- docs in Japanese
- simple and easy-to-understand syntax

Rivals in the shortlist:

- O’Caml
- Scheme
- Javascript
- Python
- Haskell
- なでしこ･ひまわり

Who chose Ruby?

How we use Ruby; well, you might not call it Ruby…

- Through irb
- Only top-level methods—much like an impure functional language
- Minimal syntax: if-else-end, while-end, for v in x..y-end
- Avoid using
  - ▸ method calls (e.g., x.abs)
  - ▸ blocks, classes, etc.

example: sorting

```
def sort1(a)
  for i in 0..(a.size-1)
    min_value = a[i]
    min_index = i
    for j in (i+1)..(a.size-1)
      if a[j] < min_value
        min_value = a[j]
        min_index = j
      end
    end
    a[min_index] = a[i]
    a[i] = min_value
  end
  a
end
```

How Ruby is good / not as bad as we thought

- Automatic conversion to multi-precision integer is great; e.g., for fact(100)
- Rich library helps development of tooling methods (e.g., measuring execution times for plotting a graph)
- Reasonably available; a certain number of students worked at their home
- Inefficiency (we used version 1.8)
  - ► is not an issue
  - ► even helped sometimes (e.g., comparing efficiency of different algorithms)

Pitfalls uncovered

- Disclaimer:
  - ▸ I’m not complaining
  - ▸ I know we use Ruby in a non-standard way
  - ▸ I’m not an expert of Ruby
- Please let me know better ways

Pitfalls: a never ending story

- Missing an “end” causes a disaster

```
irb(main):001:0>
  load("./sort.rb")
SyntaxError: ./sort.rb:14:
  syntax error, unexpected
  $end, expecting kEND
  from (irb):1:in `load'
  from (irb):1
irb(main):002:0>
```

```
def sort1(a)
  for i in 0..(a.size-1)
  min_value = a[i]
  min_index = i
  for j in (i+1)..(a.size-1)
  if a[j] < min_value
  min_value = a[j]
  min_index = j
  end
  a[min_index] = a[i]
  a[i] = min_value
  end
  a
end
```

line 14

Pitfalls: lost in the wild

A student: “it doesn’t work!”

```
irb(main):003:0> load("find.rb")
=> true
irb(main):004:0> find("abcde",
    "bc")
NoMethodError: undefined
    method `find' for
    main:Object
    from (irb):4
    from :0
```

```
def match_at(s, p, i)
  j = 0
  while j<p.size && s[i+j]==p[j]
    j = j + 1
  end
  j == p.size
end

def find(s, p)
  i = 0
  while !match_at(s, p, i)
    i = i + 1
  end
  i
end
```

Example of iteration

Pitfalls: Juggling with Japanese (1)

“doesn’t work for some reasons”

```
irb(main):018:0> load("./bmi.rb")
SyntaxError: ./bmi.rb:9: Invalid
    char `¥345' in expression
./bmi.rb:9: Invalid char `¥244' in
    expression
./bmi.rb:9: Invalid char `¥252' in
    expression
(略)
./bmi.rb:9: Invalid char `¥235' in
    expression
    from (irb):18:in `load'
    from (irb):18
    from :0
irb(main):019:0>
```

```
def bmi(height, weight)
  weight/height**2
end
def alarm_weight(height,weight)
  index = bmi(height,weight)
  if index >= 26.4
    "太りすぎです．”
  elsif index >= 24.0
    "太り気味です．”
  else
    "問題ありません．"
  end
end
```

全角!
(Japanese characters)

Pitfalls: Juggling with Japanese (2)

“still doesn’t …”

```
irb(main):019:0> load("./bmi.rb")
SyntaxError: ./bmi.rb:8:
    Invalid char `¥343' in
    expression
./bmi.rb:8: Invalid char
    `¥200' in expression
./bmi.rb:8: Invalid char
    `¥200' in expression
    from (irb):19:in `load'
    from (irb):19
    from :0
irb(main):020:0>
```

全角空白! (a Kanji space)

```
def bmi(height, weight)
  weight/height**2
end
def alarm_weight(height,weight)
  index = bmi(height,weight)
  if index >= 26.4
    "太りすぎです. "
 　elsif index >= 24.0
    "太り気味です. "
  else
    "問題ありません. "
  end
end
```

changed to ASCII chars.

Pitfalls: Juggling with Japanese (3)

“It gets loaded, but shows something strange”

```
irb(main):020:0> load("./bmi.rb")
=> true
irb(main):021:0> alarm_weight(1.75,60.0)
=> "¥345¥225¥217¥351¥241¥214¥343¥201¥202¥343¥202¥212¥343¥201¥276¥343¥201¥233¥343¥202¥223¥357¥274¥216"
irb(main):022:0>
```

```
def bmi(height, weight)
  weight/height**2
end
def alarm_weight(height,weight)
  index = bmi(height,weight)
  if index >= 26.4
    "太りすぎです．"
  elsif index >= 24.0
    "太り気味です．"
  else
    "問題ありません．"
  end
end
```

Pitfalls: p puts a print of pickled peppers

- Mostly avoided, but there are a few cases of printing
- Any ways to get this?

```
i=0, w=[3,nil,nil]
i=1, w=[3,8,nil]
i=2, w=[3,8,5]
```

- None of them work well
  - p("i=", i, ", w=", w)
  - print("i=", i, ", w=", w, "¥n")
  - puts("i=", i, ", w=", w)

```
def vector_add(v1, v2)
  w=Array.new(v1.size)
  for i in 0..(v1.size-1)
    w[i] = v1[i] + v2[i]
    show i and w
  end
  w
end
```

- Current workaround:
  - p(["i=", i, ", w=", w])

Pitfalls: tails of two sets of binary operators

```
irb(main):008:0> false || not(true)
SyntaxError: compile error
(irb):8: syntax error, unexpected kNOT
false || not(true)
          ^

    from (irb):8
    from :0
irb(main):009:0>
```

- should either be
  "`false || (not(true))`" or "`false || !true`"
- due to different operator precedence

A wicked wishlist

- An ubiquitous graphics library
  - ► for visualizing results
  - ► Star Ruby, Ruby/Tk, JRuby, external browser, …
- 32 bit floating point number
  - ► to experience numerical errors (cf. too many iterations required with 64 bits FP numbers)
- 32 bit integer
  - ► to demonstrate O(log n) Fibonacci algorithm which assumes O(1) arithmetic ops.
- Field declaration with a type
  - ► to demonstrate data modeling

Summary

- We teach with Ruby
  - ► not training Ruby programmers (sorry!)
- Ruby worked well in general
  - ► thanks to irb, dynamic typing, multi-precision integer support, etc.
- Uncovered pitfalls and a wishlist