Route 477

2010-01-25 Ruby基礎文法最速マスター

■ Perl基礎文法最速マスターをだいたいそのまま、Rubyに置き換えてみました。

他の言語をある程度知っている人はこれを読めばRubyの基礎をマスターしてRubyを書くことができるようになる…かも知れません。無保証です。

参考:

- オブジェクト指向スクリプト言語 Ruby リファレンスマニュアル (1.9.2)
- オブジェクト指向スクリプト言語 Ruby リファレンスマニュアル (1.8.7)

■ 1. 基礎

インタラクティブRuby (irb)

irbを使うと、Rubyのプログラムを簡単に練習することができます。

```
/Users/yhara $ irb
irb(main):001:0> puts "hello"
hello
=> nil
irb(main):002:0> 1 + 1
=> 2
```

以下の説明は、irbを起動して、自分で試しながら読むと習得が早いと思います。

表示 (print, puts, p)

```
print "foo"   #=> 改行なし
puts "foo"    #=> 改行あり
p 123         #=> デバッグ用
```

いわゆる「printfデバッグ」は、Rubyでは「pデバッグ」になります。

変数の宣言

Rubyには変数宣言はありません。また変数の型もないので、一つの変数にいろいろなクラスのオブジェクトを代入することができます。

```
a = 123
a = "abc"   #=> aを上書きする
```

Rubyでは先頭の一文字によって変数の種類が変わります。

```
foo   #=> ローカル変数
@foo  #=> インスタンス変数(@)
@@foo #=> クラス変数(@@)
$foo  #=> グローバル変数($)
FOO   #=> 定数(大文字から始まる)
```

コメント

「#」以降はコメントになります。

```
# コメント
```

スクリプトの実行

```
$ ruby foo.rb
```

出力結果をファイルに書き出すにはリダイレクトを使います。

```
$ ruby foo.rb > out.txt
```


■ 2. 数値

整数や小数が使えます。

```
num = 1
num = 1.234
num = 100_000_000
```

四則演算

```
num = 1 + 1
num = 1 - 1
num = 1 * 2
num = 5 / 2    #=> 2（整数どうしの割り算は整数になる）
num = 5.0 / 2  #=> 2.5（どちらかが小数なら結果も小数）
num = 5 % 2    #=> 1（余りをとる）
```

インクリメントとデクリメント

Rubyには++演算子がないので、+=や-=を使います。

```
i += 1
i -= 1
```


■ 3. 文字列

文字列はシングルクォートかダブルクォートで囲みます。ダブルクォートの中では\t(タブ)や\n(改行)などの特殊文字を利用することができます。またダブルクォートで囲まれた文字列の中では「#{}」を使って任意の式を展開することができます。

```
str1 = 'abc'
str2 = "def"
str3 = "a¥tbc¥n"
str4 = "#{str1} def"  #=> "abc def"
```

文字列操作

結合

```
str1 = "aaa" + "bbb"
str2 = ["aaa", "bbb", "ccc"].join(",")
```

分割

```
record = "aaa,bbb,ccc".split(/,/)
```

長さ(文字数)

```
length = "abcdef".length
length = "abcdef".size     # String#sizeとString#lengthは同じ動作。好きな方をどうぞ
```

切り出し

```
substr = "abcd"[0, 2]   #=> "ab"（0番目から2文字）
```

検索

```
idx = "abcd".index(/bc/)  #=> 1
```


■ 4. 配列

配列は「[]」を使います。

```
ary = [100, 200, 300]
```

要素の参照と代入

```
a = ary[0]   #=> 100
b = ary[1]   #=> 200
```

```
ary[0] = 1
ary[1] = 2
```

要素の個数

```
n = ary.length
n = ary.size     # Array#sizeとArray#lengthは同じ動作。好きな方をどうぞ
```

配列の操作

```
ary = [1, 2, 3]
# 先頭を取り出す
a = ary.shift   #=> aは1、aryは[2,3]に
# 先頭に追加
ary.unshift(5)  #=> aryは[5,2,3]に
# 末尾を取り出す
b = ary.pop     #=> bは3、aryは[5,2]に
# 末尾に追加
ary.push(9)     #=> aryは[5,2,9]に
```


■ 5. ハッシュ

ハッシュ(辞書)は「{}」を使います。

```
hash = {"a" => 1, "b" => 2}
```

キーには文字列よりも、シンボル(英語しか使えないが軽い文字列みたいなもの)を使うことが多いですが。

```
hash = {:a => 1, :b => 2}
```

要素の参照と代入

```
hash["a"]  #=> 1
hash["b"]  #=> 2
```

```
hash["c"] = 5
hash["d"] = 7
```

ハッシュの操作

キーの取得

```
hash.keys   #=> ["a", "b", "c", "d"]  (Ruby 1.8.xでは順不動)
```

値の取得

```
hash.values #=> [1, 2, 5, 7] (同上)
```

キーの存在確認

```
hash.key?("a")  #=> true
```

ハッシュのペアの削除

```
hash.delete("a")
```


■ 6. 制御文

Rubyでは、条件文で偽と見なされるのは**false**と**nil**のみで、それ以外のオブジェクトは 全て真と見なされます(0や空文字列も真)。

if文

```
if a == 1
  ...
elsif b == 2
  ...
else
  ...
end
```

while文 (あまり使いませんが)

```
i = 0
while i < 5
  ...
  i += 1
end
```

Array#eachとブロックを使った繰り返し

```
[1, 2, 3].each do |i|    # 「do |..| .. end」のことをブロックと呼びます
  puts i                 # Array#eachは、各要素をブロックに渡して実行してくれます
end                      # 1, 2, 3と順に表示します。

[1, 2, 3].each{|i|       # ブロックには「{|..| .. }」というちょっと短い記法もあります。
  puts i                 # 上の記法とは(結合順位が違うだけで)全く同じ動作になります。
}
```

Integer#timesとブロックを使った繰り返し

```
5.times do |i|           # 0, 1, 2, 3, 4と順に表示します
  puts i
end

5.times do               # 値を使わないときは、「|..|」の部分は書かなくて構いません
  puts "hello"
end
```

Kernel#loopとブロックを使った無限ループ

```
loop do
  puts "stop me!"        # 無限に表示し続けるので、Ctrl-Cで止めてください。
end
```

その他の繰り返し

Enumerableモジュールには、繰り返しを行う便利なメソッドがたくさん定義されています。

条件に合うものだけを選ぶ

```
[1,2,3,4,5].select{|x| x.even?}  #=> [2,4]
```

条件に合うものを除く

```
[1,2,3,4,5].reject{|x| x.even?}  #=> [1,3,5]
```

条件に合う最初のものを返す

```
[1,2,3,4,5].detect{|x| x.even?}  #=> 2
```

等しい値があるか調べる

```
[1,2,3,4,5].member?(3)           #=> true
```

条件に合うものがあるか調べる

```
[1,2,3,4,5].any?{|x| x.even?}    #=> true
```

全ての要素が条件に合うかを調べる

```
[1,2,3,4,5].all?{|x| x.even?}    #=> false
```

条件に合うものの個数を数える

```
[1,2,3,4,5].count{|x| x.even?}   #=> 2
```

最大のものを返す

```
[1,2,3,4,5].max                  #=> 5
```

最小のものを返す

```
[1,2,3,4,5].min                  #=> 1
```

加工結果が最大のものを返す

```
[1,-2,3,4,-5].max_by{|x| x.abs}  #=> -5
```

加工結果が最小のものを返す

```
[1,-2,3,4,-5].min_by{|x| x.abs}  #=> 1
```

昇順にソートする

```
[1,-2,3,4,-5].sort               #=> [-5, -2, 1, 3, 4]
```

加工結果で昇順にソートする

```
[1,-2,3,4,-5].sort_by{|x| x.abs} #=> [1, -2, 3, 4, -5]
```

加工結果を配列で返す

```
[1,-2,3,4,-5].collect{|x| x.abs} #=> [1, 2, 3, 4, 5]
```

selectにはfind_all、rejectにはdelete_if、detectにはfind、member?にはinclude?、collectにはmapという別名があります。[*1]

[*1] Rubyistはcollect派とmap派に二分されますが、筆者はmap派です。理由はタイプ数が少ないからです

■ 7. サブルーチン

Rubyには「サブルーチン」というものはありませんが、トップレベルでメソッド定義を行うことで、どこからでも呼べるメソッドを定義することができます。これはサブルーチンのように使うことができます。

```
def sum(x, y)
  return x + y
end

p sum(1,2)        #=> 3
```


■ 8. ファイル入出力

読み込み

とりあえずFile.readだけ覚えておけばなんとかなると思います。

```
str = File.read("foo.txt")  #=> strに、foo.txtの内容を丸ごと含む文字列が代入される
```

書き込み

File.openで開いて、IO#writeで書き込みます (IO#putsとかもあります)。

```
File.open("out.txt", "wb") do |f|
  f.write str
end
```


■ 知っておいた方がよい文法

Rubyの真偽値

前述のように、falseとnilが偽、それ以外の全ての値は真です。

「nil」は値がないことを示す特別な値です。例えば、配列で範囲外の要素を取ろうとするとnilが返ってきます。

```
ary = [1,2,3]
ary[100]          #=> nil
```

コマンドライン引数

ARGVという組み込み定数に入っています。

```
# ruby foo.rb a.txt b.txt としたとき：
p ARGV     #=> [a.txt, b.txt]
```

C言語と違って、ARGV[0]にプログラム名が入っていたりはしません。プログラム名は$0という変数から取得できます。

```
# ruby foo.rb a.txt b.txt としたとき：
p $0       #=> foo.rb
```

unless式

ifの逆です。

```
unless a == 1 do
  ...
end
```

後置のif (if修飾子)

Rubyではifやunlessを文の後ろに置くことができます。

```
puts "ok" if x == 1
puts "ng" unless x == 1
```

クラス定義

クラスは「class..end」で定義します。

```
class Person
  def initialize(name, age)
    @name, @age = name, age
    @address = nil
  end
  attr_reader :name, :age
  attr_accessor :address
end

jhon = Person.new("Jhon", 15)
p jhon.name
jhon.address = "USA, Earth"
```

その他のよく使うクラス

あとは Dir、Range、Time くらいですかね。

- requireなしで使えるクラス→組み込みクラス
- requireすると使えるクラス→添付ライブラリ

■ 余談

本体に入らなかった、Rubyの「仕組み」についての話です。

全てがオブジェクト

Rubyでは、全ての値が、いずれかのクラスに属するオブジェクトです。例えば数値の「1」はFixnumクラス、文字列「"foo"」はStringクラス、配列「[1,2,3]」はArrayクラスのオブジェクトです。

全てがメソッド

Rubyには「関数」のようなものはなく、printやputsでさえも**Kernelモジュール**に定義されたメソッドです。

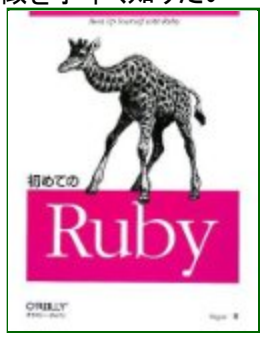
```
puts "foo"              # これは、
self.puts "foo"         # これの省略形
p self                  #=> main（トップレベルのselfは、mainという特別なオブジェクト）
p self.class            #=> Object（mainは、Objectクラスのインスタンス）
                        #          （Objectクラスは、Kernelモジュールをインクルードしている）
                        #          （なので、トップレベルでputsと書ける）
```

Kernelモジュールに定義されたメソッドは、プログラムのどこからでも呼び出せます。

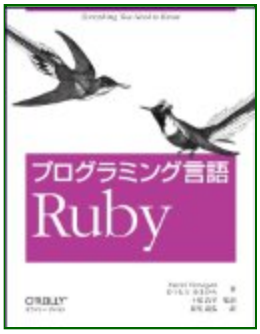
Rubyでは演算子もメソッドです。例えば「1＋2」は**Integer#+**の呼び出し、「ary[1]」は**Array#[]**の呼び出し、「ary[1]＝2」は**Array#[]=**の呼び出しです。

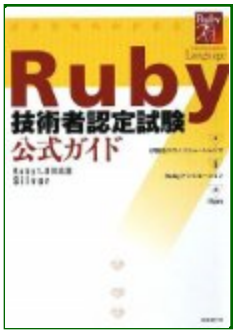
■ (おまけ)Ruby書籍紹介 ★★★★★

筆者の独断と偏見によるRuby書籍紹介です。

- 他のプログラミング言語をマスターしていて↓
  - Rubyの特徴を手早く知りたい
    - ■ 

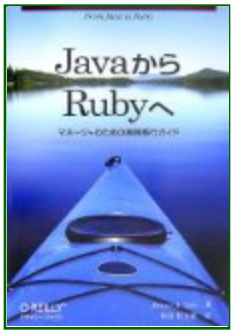
  - Rubyを網羅的に学びたい

    - 

  - 認定試験に備えたい
    - 

  - Java方面から来ました
    - Rubyとの違いを知りたい
      - 

    - JRubyに興味がある
      - 

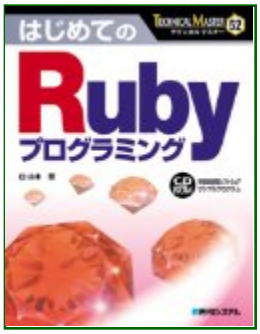
    - 上司を説得したい
      - 

- プログラミングとかあんまり慣れてない↓

    - これ一冊でOK、的な本がほしい
        - 

    - 何かを作りながら学びたい
        - 

- コンピュータにあまり慣れてない↓
    - 初歩から応用まで一冊で
        - 

    - 基礎から応用までじっくりと
        - 

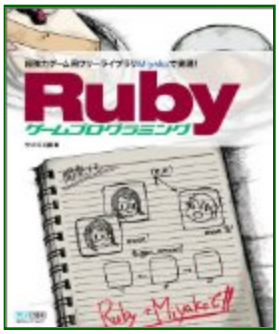
        

    - 堅苦しいのはいやなの
        - 

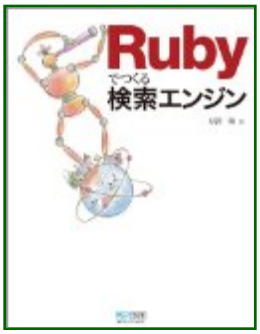
- 基礎は分かったんで↓

- どんなことができるのか知りたい
  - 

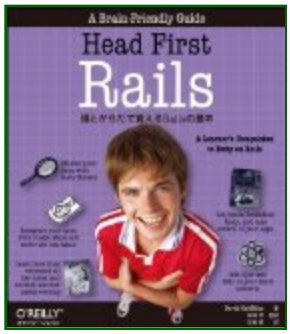
  - 

- なんか作ってみたい
  - 

  - 

  - 

- Webアプリがやりたい
  - 

  - 

- 「まつもとゆきひろ」を「ひろゆき」と間違えないようになりたい
  - 

ツッコミ・コメントがあればどうぞ! E-mailアドレスは公開されません。

お名前:

コメント:

投稿

本日のリンク元

アンテナ

- http://b.hatena.ne.jp/hotentry ×686
- http://reader.livedoor.com/reader/ ×480
- http://b.hatena.ne.jp/hotentry/it ×445
- http://b.hatena.ne.jp/entry/route477.net/d/?date=2... ×66 : 65, 1
- http://b.hatena.ne.jp/hotentry/daily ×45
- http://b.hatena.ne.jp/entrylist ×33
- http://b.hatena.ne.jp/entry.touch/route477.net/d/?... ×23
- http://b.hatena.ne.jp/entrylist/it ×21
- http://clip.livedoor.com/clip/add?link=http://rout... ×13 : 5, 3, 3, 2
- http://b.hatena.ne.jp/hotentry/20100128 ×9
- http://b.hatena.ne.jp/entrylist?sort=hot&threshold... ×8 : 4, 1, 1, 1, 1
- http://feeds.feedburner.com/hatena/b/hotentry ×7
- http://b.hatena.ne.jp/hotentry/20100126 ×7
- http://b.hatena.ne.jp/entrylist?sort=hot ×5
- http://b.hatena.ne.jp/hotentry/ ×5
- http://b.hatena.ne.jp/hotentry? ×5
- http://b.hatena.ne.jp/hotentry.touch ×5
- http://b.hatena.ne.jp/yaotti/rss ×4
- http://www.google.com/reader/view/feed/http://rssb... ×4
- http://b.hatena.ne.jp/entrylist/it?of=140 ×4
- http://b.hatena.ne.jp/hotentry/20100125 ×3
- http://b.hatena.ne.jp/entrylist?of=20 ×3
- http://b.hatena.ne.jp/hotentry/it.touch ×3
- http://www.akiyan.com/bxi/?act=entry&url=http://ro... ×3
- http://www.bloglines.com/myblogs_display?sub=89171... ×2
- http://clip.livedoor.com/rss/hot ×2
- http://b.hatena.ne.jp/entrylist/it?sort=hot&thresh... ×2
- http://b.hatena.ne.jp/entry/d.hatena.ne.jp/perlcod... ×2
- http://www.hatena.ne.jp/naoya/antenna ×2
- http://b.hatena.ne.jp/hotentry?cname=elec ×2
- http://b.hatena.ne.jp/entrylist/it?of=20 ×2
- http://b.hatena.ne.jp/entrylist/it?of=40 ×1
- http://www.gabean.com/link-sports.html ×1
- http://b.hatena.ne.jp/entrylist/it?of=240 ×1
- http://jp.visualusion.com/RssReader/ ×1
- http://feed.distantplace.org/はてなブックマーク_-_人気エントリー/ ×1
- http://www.gabean.com/link-none.html ×1
- http://www.hatena.ne.jp/nagoya313/antenna ×1
- http://b.hatena.ne.jp/entrylist?threshold=5&url=ht... ×1
- http://tweetbuzz.jp/entry/8055264/route477.net/d/?... ×1

- http://b.hatena.ne.jp/entry/ ×1
- http://reader.freerss.net/usr/ktigers/show.cgi?cmd... ×1
- http://b.hatena.ne.jp/entry/www.1x1.jp/blog/2010/0... ×1
- http://b.hatena.ne.jp/entrylist?sort=hot&of=0&thre... ×1
- http://bbeta.hatena.ne.jp/hotentry ×1
- http://dev.ctor.org/f2p/entry/inbox ×1
- http://b.hatena.ne.jp/hotentry/20100127 ×1
- http://ashitani.jp/bookmark/hotentry.rb ×1
- http://b.hatena.ne.jp/entry/d.hatena.ne.jp/nattou_... ×1
- http://rss.tmtr.dvrdns.org:8080/reader/ ×1
- http://vine/_dev/hateRssMail/debugLogDir/index.php... ×1
- http://www.hanrss.com/myfeeds_main.qst?fsrl=518571... ×1
- http://www1.nanto.ne.jp/~takakou/rssreader/feedsho... ×1
- http://www.bloglines.com/myblogs_display?folder=75... ×1
- http://twib.jp/entry/0fb3c3fec4a23f22b91bc93edd0e9... ×1
- http://b.hatena.ne.jp/entrylist?sort=hot&of=20 ×1
- http://heppokone.blog27.fc2.com/blog-entry-416.htm... ×1
- http://jp.hanrss.com/myfeeds_main.qst?fsrl=641&ssr... ×1
- http://reader.freerss.net/usr/corini/show.cgi?cmd=... ×1
- http://b.hatena.ne.jp/entry/d.hatena.ne.jp/gifnksm... ×1
- http://applepedlar.ddo.jp/RSSConciergeViewerPlusPl... ×1
- http://feeds.delicious.com/v2/rss/popular?count=15... ×1

その他のリンク元

- http://b.hatena.ne.jp/ ×709
- http://www.1x1.jp/blog/2010/01/php-basic-syntax.ht... ×477
- http://d.hatena.ne.jp/dplusplus/20100126/p1 ×245
- http://www.google.co.jp/reader/view/ ×173
- http://d.hatena.ne.jp/gifnksm/20100131/1264934942 ×151
- http://www.google.com/reader/view/ ×140
- http://twitter.com/ ×139
- http://d.hatena.ne.jp/perlcodesample/20091227/1264... ×124
- http://www.google.co.jp/reader/view/?hl=ja&tab=wy ×124
- http://www.sleipnirstart.com/ ×115
- http://route477.net/ ×111
- http://news.atode.cc/ ×93
- http://d.hatena.ne.jp/rubikitch/20100201/elispsynt... ×89
- http://www.google.co.jp/reader/view/?tab=my ×70
- http://pipes.yahoo.com/pipes/pipe.info?_id=bG0Ifxj... ×62
- http://www.google.co.jp/ ×54
- http://d.hatena.ne.jp/thinca/20100201/1265009821 ×49
- http://d.hatena.ne.jp/perlcodesample/ ×46
- http://d.hatena.ne.jp/nattou_curry_2/20100130/1264... ×42
- http://www.mixclips.org/ ×39
- http://anond.hatelabo.jp/20100129134601 ×32
- http://b.hatena.ne.jp/add?mode=confirm&title=Route... ×28
- http://www.google.co.jp/ig ×28
- http://b.hatena.ne.jp/?from=firefox ×26
- http://d.hatena.ne.jp/nattou_curry_2/20100131/1264... ×25
- http://news.atode.cc/?d=20100127 ×25
- http://fastladder.com/reader/ ×23
- http://d.hatena.ne.jp/nattou_curry_2/20100129/1264... ×22

- http://pipes.yahoo.com/pipes/pipe.info?_id=603d636... ×21
- http://twitter.com/hatebu ×20
- http://news.qooqle.jp/ ×18
- http://www.google.co.jp/ig?hl=ja ×18
- http://egone.org/ ×18
- http://d.hatena.ne.jp/tt_clown/20100202/1265096776... ×17
- http://www.google.co.jp/ig?refresh=1 ×17
- http://popurls.com/ ×17
- http://d.hatena.ne.jp/shunsuk/20100130/1264842323 ×16
- http://www.google.com/reader/view/user/-/state/com... ×16
- http://www.google.co.jp/reader/view/?hl=ja ×14
- http://www.whocares.jp/rdr.jsp?u=http://route477.n... ×14
- http://pipes.yahoo.com/pipes/pipe.info?_id=c10ff0b... ×14
- http://d.hatena.ne.jp/dplusplus/20100126 ×13
- http://pipes.yahoo.com/pipes/pipe.info?_id=5752cea... ×13
- http://www.feedly.com/home ×12
- http://bit.ly/6ztpBP ×12
- http://blog.livedoor.jp/takaaki_bb/archives/513741... ×12
- http://d.hatena.ne.jp/rubikitch/ ×11
- http://d.hatena.ne.jp/chaichanPaPa/20100127/126459... ×11
- http://news.atode.cc/?d=20100128 ×10
- http://d.hatena.ne.jp/repeatedly/20100201/12649720... ×10
- http://www.instapaper.com/u ×9
- http://www.google.co.jp/reader/view/?tab=ny ×9
- http://zapanet.info/new/hatebu4/ ×9
- http://hatebu24h.ashitano.in/ ×9
- http://www.google.com/reader/view/?hl=ja&tab=wy ×8
- http://www.google.co.jp/ig?hl=ja&source=iglk ×8
- http://b.hatena.ne.jp/keyword/ruby?sort=hot&thresh... ×8
- http://labs.ceek.jp/hbnews/ ×8
- http://delicious.com/popular/ ×8
- http://hootsuite.com/dashboard ×8
- http://bit.ly/7HTrMw ×8
- http://img.simpleapi.net/ ×8
- http://ig.gmodules.com/gadgets/ifr?view=home&url=h... ×8
- http://ig.gmodules.com/gadgets/ifr?view=home&url=h... ×8
- http://2ch-m.info/index.php?page=index.index& ×8
- http://gururin.com/ ×8
- http://www.google.com/reader/view/?tab=cy ×7
- http://ruby.designiddatabase.net/ ×7
- http://zapanet.info/new/hatebu5/ ×7
- http://wiki.onakasuita.org/pukiwiki/?基礎文法最速マスター ×7
- http://ig.gmodules.com/gadgets/ifr?view=home&url=h... ×7
- http://d.hatena.ne.jp/chaichanPaPa/20100131/126494... ×7
- http://www.google.co.jp/ig?hl=ja&refresh=1 ×6
- http://twitter.com/yugui ×6
- http://b.hatena.ne.jp/nonpori/ ×6
- http://d.hatena.ne.jp/gfnksm/ ×6
- http://ruby.designiddatabase.net/data/387407.aspx?... ×6
- http://twitter.com/home ×6
- http://d.hatena.ne.jp/ruicc/20100131/1264905896 ×6

- http://www.google.co.jp/ig?hl=ja&t=0 ×6
- http://b.hatena.ne.jp/t/ruby ×6
- http://bit.ly/8Ri0Cz ×6
- http://www.google.com/reader/view/?tab=my ×6
- http://www.google.co.jp/reader/view/?hl=ja&utm_sou... ×6
- http://ig.gmodules.com/gadgets/ifr?view=home&url=h... ×6
- http://www.rubyinside.com/ ×6
- http://bit.ly/7m1aWC ×5
- http://www.google.com/ig?refresh=1 ×5
- http://b.hatena.ne.jp/add?mode=confirm&title=Route... ×5
- http://www.google.co.jp/reader/i/?source=mog&gl=jp... ×5
- http://www.netvibes.com/ ×5
- http://friendfeed.com/ ×5
- http://www.google.co.jp/webhp?sourceid=navclient-f... ×5
- http://ig.gmodules.com/gadgets/ifr?view=home&url=h... ×5
- http://ig.gmodules.com/gadgets/ifr?view=home&url=h... ×5
- http://b.hatena.ne.jp/prepre/ ×5
- http://delicious.com/popular/ruby ×5
- http://www.jimmyr.com/ ×4
- http://www.tumblr.com/dashboard ×4
- http://zapanet.info/new/hatebu/ ×4

検索

- 基礎文法最速マスター ×26 / Ruby基礎文法最速マスター ×15 / ruby ×9 / ruby 最速 ×9 / ruby 基礎 ×9 / ruby 最速マスター ×6 / ruby 文法 ×5 / 最速マスター ×4 / ruby 基礎文法 ×4 / Rubyでサブルーチン ×4 / 文法最速マスター ×4 / Route 477 - Ruby基礎文法最速マスター ×4 / ruby 数値書き込み ×3 / http://route477.net/d/?date=20100125 ×3 / ruby 基本文法 ×3 / Route 477 ×3 / (1 .. 4) ruby ×2 / ruby 書籍 ×2 / ruby 学び方 ×2 / Ruby each do 数字 ×2 / route 477 ×2 / Ruby Debug文 ×2 / ruby find detect ×2 / ruby 最速文法 ×2 / 基礎文法最速マスターシリーズ ×2 / route477 ×1 / ruby基本文法 ×1 / Ruby 基礎 ×1 / java 基礎文法 最速マスタ ×1 / ruby 基礎文法最速マスター ×1 / ruby if 後置修飾子 ×1 / ruby ハッシュ each 代入 ファイル ×1 / 基礎文法最速 ×1 / 文字列の中 ruby 変数 ×1 / irb ×1 / ruby 文法最速 ×1 / ruby基礎文法最速マスター ×1 / Ruby 基礎文法最速 ×1 / 最速 ruby ×1 / (A+1)&(-2) ×1 / ruby route477 ×1 / ruby クラス 文法 ×1 / Ruby基礎文法 ×1 / Ruby 基礎文法最速マスター ×1 / ruby 文字列 先頭の１文字を削除 ×1 / ruby ハッシュ キー 取得 ×1 / ruby 基礎文法最速 ×1 / "基礎文法最速マスター" ×1 / ruby 基礎文法 最速マスター ×1 / ruby 変数 数値 ×1 / 基礎文法 (?) 最速マスター ×1 / Route 477 - Ruby基礎文法最速マスタ ×1 / 最速 基礎 ×1 / 最速マスター ruby ×1 / Ruby 最速 ×1 / ruby サブルーチン ×1 / ruby 基礎 | ×1 / Ruby 基礎文法 最速 ×1 / 最速基礎文法マスター ×1