Rubyで作る奇妙なプログラミング言語

ヘンな言語のつくりかた

原 悠 [著]

Esoteric Language programming in Ruby

あなたは「+ - ><.,[]」の8つの記号しかない言語や、
空白だけで構成される言語など、奇妙な言語
（Esoteric Language）があるのを知っていますか？

本書ではこのような言語を題材に
プログラミング言語の作り方を詳しく解説しています。

マイナビ

Mynavi Advanced Library

Rubyで作る奇妙なプログラミング言語

ヘンな言語のつくりかた

原 悠 [著]

マイナビ

Mynavi Advanced Library

●本書のサポートサイト

http://book.mynavi.jp/support/pc/4953/

●本書は

「Rubyで作る奇妙なプログラミング言語」（2008年12月刊行）

を元にした電子版/オンデマンド版です。

●本書中では、電子版/オンデマンド版制作にあたり、P.6からの「本書の概要とRubyのインストール」について、見直しを行っています。

●そのほかの本書中の情報は、基本的に上記書籍執筆段階のものです。ソフトウェアのバージョン、URL、製品のスペックなどの仕様や情報は、すべてその原稿執筆時点でのものです。執筆以降に変更されている可能性がありますので、ご了承ください。

・本書に記載された内容は、情報の提供のみを目的としております。したがって、本書を用いての運用はすべてお客様自身の責任と判断において行ってください。
・本書の制作にあたっては正確な記述につとめましたが、著者や出版社のいずれも、本書の内容に関してなんらかの保証をするものではなく、内容に関するいかなる運用結果についてもいっさいの責任を負いません。あらかじめご了承ください。

・本書中の会社名や商品名は、該当する各社の商標または登録商標です。本書中では™および®マークは省略させていただいております。

はじめに

あなたは「プログラミング言語」と聞いて何を思い浮かべるだろうか。C言語、Java、PHPなどの有名どころに、Perl、Python、Rubyのようなスクリプト言語、Lisp、Haskell、OCamlのような関数型言語と、世の中には実にたくさんのプログラミング言語が存在する。

例えば、Javaで2つの整数の足し算を行うプログラムは以下のような感じになる。

```
public static add(int x, int y) {
  return x + y;
}
```

一方、Lispの一種であるSchemeではこんな風になる。

```
(define (add x y)
  (+ x y))
```

同じことを行うプログラムのはずなのに、どうしてこんなに違った見た目になるのだろうか？　いやいや、これくらいで驚いていてはいけない。本書で紹介する言語の1つ、Brainf*ckでは、2つの整数の足し算を行うプログラムは以下のようになる。

```
[>>+<<-]>[>+<-]>
```

もはやどうして足し算になるのか、まったくわからないだろう。

本書ではこのような「奇妙な言語（Esoteric Language）」の処理系を実装しながら、プログラミング言語がどのようにできているのか、その秘密に迫っていく。最後まで読み終えれば、きっと自分でも新しいプログラミング言語を作れるようになっているだろう。

「プログラミング言語」というと難解で複雑なイメージがあるかもしれないが、本書を通して少しでもプログラミング言語に親しみを覚えてもらえれば幸いだ。

2008年11月　原　悠

CONTENTS

本書の概要とRubyのインストール

まずはじめに、この本の概要とRubyのインストール方法について説明する。

本書の構成

1章では、4つのEsoteric Languageの処理系を作成しつつ、プログラミング言語の秘密に迫る。

2章では、本書オリジナルの2つのEsoteric Languageを作成し、新しいプログラミング言語の作り方について説明する。

付録では練習問題の解答と、本編で紹介しきれなかったさまざまなEsoteric Languageについてまとめた。

対象読者

本書は以下のような読者を想定している。

- プログラミング言語を作ってみたいけど、何から初めていいかわからない人
- Esoteric Languageに興味がある人
- Rubyで実用的なアプリケーションを作れるようになりたい人
- Rubyの文法は覚えたけど、何を作っていいかわからない人
- プログラミング言語の未知の世界を見てみたい人

本書ではRuby言語そのものについての解説はしないので、Rubyの文法などについては適宜Ruby関係の書籍をあたってほしい。おすすめのものを以下に挙げておく。

『たのしいRuby』(ソフトバンククリエイティブ)
Rubyの定番の入門書。第四版でRuby 2.0に対応した。

『初めてのRuby』(オライリージャパン)
プログラミング経験者向け。Ruby 1.8/1.9対応。

『パーフェクトRuby』（技術評論社）
Rubyの基本から実践的知識までをカバーした中上級者向けガイド。Ruby 2.0対応。

また、Rubyの日本語リファレンスマニュアルがWebから参照可能だ。

Rubyリファレンスマニュアル
http://doc.ruby-lang.org/ja/

Rubyのバージョンについて

Rubyのインストール方法を説明する前に、Rubyのどのバージョンをインストールすべきかについて簡単に説明しておく。

2.x系

Rubyの最新バージョンは2.x系である。2013年9月現在は2.0.0が最新版だが、この冬には2.1.0のリリースが予定されている。本書のプログラムを動かすくらいなら差はないと思われるので、どちらをインストールしてもいい。

1.9系

Ruby 2.x系の前には、Ruby 1.9系が広く使われていた。1.9.3が最後のバージョンになるので、今からRubyを始めるのであれば2.x系を使ったほうがいい。

本書のプログラムは基本的にRuby 1.9でも動くが、一点、ソースコードにマルチバイト文字を使う場合のみ注意が必要になる。例えば、

```
puts "テスト"
```

というプログラムを実行しようとすると、Ruby 1.9では以下のようなエラーになるだろう。

```
$ ruby test.rb
test.rb:1: invalid multibyte char (US-ASCII)
test.rb:1: invalid multibyte char (US-ASCII)
```

これを回避するには、

```
# coding: utf-8
puts "テスト"
```

のように書かなければならない。

1行目の「# coding: utf-8」はマジックコメントと呼ばれ、ソースファイルの文字コードをRubyに伝える役目がある（上記はソースファイルをUTF-8で保存する場合の例で、Windowsの場合はエディタによっては「# coding: windows-31j」でないと動かないかもしれない）。

Ruby 2.0ではソースファイルはデフォルトでUTF-8と仮定されるようになったため、UTF-8を使う限りはマジックコメントは必要ない。

1.8系

1.8系は長く使われたバージョンであるが、2013年6月をもってサポートが終了している。まだRuby 1.8.xを使っている場合は、すみやかに2.x系に移行してほしい。

一応、どうしても1.8系しか使えないというシチュエーションに備えて、本書のサンプルコードをRuby 1.8系で動かすための互換パッチを用意した。スクリプトの先頭にリスト1をそのまま貼り付ける※1と、Rubyの組み込みクラスが以下のように変更される。

※1：またはcompat.rbなどの名前で保存し、スクリプトの先頭にrequire './compat'と書いて読み込ませる。

- String#charsが使えるようになる。
- String#each_charが使えるようになる。
- String#ordが使えるようになる。
- IO#getcが文字列を返すようになる。

完全な互換性を保証するものではないが、本書に出てくるサンプルコード程度ならこれで十分動くだろう。

リスト1: 1.8系のための簡易互換パッチ

```
require 'enumerator'

class String
  def chars
    self.scan(/./)
  end

  def each_char(&block)
    chars.each(&block)
  end

  def ord
    raise TypeError if self.size == 0
    self[/./].unpack("C")
  end
end

class IO
  alias original_getc getc
  def getc
    original_getc.chr
  end
```

Rubyのインストール

Rubyのインストール方法について、執筆時点での情報をプラットフォームごとに簡単にまとめた。最新の情報については、Web雑誌「Rubyist Magazine」の以下のページが参考になる。

Rubyの歩き方 - Rubyist Magazine
http://magazine.rubyist.net/?FirstStepRuby

Windows

WindowsでRubyを使う場合は、以下のいずれかのインストーラを利用するといい。

RubyInstaller
http://rubyinstaller.org/

ActiveScriptRuby
http://www.geocities.co.jp/SiliconValley-PaloAlto/9251/ruby/

Mac OS X

Macについては、もしパッケージマネージャhomebrewを使っているのなら、

```
$ brew install ruby
```

でRuby 2.xをインストールすることができる。

そうでない場合、バージョンは古いがMac OS Xに最初から入っているRubyを使うという手がある。筆者の環境(Mountain Lion)にはRuby 1.8.7が入っていた。ターミナルを起動し、

```
$ ruby -v
```

として確かめてみよう。

Linuxなど、Unix系OS

ディストリビューションによるが、たいていは標準のパッケージマネージャでインストールできるだろう。例えば筆者の手元の環境(Ubuntu 12.05)では、

```
$ sudo aptitude install ruby1.9.3
```

でRuby 1.9.3をインストールすることができる。

また、rbenv(+ruby-build)もしくはRVMといったツールを使うと、比較的簡単に最新版のRubyをソースコードからコンパイルしてインストールすることができる。興味があれば調べてみてほしい。

Chapter 1

Esoteric Languageを知る

1-1 Ruby練習問題

本題に入る前に、Rubyの文法のおさらいをしておこう。

初級編

まず最初は、「プログラミング言語C」から続く伝統的なサンプルプログラムを書いていただこう。

問1　画面に「Hello, world!」という文字列を出力するRubyプログラムを書いてください。

回答例をリスト1に示す。

リスト1: RubyによるHello World!

```
puts "Hello, world!"
```

ご覧の通り、Rubyのプログラムはとてもシンプルだ。「オブジェクト指向スクリプト言語」という難しげな肩書きが付いてはいるが、簡単なプログラムを書くだけならクラスを定義する必要もないし、「オブジェクト指向しなければ」と身構えることはない。

ただしリファレンスマニュアルを読むときは、「すべてがオブジェクトである」というRubyの原則を知っておいたほうがいいだろう。例えば「1」とか「32」のような数値はFixnumというクラスのオブジェクトだし、「"Hello, world"」のような文字列はStringクラスのオブジェクトだ。Rubyで扱うデータはすべて、必ず何らかのクラスに属している。

さらに、普通は言語に組み込まれている演算子も、Rubyでは単に1つのメソッドに過ぎなかったりする。例えば「1 + 2」と書いた場合、これは1というオブジェクトの「+」という名前のメソッドを、2を引数に呼び出すという意味になる。Rubyではすべての操作がメソッド呼び出しなのだ（ちなみに、「Fixnumクラスの+というメソッド」のことを、Ruby界の慣習で「Fixnum#+」と書く。クラスメソッドなら「Math.pow」のようにピリオドを使う）。

ではリスト1で使っている「puts」という命令は何なのだろうか。実はこれもKernelというクラスに定義されているメソッドである。RubyではKernelクラスのメソッドはプログラムのどこからでも呼び出せるので、putsなどの便利なメソッドがあたかもグローバル関数のように使えるというわけだ。

中級編

次はもう少し複雑なプログラムにチャレンジしてみよう。

問2 「99 bottles of beer」の全文を出力するRubyプログラムを書いてください。

「99 bottles of beer」というのは英語で書かれた詩で、壁の上に置かれた99本のビールを一本ずつ取っていくという内容だ（内容に特に意味はない）。もとはマザーグースに収録されている詩だが、アメリカでは遠足のときなどに歌われるそうだ。99から0まで繰り返すので全文がやたら長い上に、0本になったら店に99本買い出しに行ってふりだしに戻るため、相当長い遠足でも大丈夫そうだ。

「99 bottles of beer」の全文をリスト2に示す。

リスト2: 99 bottles of beer

```
99 bottles of beer on the wall, 99 bottles of beer.
Take one down and pass it around, 98 bottles of beer on the wall.

98 bottles of beer on the wall, 98 bottles of beer.
Take one down and pass it around, 97 bottles of beer on the wall.

（中略）

2 bottles of beer on the wall, 2 bottles of beer.
Take one down and pass it around, 1 bottle of beer on the wall.

1 bottle of beer on the wall, 1 bottle of beer.
Take one down and pass it around, no more bottles of beer on the
wall.

No more bottles of beer on the wall, no more bottles of beer.
Go to the store and buy some more, 99 bottles of beer on the wall.
```

念のため書いておくが「(中略)」というのは詩の一部ではない。最初99本あったビールが1段落ごとに1本ずつ減っていて、これが0本になるまで繰り返されるので、合計で100個の段落が存在する。

「99 bottles of beer」はプログラミング言語を紹介する際の例題として使われるようで、その名もhttp://99-bottles-of-beer.net/というサイトにて、1000を超えるプログラミング言語でこの詩を出力するプログラムが書かれている。

それでは、我々もひとつRubyで挑戦してみよう。

▌回答例

回答例をリスト3に示す。

リスト3: 99 bottles of beerの全文を表示するRubyプログラム

```
#
# 99.rb - 99 bottles of beerの詩を表示する
#

99.downto(0) do |k|
  case k
  when 0
    before = "no more bottles"
    after = "99 bottles"
  when 1
    before = "1 bottle"
    after = "no more bottles"
  when 2
    before = "2 bottles"
    after = "1 bottle"
  else
    before = "#{k} bottles"
    after = "#{k-1} bottles"
  end

  if k == 0
    action = "Go to the store and buy some more"
  else
    action = "Take one down and pass it around"
  end

  puts "#{before.capitalize} of beer on the wall, #{before} of
```

```
beer."
  puts "#{action}, #{after} of beer on the wall."
  puts "" unless k == 0
end
```

「#」から始まる最初の3行はコメントで、実行には影響しない。どのプログラミング言語でもそうだが、適当な場所にコメントを入れることで、他人にも、自分にも読みやすいプログラムを書くことができる。

上級編

ここまではRubyに標準で用意されているクラスのみを使ってきたが、次は自分でクラスを定義してみよう。

問3　数を1ずつカウントしていくクラスを作ってください。クラス名は「Counter」で、生成時はカウンタの初期値を表す整数1つを引数にとり、「inc」というメソッドでカウンタを1増やし、「value」というメソッドで現在のカウンタ値を取得できるものとします。

要は、こんな風に使えるクラスを作ってくださいということだ。

```
ct = Counter.new(2)
p ct.value   #=> 2
ct.inc
ct.inc
ct.inc
p ct.value   #=> 5
```

すでに見たように、Rubyでは「#」から行末までがコメントとして無視されるのだった。Rubyのサンプルプログラムでは、慣習的に「#=> 2」のような書き方で「この行を実行すると 2 と表示されます」という情報を表す。

クラスを定義する

回答例をリスト4に示す。

リスト4: counter.rb

```
class Counter
  def initialize(n)
    @value = n
  end

  def inc
    @value += 1
  end

  def value
    @value
  end
end
```

ここではCounterという名前の新しいクラスを定義している。クラス名には英数字とアンダーバーが使えるが、1文字目は必ず大文字でなくてはいけない。

initializeはオブジェクトの生成時に呼ばれるメソッドで、Counter.newに渡した引数がそのまま渡される。ここでは、カウンタ値を表すインスタンス変数@valueを引数のnで初期化している。Rubyでは「@」から始まる名前の変数がインスタンス変数と見なされる。Javaなどの言語でプログラマが独自に「インスタンス変数名は『_』から始める」のように決める場合があるが、Rubyではこのようなルールが言語として決められていると思えばいい。もしあなたがPerlプログラマならvalueが配列を表す変数に見えてしまうと思うが、Rubyではデータの型ではなく変数のスコープによって先頭の1文字が決まることに注意してほしい。

次のincというメソッドはカウンタ値を1増やす。C言語などと違い、「@value++」という書き方はできないことに注意。「なんでRubyには『++』がないの？」という質問は定期的にメーリングリストに流れるが、「オブジェクト指向的に自然な定義をするのが難しいから」というのがRubyの作者、まつもとゆきひろ氏の答えだ。もちろん、この意見を覆すだけのメリットと一貫性のある仕様を誰かが提示できればRubyに「++」が導入される可能性はあるが、今のところそれに成功した人はいないようだ。

次のvalueというメソッドはカウンタ値を返すメソッドだ。値を返すのに、「return」がないことにお気づきだろうか？　Rubyでは

```
  def value
    return @value
  end
```

のように、returnを使って明示的に値を返すこともできるが、「returnがない場合、メソッド内で最後に評価された値が返り値になる」というルールがあるため、プログラマはいちいちreturnを書く必要がない。本書では、特に必要のない限りreturnを省略する。

このような、インスタンス変数の値を取得するためのメソッドは「ゲッター」と呼ばれることがある。値を設定する方は「セッター」だ。Rubyではゲッターやセッターをより簡潔に定義する方法が用意されている。リスト5を見てほしい。

リスト5: attr_readerを使ったcounter.rb

```
class Counter
  def initialize
    @value = 0
  end
  attr_reader :value

  def inc
    @value += 1
  end
end
```

このプログラムではvalueというメソッドを定義していないが、先ほどのリスト4と同じ動作をする。

秘密は「attr_reader :value」という行にある。クラス定義の中では、attr_readerにインスタンス変数名を表すシンボル（後述）を渡すことでゲッターを定義できる。セッターを定義したいときは「attr_writer :value」、ゲッターとセッターを両方とも定義したいときは「attr_accessor :value」のようにすればいい。

attr_readerはRuby言語に用意されている構文のようにも見えるが、実際はModuleクラスの一メソッドとして定義されている。Rubyではクラス定義も上から順に実行され、

例えばクラスの定義中にputsなどのメソッドを呼ぶこともできる。試しに、

```
class Test
  puts "this is a test"
end
```

というプログラムを実行してみてほしい。putsメソッドが実行され、画面に「this is a test」という文字列が表示されたはずだ。

▌シンボルとは？

シンボルは「:」のあとに英数字と一部の記号を続けたもので、Symbolクラスのインスタンスになる。シンボルは文字列とよく似ている（実際、attr_readerは文字列も引数にとることができる）が、以下のような違いがある。

まず、シンボルは内容を破壊的に書き換えることができない。「a = b = "foo"」のように、複数の変数が同じ文字列を指していたとき、「a.upcase!」とすると変数bの値も書き換わる。同じ文字列を指しているのだから当然だ。String#upcase!のように、オブジェクトの中身を書き換えるメソッドを「破壊的なメソッド」と言う。シンボルにはこのような破壊的なメソッドは用意されていない。

また、文字列の場合は「"foo"」と書くたびに別の文字列オブジェクトが作られるのに対し、シンボルは何度「:foo」と書いても、シンボル名ごとに1つのオブジェクトしか作られない。このためシンボルのほうが文字列より「軽い」と言えるだろう。

シンボルと文字列の使い分けとしては、「プログラムの内部だけで使われ、外部に出力されることがない文字列」にはシンボルが使われるようだ。例えば、C言語でいうenum（列挙型）のようなことをRubyでしたいときは、シンボルを使うことが多い。シンボルは文字列より使い道がはっきりしているので、適切に使えばより読みやすいRubyプログラムを書くことができる。

▌クラスを使ってみる

それでは、このクラスを使ったプログラムを書いてみよう。counter.rbというファイルにリスト4を保存したら、同じディレクトリにcounter_main.rbというファイルを作り、リスト6の内容を書き込んでほしい。

リスト6: counter_main.rb

```
require './counter'

ct = Counter.new(2)
p ct.value
ct.inc
ct.inc
ct.inc
p ct.value
```

コマンドラインから

```
$ ruby counter_main.rb
```

のようにして実行すると、

```
2
5
```

のように表示されたはずだ。

require

Rubyではrequireというメソッドを使うことで、他のソースファイルを読み込むことができる。requireには読み込みたいソースファイルのパス名を文字列で渡す。拡張子の「.rb」は省略してかまわない。

リスト6では、

```
require './counter'
```

のようにrequireを呼び出すことで、カレントディレクトリのcounter.rbを読み込んだ。

試し読みはお楽しみ
いただけましたか？

ここからはManatee
おすすめの商品を
ご紹介します。

Tech Book Zone Manatee

3.6
2017

おすすめ電子書籍

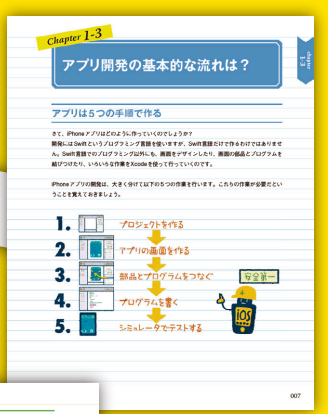
Tech Book Zone Manatee

1

体験型の本書でプログラミングの第1歩を踏みだそう!

『やさしくはじめるiPhoneアプリ作りの教科書【Swift 3&Xcode 8.2対応】』は、iPhoneアプリを作ってみたい初心者のための入門書です。プログラミングが初めての人、苦手意識がある人でも楽しく学んでいけるよう、なるべくやさしく、イラストや図をたくさん使って解説しています。本書では実際にサンプルアプリを作りながら学んでいきますが、イラストによる解説で、一歩ずつ丁寧に、iPhoneアプリ作りの基本と楽しさを学べます。

イラストによる解説で、プログラミングがはじめての人でも学べる



作成するサンプルアプリはシンプルで、意味を理解しながら作っていける

やさしくはじめる iPhoneアプリ作りの教科書【Swift 3&Xcode 8.2 対応】

Swift (iOS開発)

マイナビ出版
森巧尚(著者)、
まつむらまきお(イラスト)
312 ページ　価格：3,002 円(PDF)

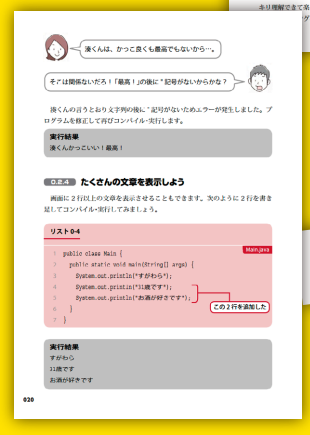
2

豊富なイラストで「なぜ?」を解消! Javaの第一歩を踏み出そう

『スッキリわかるJava入門 第2版』は、Javaの基礎から初学者には難しいとされるオブジェクト指向まで、膨らむ疑問にしっかり対応しました。Javaプログラミングの「なぜ?」がわかる解説と約300点の豊富なイラストで、楽しく・詳しく・スッキリとマスターできる構成となっています。「なんとなくJavaを使っているけれど、オブジェクト指向の理解には自信がない」「学習の途中で挫折してしまった」という方にもおススメです。

300ものイラストで楽しく・詳しく・スッキリとマスター!

「プログラミング」



会話のやりとりの中にも、開発現場でのヒントが詰め込まれている

スッキリわかる Java 入門 第2版

Java

インプレス
中山清喬・国本大悟(著者)
658 ページ
価格：2,376 円(PDF)

幅広いジャンルで活躍！ C# がキホンから学べる本

3

基礎からわかる C#

本書はプログラミングの経験がある人を対象とした、プログラミング言語「C#」の入門書です。C# の概要から基本的な文法、特徴的な機能まで、わかりやすく解説しています。C# 6.0 の新機能についても解説しています。

シーアンドアール研究所
西村誠（著者）
168 ページ　価格：1,944 円（PDF）

&

プログラミング未経験でも Android アプリを開発！

4

イラストでよくわかる Android アプリのつくり方 Android Studio 対応版

親しみやすいイラストや、ステップバイステップでの丁寧な解説が基本コンセプト。開発環境「Android Studio」に対応し、Android のプログラムを作りながら、自然に Java というプログラム言語の知識が身につきます。

インプレス
羽山博・めじろまち（著者）
価格：2,138 円（PDF）

&

JavaScript を網羅的に取り上げた骨太の 1 冊

5

JavaScript 逆引きハンドブック

JavaScript の逆引きリファレンスの決定版。JavaScript の機能を網羅的に取り上げていて、骨太の 1 冊になっています。JavaScript の基本的な処理や便利な Tips はもちろん、HTML5 の API についても数多く掲載しています。

シーアンドアール研究所
古籏一浩（著者）
993 ページ　価格：3,694 円（PDF）

初めてのウェブ開発も安心 Ruby の文法を基礎から解説

6

改訂 3 版 基礎 Ruby on Rails

Ruby の文法やオブジェクト指向の考え方を初歩から解説。アプリケーションのモックアップ作り、データベースを導入し、ログイン・ログアウト機能を加え、最終的にはメンバーや記事の管理ページまでできあがります。

インプレス
黒田努・佐藤和人（著者）
536 ページ　価格：3,240 円（PDF）

&

初歩から順に理解できる PHP とデータベース

7

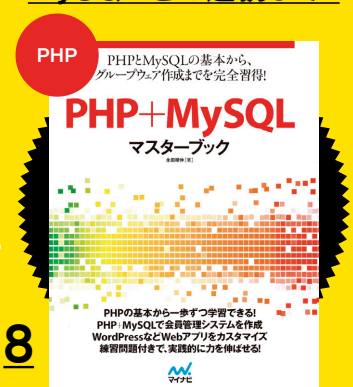
いちばんやさしい PHP の教本

PHP とデータベースの基本を順番に学んで、実践的なプログラムを完成させていく PHP の入門書です。大事なポイントや勘違いしやすいポイントは講師がフォロー。セミナーを受けている感覚で読み進められます。

インプレス
柏岡秀男・池田友子（著者）
240 ページ　価格：1,836 円（PDF）

&

PHP5.4 の基本から MySQL との連携まで！



8

PHP+MySQL マスターブック

この一冊で PHP と MySQL の基本と Web アプリケーションの構築法について学習できる実践的なプログラミング入門です。現場必須のプログラム構築法、API の活用法から、セキュリティ技術まで詳しく解説します。

マイナビ出版
永田順伸（著者）
384 ページ　価格：2,916 円（PDF）

はじめてプログラミングに触れる前に読んでおこう

9

プログラミングの基本から手取り足取りじっくり解説

10

必ずアルゴリズムの意味がわかるようになる入門書!

11

プログラミングの世界へようこそ

全くの初心者がプログラミングを勉強したいとき、さまざまな疑問が湧いてきます。「どの言語を覚えればいい？」「文系でも大丈夫？」本書はプログラミングに触れる前に知っておきたい基本をイラスト付きで解説します。

マイナビ出版
尾川一行・中川聡(著者)
192 ページ　価格：1,933 円(PDF)

目指せプログラマー！プログラミング超入門

本書は Windows 開発の標準ツールとも言える「Visual Studio」を使い、C# というプログラミング言語でプログラミングの基本を学びます。最終的には、ちょっとしたアクションゲームが作れるくらいになるのが目標です。

マイナビ出版
掌田津耶乃(著者)
312 ページ　価格：2,074 円(PDF)

楽しく学ぶ アルゴリズムとプログラミングの図鑑

図解とイラストを豊富に使ったアルゴリズムの入門書。アルゴリズムとは『問題を解決するための考え方』です。それが分かってきたら、8 種類のプログラミング言語を使ったサンプルプログラムを実際に試しましょう。

マイナビ出版　森巧尚(著者)、まつむらまきお(イラスト)
300 ページ　価格：2,689 円(PDF)

プログラミングの初心者がPython 3 を学ぶのに最適

12

Go 言語の基礎から応用までポイントがよくわかる

13

R 言語の機能を目的から見つけ出せる！

14

基礎 Python 基礎シリーズ

プログラミングの初心者を対象にした Python 3 の入門書です。変数の取り扱いから、リスト、タプルといった Python 固有のデータの操作、制御構造や関数などについて、初心者でも基礎から学習できるように説明しました。

インプレス
大津真(著者)
312 ページ　価格：2,894 円(EPUB)

改訂 2 版 基礎からわかる Go 言語

Google が開発したプログラミング言語「Go」の基礎から応用までをわかりやすく解説した 1 冊です。最新の Go 1.4 のバージョンに対応して改訂しました。Linux、Mac OS X、Windows の各環境に対応しています。

シーアンドアール研究所　古川昇(著者)
240 ページ　価格：2,138 円(EPUB)

改訂 3 版 R 言語逆引きハンドブック

本書では、最新バージョンの R 3.3.0 に対応し、R 言語の機能を目的から探すことができます。統計が注目を集めるなか、R を利用するユーザーも増えています。初心者でも使えるように、導入から丁寧に解説しています。

シーアンドアール研究所
石田基広(著者)
800 ページ　価格：4,860 円(PDF)