

日本Rubyの リファレンスマニュアル 2008・初夏

日本Rubyの会 リファレンスマニュアル支部
青木峰郎

About Speaker

- 青木峰郎 (Minero Aoki)
- 某DWHベンダ勤務
- 主著 『ふつうのLinuxプログラミング』
『Rubyist Magazine出張版』
『Rubyソースコード完全解説』 (RHG)
- 標準ライブラリとかいろいろ

最近の

Rubyとの関わり

My latest use of Ruby

Teach Yourself Java

独習Java 第4版

ジョセフ・オニール 著

トップスタジオ 訳

武藤 健志 監修



シリーズ60万部

圧倒的定評!定番中の大定番

コンパクト&リーズナブルになって

JDK6対応で新登場!!

JDK6
ソフトウェア開発キット
収録CD-ROM付属
オールインワン
学習対応

3ステップだからよく分かる

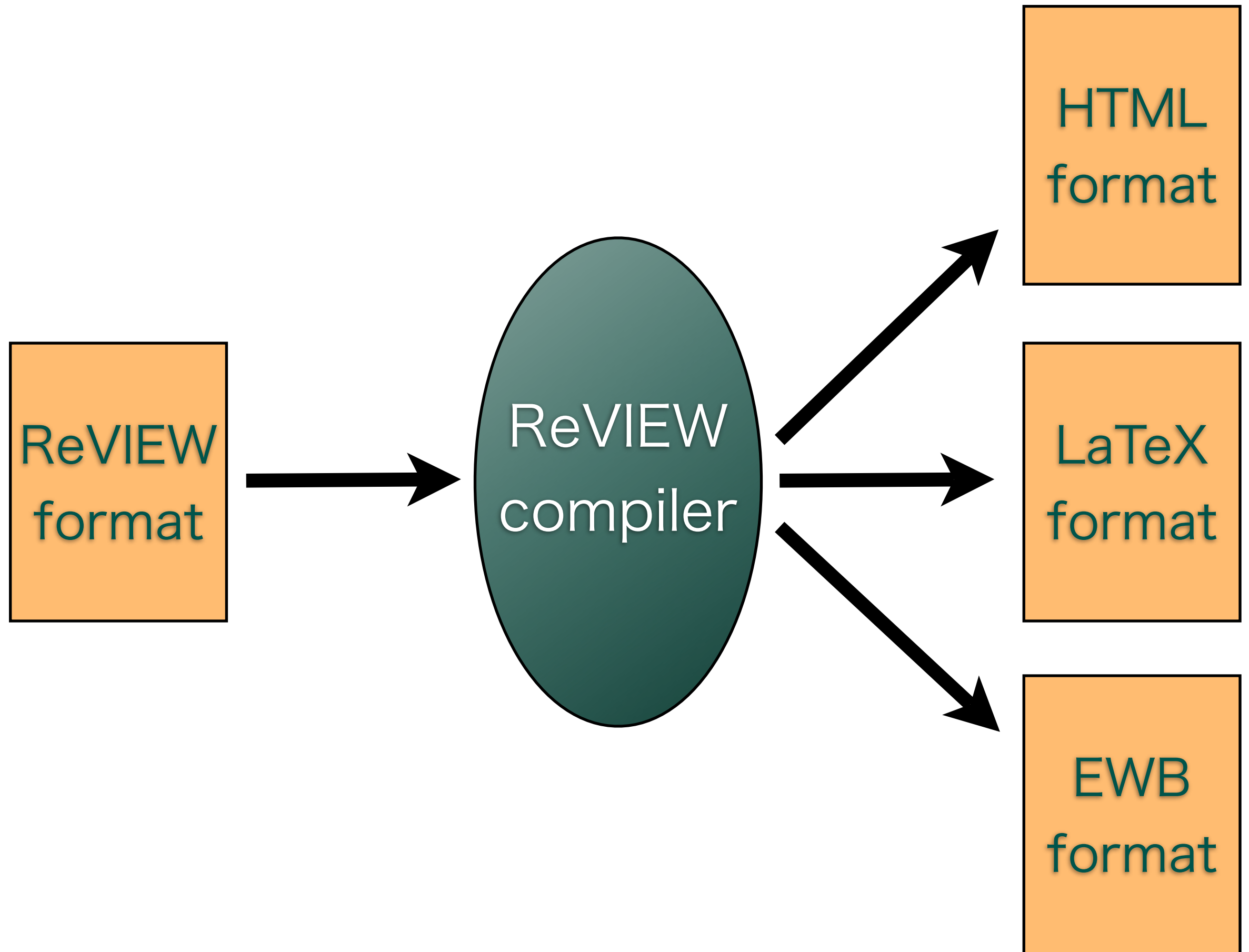
SE
SHOENISHA

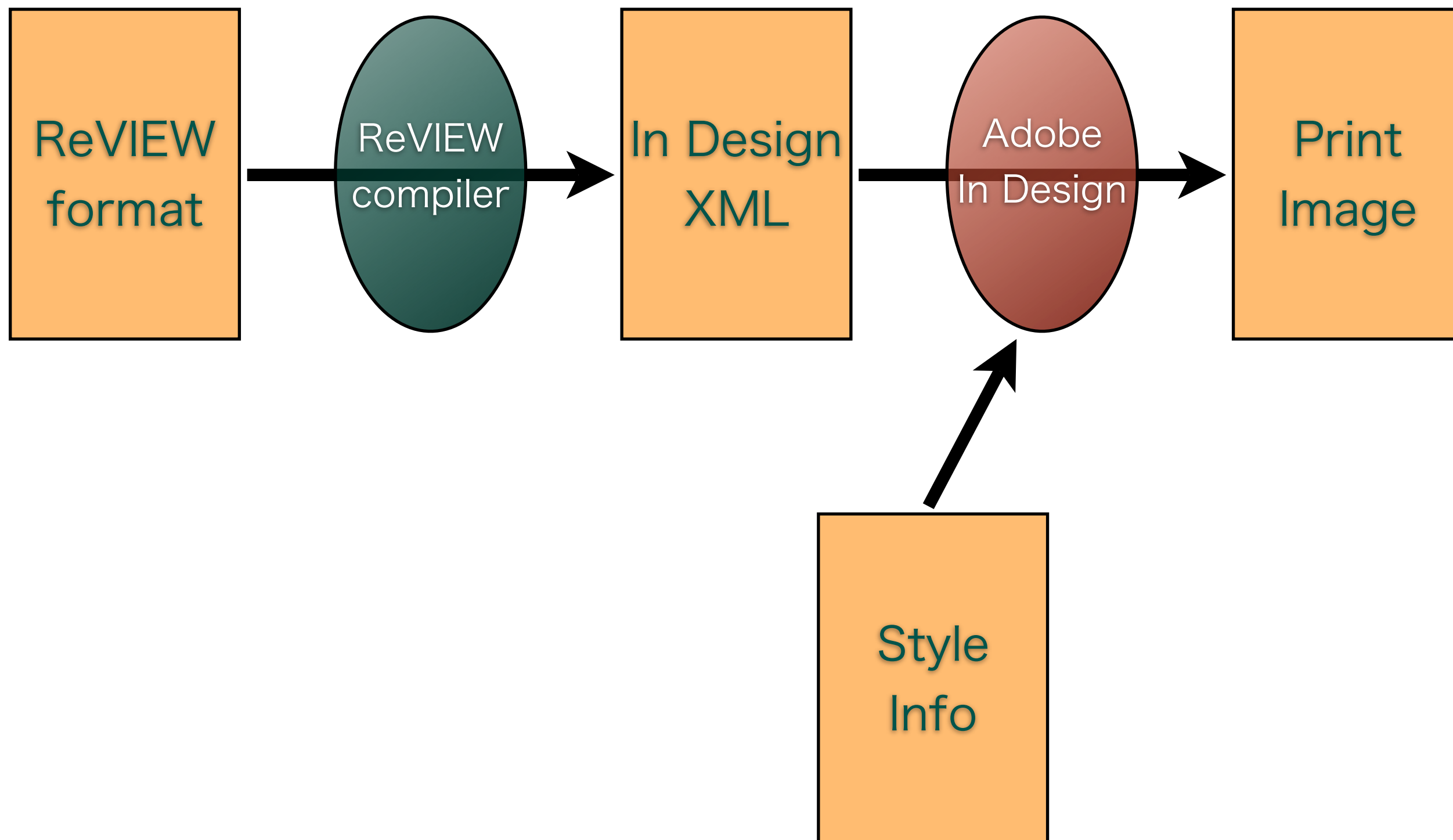
1 解説 2 例題 3 練習問題

『独習Java第4版』

執筆支援システム ReVIEW

Book Compiler "ReVIEW"





= フロントエンドの構成

```
#@defvar(src, src/net/loveruby/cflat)
```

```
//read{
```

この章では、

ソースコードの解析を担当するフロントエンドの構成について一般的な話を述べたあと、

パーサを記述するために使うJavaCCというツールについて概要を説明します。

```
//}
```

== フロントエンド構築の手法

この節では、フロントエンドの一般的な構築手法について説明します。

=== ソースコード解析の問題点

ソースコードの解析は一筋縄ではいきません。

例えばC言語の数式の解析を考えてみましょう。

C言語で

「 $8 + 2 - 3$ 」と書いてあったら「 $(8 + 2) - 3$ 」と解釈すべきですが、

「 $8 + 2 * 3$ 」と書いてあったら「 $8 + (2 * 3)$ 」と解釈すべきです。

このように、数式を解析するときは@<kw>{演算子の優先順位, operator precedence}を考慮する必要があります。



2.1 フロントエンド構築の手法

この節では、フロントエンドの一般的な構築手法について説明します。



ソースコード解析の問題点

ソースコードの解析は一筋縄ではいきません。例えばC言語の数式の解析を考えてみましょう。C言語で「 $8 + 2 - 3$ 」と書いてあったら「 $(8 + 2) - 3$ 」と解釈すべきですが、「 $8 + 2 * 3$ 」と書いてあったら「 $8 + (2 * 3)$ 」と解釈すべきですね。このように、数式を解析するときは演算子の優先順位 (operator precedence) を考慮する必要があります。

また、C言語では、数値が書ける部分には変数や配列アクセスや、構造体メンバへのアクセスも記述できます。関数呼び出しも書けます。コンパイラはこのような多様性を扱わなければなりません。

さらに、数式も変数も関数呼び出しも、コメントの中に書かれていたら無視しなければなりません。文字列の中も同様です。コンパイラはこのような文脈の違いも考えなければならないのです。

このように、プログラミング言語のソースコードを解析するときには、考慮すべきことがいろいろあり、なかなか厄介です。



ソースコード解析の定石

ソースコードを解析するときの様々な問題に対処するため、これまでいろいろな手段が試みられてきました。その成果として、プログラミング言語のソースコードの解析については多くの範囲で定石ができあがっています。基本的には、その定石に従っていけばほとんどのプログラミング言語は解析できてしまいます。

また、定石に従って解析できるようにプログラミング言語を設計しておけば楽ができると言うこともできます。C++言語もそうやって設計した言語です。C言



MYCOMには負けん

Agenda

1.刷新計画の概要

About Project

2.現在の状況

Current Status

3.これからの予定

Next Step

刷新計画の概要

about Project

Rubyリ (ry計画とは

About Project

- Rubyのリファレンスマニュアルを刷新する計画

The project to refine the Ruby reference manual

- より完全で便利なマニュアルを目指す

More complete, useful manual

- 2006年8月開始

Started on Aug, 2006

主な改善点

Major improvement

- しっかりしたプロジェクト体制
Well-organized project team
- 品質の高いドキュメント
High-quality documentation
- より厳密でメタデータを取りやすい
ファイルフォーマット
More strict, reusable file format

Before

Array

配列クラス。配列の要素は任意の Ruby オブジェクトです。一般的には配列は配列式を使って

```
[1, 2, 3]
```

のように生成します。

スーパークラス:

- [Object](#)

インクルードしているモジュール:

- [Enumerable](#)

メソッド一覧:

クラスメソッド:

[Array\[item,...\]](#) [Array.new](#)

メソッド:

[self\[nth\]](#) [self\[start..end\]](#) [self\[start, length\]](#) [self\[nth\]=val](#) [self\[start..end\]=val](#) [self\[start, length\]=val](#) [+ * - & | << <=> == assoc at](#)
[clear](#) [clone](#) [collect!](#) [compact](#) [compact!](#) [concat](#) [delete](#) [delete_at](#) [delete_if](#) [dup](#) [each](#) [each_index](#) [empty?](#) [eql?](#) [fetch](#) [fill](#) [first](#) [flatten](#)
[flatten!](#) [include?](#) [index](#) [indexes](#) [indices](#) [insert](#) [join](#) [last](#) [length](#) [map!](#) [nitems](#) [pack](#) [pop](#) [push](#) [rassoc](#) [reject!](#) [replace](#) [reverse](#) [reverse!](#)
[reverse_each](#) [rindex](#) [shift](#) [size](#) [slice](#) [slice!](#) [sort](#) [sort!](#) [to_a](#) [to_ary](#) [to_s](#) [transpose](#) [uniq](#) [uniq!](#) [unshift](#) [values_at](#)

クラスメソッド:

`Array[item,...]`

引数を要素として持つ配列を生成します。

`Array.new([size[, val]])`

`Array.new(size) { |ruby 1.7 feature| }`

After

Ruby 1.9.0 > [Home](#) > [All Libraries](#) > library [_builtin](#) > class Array

class Array

ancestors: Array < [Enumerable](#) < [Object](#) < [Kernel](#) < [BasicObject](#)

Abstract

配列クラスです。配列は任意の Ruby オブジェクトを要素として持つことができます。

一般的には配列は配列式を使って

```
[1, 2, 3]
```

のように生成します。

Singleton Methods

signature

description

[\[\]](#)(*item) -> Array

引数 item を要素として持つ配列を生成して返します。

[new](#)(size = 0, val = nil) -> Array

長さ size の配列を生成し、各要素を val で初期化して返します。

[new](#)(ary) -> Array

指定された配列 ary を複製して返します。 [Array#dup](#) 同様 要素を複製しない浅い複製です。

[new](#)(size) { |index| ... } -> Array

長さ size の配列を生成し、各要素のインデックスを引数としてブロックを実行し、各要素の値をブロックの評価結果に設定します。

[try_convert](#)(obj) -> Array | nil

to_ary メソッドを用いて obj を配列に変換しようとします。

Instance Methods

<http://doc.loveruby.net>

現在の状況

Current Status

Timeline

- 2006-08-27: Project started
- 2007-01-08: Method entries completed
- **2007-12-25: 1.8.6 manual released**
- 2008-05-03: snapshot #1 released
- 2008-05-30: snapshot #2 released

メソッドカバー率

Standard Libraries

31.2%

5493 / 17588

メソッドカバー率を

劇的に向上させる

魔法の呪文 (消費MP3)

「tkとsoap抜き」

"Without tk and sorp"

Standard Libraries

52.0%

4862 / 9357

来年のRuby会議
までに終わる！

Built-in Library

97.2%

1875 / 1929

コミット数ランキング

1089 sheepman

317 okkez

225 iwadon

202 aamine

132 date

105 moriq

103 kouya

70 eklerni

61 snoozer05

39 tadf

32 bornite

RedMine導入

ReferenceManualRenewalProject - チケット - Ruby Issue Tracking System

http://redmine.ruby-lang.org/projects/rurema/issues

インデックス Remember The Milk Amazon.co.jp [mixi] アップル (11) News (348) ニュース お役立ち プログラミング

ホーム プロジェクト ヘルプ ログイン 登録する

ReferenceManualRenewalProject

概要 活動 チケット 新しいチケット ニュース ファイル リポジトリ

チケット

フィルタ

☒ ステータス 未完了 フィルタ追加:

☒ 適用

✓ #	▼ トラッカー	ステータス	優先度	題名	担当者	更新日
☐ 196	Bug	Open	Low	rake のマニュアル		2008年06月21日 17:08 PM
☐ 195	Bug	Open	Low	イテレータ・ブロック・呼び出しブロックの用語の使いかたを統一		2008年06月21日 13:21 PM
☐ 192	Bug	Open	Normal	Time#succのタイムゾーン		2008年06月18日 23:26 PM
☐ 191	Bug	Open	Low	refe succ		2008年06月18日 23:09 PM
☐ 188	Feature	Open	Normal	メソッド名のリンクをもっと目立たせる。		2008年06月19日 00:45 AM
☐ 187	Feature	Open	Low	用語集に「ヒアドキュメント」を追加する。		2008年06月18日 19:16 PM
☐ 186	Feature	Open	Low	String#succの命名の由来や読み方があるといい		2008年06月18日 23:13 PM
☐ 184	Feature	Open	Normal	Fixnumのabstractに、Bignumとの自動変換の例を追記する。		2008年06月18日 15:10 PM
☐ 183	Feature	Open	Normal	Array.[]の使いどころについて追記する。		2008年06月18日 15:08 PM
☐ 182	Feature	Open	Normal	Array.newで、説明の直後に例があったほうがよい。		2008年06月18日 15:06 PM
☐ 181	Feature	Open	Low	用語集に「浅い複製」を追加する。		2008年06月18日 15:05 PM
☐ 180	Feature	Open	Normal	Array.new(size) { index ... } に例を追加する		2008年06月18日 14:58 PM
☐ 176	Feature	Open	Normal	「Abstract」や「signature」などを日本語にする。		2008年06月17日 19:14 PM
☐ 173	Bug	Open	Normal	組み込み変数のページを復活させる		2008年06月17日 08:12 AM
☐ 170	Feature	Open	Normal	演算子のsignatureがわかりにくい		2008年06月16日 18:21 PM
☐ 158	Bug	Open	Low	matrix ライブラリの説明		2008年06月15日 12:32 PM
☐ 157	Bug	Open	Low	JSON→JSON (JavaScript Object Notation)		2008年06月15日 12:20 PM
☐ 139	Bug	Open	Normal	仕様のページの (<ruby 1.7 feature>)などをどうにかする。	Takashi Tamura	2008年06月11日 00:44 AM
☐ 137	Bug	Assigned	Low	Complex#% の説明	Tomohiro Koike	2008年06月16日 23:52 PM
☐ 136	Bug	Assigned	Low	Complex.polar の説明	Tomohiro Koike	2008年06月16日 23:50 PM
☐ 135	Bug	Assigned	Low	Complex.new の説明	Tomohiro Koike	2008年06月17日 22:39 PM
☐ 134	Bug	Open	Low	Kernel.#Complex が無い		2008年06月10日 13:41 PM

チケット

チケットを全て見る
サマリ
変更記録

システムの改善

- 「Ruby言語仕様」などが表示できるようになった
- メソッド検索ができるようになった

これから の 予定

Next Step

リリース

- 6月内に1.8.7対応マニュアルをリリース予定

We will release Ruby reference manual 1.8.7 in this month.

- 組み込みライブラリは100%を目指す
100% coverage for built-in library

クラスリファレンス

- 来年のRuby会議までにメソッドカバー率100%を目指す (tk, soap抜きで)

We will cover all method until Ruby Kaigi 2009.

- 合宿でREXMLを撃破

Documentation camp to write REXML manual

Ruby言語仕様

- 青木が執筆予定だったが無理ぽい
I give up writing Ruby language specification.
- 水面下で交渉中
I'm contacting another person.

C API リファレンス

- プランなし

No plan.

システム (BitClust)

- 静的HTML出力の実装

Static HTML generation

- 新しいデザインの実装

New page design

ライセンス変更

- 現在は（変な）独自ライセンス

Current license is original, ambiguous one

- CCの「表示・継承」  を検討

Considering Creative Commons Share-Alike License

最後に一言

オレ達はようやくのぼりは
じめたばかりだからな……

このはてしなく遠い
ドキュメント坂をよ……

未
完

※青木先生の次回作にご期待ください。

Q?

『ふつうのコンパイラ』 章目次

1. コンパイラを作り始めよう
2. フロントエンドの構成
3. 字句解析
4. JavaCCによるパーサの記述
5. 構文解析
6. JavaCCのアクションと抽象構文木
7. 抽象構文木の作成
8. 意味解析 (1) 参照の解決
9. 意味解析 (2) 静的型チェック
10. x86アーキテクチャの概要
11. x86アセンブラプログラミング
12. 関数呼び出しと変数参照のコンパイル
13. 式と文のコンパイル
14. 最適化
15. リンクとライブラリ
16. プログラムの起動とダイナミックリンク
17. 共有ライブラリの生成
18. この本を読み終えたあとに

8月出版予定

<----- いまココ！

