

Ruby による学習教材の開発

松永 豊

情報教育講座

Development of the class teaching material with Ruby

Yutaka MATSUNAGA

Department of Information Sciences, Aichi University of Education, Kariya 448-8542, Japan

1. はじめに

周知の通り, コンピュータ上で動作するあらゆるアプリケーションやシステムは人間が予め入力しておいた命令の集まり, すなわち, プログラムによって動作している。コンピュータに命令するための言葉, すなわち, プログラミング言語は, 今日では非常にたくさんの種類がある。有名なものをざっと挙げると, C, C++, JAVA, C#, BASIC, FORTRAN, PASCAL, LISP, PHP, LOGO, Smalltalk, JavaScript, Perl, Python, Ruby などがある。どの言語にもそれぞれ特徴があり, 用途によって向き不向きがあるため優劣を付けることは難しいが, 時代とともに古いプログラミング言語の欠点を補うべく新しいプログラミング言語が登場しており, プログラミング言語も進化してきたことは間違いない。

プログラミング言語の進化とは, 大雑把に言えば, より能率よく開発できて, バグ(プログラムミスのこと)が混ざりにくい構造への進化といえる。一昔前はコンピュータの性能(計算速度や記憶容量など)が低かったため, 1バイトでも短くなるようなテクニックがもてはやされた時代もあったが, 現在では性能が大幅にアップしたため, 多少, 計算速度やプログラムサイズを犠牲にしてでも, 見やすいプログラムを書くことが重要になってきている。つまり, コンピュータ寄りのプログラミングから, 人間寄りのプログラミングが重要とされてきている。例えば, 構造化プログラミングやオブジェクト指向プログラミングなどはこのような発想のもとに提唱されてきた概念である。

さて, 近年, Ruby というプログラミング言語が注目されている^{[1][2]}。Ruby はオブジェクト指向型のスクリプト言語であるが, 利便性の高さから, その場限りの小規模なユーティリティは元より, 本格的なシステムにも使用され始めている。また, 読みやすく書きやすい文法なため, プログラミング学習者にも適してい

ると思われる^{[3][4][5]}。

Ruby 自体は1990年代後半あたりから徐々に使用されてきてはいたが, それほどユーザ数は多くなかった。ところが2004年に発表されたRuby on Rails が2005年に大ブレイクしたため, Ruby 人口が大幅に増加したといわれる。今後, ますます, Ruby 人口が増えることが予想され, プログラミング教育という観点からも重要になってくるかもしれない。

そこで, 本研究ではRuby を用いた教育教材の開発を行った。ここでは, 各自がダウンロードして実体験するタイプの教育教材の作成を目的とした。基本的には完成した教育教材としての利用が目的ではあるが, Ruby プログラムは他人が書いたプログラムでも比較的読みやすいため, 少々の変更であればソースレベルでの拡張等も授業時間内に可能であると考えられ, 幅広い授業構成が期待できる。また, プログラミング学習にも役立つと思われる。

2. Ruby について

ここで, Ruby について少し説明しておく。Ruby とはまつもとひろゆき氏によって開発された国産のプログラミング言語であり, 主な特徴として以下のようなものが挙げられる。

- ・スクリプト言語
- ・完全なオブジェクト指向言語仕様
- ・クラスライブラリが豊富
- ・ネットワークプログラミング対応
- ・ガーベージコレクタ(GC)によるメモリ管理
- ・GUI 対応
- ・マルチバイト対応
- ・GPL

ここで, もっとも大きな特徴は完全なオブジェクト指向型のプログラミング言語であるということである。オブジェクト指向をうたったプログラミング言語は複数存在するが, 真のポリモーフィズム等を兼ね備

えた「完全な」オブジェクト指向言語はそれほど多くはない。

また、Ruby はプログラミングを「楽に」するだけでなく、プログラミングを「楽しく」するように設計されており、プログラミング学習者にも適している言語であると考えられる。

3. オブジェクト指向について

オブジェクト指向言語について述べる前に、よいプログラミング言語とは何かについて考えておこう。最も重要な要素は一言で言えば、

『期待通りの動作をするプログラムを能率よく書くこと』

に尽きると思われる。「期待通りの動作をする」ということは当然バグが入りにくいということの意味するため、後で見直したときにも読みやすい、すなわち、他人にも理解しやすい、保守管理が容易、などにもつながる。

ところで、人間にとってきれいな(分かりやすい)プログラムというのは計算機自体にとっては必ずしも最適なものではない。例えば構造化プログラミングなどは良い例である。昔のプログラミング言語(FORTRAN や BASIC 等)においては goto 文が頻繁に使用されていた。しかし、構造化プログラミングにおいては、goto を使わずに順次、反復、分岐の3つの構造のみで構成することを推奨している。無論、goto 文を使えば、この3つの構造は実現可能なわけだから、明らかにプログラミングの自由度が失われることになる。また、自由度が失われるということは完璧な最適化が行き届かず、処理速度やプログラムサイズを若干犠牲にするかもしれない。一見すると悪いように思うかもしれないがそうではない。自由度の高いgoto文を濫用すると処理の流れが追いにくい可読性の悪いプログラム(俗にスパゲッティプログラムという)になり、バグの発見が極めて困難になる。つまり、この自由度の制限こそがバグの混入を低減させることにつながる。goto 文自体はC や PASCAL 等にも存在するが、なるべく goto 文は使用せずに、while 文や for 文等で作成すべきだという解説書が多いのはそのためである。

プログラム中のコメントも、基本的に人間用と考えてよいだろう。計算機自体はコメント行を無視するわけだから、shebang line 等を除けば書いても書かなくても影響はない。すなわち、純粋にソースコードサイズという点から言えば容量を犠牲にしているわけである。しかし、コメントが全くないプログラムとコメントがしっかり入ったプログラムのどちらが良いかといえば、当然、後者になる。複数の人間が開発に携わる場合などはむしろ必須で、プログラム本体よりもコメントのほうが長いなどという例も多々ある。

さて、オブジェクト指向というのも人間寄りの概念である。これは、オブジェクト(もの)を中心としたプログラミングであり、データと関連性が深い手続きをまとめた構造になっている。主な特徴としては、クラスとインスタンスの分離、クラスの継承、カプセル化、ポリモーフィズムなどが挙げられる。また、若干の賛否はあるものの、人間の精神構造に近いという考えもある。いずれにせよ、GUI プログラミング等では大きな成果を上げているといえる。

Ruby においては、オブジェクト指向的な構造が徹底している。まず、文字リテラルも含め、すべてがオブジェクトである。また、いわゆる変数の型というものが存在せず、完全なポリモーフィズムを実現している。Ruby においては、「オブジェクトの型はそのオブジェクトのクラスではなく、そのオブジェクトに何ができるかによって決まる」という発想が根底にある。(これを duck typing という)

4. 教育教材の開発

近年、e-Learning が盛んに叫ばれている。とりわけ、本学においては全学生がノート PC を所有していることもあり、ノート PC 上で稼動する教育教材を利用した授業が極めて重要であると考えられる。これは、全学レベルでいえることであるが、とりわけ情報教育課程・情報科学課程に限って言えば、プログラミングにも力を入れていることもあり、教育教材の利用が教育教材開発へのステップにもなりうる。

今回は、筆者が担当している授業の教育教材として、人工生命関連の説明の際に使用するシミュレーションソフトの開発を行った。これまで数々の教育教材ソフトを主に C++ を用いて開発してきたが、ここでは Ruby を使って作成することを目的とした。

5. 開発環境

Ruby は C 言語で移植性を考慮して書かれているため、Unix, Mac, Windows など、さまざまなプラットフォームで動作可能である。しかし、今回は学生が所有しているノート PC 上での動作を考えて、Windows 用ソフトの開発とする。

本研究で使用した開発環境は図1の通りである。なお、バージョンは本稿執筆時における最終的なものである。

```
Windows XP Professional
RDE 1.1.1
Apollo 0.841a
ruby 1.8.4
exerb 4.2.0
```

図1. 開発環境

RDE は sakazuki 氏による Windows ユーザのための Ruby 開発環境である^[6]。スクリプトの作成, 変更, 実行, デバッグ等が利用できる。

Apollo はもりきゅう氏による Ruby と Delphi の相互インタフェースである^[7]。GUI を提供する拡張ライブラリとしては Tk 等の選択肢もあったが, 今回はターゲットが Windows に限定していたので, Delphi インタフェースを導入した。これは, これまで筆者が C++Builder を使って教育教材等を作成してきたことや, 本学情報教育コースにおいて C++Builder (あるいは Turbo C++ Explorer) を用いたプログラミング授業を行っていたことなどが理由である。

exerb は加藤勇也氏による Ruby スクリプトを Windows 実行形式ファイルに変換するためのソフトウェアである^[8]。これにより Ruby ソースコードでなく, EXE ファイルで配布することも可能である。なお, この場合, 当然授業時にソースレベルでの変更等を含む演習はできない。

6. 開発にあたって感じたこと

Ruby は開発段階で Perl の代替となることが重視されており, オブジェクト指向要素を強くした Perl のような存在という表現が使われることがある。実際, 文字列処理等においては, C や C++ と比べるとはるかに高い処理能力を持っていることは Perl と同様である。一方, オブジェクト指向が強いことにより, コンパクトでありながら可読性にも優れたプログラムが比較的容易に書けると感じた。

以下, Ruby を用いてプログラミングする際に感じたことを述べる。

すべて (数値リテラルや文字リテラルすら) がオブジェクトであることは, 慣れると極めて生産性が高いものであることがわかる。例えば, 以下のようなプログラムが可能である。

ソース	
1	5.times do
2	puts "Hello"
3	end
実行結果	
1	Hello
2	Hello
3	Hello
4	Hello
5	Hello

図2. 数値リテラル (5) すらもオブジェクトという例

コンテナ, ブロック, イテレータは非常に強力。この概念にしたがってプログラムすると, 従来のループ構文を使わなくなる。

ソース	
1	a = [1, 3, 5, 7, 9]
2	a.each { i puts i+1 }
実行結果	
1	2
2	4
3	6
4	8
5	10

図3. イテレータ

配列, 文字列, 連想記憶等の機能が充実しており, 動的に管理されているため, 使用が簡単。無論, これはガーベジコレクタ (GC) によってメモリ管理されていることも大きな要因である。

イベントハンドラ用のメソッドの定義も容易。

ソース	
1	#・・・省略・・・
2	button1 = Phi::Button.new form1
3	button1.on_click = proc do
4	#イベント処理関数
5	#・・・省略・・・
6	end

図4. イベント処理関数の設定

新規クラスの定義も簡単であり, さらにメソッドにブロックを有効に使うことで既に組み込まれているライブラリと同様, 拡張性の高いクラスが作成できる。

今回作成した学習ソフトにおいても, 作成したクラス (ライブラリ) においてはブロックを積極的に取り入れた。その結果, 新しく作成したクラスをデフォルトで使う場合はプログラムが極めてコンパクトに記述できる。と同時に条件の変更などがブロックで実現できるため, 拡張性の高いクラスが定義できたと思われる。

7. ま と め

本研究では, Ruby を用いた学習教材の開発を行った。また, Ruby でのプログラミングについて調べた。その結果, Ruby は極めて生産性の高いプログラミング言語であることが分かり, プログラミング教育への使用についても期待ができる言語であることが分かった。

Ruby にはさまざまなライブラリが存在している。Web, FTP, SOAP 関係のライブラリ等, ネットワーク関係のライブラリも充実しているため, 今後, ネットワーク対応の学習教材も作成する予定である。その

他，データベース関連やマルチスレッドなど極めて高い拡張性を有していることも魅力的である。

Ruby 自体はスクリプト言語ということもあり，C 言語等の実行速度は期待できない。しかしながら，必ずしも実行速度が最優先されるとは限らず，それよりも，バグが混在しにくいことのほうが重要な場合も多いので，さまざまな用途での利用が可能であると思われる。現在も Ruby 人口は確実に増えているが，今後，ますます Ruby 人口が増えれば，利用されるシステム，分野も増えていくため，プログラミング教育が大いに意味が出てくる可能性がある。そういう意味でも注目していきたい言語の一つである。

参考文献

- [1] 原 信一郎 『Ruby プログラミング入門』 オーム社，2000
- [2] 熊谷 秀武 『Ruby の冒険(旅立ち篇・遊々篇)』 MYCOM，2003
- [3] Dave Thomas 他 『プログラミング Ruby 第 2 版(言語編・ライブラリ編)』 オーム社，2006
- [4] 青木 峰郎 他 『Ruby レシピブック 第 2 版』 SoftBank Creative，2007
- [5] Lucas Carlson 他 『Ruby クックブック』 オライリー・ジャパン，2007
- [6] <http://homepage2.nifty.com/sakazuki/rde/>
- [7] <http://www.moriq.com/apollo/>
- [8] <http://exerb.sourceforge.jp/>

(平成19年9月18日受理)

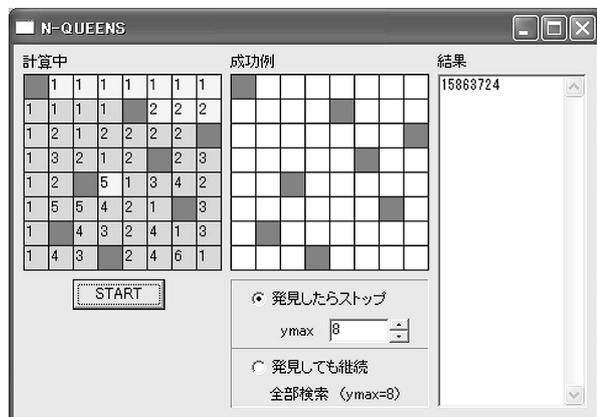
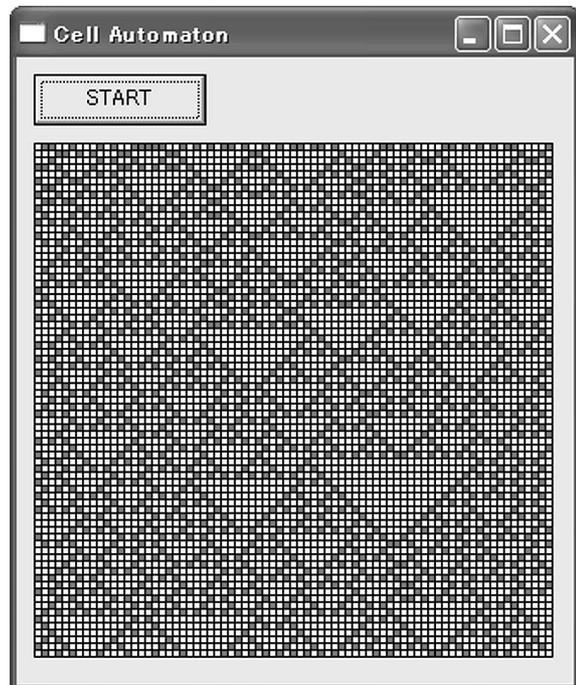
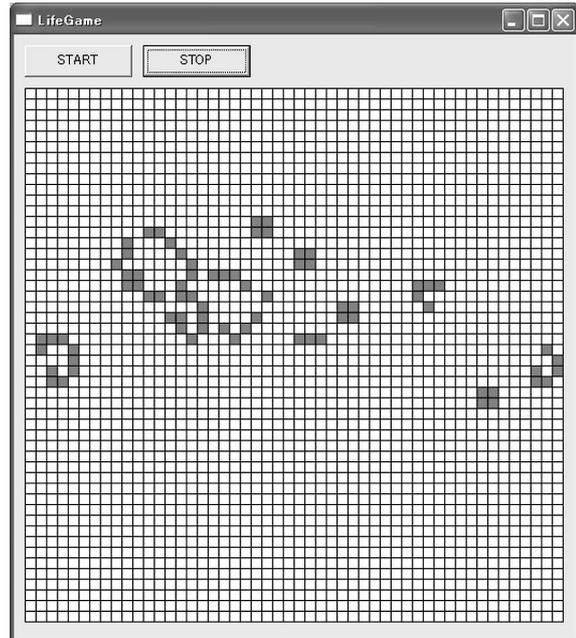


図 5 . 開発したソフトの例