Ruby/SDL の現在と未来

大林一平

KMC/京都大学理学研究科数学教室

2009/7/25

目次

目次

内容はあまりまとまりがありません。

目次

質問等あれば発表中でも随時聞いてください。

自己紹介

- 大林一平 (ohai/おはい)
- 京都大学理学研究科数学専攻の博士課程の学生
  - 専門は力学系 (Dynamical system)
  - 力学 (Mechanics) ではない
- 京大マイコンクラブ
- Ruby/SDL, RRSE, SDLSKK

自己紹介

- 使いこなせる言語
  - C
  - D
  - Ruby
  - Haskell/ML
- まあそれなりに使える
  - シェルスクリプト
  - C++
  - Scheme
  - Emacs Lisp
- 一時期かなり使ってたい
  - x86 アセンブリ言語、Data Flow C、N88 Basic、awk、Pascal
- 触れたことがある
  - Erlang, Python, SQL, Fortrun, BF, Scala, lua

自己紹介

- 使いこなせる VCS
  - CVS
  - Subversion
  - Mercurial
- 一時期かなり使ってたい
  - Monotone
- 触れたことがある
  - darcs, git

自己紹介

- 好きなビデオゲームはシューティングと Rogue like
  - R-TYPE Δ
  - Darius 外伝
  - Thunder Force IV, V
  - Metal Black
  - NetHack(4)
  - Hengband(3)
  - DungeonCrawl(10 ↑)
- ボードゲーム、カードゲーム

最近やっていること

- Scheme の処理系を書いたり
- LiveCoding にでたり
- Ruby 会議に行ったり

Ruby/SDL の概要

- SDL とは何か
- Ruby/SDL とは何か
- Ruby/SDL でできること
- Ruby/SDL でできないこと

SDL とは何か

SDL(Simple Directmedia Layer) とはクロスプラットフォームなマルチメディアライブラリである。
Simple DirectMedia Layer is a cross-platform multimedia library.

マルチメディア？

マルチメディアとは要するに

- 音声
- 映像
- 入力 (マウス、キーボード、ジョイスティック)

を扱うという意味。

で

動画再生ソフトやゲーム、エミュレータを作るのによく利用される。

クロスプラットフォーム？

いろんな環境で使える。

- Linux
- Widows
- Mac OS X
- *BSD
- etc...

Ruby/SDL とは

RubyでSDLが使える。
つまり、「Ruby」で「様々なプラットフォーム上において」「ゲームのような映像、音声、入力を取り扱う」ソフトウェアが作成できる。
ライセンスはLGPL。

Ruby/SDLでできること

- 2D, 3Dの表示
- ゲーム的UIの実現
  - 入力を直接取り扱える
  - よくあるGUIフレームワークではこれがめんどくさい
- 音声の再生

Ruby/SDLでできること

- 2D 高速描画
- キーボードやマウス、ジョイスティックからの入力の取り扱い
- SDL_mixer による音声再生
- CD-ROM の再生
- SDL_ttf、 SDL_kanji、 SGE による文字列の描画
- OpenGL による 3D 描画
- 時間の計測および一時停止
- SDLSKK による、行単位の日本語入力
- SMPGE による mpeg の再生
- その他にも、Window の管理や衝突判定など

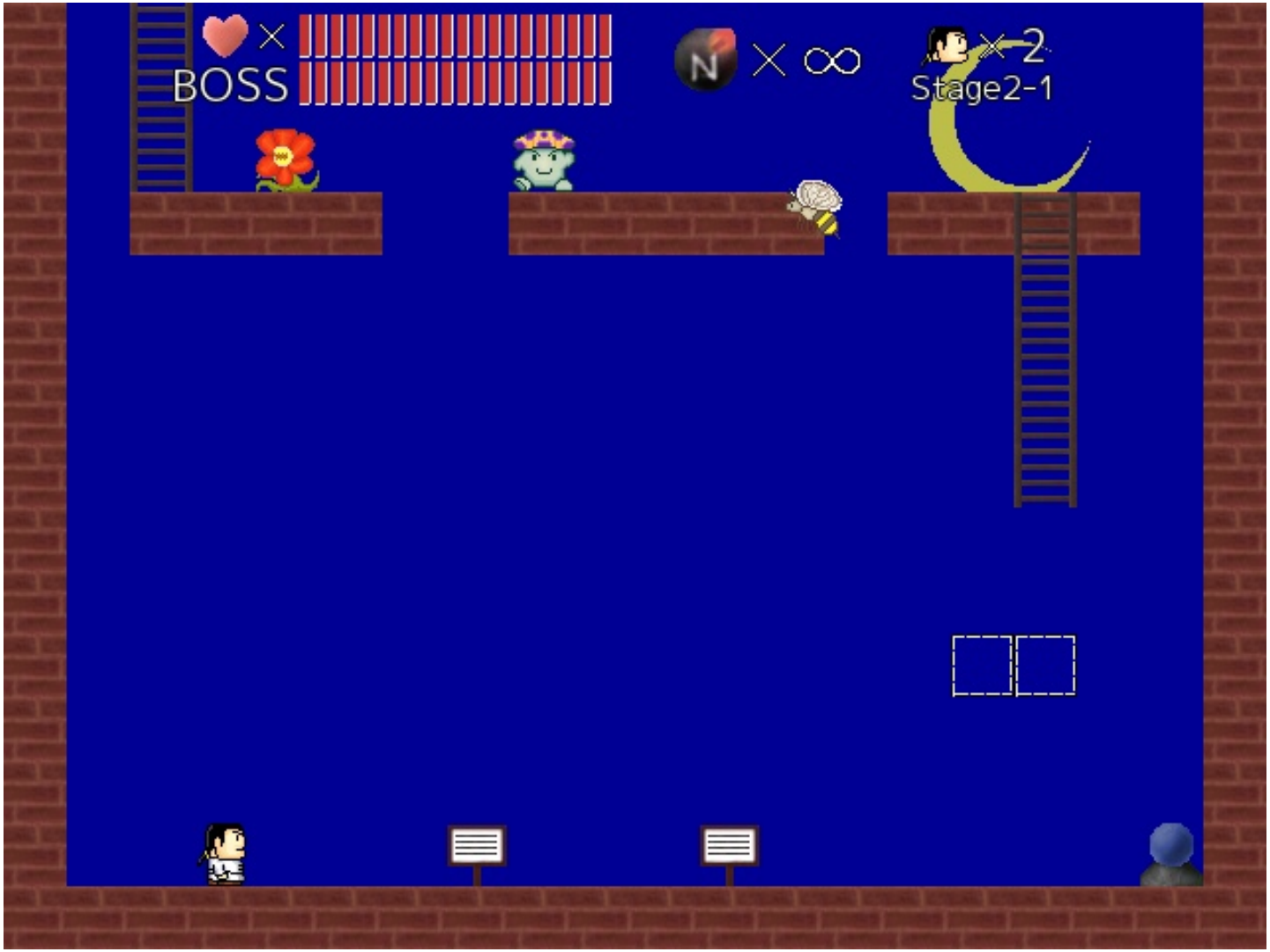
Ruby/SDLでできないこと

- よくあるGUIの実現
  - button, label, check box, etc...
  - lowlevelな要素から作る必要がある
- 複数のwindowを開く
- 普通の日本語入力

Ruby/SDLで作られたソフトウェア

Ruby/SDLで作られたゲームをいくつか紹介します。

熊カレー



Down!!


SCORE RANKING
FPS:61
1: 6609 pede
2: 6179 tic
3: 4989
4: 3931 T.HGW/A
5: 3793 takemaru
6: 3691 nemo--i
7: 3687
8: 3685 anata
9: 3475 atushi
10: 3191 henkma
SCORE:447
LIFE
Ver.0.51

初歩の Ruby/SDL

- インストール
- window を表示
- ちょっと描画

インストール

- Linux や FreeBSD ではパッケージで簡単
- Windows の場合
  - Ruby をインストール
  - Ruby/SDL の zip(win32-bin) を取ってくる
  - zip の中の install_rubysdl.rb を実行
  - Vista の場合 UAC のせいでインストールできない
  - その場合は ASR の場合は回避策がある
- Mac OS X

windowを出す

SDL.init で初期化。SDL::Screen.open で Window を表示。

```
require 'sdl'
SDL.init(SDL::INIT_EVERYTHING)
screen = SDL::Screen.open(640, 480, 0, 0)
loop do
  while ev = SDL::Event.poll
    case ev
    when SDL::Event::KeyDown
      exit if ev.sym == SDL::Key::ESCAPE
    when SDL::Event::Quit
      exit
    end
  end
  # do anything
end
```

描画

SDL::Surface クラスを利用する。

```
require 'sdl'
SDL.init(SDL::INIT_EVERYTHING)
screen = SDL::Screen.open(640, 480, 0, 0)

image = SDL::Surface.load("sample.bmp")
image.set_color_key(SDL::SRCCOLORKEY|SDL::RLEAC
                    image.get_pixel(0, 0))
image = image.display_format
```

```
loop do
  while ev = SDL::Event.poll
    case ev
    when SDL::Event::KeyDown
      exit if ev.sym == SDL::Key::ESCAPE
    when SDL::Event::Quit
      exit
    end
  end

  screen.fill_rect(0, 0, 640, 480, [0,0,0])
  SDL::Surface.blit(image, 0, 0, image.w, image
                    screen, 100, 100)
  screen.update_rect(0, 0, 640, 480)
end
```

初歩の Ruby/SDL

初歩的な話をしていると時間がなくなってしまうのでこのあたりでおわりににします。入門的にはとりあえず以下の資料を参考にしてください。

- 付属のリファレンスマニュアル (http://www.kmc.gr.jp/~ohai/rubysdl_ref_2.html)
- るびまの記事 by yhara (http://jp.rubyist.net/magazine/?0018-GameProgramingForRubySDL)
- http://www.kmc.gr.jp/~ohai/rubysdl_intro.html

初歩でないRuby/SDL

高速な描画のために

ゲームを作るにあたっては描画の速度がゲーム全体の速度の問題をかなりの部分決定します。そのため、この部分を高速化することで全体の高速化が実現されます。そこで、高速化のこつを説明しましょう。

高速な描画のために

高速な描画にはハードウェアのサポートが必須。そこで以下のようなことが言えます。

- ハードウェアサーフェスを使う
- SDL::Surface#display_format を使う
- blit と fill_rect のみを使う
- アルファブレンドはできるかぎり使わない
- 回転や拡大縮小は描画時にしない

高速な描画のために

- ハードウェアサーフェスを使う

SDL::Surface.open に
SDL::HWSURFACE|SDL::FULLSCREEN を渡すとハードウェアによる高速化が利用可能になります。
フルスクリーンでしか使えない、Linux は？とかいろいろ難しい点もありますが。

高速な描画のために

- SDL::Surface#display_format を使う

サーフェスの形式を変更することで高速な描画が可能になります。形式は沢山あるんですが、どれが高速なのかはSDLが判定します。これを使うことが高速化の重要な点です。

高速な描画のために

- blit と fill_rect のみを使う
- アルファブレンドはできるかぎり使わない
- 回転や拡大縮小は描画時にしない

高速描画が可能なのは

- SDL::Surface.blit
- SDL::Surface#put
- SDL::Surface#fill_rect

のみ。回転などは暇なとき (初期化時など) にあらかじめ変換しておこう。

Ruby/SDL の過去と現在

- Ruby/SDL 本体について
- Ruby/SDL 周辺について

Ruby/SDL の過去

- 2001/1 ごろ 開発開始
- 2001/2/26 0.1 公開
  2D 描画、入力、音声出力、同じようなことをしている人がいた
- 2001/3 0.3 公開
  TTF のレンダリング, BMP 以外の画像を読み込み
- 2001/3 RUDL の存在が明らかに

Ruby/SDL の過去

- 2001/4 0.4 公開
  英語ドキュメントが付いた英語ページも作った春休みの間に書いた
- 2001/4 0.5 公開
  OpenGL 対応色の扱いを拡張 Event2 ができる
- 2001/4 FreeBSD の ports になった
- 2001/5 0.6 公開
  このあたりから Windows 版バイナリも配布

Ruby/SDL の過去

- 2001/10 Vine と Debian のパッケージに
- 2001/10 0.7 公開
  MPEG が (一応) 再生できるように
- 2002/2 本がでた
- 2002/5 0.8 公開
  SDLSKK 対応
- 2002/6 0.8.1
- 2002/10 0.8.2
- 2003/1 0.8.3
- 2003/8 0.9
  衝突判定が付いた

Ruby/SDLの過去

- 2003/10 0.9.1
- 2003/11 NF(Down!!)
- 2003/12 OS Xで動くように
- 2004/3 0.9.2
- 2004/7 0.9.3
  フォント関連追加(BMPFont, Kanji)
- 2004/12 0.9.4
- 2005/6 0.9.5

Ruby/SDL の過去

- 2005/10 1.0.0 公開!
  仕様を固定
- 2005/11 NF(Apple)
- 2006/4 1.1.0 リファレンスを大改訂
- 2006/9 1.2.0
- 2006/11 NF(熊カレー)
- 2007/2 1.3.0
- 2007/6 Ruby 会議 2007 で発表
- 2007/7 Ruby ではじめるゲームプログラミング
- 2007/9 1.3.1

Ruby/SDLの過去

- 2008/2 開発者が増えた→ Windows版メンテナ - サイロスさん
- 2008/4 2.0.1
  メジャーバージョンアップ! 英語ドキュメントを大改訂
- 2009/3 2.1.0 公開

Ruby/SDL 2.0で何が変わった？

- APIを1.Xから変更
  - よりRubyらしいAPIに
  - 以前のも残してあるのでほぼ後方互換
- リソースの明示的開放が可能に
  - 以前はGC頼りだった
  - 実は2.Xの最大の特長
- Ruby 1.9対応
  - m17nとかthread関連とか
  - SDL::Event.waitでスレッドが切り替わり可能に

Ruby/SDL の現在

- 最初のリリースから 8 年以上
- SDL の薄いラッパ
  - SDL でできることはだいたいできることを目指した
  - 低レベルに徹している
- 充実したリファレンス
- 使った人が結構多く、わりと Web 上に文章がある

Ruby/SDL 周辺について

- ラッパライブラリ
  - Miyako
  - MyGame
  - BABY
- 同時利用可能
  - ruby-opengl
  - Riko
- 競争相手
  - RUDL
  - rubygame
  - StarRuby
  - DXRuby

Miyako

サイロスさん作の Ruby/SDL ラッパライブラリ。けっこう規模が大きい。
いくつかのフレームワークの集合体、というような作りになっている。
http://www.twin.ne.jp/ cyross/Miyako/

MyGame

書籍「Rubyではじめるゲームプログラミング」を書くためにつくられたRuby/SDLのラッパライブラリ。
わりと薄めで小さなラッパライブラリ。自分でラッパを作りたいときの参考にもできるかな。
http://dgames.jp/ja/projects/mygame/

BABY

先日の Ruby 会議 2009 で発表があったもの。近日公開、らしいです。
昔の (N88-)BASIC あたりを参考にしているらしい。つまり線を書いたり円を書いたりを重視しているよう。

ruby-opengl

以前は OpenGL Interface と呼ばれていたもの。
rubyforge に移動して開発再開。
そこそこ活動しているぽい。最近は OpenGL 3.0 対応とかもしている。基本的には OpenGL の薄いラッパ。
GLUT なども含んでいる。
http://ruby-opengl.rubyforge.org/

Riko

OpenGL のラッパ。OpenGL のわりと新しい機能を使えることを (vertex_program, fragment_program) 主眼においている。ruby-opengl よりは Ruby っぽいかな。
http://www.kumaryu.net/?(Ruby)+Riko

RUDL

Ruby/SDL alternative としてはかなり古い。2005 年ごろに開発が止まっている？
http://rudl.sourceforge.net/

rubygame

RUDL よりは新しい。わりとふつう。Sprite 関連が特徴的かな。http://rubygame.org/

StarRuby

2007年くらいから作られている。画像の取り扱いをTextureという概念でまとめているのが特徴。
http://www.starruby.info/

DXRuby

新しいライブラリで、本格的に動きはじめたのは今年に入ってから。バックエンドがDirectXなのでWindowsでしか動かない。最近かなり活発に開発している。
http://dxruby.sourceforge.jp/

その他

http://ruby-game-dev.org/wiki/ にいろいろ情報がある

Ruby/SDL の未来

- 次回リリースは 8 月くらい
- 地味なバージョンアップは続く
- 半年に 1 回くらいはバージョンアップしたい
- 派手な変更は年単位ではなさそう
- とくに変更することもなくなってきた
- おそらく枯れる方向で

Ruby/SDL の未来

- 音声まわりがいまいちなんでなんとかしたい気もする
- OpenGL 使ってライブラリを書くとアルファブレンドとか回転拡大縮小も高速化!
  - 3D で 2D を作るノウハウは世の中にはあるはず
  - だれかやってくれないかな
- SDL 1.3 がリリースされたら大変更になるかも

まとめ

- Ruby/SDL を使えばRubyでゲームが作れる
- 継続は力
- Ruby/SDL 自体はそんなに劇的な変更はなさそうかな