Ruby for one day game programming camp for beginners

Ippei Obayashi

KMC/WPI-AIMR Tohoku Univ.

Dec. 12, 2015

RubyKaigi 2015

Thanks to KMC members,
especially: spi8823, hideya, dis, jf, and seikichi

Summary

KMC, a computer circle in Kyoto University, holds one-day game programming camp for newcomers to get familiar with KMC, have an experience of game programming, and see the fun of programming. Ruby is used for the event since Ruby has some advantages to learn program. From the experience, we can find a hint to teach programming beginners.

Summary

KMC, a computer circle in Kyoto University, holds one-day game programming camp for newcomers to get familiar with KMC, have an experience of game programming, and see the fun of programming. Ruby is used for the event since Ruby has some advantages to learn program. From the experience, we can find a hint to teach programming beginners.

This talk is a case study, and has no clear story line.

Self introduction

- Ippei Obayashi
- A maintainer of rurema
- The developer of Ruby/SDL, Ruby/SDL2, rrse, and some numerical computation libraries for Ruby
- An applied mathematician belonging to WPI-AIMR, Tohoku University
    - Assistant professor
    - Now I'm mainly working on the collaboration of topological data analysis and material science
    - I'm also working on dynamical systems and the application to locomotion models
    - Now I write python codes more than Ruby
- I lived in Kyoto, and participated in RubyKansai

Interests:

- Programming languages
  - I mainly use: C, C++, Ruby, Python
  - I have Written codes more than 1000 lines: OCaml, D, Haskell, elisp, scheme, shell script, data flow C
  - I have an experience: JavaScript (coffeescript), golang, julia, vala, lua, Arduino (C++?)
  - I want to learn: erlang (or elixir?)
- Blame Git UI
- Rogue-like games
- Stationery
- Beer, wine and sake
- `https://github.com/ohai`
  `https://bitbucket.com/ohai`

About KMC

KMC, Kyoto university Microcomputer Club, is a computer circle in Kyoto University found in 1977. More than 50 undergraduate and graduate students do various activities with computers such as

- Programming
  - Game programming
  - Programming contests (competitive programming)
  - Others (smartphone, web, programming language, AI, etc.)
- Computer graphics
- Computer music
- Network administration
- Studying computer science
- etc.

Ancient legacies:

- PLANET (A network system developed in early 80')
- 386BSD (98)
- Linux/98
- Anthy

In fact, main activities are chatting, gaming, and so on.

`https://www.kmc.gr.jp`

Kyoto University Micro Computer Club

About one day game programming camp

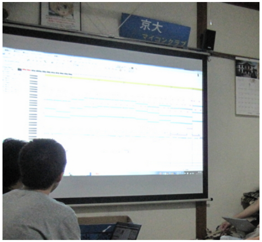
KMC hold the event called one day game programming camp (ラピッドゲームコーディング祭り) to give an opportunity to create a computer game from scratch for newcomers within one day.
The purposes are:

- Getting familiar with people in KMC
- Having an experience of programming
- Finding their interests
- Finding their potential

Participants try to complete games within one day.

Photo of the event (in 2013)

This event started since Kyoto city forbade BBQ at the riverside of Kamogawa in 2007.
In other words, KMC holds the event to invite new students to KMC.



Participants

- About ten participants in one day
- 80% participants have no programming experience
- 80% participants have no idea to create
  - 20% participants have some ideas
- Almost all participants prepare Windows laptop PCs.
  - Some participants do not have their own laptops. They use PCs owned by KMC.
  - No Mac, No Linux.

Environment

- At the club room
- A modified version of Ruby/SDL starter kit
  - `http://route477.net/w/RubySDLStarterKit.html` (by yhara)
  - Ruby 1.8.4
  - MyGame (by dan) and Ruby/SDL
- Sakura editor (locally modified for Ruby)

This environment is focused on easy installation.

- Unzip two files
- Edit “main.rb”
- Double click game.exe or debug.exe

Timetable

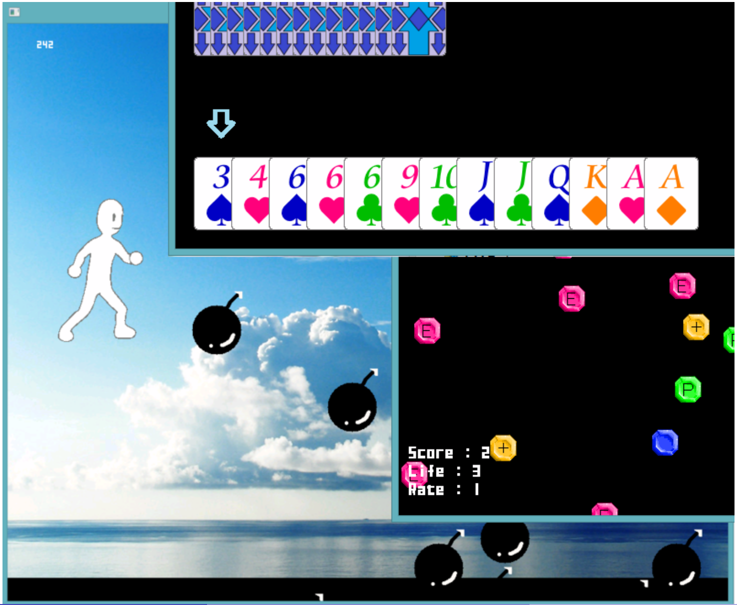
9:00	Opening, Assigning trainers
9:15	Installation and coding
12:00	Lunch break
17:00	Presentation of products
18:00	Closing

Style

- No lecture
- *One-on-one*
  - We prepare the same number of trainers as participants
  - They are often not familiar with Ruby
- No textbook or standard for “how to teach”
  - Each trainer (a older member in KMC) teach one’s student in one’s own way
  - All trainers forget how to teach in last year
- The textbook of 「たのしいRuby」 is used when they require a reference

These facts mean that educational contents depend on participants.

Products



- 80 $\sim$ 90% of participants complete something working within a day.
    - Although this is the first programming experiment
- Participants are well motivated
    - It is easy for motivated people to continue learning
- Some people try to create a new game after the event
    - Some people build a team to create a larger game
    - Not so many people continue to use Ruby

Why this event works out successfully?

One-on-one

- One trainer is assigned to one participant
- Many people in KMC believe that this way is the key of success
- Participants can determine the goal by consulting their trainer
- Trainers can adapt their trainees.
- Trainers have little experience of education
    - They learn how to teach by seeing each others way of teaching
- *Very flexible, and the way solve various kinds of poor preparation*

Effective but high-cost way of education. This way works since this is a one-day event.

- Some trainers use Struct, and others use classes.
- A trainer says that participants should focus on solving local problems, but another wants participants to learn the global structure of game programming
- Someone types his trainee's keyboard when he introduce a new Ruby syntax, and another uses a whiteboard.
- This way works since all trainers know how to write a video game very well
- *In KMC, all members are expected to learn anything by themselves, and they think that the event is a trigger of self-learning*

Deadline

- The goal is clear
- What to create is determined by the deadline
  - There are many problems that is impossible to solve
- Difficult to estimate what they can in a month, but easy to estimate in three hours

For more details:

- They can only make classical action games.
- Show objects, handling input and move objects, collision detection, scoring, intro and gameover (i.e. scene transition) and that's all
- Trainers said:
  - "Implementing the gravity is hard, Shoot'em up is easier."
  - "Collision detection is easy to implement, but it makes a program look more like game, and a programmer gets satisfaction"
  - "Scene transitions improve the look and feel of games, and scoring improve the fun of games"

Ruby

- In fact, in the first year, we use C and SDL
- However, it is difficult to install
- Ruby/SDL starter kit shows the solution
  - Unzip a file, and everything is OK
  - I built a modified version for the event in 2009
- Easy to learn, easy to teach

Movie here.

- They can write a program with a few elements
  - Variables, if, loop, and arrays
  - Not be afraid to use global variables
- If we use a statically-typed language, we need to teach more elements about the language and it is a hard work for one day.
- Less “magics” from the viewpoint of beginners
- One important fact is that trainers do not need to prepare very much

```
Zonbie = Struct.new(:x, :y, :dx, :dy, :count, :die)
Player = Struct.new(:x, :y)
  :
main_loop do
  :
 zonbies.each do |zonbie|
   balls.each do |ball|
    if detect_collision(zonbie, ball)
     zonbie.die=true
     effects.push(Effect.new(zonbie.x, zonbie.y, 50))

     score+=10
  #TransparentImage.render("image/nc82649.png",
  #:x => zonbie.x, :y => zonbie.y , :scale => 0.1)
     #ball.fade=true
  :
```

```
class Unit
        def initialize(img,atimg,x,y,hp,at,de,f)
                @img=img
                @atimg=atimg
                  :
                @f=f
                @mode=0
        end
        attr_accessor :img,:atimg,:x,:y,:speedx,:speedy,:hp,:at,:de,:f,:cost
        def render
                if(@hp>0)

   :
main_loop do
       :
       $enemy.each do |teki|
                teki.render()
        end
        hu.render()
        kyousya.render()
       :
```

- How about Unity?

- How about Unity?
  - Such a useful but complicated tool like IDE requires participants to learn much
  - Trainers also need to learn how to teach

- How about Unity?
  - Such a useful but complicated tool like IDE requires participants to learn much
  - Trainers also need to learn how to teach
- How about HSP?

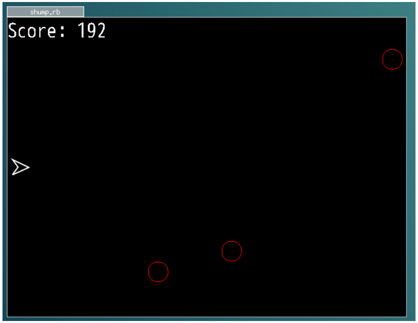
- How about Unity?
  - Such a useful but complicated tool like IDE requires participants to learn much
  - Trainers also need to learn how to teach
- How about HSP?
  - It is OK for our purpose. However we want to use more modern language
  - For further learning, modern languages such as Ruby are better
  - Almost all trainers is not familiar with languages with goto

- How about Unity?
  - Such a useful but complicated tool like IDE requires participants to learn much
  - Trainers also need to learn how to teach
- How about HSP?
  - It is OK for our purpose. However we want to use more modern language
  - For further learning, modern languages such as Ruby are better
  - Almost all trainers is not familiar with languages with goto
- How about Python, or other lightweight languages
  - Good, if we can prepare

In the future

- Probably KMC holds the event every year
- The environment should be updated
    - Ruby 1.8 is too old. Now rurema for 1.8 is purged.
- I'm working on the new version
    - `https://github.com/ohai/sdl2quick`
    - The new version will be released until next March
    - Small required knowledge about ruby
        - ⋆ For example, a programmer does not need to write a class
    - A "framework" is difficult to understand for beginners. No "inverse of control".
    - One-to-one correspondence from a programmer's hope to a function
    - Respect current version

```
def main
  put_image("shump/fly.png", x: $x-16, y: $y-16)
  $score += 1

  text("Score: #{$score}", x: 0, y: 0)


  if keypressed?("LEFT")
    $x -= 10
    $x = 0 if $x < 0
  end
  if keypressed?("RIGHT")
    $x += 10
    $x = 640 if $x > 640
  end
  if keypressed?("UP")
    $y -= 10
    $y = 0 if $y < 0
  end
  if keypressed?("DOWN")
    $y += 10
    $y = 480 if $y > 480
  end

  if rand(10) == 0
    $enemies.push(generate_enemy_randomly())
  end

  $enemies.each do |enemy|
    enemy.x += enemy.dx
    enemy.y += enemy.dy
    draw_circle(enemy.x, enemy.y, 16, RED)
  end

  $enemies.delete_if do |enemy|
```



```
Score: 192
```

Conclusion

Style and goal are important.

- The extreme style determines all
- The deadline plays an important role
- We can make great effort only within one day.

Lightweight languages such as Ruby are suitable for the education to beginners

- Non essential knowledge is not required
- Less magics

Unused words

By trainers:

- “Participants are very graceful, and listen to me carefully”
- “I hope participants learn how to arrange their code using function (method), but it is difficult for beginners”
- “Participants are confused when they write a loop. They sometimes write codes without understanding the correct execution order.”

By participants:

- “Any kinds of magics confuse me. My trainer teach me too fast.”
- “I have a chance to brush up when I try to get a driver license in Yamagata with my elders. The chance is very good for me. ”

Thank you for your attention.

補足

講師集めに関しては「お祭り」感を出すことで参加を促している側面があると思います．トレーニングといったら参加してくれるかどうかあやしい．

Q: 1 対 1 は講師集めが難しいが 2 対 1 とか 3 対 1 とかになったときにどうすれば良いか示唆はないか.

A: 難しい．講義の時間を作るとかしたほうがよいかもしれない．つまりカリキュラムをちゃんと考える必要があるのではないか．ただそうするとこの発表で挙げた知見は活かすのは難しい気がする.
最終的な目標を相談して決めるのではなく講師側から提供する手もある？

Q: Because of the one-on-one style, is there some trainers that make the progmram by themselves?

A: Yes, in some cases. We have no systematic answer to the problem and paying attention is only our answer. We have already known such problems from the previous experience, and KMC has KPT documents (wiki pages) for the event. Trainers are requested to read the pages before the event. “Brushing up” may be a good idea.