Ruby Basic

Colin Dao

Nội dung

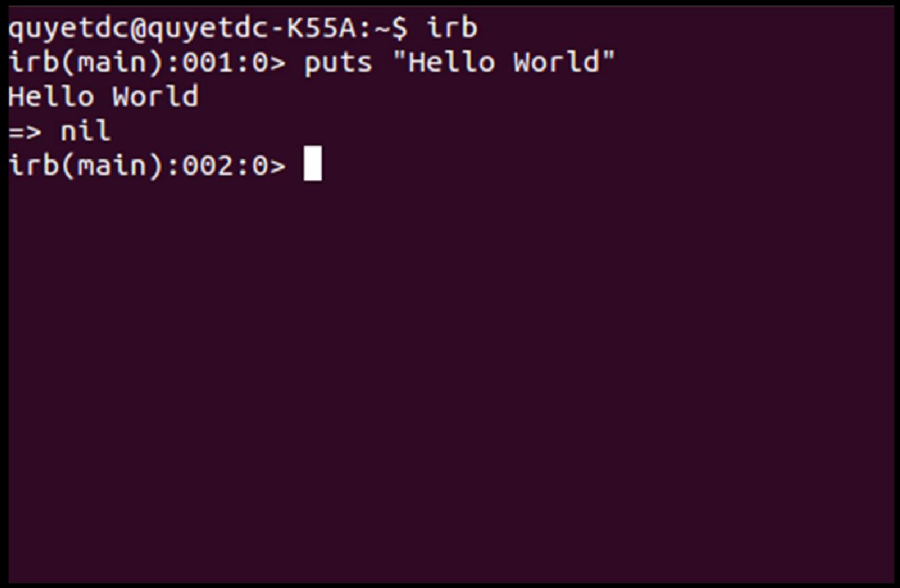
Interactive Ruby
Class
Attributes
Module

Variables
Constant
Loop statements
Conditional statements
Practices

Array
Hash
String
File
Exception
Date & Time
Practices

IRB

- Interactive console → Cho phép chúng ta lập trình Ruby trên console
- Trên terminal → $ irb



Class & Objects

- Ruby là ngôn ngữ lập trình hướng đối tượng (sẽ nói kĩ về sau)
- Mọi thứ trong Ruby đều là Đối Tượng (object) của một Lớp (Class)
- Cú pháp

```
## Class: ExampleClass.
## Created by quyetdc, June - 2014
class ExampleClass
        @@class_variable = 1          # => khai báo biến dùng cho toàn class
        def initialize(x)             # => Hàm khởi tạo ( có thể không có )
    # code                            # => Được gọi khi tạo object:
    #                                         ExampleClass.new(x, y)
 end

 def function                         # Member function
    # ...
 end
end
```

Class & Attributes

```
class ExampleClass2
    attr_reader   :only_readable_attribute
    attr_writer   :only_writable_attribute
    attr_accessor :readable_writable_attribute

    ## explanation of attr_writer
    ## calling ExampleClass2.new.only_writable_attribute
    def only_writable_attribute=(only_writable_attribute)
        @only_writable_attribute = only_writable_attribute
    end

    ## explanation of attr_reader
    ## calling ExampleClass2.new.only_readable_attribute
    def only_readable_attribute
       @only_readable_attribute
    end
end
```

Class Example

```
class  ClosedRange
   attr_accessor  :lower_end_point,  :upper_end_point,  :is_valid,  :error_message

   def  initialize(lower_point = 0, upper_point = 0)
      if  lower_end_point  <  upper_end_point
         self.lower_end_point  =  lower_point
         self.upper_end_point  =  upper_point
         is_valid = true
      else
         is_valid = false
         error_message = "invalid input"
      end
    end

    def  select_lower_end_point
      if  is_valid
         return   result
       else
          return   error_message
       end
    end
end
```

Module

Module là một cách để gộp các hàm, các constant.

Hiểu module giống như thư viện đóng gói các functions, còn Class là nói tới đối tượng

Module Có thể include các Module

Class có thể include các Module → khái niệm mixin
→ Sử dụng Module khi các class có chung behaviors

```ruby
module A
  def a
    puts "a"
  end
end

class Sample
  include A

  def s
    puts "s"
  end
end

samp = Sample.new
samp.a  ## => a
samp.s  ## => s
```

Variable

Biến (Variable): vùng bộ nhớ để lưu trữ dữ liệu

Một số loại biến chính trong Ruby

- Biến toàn cục (global variable)
- Biến đối tượng (instance variable)
- Biến lớp (class variable)
- Biến địa phương (local variable)

Global Variable

- Được bắt đầu với dấu $
- Không hay dùng vì gây khó hiểu
- Có thể sử dụng ở nhiều class
- Nếu biến không được khởi tạo trước sẽ có giá trị là *nil*

```
$global_variable = 10
class Class1
    def print_global
        puts "Global variable in Class1 is #$global_variable"
    end

    def increase_global_variable
        $global_variable += 1
     end
end

class Class2
     def print_global
        puts "Global variable in Class2 is #$global_variable"
     end
end

class1obj = Class1.new
class1obj.print_global ## => Global variable in Class1 is 10
class1obj.increase_global_variable
class2obj = Class2.new
class2obj.print_global ## => Global variable in Class1 is 11
```

Instance Variable

- Được bắt đầu với dấu @
- Có thể sử dụng để giữa các methods trong một class
- Nếu biến không được khởi tạo trước sẽ có giá trị là *nil*

```
class Customer
  def initialize(name, email)
    @customer_name = name
    @customer_email = email
  end
  def print_customer_info
    puts "Customer name: #@customer_name"
    puts "Customer email: #@customer_email"
  end
end

customer1 = Customer.new("colin", "colindao@techmaster.vn")
customer1.print_customer_info
## => Customer name colin
## => Customer email colindao@techmaster.vn
```

Local Variable

- Được viết bởi các chữ in thường và ngăn cách các chữ bởi dấu _
- Chỉ dùng được trong một hàm, hoặc khối điều kiện, lặp ...
- Chỉ cần khởi tạo, không cần khai báo
- Nếu tham chiếu tới biến chưa được khởi tạo trước đó → error

```ruby
class Class1
   def demo_local_variable(input_local_variable)
      begin
         puts "local variable as input #{input_local_variable}"
         local_variable = 100
         puts "local variable #{100}"

         [1, 2].each do |i|
            in_block_local_variable = i
            puts "in_block_local_variable #{in_block_local_variable}"
          end

         puts "call in_block_local_variable outside --> "
         puts "#{in_block_local_variable}"
      rescue Exception => e
         puts e.message
      end
   end
end

class1_obj = Class1.new
class1_obj.demo_local_variable(10)
```

Constant

- Qui ước viết bằng chữ in hoa toàn bộ
- Nếu khai báo trong Class hay Module thì chỉ dùng được trong Class hay Module đó
- Nếu khai báo ngoài Class thì dùng toàn cục
- Không khai báo trong functions
- Nếu gán lại giá trị cho hằng số → warning
- Nếu gọi tới 1 constant chưa khai báo → error

```
class ClosedRange
    ERROR_CODE_INVALID_RANGE = 1
    ERROR_CODE_NOT_NUMBER = 0
    attr_accessor :lower_end_point, :upper_end_point,
                  :is_valid, :error_code
    def initialize(lower_point = 0, upper_point = 0)
        if !lower_point.is_a?(Fixnum)
            self.is_valid = false
            self.error_code = ERROR_CODE_NOT_NUMBER
                              ## Không sử dụng magic number 0, 1 vì sẽ gây khó
hiểu
            return
        end

        if lower_end_point < upper_end_point    ## Lay gia tri khong can dung self
            self.lower_end_point = lower_point      ## Khi gan gia tri can dung self
            self.upper_end_point = upper_point
            self.is_valid = true
        else
            self.is_valid = false
            self.error_code = ERROR_CODE_INVALID_RANGE
        end
    end
end
```

Loop statements

```ruby
## Loop statements
## Created by Colin Dao, June - 2014

NUM = 5

## WHILE LOOP"#{
var1 = 0
while var1 < NUM do
   ## this block of code will be executed while conditions are true
   puts "#{var1}"
   var1 += 1
end

## UNTIL LOOP
var2 = 0
until var2 == 5 do
   ## this block of code will be executed while conditions are false
   puts "#{var2}"
   var2 += 1
end
```

```ruby
## FOR LOOP
for i in (0..5) do                    ## 0..5 == [0, 1, 2, 3, 4, 5]
   puts "#{i}"
end

## EACH LOOP
(0..5).each do |i|
   puts "#{i}"
end

## BREAK => Kết thúc khối loop
(0..5).each do |i|
   break if i  == 2
   puts "#{i}"
end

## NEXT =>  chuyển sang vòng lặp tiếp theo
(0..5).each do |i|
   next if i  == 2
   puts "#{i}"
end
```

Conditional statements

```
## Conditinal statements
## Created by quyetdc, June - 2014

if condition1
  ## Thực hiện đoạn code nếu condition1 là true, not nil
elsif condition1                    ## we use elsif, not else if
  ## Thực hiện đoạn code nếu condition 1 là false, hoặc nil và
  ##                                  condition 2 là true, not nil
else
  ## Thực hiện đoạn code nếu condition 1 false, hoặc nil và
  ##                                  condition 2 là false hoặc nil
end

## Nếu đoạn code thực hiện đơn giản
code if condition
```

```
unless condition          ## Sử dụng unless thay cho if not
  # code... nếu condition false hoặc nil
else
  # code... nếu condition true
end

code unless condition

case object
when expression1
  ## ..code: Khi expression1 true nếu object là nil
  ##         khi object = expression 1
when expression2
else
end
```

Practices

- Kiểm tra số là chẵn hay lẻ
- Tính trung bình cộng của n số tự nhiên
- Viết chương trình in ra bảng cửu chương
- Chương trình in ngược xâu kí tự
- Tìm tất hoán vị m số của n số tự nhiên
- Kiểm tra số có là số lũy thừa của 2 không
- Kiểm tra số có là số nguyên tố không
- Sắp xếp n số tự nhiên (theo chiều tăng dần, hoặc giảm dần)

Array

- Là tập hợp các đối tượng (số, chuỗi kí tự, mảng ...) có thứ tự
- Bắt đầu từ 0, 1, 2 ...
- Nếu gọi phần tử từ cuối có thể gọi qua index -1, -2, -3 ...
- www.ruby-doc.org/core-2.1.2/Array.html

```ruby
## Array.
## Created by quyetdc, June - 2014

## Khởi tạo
a = Array.new
a = []

1..5 == [1, 2, 3, 4, 5]

a = [1, 2, 3, 4, 5]
a[0]                    ##  => 1
a[1]                    ##  => 2
a[-1]                   ##  => 5
a[-2]                   ##  => 4

a.push(1)               ## => [1, 2, 3, ,4, 5, 1]
a.index(2)              ## => 3
a.include? 2            ## => true

a.each do |i|
  puts "#{i}" if i == 3
end
```

Array

```
a = [1, 2, 3, 4, 5, 1]
a.uniq                      ##  => [1, 2, 3, 4, 5]
                            ##  a => [1, 2, 3, 4, 5, 1]

a.uniq!                     ##  a => [1, 2, 3, 4, 5]

a.map {|i| i += 1}          ## you can use equivalent 'collect' command instead
                            ## create a copy array of a then do expression in block {}
                            ## ==> [2, 3, 4, 5, 6]
                            ## a ==> [1, 2, 3, 4, 5]

a.map! {|i| i += 1}         ## you can use equivalent 'collect!' command instead
                            ## do expression in block {} on each element of a
                            ## a ==> [2, 3, 4, 5, 6]

b = [1, 2, 8, 9]
a & b                       ## => [1, 2]
a | b                       ## => [1, 2, 3, 4, 5, 8, 9]
```

Hash

```
## Hash structure: { key1: value1, key2: value2, key3: value3 }

h = Hash.new                                    ## khởi tạo
h[:key] = "value"                               ## Gán giá trị, sử dụng symbol hoặc string
h                                               ## {:key => "value"}
h[:key]                                         ## "value"
h.has_key?(:key)                                ## true
h.has_value?("value")                           ## true
h.keys                                          ## trả về array các key của hash => [:key]
h.values                                        ## trả về array các value của hash => ["value"]

h2 = Hash.new(100)                              ## Khởi tạo với giá trị mặc định 100 cho key bất kì
h2[:key]                                        ## 100

h3 = { "a" => 100, "b" => 200, "c" => 300 }
h3.select {|k,v| k > "a"}                       #=> {"b" => 200, "c" => 300}

h3.merge(h)                                     ## trả về 1 hash khác là gộp của 2 hash
                                                ## { "a" => 100, "b" => 200, "c" => 300, :key => "value" }
h3.merge!(h)                                    ## Thay đổi hash
                                                ## h3 = { "a" => 100, "b" => 200, "c" => 300, :key =>
"value" }
```

More: http://www.tutorialspoint.com/ruby/ruby_hashes.htm

String

- http://www.ruby-doc.org/core-2.1.1/String.html
- Truyền giá trị của biến vào string: “puts #{x}”
- Array of string: %w(one two three) == [“one”, “two”, “three”]

File

```
puts "Enter a value: "
val = gets                                  ## Lấy giá trị mà người dùng nhập
puts val

File.open("filename", "mode") do |aFile|    ## Tạo đối tượng aFile -> phụ thuộc vào mode để có thể đọc, ghi
   #... process the file object: aFile       ##                                                  mode → r,
r+, w, w+, a, a+
end

aFile = File.new("input.txt", "r")
if aFile
  puts aFile.sysread(20)                    ##  Hiển thị 20 ký tự đầu tiên của file
else
  puts "Unable to open file!"
end

aFile = File.new("input.txt", "w+")         ## đọc - ghi, nếu file không tồn tại -> tạo  mới
aFile.syswrite("ABCDEF")                    ##  Ghi ra file

arr = IO.readlines("input.txt")             ## Đọc từng dòng của file và gán thành phần tử của mảng kết quả

File.file?( "text.txt" )                     ## true or false
```

Exception

```
## File Processing
## Created by quyetdc, June - 2014

begin
    file_name = "/unexistant_file"
    file = File.open(file_name, 'r')
    puts "File opened successfully" if file
rescue Exception => e
    puts e.message                                  ## Hiển thị exception
    file_name = "existant_file"
    retry                                           ## Thực hiện lại khối begin từ đầu
ensure
    ## Đoạn code trong ensure luôn được thực hiện dù có exception xảy ra không
    puts "End process"
end

begin
    raise 'A test exception.'
rescue Exception => e
    puts e.message                                  ## A test exception.
    puts e.backtrace.inspect                                    ## filename:line
end
```

Date & Time

```
require 'time'

time = Time.now                          ## Giờ hiện tại
time.year
time.month
time.day
time.yday                                ## Ngày thứ bn của năm

Time.parse('20070131')                   ## Đổi string to time

time.strftime("%Y-%m-%d %H:%M:%S")       ## format string time

time.to_i                                ## chuyển giờ sang số integer của giây
                                            (tính từ 0:00:00 0, 0, 0)
Time.at(time.to_i)                       ## Tính giờ từ số integer

time - 10                                ## Time at 10 seconds ago
```

Practices

- Đếm số chữ số của số tự nhiên N
- Tìm các phần tử chung của 2 mảng mà không sử dụng method có sẵn của Ruby
- Viết hàm kiểm tra xem mảng *small* có là tập con của mảng *big* hay không
- Xóa bỏ các kí tự trùng lặp trong string
- Viết chương trình đọc *input file* và in ra *output file* số lần xuất hiện của từng loại kí tự sử dụng Hash