Ruby元编程之方法

技术管理部 张哲

review

- A module is a collection of methods and constants
- A class is a module with three additional methods - new, allocate, superclass. Each class is an instance of class Class
- An object is a bunch of instance variables, plus a link to a class
- Classes are objects, Object is a class

code review

about eval

```
gary = "gary"

gary.instance_eval do
  def to_s
    "Gary Strong2"
  end
end
p gary.to_s

def gary.to_s
  "Gary Strong2"
end
p gary.to_s
```

about instance variable

```
class MyClass
  SOME_CONST = "alpha"
  @var = "beta"
end

p MyClass.const_get("SOME_CONST")
p MyClass.instance_variable_get("@var")
```

about class variable

... ...

About all methods

```
class MyClass
  def test1
  end

  def test2
  end
end

p MyClass.instance_methods#(false)
p MyClass.instance_methods(false)
p Class.instance_methods(false)
```

what is metaprogramming

Metaprogramming is
writing code that writes code


225
150
75
75
50
50
25
50
100
125
125
Menu
Last Stand - 4 flags complet
GAMERSKY

method

- send
- define_method
- method_missing

send(dynamic Dispatch)

```ruby
class MyClass
  def test_by(name)
    "test by #{name}"
  end
end

p MyClass.new.test_by("me")
p MyClass.new.send(:test_by, "me")
```

send(dynamic Dispatch)

- big power
- use symbol(string) as method_name
- symbol vs string

define_method(dynamic method)

```
class MyClass
  def peek
    p "I am peeking"
  end

  def self.prepare
    define_method :peek2 do
      p "I am peeking too"
    end
  end

  prepare
end

MyClass.new.peek
MyClass.new.peek2
```

define_method(dynamic method)

- define method in method - code that writes code
- closure

method_missing(ghost method)

```
class MyClass
  def method_missing(method, *args)
    puts "MyClass##{method} with params #{args.inspect} does not
exsit!"
  end

  def self.method_missing(method, *args)
    puts "MyClass.#{method} does not exsit!"
  end
end

MyClass.new.test1(100, 200)
MyClass.test1
```

method_missing(ghost method)

- use respond_to other than methods
- bug

a bad example

```
class Roulette
  def method_missing(name, *args)
    person = name.to_s.capitalize
    3.times do
      number = rand(10) + 1
      puts "#{number} ...}"
    end
    "#{person} got a #{number}"
  end
end

number_of = Roulette.new
puts number_of.bob
puts number_of.gary
```

another bad example

```ruby
class StrongBoy
  def display
    puts "show some muscle"
  end

  # def method_missing(name, *args)
  #     if name == "display"
  #       puts "show some muscle"
  #     end
  #   end
end

StrongBoy.new.display
```

remove_method
undef_method

And
Blank Slates

Prepare tool

```ruby
require 'ostruct'
computer = OpenStruct.new
computer.cpu = "386"
computer.price = 100
p computer
#(method_missing)
```

```ruby
require 'benchmark'
def benchmark
  Benchmark.bm do |x|
    x.report do
      100.times do
        yield
      end
    end
  end
end
```

Example

```
class ComputerInfo
  def initialize(data_source)
    @data_source = data_source
  end

  def cpu
    info = @data_source.cpu_info
    price = @data_source.cpu_price
    "Cpu: #{info} ($#{price})"
  end

  def keyboard
    info = @data_source.keyboard_info
    price = @data_source.keyboard_price
    "Keyboard: #{info} ($#{price})"
  end

  def mouse
    info = @data_source.mouse_info
    price = @data_source.mouse_price
    "Mouse: #{info} ($#{price})"
  end
end
```

small tip
do not use class variable except ...

Thanks