

Langage Ruby & microframework Sinatra

Yann Klis, 2016

Ruby

inventé

Par Yukihiro Matsumoto aka Matz

En 1995

Décollage à partir de 2004

Grâce à Ruby on Rails



multiplatforme

Linux, MacOS X, Windows, etc

interprété

$$\begin{aligned} &> 1 + 1 \\ &\Rightarrow 2 \end{aligned}$$

gestion de la mémoire

Garbage Collector

orienté objet pur

```
> 1.class
=> Fixnum
> 1.class.ancestors
=> [Fixnum,
    Integer,
    Numeric,
    Comparable,
    Object,
    Kernel,
    BasicObject]
```


orienté objet pur

```
> 1 + 1
```

```
=> 2
```

```
> 1.send(:+, 1);
```

```
=> 2
```

```
> 1.send(:+, 1)
```

```
=> 2
```

```
> 1.send :+, 1
```

```
=> 2
```

orienté objet pur

```
> nil.class  
=> NilClass  
> nil.nil?  
=> true
```

orienté objet pur

```
> true.class  
=> TrueClass
```

dynamique

```
> 1.send(:+, 1);  
=> 2
```

```
a = "+"  
> 1.send(a.to_sym, 1)  
=> 2  
> 1.send :+, 1  
=> 2
```

dynamique

```
> ary = []  
=> []  
> ary.class.class_eval {  
  attr_accessor :total_entries  
}  
=> nil  
> ary.total_entries = 42  
=> 42  
> ary.total_entries  
=> 42
```

dynamique

```
class Printing
  def method_missing(m, *args)
    if (name = /^test_(.+)$/ .match(m.id2name))
      print name[1]
    end
  end
end
End
```

```
> printing = Printing.new
> printing.test_hello
=> "hello"
> printing.test_pipo
=> "pipo"
> printing.test_pipo("a", 1, 3)
=> "pipo"
```

block

lambda, fonction anonyme, etc.

```
toast = Proc.new {  
  puts 'Bravo !!!'  
}  
toast.call
```


block

lambda, fonction anonyme, etc.

```
toast = Proc.new do  
  puts 'Bravo !!!'  
end  
toast.call
```

block

lambda, fonction anonyme, etc.

```
toast = Proc.new do |var|  
  puts var  
end  
toast.call('Bravo !!!')
```

block

lambda, fonction anonyme, etc.

```
toast = Proc.new { |var|  
  puts var  
}  
toast.call('Bravo !!!')
```

block

lambda, fonction anonyme, etc.

```
> ["a", "b", "c"].each{|elt|  
  puts elt
```

```
}
```

```
a
```

```
b
```

```
c
```

block

lambda, fonction anonyme, etc.

```
> ["a", "b", "c"].each_with_index{|elt, index|  
  puts index  
}  
0  
1  
2
```

block

lambda, fonction anonyme, etc.

```
> ["a", "b", "c"].map{|element|  
  element.upcase  
}  
=> ["A", "B", "C"]
```

```
> ary = ["a", "b", "c"]
["a", "b", "c"]
> ary.map{|element|
  element.upcase
}
=> ["A", "B", "C"]
> ary
["a", "b", "c"]
> ary.map !{|element|
  element.upcase
}
=> ["A", "B", "C"]
> ary
["A", "B", "C"]
```


block

lambda, fonction anonyme, etc.

```
>(5..10).inject {|sum, n| sum + n }  
=> 45
```

```
>(5..10).inject(1) {|sum, n| sum * n }  
=> 151200
```

lisibilité

```
> 5.times { print "0delay!" }
```

lisibilité

```
> 5.times { puts "0delay!" }  
0delay!  
0delay!  
0delay!  
0delay!  
0delay!
```

lisibilité

```
> exit if "restaurant".include?("aura")
```

lisibilité

```
> exit if "restaurant".include?("aura")  
#
```

lisibilité

```
> ['toast', 'cheese', 'wine'].each{|food|  
  puts food.capitalize  
}
```

lisibilité

```
> ['toast', 'cheese', 'wine'].each{|food|  
  puts food.capitalize  
}  
Toast  
Cheese  
Wine
```


lisibilité

```
> ['toast', 'cheese', 'wine'].each{|food|  
  puts food.upcase  
}
```

lisibilité

```
> ['toast', 'cheese', 'wine'].each{|food|  
  puts food.upcase  
}  
TOAST  
CHEESE  
WINE
```

typage ?

Duck typing !



```
class Duck
  def quack
    puts "Quaaaaaack!"
  end

  def feathers
    puts "The duck has white and gray feathers."
  end
end

class Person
  def quack
    puts "The person imitates a duck."
  end

  def feathers
    puts "The person takes a feather from the ground and shows it."
  end
end

def in_the_forest(duck)
  duck.quack
  duck.feathers
end

def game
  donald = Duck.new
  john = Person.new
  in_the_forest donald
  in_the_forest john
end

game
```

```
module ToString
  def to_s
    "Quaaaaaack!"
  end
end

class Camion
  include ToString
end

class Voiture
  include ToString
end

puts Camion.new.to_s
puts Voiture.new.to_s
```

redéfinition dynamique

```
# fixnum.rb
class Fixnum
  def +(other)
    self * other
  end
end

puts 2+3
```

Ruby ? Quel Ruby ?

1.9.x – 1.9.3	jan 2009
2.0.x – 2.0.0	feb 2013
2.1.x – 2.1.0	dec 2013
2.2.x – 2.2.0	dec 2014
2.3.x – 2.3.0	dec 2015

Ruby ? Quel Ruby ?

2.0.0 MRI

2.3.0 MRI

rbx 2.0.0

JRuby 1.7.4

rbenv

rvm

```
# rbenv list
2.1.2
2.2.4
* 2.3.0 (set by /home/yannski/.rbenv/version)
```

rbenv

```
# rbenv install jruby-1.7.4  
# rbenv local jruby-1.7.4
```

bundler & Gemfile

```
source 'https://rubygems.org'
```

```
gem 'rails', '4.0.0'
```

```
gem 'rails-i18n'
```

```
gem 'sass-rails', '~> 4.0.0'
```

```
gem 'haml-rails', '>= 4.0.0'
```

```
gem 'mongoid', github: 'mongoid/mongoid'
```

Communauté Ruby

rubygems.org

repository centralisé des gems



a gem
There's ~~an App~~
For That

<http://ruby-doc.org/>

<http://apidock.com/ruby>

Sinatra

microframework

```
require 'sinatra'

get '/hi' do
  "Hello World!"
end
```

```
$ gem install sinatra
$ ruby hi.rb
== Sinatra has taken the stage ...
>> Listening on 0.0.0.0:4567
```

```
get '/' do
  .. show something ..
end

post '/' do
  .. create something ..
end

put '/' do
  .. replace something ..
end

patch '/' do
  .. modify something ..
end

delete '/' do
  .. annihilate something ..
end

options '/' do
  .. appease something ..
end

link '/' do
  .. affiliate something ..
end

unlink '/' do
  .. separate something ..
end
```