SACTA 2016 - UNIPAMPA

Introdução a Linguagem de Programação

Ruby

O melhor amigo do programador

Quarta, 11 de maio de 2016

Giulliano Paz
Ciência da Computação - UNIPAMPA

Roteiro

- História
- Sobre o Ruby
- Quem usa Ruby?
- Instalando Interpretador
- Utilizando Interpretador
- Variáveis e Constantes
- Tipos
- Saída e Entrada de Dados
- Strings
- Symbols
- Arrays
- Hashes
- Estruturas de Controle
- Arquivos
- Métodos
- Classes
- Mais de Ruby
- Bibliografia

História

- Desenvolvida por Yukihiro “Matz” Matsumoto(Japão, 1995)
- Unir suas linguagens favoritas(Perl, Python, Smalltalk, Eiffel, Ada e Lisp)
- Objetivo é Ruby ser *Natural* e não *Simples*
- Ruby era a pedra zodiacal de um colega de Matz

> “O Ruby é simples na aparência, mas muito complexo no interior, tal como o corpo humano”
>
> Yukihiro “Matz” Matsumoto

Sobre a Ruby

- Linguagem de Script, Interpretada, fortemente Tipada e Dinâmica
- Totalmente orientada a objetos(Tudo é objeto)
- Escrita em C
- Programação Funcional, Orientada a Objetos, Imperativa, Reflexiva
- Muito Flexível

Quem usa Ruby?

- **Nasa:** para realizar simulações;
- **Motorola:** para simulação, geração de cenários e tratamento dados;
- **Google SketchUp:** para modelagem 3D;
- **Siemens:** para implementar a parte do controle reativo de um robô no projeto MORPHA;
- **NuBank, GitHub, Globo.com, Abril, Groupon, Ask.fm, SlideShare...**

Instalando o Interpretador

```
Abra o Terminal do Ubuntu e digite

~$ sudo apt-get install ruby-full
```

⇨ # Multiparadigma: funciona em diversos Sistemas Operacionais

Utilizando o Interpretador

```
Abra o Terminal do  Ubuntu e digite
```

```
~$ irb
```

ou

```
~$ ruby codigo.rb
```

Variáveis e Constantes

- **Primeira letra maiúscula:** Constante
- **$** : Variável global
- **@** : Variável de instância
- **@@** : Variável de classe

Tipos

```
$> 3.14.class                                                    #Float
$> 3.class                                                       #Fixnum
$> “string”.class                                                #String
$> :symbol.class                                                 #Symbol
$> [1, :sy, ”*--*”, 3.14].class                                  #Array
$> {curso: “CC”, :universidade => “UNIPAMPA”}.class              #Hash
$> true.class                                                    #TrueClass
$> false.class                                                   #FalseClass
```

Saída e Entrada de Dados

```
puts "Digite algo: "              # imprime e quebra linha

print "Digite algo: "             # só imprime

p "Digite algo: "                 # chama método inspect

entrada = gets                    # “guarda” o que foi digitado na variável

puts “Você digitou #{entrada}”    # imprime o que o usuário digitou
```

⇨ # inspect permite “ver” o conteúdo de um array

Strings

```
mensagem = String.new “Hello, ”              # “Hello, ”

mensagem = “Hello, ”                         # “Hello, ”

mensagem << “World!!”

puts mensagem                                #Hello, World!!

mensagem = []

mensagem = “  oLá, tUdO bEm?\n”

puts mensagem.chomp.downcase.capitalize      #Olá, tudo bem?
```

Symbols

```ruby
mensagem = :string_imutavel

mensagem << “algo”

puts mensagem     #string_imutavel

mensagem = []

mensagem = :sTrinG_iMutAvEl2

puts mensagem.downcase.capitalize     #String_imutavel2
```

Arrays

```
$> arr = Array.new [1, “a”, “ruby”, 3.14]
$> arr = [1, “a”, “ruby”, 3.14]
$> arr[1]                    # “a”
$> arr[-3]                   # “a”
$> arr[100]                  # nil
$> arr.at(0)                 # 1
$> arr.first                 # 1
$> arr.last                  # 3.14
```

Arrays

```
$> arr = [1, “a”, “ruby”, 3.14]
$> arr.drop 2                    # [“ruby”, 3.14]
$> arr.take 2                    # [1, “a”]
$> arr.length                    # 4
$> arr.index(1)                  # “a”
$> arr.index 100                 # nil
$> arr.fench 100,”Não”           # 3.14
$> arr.delete “a”                # “a”
```

Hashes

```ruby
aluno = Hash.new                    # {}

aluno = {}                          # {}

aluno = {:nome => “Fulano”, curso: “CC”, matricula: 151161511}

aluno[:idade] = 20

puts aluno          # {:nome => “Fulano”, :curso => “CC”, :matricula => 151161511, :idade => 20}

aluno[“sobrenome”] = “de tal”

aluno[313135] = “id”

puts aluno              # {:nome => “Fulano”, :curso => “CC”, :matricula => 151161511,
                           :idade => 20, “sobrenome” => “de tal”, 313135 => “id”}
```

Hashes

```
aluno = {:nome => “Fulano”, curso: “CC”, matricula: 151161511}

aluno.has_key? :nome          #true

aluno.has_key? :idade         #false

aluno.has_value? “CC”         #true

aluno.has_value? 10           #false

aluno.key “CC”                #:curso

aluno.keys                    #[:nome, :curso, :matricula]

aluno.values                  #[“Fulano”, “CC”, 151161511]

aluno.invert                  #{“Fulano” => :nome, “CC” => :curso, 151161511 => :matricula}
```

Estruturas de Controle

```ruby
variavel = “voce” == “outros”          #false

variavel = 2+2 == 4                    #true

if variavel

      variavel = variavel.to_s.upcase
      puts “#{variavel} STORY”

end
```

⇨

```ruby
puts “TRUE STORY” if variavel == true
```

Estruturas de Controle

```
puts “Informe um número: ”

num = gets.chomp.to_i.abs

case num

when 0..100
      puts “Entre 0 e 100.”

when 100..1000
      puts “Entre 100 e 1000.”

else
      puts “Maior que 1000.”

end
```

Estruturas de Controle

```
dCC = [“A&P”, “ArqI”, “SO”, “PAA”]

for d in dCC                    # A&P
                                # ArqI
      puts d                    # SO
                                # PAA
end


for i in 1..10

      print i                   # 12345678910

end
```

```
next    #pula iteração do for

redo    #repete a ultima iteração

retry   #reinicia todo o loop
```

Estruturas de Controle

```ruby
i = 1
while i <= 10

      print i     # 12345678910
      i += 1

end
```

⟹

```ruby
print “loop infinito” while true
      #Imprime infinitamente
```

```ruby
i = 1
begin

      print i      # 12345678910
      i += 1

end while i <= 10
```

Estruturas de Controle

```ruby
lista = ["a", "b", "c", "d"]

lista.each do |letra|

      puts letra

end
```

```ruby
10.times{ puts “Vai imprimir 10 vezes” }
```

Arquivos

```
arq = File.new(“arquivo.txt”, “r”)

puts arq.read  #lê todo o arquivo

arq = File.new(“arquivo.txt”, “w+”)

arq.write “Sobrescreve arquivo”

arq = File.read(“arquivo.txt”)

arq = File.readlines(“arquivo.txt”)

puts File.exists?(“arquivo.txt”)   #true

puts File.size(“arquivo.txt”)
```

- **r** - Abre o arquivo para leitura;
- **w** - Abre o arquivo para escrita;
- **a** - Anexa ao final do arquivo, caso você queira escrever no final do arquivo;
- **r+** - Abre o arquivo para leitura e escrita;
- **w+** - Cria um arquivo vazio para leitura e escrita;
- **a+** - Abre o arquivo para leitura e anexação, ou seja, você pode ler qualquer parte do arquivo, mas só pode escrever no final do arquivo.

Métodos

```ruby
def soma a, b
      return a + b
end
puts “Resultado: #{soma 10, 90}”          #100


def ler_arquivo nome_arquivo
      return File.read(nome_arquivo)
end
puts “Conteúdo do arquivo: #{ler_arquivo(“arquivo.txt”)}”
```

Classes

```
class Arquivo

      def ler_arquivo(nome_arquivo)

            return File.read(nome_arquivo)

      end

      def mostrar_arquivo nome_arquivo

            return ler_arquivo nome_arquivo

      end
end

arq = Arquivo.new

puts arq.mostra_arquivo “arquivo.txt”    # Imprime todo o conteúdo do arquivo
```

Mais de Ruby

```
a ||= 10   # a = 10
a ||= 20   # a = 10
```

```
/rio/ =~ “sao paulo”      # nil
/paulo/ =~ “sao paulo”    # 4
```

```
24.div 2                  # 12
12.modulo 5               # 2
12.divmod 5               # [2,2]
12.quo 5                  # 2.4
24.eql?(12*2)             #igualdade
```

```
1.integer?                #true
1.zero?                   #false
(-1.0/0.0).finite?        #true
```

Bibliografia

- http://ruby-doc.org/core-1.9.3/index.html
- https://www.caelum.com.br/apostila-ruby-on-rails
- https://www.ruby-lang.org/pt/
- https://www.codecademy.com/pt-BR/learn/ruby
- http://ruby-doc.org/
- http://rubyonrails.org/

Obrigado

https://github.com/giullianopaz

giulliano94@gmail.com

Quarta, 11 de maio de 2016

Giulliano Paz
Ciência da Computação - UNIPAMPA