





Pablo Hildo

Desenvolvedor web e mobile;

Rubyista há 8 anos;

Pesquisador no LACE - Laboratório de
Computação e Engenharia.

 /pablohildo

 /pablohildo

 /pablohildo



Natan Moura

Desenvolvedor web;

Pesquisador no LACE - Laboratório de
Computação e Engenharia.

/ntsmoura



/ntsmoura



/ntsmoura





História

- Criada em 1995 por **Yukihiro Matsumoto** (Mattz);
- Momento de ascensão de linguagens de scripting;
- Poucas novas linguagens práticas e com boa usabilidade;
- **Ruby 1.0** em 1996;
- **Ruby 1.8** em 2003;
- **Ruby Gems** em 2004;
- **Rails** em 2005;
- **Ruby 1.9.3** em 2011;
- **Ruby 2.0** em 2013;
- **Ruby 2.5** desde 2017.





Por que Ruby?

- Produtiva;
- Divertida;
- Representante de Orientação a Objetos **real**;
- Fácil de usar;
- Intuitiva;
- Sintaticamente simples e bonita;
- Influenciadora de boas práticas;
- POLA (princípio da menor surpresa);
- Útil;
- Comunidade ativa e em crescimento.



Quem usa Ruby?

- Twitter (em seu lançamento);
- NASA Langley Research Center;
- Slideshare;
- Github;
- AirBnB;
- Urban Dictionary.

Começando a Programar



Entrada e Saída

```
# Output/Saída  
puts "Olá Mundo"  
# Input/Entrada  
x = gets  
y = gets.chomp
```

```
1 puts "Olá Mundo"  
2 p "Olá Mundo"  
3 print "Olá Mundo"
```



Variáveis e Tipos

- Tudo é um Objeto



“Se caminha como um pato e
grasna como um pato é um pato”

- Conversão

```
'9'.to_c      #=> (9+0i)
'2.5'.to_c     #=> (2.5+0i)
'2.5/1'.to_c   #=> ((5/2)+0i)
'-3/2'.to_c    #=> ((-3/2)+0i)
'-i'.to_c      #=> (0-1i)
'45i'.to_c     #=> (0+45i)
'3-4i'.to_c    #=> (3-4i)
'-4e2-4e-2i'.to_c #=> (-400.0-0.04i)
'-0.0-0.0i'.to_c #=> (-0.0-0.0i)
'1/2+3/4i'.to_c #=> ((1/2)+(3/4)*i)
'ruby'.to_c    #=> (0+0i)
```

```
' 2 '.to_r     #=> (2/1)
'300/2'.to_r    #=> (150/1)
'-9.2'.to_r     #=> (-46/5)
'-9.2e2'.to_r   #=> (-920/1)
'1_234_567'.to_r #=> (1234567/1)
'21 june 09'.to_r #=> (21/1)
'21/06/09'.to_r #=> (7/2)
'bwv 1079'.to_r #=> (0/1)
```

```
"12345".to_i      #=> 12345
"99 red balloons".to_i #=> 99
"0a".to_i         #=> 0
"0a".to_i(16)     #=> 10
"hello".to_i      #=> 0
"1100101".to_i(2) #=> 101
"1100101".to_i(8) #=> 294977
"1100101".to_i(10) #=> 1100101
"1100101".to_i(16) #=> 17826049
```

```
"123.45e1".to_f   #=> 1234.5
"45.67 degrees".to_f #=> 45.67
"thx1138".to_f    #=> 0.0
```

```
12345.to_s      #=> "12345"
12345.to_s(2)   #=> "11000000111001"
12345.to_s(8)   #=> "30071"
12345.to_s(10)  #=> "12345"
12345.to_s(16)  #=> "3039"
12345.to_s(36)  #=> "9ix"
```



Exercício 1

Fazer um programa que receba 2 números inteiros, realize uma soma entre eles, depois uma subtração pelo primeiro número e com esse resultado uma multiplicação pelo segundo número. Deseja-se saber o valor da soma e da subtração na mesma linha (separados por uma "/"), e o resultado da multiplicação na linha abaixo.

Dica: Lembre-se de converter String para Inteiro.



Vetores e Hashes

- Vetores

```
arr = [1, 2, 3, 4, 5, 6]
arr[2]    #=> 3
arr[100]  #=> nil
arr[-3]   #=> 4
arr[2, 3] #=> [3, 4, 5]
arr[1..4] #=> [2, 3, 4, 5]

arr.first #=> 1
arr.last  #=> 6

arr.take(3) #=> [1, 2, 3]

arr.drop(3) #=> [4, 5, 6]

arr.empty? #=> false

arr.include?(7) #=> false

arr.push(7) #=> [1, 2, 3, 4, 5, 6, 7]
arr << 8    #=> [1, 2, 3, 4, 5, 6, 7, 8]
arr.pop     #=> 8
arr.delete_at(0) #=> 1
arr.delete(3) #=> [2, 4, 5, 6, 7, 8]
```

- Hashes

```
grades = { "Jane Doe" => 10, "Jim Doe" => 6 }
options = { :font_size => 10, :font_family => "Arial" }
options = { font_size: 10, font_family: "Arial" }
options[:font_size] # => 10

grades = Hash.new
grades["Dorothy Doe"] = 9
```



Estruturas condicionais

- If/Else

```
x = 1
if x > 2
  puts "x é maior que 2"
elsif x <= 2 and x!=1
  puts "x é menor que 2 e diferente de 1"
else
  puts "não sei o valor de x"
end
```

- Unless

```
x = 1
unless x>=2
  puts "x é menor que 2"
else
  puts "x é maior que 2"
end
```

- Case

```
age = 5
case age
when 0 .. 2
  puts "bebê"
when 3 .. 6
  puts "criança"
when 7 .. 12
  puts "pré adolescente"
when 13 .. 18
  puts "adolescente"
else
  puts "adulto"
end
```



Estruturas de repetição

- Until

```
until i > num do  
  puts("Inside the loop i = #{i}" )  
  i +=1;  
end
```

- For

```
for i in 0..5  
  puts "Value of local variable is #{i}"  
end
```



Estruturas de repetição

- Each

```
ary = [1,2,3,4,5]
ary.each do |i|
  puts i
end
```

- While

```
while i < num do
  puts("Inside the loop i = #{i}" )
  i +=1
end
```



Exercício 2

Fazer um programa que receba uma lista (Vetor ou Hash) com nome e idade de 5 pessoas.
O programa deverá correr a lista e exibir:

Nome das pessoas com 20 ou mais com a notação:

"Nome" tem 20 ou mais anos!

Nome das pessoas com menos de 20 anos com a notação:

"Nome" tem menos de 20 anos!

Caso a pessoa tenha 0 anos o programa deverá exibir:

"Nome" é um(a) bebê!

Dica: Sempre bom começar os testes da condição exclusiva.



Enumerables

- Times

```
5.times do |i|  
  print i.to_s  
end
```

- Map

```
arr = [1, 2, 3, 4, 5, 6]  
arr.map {|i| i*20}
```

- Any/All/None/Find

```
arr = [1, 2, 3, 4, 5, 6]  
arr.any? {|i| i%2}  
arr.all? {|i| i%2}  
arr.none? {|i| i%2}  
arr.find {|i| i%2}  
arr.find_all {|i| i%2}
```

- each with index

```
grades = { "Jane Doe" => 10, "Jim Doe" => 6 }  
grades.each_with_index do |k, v|  
  print k + " tirou a nota " + v  
end
```

Orientação a Objetos

```
class Pato
  def voar
    puts "Pato voando"
  end
end

class Aviao
  def voar
    puts "Avião voando"
  end
end

class Baleia
  def nadar
    puts "Baleia nadando"
  end
end

def fazer_voar(e)
  e.voar
end

fazer_voar(Pato.new)
fazer_voar(Aviao.new)
fazer_voar(Baleia.new)
```



```
class Livro
  def initialize(autor, titulo, paginas)
    @autor = autor
    @titulo = titulo
    @paginas = paginas
    #ou
    @autor, @titulo, @paginas = autor, titulo, paginas
  end

  def autor
    @autor
  end

  def autor=(autor)
    @autor = autor
  end

  def titulo
    @titulo
  end

  def titulo=(titulo)
    @titulo = titulo
  end

  def paginas
    @paginas
  end

  def paginas=(paginas)
    @paginas = paginas
  end
end
```

```
class Livro
  attr_accessor :autor, :titulo, :paginas
  def initialize(autor, titulo, paginas)
    @autor = autor
    @titulo = titulo
    @paginas = paginas
  end
end
```

```
class Livro
  attr_accessor :autor, :titulo, :paginas
  def initialize(autor, titulo, paginas)
    @autor = autor
    @titulo = titulo
    @paginas = paginas
  end

  def to_s
    "#{@titulo}, por #{@autor}. #{@paginas} páginas."
  end
end

livro = Livro.new("Machado de Assis", "A Mão e a Luva", 300)
puts livro.autor
puts livro
puts livro.to_s
print livro
p livro
```

```
class Animal
  attr_accessor :especie, :nome, :sexo
  def initialize(especie, nome, sexo)
    @especie = especie
    @nome = nome
    @sexo = sexo
  end

  def to_s
    "#{@nome}, sexo #{@sexo} da espécie #{@especie}"
  end
end

class Cachorro < Animal
  attr_accessor :nome, :sexo
  @ESPECIE = "Canis lupus"
  def initialize( nome, sexo)
    @nome = nome
    @sexo = sexo
  end

  def especie
    @ESPECIE
  end

  def to_s
    "#{@nome}, sexo #{@sexo} da espécie #{@especie}"
  end
end
```

```
class Retangulo
  attr_accessor :largura, :altura
  def initialize(l,a)
    @largura = l
    @altura = a
  end

  def +(other)
    Retangulo.new(@largura + other.largura, @altura + other.altura)
  end

  def -(other)
    Retangulo.new(@largura - other.largura, @altura - other.altura)
  end

  def /(other)
    Retangulo.new(@largura / other.largura, @altura / other.altura)
  end

  def *(other)
    Retangulo.new(@largura * other.largura, @altura * other.altura)
  end

  def to_s
    "Retângulo de largura #{@largura} e altura #{@altura}"
  end
end

retangulo_1 = Retangulo.new(5,5)
retangulo_2 = Retangulo.new(10,10)

puts retangulo_1
puts retangulo_2
puts retangulo_1 + retangulo_2
puts retangulo_1 - retangulo_2
puts retangulo_1 * retangulo_2
puts retangulo_2 / retangulo_1
```