

RUBY



Yukihiko Matsumoto

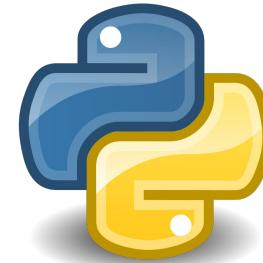
Cientista da Computação

HISTÓRICO





OBJETIVO



- Desenvolvida para ser utilizada como linguagem de script
- Open Source

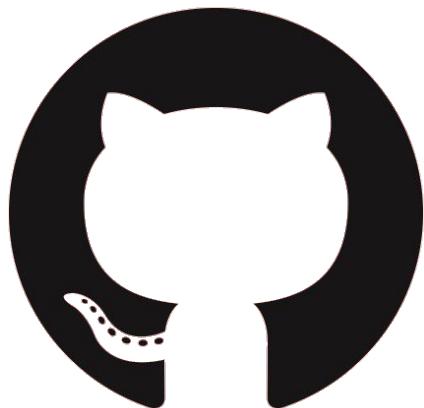
Utilização do Ruby

01 Web (através do RoR)

02 Testes

03 DevOps

QUEM UTILIZA RUBY?



CARACTERÍSTICAS

- *Linguagem interpretada*
- *Multiparadigma (predomina a orientação a objetos)*
- *Portável*
- *Tipagem dinâmica e forte*
- *Gerenciamento de memória automático*
- *Open Source*
- *Gerenciador de versão (Rbenv e RVM)*
- *RubyGems*
- *Extensão .rb*
- *Sempre terá o delimitador end em cada código de bloco isolado*

PALAVRAS RESERVADAS

__ENCODING__	def	module	then
__LINE__	defined?	next	true
__FILE__	do	nil	undef
BEGIN	else	not	unless
END	elsif	or	until
alias	end	redo	when
and	ensure	rescue	while
begin	false	retry	yield
break	for	return	
case	if	self	
class	in	super	

OPERADORES ARITMÉTICOS

Para $a = 4$ e $b = 2$

Operadores	O que faz?	Exemplos
+	O operador adiciona dois operandos	$a + b = 6$
-	O operador subtrai dois operandos	$a - b = 2$
*	O operador multiplica dois operandos	$a * b = 8$
/	O operador divide o primeiro operando pelo segundo	$a / b = 2$
%	O operador retorna o resto da divisão do primeiro operando pelo segundo	$a \% b = 0$
**	O operador calcula o exponencial dos operandos	$a ** b = 16$

OPERADORES DE COMPARAÇÃO

Operadores	O que faz?	Exemplos
<code>==</code>	Verifica se o valor dos dois operandos são iguais ou não	<code>a == b</code> - false
<code>!=</code>	Verifica se o valor dos dois operandos são diferentes ou não	<code>a != b</code> - true
<code>></code>	Verifica se o primeiro operando é maior que o primeiro	<code>a > b</code> - true
<code><</code>	Verifica se o primeiro operando é menor que o primeiro	<code>a < b</code> - false
<code>>=</code>	Verifica se o primeiro operando é maior ou igual que o primeiro	<code>a >= b</code> - true
<code><=</code>	Verifica se o primeiro operando é menor ou igual que o primeiro	<code>a <= b</code> - false

Para `a = 4` e `b = 2`

OPERADORES DE COMPARAÇÃO

Para a = 4 e b = 2

Operadores	O que faz?	Exemplos
<code><=></code>	Operador de comparação combinada. Retorna 0 se o primeiro operando é igual ao segundo, retorna 1 se o primeiro operando for maior que o segundo e retorna -1 se o primeiro operando for menor que o segundo.	<code>a <=> b</code> 1
<code>==</code>	O operador testará a igualdade dentro de uma cláusula que contenha instruções	<code>(1..10) == 5 - true</code>
<code>.eq?</code>	O operador retorna true se o receptor e o argumento tiverem valores e tipos iguais	<code>1 .eq? 1.0 - false</code>
<code>equal?</code>	O operador retorna true se o receptor e o argumento tiverem a mesma identificação do objeto	<code>valor = 10 valor .equal? 10 true</code>

OPERADORES DE ATRIBUIÇÃO

Para $a = 4$ e $b = 2$

Operadores	O que faz?	Exemplos
=	Operador de atribuição simples, atribui o valor à direita do operador à sua esquerda	$c = a + b$, é atribuído o valor da soma $a + b$ para c
+=	Adiciona e faz atribuição	$c += a$ é equivalente a $c = c + a$
-=	Subtrai e faz atribuição	$c -= a$ é equivalente a $c = c - a$
*=	Multiplica e faz atribuição	$c *= a$ é equivalente a $c = c * a$
/=	Divide e faz atribuição	$c /= a$ é equivalente a $c = c / a$
%=	Divide e faz a atribuição do resto	$c %= a$ é equivalente a $c = c \% a$
**=	Exponencia e faz atribuição	$c **= a$ é equivalente a $c = c ** a$

OPERADORES LÓGICOS

Operadores	O que faz?	Exemplos
and ou &&	Se ambas as condições forem verdadeiras é retornado true	a && b - true
or ou	Se pelo menos uma das condições for verdadeira é retornado true	a b - true
not ou !	Inverte o estado lógico do operando	!(a && b) - false

OPERADORES BINÁRIOS

Para $a = 60$ e $b = 13$. Em binário, ($a = 0011\ 1100$) e ($b = 0000\ 1101$)

$$a \sim b = 0000\ 1100$$

$$a | b = 0011\ 1101$$

$$a ^ b = 0011\ 0001$$

$$\sim a = 1100\ 0011$$

Operadores	O que faz?	Exemplos
&	Operador AND	($a \& b$) dará 12, em binário 0000 1100
	Operador OR	($a b$) dará 61, em binário 0011 1101
^	Operador XOR	($a ^ b$) dará 49, em binário 0011 0001
~	Operador NOT	$\sim a$ dará -61, em binário 1100 0011
<<	Operador com desvio à esquerda	$a << 2$ dará 240, em binário 1111 0000
>>	Operador com desvio à direita	$a >> 2$ dará 15, em binário 0000 1111

OPERADOR TERNÁRIO

Para $a = 4$ e $b = 2$

$a > b ? \text{'verdadeiro'} : \text{'falso'}$

Operador	O que faz?	Exemplos
$? :$	Expressão condicional	Se a condição é verdadeira? então o valor é x : senão o valor é y

OPERADORES DE ALCANCE

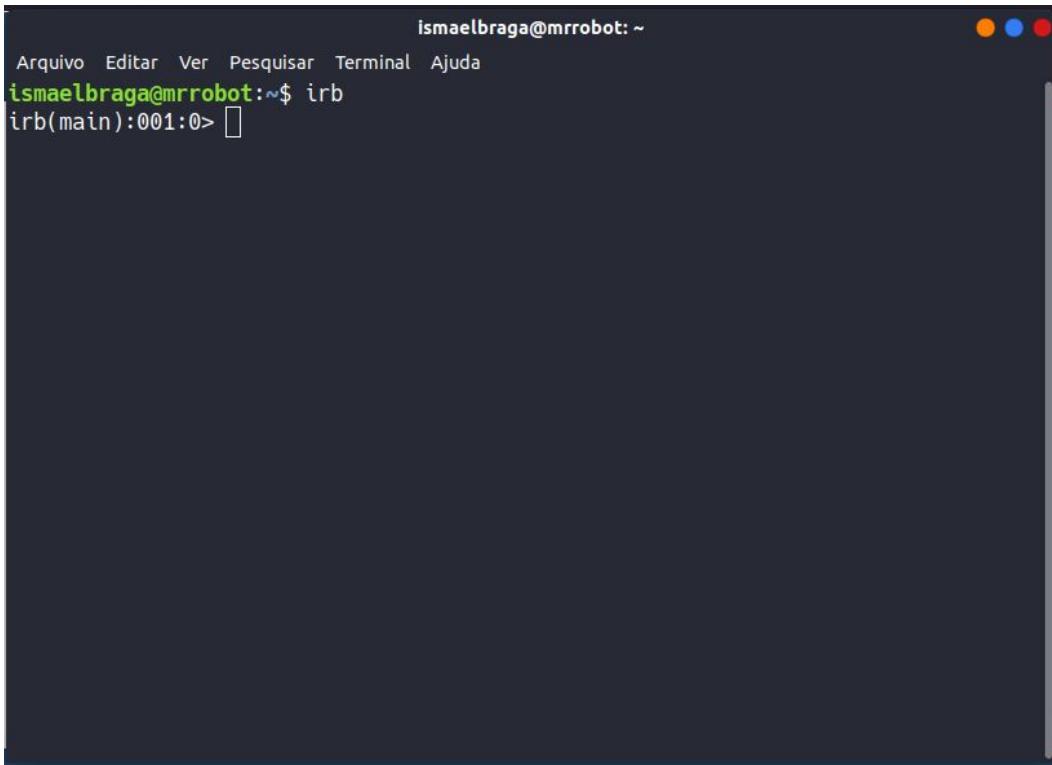
Operadores	O que faz?	Exemplos
..	Cria um intervalo do ponto inicial ao ponto final, incluindo o último valor	1..20 cria um intervalo de 1 a 20
...	Cria um intervalo do ponto inicial ao ponto final, excluindo o último valor	1...20 cria um intervalo de 1 a 19

OPERADOR DEFINED?

Para PI = 3.14

Operador	O que faz?	Exemplo
defined?	Irá retornar uma string do argumento passado informando o que o define. Caso o argumento passado não esteja definido, será retornado nil	defined?(PI) o retorno será constant

INTERACTIVE RUBY SHELL - IRB



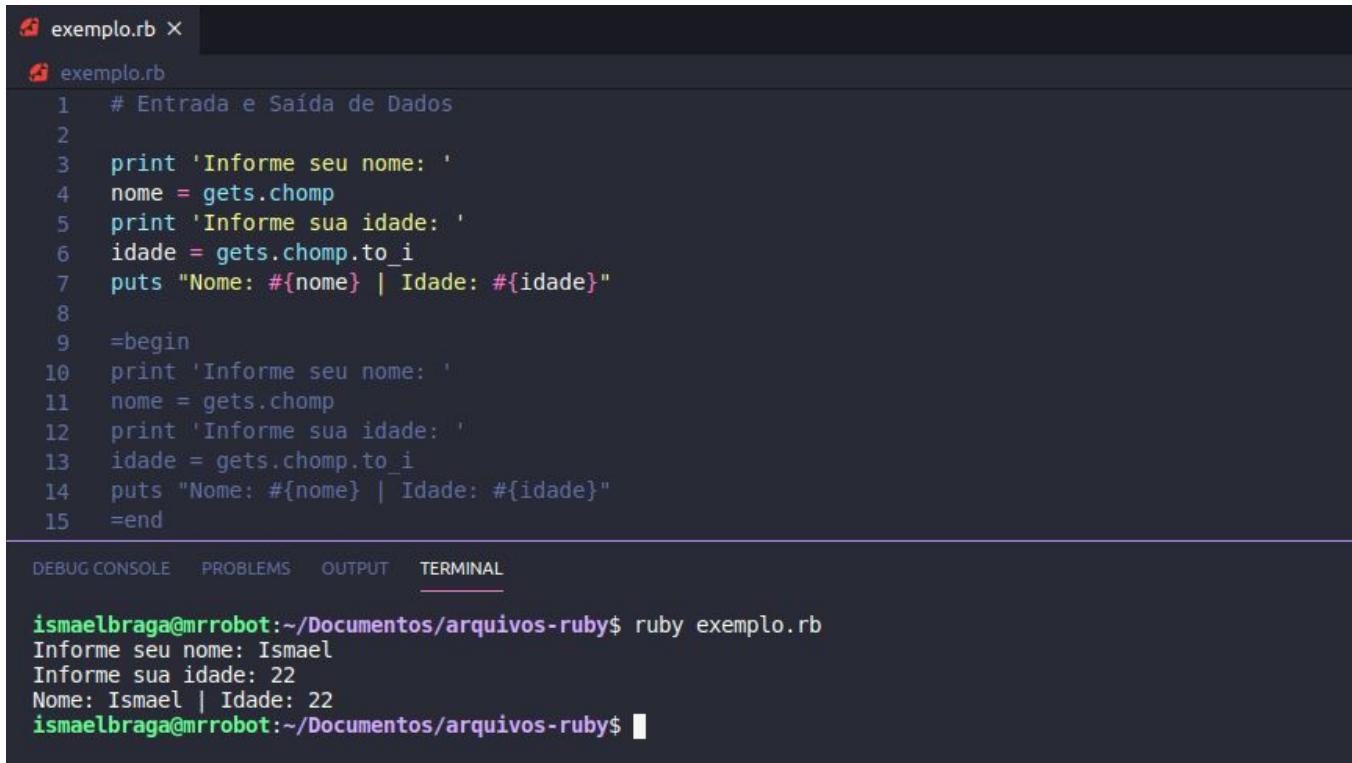
A screenshot of a terminal window titled "ismaelbraga@mrrobot: ~". The window has a dark background and a light gray border. At the top, there is a menu bar with options: Arquivo, Editar, Ver, Pesquisar, Terminal, Ajuda. Below the menu, the command "ismaelbraga@mrrobot:~\$ irb" is typed, followed by the IRB prompt "irb(main):001:0>". The rest of the window is blank, showing a large white space.

```
ismaelbraga@mrrobot: ~
Arquivo Editar Ver Pesquisar Terminal Ajuda
ismaelbraga@mrrobot:~$ irb
irb(main):001:0>
```

TIPOS DE DADOS

- Não há tipos primitivos. Tudo em Ruby é um objeto
- *String*
- *Numeric*
 - *Integer (Fixnum, Bignum)*
 - *Float*
- *Boolean*
- *Symbol*
- *Array*
- *Hash*

EXEMPLO DE CÓDIGO



The screenshot shows a code editor window with a dark theme. At the top, there are two tabs: "exemplo.rb X" and "exemplo.rb". The code editor displays the following Ruby script:

```
1 # Entrada e Saída de Dados
2
3 print 'Informe seu nome: '
4 nome = gets.chomp
5 print 'Informe sua idade: '
6 idade = gets.chomp.to_i
7 puts "Nome: #{nome} | Idade: #{idade}"
8
9 =begin
10 print 'Informe seu nome: '
11 nome = gets.chomp
12 print 'Informe sua idade: '
13 idade = gets.chomp.to_i
14 puts "Nome: #{nome} | Idade: #{idade}"
15 =end
```

Below the code editor, there is a navigation bar with four items: DEBUG CONSOLE, PROBLEMS, OUTPUT, and TERMINAL. The TERMINAL item is underlined, indicating it is the active tab. The terminal window below shows the output of running the script:

```
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby exemplo.rb
Informe seu nome: Ismael
Informe sua idade: 22
Nome: Ismael | Idade: 22
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ █
```

ESTRUTURAS DE CONTROLE

- *Condicional*
 - *If*
 - *Else*
 - *Elsif*
 - *Unless*
 - *Case*
- *Iteração*
 - *For*
 - *While*
 - *Times*
 - *Loop (Do While)*
 - *Until*

UNLESS



```
unless.rb ✘
unless.rb
1 sinal = 'vermelho'
2
3 unless sinal == 'verde' # É equivalente ao if !=
4   pedestre = 'pode atravessar a rua'
5 else
6   pedestre = 'precisa aguardar o sinal fechar'
7 end
8
9 puts "O pedestre #{pedestre}"
10
11 sinal = 'vermelho'

DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL

ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby unless.rb
O pedestre pode atravessar a rua
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ █
```

CASE

```
case.rb  X
case.rb
1 print 'Informe o número do mês em que você nasceu? '
2 mes = gets.chomp.to_i
3
4 case mes
5 when 1..3
6   puts 'Você nasceu no início do ano.'
7 when 4..6
8   puts 'Você nasceu na primeira metade do ano.'
9 when 7..9
10  puts 'Você nasceu na segunda metade do ano.'
11 when 10..12
12  puts 'Você nasceu no final do ano.'
13 end

DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby case.rb
Informe o número do mês em que você nasceu? 9
Você nasceu na segunda metade do ano.
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$
```

TIMES

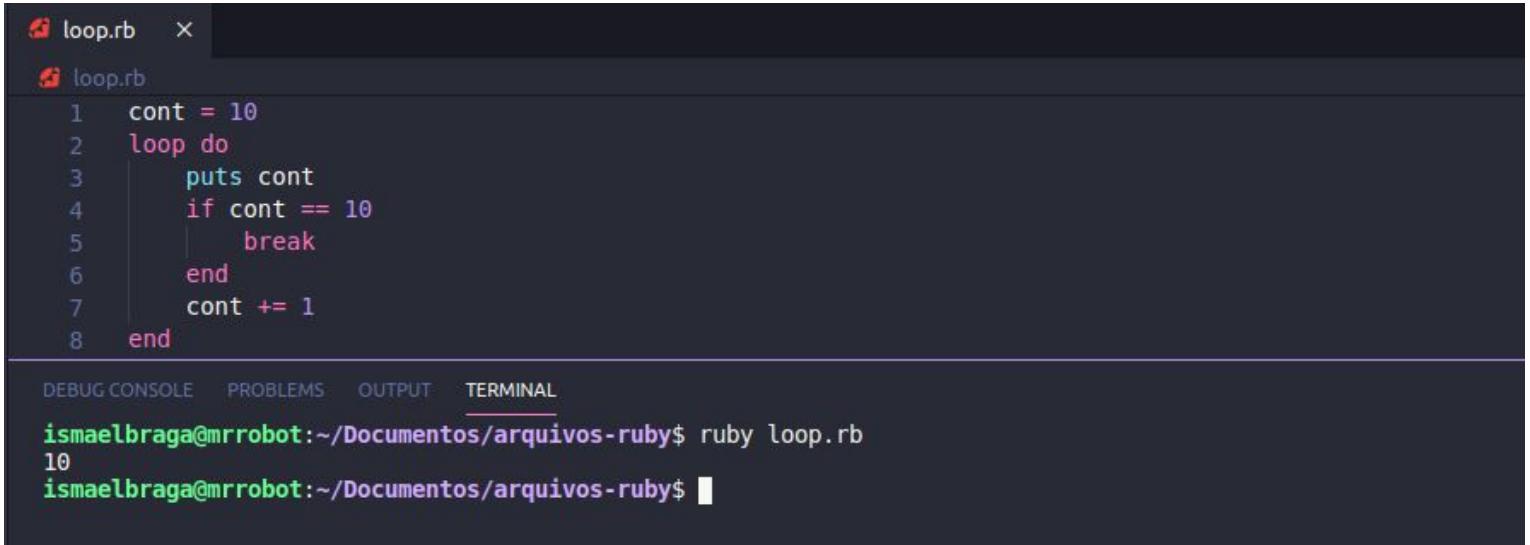
A screenshot of a code editor interface, likely Visual Studio Code, showing a Ruby script named `times.rb`. The code consists of three lines:

```
1 5.times do
2   puts 'Conceitos de Linguagem de Programação'
3 end
```

The code editor has tabs for DEBUG CONSOLE, PROBLEMS, OUTPUT, and TERMINAL. Below the code editor, a terminal window shows the output of running the script:

```
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby times.rb
Conceitos de Linguagem de Programação
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$
```

LOOP



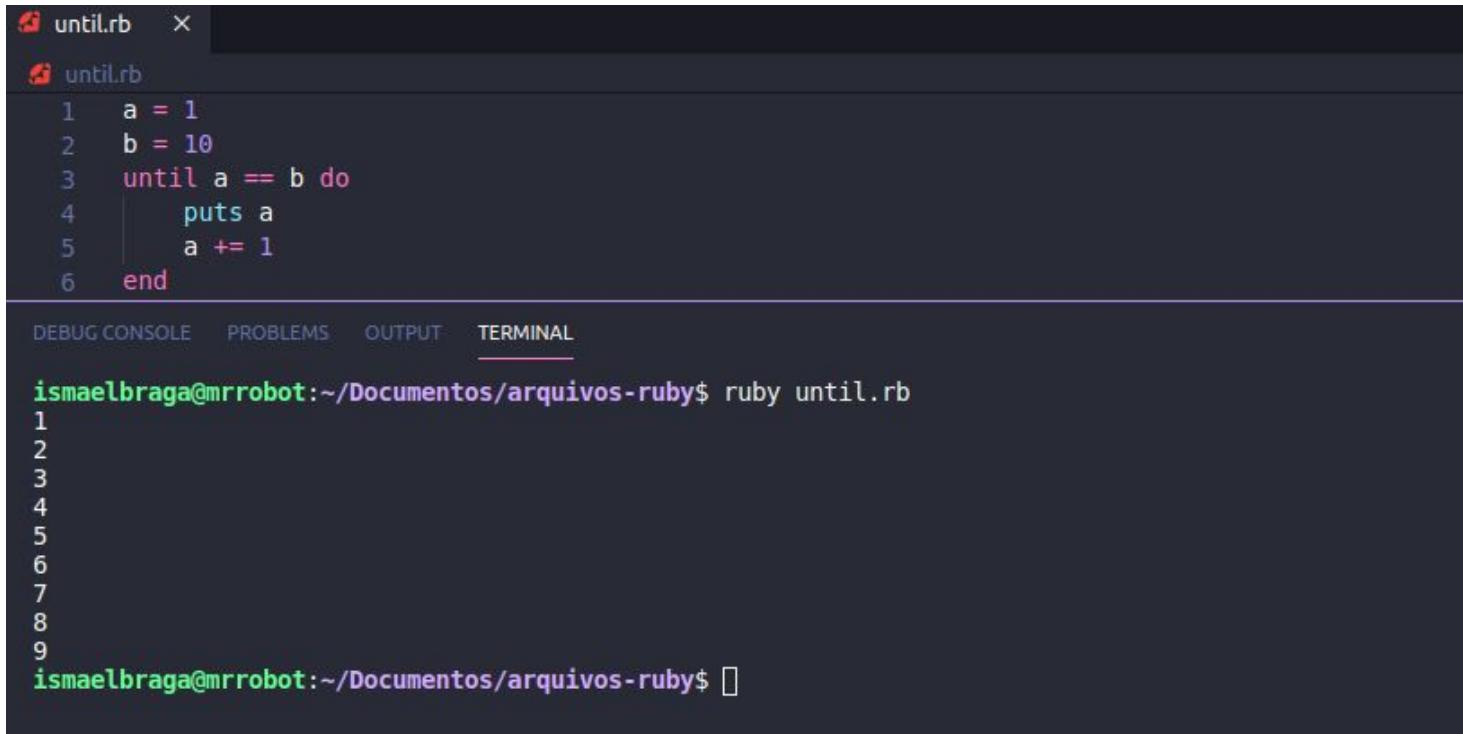
The screenshot shows a code editor interface with a dark theme. On the left, there's a file tree with a single file named "loop.rb". The main area displays the following Ruby code:

```
loop.rb
1 cont = 10
2 loop do
3   puts cont
4   if cont == 10
5     break
6   end
7   cont += 1
8 end
```

Below the code editor, there are tabs for DEBUG CONSOLE, PROBLEMS, OUTPUT, and TERMINAL. The TERMINAL tab is active, showing the command-line output of running the script:

```
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby loop.rb
10
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$
```

UNTIL



A screenshot of a terminal window titled "until.rb". The code inside the file is:

```
1 a = 1
2 b = 10
3 until a == b do
4   puts a
5   a += 1
6 end
```

The terminal interface includes tabs for DEBUG CONSOLE, PROBLEMS, OUTPUT, and TERMINAL, with TERMINAL being active. The command run in the terminal is:

```
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby until.rb
```

The output of the script is displayed below the command:

```
1
2
3
4
5
6
7
8
9
```

The terminal prompt at the bottom is:

```
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$
```

MÉTODO

```
 metodo.rb X  
 metodo.rb  
 1  def operacoes_basicas(a, b)  
 2      puts "#{a} + #{b} = #{a + b}"  
 3      puts "#{a} - #{b} = #{a - b}"  
 4      puts "#{a} / #{b} = #{a / b}"  
 5      puts "#{a} * #{b} = #{a * b}"  
 6  end  
 7  
 8  operacoes_basicas(10, 5)
```

ESCOPO DE VARIÁVEL

- *Local*
- *Global* (`$nome_da_variável`)

LEGIBILIDADE

- *Sintaxe simples e de fácil entendimento; o que acaba facilitando o aprendizado e a manutenibilidade*
- *Os tipos das variáveis não precisam ser definidos*
- *A linguagem é case sensitive (há diferenciação entre letras minúsculas e maiúsculas)*
- *Sintaxe semelhante ao Python*

REDIGIBILIDADE

- *Possui uma sintaxe enxuta, com poucas linhas de código você consegue fazer um algoritmo mais complexo*
- *Existem diversas maneiras de resolver um mesmo problema usando comandos nativos do Ruby*

CONFIABILIDADE

- *Possui suporte ao tratamento de exceções semelhante ao de Python e Java*

ORTOGONALIDADE

- É uma linguagem pouco ortogonal
- Yukihiro Matsumoto optou pela simplicidade em detrimento da ortogonalidade. Segundo Matz, quando características ortogonais são combinadas podem acarretar em uma grande complexidade.

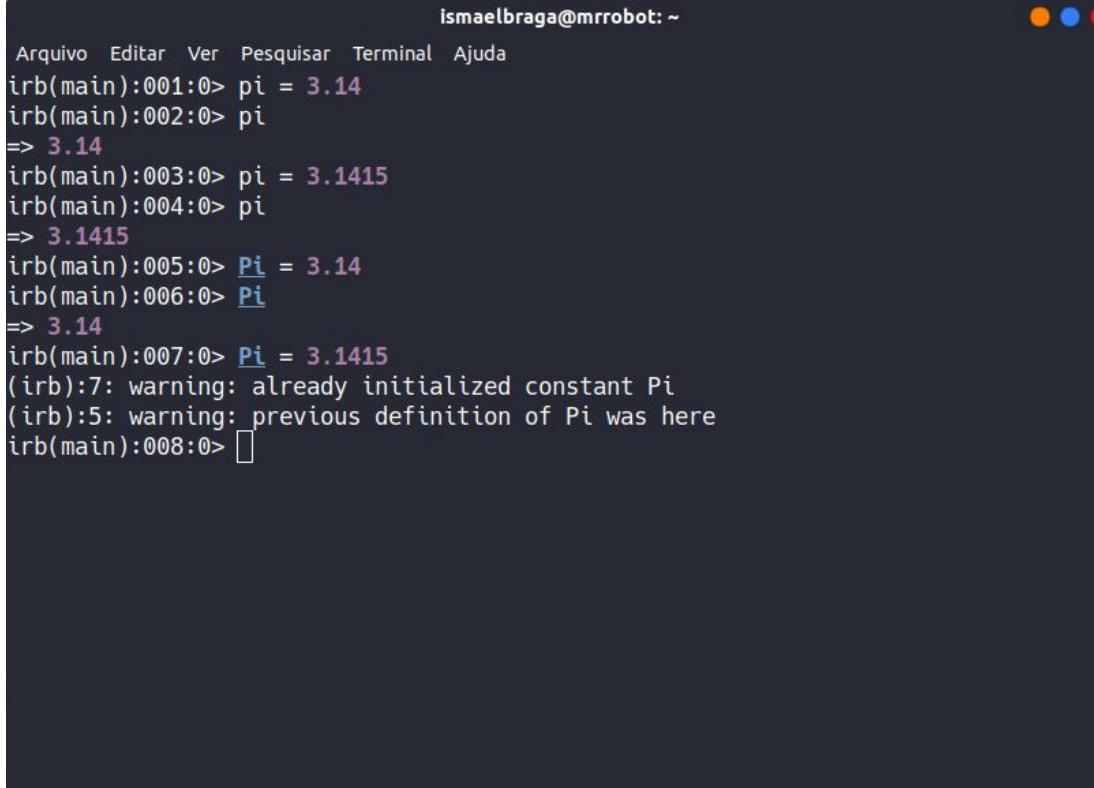
REUSABILIDADE

- Ruby é uma linguagem multiparadigma, mas a predominância explícita é a Orientação a Objetos, por conta disso possui alta reusabilidade
- Além disso, Ruby é altamente modular e possui uma vasta quantidade de gems (bibliotecas)

CONCORRÊNCIA

- Possui suporte? **SIM**
class Threads
- Para o uso de semáforos
class Mutex

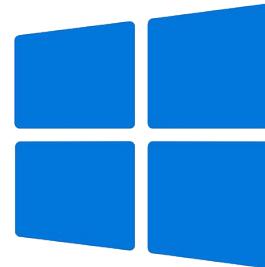
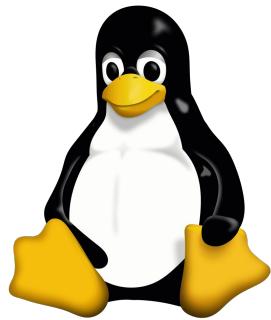
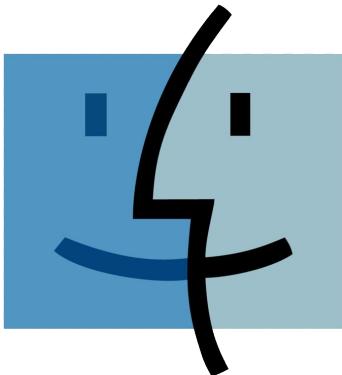
CONSTANTE



A screenshot of a macOS terminal window titled "ismaelbraga@mrrobot: ~". The window has a dark theme with orange, blue, and red window control buttons. The terminal shows the following IRB session:

```
Arquivo Editar Ver Pesquisar Terminal Ajuda
irb(main):001:0> pi = 3.14
irb(main):002:0> pi
=> 3.14
irb(main):003:0> pi = 3.1415
irb(main):004:0> pi
=> 3.1415
irb(main):005:0> Pi = 3.14
irb(main):006:0> Pi
=> 3.14
irb(main):007:0> Pi = 3.1415
(irb):7: warning: already initialized constant Pi
(irb):5: warning: previous definition of Pi was here
irb(main):008:0> □
```

PORTABILIDADE



EFICIÊNCIA

- *Linguagem Interpretada*
 - Se comparada a linguagens compiladas como C e C++, a sua velocidade será muito mais lenta no momento da execução
- *Tipagem dinâmica e forte*
 - Por não precisar especificar o tipo de dado da variável ao criá-la, ela tende a ser mais lenta durante a execução do algoritmo justamente por precisar fazer essa verificação

PASSAGEM DE PARÂMETROS

- A passagem de parâmetro é feita por cópia da referência



```
parametro.rb X
parametro.rb
1 def numero_qualquer(n)
2     n = 26
3     puts n
4 end
5
6 numero = 13
7 numero_qualquer(numero)
8 puts numero

DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby parametro.rb
26
13
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$
```

PASSAGEM DE PARÂMETROS

- Quantidade variável de passagem de parâmetros

```
parametro.rb ×
parametro.rb
1 def lista(*valores)
2   puts valores
3 end
4
5 lista('Bernardo', 'Ismael', 'Rafael')

DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby parametro.rb
Bernardo
Ismael
Rafael
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ █
```

PASSAGEM DE PARÂMETROS

- Passagem de parâmetro tendo valor padrão



```
parametro.rb ×
parametro.rb
1 def exponencial(x, y = 2)
2   puts "#{x}^#{y} = #{x ** y}"
3 end
4
5 exponencial(5)
6 exponencial(5, 3)

DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby parametro.rb
5^2 = 25
5^3 = 125
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$
```

CURTO-CIRCUITO

- Avaliação curto-circuito de uma expressão é quando o resultado é determinado sem a necessidade de avaliar todos os operandos e/ou operadores da expressão
- Os operadores `&&` e `||` são avaliados em curto-circuito

```
curto_circuito.rb ×
curto_circuito.rb
1 a = false
2 b = false
3 puts "curto-circuito" if a && b
4
5 a = true
6 b = false
7 puts "curto-circuito" if a && b
8
9 a = false
10 a = true
11 puts "curto-circuito" if a && b
12
13 a = true
14 b = true
15 puts "curto-circuito" if a && b
16
```

```
curto_circuito2.rb ×
curto_circuito2.rb
1 a = false
2 b = false
3 puts "curto-circuito" if a || b
4
5 a = true
6 b = false
7 puts "curto-circuito" if a || b
8
9 a = false
10 a = true
11 puts "curto-circuito" if a || b
12
13 a = true
14 b = true
15 puts "curto-circuito" if a || b
16
```

POO

- *Sintaxe de uma classe*
- *Herança*
- *Polimorfismo (Subtipo, Coerção, Sobrecarga)*
- *Modificadores de acesso (public, private, protected)*
- *Atributos (@nome_da_variavel)*

TRATAMENTO DE EXCEÇÃO

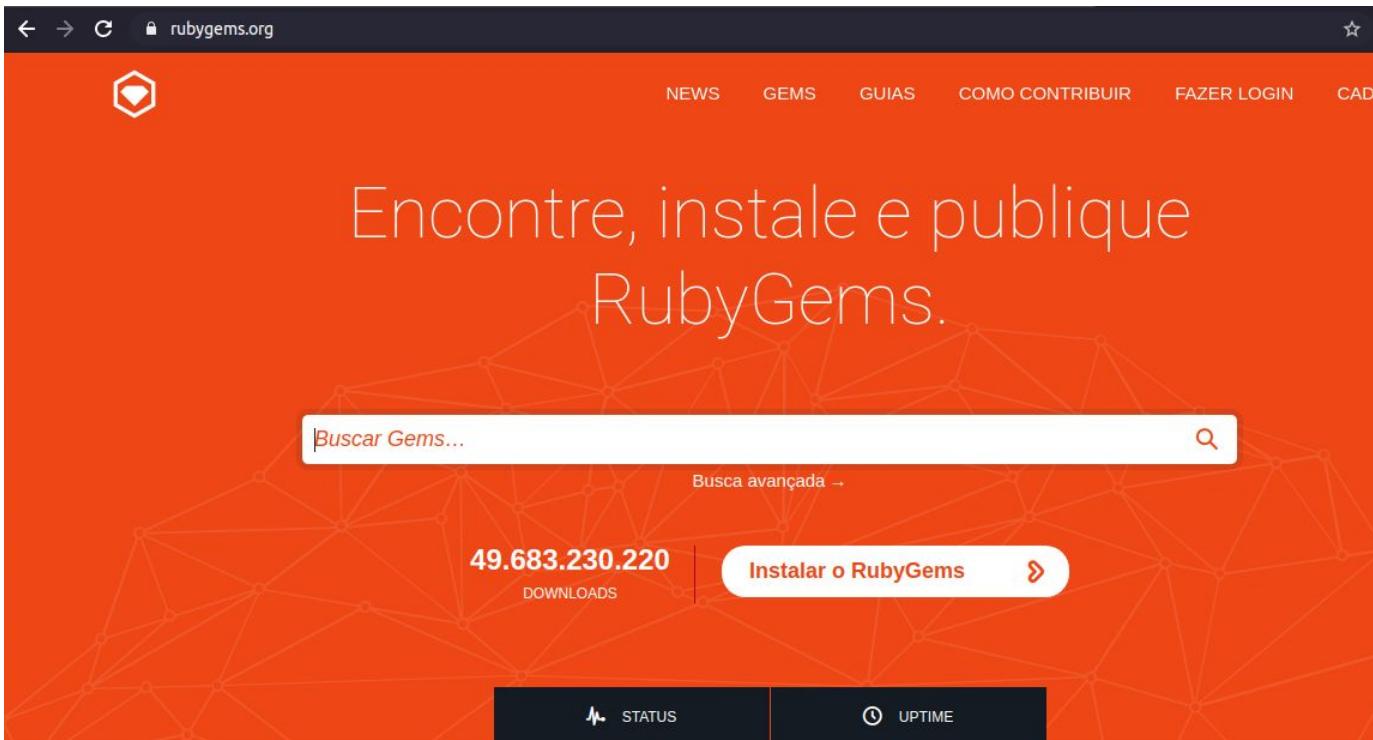
A screenshot of a code editor and terminal window. The code editor shows a file named 'tratamento.rb' with the following content:

```
tratamento.rb
tratamento.rb
1 def divide(a, b)
2   begin
3     puts a / b
4   rescue
5     puts 'Erro: impossível dividir por zero'
6   ensure
7     puts 'Sempre serei executado independente se houver exceção ou não'
8   end
9 end
10
11 divide(2, 0)
12
13 divide(4, 2)
```

The terminal below shows the execution of the script:

```
DEBUG CONSOLE PROBLEMS OUTPUT TERMINAL
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ ruby tratamento.rb
Erro: impossível dividir por zero
Sempre serei executado independente se houver exceção ou não
2
Sempre serei executado independente se houver exceção ou não
ismaelbraga@mrrobot:~/Documentos/arquivos-ruby$ █
```

RUBY GEMS



The screenshot shows the RubyGems.org homepage with a vibrant orange background featuring a network graph pattern. At the top, there's a navigation bar with links for NEWS, GEMS, GUIAS, COMO CONTRIBUIR, FAZER LOGIN, and CADASTRAR. Below the navigation is a large, central text area with the slogan "Encontre, instale e publique RubyGems." A search bar with the placeholder "Buscar Gems..." and a magnifying glass icon is positioned below the slogan. To the right of the search bar is a link to "Busca avançada →". In the bottom left corner, there's a statistic showing "49.683.230.220 DOWNLOADS". Next to it is a button labeled "Instalar o RubyGems" with a right-pointing arrow. The bottom of the page features a dark footer bar with two sections: "STATUS" and "UPTIME".

rubygems.org

NEWS GEMS GUIAS COMO CONTRIBUIR FAZER LOGIN CADASTRAR

Encontre, instale e publique RubyGems.

Buscar Gems... 

Busca avançada →

49.683.230.220 DOWNLOADS

Instalar o RubyGems 

STATUS UPTIME

PONTOS

Vantagens

- Fácil de aprender
- Foco na simplicidade e na produtividade
- Totalmente livre
- Comunidade ativa
- Grande número de gems

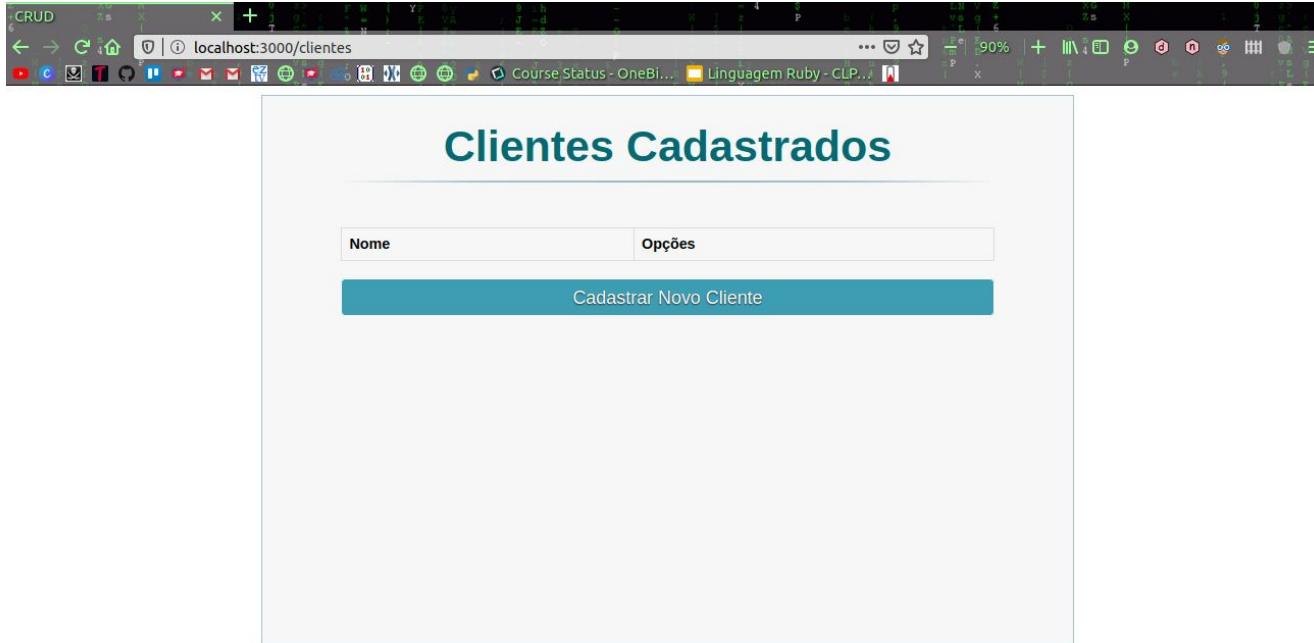
Desvantagens

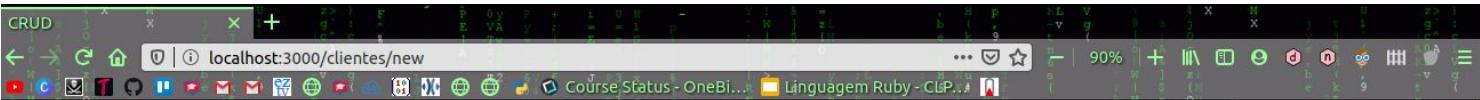
- Nicho menor dentro do Desenvolvimento Web
- Tempo de execução lento

“O Ruby é simples na aparência, mas muito complexo no interior, tal como o corpo humano.”

— Yukihiro Matsumoto

Implementação





Cadastrar Cliente

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

Selecionar ▾

Enviar

Voltar

CRUD

localhost:3000/clientes

Course Status - OneBi... Linguagem Ruby - CLP...

Cadastrar Cliente

Preencha todos os campos!

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

Selecionar

Enviar

Voltar

GRUD

localhost:3000/clientes

Course Status - OneB... Linguagem Ruby - CLP...

Cadastrar Cliente

Email Inválido!

Nome
rafael de almeida menezes

Email
teste

CPF
038 [REDACTED] 03

Telefone(DDD)
85997305905

Idade
19

Sexo
Selecionar ▾

[Enviar](#)

[Voltar](#)

The screenshot shows a web application interface for client registration. At the top, there is a browser header with the URL 'localhost:3000/clientes'. Below the header is a main content area with a title 'Cadastrar Cliente'. Inside the form, there is an error message 'Email Inválido!' (Invalid Email!). The form fields include 'Nome' (Name) with value 'rafael de almeida menezes', 'Email' (Email) with value 'teste', 'CPF' (CPF) with value '038 [REDACTED] 03', 'Telefone(DDD)' (Phone DDD) with value '85997305905', 'Idade' (Age) with value '19', and 'Sexo' (Sex) with a dropdown menu labeled 'Selecionar ▾' (Select). At the bottom of the form are two buttons: a blue 'Enviar' (Send) button and a teal 'Voltar' (Back) button.

localhost:3000/clientes

Cadastrar Cliente

CPF Inválido!

Nome

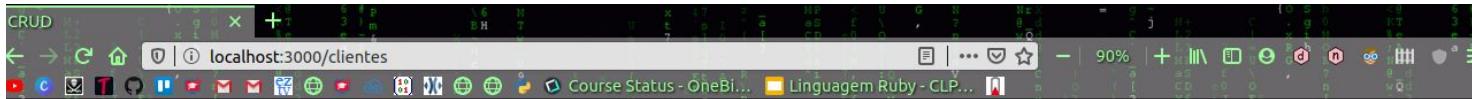
Email

CPF

Telefone(DDD)

Idade

Sexo



Cadastrar Cliente

Número Inválido!

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

localhost:3000/clientes

Cadastrar Cliente

Idade Inválida!

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

localhost:3000/clientes

Cadastrar Cliente

Selezione o Sexo!

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

The screenshot shows a web browser window with a dark theme. The address bar displays "localhost:3000/clientes/24". The main content area features a title "Informações do Cliente" and a success message "Cliente cadastrado com sucesso!". Below this, various client details are listed: Nome: Rafael De Almeida Menezes, Email: rafaelalmeida2909@gmail.com, CPF: 038-XXXX-03, Telefone: (85)99730-5905, Idade: 19 anos, and Sexo: Masculino. At the bottom are two teal buttons labeled "Editar" and "Voltar".

Informações do Cliente

Cliente cadastrado com sucesso!

Nome: Rafael De Almeida Menezes

Email: rafaelalmeida2909@gmail.com

CPF: 038-XXXX-03

Telefone: (85)99730-5905

Idade: 19 anos

Sexo: Masculino

[Editar](#)

[Voltar](#)

A screenshot of a web browser displaying a client management application. The title bar shows the URL `localhost:3000/clientes`. The main content area has a heading **Cientes Cadastrados**. Below it is a table with one row containing the name **Rafael De Almeida Menezes** and three buttons: **Mostrar**, **Editar**, and **Deletar**. At the bottom of the page is a blue button labeled **Cadastrar Novo Cliente**.

Nome	Opções
Rafael De Almeida Menezes	Mostrar Editar Deletar

[Cadastrar Novo Cliente](#)

CRUD

localhost:3000/clientes/23/edit

Course Status - OneBi... Linguagem Ruby - CLP...

Editar Cliente

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

localhost:3000/clientes

Cadastrar Cliente

Email já cadastrado!

Nome

Email

CPF

Telefone(DDD)

Idade

Sexo

CRUD

localhost:3000/Clientes

Course Status - OneB... Linguagem Ruby - CLP...

Cadastrar Cliente

CPF já cadastrado!

Nome

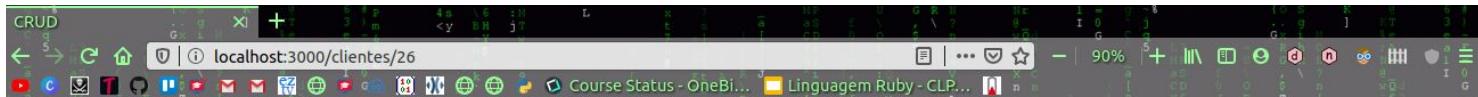
Email

CPF

Telefone(DDD)

Idade

Sexo



Informações do Cliente

Nome: Ismael Braga

Email: ismael.braga@aluno.uece.br

CPF: 060 [REDACTED] 44

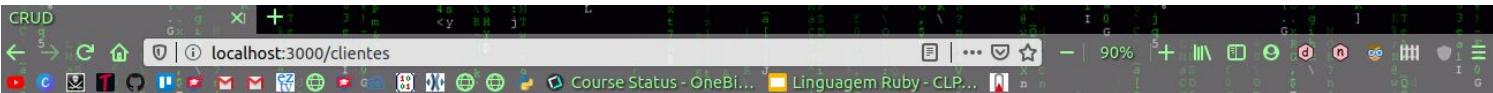
Telefone: (85)99945-0763

Idade: 22 anos

Sexo: Masculino

[Editar](#)

[Voltar](#)



Cientes Cadastrados

Nome	Opções
Rafael De Almeida Menezes	Mostrar Editar Deletar
Ismael Braga	Mostrar Editar Deletar

[Cadastrar Novo Cliente](#)

A screenshot of a web browser window titled "localhost:3000/clientes". The title bar also displays "CRUD" and a list of icons. The main content area has a header "Clientes Cadastrados" and a success message "Cliente deletado com sucesso!". Below this, there is a table with one row containing the name "Ismael Braga" and three buttons: "Mostrar", "Editar", and "Deletar". At the bottom of the page is a blue button labeled "Cadastrar Novo Cliente".

localhost:3000/clientes

Clientes Cadastrados

Cliente deletado com sucesso!

Nome	Opções
Ismael Braga	Mostrar Editar Deletar

Cadastrar Novo Cliente

Obrigado!