

(48) 3285 5615 / 9645 5506  
[contato@qualister.com.br](mailto:contato@qualister.com.br)



- Terceirização de profissionais
- Consultoria de teste
- Avaliação de usabilidade
- Automação de testes
- Testes de performance
- Treinamentos

## Automação de testes com Cucumber

[www.qualister.com.br](http://www.qualister.com.br)

## Importante

- É proibida a cópia e reprodução de qualquer parte do conteúdo desta apresentação incluindo, mas não limitado a, textos, imagens, gráficos e tabelas. Esta apresentação é protegida pelas leis de Copyright e são propriedade da Qualister Consultoria e Treinamento LTDA.
- Não é permitido modificar, copiar, guardar em banco de dados público, alugar, vender ou republicar qualquer parte desta apresentação, sem prévia permissão explícita do autor.
- Quando houver permissão de uso deste material, é obrigatória a referência bibliográfica conforme as normas vigentes.



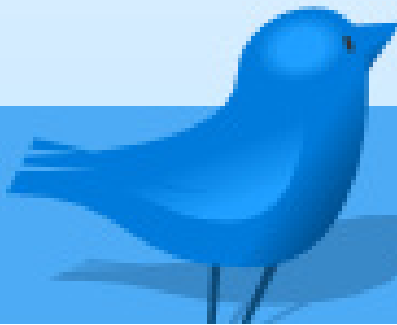
## Júlio de Lima

**E-mail:** [julio.lima@qualister.com.br](mailto:julio.lima@qualister.com.br)

**Apresentações:** [slideshare.net/juliodelimas](https://slideshare.net/juliodelimas)

Especialista em automação de testes de software, possui formação em Tecnologia da Informação e certificações internacionais (CTFL e CTAL-TM pelo ISTQB) e nacional (CBTS pela ALATS). Atualmente é consultor de automação de testes e instrutor na Qualister. Trabalhou, entre outros, em projetos de teste em Cloud Computing na UOL, em ferramentas de negociação do mercado de ações (MegaBolsa e E-PUMA) na BM&FBovespa, em projetos de telecomunicações na Vivo via Aitec do Brasil e em softwares de gestão pública no Assessor Público. É grande entusiasta da disciplina de testes de software e práticas ágeis.

[twitter.com/juliodelimas](https://twitter.com/juliodelimas)



# Parcerias internacionais

	<p>Soluções para automação, profiling e gestão de testes</p>  <p>TestComplete</p>  <p>ALMComplete</p>  <p>AQtime Pro</p>
	<p>Soluções para testes de performance</p>  <p>WAPT PRO</p> <p>WEB APPLICATION TESTING</p>  <p>WAPT</p>  <p>SERVER SUPERVISOR</p>
	<p>Soluções de apoio a avaliação de usabilidade</p>  <p>Morae</p> <p>TechSmith</p>

A group of six business professionals, three men and three women, are standing in a row and holding a large white rectangular sign. They are all smiling and looking towards the camera. The man on the far left is wearing a blue and white striped short-sleeved shirt and dark trousers. The woman next to him is wearing a white long-sleeved shirt. The man in the center is wearing a white long-sleeved shirt and a grey tie. The woman next to him is wearing a white sleeveless top. The man next to her is wearing a white long-sleeved shirt, a blue tie, and a dark vest. The woman on the far right is wearing a white short-sleeved top with a subtle pattern and dark trousers. The sign they are holding is large and white, with the word "Cucumber" written on it in a bold, black, sans-serif font.

**Cucumber**

- É um framework escrito em Ruby utilizado para automação de testes de aceitação
- Testes escritos com Cucumber descrevem como a aplicação deve se comportar
- A descrição dos comportamentos da aplicação pode ser feitos em diversos idiomas
- Cucumber é escrito em Ruby, mas pode ser utilizado em diversas linguagens de programação, como: Java, C# e Python

- Scripts Cucumber são executados a partir do prompt de comando ou ferramentas de integração contínua
- A descrição dos comportamentos da aplicação é composta basicamente por Funcionalidade, Cenários e Passos do cenário. Esta descrição é feita utilizando Gherkin, um modelo de escrita compreendida pelo Cucumber
- A linguagem usada para escrever as histórias é de fácil compreensão e pode ser escrita por usuários não técnicos



## ***Cucumber pode...***

- ...escrever comportamentos do sistema em texto puro, em diversos idiomas
- ...testar classes e métodos Ruby, Java, C# e Python
- ...executar testes a partir de linha de comando
- ...integrar-se com ferramentas de integração contínua
- ...gerar documentação dos comportamentos da aplicação
- ...gerar relatórios de execução dos testes

# Cucumber: Exemplo de relatório de execução

## Cucumber Features

0 scenarios  
1 step (1 passed)  
Finished in 0m0.021s seconds  
[Collapse All](#) [Expand All](#)

# language: pt

@unidade @fazendo

### Funcionalidade: Gerenciar produtos

Para controlar os produtos da loja  
Como administrador  
Eu quero poder gerenciar produtos no sistema

#### Contexto Posso gerenciar produtos

Dado que posso gerenciar produtos	features/step_definitions/gerenciarprodutos_steps.rb:2
-----------------------------------	--

Executando o cenário: Cadastrar produtos
--

Um Passo foi executado!
-------------------------

@lentos @exemplos @fazendo

#### Esquema do Cenário: Cadastrar produtos

features\gerenciarprodutos.feature:12

Quando informo "<nome>" como nome do produto	features/step_definitions/gerenciarprodutos_steps.rb:6
--	--

E informo R\$ <valor> como valor do produto	features/step_definitions/gerenciarprodutos_steps.rb:10
---	---

E cadastro este produto	features/step_definitions/gerenciarprodutos_steps.rb:20
-------------------------	---

Então vejo a mensagem "Produto cadastrado com sucesso"	features/step_definitions/gerenciarprodutos_steps.rb:24
--	---

#### Exemplos

nome	valor
Ocorre durante a execução de todo o cenário	
Ocorre durante a execução de todo o cenário	

@rapidos

features\gerenciarprodutos.feature:24

#### Cenário: Excluir produtos



## ***Cucumber não...***

- ...executa testes na camada front-end (como Selenium, WebRat, Watir, etc)
- ..é usado para fazer testes unitários
- ...é uma IDE
- ...funciona sem uma linguagem de programação



- Gherkin é a sintaxe utilizada para escrever funcionalidades no Cucumber
- É um domínio específico de linguagem para escrever regras de negócio de fácil leitura
- Sintaxe criada para ser utilizada por usuário técnicos e não técnicos
- Funcionalidades escritas no formato Gherkin também são usadas como documentação do sistema
- Não detalha como a funcionalidade deverá ser implementada

- Cada linha da descrição da funcionalidade é iniciada por uma palavra chave e o escopo é definido utilizando indentação
- Funcionalidades em Gherkin podem ser escritas em português ou em mais de 30 idiomas
- Linhas podem ser comentadas utilizando “#” no início da linha

# Gherkin: Exemplo de funcionalidade

```
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos
    Dado que posso gerenciar produtos
    Quando informo "Camiseta Qualister" como nome do produto
      E informo R$ 99,00 como valor do produto
      E informo 50 como estoque do produto
      E cadastro este produto
    Então vejo a mensagem "Produto cadastrado com sucesso"
```

*# language: pt*

**Funcionalidade:** Descrição da funcionalidade desejada

Para fornecer um benefício e valor ao negócio

Como um utilizador do sistema

Eu quero ter um resultado que me beneficie



Cenário: Uma situação de negócio

Dado que tenho uma precondição


Quando eu executo alguma ação

E executo alguma outra ação

Então recebo uma saída resultante das ações executadas

Mas algo muda no estado atual do sistema

- Os arquivos são salvos com a extensão .feature
- Funcionalidades podem múltiplos cenários e cada cenário é iniciado pela palavra chave “Cenário”
- Cada cenário possui passos, sempre iniciados pelas palavras chave: Dado, Quando, Então, E e Mas

A group of six diverse professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held by the first four people from the left. The text on the sign is in a bold, black, sans-serif font.

# **Uma breve introdução à linguagem Ruby**

## ***Características do Ruby***

Fonte: <http://pt.wikipedia.org/wiki/Ruby>

- Todas as variáveis são objetos
- Através do RubyGems, é possível instalar e atualizar bibliotecas com uma linha de comando
- Tipagem dinâmica, mas forte. Isso significa que todas as variáveis devem ter um tipo (fazer parte de uma classe), mas a classe pode ser alterada dinamicamente
- Ruby está disponível para diversas plataformas, como Microsoft Windows, Linux, Solaris e Mac OS X, além de também ser executável em cima da máquina virtual Java (JRuby) e Microsoft .NET (IronRuby)
- Não existem "tipos primitivos" em Ruby; todos os tipos são classes: tais como inteiro, real, entre outros

## ***Características do Ruby***

Fonte: <http://pt.wikipedia.org/wiki/Ruby>

- Uma variável local é declarada normalmente.
- Uma variável de instância é declarada com um "@" no nome.
- Uma variável de classe é declarada com "@@"
- Uma variável global é declarada com "\$".
- Variáveis que iniciam com uma letra maiúscula são constantes.
- Strings simples são declaradas entre apóstrofes ou aspas duplas
- Strings também podem ser escritas utilizando %q{} ex. %q{Isto é uma string de aspas simples}

# Uma breve introdução à linguagem Ruby

## Condicionais

```
if [condição]
  # linhas de comando
else
  # linhas de comando
elsif
  # linhas de comando
end
```

## Vetores

```
linguagens = Array.new
linguagens << "Ruby"
linguagens << "Java"
linguagens << "PHP"

puts linguagens[0]
# imprime "Ruby"

linguagens.each do |linguagem|
  puts linguagem
end
# imprime "Ruby" "Java" "PHP"
```

## Variáveis e Atributos

Atributos da Classe  
@atributo = "Valor"

Variáveis  
nome = %q{Júlio}

Mostrar variáveis dentro de Strings  
puts "Meu nome é #{nome}"  
# imprime "Meu nome é Júlio"

## Hashes

```
pessoa = Hash.new

pessoa["nome"] = "Júlio"
pessoa["idade"] = 27
pessoa["sexo"] = 49.90

puts pessoa["nome"]
puts pessoa["idade"]
puts pessoa["sexo"]

# imprime "Júlio"
# imprime 27
# imprime 49.90
```

## Métodos

```
def somar(num1, num2)
  # escopo do método
  puts num1 + num2
end

# utilização
somar(1, 2)
# imprime "3"
```

## Classes

```
class Exemplo
  def initialize
    # escopo do construtor
  end
  def metodo
    # escopo do método
    puts "Olá"
  end
end

# utilização
exemplo1 = Exemplo.new
exemplo1.metodo
# imprime "Olá"
```

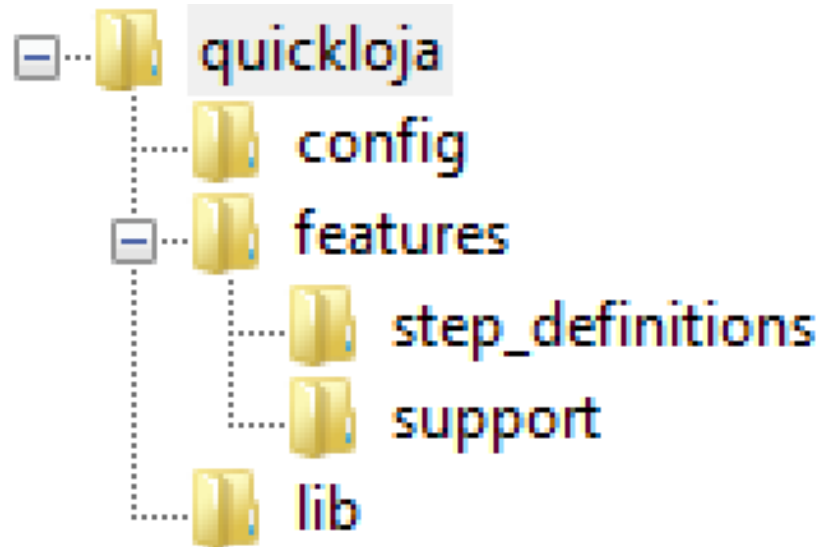


A group of six diverse professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held by all of them and has the text "Exercícios Práticos" written on it in a bold, black, sans-serif font. The background is a plain, light color.

# Exercícios Práticos

## Exercício: Criando a estrutura de diretórios

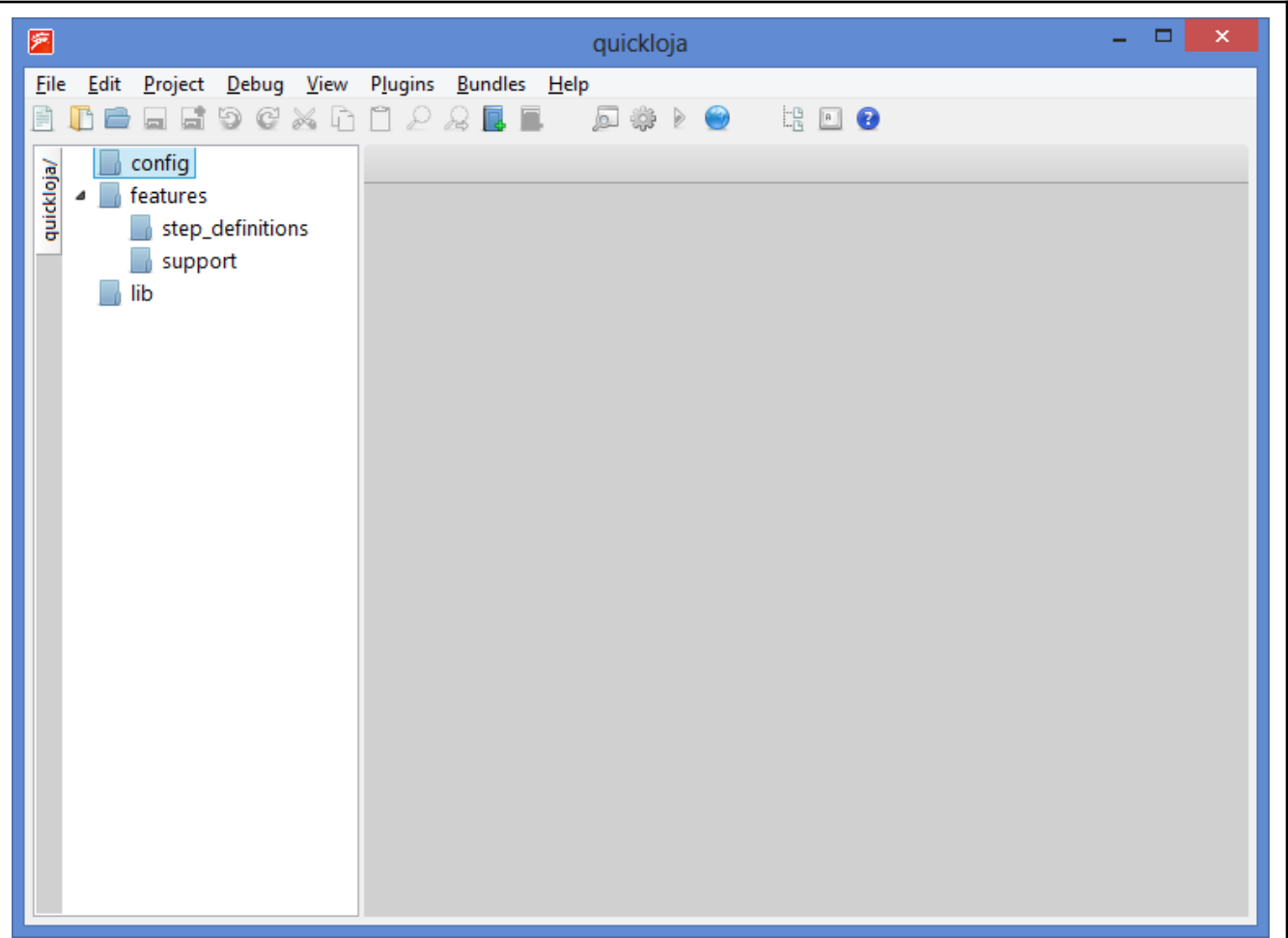
- 1) Criar um diretório chamado “quickloja” na raiz do disco seguindo a estrutura proposta ao lado





# Exercício: Abrindo o projeto no Redcar Editor

- 1) Abrir o prompt de comando e digite **redcar** para abrir o editor
- 2) Clique em “File”, e em “Open directory”
- 3) Selecione o diretório “quickloja” e pressione “Ok”



## Exercício: Criando o arquivo de configurações de ambiente

- 1) Expandir o diretório “features”
- 2) Clicar com o botão direito sobre o diretório “support”
- 3) Selecionar a opção “New File”
- 4) Dar o nome “env.rb” ao arquivo criado
- 5) Abrir o arquivo que foi criado, utilizando duplo clique
- 6) Coloque na primeira linha o código # encoding: utf-8 teremos caracteres latinos
- 7) Digitar o “require” para disponibilizar a formatação dos caracteres latinos

```
1 # encoding: utf-8
2 require 'cucumber/formatter/unicode'
```

# Exercício: Escrevendo a primeira Funcionalidade (Feature)

- 1) Clique com o botão direito do mouse sobre o diretório “features”
- 2) Selecione a opção “New file”
- 3) Coloque o nome “gerenciarprodutos.feature”
- 4) Abra o arquivo que foi criado, utilizando duplo clique
- 5) Coloque na primeira linha o código # language: pt simbolizando que iremos escrever em português
- 6) Escreva a Funcionalidade segundo a imagem ao lado

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Cenário: Cadastrar produtos
8     Dado que posso gerenciar produtos
9     Quando informo "Camiseta Qualister" como nome do produto
10      E informo R$ 99,00 como valor do produto
11      E informo 50 como estoque do produto
12      E cadastro este produto
13      Então vejo a mensagem "Produto cadastrado com sucesso"
```

# Exercício: Escrevendo a primeira Funcionalidade (Feature)

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd C:\quickloja

C:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
ar_produtos.feature:7
    Dado que posso gerenciar produtos # features\gerencia
r_produtos.feature:8
    Quando informo "Camiseta Qualister" como nome do produto # features\gerencia
r_produtos.feature:9
    E informo R$ 99.00 como valor do produto # features\gerencia
r_produtos.feature:10
    E informo 50 como estoque do produto # features\gerencia
r_produtos.feature:11
    E cadastro este produto # features\gerencia
r_produtos.feature:12
    Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerencia
ar_produtos.feature:13

1 cenário (1 undefined)
6 steps (6 undefined)
0m0.007s

You can implement step definitions for undefined steps with these snippets:

Dado(/^que posso gerenciar produtos$/ do
  pending # express the regexp above with the code you wish you had
end

Quando(/^informo "(.*)" como nome do produto$/ do |arg1|
  pending # express the regexp above with the code you wish you had
end

Quando(/^informo R$(\d+),(\d+) como valor do produto$/ do |arg1, arg2|
  pending # express the regexp above with the code you wish you had
end

Quando(/^informo (\d+) como estoque do produto$/ do |arg1|
  pending # express the regexp above with the code you wish you had
end

Quando(/^cadastro este produto$/ do
  pending # express the regexp above with the code you wish you had
end

Então(/^vejo a mensagem "(.*)"$/ do |arg1|
  pending # express the regexp above with the code you wish you had
end
```

# Exercício: Definindo Passos do Cenário Cadastrar Produtos

- 1) Clique com o botão direito do mouse sobre o diretório “step\_definitions”
- 2) Selecione a opção “New file”
- 3) Coloque o nome “gerenciarprodutos\_steps.rb”
- 4) Abra o arquivo que foi criado, utilizando duplo clique
- 5) Coloque na primeira linha o código # encoding: utf-8 teremos caracteres latinos
- 6) Escreva o Passo segundo a ponta a imagem ao lado

```
1 # encoding: utf-8
2 Dado /que posso gerenciar produtos/ do
3   @produto = Produto.new
4 end
```

- A descrição do Passo deve estar entre “/”
- O “do” e “end” é o escopo do que será ocorrerá quando o passo for executado
- @produto é uma variável que estará disponível em todos os passos
- Se alguma linha lançar alguma exceção o passo será considerado “Falho”

# Exercício: Executando as Funcionalidades do projeto

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar todas as Funcionalidades contidas neste projeto
- 4) O Cucumber apresenta um resumo do que foi executado e quais falhas ocorreram
- 5) 1 cenário falhou e outros 5 Passos não foram definidos
- 6) O primeiro Passo possui uma falha: Não existe o objeto Produto, o que quer dizer que teremos que criar esta Classe

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
initions\gerenciarprodutos_steps.rb:1
      uninitialized constant Object::Produto (NameError)
      ./features/step_definitions/gerenciarprodutos_steps.rb:2:in `que posso ca
dastrar produtos/'
      features\gerenciarprodutos.feature:8:in `Dado que posso cadastrar produtos
'
      Quando informo "Camiseta Qualister" como nome do produto # features\gerencia
rprodutos.feature:9
        E informo R$ 99,00 como valor do produto # features\gerencia
rprodutos.feature:10
        E informo 50 como estoque do produto # features\gerencia
rprodutos.feature:11
        E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
      Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

Failing Scenarios:
cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos

1 cenário (1 failed)
6 steps (1 failed, 5 undefined)
0m0.015s
```

# Exercício: Criando a classe Produto

- 1) Clique com o botão direito do mouse sobre o diretório "lib"
- 2) Selecione a opção "New file"
- 3) Coloque o nome "produto.rb"
- 4) Abra o arquivo que foi criado, utilizando duplo clique
- 5) Coloque na primeira linha o código # encoding: utf-8 teremos caracteres latinos
- 6) Vamos criar a classe Produto segundo a imagem ao lado

```
1 # encoding: utf-8
2 class Produto
3
4 end
```

- 7) Adicione as linhas 4 e 5 da imagem ao lado no arquivo env.rb para poder utilizar a classe Produto

```
1 # encoding: utf-8
2 require 'cucumber/formatter/unicode'
3
4 $:.unshift('c:/quickloja/lib')
5 require 'produto'
```

# Exercício: Executando as Funcionalidades novamente

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o primeiro Passo obteve sucesso
- 5) Agora iremos definir o segundo Passo

```
C:\Windows\system32\cmd.exe

Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
initions\gerenciarprodutos_steps.rb:1
    Quando informo "Camiseta Qualister" como nome do produto # features\gerencia
rprodutos.feature:9
    E informo R$ 99,00 como valor do produto # features\gerencia
rprodutos.feature:10
    E informo 50 como estoque do produto # features\gerencia
rprodutos.feature:11
    E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

1 scenario <1 undefined>
6 steps <5 undefined, 1 passed>
0m0.008s
```




# Exercício: Definindo passos com valores variáveis

- 1) O segundo passo possui um valor variável, por é necessário utilizar uma expressão regular para extraí-lo
- 2) A expressão regular `(.*?)` extraí um conjunto alfanumérico de caracteres
- 3) Este valor é colocado em uma variável, localizada entre “pipes” e após o comando “do”
- 4) Esta variável é válida apenas dentro do escopo do Passo

## Dicas:

- O tipo do valor extraído pela expressão regular sempre será String

```
6 Quando /informo "(.*?)" como nome do produto/ do |nome|
7   @produto.informarNome(nome)
8 end
9
```



- O valor extraído pela expressão regular é colocado dentro da variável “nome”

- Este passo adiciona ao atributo nome, da classe Produto, o valor que foi extraído da frase

# Exercício: Executando as Funcionalidades novamente

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o segundo Passo falhou
- 5) Isso ocorre porque o atributo "nome" não existe na classe Produto
- 6) Vamos criá-lo e executar novamente a funcionalidade

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
initions\gerenciarprodutos_steps.rb:1
    Quando informo "Camiseta Qualister" como nome do produto # features/step_def
initions\gerenciarprodutos_steps.rb:5
      undefined method `nome=' for #<Produto:0x2e1c628> (NoMethodError)
      ./features/step_definitions/gerenciarprodutos_steps.rb:6:in `informo "<.*
?>" como nome do produto/'
      features\gerenciarprodutos.feature:9:in `Quando informo "Camiseta Qualiste
r" como nome do produto'
    E informo R$ 99,00 como valor do produto # features\gerencia
rprodutos.feature:10
    E informo 50 como estoque do produto # features\gerencia
rprodutos.feature:11
    E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

Failing Scenarios:
cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos

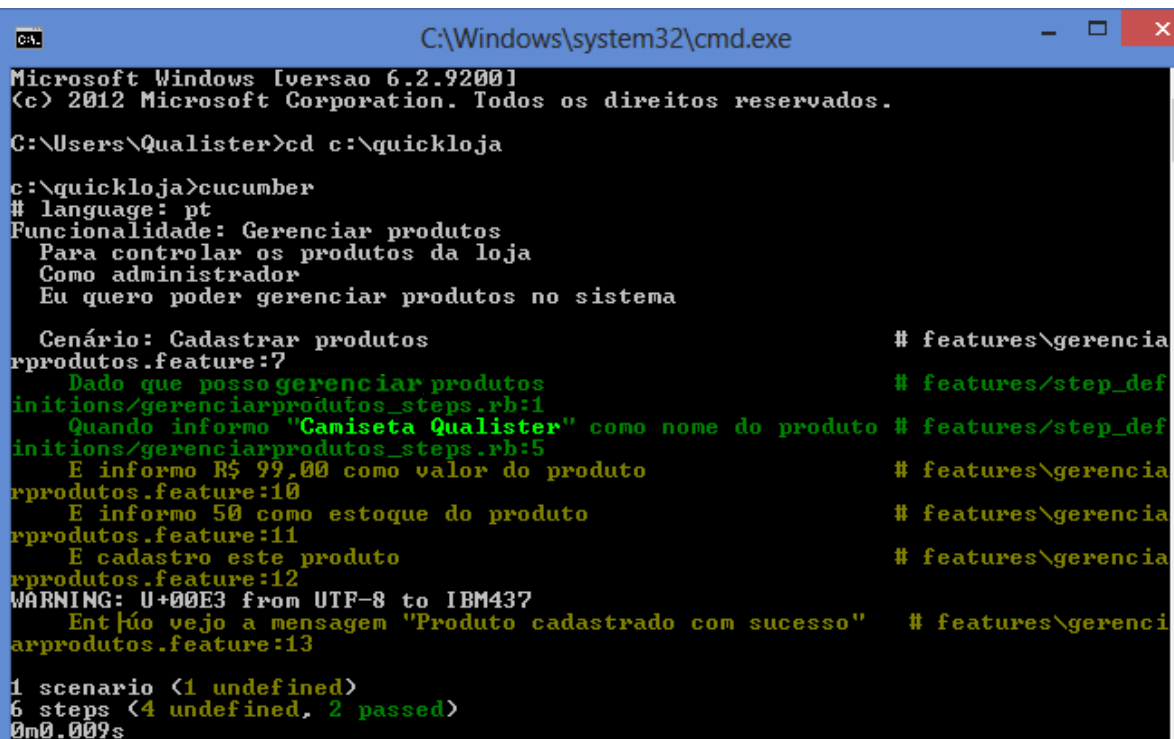
1 scenario (1 failed)
6 steps (1 failed, 4 undefined, 1 passed)
0m0.010s
```

# Exercício: Alterando a classe Produto

- 1) Abra o arquivo produto.rb contido no diretório "lib" e adicione o atributo nome à classe Produto conforme imagem ao lado

```
1 class Produto
2   attr_accessor :nome
3 end
```

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o segundo Passo obteve sucesso
- 5) Agora iremos definir os próximos passos iniciados pela palavra chave "E"



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7          # features\step_def
  Dado que posso gerenciar produtos # features/step_def
initions/gerenciarprodutos_steps.rb:1
  Quando informo "Camiseta Qualister" como nome do produto # features/step_def
initions/gerenciarprodutos_steps.rb:5
  E informo R$ 99,00 como valor do produto # features\gerencia
rprodutos.feature:10
  E informo 50 como estoque do produto # features\gerencia
rprodutos.feature:11
  E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
  Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

1 cenário (1 undefined)
6 passos (4 undefined, 2 passed)
0m0.000s
```

# Exercício: Definindo passos com valores variáveis numéricos

- 1) O terceiro passo possui um valor variável numérico, mas por possuir virgula utilizaremos a expressão regular usada para extrair um conjunto alfanumérico de caracteres
- 2) O quarto passo é um valor variável numérico, por isso posso extrair o valor usando uma expressão regular que captura dígitos (`\d+`)
- 3) Mesmo usando a expressão regular (`\d+`) ainda preciso converter o valor contido na variável `quantidade` para inteiro `quantidade.to_i`

```
1 Dado /que posso gerenciar produtos/ do
2   @produto = Produto.new
3 end
4
5 Quando /informo "(.+?)" como nome do produto/ do |nome|
6   @produto.nome = nome
7 end
8
9 E /informo R\$ (.+?) como valor do produto/ do |valor|
10  @produto.valor = valor
11 end
12
13 E /informo (\d+) como estoque do produto/ do |quantidade|
14  @produto.quantidade = quantidade.to_i
15 end
```

- Na linha 9, vemos que o símbolo “\$” é precedido por \ isto é um pré-requisito para qualquer caractere utilizado pelo Ruby

# Exercício: Executando as Funcionalidades novamente

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o terceiro e Quarto passos falharam e o quinto aguarda que todos anteriores a ele passem
- 5) Vamos adicionar o atributo para continuar

```
C:\Windows\system32\cmd.exe

Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
initions/gerenciarprodutos_steps.rb:1
    Quando informo "Camiseta Qualister" como nome do produto # features/step_def
initions/gerenciarprodutos_steps.rb:5
    E informo R$ 99,00 como valor do produto # features/step_def
initions/gerenciarprodutos_steps.rb:9
    undefined method `valor=' for #<Produto:0x29ac660 @nome="Camiseta Qualiste
r"> (NoMethodError)
    ./features/step_definitions/gerenciarprodutos_steps.rb:10:in `informo R\
$
(.*?) como valor do produto/'
    features\gerenciarprodutos.feature:10:in `E informo R$ 99,00 como valor do
produto'
    E informo 50 como estoque do produto # features/step_def
initions/gerenciarprodutos_steps.rb:13
    E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

Failing Scenarios:
cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos

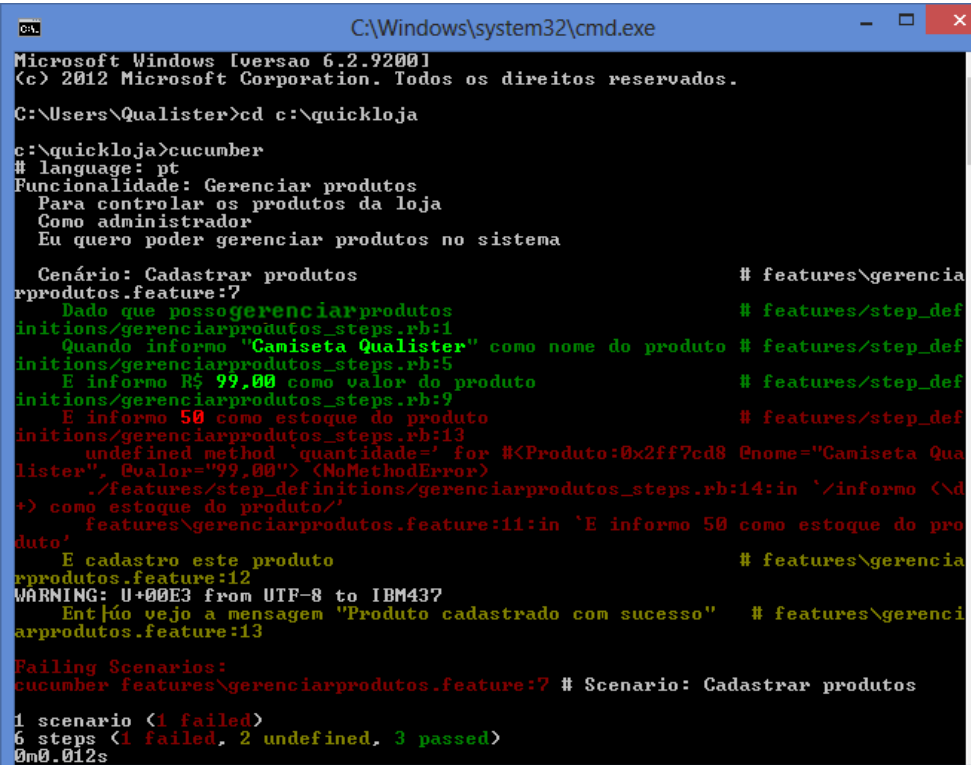
1 scenario (1 failed)
6 steps (1 failed, 1 skipped, 2 undefined, 2 passed)
0m0.018s
```

# Exercício: Alterando a classe Produto

- 1) Abra o arquivo produto.rb contido no diretório "lib" e adicione o atributo valor à classe Produto conforme imagem ao lado

```
1 class Produto
2   attr_accessor :nome
3   attr_accessor :valor
4 end
```

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o terceiro Passo obteve sucesso, mas o quarto falhou
- 5) Precisamos agora adicionar o atributo quantidade à classe Produto



```
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
  Dado que posso gerenciar produtos # features/step_def
initions/gerenciarprodutos_steps.rb:1
  Quando informo "Camiseta Qualister" como nome do produto # features/step_def
initions/gerenciarprodutos_steps.rb:5
  E informo R$ 99,00 como valor do produto # features/step_def
initions/gerenciarprodutos_steps.rb:9
  E informo 50 como estoque do produto # features/step_def
initions/gerenciarprodutos_steps.rb:13
  undefined method `quantidade=' for #<Produto:0x2ff7cd8 @nome="Camiseta Qua
lister", @valor="99,00"> (NoMethodError)
./features/step_definitions/gerenciarprodutos_steps.rb:14:in `informo (<d
+> como estoque do produto'
features\gerenciarprodutos.feature:11:in `E informo 50 como estoque do pro
duto'
  E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

Failing Scenarios:
cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos

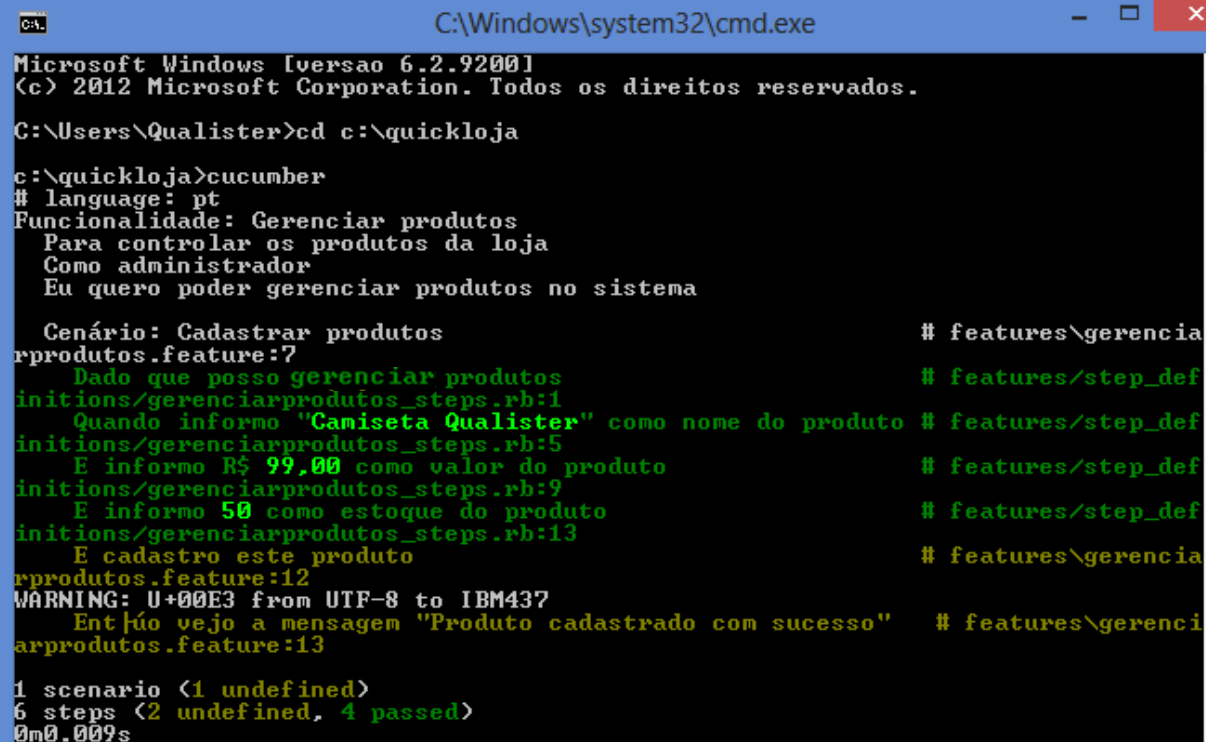
1 scenario (1 failed)
6 steps (1 failed, 2 undefined, 3 passed)
0m0.012s
```

# Exercício: Alterando a classe Produto

- 1) Abra o arquivo produto.rb contido no diretório "lib" e adicione o atributo quantidade à classe Produto conforme imagem ao lado

```
1 class Produto
2   attr_accessor :nome
3   attr_accessor :valor
4   attr_accessor :quantidade
5 end
```

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o quarto Passo obteve sucesso
- 5) Vamos agora definir o quinto passo



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
    initions/gerenciarprodutos_steps.rb:1
    Quando informo "Camiseta Qualister" como nome do produto # features/step_def
    initions/gerenciarprodutos_steps.rb:5
    E informo R$ 99.00 como valor do produto # features/step_def
    initions/gerenciarprodutos_steps.rb:9
    E informo 50 como estoque do produto # features/step_def
    initions/gerenciarprodutos_steps.rb:13
    E cadastro este produto # features\gerencia
rprodutos.feature:12
WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

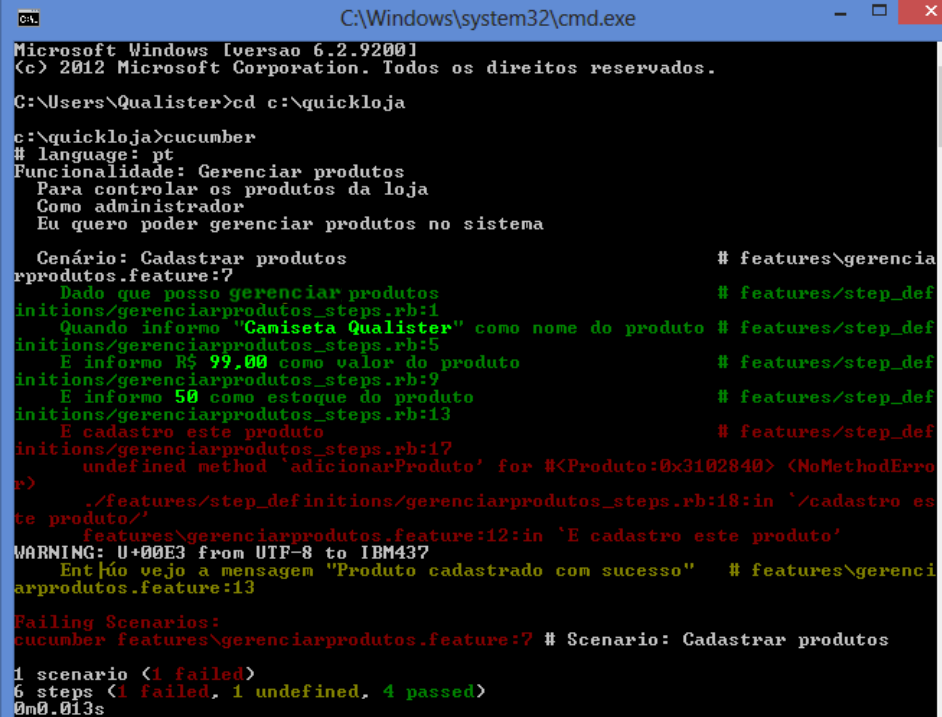
1 scenario (1 undefined)
6 steps (2 undefined, 4 passed)
0m0.009s
```

# Exercício: Definindo o quinto Passo do Cenário

- 1) Abra o arquivo `gerenciarprodutos_steps.rb` e adicione a definição do Passo, conforme a imagem ao lado
- 2) O retorno será armazenado na variável `@resultado`

```
17 E /cadastro este produto/ do
18   @resultado = @produto.cadastrarProduto()
19 end
```

- 1) Abra o prompt de comando
- 2) Digite `cd` [diretório\_do\_projeto] e pressione Enter
- 3) Digite `cucumber` e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o quinto Passo falhou, pois o método `cadastrarProduto` ainda foi definido na classe `Produto`
- 5) Vamos definí-lo



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
initions/gerenciarprodutos_steps.rb:1
    Quando informo "Camiseta Qualister" como nome do produto # features/step_def
initions/gerenciarprodutos_steps.rb:5
    E informo R$ 99,00 como valor do produto # features/step_def
initions/gerenciarprodutos_steps.rb:9
    E informo 50 como estoque do produto # features/step_def
initions/gerenciarprodutos_steps.rb:13
    E cadastro este produto # features/step_def
initions/gerenciarprodutos_steps.rb:17
    undefined method 'adicionarProduto' for #<Produto:0x3102840> (NoMethodError)
    ./features/step_definitions/gerenciarprodutos_steps.rb:18:in '/cadastro es
te produto/'
    features\gerenciarprodutos.feature:12:in 'E cadastro este produto'
WARNING: U+00E3 from UTF-8 to IBM437
Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
arprodutos.feature:13

Failing Scenarios:
cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos

1 scenario (1 failed)
6 steps (1 failed, 1 undefined, 4 passed)
0m0.013s
```

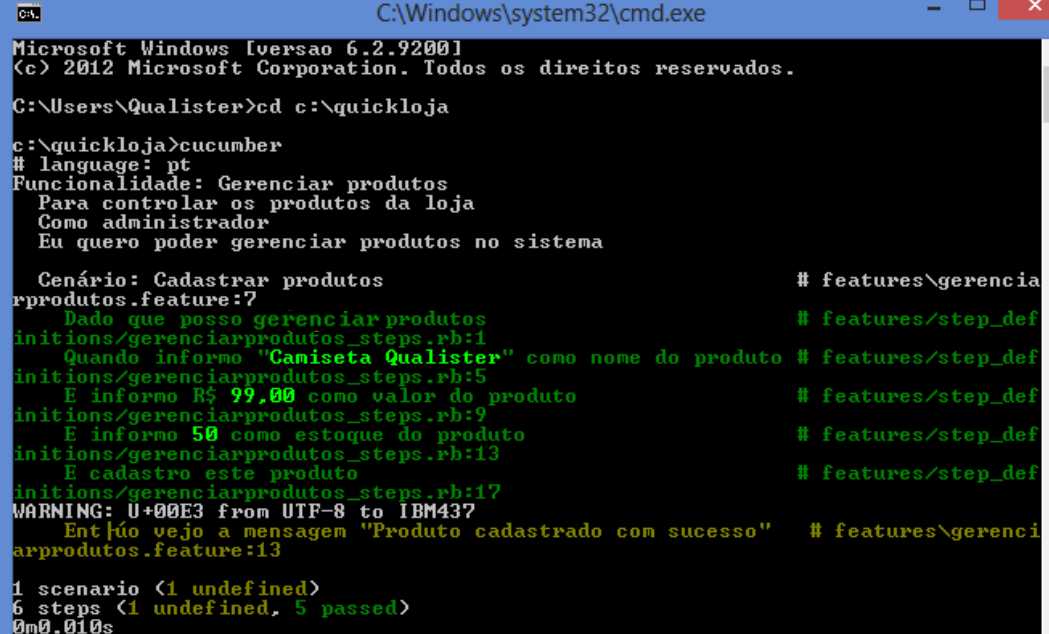


# Exercício: Alterando a classe Produto

- 1) Abra o arquivo produto.rb contido no diretório "lib" e adicione os métodos initialize e cadastrarProduto à classe Produto conforme imagem ao lado
- 2) O método retorna um erro ou uma mensagem de sucesso

```
6 def initialize
7   @produtos = Array.new
8 end
9
10 def cadastrarProduto
11   qtdprodutos = @produtos.count
12   @produtos << {:nome => @nome, :valor => @valor, :quantidade => @quantidade}
13   if @produtos.count > qtdprodutos
14     return "Produto cadastrado com sucesso"
15   else
16     raise "Falha ao cadastrar"
17   end
18 end
```

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que o quinto Passo obteve sucesso



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema


  Cenário: Cadastrar produtos # features\gerencia
    rprodutos.feature:7 # features/step_def
      Dado que posso gerenciar produtos # features/step_def
    initions/gerenciarprodutos_steps.rb:1
      Quando informo "Camiseta Qualister" como nome do produto # features/step_def
    initions/gerenciarprodutos_steps.rb:5
      E informo R$ 99,00 como valor do produto # features/step_def
    initions/gerenciarprodutos_steps.rb:9
      E informo 50 como estoque do produto # features/step_def
    initions/gerenciarprodutos_steps.rb:13
      E cadastro este produto # features/step_def
    initions/gerenciarprodutos_steps.rb:17
    WARNING: U+00E3 from UTF-8 to IBM437
      Então vejo a mensagem "Produto cadastrado com sucesso" # features\gerenci
    arprodutos.feature:13

1 scenario <1 undefined>
6 steps <1 undefined, 5 passed>
0m0.010s
```

## Exercício: Definindo o sexto Passo do Cenário


- 1) No sexto passo iremos verificar se o resultado do cadastro do produto foi igual à mensagem extraída do Passo
- 2) O método “expect” pertence ao framework Rspec Expectations, e serve para validar condições lançar uma exceção caso esta resulte em uma falha

```
32 Então /a mensagem "(.*?)" é apresentada/ do |mensagem|
33   expect(@resultado).to eq(mensagem)
34 end
```



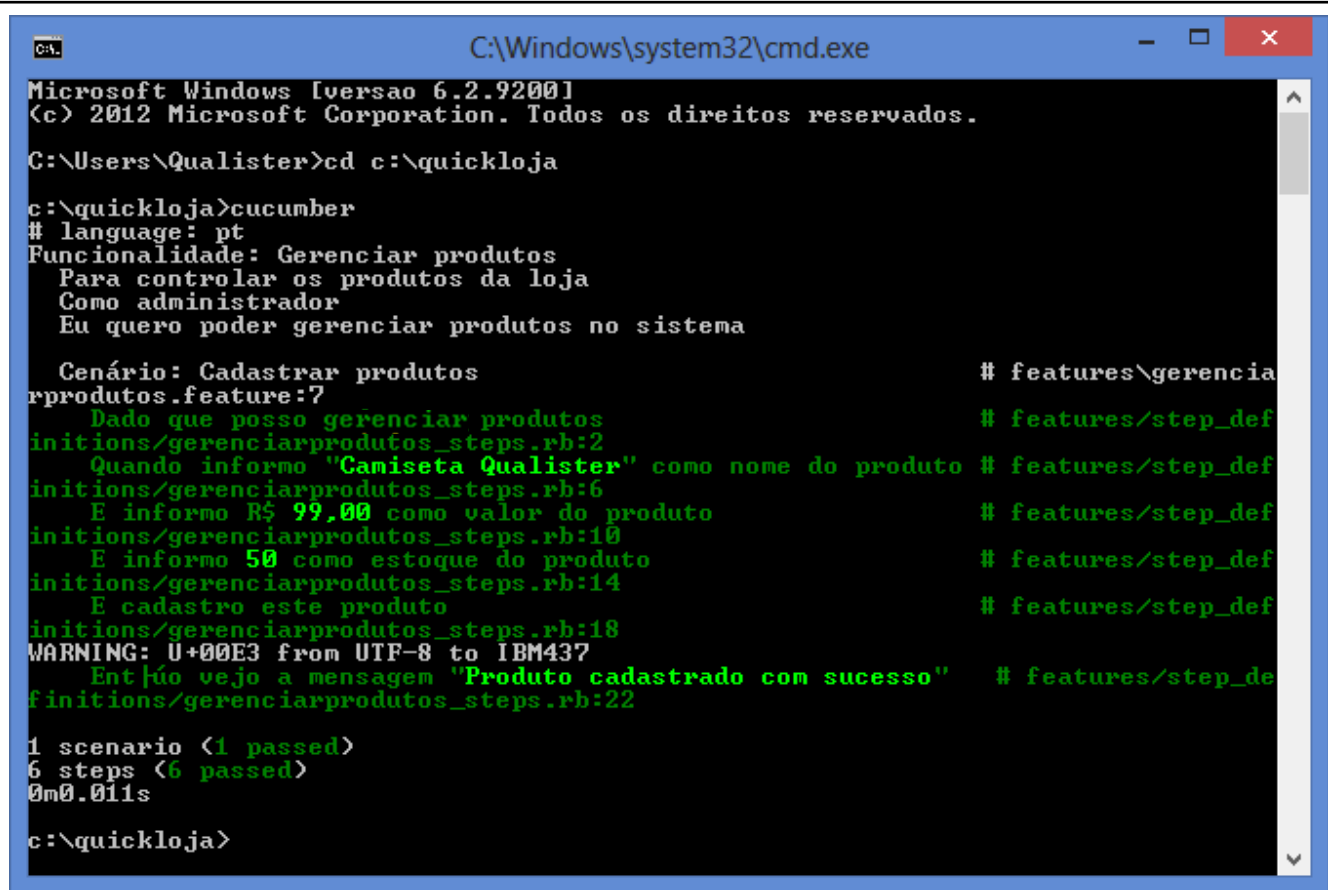
- 1) Para que os métodos validadores possam ser utilizados é necessário requisitar o framework do Rspec Expectations
- 2) Abra o arquivo “env.rb” e adicione a linha 7 a este arquivo, conforme imagem ao lado
- 3) Agora vamos executar a funcionalidade novamente

```
1 # encoding: utf-8
2 require 'cucumber/formatter/unicode'
3
4 $:.unshift('c:/quickloja/lib')
5 require 'produto'
6
7 require 'rspec/expectations'
```



# Exercício: Sucesso ao executar a funcionalidade

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez o relatório exibido pelo Cucumber mostrará que todos os passos pertencentes a este cenário foram executados corretamente



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.


C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
rprodutos.feature:7
    Dado que posso gerenciar produtos # features/step_def
initions/gerenciarprodutos_steps.rb:2
    Quando informo "Camiseta Qualister" como nome do produto # features/step_def
initions/gerenciarprodutos_steps.rb:6
    E informo R$ 99,00 como valor do produto # features/step_def
initions/gerenciarprodutos_steps.rb:10
    E informo 50 como estoque do produto # features/step_def
initions/gerenciarprodutos_steps.rb:14
    E cadastro este produto # features/step_def
initions/gerenciarprodutos_steps.rb:18
WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_de
finitions/gerenciarprodutos_steps.rb:22

1 scenario (1 passed)
6 steps (6 passed)
0m0.011s

c:\quickloja>
```

A group of six diverse professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held up by their hands and contains the text 'Usando Verificadores (Matchers) do Rspec para validar resultados'. The background is a plain, light color.

**Usando Verificadores  
(Matchers) do Rspec para  
validar resultados**

## Exercício: Usando Verificadores (Matchers) do Rspec para validar resultados

1) Verificador de equivalência	<pre>expect("qualister").to eq("qualister")</pre> <b>Negativa:</b> <pre>expect("qualister").not_to eq("qualister")</pre>
2) Verificador de expressão regular	<pre>expect("qualister").to match(/expressao/)</pre>
3) Verificador de vetores	<pre>expect([1,2,3]).to match_array([3,2,1])</pre>
4) Verificador de alcance (range)	<pre>expect(0..5).to cover(2) <b>#somente Ruby &gt;= 1.9</b></pre>
4) Verificadores comparativos	<pre>expect(10).to be &gt; 5 expect(5).to be &gt;= 5 expect(1).to be &lt; 5 expect(10).not_to be &lt;= 5</pre>
5) Verificadores booleanos	<pre>expect(true).to be_true expect(false).to be_false expect(nil).to be_nil</pre>
5) Verificadores de coleção	<pre>expect("qualister").to include("lis") expect("qualister").to start_with("q") expect("qualister").to end_with("er")</pre>

A group of six smiling professionals, three men and three women, are standing behind a large white rectangular sign. They are dressed in business casual attire. The man on the far left is wearing a blue and white striped shirt and a blue tie. The woman next to him is wearing a white blouse. The man in the center is wearing a white shirt and a grey tie. The woman next to him is wearing a white sleeveless top. The man next to her is wearing a white shirt, a blue tie, and a dark vest. The woman on the far right is wearing a white patterned top and dark trousers. The sign they are holding is white and has the text "Reaproveitando Passos" written on it in a bold, black, sans-serif font.

**Reaproveitando Passos**



# Exercício: Utilizando Passos prontos para criar novos Cenários

É possível reaproveitar os passos que já foram escritos, alterando os valores variáveis neles contidos

- 1) Abra o arquivo gerenciarprodutos.feature
- 2) Copie as linhas correspondentes ao cenário "Cadastrar produtos" e cole abaixo dele
- 3) Altere o nome do Cenário para "Cadastrar outro produto"
- 4) Altere os valores variáveis contidos nos passos e execute a Funcionalidade novamente

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Cenário: Cadastrar produtos
8     Dado que posso gerenciar produtos
9     Quando informo "Camiseta Qualister" como nome do produto
10      E informo R$ 99,00 como valor do produto
11      E informo 50 como estoque do produto
12      E cadastro este produto
13     Então vejo a mensagem "Produto cadastrado com sucesso"
14
15   Cenário: Cadastrar outro produto
16     Dado que posso gerenciar produtos
17     Quando informo "Boné Qualister" como nome do produto
18      E informo R$ 35,90 como valor do produto
19      E informo 100 como estoque do produto
20      E cadastro este produto
21     Então vejo a mensagem "Produto cadastrado com sucesso"
```

# Exercício: Utilizando Passos prontos para criar novos Cenários

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Podemos ver nos resultados que desta vez 2 Cenários e 12 Passos foram executados e passaram

## Dica:

- Em casos em que Passos já existentes são utilizados não é necessário definí-los novamente

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja


c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Cenário: Cadastrar produtos # features\gerencia
    produtos.feature:7
      Dado que posso gerenciar produtos # features/step_def
        initions/gerenciarprodutos_steps.rb:2
        Quando informo "Camiseta Qualister" como nome do produto # features/step_def
          initions/gerenciarprodutos_steps.rb:6
          E informo R$ 99,00 como valor do produto # features/step_def
            initions/gerenciarprodutos_steps.rb:10
            E informo 50 como estoque do produto # features/step_def
              initions/gerenciarprodutos_steps.rb:14
              E cadastro este produto # features/step_def
                initions/gerenciarprodutos_steps.rb:18
                WARNING: U+00E3 from UTF-8 to IBM437
                Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_def
                initions/gerenciarprodutos_steps.rb:22

      Cenário: Cadastrar outro produto # features\gerenciarp
        rodutos.feature:15
          Dado que posso gerenciar produtos # features/step_defin
            itions/gerenciarprodutos_steps.rb:2
            Quando informo "Boné Qualister" como nome do produto # features/step_defin
              itions/gerenciarprodutos_steps.rb:6
              E informo R$ 35,90 como valor do produto # features/step_defin
                itions/gerenciarprodutos_steps.rb:10
                E informo 100 como estoque do produto # features/step_defin
                  itions/gerenciarprodutos_steps.rb:14
                  E cadastro este produto # features/step_defin
                    itions/gerenciarprodutos_steps.rb:18
                    WARNING: U+00E3 from UTF-8 to IBM437
                    Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defi
                    initions/gerenciarprodutos_steps.rb:22

2 scenarios (2 passed)
12 steps (12 passed)
0m0.024s
```



A group of six diverse professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held by all of them, with their hands visible at the top and sides. The background is a plain, light color.

**Usando Esquema do  
Cenário e Exemplos**

# Exercício: Utilizando Exemplos e Esquema do Cenário

- **Exemplos** são conjuntos de valores contidos em uma tabela
- Os dados de cada uma das linhas são passados para dentro de um cenário, chamado **Esquema do Cenário**
- Usando estes recursos é possível executar um mesmo Cenário com dados diferentes sem precisar escrever o Cenário e os Passos diversas vezes
- Vamos implementar este recurso

- 1) Remova o Cenário “Cadastrar outro produto” e seus Passos
- 2) Altere a palavra chave “Cenário” para “Esquema do Cenário”
- 3) Crie a tabela de Exemplos com os dados que serão utilizados
- 4) Altere os dados fixos nos Passos pelo título das colunas entre <>

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Esquema do Cenário: Cadastrar produtos
8     Dado que posso gerenciar produtos
9     Quando informo "<nome>" como nome do produto
10      E informo R$ <valor> como valor do produto
11      E informo <quantidade> como estoque do produto
12      E cadastro este produto
13      Então vejo a mensagem "Produto cadastrado com sucesso"
14
15 Exemplos:
16   | nome           | valor  | quantidade |
17   | Boné Qualister | 150,50 | 50          |
18   | Camiseta Qualister | 99,00 | 100         |
19
```

# Exercício: Utilizando Exemplos e Esquema do Cenário

- Quando Esquemas de Cenários são executados os linha executadas que passaram são exibidas em verde
- Os totalizadores continuam calculando quantas vezes o Cenário foi executado e quantas vezes os Passos do Cenário foram executados

## Dica:

- É possível descrever outros Cenários dentro desta mesma funcionalidade, mas apenas o Esquema do Cenário se repetirá usando os dados da tabela de Exemplos

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Esquema do Cenário: Cadastrar produtos
  produtos.feature:7
    Dado que posso gerenciar produtos
    itions/gerenciarprodutos_steps.rb:2
    Quando informo "<nome>" como nome do produto
    itions/gerenciarprodutos_steps.rb:6
    E informo R$ <valor> como valor do produto
    itions/gerenciarprodutos_steps.rb:10
    E informo <quantidade> como estoque do produto
    itions/gerenciarprodutos_steps.rb:14
    E cadastro este produto
    itions/gerenciarprodutos_steps.rb:18
    WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso"
    nitions/gerenciarprodutos_steps.rb:22

    Exemplos:
    | nome           | valor  | quantidade |
    | Boné Qualister | 150,50 | 50          |
    | Camiseta Qualister | 99,00 | 100         |

  2 scenarios (2 passed)
  12 steps (12 passed)
  0m0.022s
```

A group of six business professionals, three men and three women, are standing in a row and holding a large white rectangular sign. They are all smiling and looking towards the camera. The man on the far left is wearing a blue and white striped shirt and a blue tie. The woman next to him is wearing a white blouse. The man in the center is wearing a white shirt and a grey tie. The woman next to him is wearing a white sleeveless top. The man next to her is wearing a white shirt, a blue tie, and a dark vest. The woman on the far right is wearing a white patterned top and dark trousers. The sign they are holding is white and has the text "Refatorando código" written on it in a bold, black, sans-serif font.

**Refatorando código**

# Exercício: Refatorando o código

- Precisamos validar se os valores que estão sendo informados não estão vazios antes deles serem cadastrados
  - Uma forma simples de fazer isso é seguir recomendações da orientação a objetos criando métodos para atribuir valores a variáveis da classe:
    - a. `informarNome`
    - b. `informarValor`
    - c. `informarQuantidade`
- 1) Retire os `attr_accessors` da classe `Produto`
  - 2) Adicione os métodos conforme a imagem ao lado

```
18 def informarNome(nome)
19   raise "Informe o Nome" if nome.empty?
20   @nome = nome
21 end
22
23 def informarValor(valor)
24   raise "Informe o Valor" if valor.empty?
25   @valor = valor
26 end
27
28 def informarQuantidade(quantidade)
29   raise "Informe uma Quantidade maior que 0" if quantidade <= 0
30   @quantidade = quantidade
31 end
```

# Exercício: Executando a Funcionalidade após a refatoração

- 1) Abra o prompt de comando
  - 2) Digite cd [diretório\_do\_projeto] e pressione Enter
  - 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- Vamos ver que ocorreram falhas ao executar os testes, isso porque os Passos continuam a atribuir valores utilizando os "attr\_accessors"
  - Vemos que a falha aponta um erro na linha 7 do arquivo "gerenciarprodutos\_steps.rb"
  - Vamos refatorar os Passos

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Esquema do Cenário: Cadastrar produtos
  produtos.feature:7
    Dado que posso gerenciar produtos
    itens/gerenciarprodutos_steps.rb:2
      Quando informo "<nome>" como nome do produto
      itens/gerenciarprodutos_steps.rb:6
        E informo R$ <valor> como valor do produto
        itens/gerenciarprodutos_steps.rb:10
        E informo <quantidade> como estoque do produto
        itens/gerenciarprodutos_steps.rb:14
        E cadastro este produto
        itens/gerenciarprodutos_steps.rb:18
  WARNING: U+00E3 from UTF-8 to IBM437
  Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defin
  itens/gerenciarprodutos_steps.rb:22

  Exemplos:
    ! nome ! valor ! quantidade !
    ! Boné Qualister ! 150,50 ! 50 !
  error) undefined method `nome=' for #<Produto:0x2cf58c0 @produtos=[]> <NoMethodEr
  ?)" como nome do produto/'
    features\gerenciarprodutos.feature:9:in 'Quando informo "<nome>" como nome
    do produto'
    ! Camiseta Qualister ! 99,00 ! 100 !
  error) undefined method `nome=' for #<Produto:0x2ca38d0 @produtos=[]> <NoMethodEr
  ?)" como nome do produto/'
    features\gerenciarprodutos.feature:9:in 'Quando informo "<nome>" como nome
    do produto'

  Failing Scenarios:
  cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos
  cucumber features\gerenciarprodutos.feature:7 # Scenario: Cadastrar produtos

  2 scenarios (2 failed)
  12 steps (2 failed, 8 skipped, 2 passed)
  0m0.028s
```



# Exercício: Refatorando os Passos

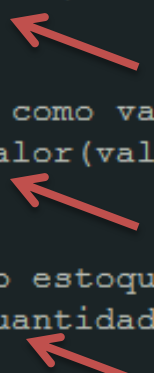
- Precisamos alterar a forma que informamos os valores antes de cadastrar o produto
- 1) Troque a forma que informamos o nome, valor e quantidade antes de serem cadastrados baseando-se na imagem ao lado

## Antes:

```
6 Quando /informo "(.*?)" como nome do produto/ do |nome|
7   @produto.nome = nome
8 end
9
10 E /informo R\$ (.*)? como valor do produto/ do |valor|
11   @produto.valor = valor
12 end
13
14 E /informo (\d+) como estoque do produto/ do |quantidade|
15   @produto.quantidade = quantidade.to_i
16 end
```

## Depois:

```
6 Quando /informo "(.*?)" como nome do produto/ do |nome|
7   @produto.informarNome(nome)
8 end
9
10 E /informo R\$ (.*)? como valor do produto/ do |valor|
11   @produto.informarValor(valor)
12 end
13
14 E /informo (\d+) como estoque do produto/ do |quantidade|
15   @produto.informarQuantidade(quantidade.to_i)
16 end
```



# Exercício: Executando a Funcionalidade após a refatoração dos Passos

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Agora, após a refatoração dos Passos, o resultado dos testes voltam a passar

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Esquema do Cenário: Cadastrar produtos # features\gerenciarp
  produtos.feature:7
    Dado que posso gerenciar produtos # features/step_defin
    itions/gerenciarpodutos_steps.rb:2
    Quando informo "<nome>" como nome do produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:6
    E informo R$ <valor> como valor do produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:10
    E informo <quantidade> como estoque do produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:14
    E cadastro este produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:18
  WARNING: U+00E3 from UTF-8 to IBM437
  Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defi
  nitions/gerenciarpodutos_steps.rb:22

  Exemplos:
    | nome | valor | quantidade |
    | 150 |
  Boné Qualister | 150,50 | 50 |
    | 100 |
  Camiseta Qualister | 99,00 | 100 |

  2 scenarios (2 passed)
  12 steps (12 passed)
  0m0.025s
```



A group of six business professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held up by their hands and features the text "Usando Grupos de Atributos" in a bold, black, sans-serif font. The background is a plain, light color.

**Usando Grupos de  
Atributos**

# Exercício: Utilizando Grupos de Atributos

- 1) Vamos alterar o controle de estoque do produto, adicionando um grid de itens de estoque ao invés de apenas a quantidade
- 2) Para isso, iremos fazer algumas alterações no arquivo feature:
- 3) substituir o Passo:  
E informo  
<quantidade>  
como estoque do produto  
Pela grupo de atributos, conforme imagem ao lado
- 4) Retirar a coluna quantidade da tabela de Exemplos

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Esquema do Cenário: Cadastrar produtos
8     Dado que posso gerenciar produtos
9     Quando informo "<nome>" como nome do produto
10      E informo R$ <valor> como valor do produto
11      E informo o estoque para o produto:
12      {
13        | cor      | material | tipo    | estoque |
14        | Verde   | Couro    | Social  | 100     |
15        | Azul    | Linho    | Casual  | 50      |
16        | Cinza   | Algodão  | Sport   | 75      |
17      }
18      E cadastro este produto
19      Então vejo a mensagem "Produto cadastrado com sucesso"
20
21 Exemplos:
22 {
23   | nome                | valor |
24   | Boné Qualister     | 150,50 |
25   | Camiseta Qualister  | 99,00  |
26 }
```

# Exercício: Utilizando Grupos de Atributos

- 1) Abra o prompt de comando
- 2) Digite `cd [diretório_do_projeto]` e pressione Enter
- 3) Digite `cucumber` e pressione Enter para executar as Funcionalidades novamente
- 4) O Cucumber mostra agora que temos um 2 Passos não definidos, na verdade é um Passo apenas, que foi executado duas vezes
- 5) Precisamos definir este Passo para que este erro pare de ocorrer
- 6) O Cucumber nos dá um exemplo de como implementar este passo na imagem ao lado, em amarelo

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Esquema do Cenário: Cadastrar produtos
  produtos.feature:7
    Dado que posso gerenciar produtos
    itions/gerenciarprodutos_steps.rb:2
    Quando informo "<nome>" como nome do produto
    itions/gerenciarprodutos_steps.rb:6
    E informo R$ <valor> como valor do produto
    itions/gerenciarprodutos_steps.rb:10
    E informo o estoque para o produto:
    produtos.feature:11
      cor | material | tipo | estoque |
      Verde | Couro | Social | 100 |
      Azul | Linho | Casual | 50 |
      Cinza | Algodão | Sport | 75 |
      E cadastro este produto
    itions/gerenciarprodutos_steps.rb:24
    Quando informo o estoque para o produto:
    Então vejo a mensagem "Produto cadastrado com sucesso"
    nitions/gerenciarprodutos_steps.rb:28

  Exemplos:
    nome | valor |
    Boné Qualister | 150,50 |
    Camiseta Qualister | 99,00 |

  2 scenarios (2 undefined)
  12 steps (4 skipped, 2 undefined, 6 passed)
  0m0.027s

  You can implement step definitions for undefined steps with these snippets:

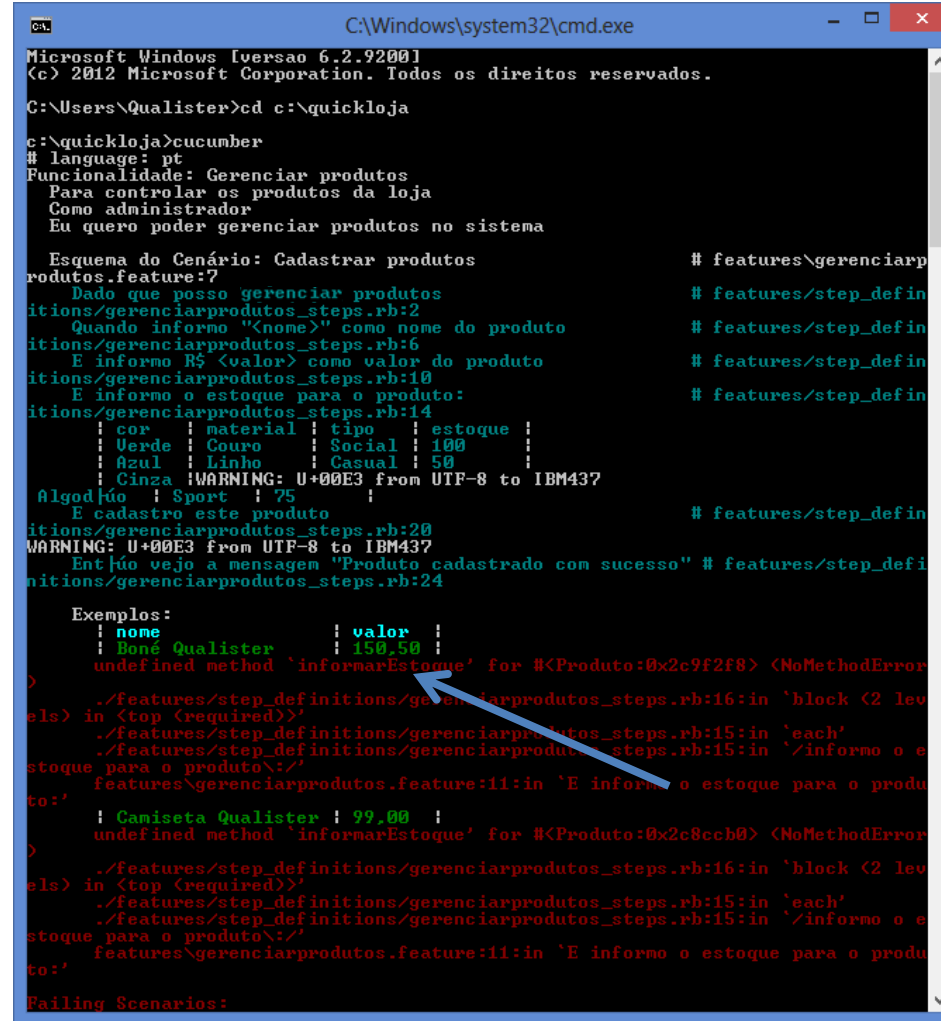
  Quando(/^informo o estoque para o produto:$/> do |table|
    # table is a Cucumber::Ast::Table
    pending # express the regexp above with the code you wish you had
  end
```

# Exercício: Utilizando Grupos de Atributos

<ol style="list-style-type: none"><li>1) Abra o arquivo <code>gerenciarprodutos_steps.rb</code> que está dentro do subdiretório <code>step_definitions</code></li><li>2) Remova o Passo que antes definia a ação de informar quantidade</li></ol>	<pre>14 E /informo (\d+) como estoque do produto/ do  quantidade  15   @produto.informarQuantidade(quantidade.to_i) 16 end</pre>
<ol style="list-style-type: none"><li>3) Crie o Passo que captura a tabela do Grupo de Atributos</li><li>4) Itere com todas as linhas da tabela, convertendo cada linha da tabela em Hash</li><li>5) Execute o método <b>informarEstoque</b> passando como parâmetro cada uma das <b>linhas da tabela</b></li><li>6) Execute novamente a funcionalidade</li></ol>	<pre>14 E /informo o estoque para o produto\:/ do  tabela  15   tabela.hash.each do  linha  16     @produto.informarEstoque(linha) 17   end 18 end</pre>

# Exercício: Utilizando Grupos de Atributos

- 1) Abra o prompt de comando
- 2) Digite `cd [diretório_do_projeto]` e pressione Enter
- 3) Digite `cucumber` e pressione Enter para executar as Funcionalidades novamente
- 4) Ao executar o Passo o Cucumber detectou que o método `informarEstoque` ainda não existe dentro da classe `Produto`



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.92001]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Qualister>cd c:\quickloja

c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
Para controlar os produtos da loja
Como administrador
Eu quero poder gerenciar produtos no sistema

Esquema do Cenário: Cadastrar produtos # features\gerenciarp
rodutos.feature:7
Dado que posso gerenciar produtos # features/step_defin
itions/gerenciarpodutos_steps.rb:2
Quando informo "<nome>" como nome do produto # features/step_defin
itions/gerenciarpodutos_steps.rb:6
E informo R$ <valor> como valor do produto # features/step_defin
itions/gerenciarpodutos_steps.rb:10
E informo o estoque para o produto: # features/step_defin
itions/gerenciarpodutos_steps.rb:14
| cor | material | tipo | estoque |
| Verde | Couro | Social | 100 |
| Azul | Linho | Casual | 50 |
| Cinza | Algodão | Sport | 75 |
E cadastro este produto # features/step_defin
itions/gerenciarpodutos_steps.rb:20
WARNING: U+00E3 from UTF-8 to IBM437
Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defi
nitions/gerenciarpodutos_steps.rb:24

Exemplos:
| nome | valor |
| Boné Qualister | 150.50 |
undefined method `informarEstoque' for #<Produto:0xc9f2f8> (NoMethodError)
>
./features/step_definitions/gerenciarpodutos_steps.rb:16:in `block (2 lev
els) in <top (required)>'
./features/step_definitions/gerenciarpodutos_steps.rb:15:in `each'
./features/step_definitions/gerenciarpodutos_steps.rb:15:in `/'informo o e
stoque para o produto\:'
features\gerenciarpodutos.feature:11:in `E informo o estoque para o produ
to:'
| Camiseta Qualister | 99.00 |
undefined method `informarEstoque' for #<Produto:0xc8ccb0> (NoMethodError)
>
./features/step_definitions/gerenciarpodutos_steps.rb:16:in `block (2 lev
els) in <top (required)>'
./features/step_definitions/gerenciarpodutos_steps.rb:15:in `each'
./features/step_definitions/gerenciarpodutos_steps.rb:15:in `/'informo o e
stoque para o produto\:'
features\gerenciarpodutos.feature:11:in `E informo o estoque para o produ
to:'

Failing Scenarios:
```

# Exercício: Utilizando Grupos de Atributos

1) Abra o arquivo produto.rb que está dentro do subdiretório lib

2) Remova o método informarQuantidade

```
28 def informarQuantidade(quantidade)
29   raise "Informe uma Quantidade maior que 0" if quantidade <= 0
30   @quantidade = quantidade
31 end
```

3) Declare o vetor @estoque dentro do método initialize

```
3 def initialize
4   @produtos = Array.new
5   @estoque = Array.new ←
6 end
```

3) Agora crie o método informarEstoque que recebe o parâmetro item, um Hash

4) Esse método lança uma exceção caso uma das colunas esteja vazia

5) Caso todas as colunas tenham sido preenchidas estas são armazenadas no vetor @estoque

```
28 def informarEstoque(item)
29   if item["cor"].empty? || item["material"].empty? || item["tipo"].empty? || item["estoque"].empty?
30     raise "Informe todos os dados"
31   end
32   @estoque << item
33 end
```

# Exercício: Utilizando Grupos de Atributos

- 1) Abra o prompt de comando
- 2) Digite cd [diretório\_do\_projeto] e pressione Enter
- 3) Digite cucumber e pressione Enter para executar as Funcionalidades novamente
- 4) Desta vez vemos que todos os Passos foram executados com sucesso

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Esquema do Cenário: Cadastrar produtos # features\gerenciarp
produtos.feature:7
    Dado que posso gerenciar produtos # features/step_defin
itions/gerenciarprodutos_steps.rb:2
    Quando informo "<nome>" como nome do produto # features/step_defin
itions/gerenciarprodutos_steps.rb:6
    E informo R$ <valor> como valor do produto # features/step_defin
itions/gerenciarprodutos_steps.rb:10
    E informo o estoque para o produto: # features/step_defin
itions/gerenciarprodutos_steps.rb:14
      cor | material | tipo | estoque |
      Verde | Couro | Social | 100 |
      Azul | Linho | Casual | 50 |
      Cinza |WARNING: U+00E3 from UTF-8 to IBM437
      Algodão | Sport | 75 |
    E cadastro este produto # features/step_defin
itions/gerenciarprodutos_steps.rb:20
WARNING: U+00E3 from UTF-8 to IBM437
    Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defi
nitions/gerenciarprodutos_steps.rb:24

  Exemplos:
    nome | valor |
    Boné Qualister | 150,50 |
    Camiseta Qualister | 99,00 |

2 scenarios (2 passed)
12 steps (12 passed)
0m0.041s
```

# Exercício: Criando outros Cenários

- 1) Abra o arquivo `gerenciarprodutos.feature` e descreva um novo Cenário: "Excluir produtos"
- 2) Siga o modelo contido na imagem ao lado
- 3) Execute esta Funcionalidade

Dica:

- Apenas o **Esquema do Cenário** utilizará os dados da tabela de Exemplos. Por esta causa o Cenário que acabou de ser declarado será executado apenas uma vez

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Esquema do Cenário: Cadastrar produtos
8     Dado que posso gerenciar produtos
9     Quando informo "<nome>" como nome do produto
10      E informo R$ <valor> como valor do produto
11      E informo o estoque para o produto:
12        | cor | material | tipo | estoque |
13        | Verde | Couro | Social | 100 |
14        | Azul | Linho | Casual | 50 |
15        | Cinza | Algodão | Sport | 75 |
16      E cadastro este produto
17      Então vejo a mensagem "Produto cadastrado com sucesso"
18
19   Exemplos:
20     | nome | valor |
21     | Boné Qualister | 150,50 |
22     | Camiseta Qualister | 99,00 |
23
24   Cenário: Excluir produtos
25     Dado que posso gerenciar produtos
26     Quando excluo o produto "Boné Qualister"
27     Então a mensagem "Produto excluído" é apresentada
```



# Exercício: Criando outros Cenários

- 1) Abra o prompt de comando
- 2) Digite `cd [diretório_do_projeto]` e pressione Enter
- 3) Digite `cucumber` e pressione Enter para executar as Funcionalidades novamente
- 4) O Cucumber mostra que o primeiro Passo do Cenário que criamos passou, isso ocorre porque já tínhamos definido este Passo anteriormente.
- 5) O Cucumber também acusa que 2 Passos ainda não foram definidos. Vamos definí-los no próximo slide

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

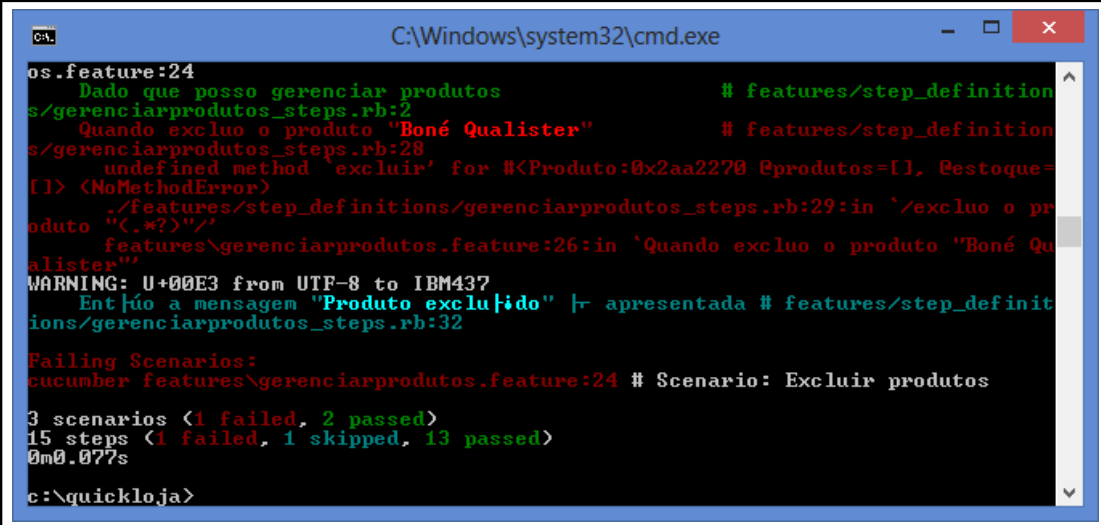
  Esquema do Cenário: Cadastrar produtos # features\gerenciarp
  produtos.feature:7
    Dado que posso gerenciar produtos # features/step_defin
    itions/gerenciarpodutos_steps.rb:2
      Quando informo "<nome>" como nome do produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:6
      E informo R$ <valor> como valor do produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:10
      E informo o estoque para o produto: # features/step_defin
    itions/gerenciarpodutos_steps.rb:14
      | cor | material | tipo | estoque |
      | Verde | Couro | Social | 100 |
      | Azul | Linho | Casual | 50 |
      | Cinza | WARNING: U+00E3 from UTF-8 to IBM437
      | Algodão | Sport | 75 |
      E cadastro este produto # features/step_defin
    itions/gerenciarpodutos_steps.rb:20
    WARNING: U+00E3 from UTF-8 to IBM437
      Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defi
    nitions/gerenciarpodutos_steps.rb:24

  Exemplos:
      | nome | valor |
      | Boné Qualister | 150,50 |
      | Camiseta Qualister | 99,00 |

  Cenário: Excluir produtos # features\gerenciarpodut
  os.feature:24
    Dado que posso gerenciar produtos # features/step_definition
    s/gerenciarpodutos_steps.rb:2
      Quando excluo o produto "Boné Qualister" # features\gerenciarpodut
    os.feature:26
    WARNING: U+00E3 from UTF-8 to IBM437
      Então a mensagem "Produto excluído" é apresentada # features\gerenciarp
    odutos.feature:27

  3 scenarios (1 undefined, 2 passed)
  15 steps (2 undefined, 13 passed)
  0m0.033s
```

# Exercício: Criando outros Cenários

<ol style="list-style-type: none"><li>1) Abra o arquivo <code>gerenciarproduto.feature</code> e descreva os Passos contidos na imagem ao lado</li></ol>	<pre>28 Quando /excluo o produto "(.*)"/ do  produto  29   @resultado = @produto.excluir(produto) 30 end 31 32 Então /a mensagem "(.*)" é apresentada/ do  mensagem  33   expect(@resultado).to eq(mensagem) 34 end</pre>
<ol style="list-style-type: none"><li>1) Abra o prompt de comando</li><li>2) Digite <code>cd [diretório_do_projeto]</code> e pressione Enter</li><li>3) Digite <code>cucumber</code> e pressione Enter para executar as Funcionalidades novamente</li><li>4) O Cucumber mostra que o método utilizado na linha 29 ainda não foi criado na classe <code>Produto</code>, por isso vamos criá-lo</li></ol>	 <pre>C:\Windows\system32\cmd.exe os.feature:24   Dado que posso gerenciar produtos # features/step_definition s/gerenciarprodutos_steps.rb:2   Quando excluo o produto "Boné Qualister" # features/step_definition s/gerenciarprodutos_steps.rb:28     undefined method `excluir' for #&lt;Produto:0x2aa2270 @produtos=[], @estoque= [1]&gt; (NoMethodError)     ./features/step_definitions/gerenciarprodutos_steps.rb:29:in `/excluo o pr oduto "(.*)"/     features\gerenciarprodutos.feature:26:in `Quando excluo o produto "Boné Qu alister"' WARNING: U+00E3 from UTF-8 to IBM437   Então a mensagem "Produto excluído" é apresentada # features/step_definit ions/gerenciarprodutos_steps.rb:32  Failing Scenarios: cucumber features\gerenciarprodutos.feature:24 # Scenario: Excluir produtos  3 scenarios (1 failed, 2 passed) 15 steps (1 failed, 1 skipped, 13 passed) 0m0.077s  c:\quickloja&gt;</pre>
<ol style="list-style-type: none"><li>1) Abra o arquivo <code>produto.rb</code> contido no diretório "lib"</li><li>2) Adicione o método contido na imagem ao lado</li></ol>	<pre>35 def excluir(produto) 36   if produto.empty? 37     raise "Informe o Produto que quer excluir" 38   else 39     return "Produto excluído" 40   end 41 end</pre>

# Exercício: Criando outros Cenários

- 1) Abra o prompt de comando
- 2) Digite `cd [diretório_do_projeto]` e pressione Enter
- 3) Digite `cucumber` e pressione Enter para executar as Funcionalidades novamente
- 4) O Cucumber mostra que 3 cenários passaram:
  - a. 2 execuções do Esquema do Cenário, utilizando as linhas da tabela de Exemplos
  - b. 1 execução do Cenário “Excluir produto”
- 5) Vemos que o Passo “Dado que posso gerenciar produtos” é utilizado por todos os cenários, como uma pré-condição de execução.
- 6) No próximo slide veremos como estruturar melhor pré-condições no Cucumber

```
C:\Windows\system32\cmd.exe
Microsoft Windows [versao 6.2.9200]
(c) 2012 Microsoft Corporation. Todos os direitos reservados.
C:\Users\Qualister>cd c:\quickloja
c:\quickloja>cucumber
# language: pt
Funcionalidade: Gerenciar produtos
  Para controlar os produtos da loja
  Como administrador
  Eu quero poder gerenciar produtos no sistema

  Esquema do Cenário: Cadastrar produtos                                # features\gerenciarp
produtos.feature:7
    Dado que posso gerenciar produtos                                  # features/step_defin
itions/gerenciarpdutos_steps.rb:2
      Quando informo "<nome>" como nome do produto                  # features/step_defin
itions/gerenciarpdutos_steps.rb:6
        E informo R$ <valor> como valor do produto                  # features/step_defin
itions/gerenciarpdutos_steps.rb:10
        E informo o estoque para o produto:                          # features/step_defin
itions/gerenciarpdutos_steps.rb:14
          | cor | material | tipo | estoque |
          | Verde | Couro | Social | 100 |
          | Azul | Linho | Casual | 50 |
          | Cinza | WARNING: U+00E3 from UTF-8 to IBM437
          | Algodão | Sport | 75 |
        E cadastro este produto                                        # features/step_defin
itions/gerenciarpdutos_steps.rb:20
        WARNING: U+00E3 from UTF-8 to IBM437
        Então vejo a mensagem "Produto cadastrado com sucesso" # features/step_defi
nitions/gerenciarpdutos_steps.rb:24

  Exemplos:
    | nome | valor |
    | Boné Qualister | 150,50 |
    | Camiseta Qualister | 99,00 |

  Cenário: Excluir produtos                                            # features\gerenciarpdutos
os.feature:24
    Dado que posso gerenciar produtos                                  # features/step_defin
s/gerenciarpdutos_steps.rb:2
      Quando excluo o produto "Boné Qualister"                        # features/step_defin
s/gerenciarpdutos_steps.rb:28
      WARNING: U+00E3 from UTF-8 to IBM437
      Então a mensagem "Produto excluído" é apresentada # features/step_definit
ions/gerenciarpdutos_steps.rb:32

3 scenarios (3 passed)
15 steps (15 passed)
0m0.050s
```

A diverse group of six business professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held up by their hands and features the text "Usando Contextos" in a bold, black, sans-serif font. The background is a plain, light color.

**Usando Contextos**

# Exercício: Utilizando Contextos

- 1) Vamos descrever um novo Cenário, para excluir produtos
- 2) Vemos que o primeiro Passo deste Cenário é uma pré-condição em todos os cenários
- 3) Em casos como este, onde temos pré-condições para todos os cenários, podemos usar um recurso chamado **Contexto**, que executa uma seqüência de Passos antes de executar cada um dos Cenários
- 4) Vemos no próximo slide como descrever este pré-requisito na Funcionalidade

## Dica:

- Passos que já foram definidos não necessitam ser definidos novamente

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Esquema do Cenário: Cadastrar produtos
8     Dado que posso gerenciar produtos ←
9     Quando informo "<nome>" como nome do produto
10      E informo R$ <valor> como valor do produto
11      E informo o estoque para o produto:
12        | cor    | material | tipo    | estoque |
13        | Verde | Couro    | Social  | 100      |
14        | Azul  | Linho    | Casual  | 50       |
15        | Cinza  | Algodão  | Sport   | 75       |
16      E cadastro este produto
17      Então vejo a mensagem "Produto cadastrado com sucesso"
18
19   Exemplos:
20     | nome                | valor |
21     | Boné Qualister     | 150,50 |
22     | Camiseta Qualister | 99,00  |
23
24   Cenário: Excluir produtos
25     Dado que posso gerenciar produtos ←
26     Quando excluo o produto "Boné Qualister"
27     Então a mensagem "Produto excluído" é apresentada
```

# Exercício: Utilizando Contextos

- 1) Retire o passo “Dado que posso gerenciar produtos” de todos os cenários
- 2) Descreva o contexto “Possibilidade de gerenciar produtos”
- 3) Descreva o Passo “Dado que posso gerenciar produtos” dentro do Contexto

## Dica:

- Contexto é um ótimo recurso quando se tem pré-condições em Funcionalidades
- Contexto podem ter diversos passos
- As variáveis globais (declaradas com @) contidas nos Passos do Contexto podem ser utilizadas dentro dos Passos dos demais Cenários

```
1 # language: pt
2 Funcionalidade: Gerenciar produtos
3   Para controlar os produtos da loja
4   Como administrador
5   Eu quero poder gerenciar produtos no sistema
6
7   Contexto: Possibilidade de gerenciar produtos
8     Dado que posso gerenciar produtos
9
10  Esquema do Cenário: Cadastrar produtos
11    Quando informo "<nome>" como nome do produto
12    E informo R$ <valor> como valor do produto
13    E informo o estoque para o produto:
14      | cor      | material | tipo    | estoque |
15      | Verde   | Couro    | Social  | 100     |
16      | Azul    | Linho    | Casual  | 50      |
17      | Cinza   | Algodão  | Sport   | 75      |
18    E cadastro este produto
19    Então vejo a mensagem "Produto cadastrado com sucesso"
20
21  Exemplos:
22    | nome          | valor |
23    | Boné Qualister | 150,50 |
24    | Camiseta Qualister | 99,00 |
25
26  Cenário: Excluir produtos
27    Quando excluo o produto "Boné Qualister"
28    Então a mensagem "Produto excluído" é apresentada
```





# Exercício: Usando Hooks

1) Crie o arquivo “hooks.rb” dentro do subdiretório “support”	<pre>1 # encoding: utf-8 2 3 Before do  cenario  4   puts "Antes de iniciar o primeiro passo de cada cenário" 5 end 6 7 After do  cenario  8   puts "Depois de finalizar o último passo de cada cenário" 9 10 end</pre>
2) Coloque o comando contido na imagem ao lado	
3) Atribuímos o objeto Cenário à variável “cenario” para que possamos utilizar métodos e atributos deste objeto	
4) Recuperando o nome do Cenário ▪ <b>Obs. Não funciona no escopo After</b>	<pre>3 Before do  cenario  4   puts "Executando o cenário: #{cenario.name}" 5 end</pre>
5) Verificando se o Cenário falhou ▪ Também é possível saber se ele passou usando o método “.passed?”	<pre>7 After do  cenario  8   # retorna "true" se o cenário falhar ou "false" se passar 9   puts cenario.failed? 10 end</pre>
6) Agora vamos verificar se o Cenário falhou, se sim, iremos capturar o motivo do erro e apresentá-lo em uma mensagem personalizada	<pre>7 After do  cenario  8   if cenario.failed? 9     erro = "Cenário falhou, motivo: #{cenario.exception.message}" 10    puts erro 11  end 12 end</pre>
7) Poderíamos ter uma função, por exemplo, que cadastra um bug quando cenários falham, digamos “addBug()”	<pre>10 # puts erro 11 addBug(erro) ←</pre>



## Exercício: Usando Hooks

<p>8) Adicione o comando ao lado para que seja apresentada esta mensagem sempre depois que um Passo seja executado</p> <ul style="list-style-type: none"><li>▪ <b>Não funciona em Contextos</b></li></ul>	<pre>14 AfterStep do  cenario  15   puts "Ocorre após cada Passo ser executado" 16 end</pre>
<p>9. Adicione o comando ao lado para que seja apresentada esta mensagem depois que os arquivos de configuração (support) sejam carregados, e antes que as Funcionalidades sejam carregadas</p>	<pre>18 AfterConfiguration do  config  19   puts "Arquivos de 'support' foram carregados, as Funcionalidades não!" 20 end --</pre>
<p>10. Adicione o comando abaixo para manipular, simultaneamente, a execução do Cenário</p> <ul style="list-style-type: none"><li>▪ Na linha 24 estamos disparando um comando que aguarda 1 segundo, se a execução do Cenário atingir este tempo o comando para.call fará com que a execução seja cancelada</li></ul>	<pre>22 Around do  cenario, parar  23   puts "Ocorre durante a execução de todo o cenário" 24   Timeout.timeout(1.0) do 25     parar.call 26   end 27 end</pre>

A group of six business professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held by all of them, with their hands visible at the top edge. The background is a plain, light color.

**Relatórios de execução**

# Exercício: Relatórios de execução

- Cucumber possui diversos formatos de relatórios de execução, segue ao lado os formatos disponíveis
- Passos para obter o relatório da execução basta adicionar o comando `--format [Tipo_do_Relatório]`  
**Exemplo:**  
`cucumber --format pretty`
- Alguns formatos de relatório podem gerar Arquivos Externos. Para estes é necessário indicar em que arquivo iremos adicionar os resultados, indicamos o arquivo externo adicionando `--out [Nome_do_Arquivo_Externo]`  
**Exemplo:**  
`cucumber --format html --out Relatório.html`

Tipo	Descrição	Arquivo Externo?
<b>pretty (padrão)</b>	Gera os resultados da execução no próprio prompt de comando utilizando cores	Sim
<b>debug</b>	Gera as chamadas feitas aos ouvintes (Para formatadores de desenvolvimento)	Não
<b>html</b>	Gera o relatório de execução em formato HTML	Sim
<b>json</b>	Gera o relatório em formato json	Sim
<b>junit</b>	Gera o relatório no formato XML com estrutura semelhante ao gerado por Junit e Ant	Sim
<b>progress</b>	Apresenta uma linha pontilhada, onde cada ponto equivale a um Cenário executado	Sim
<b>rerun</b>	Apresenta o número das linhas dos arquivos que falharam durante a execução	Sim
<b>stepdefs</b>	Mostra o progresso, mostra o tempo levado para executar cada Passo, a definição do Passo e a linha em que ele se encontra	Sim
<b>usage</b>	Mostra o progresso, mostra o tempo levado para executar cada Passo, a definição do Passo e a linha em que ele se encontra. Se alguma mensagem for impressa na tela, também pe apresentada neste relatório. Os passos que tomaram mais tempo para serem executados são apresentados primeiro. Se usar o comando <code>--dry-run</code> na frente deste formato ele não contabilizará o tempo de performance de cada Passo e organizara os passos pelo nome do arquivo	Sim

A group of six business professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held by all six individuals. The background is a plain, light-colored wall.

**Cucumber e Selenium**

- Selenium é um framework para usado para automação de testes funcionais no Front-end
- Está disponível em diversas linguagens de programação como: Java, Ruby, Python, C#, Perl, Javascript
- Testa aplicações web através de drivers, que são simuladores ou controladores de navegador
- Os scripts consistem em identificar elementos e interagir com eles, através de ações como: Clicar, Digitar, Arrastar, etc

# Identificando elementos

## QuickLoja

Login

Senha

Entrar

Login

`input#usuariologin 220px × 30px`

Senha

Senha

`input#usuariosenha 220px × 30px`

Entrar

`button.btn.btn-medium.btn-primary 64px × 30px`

# Comandos básicos

```
1 # encoding: utf-8
2
3 # Instanciando o navegador Firefox
4 @navegador = Selenium::WebDriver.for :firefox
5
6 # Acessando a URL da aplicação
7 @navegador.get "http://quickloja.planned.by/"
8
9 # Instanciando o campo "Login" na variável campologin
10 campologin = @navegador.find_element(:id, "usuariologin")
11 # Digitando a palavra "teste" no campo "Login"
12 campologin.send_keys("teste")
13
14 # Instanciando o campo "Senha" na variável camposenha
15 camposenha = @navegador.find_element(:id, "usuariosenha")
16 # Digitando o número "123" no campo "Senha"
17 camposenha.send_keys("123")
18
19 # Instanciando o botão "Enviar" na variável botao
20 botao = @navegador.find_element(:css, "button.btn.btn-medium.btn-primary")
21 # Clicando no botão "Enviar"
22 botao.click
23
24 # Espera-se que o título do navegador seja "Administração QuickLoja"
25 expect(@navegador.title).to eq("Administração QuickLoja")
26
27 # Fechando o navegador
28 @navegador.quit
```

## Possibilitando a utilização do Selenium no Cucumber

- Adicione a importação ao Gem “selenium-webdriver” no arquivo env.rb contido no diretório support, conforme a linha 9 na imagem abaixo:

```
1 # encoding: utf-8
2 require 'cucumber/formatter/unicode'
3
4 $:.unshift('c:/quickloja/lib')
5 require 'produto'
6
7 require 'rspec/expectations'
8
9 require 'selenium-webdriver' ←
```



# Exercício: Escrevendo a Funcionalidade para Gerenciar Categorias

- 1) Clique com o botão direito do mouse sobre o diretório "features"
- 2) Selecione a opção "New file"
- 3) Coloque o nome "gerenciarcategorias.feature"
- 4) Abra o arquivo que foi criado, utilizando duplo clique
- 5) Coloque na primeira linha o código # language: pt simbolizando que iremos escrever em português
- 6) Escreva a Funcionalidade segundo a imagem ao lado
- 7) Escreveremos o cenário de Login como um **Contexto**, pois **sempre teremos que fazer login antes de cada Cenário ser executado**

```
1 # language: pt
2 Funcionalidade: Gerenciar categorias
3   Para conseguir categorizar os produtos
4   Como administrador
5   Eu quero poder gerenciar categorias
6
7 Contexto: Fazer login na aplicação
8   Dado que estou na tela de login
9   Quando digito meu login "teste"
10    E digito minha senha "123"
11    E submeto o formulário
12    Então eu vejo o título "Administração QuickLoja" no navegador
13
```

# Exercício: Definindo Passos do Contexto “Fazer login na aplicação”

- 1) Clique com o botão direito do mouse sobre o diretório “step\_definitions”
- 2) Selecione a opção “New file”
- 3) Coloque o nome “gerenciarcategorias\_steps.rb”
- 4) Abra o arquivo que foi criado, utilizando duplo clique
- 5) Coloque na primeira linha o código # encoding: utf-8 teremos caracteres latinos
- 6) Neste passo faremos com que o Selenium acesse a tela de login, para isso teremos que instanciar uma variável com um navegador e acessar a URL da página de login
- 7) Escreva o Passo segundo a ponta a imagem ao lado

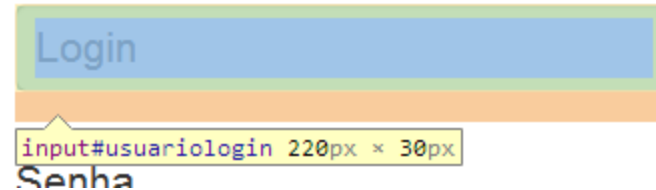
```
1 # encoding: utf-8
2
3 Dado /que estou na tela de login/ do
4   @navegador = Selenium::WebDriver.for :firefox
5   @navegador.get "http://quickloja.planned.by/"
6 end
```

- Na linha 4 estamos instanciando na variável @navegador o navegador Firefox
- Na linha 5 estamos utilizando o método “get” para acessar a URL da aplicação QuickLoja

# Exercício: Definindo Passos do Contexto “Fazer login na aplicação”

- 1) Vamos adicionar o Passo onde o usuário informa o seu login no campo “login”
- 2) Escreva o Passo segundo a ponta a imagem ao lado

Login



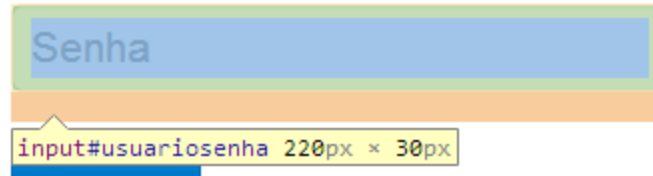
```
8 Quando /digito meu login "(.*?)" / do |login|
9   campologin = @navegador.find_element(:id, "usuariologin")
10  campologin.send_keys(login)
11 end
```

- Na linha 8 estamos descrevendo o passo e extraíndo o login com a expressão regular `(.*?)` e atribuindo-o à variável `login`
- Na linha 9 estamos instanciando a variável `campologin` com o campo Login do formulário, este campo foi identificado pelo ID `usuariologin`
- Na linha 10 estamos utilizando o método `send_keys` para digitar o texto contido na variável `login`

# Exercício: Definindo Passos do Contexto “Fazer login na aplicação”

- 1) Vamos adicionar o Passo onde o usuário informa o sua senha no campo “senha”
- 2) Escreva o Passo segundo a ponta a imagem ao lado

Senha



```
13 E /digito minha senha "(.*?)" / do |senha|
14   camposenha = @navegador.find_element(:id, "usuariosenha")
15   camposenha.send_keys(senha)
16 end
```

- Na linha 13 estamos descrevendo o passo e extraindo a senha com a expressão regular `(.*?)` e atribuindo-a à variável “senha”
- Na linha 14 estamos instanciando a variável “camposenha” com o campo Senha do formulário, este campo foi identificado pelo ID “usuariosenha”
- Na linha 15 estamos utilizando o método “send\_keys” para digitar o texto contido na variável “senha”

# Exercício: Definindo Passos do Contexto “Fazer login na aplicação”

- 1) Vamos adicionar o Passo onde o usuário submete o formulário para conseguir acesso ao painel administrativo
- 2) Escreva o Passo segundo a ponta a imagem ao lado



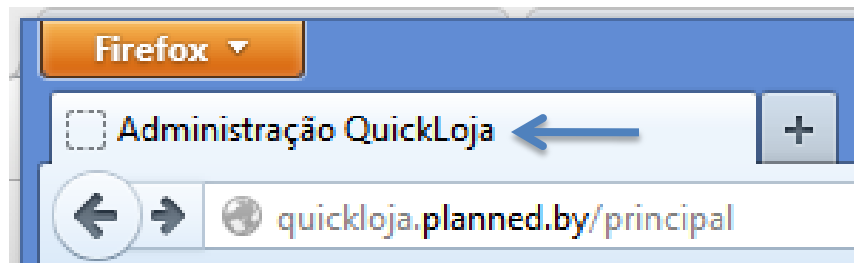
`button.btn.btn-medium.btn-primary 64px × 30px`

```
18 E /submeto o formulário/ do
19   botao = @navegador.find_element(:css, "button.btn.btn-medium.btn-primary")
20   botao.click
21 end
```

- Na linha 18 estamos descrevendo o passo em que o usuário submete o formulário
- Na linha 19 estamos instanciando a variável “botao”, este elemento foi identificado pelo CSS contido no atributo class deste elemento
- Na linha 20 estamos utilizando o método “click” para clicar no botão “Enviar”, para que o formulário seja submetido

# Exercício: Definindo Passos do Contexto “Fazer login na aplicação”

- 1) Vamos adicionar o Passo onde o usuário submete o formulário para conseguir acesso ao painel administrativo
- 2) Escreva o Passo segundo a ponta a imagem ao lado



```
29 Então /eu vejo o título "(.*?)" no navegador/ do |titulo|
30   expect(@navegador.title).to eq(titulo)
31   @navegador.quit
32 end
```

- Na linha 29 estamos descrevendo o passo e extraindo um texto que diz respeito ao título do navegador quando o usuário faz login, faremos isso utilizando expressão regular (.\*?), o valor extraído está sendo atribuído-o à variável “titulo”
- Na linha 30 estamos utilizando o método “expect to eq”, para verificar se o conteúdo da variável “titulo”, que foi extraído pela expressão regular, é igual ao que está sendo exibido no título do navegador
- Na linha 31 estamos utilizando o método “quit” para fechar o navegador

A group of six diverse business professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held up by their hands and features the text "Variáveis de ambiente" in a bold, black, sans-serif font. The background is a plain, light color.

**Variáveis de ambiente**

# Exercício: Variáveis de ambiente

- 1) Adicione o comando `NAVEGADOR=CHROME` após a chamada ao Cucumber no prompt de comando

```
C:\Users\Qualister>cd c:\quickloja  
c:\quickloja>cucumber NAVEGADOR=CHROME
```

Variável de Ambiente

Valor

- 2) Abra o arquivo `gerenciarcategorias_steps.rb` e adicione o comando condicional “if/elsif/else” para verificar qual é o navegador que está na variável e então, qual navegador será usado nos testes

- Para utilizar os navegadores “:chrome” e “:ie” é necessário instalar seus drivers

```
4  if ENV['NAVEGADOR'] == "CHROME"  
5    @navegador = Selenium::WebDriver.for :chrome  
6  elsif ENV['NAVEGADOR'] == "IE"  
7    @navegador = Selenium::WebDriver.for :ie  
8  else  
9    @navegador = Selenium::WebDriver.for :firefox  
10 end
```

- Usamos o vetor `ENV[]` para manipular as variáveis de ambiente (**ENV**ironment variables)
- Na linha 4 verificamos se a variável `NAVEGADOR` possui o valor `CHROME`, se sim, utilizamos o navegador Chrome nos testes
- Na linha 6 verificamos se a variável `NAVEGADOR` possui o valor `IE`, se sim, utilizamos o navegador Internet Explorer nos testes
- Na linha 8 declaramos um `ELSE` para descrever que, se nenhum navegador for informado, utilizamos o navegador Firefox nos testes



A diverse group of six business professionals (three men and three women) are standing behind a large white rectangular sign. They are all smiling and looking towards the camera. The sign is held up by their hands and features the text "Organizando seus Testes" in a bold, black, sans-serif font. The background is a plain, light color.

**Organizando seus Testes**

# Exercício: Organizando seus testes

- 1) Tags são utilizadas para organizar os testes, agrupando-os por assuntos, para que depois possamos executar testes baseados nas tags
- 2) Podemos utilizar Tags em Funcionalidades, Cenários, Esquemas de Cenários e Exemplos
- 3) Podemos utilizar quantas tags acharmos necessárias, basta separá-las por espaços
- 4) Quando executamos uma tag de uma Funcionalidade, todas as tags contidas dentro desta Funcionalidade são executadas
- 5) Quando executamos uma tag de um Cenário, Esquema de Cenário ou Exemplo apenas esta tag e a Funcionalidade, e Contexto da Funcionalidade, onde ela está são executadas

```
1 # language: pt
2 @unidade @fazendo ←
3 Funcionalidade: Gerenciar produtos
4   Para controlar os produtos da loja
5   Como administrador
6   Eu quero poder gerenciar produtos no sistema
7
8 Contexto: Posso gerenciar produtos
9   Dado que posso gerenciar produtos
10
11 @lentos @exemplos ←
12 Esquema do Cenário: Cadastrar produtos
13   Quando informo "<nome>" como nome do produto
14   E informo R$ <valor> como valor do produto
15   E informo o estoque para o produto:
16     | cor   | material | tipo   | estoque |
17     | Verde | Couro    | Social | 100      |
18     | Azul  | Linho    | Casual | 50        |
19     | Cinza | Algodão  | Sport  | 75        |
20   E cadastro este produto
21   Então vejo a mensagem "Produto cadastrado com sucesso"
22
23 Exemplos:
24   | nome          | valor |
25   | Boné Qualister | 150,50 |
26   | Camiseta Qualister | 99,00 |
27
28 @rapidos ←
29 Cenário: Excluir produtos
30   Quando excluo o produto "Boné Qualister"
31   Então a mensagem "Produto excluído" é apresentada
```

## Exercício: Organizando seus testes

<ul style="list-style-type: none"><li>▪ Executando todos os testes que foram marcados com a tag @web</li></ul>	<code>cucumber --tags @unidade</code>
<ul style="list-style-type: none"><li>▪ Executando todos os testes que <b>NÃO</b> foram marcados com a tag @unidade</li></ul>	<code>cucumber --tags ~@unidade</code>
<ul style="list-style-type: none"><li>▪ Executando testes que estão marcados com as tags @unidade “E” @web</li></ul>	<code>cucumber --tags @unidade --tags @web</code>
<ul style="list-style-type: none"><li>▪ Executando testes que estão marcados com as tags @unidade “OU” @web</li></ul>	<code>cucumber --tags @unidade,@web</code>
<ul style="list-style-type: none"><li>▪ Lança um erro caso haja mais do que 3 testes marcados com a tag @fazendo, se não, executa os testes marcados com a tag</li></ul>	<code>cucumber --tags @fazendo:3</code>

## Estudo de Caso “QuickLoja”



## Estudo de Caso “QuickLoja”

- Apresentação do estudo de caso no QuickLoja

## Contato:

- **Email:** [julio.lima@qualister.com.br](mailto:julio.lima@qualister.com.br)
- **Telefone:** (48) 3285 5615 / 9645 5506
- **Endereço:** Rua Patrício Antônio Teixeira, 317, Sala 406-A, Jardim Carandaí. Biguaçu/SC. CEP 88160-000

