07

Resources

Capítulo 9 - *Resourses*

Excluindo produtos

Já implementamos nosso site para poder incluir produtos na tabela. Agora iremos descobrir como fazer para excluí-los. Primeiramente criamos os links que, quando clicados, excluirão os itens:

```
<tr>
    <td><%= produto.nome %></td>
    <td><%= produto.descricao %></td>
    <td><%= produto.preco %></td>
    <td><%= produto.quantidade %></td>
    <td><a href="/produtos/<%= produto.id %>/remove">Remover</a></td>
</tr>
```

	Preço	Quantidade	
	120.3	13	Remover
	37.66	8	Remover
	70.5	10	Remover
	100.0	300	Remover
	3.0	10	Remover

	Quantidade
	10

Precisa haver alguém que atenda essa URL, então vamos ao *routes*:

```
get "/produtos/:id/remove" => "produtos#destroy"
```

Precisamos criar no *controller* a *regra de negócio* que "destrói" o produto. Esta irá redirecionar para a página das tabelas quando clicamos no botão para remover. Façamos isso primeiro:

```
def destroy
    redirect_to root_url
end
```

É bom colocar o `redirect_to root_url` no método *create* também, pois queremos que todas as ações direcionem para nossa página raiz (sem esquecer de excluir "create.html.erb"). Falta implementar o código que exclui o item. Como a propriedade

do produto que pegamos como parâmetro foi o *id*, fazemos:

```
def destroy
    id = params[:id]
    Produto.destroy id
    redirect_to root_url
end
```

Pronto, agora estamos excluindo os produtos!

Melhorando o código para excluir produtos

Para montar esse link de uma maneira mais fácil, com uma URL melhor e que não necessita decorar o *id* dos produtos, o *Rails* possui o o helper `link_to` que nos ajuda:

```
<tr>
     <td><%= produto.nome %></%></td>
     <td><%= produto.descricao %></%></td>
     <td><%= produto.preco %></%></td>
     <td><%= produto.quantidade %></%></td>
     <td>
     <%= link_to "Remover", "/produtos/#{produto.id}/remove" %>
 </tr>
```

Desse jeito a URL está chumbada no código. Se a usarmos em outros lugares pode ser um problema se em algum momento quisermos mudá-la. O próprio *Rails* pode criar uma URL para nós. Faremos isso através do método HTTP `delete` no *routes*. Mas antes, implementemos no HTML:

```
<%= link_to "Remover", produto, method: :delete %>
```

E no *routes*, no lugar de `get "/produtos/:id/remove" => "produtos#destroy"` , fazemos:

```
delete "/produtos/:id" => "produtos#destroy", as: :produto
```

Existe uma rota "delete" que, quando chamada na URL, por sua vez chamará o método `destroy` no *controller*. O `as: :produto` serve para encontrar o "produto_path".

Segurança e aviso na hora de remover um produto

O problema de termos links que removem na nossa página é que eles podem ser seguidos pelo Google. Então, em vez de link, usaremos um botão. Inclusive é muito mais semântico, uma vez que o link direciona para mais informações, enquanto que o botão executa uma ação. Do mesmo jeito que existe um `link_to` , existe um `button_to` :

```
<%= button_to "Remover", produto, method: :delete %>
```

Preço	Quantidade	
120.3	13	Remover
37.66	8	Remover
70.5	10	Remover
100.0	300	Remover
20.0		Remover

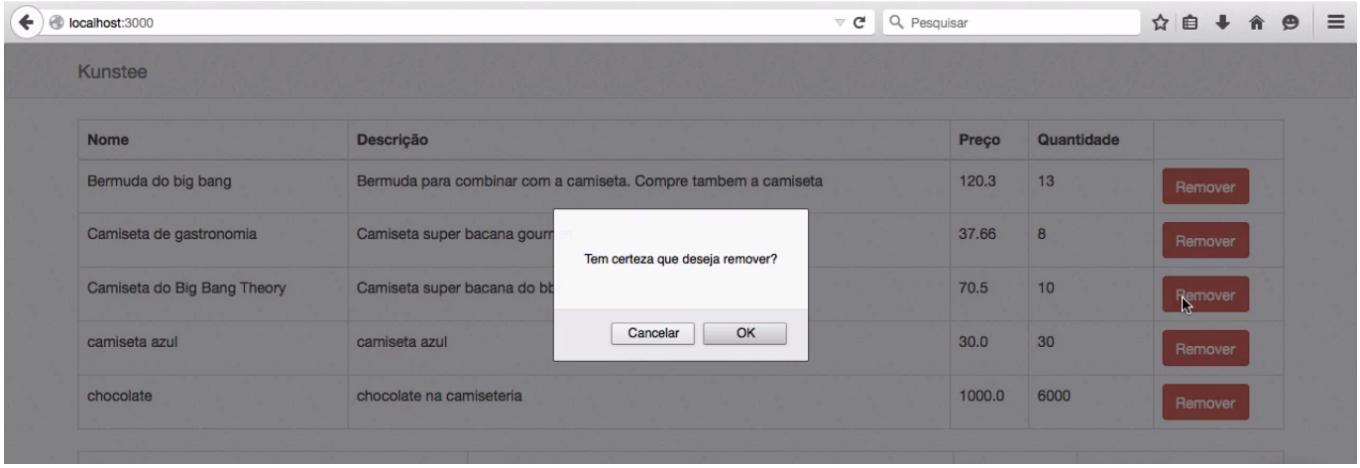
Vamos deixar o botão vermelho - esta é uma ação perigosa - usando o Bootstrap:

```
<%= button_to "Remover", produto, method: :delete, class: "btn btn-danger" %>
```

Preço	Quantidade	
120.3	13	Remover
37.66	8	Remover
70.5	10	Remover
20.0		Remover
30.0	30	Remover

Já que é uma ação perigosa, pois não há volta quando removemos um produto, vamos implementar um *pop-up* de aviso usando `data{}` :

```
<%= button_to "Remover", produto,
                         method: :delete,
                         class: "btn btn-danger",
                         data: { confirm: "Tem certeza que deseja remover?" } %>
```



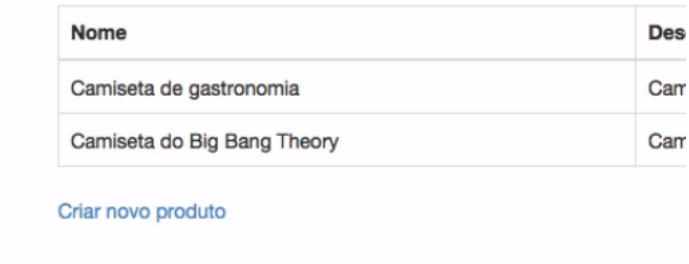
Podemos implementar para aparecer o nome do produto:

```
<%= button_to "Remover", produto,
                         method: :delete,
                         class: "btn btn-danger",
                         data: { confirm: "Tem certeza que deseja remover #{produto.nome}?" } %>
```

_path e *_url*

Toda vez que estamos na página das listas precisamos digitar a URL do formulário para acessá-lo. Vamos criar um link para facilitar esse trabalho. Aqui faz sentido utilizarmos o `link_to` (logo abaixo das tabelas):

```
<%= link_to "Criar novo produto", produtos_new_path %>
```



Perceba que às vezes usamos `_path` e às vezes usamos `_url` :

- *url*: sempre mostra o caminho completo (localhost:3000/produtos/new)". Serve para encaminhar para um outro domínio.
- *path*: mostra o caminho a partir da barra (/produtos/new). É relativo ao domínio local (localhost:3000).

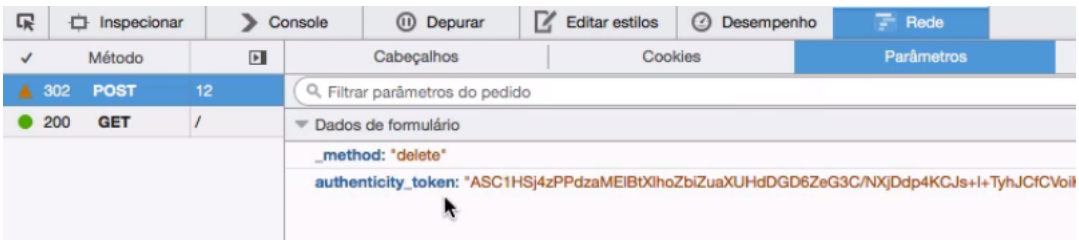
No botão de remover não usamos nenhuma das duas opções, mas se inspecionarmos o elemento no navegador, percebemos que o caminho é relativo, ou seja, do tipo `_path` .

O que acontece por de baixo dos panos com o DELETE

Ainda inspecionando o botão que remove o produto, podemos observar tal código:

```
... method="post" ... type="hidden" name="_method" value="delete" ...
```

Alguns navegadores podem não estar preparados para utilizarem todos os métodos do protocolo HTTP, como é o caso do *delete*. Por essa razão o *Rails* manda tudo via *post* com um parâmetro *delete* escondido. No mapa de rede do elemento conseguimos ver que realmente o que ocorre é um "POST" e não um "DELETE".



Na prática estamos usando os verbos do HTTP (delete, get, post), os quais são suportados pelo *Rails*, porém como o cliente é um navegador, este não suporta o *delete*. O *Rails*, então, faz um *post* em seu lugar.

Resources

Toda vez que precisamos gerar uma rota para alguma ação de produto, seguimos a convenção "/produtos/...":

```
Rails.appplication.routs.draw do
    post /produtos" => "produtos#create"
    get "/produtos/new" => "produtos#new"
    delete "/produtos/:id" => "produtos#destroy", as: :produto
    root "produtos#index"
end
```

O *Rails* consegue gerar todas as rotas para nós se passarmos esta informação. Assim elas conseguirão ficar menores e não precisaremos cair em repetição.

Conseguimos substituir tudo isso por uma rota só chamada *resources*:

```
Rails.application.routes.draw do
    resources :produtos
    root "produtos#index"
end
```

Com isso o *rake* gera diversas outras rotas, com outros verbos HTTP, que poderíamos precisar - para vê-las basta fazer `rake routes` no Terminal. Mas, quando não é o caso, podemos escolhê-las usando o `:only` :

```
Rails.application.routes.draw do
    resources :produtos, only: [:new, :create, :destroy]
    root "produtos#index"
end
```

Porém, perceba que não existe mais o "produtos_new", mas sim "new_produto". Vamos ter que mudar o *path* na página raiz:

```
<%= link_to "Criar novo produto", new_produto_path %>
```

Sempre que trabalhamos com o *Rails* devemos pensar nos *resourses*. Muitas vezes fará mais sentido utilizá-las, mas pode ser que uma rota solta também faça em algum caso específico.