01

API com produtos em JSON

Transcrição

[00:00] Ao longo do curso, nós já aprendemos a preencher formulários, gravar no banco, criar controles, rotas, views, vimos também como funciona validação de formulários e aprendemos a fazer notificações para o usuário. Nós já temos tudo isso pronto, mas o nosso cliente ainda não consegue consumir os produtos, aliás, nem nós conseguimos consumir os produtos, nós não conseguimos nem acessar os produtos sem acessar direto o banco de dados.

[00:31] Então, vamos criar uma rota para que consigamos listar todos os nossos produtos? Apesar de ser uma lógica de produtos, vamos fazer diretamente o ponto de acesso do cliente e, por fazer parte desse ponto de acesso, vamos criar um controller que seja exclusivamente para essa lógica de acesso? Então, vamos lá no Eclipse e vamos primeiro criar a rota.

[00:57] Abrimos o arquivo de rotas e criamos, vou até fazer um comentário aqui, “API”, criamos uma rota get para o nosso ponto de acesso. Eu vou chamar de api de produtos porque vamos retornar os produtos para o cliente, vamos fazer que ela esteja no controllers.ApiController e vamos retornar que produtos? Por enquanto, vamos retornar absolutamente todos os produtos, sem condição nenhuma? Então, vamos chamar de “todos”.

[01:39] Então, agora precisamos criar o que? Precisamos criar o controller, então vamos abrir um ProdutoController aqui só para ficar mais fácil e “Ctrl + N”, class, nós já estamos no pacote controllers, “ApiController”, temos que estender a classe controllers do Play, então controller do Play, dá Ok.

[02:05] O que precisamos aqui? Precisamos de um método que retorne todos os produtos, então “public Result todos()” e vamos retornar todos os produtos em uma mensagem de 200, eu vou importar aqui todos os elementos do pacote Play MVC.

[02:30] E como que buscamos os produtos, como que listamos os produtos? É a partir do DAO, então eu já vou aqui injetar o nosso DAO, “@inject” do javax inject, “private ProdutoDAO produtoDAO” e podemos utilizar um pouco de desenvolvimento ágil aqui, então eu vou “produtoDAO.todos” e isso vai me retornar uma lista de produtos.

[03:07] Repara que esse método ainda não existe, então vamos ter que fazer ele aqui. O método “todos” ainda não existe, então apertamos “Ctrl + 1”, cria método “todos” no ProdutoDAO. Ele é público, retorna uma lista de produtos e chama “todos”.

[03:34] E como que eu faço para pegar todos os produtos daquele meu buscador de produtos? Eu chamo o buscador de produtos, “produtos.all”, ele lista todos os produtos. Então, agora temos uma rota que lista todos os produtos, retorna um ok vazio, vamos retornar os produtos.

[04:02] Mas o Play ainda não entende direito como retornar uma lista de produtos, está vendo que tem um erro aqui? Nós não podemos simplesmente retornar uma lista de objetos Java, ele não sabe como lidar com isso, poderíamos transforma em string e ele entende, mas vamos fazer essa consulta e ver que não dá exatamente certo.

[04:27] Era “/api/produtos”, ele vai pegar todos os nossos produtos e, olha o que ele retorna, uma BeanList, que tem um size 1 e uma lista de models produto@1. Não é o que queremos retornar para o usuário, queremos retornar dados em algum formato legível.

[04:47] Vamos retornar em JSON? Como que transformamos esses dados em JSON? O Play já vem com uma biblioteca embutida que é a biblioteca do jackson, nós só precisamos chamar a biblioteca aqui, Json.toJson e passamos a nossa

lista de produtos e ele vai converter isso para um objeto JSON.

[05:12] E agora se fizermos a consulta, vamos retornar, vamos ter, um objeto válido, está vendo? O id 1, titulo livro de play com modelagem, código livro de play, tipo, descrição e preço.

[05:29] Mas ainda tem uma coisa estranha aqui, olha só, se eu, por exemplo, não tiver nenhum, delete from produto, produto cadastrado e eu fizer a consulta, ele retorna esse objeto vazio, que é uma lista, mas nós não sabemos do que. Se o cliente fizer essa consulta, talvez ele fique surpreso.

[05:52] Então que que fazemos? Vou fazer uma analogia aqui, eu não quero simplesmente pegar uma lista dos meus produtos em papel e entregar para o cliente, eu quero colocar elas dentro de um envelopinho e escrever no envelope "produtos". É a mesma coisa que vamos fazer aqui, podemos criar uma classe que funciona como esse envelope.

[06:13] Então, vamos lá, essa classe representa uma modelagem de alguma lógica do nosso sistema, então eu vou criar ela no pacote models, só que, vamos deixar bem claro o que que ela é, ela é um "EnvelopeDeProdutos". Nós não precisamos estender nada porque a única função dessa classe é servir de envelope para esses produtos, servir de wrapper.

[06:44] Então, que que ela vai ter? Ela vai ter uma "private list de produto" chamada "produtos", como que vamos receber essa lista? Nós que vamos fazer a envelopagem dos nossos produtos, então podemos criar isso no próprio construtor, "public EnvelopeDeProdutos" e recebe uma lista de produtos chamada "produtos".

[07:11] Atribuímos essa lista ao nosso atributo, à nossa variável, "produtos = produtos". Nós também precisamos de um getter porque o nosso atributo é privado, então "public list de produto getProdutos" retorna produtos.

[07:43] Agora nós já temos o nosso envelope, vamos usar ele, invés de retornar aqui uma lista de produtos direto, nós já podemos envelopar isso, "new EnvelopeDeProdutos" e recebemos essa lista direto dentro do envelope, vamos guardar isso em uma variável chamada "envelope" e ao invés de transformar em JSON os nossos produtos, transformamos o próprio envelope.

[08:14] Vamos ver como é que funcionou isso, como é que ficou. Atualizamos aqui, faz uma nova requisição e agora temos um envelope escrito "produtos" com uma lista, no caso a lista está vazia, mas agora, pelo menos o cliente já sabe que ele deveria ter uma lista de produtos, caso é que a lista está vazia, então está tudo ok, ele sabe o que ele recebeu.

[08:37] Podemos cadastrar aqui um produto novo agora, vamos lá, "produto/novo", vamos cadastrar o nosso livro de play de novo, livro de play, preço 12 e agora se fizermos a mesma requisição aqui, "api/produtos", temos um produto na nossa lista e se fizermos aqui a nossa requisição de novo, temos um produto dentro da lista que está dentro do nosso envelope.

[09:09] Então, o que que vimos nessa aula? Vimos como fazer para recuperar todos os produtos de um banco de dados, de uma dada tabela, sem filtro nenhum. Vimos também o suporte do Play nativo utilizando a biblioteca jackson para converter qualquer objeto em JSON e vimos também uma vantagem de envelopar os nossos produtos, o cliente tem uma mensagem mais clara do que está acontecendo ali.

[09:34] Na aula que vem, vamos aprender a criar esses filtros que permitam que o usuário, que o cliente, receba só os produtos com dado preço, ou com dado tipo, ou até com dado código.