01

O formulário de Produto

Transcrição

[00:00] Na última aula, vimos todo o processo de setup do nosso ambiente. Como instalar o Activator, como criar um projeto, como entrar naquele console e conseguir subir o servidor e como configurar tudo para rodar na IDE Eclipse.

[00:17] Agora precisamos codar, fazer o nosso trabalho. Primeiro, vamos subir o servidor de novo. Eu vou subir aqui o nosso servidor e eu vou usar um comando do Activator que agiliza pra caramba o desenvolvimento, ele é o til (~) e junto com o comando run, ele faz com que cada vez que um arquivo for salvo, o Activator reconheça isso, identifique que o arquivo foi alterado e recompile esse arquivo alterado, depois ele atualiza o servidor e faz tudo isso sozinho.

[00:56] Facilitando pra caramba a nossa vida porque nós não precisamos ficar parando servidor, recompilando todos os arquivos e subindo servidor de volta. Ele terminou de compilar aqui, eu vou só entrar no site para garantir que está tudo ok.

[01:11] Vamos botar a mão na massa? Vamos fazer uma loja e para ter uma loja, precisamos ter produtos para vender, para ter produtos precisamos cadastrar eles e como se cadastra um produto? Usando um formulário.

[01:21] Então, vamos fazer um formulário. Onde que fazemos esse formulário? Podemos criar uma página nova com um endereço novo, mas por enquanto vamos só substituir essa página que temos aqui, de boas vindas do Play. Ela é a raiz do nosso site e a maior parte dos sites usa a página raiz com o nome “index”, então vamos lá no Eclipse e vamos abrir a pasta app, que é aquela que eu falei que tem todo o nosso código, seja ele Java ou HTML e vamos aqui nesse pacote Views.

[01:58] E, por acaso, eles também usam a palavra “index” para designar a página inicial, esse “index.scala.html”. Então vamos abrir esse arquivo e ver o que tem aqui. Tem um monte de comentários aqui, uns códigos em Scala e aqui tem um código em Scala com a mensagem “Welcome to Play”. É esse trecho aqui que interessa para nós, é ele que define o conteúdo do nosso site.

[02:30] Então vamos alterar ele para criar um título aqui pro nosso produto. Vamos dizer que essa página, é uma página de cadastro de produto. Se eu vier aqui no terminal, ele diz que ele compilou mais uma fonte e que ele terminou com sucesso. Então agora podemos vim aqui, atualizar o site e ele já fez a alteração para nós.

[03:00] Agora vamos fazer esse formulário. Vamos vim aqui no Eclipse e criar um formulário. Para onde que ele vai levar? Temos que criar uma ação para ele, então eu vou criar uma action e eu vou definir que ela vai acessar o endereço “/produto/novo” porque estamos criando um novo produto. Como é um formulário de cadastro, eu vou usar o método post por segurança.

[03:29] Inserimos esse título dentro do formulário e vamos botar a mão na massa. Primeira coisa que um produto precisa ter, é um título, então vamos criar um campo input do tipo texto, “type=text” e que tenha o nome “titulo”, que é o nome da variável que vai ser enviada para onde quer que enviemos isso, para a url “/produto/novo”.

[03:59] Temos o nosso campo, vamos dar uma olhada aqui no site, temos um campo mas, esse campo podia ser de qualquer coisa, ele deveria ter um rótulo. Vamos criar um rótulo? O rótulo é uma label para o campo título, só que pra essa relação, entre o campo título e o rótulo título funcionar, precisamos que o nosso campo tenha um id, id título, igual a esse campo “for” aqui, esse atributo for.

[04:37] Agora nós já temos um campo com um rótulo assim que a página atualizar. Como funciona os rótulos? Se clicarmos no próprio rótulo, ele vai direto para o campo. Vamos precisar também do que? Todo produto deveria ter um identificador único, um código único. Então, vamos ter um código.

[05:00] Os produtos da nossa loja também vão poder ter vários tipos diferentes como livros, e-books, camisetas, que nem nós temos na casa do código. Nós também vamos precisar ter uma descrição para que possamos dizer pro nosso usuário, pro nosso cliente o que que é aquele produto e claro, todo produto nós precisamos ter um preço.

[05:25] Só que para não ficar fazendo toda a codificação aqui, eu tenho uma colinha e eu vou copiar aqui dela. Então ok, agora temos todos os nossos campos e temos também o nosso botão de enviar, que é um input do tipo submit.

[05:45] Tem uma mudança que eu preciso fazer que é: A descrição, ela pode ser extensa, então o ideal é usar uma textarea aqui, aí ela não precisa ter o tipo texto e precisamos fechar essa tag porque toda text area tem um conteúdo definido dentro.

[06:06] Vamos ver como é que ficou lá na página? Damos uma atualizada aqui e ok, temos um título, um código único, um tipo, uma descrição e um preço. Mas se preenchermos esse formulário, digamos, com, vou vender um Livro de Play, com código livro-play, com o tipo livro, a descrição “é um livro de Play” e preço 10 reais e eu vou cadastrar, só que, tomamos um erro na nossa cara.

[06:47] Porque aquela rota /produto/novo, esse endereço, não existe no nosso servidor, no nosso sistema, então precisamos criar isso. E como criamos uma rota, um endereço novo? Existe aquela pasta dentro do nosso projeto que é a pasta conf, que eram os arquivos de configuração do projeto e também das nossas rotas.

[07:09] Aqui dentro tem um arquivo chamado “rotas”, é ele que define todas as rotas, todos os endereços que o nosso site vai ter. No caso, eu vou apagar esses comentários para ficar um pouco mais legível. E, se você reparar, esse conteúdo que está aqui, é exatamente o conteúdo que é listado aqui. Então, se erramos um endereço, ele mostra para nós todos os endereços que estão disponíveis no nosso sistema, isso também é super legal.

[07:41] Eu posso explicar aqui um pouquinho, o primeiro endereço é da home, aquela página inicial quando acessamos localhost, na porta 9000, sem nenhum subdiretório. Está usando o método get, que é para nós podermos acessar no nosso browser e ver conteúdo. E está mapeada para esse objeto Java aqui, que no caso é um controller chamado HomeController, está no pacote controllers chamando HomeController com o método index.

[08:14] Então, o método Index, do objeto HomeController, que está no pacote controllers é referente a rota raiz. Vamos dar uma olhada nisso? Aqui temos o pacote controllers e o HomeController e procuramos o método index e vê o retorno dele, o retorno é um resultado, é o que queremos devolver pro nosso usuário.

[08:40] E o que que é esse resultado? É um tipo de resposta HTML, que pode ter qualquer um dos códigos HTTP como 200 para ok, 301 para redirect e 400 para uma requisição inadequada. No caso, estamos retornando um ok, que é o código 200 e renderizando a página index com essa mensagem aqui.

[09:07] O que queremos em seguida é criar uma rota própria nossa. Então, vamos apagar essas duas rotas aqui que nós não vamos ver nesse curso e vamos criar uma nova. Ela era um post e tínhamos mapeado ela para “produto/novo”.

[09:26] E onde queremos mapear ela? Toda lógica de produtos, faz sentido estar em um controlador de produtos, ou seja, no pacote controllers, no controlador de produtos, ou ProdutoController e criamos um método para salvar um novo produto, podemos chamar de salvaNovoProduto.

[09:51] Já temos uma nova rota e o Play já identifica isso, tanto é que se viermos aqui no terminal, ele lançou uma exceção porque a rota existe, ele sabe disso, mas ele não conseguiu encontrar o nosso controller, então o próximo passo é criar o controller e criar o método salvaNovoProduto.

[10:11] Então vamos lá, vamos aqui no pacote controllers e cria uma nova classe. A classe vai chamar ProdutoController e ela vai estender o objeto do tipo controller do Play MVC para ele conseguir identificar automaticamente as nossas rotas e saber mapear os endereços com elas.

[10:38] Eu vou alterar isso aqui para asterisco porque vamos importar várias coisas daqui, vamos criar um método, como é que era o método? Era um método público que retorna um resultado e o nome do método era salvaNovoProduto, eu vou copiar aqui e colar para garantir que não vou ter nenhum erro e que que ele vai retornar? Por enquanto podemos só retornar uma mensagem ok, falando que o formulário foi recebido.

[11:15] Parece ok, fizemos todos os passos, vamos aqui no terminal, ele fala que ele conseguiu compilar com sucesso, então vamos tentar enviar o formulário de novo. Eu vou voltar aqui e tentar cadastrar o mesmo produto novamente, tentando cadastrar, ele vai mandar a nossa mensagem de que o formulário foi recebido.

[11:38] Mas a lógica do formulário em si também faz parte da criação de um produto, então podemos alterar ela e levar ela pro controller de produtos. Vamos fazer isso? Nós primeiro podemos renomear o nosso arquivo index para fazer um pouco mais de sentido, vamos renomear ele para, digamos, ”formularioDeNovoProduto” porque é exatamente isso que ele é, um formulário de novo produto, eu vou até copiar isso aqui.

[12:12] Temos o nosso formulário, agora podemos criar um método aqui no nosso ProdutoController que mostre esse formulário mas, para isso, precisamos antes do que? De uma rota. Então vamos no arquivo de rotas antes, vamos criar uma nova rota, uma nova entrada nesse arquivo que seja um get, o endereço pode ser produto novo também porque daí conseguimos deixar isso um pouco mais fluido pro usuário mais pra frente.

[12:43] Só que o método não pode ser o mesmo, então vamos aqui alterar para formularioDeNovoProduto, que é o nome do nosso método, eu vou copiar isso aqui e vamos criar o nosso método.

[13:05] Agora a nossa compilação está com erro porque o método ainda não existe. Então vamos lá, “public Result formularioDeNovoProduto” e ele vai retornar, o que que ele vai retornar? Ele tem que renderizar aquela view lá, então vamos importar o pacote views.html, que é onde ficam todas as nossas views, vou importar todas elas e aqui vamos retornar uma mensagem de ok.

[13:36] A view também chama formularioDeNovoProduto e queremos renderizar ela, só que, se você lembra, aqui no HomeController, ela recebe uma mensagem, então vamos escrever uma mensagem qualquer aqui por enquanto. Ela vai receber a mensagem “Formulário de novo produto” ou “Cadastro de produto”.

[14:06] Aqui ainda está dando erro mas, se você vier aqui no terminal, ele mostra que ele conseguiu compilar com sucesso, isso pode acontecer de vez em quando, depois que ele termina de compilar, se você vier aqui e atualizar o seu projeto, esses erros tendem a sumir. Isso é bem comum, tem alguma discrepância entre o tempo de atualização do Eclipse com o Activator e acabamos perdendo essa agilidade.

[14:39] Mas parece que está tudo funcionando, não, olha só, tem um erro de compilação aqui do HomeController, por que? Vamos ver, ele está dizendo que ele não acha o index porque apagamos esse arquivo, mas nós não vamos mais usar o HomeController, então podemos só apagar ele inteiro, vou lá, apago o HomeController, como nós não vamos usar esses dois outros aqui também, vou apagar eles.

[15:08] E para não ter problemas, como nós não vamos entrar no mérito dos testes aqui nesse projeto, apagamos os pacotes de testes também. Em um próximo curso falamos de como funcionam os testes em Play.

[15:23] Depois de apagar todos esses arquivos, nós ainda temos uma última coisa a fazer, que é entrar aqui nas rotas e apagar essa rota do HomeController porque afinal ele não existe mais, então se ele continuar aqui, vamos tomar um erro na cara. Agora deve estar tudo certo.

[15:41] Vamos vir aqui no terminal, ele compilou tudo com sucesso, então entramos aqui e se tentarmos entrar na nossa home de novo, ela não existe mais, as únicas rotas que existem são as produto/novo e essa rota aqui representa os arquivos públicos, daquela pasta public que eu falei anteriormente.

[16:01] Então vamos acessar a página produto/novo e ela representa agora o nosso formulário. No próximo vídeo vamos dar uma olhada em como retirar os dados desse formulário, transformar em um modelo e salvar no banco.