02

Estrutura da Specification

Transcrição

[00:00] Agora que nós já estamos aqui trabalhando de fato bastante com o Framework do Spring Data, vendo os benefícios que ele traz para a nossa aplicação, como ele é mais ágil. O pessoal da Recrutei pediu algo que realmente é mais complicado, é mais difícil de fazer quando pensamos em JPA. Eles querem certo dinamismo na consulta deles para a entidade de funcionário.

```
CriteriaBuilder builder = //...

CriteriaQuery<Funcionario> query =
builder.createQuery(Funcionario.class);

Root<Funcionario> root = query.from(Funcionario.class);

List<Predicate> filtros = newArrayList<>();
if(nome != null) {
    Predicate filtro = builder.like(root.get("nome")), "%" +
nome + "%");
    filtros.add(filtro);
}
//outros filtros

query.where(filtros.toArray(new Predicate[0]));
```

COPIAR CÓDIGO

[00:26] Caso você já tenha conhecimento com o JPA, você pode estar familiarizado com o código que está na tela para fazer Querys dinâmicas. Você cria um CriteriaBuilder, cria um CriteriaQuery, cria um Root, faz a validação para verificar se o atributo que o cliente quer está nulo ou não, se não estiver nulo você cria a ação que você quer executar dentro do Builder para depois fazer a consulta.

[00:58] Fica meio complexo e dependendo também da quantidade de atributos que você quer deixar dinâmico, vai ter muita validação e vai ser difícil você dar manutenção porque a classe vai ficar muito grande. Aí vem a pergunta, conseguimos fazer esse critério, essa ação de Query dinâmica dentro do Spring Data e de uma forma mais rápida e com um código mais um enxuto? Isso que nós vamos ver nessa aula agora. Então vamos a nossa IDE.

[01:28] Aqui dentro da nossa IDE, vamos entrar na pasta raiz do nosso projeto, dentro do Project Explorer, dentro da Package Repository, vamos entrar dentro da FuncionarioRepository. Aqui dentro da FuncionarioRepository já temos o PagingAndSortRepository que nos dá toda a facilidade, adicionando ele dentro da aplicação ele já dá todo o poder para que possamos fazer Cruds e paginações dentro da nossa aplicação.

[01:57] Mas e agora precisamos dar também ao nosso Repository o poder de ele fazer Querys dinâmicas. A *feature* que o Spring Data utiliza para fazer Querys dinâmicas se chama Specification. Dentro da Specification ela já abstrai todos aqueles códigos que vimos lá no modelo do JPA. Então vamos ver agora como utilizar essa *feature* do Spring Data.

[02:26] Primeiramente nós vamos adicionar o poder para o nosso Repository para trabalhar com Specification. Como fazemos isso? Vamos ter que adicionar uma nova Interface que vai trazer esse poder ao nosso Repository. Aí temos que adicionar a interface `JpaSpecificationExecutor` porque essa interface vai ser responsável por executar as Specifications dentro do nosso Repository.

[02:56] E para utilizarmos essa Interface, nós precisamos informar a ela qual é a entidade que nós queremos trabalhar, que nós queremos especificar. No nosso caso aqui é Funcionario. Então ok, nós já temos aqui, já adicionamos ao nosso Repository o poder de ele também fazer Specifications. Então vamos salvar e agora nós vamos criar a nossa Specification de fato.

[03:23] Voltando para o nosso Project Explorer, vamos ir na pasta raiz do nosso projeto e vamos clicar com o botão direito "New > Package" e vamos criar uma Package chamada Specification. Dentro dessa Package de Specification nós vamos adicionar todas as Specifications que nós tivermos dentro do nosso projeto. Então agora que nós já temos aqui a nossa Package de Specification, vamos clicar com o botão direito em cima da Package, "New > Class" e dentro dessa Class nós vamos colocar SpecificationFuncionario. Então ok, já criamos aqui a nossa Specification de Funcionario.

[04:16] Agora nós precisamos criar aqui de uma maneira mais sucinta todo aquele código que nós vimos do JPA. Porque aquilo basicamente é o que faz a Query dinâmica. Nós vamos precisar também de um Root, de um CriteriaQuery e de um CriteriaBuilder. Nós vamos ver como isso é feito dentro do Framework do Spring Data.

[04:37] Então vou criar aqui um método público, estático e ele vai retornar esse método uma Specification do pacote `SpringFramework.data.JPA.Domain` e ele tem que retornar essa `Specification<Funcionario` . Agora eu preciso dar um nome. E qual é o nome? O nome eu vou dar baseado em qual é o atributo que eu quero validar, deixar dinâmico.

[05:12] No caso aqui eu estou fazendo de nome, um atributo nome do funcionário. Para que eu possa fazer a consulta dinâmica, o usuário da aplicação tem que me informar qual é o nome que ele quer consultar na base. Vou colocar aqui `String nome` . Lembra no código de JPA que eu precisava criar todos aqueles elementos, o Root e o CriteriaQuery? Então, aqui você não precisa se preocupar com isso. O Framework do Spring Data já entrega isso pronto para você. Então o que precisamos fazer?

[05:50] Vou dar um Return, abrir parênteses e agora eu quero um Root que já está pronto, eu quero um CriteriaQuery que já está pronto e quero um CriteriaBuilder que também já está pronto. Lembre-se que essas já são as variáveis já criadas, então letras minúsculas. Não coloque o objeto com letra maiúscula porque senão não irá funcionar.

[06:28] Então vou colocar aqui a seta e agora vou falar, "Já que eu já tenho as minhas variáveis criadas, pega para mim, por favor, o CriteriaBuilder" e agora a operação que eu quero executar, um `like`. E esse like eu vou precisar que o Specification pegue o atributo lá da minha entidade. Então `root.get` e agora eu passo qual é o atributo. O nome do atributo dentro de Funcionario se nós formos ver, é nome também. Eu quero que ele pegue o atributo de nome.

[07:07] E agora como eu estou fazendo um like, quando falamos de SQL, o like é feito entre porcentagens. Então vou colocar aqui ""% +" concatenar com o "nome" que o cliente informou no nosso parâmetro e vou concatenar no fim também com "%". Então `criteriaBiulder.like(root.get("nome"), "%" + nome + "%"` está pronto. Ele já vai fazer uma consulta lá dentro do nosso banco de dados por like. Então veja, aquele código gigantesco que vimos no começo da aula, nós reduzimos a duas linhas.

[07:47] É muito simples, é algo que o Spring Data te dá essa velocidade, essa agilidade de se preocupar simplesmente com o que você tem que se preocupar mesmo e não com a criação desses objetos. Agora vamos ter que criar um novo relatório para que o usuário possa fazer lá dentro as suas Querys de forma dinâmica. Então nós vamos lá dentro, deixa eu fechar aqui, vamos vir na pasta raiz, vamos vir aqui dentro da PackageService.

[08:28] Dentro da nossa PackageService nós vamos criar uma nova classe e vamos chamar essa classe de RelatorioFuncionarioDinamico. Esse será o nome da nossa classe. Aqui no RelatarioFuncionarioDinamico eu vou precisar de um objeto injetado do nosso Repository que tem o poder de executar Specifications. Então vamos lá, `private final FuncionarioRepository` vai ser nosso FuncionarioRepository.

[09:27] Agora vou criar o construtor Public o nome da nossa classe e o Spring vai criar para nós esse camarada aqui pelo nosso construtor, `this.funcionarioRepository = funcionarioRepository` criado pelo Framework do Spring. Agora seguindo aquele padrão que nós sempre seguimos. Criar a classe que vai ser a inicial do nosso projeto. Então `void inicial(Scanner scanner)` . Ele vai ter que passar aqui um Scanner porque vamos precisar que ele escreva alguns valores dentro do Console. Então ele passou o Scanner.

[10:22] Agora vou falar para ele, `System.out.println ("Digite um nome")` , o nome que ele quer fazer a consulta. Agora vou pegar `String nomeString = scanner.next` . Aqui nós já pegamos o nome que ele deseja consultar. E como vamos fazer para que ele retorne essa consulta dinâmica? Essa consulta vai retornar uma lista de funcionários porque ele vai retornar todos os funcionários que tem na base de dados que tenha algo parecido, que tenha algo entre o início e o fim com o que o usuário digitar aqui.

[11:14] Então vamos lá, ele vai retornar uma lista de funcionários. Vou chamar isso de Funcionarios e aí vamos pegar lá. No nosso FuncionarioRepository.findAll, se nós vermos, o findAll quando adicionamos o poder de executar Specifications ele agora consegue receber uma Specification via construtor. Então vamos utilizar o findAll, deixa eu minimizar aqui.

[11:48] Aí agora no findAll nós vamos colocar uma `findAll(Specification.where)` para ele fazer uma consulta e agora nós vamos colocar a nossa Specification que nós criamos que é a SpecificationFuncionario. E dentro da SpecificationFuncionario eu quero fazer a consulta dinâmica pelo quê? Pelo nome. E vamos informar aqui o nome que o cliente informou dentro do console. Então aqui já temos uma Specification que vai retornar uma lista com apenas as especificações, o like que o cliente escreveu.

[12:39] Veja, com o mínimo de código possível nós já fizemos basicamente aquele código complexo e grande que nós vimos no exemplo do JPA. Então a

seguir vou mostrar para vocês como fazer essas especificações com os outros atributos. Nos vemos na próxima aula.