

Spring MVC I: Criando aplicações web

9%

AULA 02

Cadastro de produtos

ATIVIDADES

01 Vídeo 121min

02 Vídeo 226min

03 Vídeo 313min

04 Cadastro de produtos

05 Sobre a utilização do @AutoWired

06 Criando o nosso controller de produtos

07 Criando o formulário de cadastro

08 Pegando os parâmetros do formulário

09 Melhorando o nosso binding

10 Adicionando o hibernate ao nosso proj...

11 Configurando o hibernate

12 Transformando a classe Produto em e...

13 Criando o ProdutoDAO

14 Usando o ProdutoDAO no Controller

15 Fazendo o spring encontrar os daos

16 Adicionando a classe JPAConfiguration ...

17 Configurando transações

AULAS

02. Cadastro de produtos

OUTROS LINKS

Fórum

Trocar Curso

Altair Lage25.1k xp

09 Melhorando o nosso binding

PRÓXIMA ATIVIDADE

Nós já estamos recebendo os parâmetros, correto? Mas e se o nosso formulário tiver 30 campos ou mais, o que vai acontecer ? Teremos um método recebendo muitos parâmetros que seria super difícil de ler! Além disso, toda vez que adicionarmos um campo no formulário teremos que lembrar de criar o parâmetro no método também. Em outras palavras, ao invés de receber diversos parâmetros, o ideal é receber uma classe que representa todos esses parâmetros! Portanto, crie a classe `Produto` no pacote `br.com.casadocodigo.loja.models` com os parâmetros, `titulo`, `descricao` e `paginas` como atributos, e também, o seus *getters* e *setters*.

Por fim, no método `grava()` do `ProdutosController`, ao invés de passar todos aqueles parâmetros, envie apenas o parâmetro `Produto produto`, em seguida, imprima o parâmetro `produto`. **Lembre-se de implementar o `toString()` da classe `Produto` para visualizar os valores dos atributos.** Cole aqui o código da sua classe `Produto` e o método `grava()`.

Responda

<> INSERIR CÓDIGO

↔ EXPANDIR

📘 FORMATAÇÃO

TIRAR DÚVIDA

RESPONDER