16

Separação das responsabilidades do modelo e da view

Como vimos no vídeo, nosso *ManagedBean* ainda está com muitas informações do livro espalhadas pela classe. É importante separarmos a responsabilidade de cadastrar um livro, da responsabilidade de ser um livro, principalmente porque usaremos as informações do livro em várias partes do nosso sistema.
Vamos agora separar as responsabilidades do modelo de dados das ações de cadastrar um livro:

1. Com o arquivo **LivroBean.java** aberto no editor do Eclipse, selecione todos os atributos de livro e vá no menu **Refactor** > **Extract Class...**;
2. Na nova caixa de diálogo, em **Class name:** coloque apenas **Livro**, marque o *checkbox* **Create *getters* and *setters*** para gerar os métodos na nova classe de modelo e em **Field name:** coloque **livro**, este será o atributo em **LivroBean** que apontará para a nova classe de modelo. Clique em **OK**

```
package br.com.caelum.livraria.bean;

public class Livro {

    private String titulo;
    private String isbn;
    private double preco;
    private String dataLancamento;

    public Livro() {
    }

    public String getTitulo() {
        return titulo;
    }

    public void setTitulo(String titulo) {
        this.titulo = titulo;
    }

    public String getIsbn() {
        return isbn;
    }

    public void setIsbn(String isbn) {
        this.isbn = isbn;
    }

    public double getPreco() {
        return preco;
    }

    public void setPreco(double preco) {
        this.preco = preco;
    }

    public String getDataLancamento() {
        return dataLancamento;
    }
```

```
    public void setDataLancamento(String dataLancamento) {
        this.dataLancamento = dataLancamento;
    }
}
```

1. O Eclipse gerou uma nova classe com os atributos de livro, seus *getters* e *setters*. Sendo assim, já podemos apagar todos os *getters* e *setters* que estavam na classe **LivroBean**. Como ainda existe um atributo privado `livro` nela, vamos criar um *getter* para ele, método este que será utilizado pelos componentes da página para popular o livro.

```
// código omitido
private Livro livro = new Livro();

public Livro getLivro() {
    return livro;
}
// código omitido
```

1. Reinicie o servidor tomcat e recarregue a página **livro.xhtml** no navegado. Deu erro?
2. Esse erro ocorreu porque ainda não atualizamos a ligação (*binding*) dos componentes da página com a nova estrutura do **LivroBean**. Agora, não temos mais os atributos do livro diretamente no *ManagedBean*, mas acessamos eles através do atributo privado `livro` .

Vamos atualizar a ligação dos componentes da página com o *ManagedBean*:

1. Abra o **livro.xhtml** no editor do Eclipse e atualize as *Expressions Languages (EL)*, que fazem o *binding* dos componentes, para acessarem os atributos pelo objeto **livro** do *ManagedBean*. A nova EL do *input* que recebe o título do livro ficará assim: `#{livroBean.livro.titulo}` . Aproveite para modificar as demais.

```
<!-- trecho omitido -->
<h:outputLabel value="Titulo:" for="titulo" />
<h:inputText id="titulo" value="#{livroBean.livro.titulo}" />

<h:outputLabel value="ISBN:" for="isbn" />
<h:inputText id="isbn" value="#{livroBean.livro.isbn}" />

<h:outputLabel value="Preço:" for="preco" />
<h:inputText id="preco" value="#{livroBean.livro.preco}" />

<h:outputLabel value="Data de Lançamento:" for="dataLancamento" />
<h:inputText id="dataLancamento" value="#{livroBean.livro.dataLancamento}" />
<!-- trecho omitido -->
```

1. Abra novamente a página **livro.xhtml** no navegador e pronto! Tudo funcionando novamente.
2. Cole aqui o código dos arquivos **LivroBean.java** e **livro.xhtml**:

Responda

INSERIR CÓDIGO FORMATAÇÃO

Arquivo LivroBean.java:
```

```
Arquivo livro.xhtml:
```
Arquivo livro.xhtml:

```