05

Mãos na massa: Listando os produtos

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Com alguns livros cadastrados no banco de dados, chegou a hora de listá-los. Primeiramente, busque-os na base de dados, então crie o método `listar()` na classe `ProdutoDAO`, que devolve uma lista de produtos. Dentro desse método, utilize o `createQuery()` para realizar a *query* que busca todos os produtos. Por fim, utilize o método `getResultList()` para retornar a lista de produtos:

```
public List<Produto> listar() {
    return manager.createQuery("select p from Produto p", Produto.class)
        .getResultList();
}
```

2) Com a busca de livros no banco de dados pronta, crie o método `listar()` no `ProdutoController`, que mapeia o endereço `produtos`. Esse método irá buscar os livros no banco, criar um objeto do tipo `ModelAndView` que devolve a *view* `produtos/lista` e adicionar a lista de produtos na *view*. Ao final, retorne o objeto de `ModelAndView`:

```
@RequestMapping("produtos")
public ModelAndView listar() {
    List<Produto> produtos = produtoDAO.listar();
    ModelAndView modelAndView = new ModelAndView("produtos/lista");
    modelAndView.addObject("produtos", produtos);

    return modelAndView;
}
```

3) Agora que todos os produtos contidos no banco de dados estão sendo devolvidos, exiba-os em uma *view*. Crie a JSP **lista.jsp** dentro do diretório **src/main/webapp/WEB-INF/views/produtos**. Dentro desse arquivo, adicione o seguinte código:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset=UTF-8>
    <title>Livros de Java, Android, iPhone, PHP, Ruby e muito mais - Casa do Código</title>
</head>
<body>
    <h1>Lista de Produtos</h1>
    <table>
        <tr>
            <td>Título</td>
            <td>Descrição</td>
            <td>Páginas</td>
        </tr>
        <c:forEach items="${produtos}" var="produto">
```

```
            <tr>
                <td>${produto.titulo}</td>
                <td>${produto.descricao}</td>
                <td>${produto.paginas}</td>
            </tr>
        </c:forEach>
    </table>
</body>
</html>
```

4) Em `ProdutoController`, os métodos `gravar` e `listar` respondem à mesma URL, então indique a sua intenção utilizando os métodos de requisição do HTTP, como o `GET` para listar os dados e `POST` para inserir, por exemplo:

```
@Controller
public class ProdutosController {

    @RequestMapping(value="produtos", method = RequestMethod.POST)
    public String gravar(Produto produto) {
        // código omitido
    }

    @RequestMapping(value="produtos", method = RequestMethod.GET)
    public ModelAndView listar() {
        // código omitido
    }

        // restante do código omitido

}
```

5) Reinicie o Tomcat e acesse http://localhost:8080/casadocodigo/produtos (http://localhost:8080/casadocodigo/produtos). Na listagem dos produtos, os dados que possuem acentos estão com problemas de apresentação. Então defina qual é o *encoding* da aplicação, através de um filtro do Spring. Para isso, adicione o seguinte método na classe `ServletSpringMVC`:

```
@Override
protected Filter[] getServletFilters() {
    CharacterEncodingFilter encodingFilter = new CharacterEncodingFilter();
    encodingFilter.setEncoding("UTF-8");

    return new Filter[] {encodingFilter};
}
```

6) Para finalizar, todos os mapeamentos da classe `ProdutosController` começam com `/produtos`, mas ficar digitando e dar manutenção neles é complicado. Para resolver isso, adicione a anotação `@RequestMapping("/produtos")` na classe `ProdutosController` e apague o prefixo de todos os outros métodos:

```
@Controller
@RequestMapping("/produtos")
public class ProdutosController {

    @Autowired
```

```
    private ProdutoDao produtoDao;

    @RequestMapping("form")
    public ModelAndView form() {
        ModelAndView modelAndView = new ModelAndView("produtos/form");
        modelAndView.addObject("tipos", TipoPreco.values());

        return modelAndView;
    }

    @RequestMapping(method=RequestMethod.POST)
    public String gravar(Produto produto) {
        produtoDAO.gravar(produto);
        return "produtos/ok";
    }

    @RequestMapping(method=RequestMethod.GET)
    public ModelAndView listar() {
        List<Produto> produtos = produtoDao.listar();
        ModelAndView modelAndView = new ModelAndView("produtos/lista");
        modelAndView.addObject("produtos", produtos);

        return modelAndView;
    }
}
```