Desenvolvimento Web I

Aula 14 - Desenvolvendo uma livraria virtual – Parte 3

Apresentação

Olá, meu amigo! Na aula anterior, você estudou a modelagem das classes relacionadas ao Servlet controlador da livraria virtual e de alguns de seus arquivos JSP, em especial, a da tela inicial e a da tela de catálogo de livros. Você viu que a implementação mostrada possuía uma duplicação de código nos arquivos JSP, não é? Duplicação de código, normalmente, é uma coisa ruim e que pode ser evitada ou reduzida, e o caso específico de duplicação de código nos arquivos JSP da livraria virtual será tema de análise desta aula.

Além disso, nesta aula, você estudará a implementação de outra funcionalidade do sistema da livraria virtual e será requisitado a implementá-la, também, no projeto da locadora virtual.

Desejo-lhe uma boa leitura!

- Reconhecer a importância de reusar código em arquivos JSP.
- Descrever como adicionar cabeçalhos e rodapés a diferentes arquivos JSP.

Promovendo o reuso de cabeçalhos e rodapés de arquivos JSP

Na aula anterior, observamos que os arquivos JSP responsáveis por montar as telas do sistema da Livraria virtual possuem uma espécie de cabeçalho e rodapé em comum. Se observarmos bem, vamos notar que as páginas iniciam com a declaração dos taglibs a serem utilizados, que, no caso, são:

```
1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
2 <%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
```

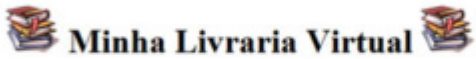
Podemos imaginar que nem todos os arquivos JSP irão usar a taglib fmt, por exemplo. Isso é verdade, inclusive, quando olhamos o código-fonte do arquivo livraria.jsp, responsável por montar a tela de entrada do sistema. Entretanto, não existe um efeito negativo maior em se colocar a declaração de uma taglib que não é utilizada no arquivo. Dessa forma, vamos considerar a situação onde todas as páginas irão conter a declaração das duas taglibs, ok?.

Além da declaração das taglibs, todas as páginas possuem um código inicial HTML em comum, mostrado a seguir:

```
1  <html>
2  <head>
3     <title>
4        Minha Livraria Virtual
5     </title>
6  </head>
7  <body bgcolor="#FFFFFF">
8     <center>
9        <hr>
10       <br>
11       <h1>
12          <img width="50" src="../imagem/livro.gif">
13          <font size="+3">Minha Livraria Virtual</font>
14          <img width="50" src="../imagem/livro.gif">
15       </h1>
16    </center>
17       <br>
18       <hr>
```

Esse código é responsável por montar a parte superior da tela, como mostrado na Figura 1.

Figura 01 - Cabeçalho utilizado em todas as telas da Livraria Virtual



Além disso, todas as páginas do sistema têm uma referência de direitos autorais da Livraria Virtual, como mostrado na **Figura 2.**

Figura 02 - Rodapé utilizado em todas as páginas do sistema

O código responsável pela geração desse rodapé foi visto na aula anterior e é replicado a seguir:

```
    <br>
    <hr>
    <center><em>Copyright © 2010 Livraria Virtual. </em></center>
</body>
</html>
```

Note que, embora o conteúdo replicado não seja muito grande, a quantidade de arquivos que o sistema tem pode ser razoavelmente grande, ou, pelo menos, suficiente para ocorrerem problemas durante as etapas de manutenção do sistema. Por exemplo, imagine que seja necessário atualizar alguma informação, como mudar de 2010 para 2015 o período de copyright mostrado no rodapé das páginas. Todas as páginas teriam que ser editadas para se realizar essa alteração.

Além do trabalho de ter que abrir todas as páginas e editar a mesma alteração, ainda corre-se o risco de esquecer-se de alterar algumas páginas e a informação de copyright ser mostrada corretamente em algumas páginas e erroneamente em outras. Nesse caso, as consequências dessa inconsistência de informação não

seriam graves, mas imagine se a informação a ser alterada fosse o meio de contato com a empresa (telefone, endereço etc.). Alguns clientes poderiam visualizar e utilizar informação de contato inválida, levando a perda de clientes, oportunidades de negócio, vendas etc.

Agora que já visualizamos bem o problema e as suas consequências, vamos ver como podemos eliminar a duplicação de código mostrada. Uma alternativa que temos é a criação de um arquivo cabeçalho para inclusão do código que aparece no início dos arquivos JSP e de um arquivo de rodapé, contendo a parte final do código a ser também incluída nos arquivos JSP do sistema. O código-fonte desses dois arquivos são mostrados nas Listagens 1 e 2.

Como você pode notar, o conteúdo desses arquivos é exatamente igual ao conteúdo que é replicado em todos os arquivos JSP. Pode até parecer estranho para você ter um arquivo JSP que abre marcadores <html> e <body> sem os fechar ou vice-versa, mas isso fará sentido em seguida.

```
1  <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
2  <%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
3  <html>
4    <head>
5      <title>
6        Minha Livraria Virtual
7      </title>
8    </head>
9
10   <body bgcolor="#FFFFFF">
11     <center>
12       <hr>
13       <br>
14       <h1>
15         <img width="50" src="../imagem/livro.gif">
16         <font size="+3">Minha Livraria Virtual</font>
17         <img width="50" src="../imagem/livro.gif">
18       </h1>
19     </center>
20
21       <br>
22       <hr>
```

Listagem 1 - Código-fonte do arquivo cabecalho.jsp

```
<br>
  <hr>
    <center><em>Copyright © 2010 Livraria Virtual. </em></center>
  </body>
</html>
```

Listagem 2 - Código-fonte do arquivo rodape.jsp

Agora que temos os arquivos de cabeçalho e rodapé criados, esse conteúdo precisa ser removido dos arquivos JSP anteriormente criados, ou seja, dos arquivos livraria.jsp e catalogo.jsp. Muito cuidado ao fazer isso, ok? Não vá remover código a mais ou a menos! Deve ser removida apenas a parte inicial e final desses arquivos, cujo conteúdo já está contemplado pelos arquivos de cabecalho.jsp e rodape.jsp.

Muito bem. O próximo passo agora é fazer com que o conteúdo do arquivo cabecalho.jsp seja adicionado no início de todos os outros arquivos JSP do sistema, e que o conteúdo do arquivo rodape.jsp seja adicionado no final desses arquivos. Uma primeira forma de realizar isso é utilizar a diretiva de inclusão de conteúdo <%@ include %> em arquivos JSP. É como ter que colocar expressão a seguir em todos os demais arquivos JSP, onde ". . ." representa o conteúdo específico de cada arquivo:

```
<%@ include file="/cabecalho.jsp" %>
 ...
<%@ include file="/rodape.jsp" %>
```

O uso dessas instruções é responsável por incluir, no arquivo JSP, o conteúdo de cabecalho.jsp no início do arquivo e o conteúdo de rodape.jsp no final do arquivo. Note que ambos os arquivos estão sendo incluídos com o seu caminho absoluto (iniciando com “/”), ou seja, eles devem ser colocados diretamente na pasta WebContent. Se você não adicionar o caminho absoluto do arquivo que deseja incluir então ele será buscado na própria pasta do jsp que o está incluindo. Apesar de ser uma boa solução, essa alternativa demanda cuidado, pois precisamos colocar essas linhas em todos os arquivos do projeto.

Existe uma outra abordagem chamada de Web Fragments que permite que através de uma configuração no projeto você inclua de forma mais automática arquivos como o nosso cabeçalho e rodapé em várias documentos baseado em um padrão de URL. Não vamos utilizar Web Fragments no nosso projeto mas você pode estudar mais e ampliar seus conhecimentos no link https://blogs.oracle.com/swchan/servlet-30-web-fragmentxml.

Atividade 01

1. No sistema da Livraria Virtual, implemente o reuso de código do cabeçalho e do rodapé. Execute e verifique o funcionamento correto das telas inicial e de catálogo de livros.
2. Implemente um reuso de código similar no seu projeto da Locadora Virtual.

Demais funcionalidades da Livraria Virtual

No restante desta aula e na próxima, nós iremos estudar a implementação das demais funcionalidades da Livraria Virtual. Lembrando, claro, que um sistema real precisa ter muitas outras funcionalidades, além das mostradas aqui.

Cada funcionalidade (ou tela) nova do sistema exige uma nova URL, e seu mapeamento no Servlet controlador deve ser colocado através de anotação. No código a seguir, podemos ver as URLs que serão desenvolvidas no restante desta aula. Essas URLs são determinadas na propriedade urlPatterns com os valores "/livros/livraria", ”/livros/catalogo”, ”/livros/detalhesLivro”, ”/livros/mostrarCarrinho”, ”/livros/comprar” e ”/livros/recibo”. Esteja certoque digitou corretamente as URLs para não que elas sejam mapeadas da forma certa para o controlador.

```
1 ...
2 @WebServlet(name="Controlador", urlPatterns={"/livros/livraria","/livros/catalogo", "/livros/detalhe
3 public class ServletControladorLivraria extends HttpServlet {
4 ...
5
```

As funcionalidades que iremos ver estão relacionadas a:

- ver detalhes de um determinado livro;
- mostrar e alterar o conteúdo do carrinho de compras;
- confirmar a compra;
- visualizar o recibo.

Os links para a maioria das funcionalidades já se encontram na tela de catálogo (ver **Figura 3**) e a implementação de cada uma dessas telas será mostrada nas próximas seções desta e da próxima aula!

Figura 03 - Tela de catálogo. Possui link para detalhar livro, ver carrinho de compras e finalizar compras.

Atividade 02

1. Altere o servlet controlador colocando o conteúdo mostrado, de forma a habilitar as URLs das funcionalidades a serem implementadas.
2. Considerando as funcionalidades que iremos trabalhar no sistema da Livraria Virtual, crie URLs para essas funcionalidades no contexto do seu projeto de Locadora Virtual e configure-as no arquivo web.xml da Locadora Virtual.

Detalhando o conteúdo de um livro

Na página de catálogo de livros existe um link de visualizar detalhes de cada livro, correto? Esse link é a URL /livros/detalhesLivro e é acionado ao se clicar no nome do livro desejado (ver **Figura 3**). Por exemplo, ao acessar o link do primeiro

livro mostrado na **Figura 3**, a tela da **Figura 4** será apresentada, mostrando informações sobre o livro selecionado.

Figura 04 - Tela mostrando informações sobre o livro selecionado

A implementação dessa funcionalidade é relativamente simples, já que na classe livraria.negocio.LivrariaBean existe o método getLivro() que, dado o código do livro, retorna o livro se ele existir no estoque. Para você relembrar o código relativo a esse método, ele é mostrado novamente a seguir.

```
1  ...
2  public class LivrariaBean {
3     private String idLivro = "0";
4     public void setIdLivro(String id) {
5        this.idLivro = id;
6     }
7     public Livro getLivro() throws LivroNaoEncontradoException {
8        return (Livro) sistema.getLivro(idLivro);
9     }
10 ...
11 }
```

No caso da implementação da funcionalidade detalhar livro, esse bean é utilizado para, dado o código de um livro (seu atributo idLivro), realizar a consulta no estoque por esse livro (método getLivro()). Vamos, então, ao conteúdo do arquivo **detalhesLivro.jsp**. Primeiramente não esqueça de adicionar o comando <%@

include file="/cabecalho.jsp" %> sempre no início e o comando <%@ include file="/rodape.jsp" %> sempre no fim dos arquivos JSP, como visto anteriormente. Em seguida segue o código para que possamos utilizar o LivrariaBean. Esse código é o mesmo encontrado no arquivo catalogo.jsp.

```
<jsp:useBean id="livrariaBean" class="livraria.negocio.LivrariaBean" scope="page" >
    <jsp:setProperty name="livrariaBean" property="sistema" value="${sistemaLivraria}" />
</jsp:useBean>
```

Em seguida, precisamos verificar se o parâmetro idLivro foi passado. Isso é feito pelo código a seguir, por meio da instrução <c:if >. Caso o valor de idLivro não seja vazio, criamos uma variável local de nome id com o valor igual ao do parâmetro idLivro (uso do <c:set >). Depois disso, utilizamos essa variável para configurar o atributo idLivro do objeto LivrariaBean (uso do <jsp:setProperty >). Por fim, o livro retornado pelo método getLivro() de LivrariaBean (expressão ${livrariaBean.livro}) é armazenado em uma variável local de nome livro, criada por outra instrução <c:set >:

```
<c:if test="${!empty param.idLivro}">
<c:set var="id" value="${param.idLivro}"/>
<jsp:setProperty name="livrariaBean" property="idLivro" value="${id}" />
<c:set var="livro" value="${livrariaBean.livro}" />
```

Muito bem, agora estamos com uma variável contendo o livro que queremos detalhar na página. Isso é feito por meio do código mostrado a seguir. Apresentamos o título do livro através da expressão ${livro.titulo}. De forma similar, apresentamos os autores, ano de publicação, descrição e preço do livro. Para apresentação do preço, utilizamos o marcador <fmt:formatNumber >, para formatar o valor adequadamente:

```
<h2>${livro.titulo}</h2>
Autoria de  <em> ${livro.autores}</em>
(${livro.ano})<br>  <br>
<h4>Descrição</h4>
<blockquote>${livro.descricao}</blockquote>
<h4>Preço: <fmt:formatNumber value="${livro.preco}" type="currency"/></h4>
```

Já que o usuário está vendo os detalhes de um livro, imagina-se que ele está interessado em comprar. Dessa forma, adicionamos o código a seguir para criar uma URL chamando a página de catálogo e passando o parâmetro Add com o

código do livro que está sendo visualizado. Lembrando o que foi discutido na aula anterior, se o usuário clicar nesse link, o livro será adicionado ao carrinho, não é mesmo?

```
<c:url var="url" value="/livros/catalogo" >
   <c:param name="Add" value="${id}" />
</c:url>
   <p><strong><a href="${url}">Adicionar ao carrinho</a>
</c:if>
```

Esse foi o código a ser executado se o código do livro for passado corretamente (ver fechamento com o). Vamos, agora, colocar apenas mais uma URL, que é a de voltar à página de catálogo, sem adicionar nenhum livro ao carrinho (parâmetro Add é vazio).

```
<c:url var="url" value="/livros/catalogo" >
   <c:param name="Add" value="" />
</c:url>
<a href="${url}">Continuar comprando</a></strong></p>
```

Só reforçando, todas as páginas JSP que possuem a URL iniciada com /livros/ incluem automaticamente o cabeçalho e o rodapé contidos nos arquivos cabecalho.jspf e rodape.jspf, o que faz com que não seja necessário adicioná-los a essa página.

Atividade 03

---

1. Implemente a página de detalhes de livro, conforme explicado. Execute a Livraria Virtual e exercite essa funcionalidade, verificando que sua implementação está funcionando corretamente.

2. Implemente essa funcionalidade de detalhar produto no contexto da Locadora Virtual, executando, depois, o sistema e verificando que sua implementação está funcionando corretamente.

Leitura Complementar

Para complementar a fixação do conhecimento, faça uma busca na internet sobre a configuração dos cabeçalhos e rodapés. A pesquisa por palavras-chave, como JSP, prelude e coda podem ajudar a trazer resultados mais relevantes nos principais mecanismos de busca da internet.

Resumo

Na aula de hoje vimos como utilizar a inclusão de arquivos para evitar a duplicação de código em páginas JSP. Vimos tanto como incluir eles através da diretiva <%@ include %> quanto através da edição do arquivo web.xml, bem como a vantagem de utilizar cada um desses métodos. Além disso, criamos a página de detalhamento de um livro e tornamos os seus links funcionais.

Na próxima aula implementaremos mais algumas funcionalidades no nosso sistema e veremos como incluir também um pouco de JavaScript para melhorar o funcionamento da página. Nos vemos!

Autoavaliação

1. Descreva quais os problemas gerados pela duplicação de código em arquivos JSP.
2. Descreva formas de se evitar essa duplicação, em particular, a duplicação de conteúdos de cabeçalho e rodapé.

Referências

AHMED, K. Z.; UMRYSH, C. E. **Desenvolvendo aplicações comerciais em Java com Java J2EE e UML.** Rio de Janeiro: Ciência Moderna, 2003. 324 p.

BASHAM, Bryan; SIERRA, Kathy; BATES, Bert. **Head First Servlets and JSP**: passing the Sun Certified Web Component Developer Exam (SCWCD). 2. ed. [s.l.]: O'Reilly Media, 2008.

CATTELL, Rick; INSCORE, Jim. **J2EE:** criando aplicações comerciais. Rio de Janeiro: Campus, 2001.

HALL, Marty. **More Servlets and JavaServer Pages (JSP).** New Jersey: Prentice Hall PTR (InformIT), 2001. 752 p. Sun Microsystems Core Series. Disponível em: <http://pdf.moreservlets.com/>. Acesso em: 20 ago. 2012.

HALL, Marty; BROWN, Larry. **Core Servlets and JavaServer Pages (JSP).** 2. ed. New Jersey: Prentice Hall PTR (InformIT), 2003. 736 p. Sun Microsystems Core Series (Core Technologies, 1). Disponível em: <http://pdf.coreservlets.com/>. Acesso em: 20 ago. 2012.

HYPERTEXT Transfer Protocol -- HTTP/1.1. **methods definition.** Disponível em: <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html>. Acesso em: 20 ago. 2012.

J2EE Tutorial. Disponível em: <http://docs.oracle.com/javaee/7/tutorial/>. Acesso em: 10 mar. 2015.