Unidade 3: Recursos Adicionais dos Servlets

Como Construir um com Duas Partes

Prof. Daniel Caetano

Objetivo: Preparar o aluno para construir Servlets para compor aplicações Web.

Bibliografia: QIAN, 2007; DEITEL, 2005.

INTRODUÇÃO

Nas aulas anteriores foram apresentados os conceitos da linguagem Java e também como construir um Servlet simples. Nesta aula apresentaremos mais algumas das capacidades de um Servlet, de maneira que seja possível transferir informações de um Servlet a outro.

Nesta aula serão apresentados:

1) Métodos doGet e doPost
2) Redirect e Forward
3) Alterando os dados da requisição

1. PARÂMETROS PARA UM SERVLET

Para colher os dados do usuário, é preciso aprender um pouco mais sobre o protocolo **http** e sobre os métodos das Servlets.

O protocolo HTTP tem duas formas principais de mandar uma requisição:

Requisição do tipo POST: Serve para enviar dados
- Enviadas, normalmente, através de formulários

Requisição do tipo GET: Serve para solicitar dados
- Enviadas, normalmente, através de links ou da barra de endereços

Em ambos os casos é possível indicar parâmetros. Estes parâmetros poderão ser lidos no servlet através do método getParameter:

```
String valorLido = request.getParameter("nomeDoParametro");
```

Cada um dos casos será apresentados a seguir.

1.1. Parâmetros POST

Os parâmetros POST são passados por meio de formulários XHTML. O nome do **servlet** é indicado através do parâmetro "action" da tag "form".

Cada campo de entrada - <**input** />, por exemplo - possui um parâmetro chamado "name". Tudo que o usuário digita no campo fica associado àquele nome de parâmetro. Por exemplo:

```
<form action="Imc" method="post">
        <p>Peso: <input type="text" name="peso" />kg</p>
        <p>Altura: <input type="text" name="altura" />m</p>
        <p><input type="submit" value="enviar"></p>
</form>
```

Neste caso, o valor digitado no campo "peso" vai estar associado ao parâmetro "peso" e o valor digitado no campo "altura" vai estar associado ao parâmetro "altura".

1.2. Parâmetros GET

Parâmetros GET são passados pela URL, similar ao que acontece quando passamos parâmetros para um programa pela linha de comando do Windows, Linux ou DOS.

Por exemplo, considere o endereço abaixo, que indica o caminho de execução de um servlet chamado **Imc**:

http://localhost:8080/Imc

... temos que ***Imc* é o nome do servlet** e, para passar parâmetros para esse programa, podemos usar o caractere "**?**" para separar o que é nome do programa do que são os parâmetros.

Por exemplo, se quisermos definir que um parâmetro chamado "**peso**" seja igual a **70**, basta indicar da seguinte forma:

http://localhost:8080/Imc?peso=70

Mas... agora uma outra dúvida pode surgir: e se quisermos passar ***mais de um*** parâmetro? Por exemplo, e se quisermos indicar, além do parâmetro **peso=70**, quisermos indicar também o parâmetro **altura=1.70**?

Existe uma solução simples e elegante para isso: iremos indicar todos os parâmetros separados pelo símbolo "**&**", como apresentado abaixo:

http://localhost:8080/Imc?peso=70&altura=1.70

2. REDIRECT E FORWARD

Quando um servlet é executado, eventualmente podemos querer redirecionar o navegador ou a execução para um outro endereço ou servlet. Neste caso, usaremos os comandos **redirect** ou **forward**.

2.1. Redirect

Eventualmente um Servlet pode querer redirecionar o usuário para uma outra página / JSP ou mesmo outro Servlet. Isso pode ser conseguido com a seguinte instrução:

response.sendRedirect("url")

Por exemplo, para redirecionar para a página do google, usa-se:

response.sendRedirect("http://www.google.com/");

2.2. Forward

Algumas vezes, um serviço pode ser executado por uma combinação de Servlets. Por exemplo: já existe um servlet que calcula um determinado valor em função do tamanho de um armazém. Este Servlet, porém, exige os dados de entrada em "metros" e, segundo os requisitos do cliente, é necessário que os dados de entrada estejam em "pés". Assim, podemos construir um servlet intermediário, que recebe os dados em "pés", converte para "metros" e, então, aciona o outro servlet já existente.

Para acionar um outro servlet repassando os dados da requisição, usa-se a seguinte sequência:

```
// Pega o "despachador de requisições" para a url desejada
RequestDispatcher rd = request.getRequestDispatcher("url");
// Envia dados de requisição e resposta
rd.forward(request,response);
```

Lembrando que os dados da requisição podem ser alterados antes que esse despacho seja feito.

3. ALTERANDO OS DADOS DE UMA REQUISIÇÃO

Quando encaminhamos uma requisição para outro servlet, com o comando **forward**, é comum queremos inserir alguns dados na requisição.

Esses dados ganham o nome de atributos e **sempre** devem ser OBJETOS. Por exemplo: para colocar uma String na requisição, fazemos o seguinte:

```
Sting nome = "Fulano de Tal";
request.setAttribute("cliente", nome);
```

Para guardar um número, fazemos o seguinte:

```
Integer anos = 50;
request.setAttribute("idade", anos)/
```

Isso irá armazenar os dados na requisição com os nomes de "cliente" e "idade", respectivamente.

Quando essa requisição for repassada a outro servlet, o novo servlet poderá obter os dados da seguinte forma:

```
String nome = (String) request.getAttribute("cliente");
Integer idade = (Integer) request.getAttribute("idade");
```

4. BIBLIOGRAFIA

DEITEL, H.M; DEITEL, P.J. **Java: como programar** - Sexta edição. São Paulo: Pearson-Prentice Hall, 2005.

QIAN, K; ALLEN, R; GAN, M; BROWN, R. **Desenvolvimento Web Java**. Rio de Janeiro: LTC, 2007.