Tutoriais

Comandos Condicionais

por: Ulisses Telemaco Neto
ulisses@jspbrasil.com.br

Embora já possamos escrever algumas aplicações em JSP com o que já aprendemos até agora, elas serão ainda fracas. Um dos grandes potenciais de qualquer linguagem de programação, é a utilização de controles de fluxo (condicionais e loops) para executar diferentes partes de um programa baseado em testes. Nas próximas duas seções, aprenderemos os seguintes tópicos:
comandos condicionais;
comandos de loops.

Como foi dito em lições anteriores, o JSP nos permite construir páginas unindo HTML e comandos Java. Mostraremos nessa lição como utilizar as conhecidas rotinas condicionais do Java em páginas JSP. Três comandos serão tratadas a seguir: if.. else, operador condicional, switch.

Comando IF

Condicionais if contém a palavra-chave if, seguida de um teste booleano, e de uma instrução (na maioria das vezes um bloco de instruções) para ser executada se o teste for verdadeiro.

Exemplo 01:

```
<html>
<body>

<%-- AS DUAS PROXIMAS LINHAS CRIA UMA VARIÁVEL
QUE RECEBE O VALOR DA DATA ATUAL DO SERVIDOR --%>

<%
java.util.Date dateNow = new java.util.Date();
int hourNow = dateNow.getHours();
%>


<%-- AS PROXIMAS LINHAS MISTURA HTML E JAVA
E PRODUZ COMO RESULTADO UMA SAUDAÇÃO QUE DEPENDE DA HORA --%>

<% if ((hourNow >= 5) && (hourNow < 13)) { %>
<font face="verdana">Bom Dia!!!</font>

<% } else if ((hourNow >= 13) && (hourNow < 19)) { %>
< font face="verdana">Bom Tarde!!!</font>

<% } else if ((hourNow >= 19) && (hourNow < 24)) { %>
<font face="verdana">Bom Noite!!!</font>

<% } else { %>
<font face="verdana">Boa Madrugada!!!</font>
<% } %>

</body>
</html>
```

Operador Condicional

Uma alternativa para o uso das palavras-chaves if e else em uma instrução condicional é usar o operador condicional, algumas chamado de operador ternário.
O operador ternário é uma expressão, significando que ele devolve um valor (diferentemente do if mais geral, o qual pode resultar em qualquer instrução ou bloco sendo executado). O operador ternário é muito útil para condicionais muito curtos ou simples, e tem a seguinte formato:
test ? trueResult : falseResul

O teste é uma expressão que devolve true e false (semelhante ao teste da instrução if). Se o teste for verdadeiro retorna o valor de trueResult, caso contrário, devolve o valor de falseResult. Vamos como podemos usar o operador condicional para obter o mesmo efeito do código anterior:

Exemplo 2:

```
<html>
<body>

<%-- AS DUAS PROXIMAS LINHAS CRIA UMA VARIÁVEL
QUE RECEBE O VALOR DA DATA ATUAL DO SERVIDOR --%>

<%
java.util.Date dateNow = new java.util.Date();
int hourNow = dateNow.getHours();
%>

<%
String mensagem;
mensagem = ((hourNow < 12)? "Onde você vai almoçar hoje?" : "Onde você almoçou
hoje?");
%>
<font face="verdana">Olá, Tudo bem? <%= mensagem %></font>

</body>
</html>
```

Apesar deste código parecer, a uma primeira vista, um pouco menos fácil de entender que o anterior, ele é menor e portanto mais prático.

Condicionais Switch

Na instrução switch, o teste (um tipo primitivo de byte, char, short ou int) é comparado com cada valor em questão. Se um valor coincidente é achado, a instrução (ou instruções) depois do teste é executada. Se nenhum valor for encontrado, a instrução default é executada. Vamos analisar o exemplo abaixo:

Exemplo 3:

```
<html>
<body>

<%-- AS DUAS PROXIMAS LINHAS CRIA UMA VARIÁVEL
QUE RECEBE O VALOR DA DATA ATUAL DO SERVIDOR --%>

<%
java.util.Date dateNow = new java.util.Date();
int monthNow = (dateNow.getMonth()+1);
%>

<%
String mes;

switch(monthNow){
case 1: mes="Janeiro"; break;
case 2: mes="Fevereiro"; break;
case 3: mes="Março"; break;
case 4: mes="Abril"; break;
case 5: mes="Maio"; break;
case 6: mes="Junho"; break;
case 7: mes="Julho"; break;
case 8: mes="Agosto"; break;
case 9: mes="Setembro"; break;
case 10: mes="Outubro"; break;
case 11: mes="Novembro"; break;
default: mes="Dezembro"; break;
}
%>

<font face="verdana"> Nós estamos em <%= mes %></font>

</body>
</html>
```

Esse código atribui a variável monthNow o valor do mês atual. Note que na instrução nós incrementamos o mês em uma unidade porque o método "getMonth()" retorna 0 para o mês de janeiro, 1 para fevereiro e assim por diante.
Observe que a limitação significativa no Java é que os testes e valores podem ser somente de tipos primitivos. Você não pode usar tipos primitivos maiores (long, float) ou objetos dentro de um switch, nem pode testar para nenhuma outra relação senão a igualdade. Isso limita a utilidade do switch para todos os casos exceto os mais simples, if´s aninhados podem funcionar para qualquer espécie de teste em qualquer tipo.