

JSP

Java Servers Pages



Introdução aos JSPs

- Java **S**erver **P**ages
- HTML + Código Java
- Tem o objetivo de facilitar a construção de páginas dinâmicas



Exemplo JSP

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="Content-Type" content="text/html;
  charset=ISO-8859-1" />
  <title>Exemplo JSP</title>
</head>
<body>

  <ul>
    <% for (int i =0; i<10; i++) { %>
      <li><%=i %></li>
    <% } %>
  </ul>

</body>
</html>
```

- Execute o JSP no servidor, olhe no browser o código fonte da página...

Elementos do JSP

- Scriptlets
- Expressions
- Declarations



Scriptlets

- São um bloco de código Java inserido no JSP
- Um scriptlet deve iniciar com `<%` e terminar com `%>`

`<%`

```
ArrayList<String> lista = new ArrayList<String>();  
lista.add("Algoritmos");  
lista.add("LTP");  
lista.add("Web");
```

`%>`

Expressions

- Código Java inserido no JSP
- Deve resultar em uma String, que será exibida na página resultante (HTML)
- Uma expression deve iniciar com `<%=` e terminar com `%>`

```
<%=lista.get(0) %>
```

Declarations

- Código Java inserido no JSP
- Utilizada para declarar **atributos** e **métodos**
- Uma expression deve iniciar com **<%!** e terminar com **%>**

<%!

String nome;

*double calculaSoma(int i, int j) {
 return i+j;*

}

%>

Comentários

- Dentro dos scriptlets, os comentários são iguais aos utilizados em classes Java:

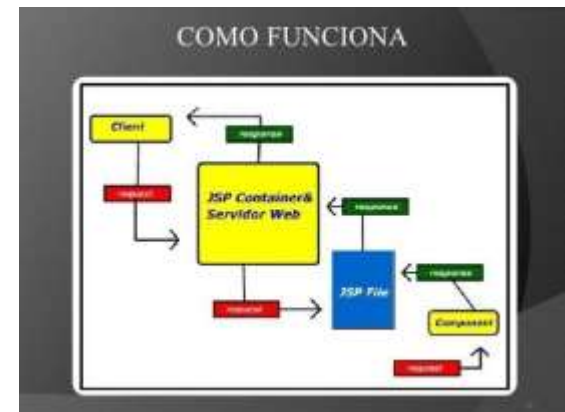
```
<%  
    //Comentário de linha  
  
    /*  
    Comentário de bloco  
    */  
%>
```

- Fora dos scriptlets, a notação é:

```
<!-- Comentário -->
```


Diretivas

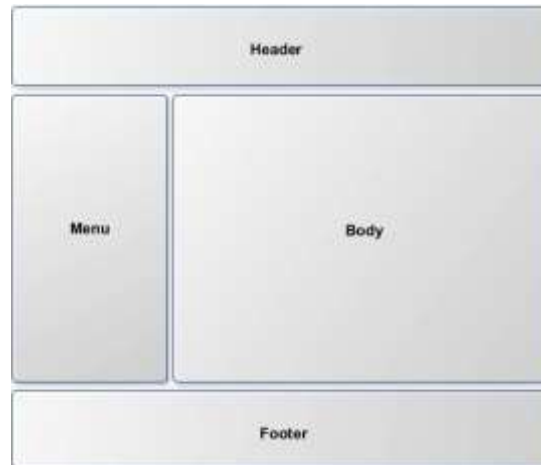
- Elas começam com **<%@** e terminam com **%>**
- Podem ser de 3 tipos:
 - **Include:**
 - Inclui código de um arquivo externo no JSP
 - **Taglib:**
 - Define uma tag library
 - **Page:**
 - Define propriedades da página



Diretiva include

- Permite incluir um arquivo externo no JSP em tempo de tradução.
- Ideal para incluir informações que se repetem em várias páginas, ex: Menu, Cabeçalhos..

```
<%@ include file="menu.jsp" %>
```



Diretiva taglib

- Permite referenciar tag libraries na página
- Podemos utilizar uma infinidade de tags para construir páginas JSPs

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

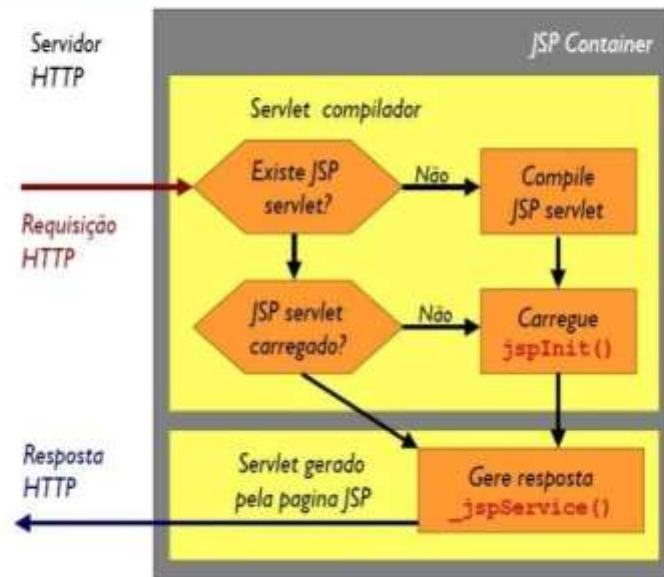


Obs. Vamos utilizar essa diretiva para estudar JSTL.

Diretiva page

- Define propriedades para a página JSP
- Podemos definir diversas propriedades:
 - language
 - contentType
 - pageEncoding
 - import

Como funciona JSP



Diretiva page

- Os atributos **language**, **contentType** e **pageEncoding** definem a linguagem e codificação da página
- Ela é definida automaticamente quando criamos páginas JSPs utilizando o Eclipse.

```
<%@ page language="java" contentType="text/html;  
charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
```

Diretiva page - Import

- O atributo **import** é utilizado para importar classes que serão utilizados na página JSP
- Funciona de forma parecida com o import em classes
- Devemos utilizar o ***pacote + nome da classe***, ou ***pacote + .**** para importar todas as classes do pacote

```
<%@ page import="java.util.ArrayList,  
java.util.Calendar" %>
```

```
<%@ page import="java.util.*" %>
```

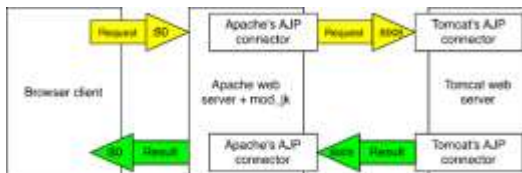
Tags XML

- Temos algumas tags XML que podem ser utilizadas diretamente no JSP.
- JSP:INCLUDE - Inclui a página indicada em tempo de solicitação no documento corrente, mas não compila o documento que será carregado.

```
<jsp:include page="pagina.jsp"></jsp:include>
```

- JSP:FORWARD - Direciona o request para a página de destino.

```
<jsp:forward page="pagina.jsp"></jsp:forward>
```



JavaServer Pages (JSP)



1. INTRODUÇÃO

Você aprenderá como construir Aplicações Web utilizando a tecnologia *JavaServer Pages*.

Na plataforma Java, os *Servlets* foram a primeira tecnologia para produção de conteúdo dinâmico na Web. Como você já aprendeu os *Servlets* são programas Java que executam do lado do servidor. Esses programas são inicializados pelos containeres (*Servidores Web*) e são capazes de processar as requisições enviadas pelos clientes.

JavaServer Pages é uma nova tecnologia, desenvolvida pela Sun, também com o objetivo de produzir conteúdo dinâmico. A principal diferença entre as tecnologias, *Servlets* e JSP, está na maneira como o conteúdo dinâmico é produzido.

Observe que enquanto os *Servlets* incorporam todo conteúdo dinâmico das páginas dentro da lógica dos programas, as páginas JSP separam a apresentação do conteúdo da lógica de negócios.

A diferença é sutil, mas em termos práticos os *Servlets* são programas totalmente escritos em Java, que produzem como saída um conteúdo dinâmico. Entretanto, as páginas JSP incorporam toda lógica da *Aplicação* (escrita em Java) dentro do conteúdo XHTML. Isso permite que as páginas possam ser escritas mais facilmente.

Bom estudo!

2. ARQUITETURA

Ao contrário dos *Servlets* que precisam ser compilados antes da execução, as páginas JSP são simplesmente copiadas para o diretório do Servidor Web.

O interessante é que no caso das páginas JSP, todo processo de compilação é de responsabilidade do *Servidor Web*. Mas, para isso, o servidor precisa traduzir a página JSP (.jsp) em uma classe Java (.java), e em seguida compilar o código transformando em *bytecodes* (.class).

Além disso, no final do processo, o que temos é simplesmente um ***Servlet Java***! Então, o container carrega o *Servlet* (método `init()`) e aguarda as solicitações por meio do método **`request()`**:

Processo de Compilação JSP



3. ESTRUTURA BÁSICA

A estrutura básica de uma página JSP é bastante simples, uma vez que o código pode ser incluído diretamente no corpo do documento XHTML.

O início de um bloco contendo código JSP/Java sempre deve iniciar com a tag `<%` e terminar com `%>`:

```
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3   "http://www.w3.org/TR/html4/loose.dtd">
4
5 <html>
6   <head>
7     <meta http-equiv="Content-Type" content="text/html;">
8     <title>Bem-vindo!</title>
9   </head>
10  <body>
11    <%= "Olá, seja bem-vindo!" %>
12  </body>
13 </html>
```

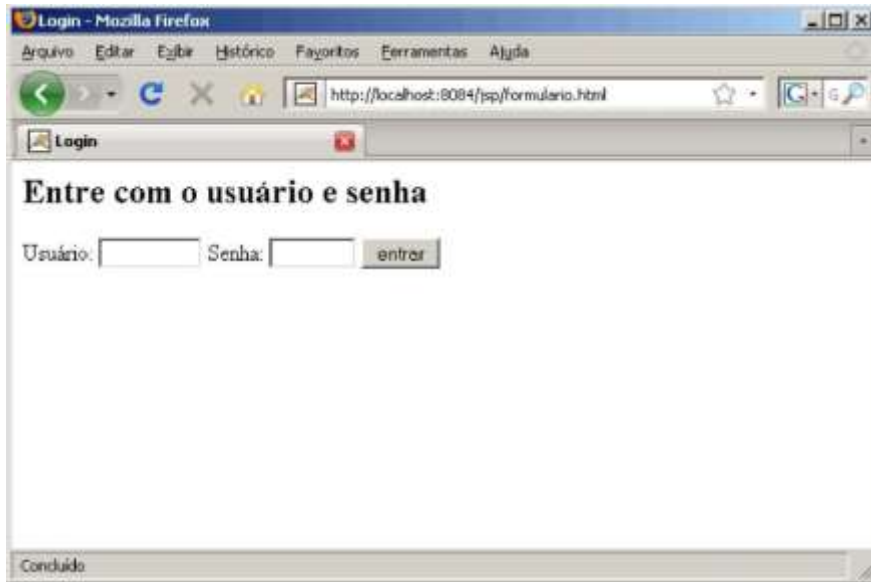
Da mesma maneira que os Servlets, a tecnologia JSP permite o processamento fácil e rápido de formulários.

Para tanto, podemos utilizar dois modelos:

No **primeiro modelo** o formulário é criado separadamente em uma página estática XHTML e o processamento é realizado por meio de uma página JSP. Nesse caso, a ação do formulário (linha 9) faz referência a página JSP que realizará o processamento:

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <html>
3   <head>
4     <title>Login</title>
5     <meta http-equiv="Content-Type" content="text/html;">
6   </head>
7   <body>
8     <h2> Entre com o usuário e senha </h2>
9     <form action="login.jsp" method="POST">
10       Usuário: <input type="text" name="txtUsuario" value="" size="8" />
11       Senha: <input type="password" name="txtSenha" value="" size="6" />
12       <input type="submit" value="entrar" />
13     </form>
14   </body>
15 </html>
```

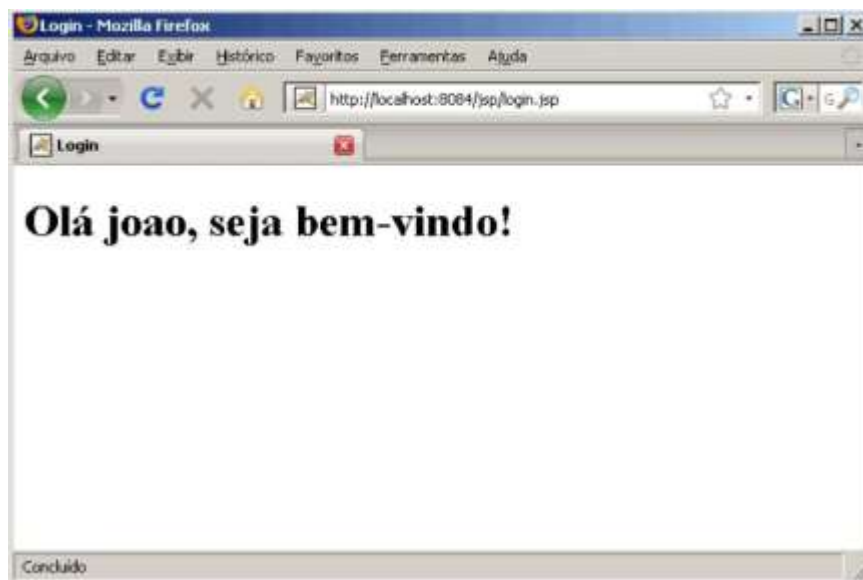
Observe:



Agora, a página JSP (login.jsp), recebe os dados fornecidos no formulário, realiza o processamento e produz a resposta para o cliente:

```
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3   "http://www.w3.org/TR/html4/loose.dtd">
4
5 <html>
6   <head>
7     <meta http-equiv="Content-Type" content="text/html; ">
8     <title>Login</title>
9   </head>
10  <body>
11    <%
12      String usuario = request.getParameter("txtUsuario");
13      String senha = request.getParameter("txtSenha");
14
15      if ( usuario.equals("joao") )
16        if ( senha.equals("123456") )
17          out.println("<h1>Olá " + usuario + ", seja bem-vindo! </h1>");
18        else
19          out.println("<h1>Senha inválida</h1>");
20      else
21        out.println("<h1>Usuário inválido</h1>");
22    %>
23  </body>
24 </html>
```

Vejamos:



No **segundo modelo**, tanto o formulário (XHTML) quando o processamento (JSP) é realizado em uma única página.

Note que nesse caso, a ação do formulário faz referência a própria página JSP (login.jsp):

```
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3   "<a href='\"http://www.w3.org/TR/html4/loose.dtd\"'>
4 <html>
5   <head>
6     <meta http-equiv="Content-Type" content="text/html;">
7     <title>Exemplo de JSP </title>
8   </head>
9   <body>
10    <%
11      String usuario = request.getParameter("txtUsuario");
12      String senha = request.getParameter("txtSenha");
13
14      if ( usuario == null && senha == null  ){
15        %>
16        <h2> Entre com o usuário e senha </h2>
17        <form action="login.jsp" method="POST">
18          Usuário: <input type="text" name="txtUsuario" value="" size="8" />
19          Senha: <input type="password" name="txtSenha" value="" size="6" />
20          <input type="submit" value="entrar" />
21        </form>
22        <%
23          }else{
24            if ( usuario.equals("joao") )
25              if ( senha.equals("123456") )
26                out.println("<h1>Olá " + usuario + ", seja bem-vindo! </
27                h1>");
28              else
29                out.println("<h1>Senha inválida</h1>");
```

```
29         else
30             out.println("<h1>Usuário inválido</h1>");
31     }
32     %>
33     </body>
34 </html>
```

3. FUNDAMENTOS SOBRE JSP

Elementos de *script*

Os elementos de *script* são utilizados para embutir código Java dentro das páginas JSP. Há três maneiras para isso: **declarações**, **expressões** e **scriptlets**.

Declarações

As **declarações** são um tipo de elemento de *script* utilizado para inserir métodos, constantes ou **declarações** de variáveis dentro das páginas JSP.

Os elementos de *script* do tipo **declarações** são especificados como se segue:

```
<%!
// inicio das declarações
...
código Java
...
// fimdas declarações
%>

1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3   "<a href='\"http://www.w3.org/TR/html4/loose.dtd\"'>http://www.w3.org/TR/html4/loose.dtd">
4
5 <html>
6   <head>
7     <meta http-equiv="Content-Type" content="text/html;">
8     <title>Exemplo de Declarações</title>
9   </head>
10  <%!
11      public int soma(int v1, int v2){
12          return v1+v2;
13      }
14
15      public int multiplicacao(int v1, int v2){
16          return v1*v2;
17      }
18  %>
19  <body>
20    <h2>
21      Soma de 10 + 20 = <%= soma(10,20) %>
22    </h2>
```

```
23         <h2>
24             Multiplicação de 10 * 20 = <%= multiplicacao(10,20) %>
25         </h2>
26     </body>
27 </html>
```

Expressões

Quando você utiliza dos elementos de *script* do tipo **expressões**, o código Java incluído na página é avaliado e convertido em texto. O resultado é colocado diretamente no documento na localização do elemento.

Observe que com as **expressões** não precisamos utilizar o comando *out.println* para exibir informações na página.

As **expressões** são definidas pelo bloco:

```
    <%= ... código java ... %>
1  <%@page contentType="text/html" pageEncoding="UTF-8"%>
2  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3      "http://www.w3.org/TR/html4/loose.dtd">
4
5  <html>
6      <head>
7          <meta http-equiv="Content-Type" content="text/html;">
8          <title>Exemplo de Expressões </title>
9      </head>
10     <%!
11         // declaracao do contador
12         private static int contador;
13
14         public static int getContador() {
15             contador++;
16             return contador;
17         }
18     %>
19     <body>
20         <h2> Essa página foi visitada <%= getContador() %> vezes</h2>
21     </body>
22 </html>
```

Note que a variável **contador** foi declarada como *static* (estática), o que faz com que essa variável seja compartilhada por todas as requisições realizadas no servidor.

Scriptlets

Os **scriptlets** permitem que você adicione blocos de código Java em suas páginas JSP.

Ao contrário das **declarações** que são limitadas à declaração de métodos, variáveis e constantes, e as **expressões** que transformam o resultado em texto, os **scriptlets** possibilitam incluir blocos completos de código Java no interior das páginas JSP.

Os **scriptlets** são delimitados pelo bloco:

```
<%
...
código Java
...
%>
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3   "http://www.w3.org/TR/html4/loose.dtd">
4 <html>
5   <head>
6     <meta http-equiv="Content-Type" content="text/html;">
7     <title>Exemplo de Scriptlets</title>
8   </head>
9   <body>
10    <%
11      String strNum = request.getParameter("txtNum");
12      if ( strNum == null ) {
13        <%
14          <h1>Tabuada </h1>
15          <form action="tabuada.jsp">
16            Número: <input type="text" name="txtNum"   size="5" />
17            <input type="submit" value="calcular" />
18          </form>
19          <%
20            }
21            else{
22              int num = Integer.parseInt(strNum);
23              for (int i = 1; i <= 10; i++){
24                <%
25                  <h2> <%= num %> * <%= i %> = <%= (num*i) %> </h2>
26                <%
27              }
28            }
29          <%
30        </body>
31 </html>
```

Diretivas

As **diretivas** permitem que o desenvolvedor envie instruções para o *container* durante o processo de tradução da página. Além disso, elas permitem ao programador especificar diversas configurações para a página JSP.

Você pode especificar **diretivas** a uma página JSP por meio dos delimitadores:

```
<% @ nome_da_diretiva atributo="valor" %>
```

Há três tipos de diretivas que podem ser utilizadas:

- **Page:** a diretiva **page** é utilizada para especificar todas as dependências que a página atual pode possuir. Por exemplo, caso a página atual necessite de algum pacote Java para ser compilada teríamos:
-

```
<%@ page import="java.util.*,      java.text.*" %>
```

- **Include:** com a diretiva **include** é possível mesclar a página atual conteúdos de outras páginas. Por exemplo, para incluir uma a página **exemplo.jsp** devemos especificar a diretiva:

```
<%@ include file="exemplo.jsp"%>
```

- **Taglib:** a diretiva **taglib** permite aos desenvolvedores utilizar outros conjuntos de tags na construção das páginas JSP. Esses conjuntos podem simplificar a codificação das páginas.

Para ilustrar a utilização das diretivas, vamos construir uma Aplicação contendo três páginas:

cabecalho.jsp

```
1 <h1>Relógio</h1>
2 <hr/>
```

principal.jsp

```
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <%@page import="java.util.Date,java.text.SimpleDateFormat" %>
3
4 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
5   "http://www.w3.org/TR/html4/loose.dtd">
6
7 <%!
8     private Date hora = new Date();
9     private SimpleDateFormat f = new SimpleDateFormat("HH:mm:ss");
10 %>
11
12 <html>
13     <head>
14         <meta http-equiv="Content-Type" content="text/html;">
15         <title>Relógio</title>
16         <style type="text/css">
17             h1.relogio{
18                 color: #FF0000;
19                 background: #000000;
20                 width: 200px;
21                 text-align:center;
22             }
23         </style>
24     </head>
25     <body>
26         <%@ include file="cabecalho.jsp" %>
27
28         <h1 class="relogio">
29             <%= f.format(hora) %>
30         </h1>
31
32         <%@ include file="rodape.jsp" %>
```

```
33     </body>
34 </html>
```

rodape.jsp

```
1 <br/>
2 <hr/>
3 <center>
4     Copyright &copy; 2008 - Todos os direitos reservados
5 </center>
```

Ações

As **ações** permitem ao desenvolvedor especificar atividades que serão executadas durante a solicitação da página, as quais são definidas pelo elemento `<jsp:ação>` e `</jsp: ação>`, no qual a ação representa o nome da ação que será executada.

A seguir são apresentadas algumas das principais **ações** disponíveis no JSP:

`<jsp:include>`

Elas permitem incluir outros conteúdos (paginas JSP) na página atual:

`<jsp:forward>`

Com essa ação você pode redirecionar a página atualmente carregada para outra página:

`<jsp:param>`

São utilizados para especificar parâmetros extras para as ações *include*, *forward* ou *plugin*.

`<jsp:useBean>`

A ação **useBean** declara uma nova variável no conteúdo da página JSP e associa essa variável a um objeto Java.

O exemplo a seguir demonstra como utilizar as ações no desenvolvimento de páginas JSP:

cabecalho.jsp

```
1 <h1> Seja bem-vindo! </h1>
```

pagina.jsp

```
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3     "http://www.w3.org/TR/html4/loose.dtd">
4 <html>
5     <head>
6         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
7         <title>Exemplo de Ações</title>
8     </head>
9     <body>
```

```

10      <%
11          String pagina = request.getParameter("grpPagina");
12          if ( pagina == null ){
13              %>
14              <jsp:include page=" cabecalho.jsp" />
15              <form action="pagina.jsp" method="get">
16                  <input type="radio" name="grpPagina" value="1" checked="checked" />
Pagina 1 <br/>
17                  <input type="radio" name="grpPagina" value="2" /> Pagina 2 <br/>
18                  Digite seu nome <input type="text" name="txtNome" />
19                  <input type="submit" value="enviar" />
20              </form>
21
22      <%
23          }else{
24              String op = request.getParameter("grpPagina");
25              String nome = request.getParameter("txtNome");
26              if ( op.equals("1") ){
27                  %>
28                  <jsp:forward page="pagina1.jsp" >
29                      <jsp:param name="nome" value="<%= nome%>" />
30                  </jsp:forward>
31
32      <%
33          }else{
34              %>
35              <jsp:forward page="pagina2.jsp" >
36                  <jsp:param name="nome" value="<%= nome%>" />
37              </jsp:forward>
38      <%
39          }
40      }
41      %>
42  </body>
43 </html>

```

pagina1.jsp

```

1  <h2>Olá <%= request.getParameter("nome") %>, você está na Página 1</h2>

```

pagina2.jsp

```

1  <h2>Olá <%= request.getParameter("nome") %>, você está na Página 2</h2>

```

4. CONSTRUINDO UMA APLICAÇÃO WEB COM JSP

Anteriormente aprendemos como desenvolver programas web utilizando *Servlets*. Agora, no exemplo a seguir criamos uma pesquisa sobre a intenção de votos dos candidatos a eleição.

Observe que vamos utilizar o mesmo exemplo, no entanto, faremos uma implementação utilizando a tecnologia *JavaServer Pages*:

```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html;
4     charset=iso-8859-1">
5     <title>Pesquisa Intenção de Votos</title>
6   </head>
7   <%!
8     public static int cand1 = 0;
9     public static int cand2 = 0;
10    public static int cand3 = 0;
11
12    public void addVoto(int op) {
13      switch(op) {
14        case 1:  cand1++; break;
15        case 2:  cand2++; break;
16        case 3:  cand3++; break;
17      }
18    }
19  %>
20  <body>
21    <%
22      String voto = request.getParameter("Candidato");
23      if ( voto == null ) {
24        %>
25        <h1> Pesquisa : Eleições </h1>
26        <h3> Selecione o candidato desejado e clique no botão votar. </h3>
27        <form action="votacao.jsp" method="get">
28          <fieldset>
29            <input name="Candidato" type="radio" value="1" />
30            Antônio das Coxinhas <br/>
31            <input name="Candidato" type="radio" value="2" />
32            João da Silva <br/>
33            <input name="Candidato" type="radio" value="3" />
34            Zé da Padaria <br/>
35          </fieldset>
36          <input type="submit" value="votar" />
37        </form>
38        <%
39          }else{
40            //computar voto dos candidatos
41            addVoto(Integer.parseInt(voto));
42            %>
43            <h1>Resultado Parcial da Pesquisa </h1>
44            <h2>Antônio das Coxinhas <%= cand1 %> votos</h2>
45            <h2>João da Silva <%= cand2 %> votos</h2>
46            <h2>Zé da Padaria <%= cand3 %> votos</h2>
47            <a href='votacao.jsp'> Voltar a página da pesquisa </a>
48            <%
49          }
50        %>
51      </body>
```