Java Server Pages Technology (JSP)

Introducción

Definición

JSP significa Java Server Pages

Es la propuesta más amigable de la empresa Sun Microsystems para programación Web

Es un lenguaje sencillo y altamente productivo

Se complementa con el uso de clases de negocios y acceso a datos

Puede utilizarse como la capa de presentación de una aplicación Web

Los archivos .jsp son archivos de texto, donde puede haber bloques de HTML junto con bloques de JSP

Relación con Servlets

Los servlets fueron la primer propuesta de Sun Microsystems para programación Web

Son más fáciles de programar que los CGIs, pero aun así no resultan altamente productivos

JSP presenta una forma de trabajo (o estructura) más organizada que los Servlets, aumentando así la productividad

Por cada archivo JSP, existe un Servlet que lo respalda

El desarrollador construye el JSP solamente

Traducción y compilación

Los archivos JSP se convierten en clases

Dichas clases representan a los Servlets

El Servlet Container se encarga de realizar la conversión del JSP a una clase, y su posterior compilación

Si el JSP “no estuviera compilado” inicialmente, el servlet-container se encarga de hacerlo, generando el Servlet correspondiente de forma automática

El código agregado dentro del JSP, termina como parte de un método del Servlet

JSP Scripts

Scriptlets

Los bloques de código encerrados por <% %> se conocen como scriptlets

El código JSP se construye a través de scriptlets

Las páginas que contiene este tipo de código son denominadas “páginas dinámicas”, ya que su contenido tiende a cambiar cada vez que se la llama

Por ejemplo:

```
<HTML>
<BODY>
Hola mundo!La fecha y hora es <%= new java.util.Date() %>
</BODY>
</HTML>
```

Declaraciones

Es posible escribir los métodos que formarán parte explícitamente del Servlet que se construirá a partir del archivo JSP

Ocurre lo mismo con los atributos

Dichos métodos pueden verse también como ”funciones” utilizables dentro del archivo JSP

Para construir dichos métodos y atributos se utiliza el pseudo-tag <%!

Por ejemplo:

```
<%@ page import="java.util.*" %>
<HTML>
<BODY>
<%!
    // Atributo
    Date fechaActual = new Date();
```

```
    // Método
    Date obtenerFechaActual()
    {
       return fechaActual;
    }
%>
La fecha actual es <%= obtenerFechaActual() %>
</BODY>
</HTML>
```

Comentarios

Para realizar un comentario en JSP existen diversas formas

Las formas más tradicionales son las propuestas por el lenguaje de programación

Para comentarios en línea se utiliza el //, por ejemplo:

```
// Soy un comentario
```

Para comentarios de bloque que incluyen únicamente código JSP, pero no HTML, se utiliza el /* */, por ejemplo:

```
/*
out.println("<B>Hola</B>");
out.println("<B>Saludos!</B>");
*/
```

Para comentar una sección de código que incluye HTML, se deberá utilizar el <%-- --%>, por ejemplo:

```
<%--
<html>Titulo</html>
```

--%>

Pseudo-Tags

El pseudo-tag <% ..... %> se utiliza para contener código JSP

El pseudo-tag <%@ .... %> se utiliza para inclusión de archivos, definición de atributos e inclusión de librerías, por ejemplo:

<%@page contentType="text/html"%>

<%@page pageEncoding="UTF-8"%>

El pseudo-tag <%= .... %> se utiliza para impresión en pantalla de un valor, es igual a hacer un out.println(), por ejemplo:

<%=”Hola, mundo!”%>

El pseudo-tag <%! .... %> se utiliza para declaración de atributos y métodos propios del archivo JSP, es decir propios al servlet, por ejemplo:

```
<%!
    Date obtenerFechaActual()
    {
       return new Date();
    }
%>
```

El pseudo-tag <%-- ... --%> se utiliza para realizar comentarios

XML Pseudo-Tags

Es un mecanismo alternativo/complementario a los scriptlets

Proporcionan simplicidad en el código y mayor organización

En su concepción, son parecidos a los tags de HTML

No utilizan los caracteres el modo scriptlet con <%, sino el modo tag de la forma <algo:tag>

Existen tags que tienen cuerpo y otros que no

Los tags que tiene cuerpo tienen la siguiente forma:

<algo:tag atrib1=val1 atribN=valN>

Aqui el cuerpo

</algo:tag>

Los tags que no tienen cuerpo tienen la siguiente forma:

<algo:tag atrib1=val1 atribN=valN />

Existen dos tipos de tags: los predefinidos y los definidos por el usuario

Los predefinidos están listos para ser utilizados, y comienzan con <jsp:

Algunos de los más utilizados son: <jsp:include page=X> o <jsp:forward page=X>

Directivas de JSP

Definición

La directiva es código en forma de tags que afecta a la estructura general del servlet a construir

Es un mecanismo que permite agregar código que formará parte del servlet durante la traducción del JSP

En general es código que impacta directamente en el funcionamiento, con lo cual se requiere poner al principio

Se evalúan al realizar la traducción del JSP al servlet

Las directivas de JSP comienzan con <%@

Tienen la siguiente forma de uso:

<%@ directiva attributo="valor" %>

Las directivas que existen son page, include y taglib

Directiva page

Entre sus atributos más conocidos se encuentran: import, pageEncoding, isErrorPage, errorPage, contentType, buffer, autoFlush, isThreadSafe, session, etc.

La más comúnmente utilizada es la “import”

La forma de realizar una importación es la siguiente:

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

Directiva include

Utilizada para incluir físicamente el contenido de un archivo dentro de otro

El archivo incluido puede ser cualquier archivo: HTML, JSP u otros.

Por ejemplo:

```
<HTML>
<BODY>
A continuación se incluye el archive hola.jsp<BR>
<%@ include file="hola.jsp" %>
</BODY>
</HTML>
```

Implicit Objects

Qué son los objetos implícitos

Son objetos instanciados por el web container para su uso posterior

Son manipulados por el usuario

No son instanciados por el usuario

También llamadas variables implícitas

El objeto out

Es un objeto que representa la salida del flujo de datos

Es una instancia de javax.servlet.jsp.JspWriter

El objeto response

Es un objeto utilizado para enviar respuestas al cliente

Representa la respuesta del servidor al cliente

Es una instancia de la clase HttpServletResponse

Los métodos más utilizados son: sendRedirect(), encodeUrl(), addCookie()

El objeto request

Es un objeto utilizado para procesar información enviada por el cliente

Es una instancia de javax.servlet.http.HttpServletRequest

Se puede obtener los datos enviados por querystring, como también los datos enviados por un formulario

Se puede obtener también datos del cliente como nombre (si lo tuviera) y la IP

El objeto session

Se utiliza para almacenar los datos de una sesión

Representa una instancia de HttpSession

Está asociado con la sesión de un usuario

Puede almacenar valores para ser compartidos dentro de la sesión de un usuario

Guarda información en pares nombre-valor

Los valores se guardan como atributos con el método setAttribute()

Para obtener dichos valores y utiliza el método getAttribute() y para removerlos el método removeAttribute()

El objeto application

Representa a la aplicación Web como un objeto

Hay un único objeto application por aplicación

Representa a una instancia del ServletContext

Puede almacenar valores para ser compartidos por la aplicación

Guarda información en pares nombre-valor

Los valores se guardan como atributos con el método setAttribute()

Para obtener dichos valores y utiliza el método getAttribute() y para removerlos el método removeAttribute()