Java Enterprise Edition

JavaServer Pages (JSP)

Copyright

JavaServer Pages

- Las JavaServer Pages son un tipo mas de componente definido por la plataforma Java EE.
- Se ubican en la capa web (contenedor web).
- Surgieron para implementar contenido dinámico en la web, pero al contrario que los Java Servlets, están enfocadas a la presentación.
- La recomendación es:
  - No añadir lógica de presentación en un Java Servlet.
  - No añadir lógica de negocio a una JavaServer Page.

JavaServer Pages (cont.)

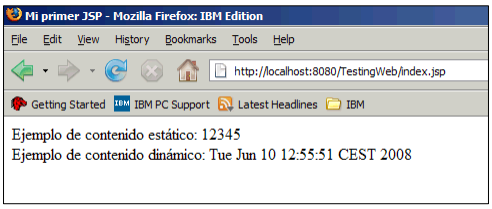
- Una página JSP es un fichero de extensión *.jsp que contiene una mezcla de:
  - Contenido estático dependiente del tipo de cliente: HTML, cHTML, SVG, WLM,…
  - Código dinámico: Java.
- Al igual que los Java Servlets, forma parte de un módulo web y pueden ser invocados directamente desde un cliente o desde un Java Servlet.
- El contenedor web, genera un Java Servlet a partir de la página JSP la primera vez que esta se solicita, y este es el que realmente se ejecuta.

Ejemplo

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1" %>
<%@ page import="java.util.Date" %>
<HTML>
 <HEAD>
  <TITLE>Mi primer JSP</TITLE>
 </HEAD>
 <BODY>

   Ejemplo de contenido estático: 12345<BR>
   Ejemplo de contenido dinámico: <%= new Date() %>

 </BODY>
</HTML>
```



Directivas de página

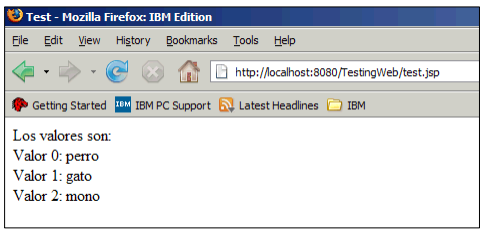
- Se utilizan para describir aspectos técnicos de la página JSP.
- Su sintaxis es la siguiente:
  - <%@ directiva [atributo=valor]… %>
- Posibles directivas:
  - page: añadir información técnica de la página.
  - include: incluir otras páginas en esta página.
  - taglib: definir una librería de tags que se van a utilizar.
- Ejemplos:
  - <%@ page contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1" %>
  - <%@ page import="java.util.Date, edu.upco.*" %>

Scriplets y expresiones

- Un scriplet es un bloque de código Java embebido en una página JSP.
- Su sintaxis es la siguiente:
  - <% código Java %>
- Una expresión, es una línea de código Java que se evalúa y traduce en un String en ejecución.
- Su sintaxis es la siguiente:
  - <%= expresión Java %>
  - Nota: la expresión Java no se finaliza con el punto y coma.
- La sintaxis de los comentarios es la siguiente:
  - <%-- comentario --%>

Ejemplo

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1" %>
<%@ page import="java.util.*" %>
<HTML>
 <HEAD>
  <TITLE>Mi primer JSP</TITLE>
 </HEAD>
 <BODY>
  <%
   List array = new ArrayList();
   array.add("perro");
   array.add("gato");
   array.add("mono");
  %>
  Los valores son:<BR>
  <%
   for(int i=0; i<array.size(); i++)
   {
  %>
    Valor <%= i %>: <%= array.get(i) %>.<BR>
  <%
   }
  %>
 </BODY>
</HTML>
```



Ciclo de vida de un JSP

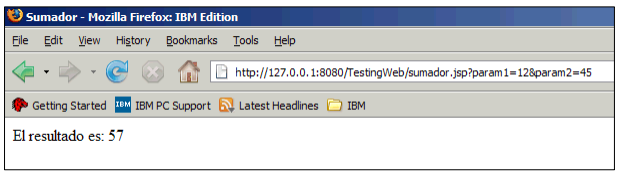
- En ejecución, el contenedor web genera un Java Servlet a partir del JSP.
- Dicho servlet implementa el interface javax.servlet.jsp.HttpJspPage que a su vez termina heredando de javax.servet.Servlet.
- El contendor web gestiona el ciclo de vida de un JSP:
  - **public void** jspInit();
  - **public void** _jspService();
  - **public void** jspDestroy();
  - Todos ellos son llamados por sus homónimos del Servlet generado. Pero en rarísimas ocasiones se van a usar.

Acceso a información

- El contenedor web, pone a disposición de las JSP los siguientes objetos implícitamente:
  - request: javax.servlet.http.HttpServletRequest
  - response: javax.servlet.http.HttpServletResponse
  - session: javax.servlet.http.HttpSession
  - out: javax.servlet.jsp.JspWriter
- Son accesibles directamente desde cualquier scriptlet o expresión.
  - Normalmente no se suele utilizar out ya que cualquier texto de la página automáticamente se contesta al cliente.

Ejemplo

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1" %>
<HTML>
 <HEAD>
  <TITLE>Sumador</TITLE>
 </HEAD>
 <BODY>
  <%
    int param1 = Integer.parseInt(request.getParameter("param1"));
    int param2 = Integer.parseInt(request.getParameter("param2"));
    int result = param1 + param2;
  %>
  El resultado es: <%= result %>
 </BODY>
</HTML>
```



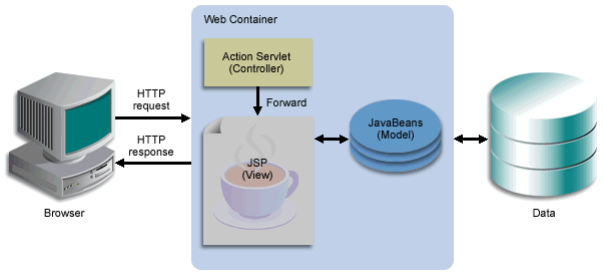
URL http://127.0.0.1:8080/TestingWeb/sumador.jsp?param1=12&param2=45

Java Servlets y JSPs

- Son dos tipos de componentes web que se complementan.
- Los Java Servlets:
  - Reciben y analizan las peticiones.
  - Deciden quien las sabe resolver y lo invocan.
  - Deciden quien se encarga de presentar el resultado.
- Las JavaServer Pages:
  - Se encargan de la presentación.
- Aunque ambos tipos de componente pueden implementar todo, no es un buen diseño. Sería difícil de mantener, y no se podría dividir el equipo en skills de desarrollo y diseño visual.

Java Servlets y JSPs

- Esquema de la ejecución:



- La lógica de negocio se implementa como:
  - JavaBeans (POJOs).
  - Enterprise JavaBeans (EJBs).
  - EIS (Enterprise Information Systems).

Java Servlets y JSPs (cont.)

- Para invocar una página JSP desde un servlet utilizamos:
  - javax.servlet.RequestDispatcher
- Se accede a través de la *request*:
  - request.getRequestDispatcher("recurso");
- Tiene dos métodos, y ambos dos reciben la *request* y la *response* como parámetros.
  - **void** forward(request, response);
    Redirige la ejecución al nuevo recurso para que responda.
  - **void** include(request, response);
    Incluye la respuesta de ese nuevo recurso como parte de su propia respuesta.

Java Servlets y JSPs (cont.)

- Para pasar información adicional del servlet al JSP hay varias alternativas:
  - Como atributo de la *request*:
    request.setAttibute(String name, Object value);
  - Como atributo de la Sesión HTTP:
    request.getSession().setAttibute(String name, Object value);
- En la JSP se reciben en los objetos existentes de manera implícita:
  - request:
    Object getAttribute(String name);
  - session:
    Object getAttribute(String name);

Ejemplo

```
package edu.upco.einf;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class Sumador extends javax.servlet.http.HttpServlet
{
  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
  ServletException, IOException
  {
    int param1 = Integer.parseInt(request.getParameter("param1"));
    int param2 = Integer.parseInt(request.getParameter("param2"));
    int result = param1 + param2;

    request.setAttribute("result", Integer.toString(result));
    request.getRequestDispatcher("result.jsp").forward(request, response);
  }
}
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1" %>
<HTML>
  <HEAD>
    <TITLE>Sumador</TITLE>
  </HEAD>
  <BODY>
    El resultado es: <%= request.getAttribute("result") %>
  </BODY>
</HTML>
```

Bibliografía

- Head First Servlets & JSP (2nd edition)
  Bryan Basham, Kathy Sierra y Bert Bates.
  O'Reilly.

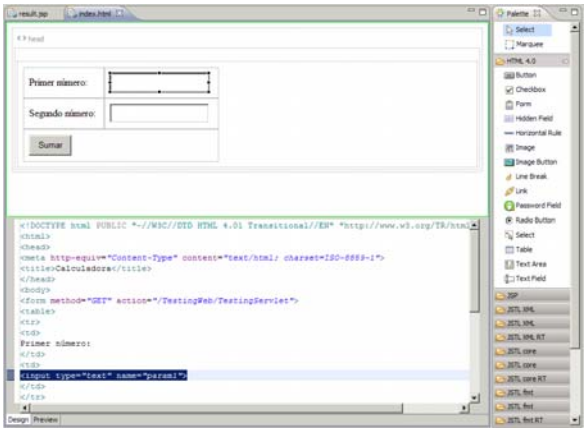
- JavaServer Pages (3rd edition)
  Hans Bergsten.
  O'Reilly.

- Beginning JavaServer Pages
  Vivek Chopra, Jon Eaves, Rupert Jones, Sing Li y John T. Bell.
  Wrox.

- The Java EE tutorial
  http://java.sun.com/javaee/5/docs/tutorial/doc/

Apéndice A: Editor Visual

- Existe un editor visual de HTML y JSP en las herramientas WTP.



- Abrir con la opción "Open with" -> "Web Page Editor".