

Desarrollo de Aplicaciones Web

JSP. Introducción.

Servlets

```
Public class miServlet extends HttpServlet {  
    public void doGet(HttpServletRequest request,  
        HttpServletResponse response)  
        throws IOException {  
  
        // Lógica de negocio, incluyendo posible  
        // acceso a DB  
  
        PrintWriter out= response.getWriter();  
  
        // Crear la vista mediante composición  
        de la página HTML  
  
        out.println(".....")  
    }  
}
```

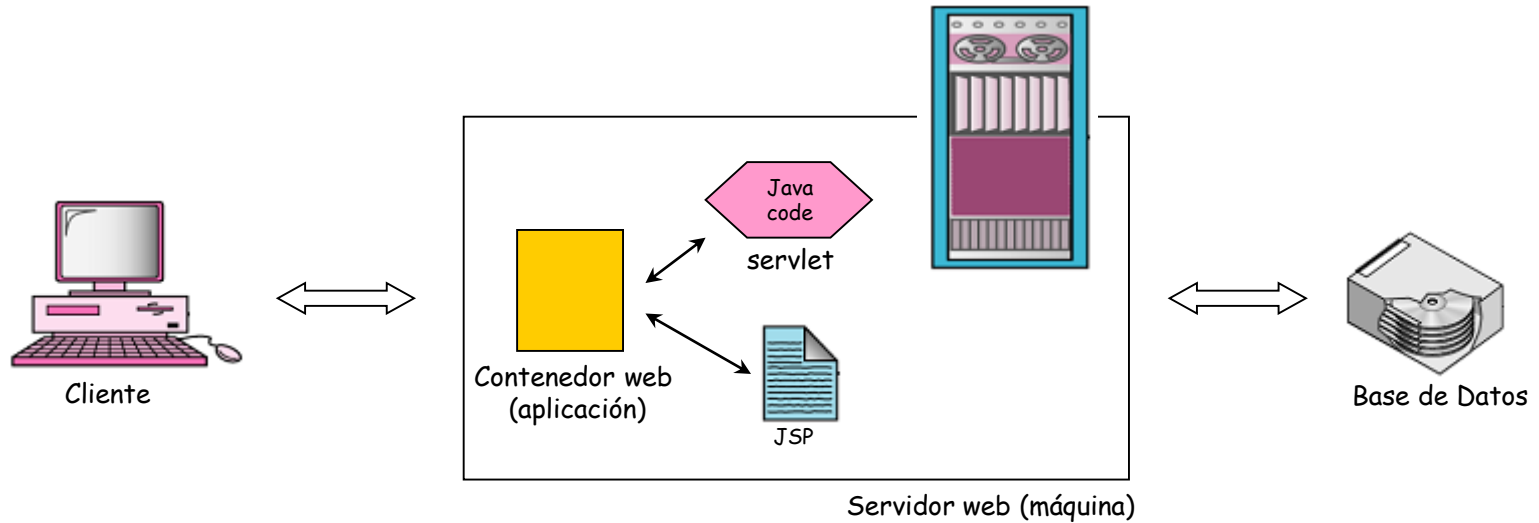
Engorroso!!!

```
Public class miServlet extends HttpServlet {  
    public void doGet(HttpServletRequest request,  
        HttpServletResponse response)  
        throws IOException {  
  
        // Lógica de negocio, incluyendo posible  
        // acceso a DB  
  
        // Invocar la vista, redirigiendo la  
        salida a una JSP  
  
    }  
}
```

**Lógica de negocio y
presentación desacoplados!!!**

Nota: **Lógica de negocio** --> parte del programa que codifica las reglas de negocio del mundo real que determinan cómo se pueden crear, almacenar y cambiar los datos.

Java Server Pages (JSP)



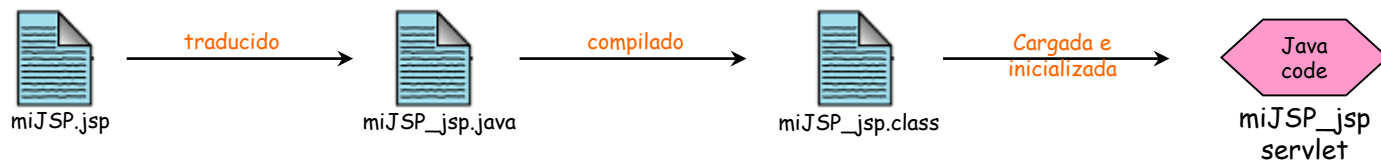
Java Server Pages (JSP)

JSP.- son páginas **HTML** que contienen líneas de código adicional que sirven para ejecutar aplicaciones que dan lugar a contenido dinámico. Dichas aplicaciones corren en el **servidor**.

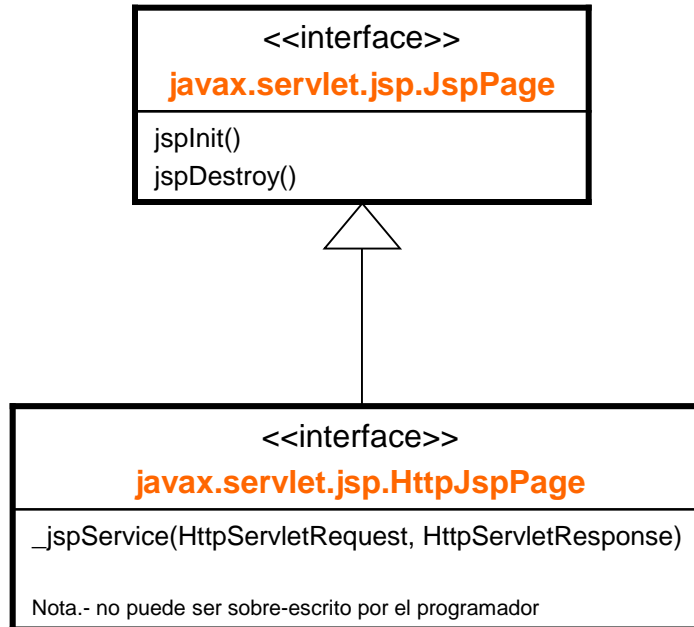
Las **JSP** permiten separar fácilmente la lógica de negocio de la presentación de resultados (**desacoplar**), encapsulando la primera → Los desarrolladores de la lógica de negocio no tienen que ser expertos en HTML y viceversa.

Las **JSP** son traducidas de forma automática a *servlets*, siendo a partir de ese momento ejecutadas como tales. En caso de modificaciones de la página, se realizará de nuevo una traducción automática.

Java Server Pages (JSP)



Java Server Pages (JSP)



Java Server Pages (JSP)

JSP

```
<html>
<body>
  <% int suma=0; %>
  El valor actual es:
  <%= ++suma %>
</body>
</html>
```

Servlet

```
public class contador_jsp extends HttpServlet {
    public void _jspService(HttpServletRequest request,
        HttpServletResponse response)
        throws java.io.IOException, ServletException {

        PrintWriter out= response.getWriter();
        response.setContentType("text/html");
        out.write("<html><body>");
        int suma= 0;
        out.write("El valor actual es:");
        out.print(++suma);
        out.write("<body><html>");
    }
}
```

TomcatHomeDir/work/Catalina/yourServerName/yourWebAppName/org/apache/jsp

Java Server Pages (JSP)

Una página **JSP** tiene una apariencia similar a una página **HTML**, donde se han introducido etiquetas encapsuladas entre `< >`.

Dentro de dichas etiquetas estarían los diferentes componentes que se pueden emplear:

- Comentarios.- `<%--.....--%>`
- Declaraciones.- `<%!.....%>`
- Expresiones.- `<%=.....%>`
- Scriptlets.- `<%.....%>`
- Directivas.- `<%@.....%>`

A mayores existen una serie de acciones, para la realización de tareas específicas

Java Server Pages (JSP)

Comentarios.- sirven para hacer más comprensible el código escrito. Su contenido no se transfiere al cliente.

Ej.-

```
<%-- Este es un comentario JSP --%>
```

Declaraciones.- utilizadas para definir variables y métodos que se utilizarán posteriormente.

Ej.-

```
<%! int contador= 0; %>
<%! int suma(int a, int b){
    _____
    _____
}
%>
```

En la traducción, aparecerán fuera del cuerpo del método `_jspService`

Java Server Pages (JSP)

Expresiones.- utilizadas para evaluar una expresión escrita en Java.

Ej.-

```
<%= clock.getYear() %>
```

```
<%= Math.random() %>
```

Scriptlets.- permiten la inclusión de cualquier código Java entre los delimitadores.

Ej.-

```
<% if (time==10) %>
```

```
<% int contador= 0; %>
```

Java Server Pages (JSP)

Directivas.- instrucciones que son procesadas por el motor JSP cuando la página es compilada en un servlet. No devuelven nada, aunque afectan a la estructura del servlet resultante de la traducción.

Ej.-

i) Definiciones a nivel de página:

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

ii) Inserciones de datos desde ficheros externos, permitiendo dividir la página en trozos más manejables

```
<%@ include file== "pbi.html"%>
```

iii) Utilización de elementos definidos en librerías estándar o propias:

```
<%@ taglib uri= http://www.usc.es/pbi/jsp_lib prefix="fun"%>
```

```
.....  
<fun:acción_1 .....>
```

Java Server Pages (JSP)

Directiva page.- define atributos que se aplican a toda la página JSP. Típicamente se localizan al comienzo de la página.

`<%@ page lista de atributos %>`

donde lista de atributos incluye:

- language= “*scriptingLanguage*” → java por defecto
- extends= “*className*” → paquete de la clase extendida
- import= “*importList*” → permite especificar lista de paquetes java importados
- session= “*true/false*” → permite o no sesiones
- buffer= “*none/default/size kb*” → tamaño del buffer de out.
- autoFlush= “*true/false*” → si true buffer salida debe descargarse automáticamente.
- isThreadSafe= “*true/false*” → si true, contenedor puede despachar varias hebras simultáneamente
- info= “*info_text*” → texto de información de lo que hace la página
- errorPage= “*error_url*” → url de la página a la que se accederá en caso de error
- isErrorPage= “*true/false*” → si true página de error tiene acceso al objeto Exception
- contentType= “*ctinfo*” → indica tipo MIME utilizado en página de respuesta (text/html por defecto)
- pageEncoding= “*peinfo*” → indica tipo de codificación de página de respuesta, por defecto
- isELIgnored= “*true/false*” → si true, expresiones de EL ignoradas

Java Server Pages (JSP)

Acciones:

Etiqueta JSP	Description
jsp:forward	Reenvía los objetos request y response a otro recurso (servlet, JSP,...).
jsp:include	Incluye otro recurso en la página. Análogo directiva include.
jsp:useBean	Localiza e instancia un objeto <i>bean</i> de Java
jsp:setProperty	Hace set sobre un atributo de un objeto <i>bean</i> de Java
jsp:getProperty	Hace get sobre un atributo de un objeto <i>bean</i> de Java
jsp:plugin	Incrusta otros componentes como un applet.
jsp:param	Da un valor a un parámetro pasado cuando se usa el jsp:forward.
jsp:fallback	Usado en jsp:plugin para imprimir un mensaje cuando el plugin está funcionando.

Nota. - La etiqueta *jsp:include* se puede utilizar para incluir tanto páginas con contenido estático como dinámico.

Java Server Pages (JSP)

Acciones:

Ej.-

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

```
<jsp:forward page="miOtraJSP.jsp" >  
    <jsp:param name="name" value="miOtraHTML.html" />  
</jsp:forward>
```

Java Server Pages (JSP)

Objetos implícitos.- Conjunto de objetos declarados e inicializados que pueden ser utilizados por el contenedor en el método `_jspService()` y que serán utilizados para su ejecución.

El acceso a los objetos implícitos se hace a través del objeto implícito *pageContext* de la clase *PageContext*. Se obtiene, a través del método *getPageContext()* de la clase *JspFactory*.

pageContext encapsula el resto de objetos implícitos, que serán accedidos a través de los métodos correspondientes.

Java Server Pages (JSP)

Objetos implícitos.- Conjunto de objetos declarados e inicializados que pueden ser utilizados por el contenedor en el método `_jspService()` y que serán utilizados para su ejecución.

Objeto	Clase	Método	Uso
out	JspWriter	getOut()	Almacenamiento de la respuesta que será enviada al cliente que solicito la página
request	HttpServletRequest	getRequest()	Encapsulamiento de los datos enviados por el cliente al servidor
response	HttpServletResponse	getResponse()	Encapsulamiento de los datos enviados por el servidor al cliente
session	HttpSession	getSession()	Identificación de usuarios que visitan las páginas (seguimiento de sesiones)
application	ServletContext	getServletContext()	Definición del conjunto de métodos que todo <i>servlet</i> utiliza para su comunicación con el contenedor
config	ServletConfig	getServletConfig()	Pasar información al <i>servlet</i> durante la inicialización
exception	JspException	getException()	Contener información acerca de cualquier excepción lanzada desde la página JSP
page	Object	getPage()	Representación del <i>servlet</i> obtenido de la página JSP

Java Server Pages (JSP)

HTML { `<form action="miJSP.jsp">`
 `<label>username: <input type="text" name="nombre"></label>`

 `</form>`

miHTML.html

JSP { `<p>El nombre del usuario es:`
 `<%= out.println(request.getParameter("nombre")) %>`
 `</p>`

miJSP.jsp

OJO!! Lógica de negocio y presentación no desacoplados!!!
JSP lo hace todo....

Java Server Pages (JSP)

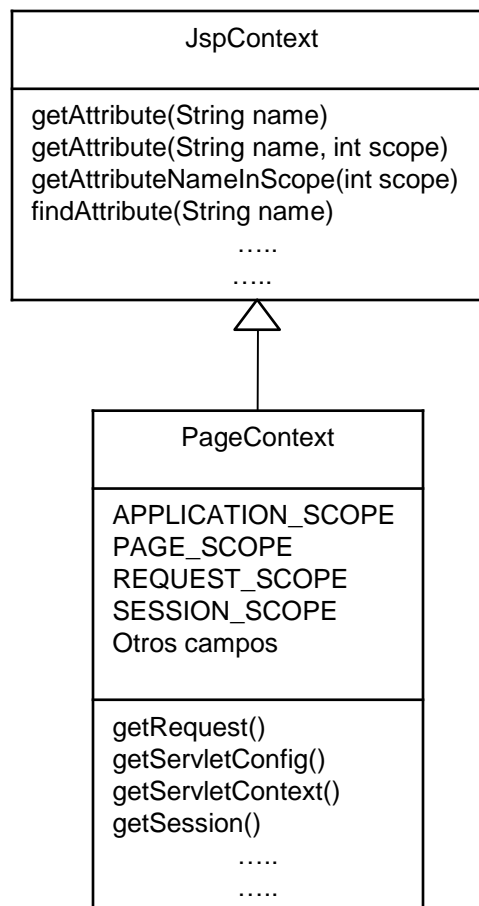
Atributos.- Su papel es análogo al visto en los *servlets*. En JSP se definirá un nuevo ámbito de uso (*page*), no presente en los *servlets*.

Se manipulan mediante los métodos correspondientes: `getAttribute()` y `setAttribute()` del objeto correspondiente:

Ámbito	Servlet	JSP (mediante objetos implícitos)
Application	<code>getServletContext.getAttribute()</code>	<code>Application.getAttribute()</code>
Request	<code>request.getAttribute()</code>	<code>request.getAttribute()</code>
Session	<code>Request.getSession().getAttribute()</code>	<code>session.getAttribute()</code>
Page	No existe	<code>pageContext.getAttribute()</code>

Java Server Pages (JSP)

Otra forma alternativa de manipulación de los objetos implícitos es mediante el uso de los métodos correspondientes del objeto implícito *pageContext*.



Java Server Pages (JSP)

Otra forma alternativa de manipulación de los objetos implícitos es mediante el uso de los métodos correspondientes del objeto implícito *pageContext*.

Ej.-

```
<%= session.getAttribute("email") %>
```

```
<%= pageContext.getAttribute("email", PageContext.SESSION_SCOPE) %>
```

Java Server Pages (JSP)

Encapsulamiento/Envío/Desencapsulamiento de la Información

Encapsulamiento → Mediante la creación de atributos en el servlet

Envío → Mediante la creación de objetos de la clase *RequestDispatcher*

Desencapsulamiento → Mediante expresiones en la página jsp del tipo:

`<%= request.getAttribute("nombre") %>`

Java Server Pages (JSP)

HTML { `<form action="miServlet">`
 `<label>username: <input type="text" name="userName"></label>`

 `</form>`

miHTML.html

servlet { `String name= request.getParameter("userName");`
 `request.setAttribute("nombre", name);`
 `RequestDispatcher view= request.getRequestDispatcher("miVista.jsp");`
 `view.forward(request, response);`

miServlet.java

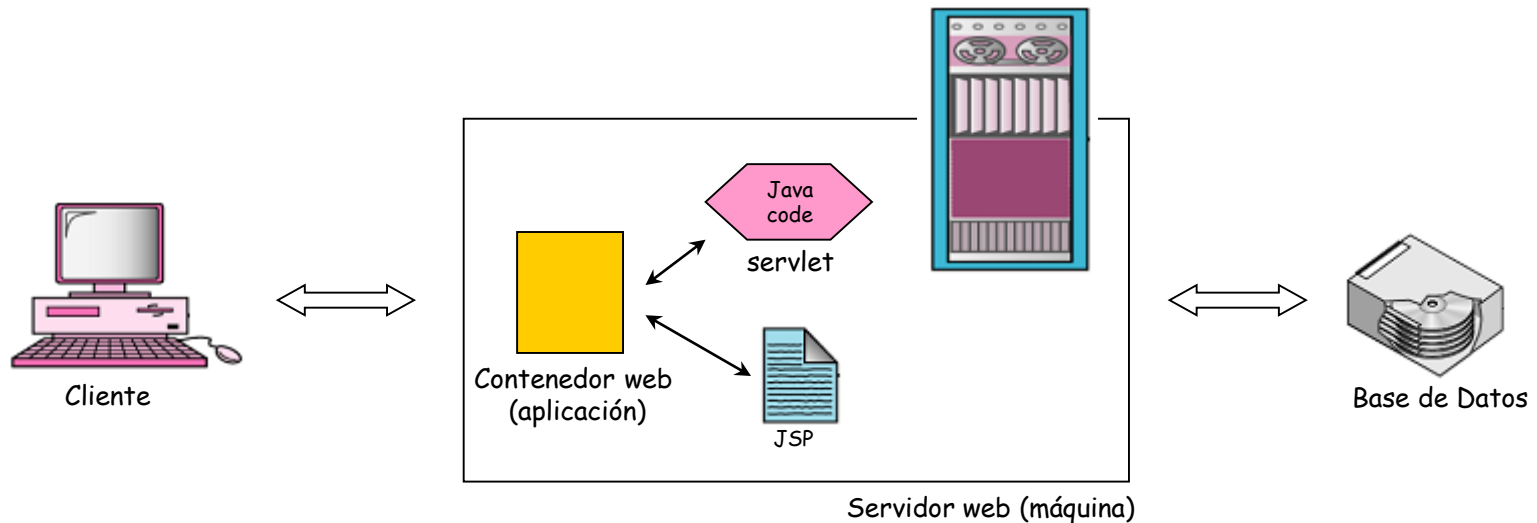
JSP { `<p>El nombre del usuario es:`
 `<%= request.getAttribute("nombre") %>`
 `</p>`

miJSP.jsp

Lógica de negocio y presentación están desacoplados!!!

Pablo García Tahoces

Java Server Pages (JSP)



```
request.setAttribute("nombre", name);
```

```
RequestDispatcher vista= request.getRequestDispatcher("miVista.jsp");  
view.forward(request, response);
```

```
<%= request.getAttribute("nombre")%>
```