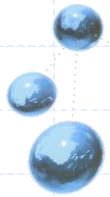
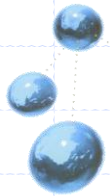
JSP

Dejando los servlets un poco de lado



Introducción General

1. Introducción
2. Elementos JSP
   a) Scripts
   b) Directivas
   c) Acciones



¿Qué es y para qué sirve?

- JSP es una tecnología que permite combinar código HTML estático con código generado dinámicamente en un mismo archivo.

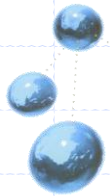
- Ventajas:
  - ✓ Separación de datos estáticos/dinámicos.
  - ✓ Sencillez (sabiendo servlets)

- Las JSP nos permiten separar la parte dinámica de nuestras páginas Web del HTML estático.

- Simplemente escribimos el HTML regular de la forma normal y encerramos el código de las partes dinámicas en unas etiquetas especiales, la mayoría de las cuales empiezan con "<%" y terminan con "%>".

Las Java Server Pages

- Las JSP se almacenan en archivos de extensión `.jsp`.
- Un JSP se traduce en un Servlet que se encarga de responder las peticiones asociadas a la página JSP.
- Una JSP tiene 3 tipos de elementos:
  - ✓ Elementos script (scriptlets)
  - ✓ Directivas
  - ✓ Acciones



Elementos de Script

- Los Scrips JSP permiten insertar código java en el servlet resultante de la compilación de la pagina JSP.

- Hay tres formas de scripts:
  1. Expresiones que se evalúan y se insertan en la salida de la web resultante:

     `<%= <JavaExpresion> %>`.

  2. Scriptlets: Es código java que se introduciría en el método `service()` del servlet:

     `<% <JavaStatement> %>`.

  3. Declaraciones: Es la manera de declarar variables de instancia para el servlet:

     `<%! <JavaMemberDeclaration> %>`.

Directivas JSP

- Una Directiva JSP afecta a la estructura general de la clase servlet.

- Sintaxis General:

```
<%@ <DirectiveType> attribute="value" %>

<%@ <DirectiveType>     attribute1="value1"
                        attribute2="value2"
                        attribute3="value3"
                        ...
                        attributeN="valueN" %>
```

- Dos tipos de directivas (`<DirectiveType>`):
  - ✓ Directiva `page`.
  - ✓ Directiva `include`.

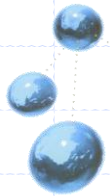
Acciones JSP

- Las Acciones JSP usan construcciones de sintaxis XML para controlar el comportamiento del motor de Servlets.
  - ✓ Permiten insertar un fichero dinámicamente,
  - ✓ reutilizar componentes JavaBeans,
  - ✓ reenviar al usuario a otra página, o
  - ✓ generar HTML para el plug-in Java.

- Las acciones disponibles incluyen:
  - ✓ Include. `<jsp:include …>`.
  - ✓ UseBean. `<jsp:useBean …>`.
  - ✓ setProperty . `<jsp:setProperty …>`.
  - ✓ getProperty . `<jsp:getProperty …>`.
  - ✓ Forward . `<jsp:forward …>`.
  - ✓ Plugin . `<jsp:plugin …>`.

Scripts JSP

1. Expresiones
2. Scriptlets
3. Declaraciones



Expresiones (1/2)

- Sintaxis:

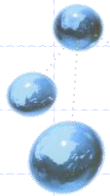
```
<%= <JavaExpression> %>
```

- La `<expresion>` es evaluada, insertada en el método `service()`, y enviada al response a través del objeto `out`.

- El equivalente XML es:

```
<jsp:expression>
  <JavaExpression>
</jsp:expression>
```

- Las variables predefinidas en este contexto son `request`, `response`, `out`, `session`, `application`, `config`, y `pageContext`.

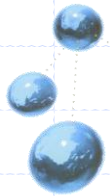


Expresiones (2/2)

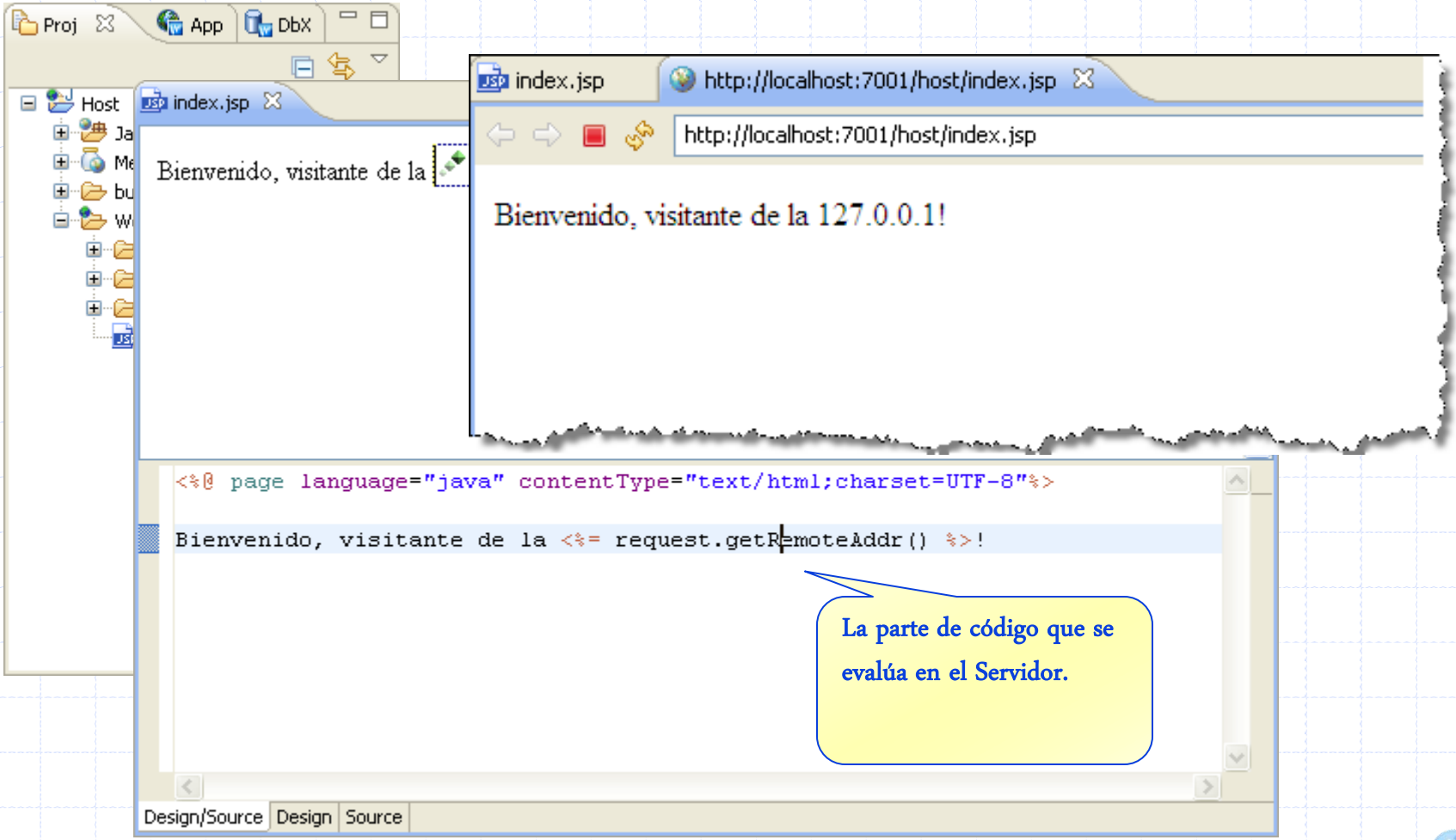
- Se tiene acceso a variables:
  - ✓ `request`, el `HttpServletRequest`
  - ✓ `response`, el `HttpServletResponse`
  - ✓ `session`, el `HttpSession` asociado con el `request` (si existe)
  - ✓ `out`, el `PrintWriter` (una versión con buffer del tipo `JspWriter`) usada para enviar la salida al cliente.

- Ejemplo:

```
Your hostname: <%= request.getRemoteHost()
  %>
```



Un ejemplo simple



Scriptlets JSP

- Sintaxis:

```
<% <JavaStatement> %>
```

- El código se inserta en el método `service()`.
- El equivalente XML es:

```
<jsp:scriptlet>
   <JavaStatement>
</jsp:scriptlet>
```

Sintaxis

- Características:
  - ✓ Se insertan dentro del método `service()` del servlet.
  - ✓ Tienen acceso a las mismas variables que las expresiones.

- El código dentro de un scriptlet se insertará exactamente como está escrito, y cualquier HTML estático anterior o posterior al scriptlet se convierte en sentencias `print()`.
- Si se quiere poder usar los caracteres "`%>`" dentro de un scriptlet, hay que usar "`%\>`"

- Esto significa que los scriptlets no necesitan completar las sentencias Java, y los bloques abiertos pueden afectar al HTML estático fuera de los scriplets.

Sintaxis

- La sentencia:

```
<% if (Math.random() < 0.5) { %>
    Have a <B>nice</B> day!
<% } else { %>
    Have a <B>lousy</B> day!
<% } %>
```

- Se traducirá en:

```
if (Math.random() < 0.5) {
  out.println("Have a <B>nice</B>
  day!");}
else {
  out.println("Have a <B>lousy</B>
  day!");}
```

Sintaxis y uso

- Sintaxis: `<%! <JavaStatement> %>`

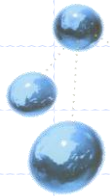
- Conversión:
  - ✓ Se insertan en el cuerpo de la clase del servlet,
  - ✓ fuera de cualquier método existente,
  - ✓ como la declaración de un miembro Java.

- Permite insertar métodos, variables...
- No generan salida alguna. Se usan combinadas con scriptlets.

```
<%! private int accessCount = 0; %>
Accesses to page since server reboot:
<%= ++accessCount %>
```

Variables Predefinidas

Variables JSP

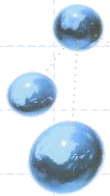


La petición: request

- Nombre de la variable: `request` .
- Este es el `HttpServletRequest` asociado con la petición, y nos permite:
  - ✓ Mirar los parámetros de la petición (mediante `getParameter()`),
  - ✓ El tipo de petición (`GET`, `POST`, `HEAD`, etc.),
  - ✓ Y las cabeceras HTTP entrantes (`cookies`, etc.).
- Estrictamente hablando, se permite que la petición sea una subclase de `ServletRequest` distinta de `HttpServletRequest`, si el protocolo de la petición es distinto de HTTP. Esto casi nunca se lleva a la práctica.

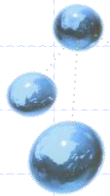
La respuesta: response

- Nombre: `response`.
- Este es el `HttpServletResponse` asociado con la respuesta al cliente.



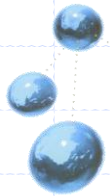
La salida: out

- Nombre: `out`.
- Este es el `PrintWriter` usado para enviar la salida al cliente.
- Sin embargo, para poder hacer útil el objeto response esta es una versión con buffer de `PrintWriter` llamada `JspWriter`.
- Podemos ajustar el tamaño del buffer, o incluso desactivar el buffer, usando el atributo buffer de la directiva page.
- Se usa casi exclusivamente en scriptlets ya que las expresiones JSP obtienen un lugar en el stream de salida, y por eso raramente se refieren explícitamente a `out`.



La sesión: session

- Nombre: `session`.
- Este es el objeto `HttpSession` asociado con la petición.
- Las sesiones se crean automáticamente, por esto esta variable se une incluso si no hubiera una sesión de referencia entrante.
- La única excepción es usar el atributo `session` de la directiva page para desactivar las sesiones, en cuyo caso los intentos de referenciar la variable `session` causarán un error en el momento de traducir la página JSP a un servlet.

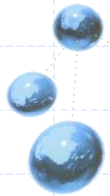


Varios: application, Config & PageContext

- Nombre: `application`.
  - ✓ El `ServletContext` obtenido mediante el método `getServletConfig().getContext()`.
- Nombre: `config`.
  - ✓ El objeto `ServletConfig`.
- Nombre: `pageContext`.
  - ✓ JSP presenta una nueva clase llamada `PageContext` para encapsular características de uso específicas del servidor como `JspWriters` de alto rendimiento.
  - ✓ La idea es que si tenemos acceso a ellas a través de esta clase en vez directamente, nuestro código seguirá funcionando en motores servlet/JSP "normales".
- Nombre: `page`.
  - ✓ Esto es sólo un sinónimo de `this`, y no es muy útil en Java.
  - ✓ Fue creado como situación para el día que los lenguajes de script puedan incluir otros lenguajes distintos de Java.

Directivas JSP <%@ ... %>

1. Directiva `page`.
2. Directiva `include`.



Resumen general

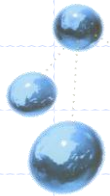
- Una Directiva JSP afecta a la estructura general de la clase servlet.

- Sintaxis General:

```
<%@ <DirectiveType> attribute="value" %>

<%@ <DirectiveType>     attribute1="value1"
                        attribute2="value2"
                        attribute3="value3"
                        ...
                        attributeN="valueN" %>
```

- Dos tipos de directivas (`<DirectiveType>`):
  - ✓ Directiva `page`.
  - ✓ Directiva `include`.

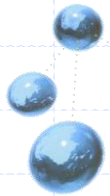


Directiva Page (1/3)

- Sintaxis:

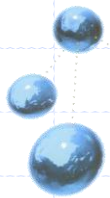
```
<%@ page att="val" %>
```

- Dirige al motor servlet sobre la configuración general.
- El equivalente XML es:
  - ✓ `<jsp:directive.page att="val"\>`.



Directiva Page: Atributos legales (2/3)

- import="package.class"
- contentType="MIME-Type"
- isThreadSafe="true|false"
- session="true|false"
- buffer="sizekb|none"
- autoflush="true|false"
- extends="package.class"
- info="message"
- errorPage="url"
- isErrorPage="true|false"
- language="java"



Directiva Include (3/3)

- Sintaxis

```
<%@ include file="url" %>
```

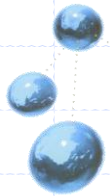
- Un archivo del sistema local se incluirá cuando la página se traduzca a un Servlet.

- El equivalente XML es:

✓ `<jsp:directive.include file="url" \>`

Directiva <%@ page ... %>

Atributos de PAGE



Atributos: import & contentType

- Import:
  - ✓ `import="package.class"` o `import="package.class1,...,package.classN".`
  - ✓ Esto permite especificar los paquetes que deberían ser importados.
  - ✓ La directiva `import` es la única que puede aparecer múltiples veces.

- Tipos de media:
  - ✓ `ContentType = "MIME-Type"` o `contentType = "MIME-Type; charset = Character-Set"`
  - ✓ Esto especifica el tipo `MIME` de la salida.
  - ✓ El valor por defecto es `text/html`. Tiene el mismo valor que el scriptlet usando "`response.setContentType`".

Atributos: Threads& Sesiones

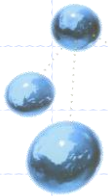
- Threads
  - ✓ `isThreadSafe="true|false"`.
  - ✓ Un valor de true (por defecto) indica un procesamiento del servlet normal, donde múltiples peticiones pueden procesarse simultáneamente con un sólo ejemplar del servlet, bajo la suposición que el autor sincroniza los recursos compartidos.
  - ✓ Un valor de false indica que el servlet debería implementar `SingleThreadModel`.

- Sesión:
  - ✓ `session="true|false"`.
  - ✓ Un valor de `true` (por defecto) indica que la variable predefinida `session` (del tipo `HttpSession`) debería unirse a la sesión existente si existe una, si no existe se debería crear una nueva sesión para unirla.
  - ✓ Un valor de false indica que no se usarán sesiones, y los intentos de acceder a la variable `session` resultarán en errores en el momento en que la página JSP sea traducida a un servlet.

Atributos: buffer & autoflush

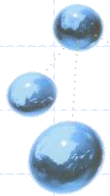
- Buffer:
  - ✓ `buffer="sizekb|none"`.
  - ✓ Esto especifica el tamaño del buffer para el `JspWriter out`.
  - ✓ El valor por defecto es específico del servidor y debería ser de al menos 8kb.

- Autoflush:
  - ✓ `autoflush="true|false"`.
  - ✓ Un valor de `true` (por defecto) indica que el buffer debería descargase cuando esté lleno.
  - ✓ Un valor de false, raramente utilizado, indica que se debe lanzar una excepción cuando el buffer se sobrecargue.
  - ✓ Un valor de false es ilegal cuando usamos `buffer="none"`.



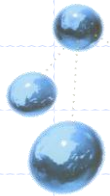
Atributos: Extends, Info& ErrorPage

- Extends:
  - ✓ `extends="package.class"`.
  - ✓ Esto indica la superclase del servlet que se va a generar.
- Info
  - ✓ `info="message"`.
  - ✓ Define un `string` que puede usarse para ser recuperado mediante el método `getServletInfo`.
- ErrorPage:
  - ✓ `errorPage="url"`.
  - ✓ Especifica una página JSP que se debería procesar si se lanzará cualquier `Throwable` pero no fuera capturado en la página actual.



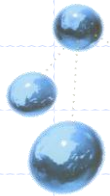
Atributos: isErrorPage & Language

- isErrorPage:
  - ✓ `isErrorPage="true|false"`.
  - ✓ Indica si la página actual actúa o no como página de error de otra página JSP.
  - ✓ El valor por defecto es `false`.
- Lenguage:
  - ✓ `language="java"`.
  - ✓ En algunos momentos, esto está pensado para especificar el lenguaje a utilizar.
  - ✓ Por ahora, no debemos preocuparnos por él ya que java es tanto el valor por defecto como la única opción legal.



Directiva Include<%@ include ... %>

Directivas JSP



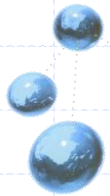
Atributos: file

- Permite incluir ficheros en el momento en que la página JSP es traducida a un servlet.

```
<%@ include file="<urlRelativa>" %>
```

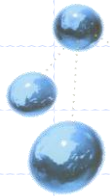
- Los contenidos del fichero incluido son analizados como texto normal JSP y así pueden incluir HTML estático, elementos de script, directivas y acciones.

- Uso típico:
  - ✓ Barras de navegación.
  - ✓ Encabezados / Pie de páginas.



Acciones

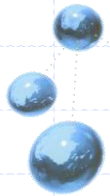
Acciones JSP



Conceptos Generales

- Usan construcciones de sintaxis XML para controlar el comportamiento del motor de Servlets.

- Podemos insertar un fichero dinámicamente, utilizar componentes JavaBeans, reenviar al usuario a otra página, o generar HTML para el plug-in Java.



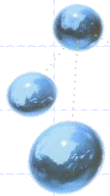
Acciones: jsp:useBean

- Ejemplo de uso:
  - ✓`<jsp:useBean att=val*/>`
  - ✓`<jsp:useBean att=val*>`
  - ✓ `...`
  - ✓`</jsp:useBean>`

- Encuentra o construye un Java Bean.
- `Ejemplo:`
  - ✓
```
<jsp:useBean class="beans.Persona"
   id="persona" scope="page"/>
```

Atributos

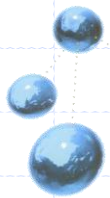
- `id`
  - ✓ Da un nombre a la variable que referencia al bean.
  - ✓ Se usará un objeto bean anterior en lugar de instanciar uno nuevo si se puede encontrar uno con el mismo `id` y `scope`.
- `class`
  - ✓ Designa el nombre completo de la clase del bean.
- `scope`
  - ✓ Indica el contexto en el que el bean debería estar disponible.
  - ✓ Hay cuatro posibles valores: `page`, `request`, `session`, y `application`.
- `type`
  - ✓ Especifica el tipo de la variable a la que se referirá el objeto.



Acciones sobre beans

- Acción `jsp:setProperty`
  - ✓ Ejemplo
    - <jsp:setProperty name="persona" property="nombre" value="Maria"/>
  - ✓ Selecciona las propiedades del bean, bien directamente o designando el valor que viene desde un parámetro de la petición.
  - ✓ Los atributos legales son:
    - `name="beanName"`
    - `property="propertyName|*"`
    - `param="parameterName"`
    - `value="val"`

- Acción `jsp:getProperty`
  - ✓ Ejemplo
    - <jsp:getProperty name="persona" property="nombre"/>
  - ✓ Recupera y saca las propiedades del Bean.



Atributos (2/2)

- param
  - ✓ Este parámetro opcional designa el parámetro de la petición del que se debería derivar la propiedad.
  - ✓ Si la petición actual no tiene dicho parámetro, no se hace nada: el sistema no pasa null al método seleccionador de la propiedad.
  - ✓ Así, podemos dejar que el bean suministre los valores por defecto, sobrescribiéndolos sólo cuando el parámetro dice que lo haga.

```
<jsp:setProperty
        name="orderBean"
        property="numberOfItems"
        param="numItems" />
```

  - ✓ Si no indicamos nada, el servidor revisa todos los parámetros de la petición e intenta encontrar alguno que concuerde con la propiedad indicada.

```
<jsp:setProperty name='user' property='firstName' />
```

  - ✓ Si no usamos el cómodin * se rellenarán todas las propiedades que coincidan con un parametro de la peticion.

```
<%!-- todos los parámetros de la petición cuyo nombre
    coincida con propiedades --%>
<jsp:setProperty name='user' property='*' />
```

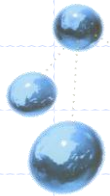
Ejemplo completo:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN""http://www.w3.org/TR/html4/loose.dtd"
>
<jsp:useBean class="beans.Persona" id="persona" scope="
page">
<html>
    <head>
        <meta http-equiv="Content-
Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
<jsp:setProperty
name="persona" property="nombre" value="Maria"/>
<%="El nombre de la persona es : "%>
<h1><jsp:getProperty
name="persona" property="nombre"/></h1>
    </body>
</html>
```

Acciones sobre beans

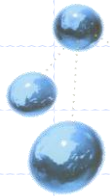
- Acción `jsp:forward`
  - ✓ Ejemplo:
    - `<jsp:forward page="relative URL"/>`
  - ✓ Reenvía la petición a otra página.

- Acción `jsp:plugin`
  - ✓ Ejemplo:
    - `<jsp:plugin attribute="value"*> ... </jsp:plugin>`
  - ✓ Genera etiquetas `OBJECT` o `EMBED`, apropiadas al tipo de navegador, pidiendo que se ejecute un applet usando el Java Plugin.



include

- `jsp:include` nos permite insertar ficheros en una página que está siendo generada.
- La sintaxis se parece a esto:
  - `<jsp:include page="relative URL" flush="true" />`
- Al contrario que la directiva `include`, que inserta el fichero en el momento de la conversión a un Servlet, ésta inserta el fichero cuando la página es solicitada.
  - ✓ Se pierde eficiencia, e imposibilita a la página incluida contener código JSP general pero se obtiene gran flexibilidad.
  - ✓ Uso: Noticias...



Ejercicio

- Realiza el mismo ejercicio que hiciste en Servlet (Agenda), pero esta vez para JSP.
- En este caso, el ejercicio plus será obligatorio