Escuela Politécnica de Ingeniería
Grado de Ingeniería Informática en Tecnologías de la Información

Tecnologías Web

Tema 3

Tecnologías web de servidor: JEE (JSPs) 2/2

Índice

- Encadenamiento de Servlets/JSPs
- Introducción
- Elementos de JSP
  - Elementos de secuencias/Scripting
  - Directivas
  - **Acciones**

JSP: Acciones

- Usan construcciones de sintaxis XML para controlar el comportamiento del motor de servlets. Podemos insertar un fichero dinámicamente, reutilizar componentes JavaBeans, reenviar al usuario a otra página, etc.
- Tipos de acciones
  - Estándar
  - A medida
  - JSTL

Nota: Los elementos XML, al contrario que los HTML, **son sensibles a las mayúsculas**

JSP: Acciones ejemplo

```
<HTML>
<HEAD>
<TITLE>Validación de usuario</TITLE>
</HEAD
<BODY>

…… Body contents ……

<sig:signature name="Enrique DLC"/>
</BODY>
</HTML>
```

JSP

HTML generado

Bienvenido a la aplicación

Enrique A. de la Cal Marin
Profesor T.U.
Universidad de Oviedo
Edificio Departamental zona Oeste, Edificio 1
Desp. 1.2.16
Departamento de Informatica
Tlfo. +34 985182520
Fax. +34 985181986

Acciones

- Elementos XML que realizan determinadas acciones
  - JSP define las siguientes (estándar):

Elemento	Descripción
`<jsp:useBean>`	Permite usar un *JavaBean* desde la página
`<jsp:getProperty>`	Obtiene el valor de una propiedad de un componente JavaBean y lo añade a la respuesta
`<jsp:setProperty>`	Establece el valor de una propiedad de un JavaBean
`<jsp:include>`	Incluye la respuesta de un servlet o una página JSP
`<jsp:forward>`	Redirige a otro servlet o página JSP
`<jsp:param>`	Envía un parámetro a la petición redirigida a otro servlet o JSP mediante <jsp:include> o <jsp:forward>
`<jsp:plugin>`	Genera el HTML necesario para ejecutar un ***applet***

JSP: Acción useBean

- Permite cargar y utilizar un JavaBean en la página JSP y así utilizar la reusabilidad de las clases Java

```
<jsp:useBean id="name" class="package.class" />
```

- Esto significa: "usa un objeto de la clase especificada por class, y únelo a una variable con el nombre especificado por id"
- Ahora podemos modificar sus propiedades mediante `<jsp:setProperty>`, o usando un scriptlet y llamando a un método del objeto referenciado por id
  - Para recoger una propiedad se usa `<jsp:getProperty>`

JSP: Acción useBean

- **`id`**
  - Da un nombre a la variable que referenciará el bean. Se usará un objeto bean anterior en lugar de instanciar uno nuevo si se puede encontrar uno con el mismo id y ámbito (scope)
- **`class`**
  - Designa el nombre cualificado completo 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
- **`beanName`**
  - Da el nombre del bean, como lo suministraríamos en el método instantiate de Beans. Está permitido suministrar un `type` y un `beanName`, y omitir el atributo `class`

JSP: Acciones: setProperty

- Para asignar valores a propiedades de los beans que se han referenciado anteriormente
- 2 usos:
  - Despues de un useBean

```
<jsp:useBean id="myName" ... />
...
<jsp:setProperty name="myName"
                 property="someProperty" ... />
```

  Se ejecuta siempre que haya una solicitud

JSP: Acciones: setProperty

- Dentro de un useBean

```
<jsp:useBean id="myName“ ... >
  ...
  <jsp:setProperty name="myName"
          property="someProperty" ... />
</jsp:useBean>
```

  Solo se ejecuta cuando haya que instanciar un bean

JSP: Acciones: setProperty

- **`name`**:
  - Este atributo requerido designa el bean cuya propiedad va a ser seleccionada. El elemento `jsp:useBean` debe aparecer antes del elemento `jsp:setProperty`.
- **`property`**:
  - Este atributo requerido indica la propiedad que queremos seleccionar
  - Hay un caso especial: un valor de "*" significa que todos los parámetros de la petición cuyos nombres correspondan con nombres de propiedades del Bean serán pasados a los métodos de selección apropiados
- **`value`**:
  - Este atributo opcional especifica el valor para la propiedad. Los valores string son convertidos automáticamente a lo que corresponda mediante el método estándard valueOf. No se pueden usar `value` y `param` juntos, pero si está permitido no usar ninguna

JSP: Acciones: setProperty

- **`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 parametros de la petición e intenta encontrar alguno que concuerde con la propiedad indicada

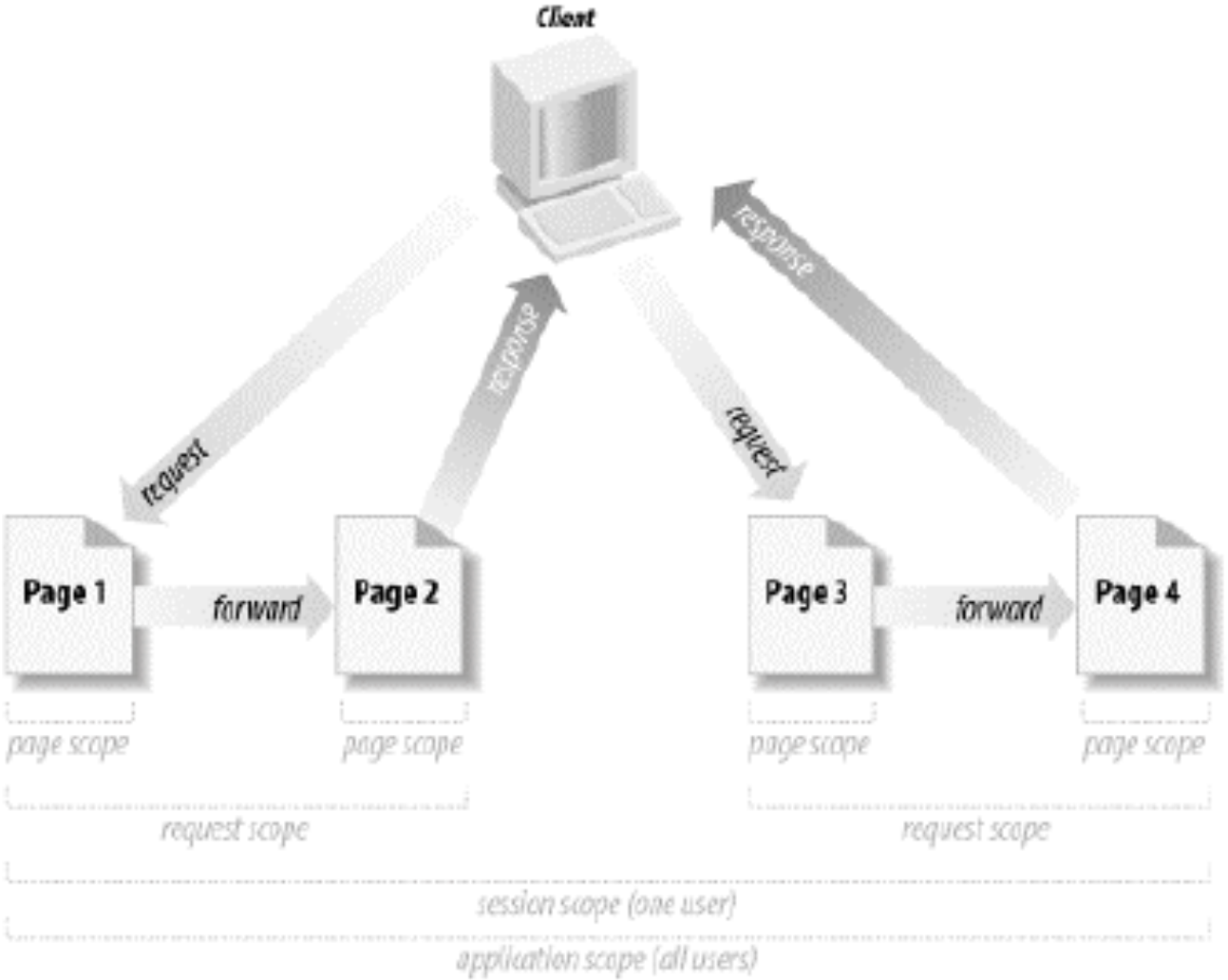
Ejemplo de uso <jsp:xxx

```
<%@ page contentType="text/html" pageEncoding="iso-8859-1"
    isELIgnored="false"%>

<jsp:useBean id="date" class="uo.dasdi.beans.DateBean" />

<HTML>
<HEAD>
<TITLE>Hola Mundo!</TITLE>
</HEAD>
<BODY>
<center>Bienvenido a mi primera página Web!</center>

<%
    if (date.getHour() > 12) {
%>
    Son mas de las 12, son ya las las <%=date.getHour() %>
<%
    }
%>
</BODY>
</HTML>
```

Ejemplo de uso <jsp:xxx código del bean

```java
package uo.dasdi.beans;

import java.util.Calendar;

public class DateBean {
    private Calendar calendar = Calendar.getInstance();

    public Date getDate(){
        return calendar.getTime();
    }

    public int getHour(){
        return calendar.get(Calendar.HOUR_OF_DAY);
    }

    public String getGreeting(){
        int hour = calendar.get(Calendar.HOUR_OF_DAY);
        String greeting;
        if (hour < 13) {
            greeting = "Buenos días";
        } else if (hour < 20) {
            greeting = "Buenas tardes";
        } else {
            greeting = "Buenas noches";
        }
        return greeting;
    }
}
```

Comunicación entre jsp
Posibles ámbitos



Acciones JSTL, ejemplo

```
<%@ page contentType="text/html" pageEncoding="iso-8859-1" isELIgnored="false"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<jsp:useBean id="date" class="uo.dasdi.beans.DateBean" />

<HTML>
<HEAD>
<TITLE>Hola Mundo!</TITLE>
</HEAD>
<BODY>
<center>Bienvenido a mi primera página Web!</center>
<c:if test="${date.hour > 12}">
    <c:out value="Son mas de las 12, son las ${date.hour}" />
</c:if>
</BODY>
</HTML>
```

Con tag-libs no hay scriplet

Desarrollo de TAG-LIBs

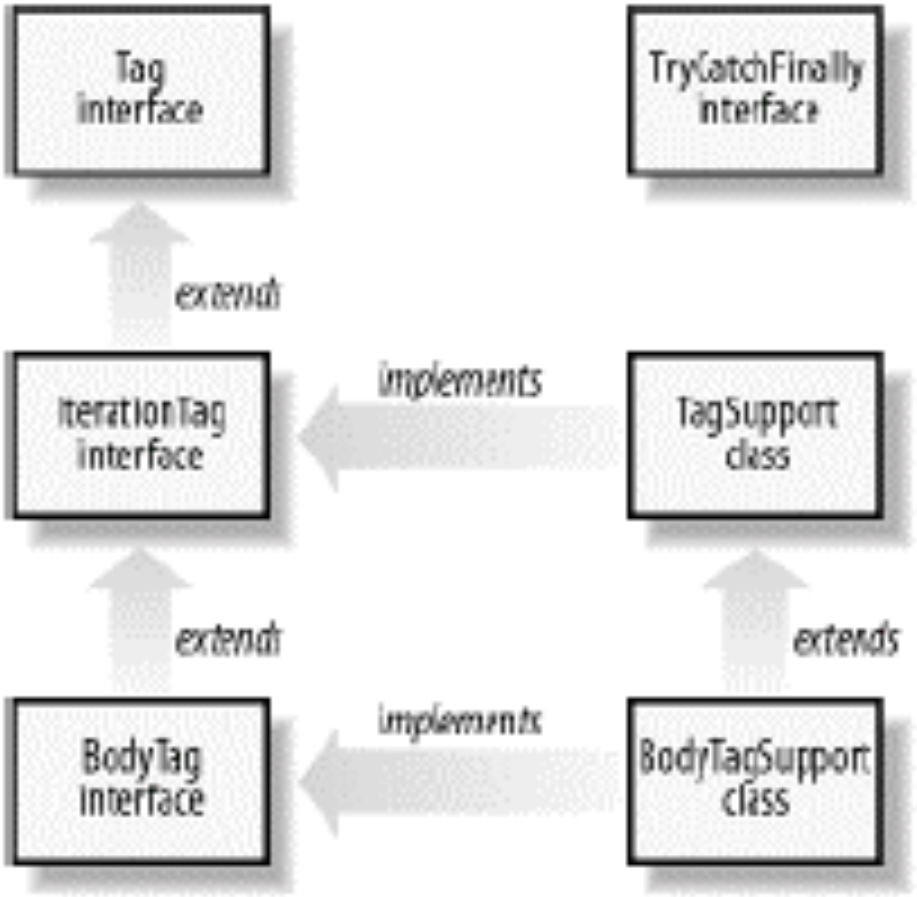
- Podemos desarrollar nuestras propias etiquetas de acción
- Para ello:
  - Desarrollamos la clase que implementa la acción
  - Describimos la etiqueta en la TLD
  - Para usarla, la damos de alta en el web.xml (puesto que no es una etiqueta estándar)

Desarrollo de TAG-LIBs

- Desarrollo de la clase (manejador del tag):
  - Los interfaces y clases necesarios están definidos en javax.servlet.jsp.tagext
  - Tres interfaces
  - Y dos clases de soporte:
    - TagSupport
    - BodyTagSupport

Tag	Define los métodos necesarios para implementar cualquier acción
IterationTag	Extiende el interface Tag y define los métodos necesarios para iterar sobre los elementos del cuerpo del tag
BodyTag	Extiende el IterationTag y define los métodos para acceder al cuerpo del tag

Desarrollo de TAG-LIBs



Ejemplo: implementación de tag signature

```
<HTML>
<HEAD>
<TITLE>Validación de usuario</TITLE>
</HEAD>
<BODY>

    . . . Body contents ...

<sig:signature name="Alberto MFA"/>
</BODY>
</HTML>
```

JSP

HTML generado

Bienvenido a la aplicación

--
Alberto MFA
@2008 Universidad de Oviedo
+034 985.10.4339 | FAX: +034 985.105.094
alb@uniovi.es

Ejemplo: implementación de tag Pasos

- Crear una nueva clase
  - extiende TagSupport
  - Añadir atributos y sus setters
  - Implementar método doEndTag()
- Crear archivo TLD
  - Copiar en WEB-INF/tlds
- Registrar TagLib en web.xml
- Declararla en JSP <% taglib … %>

Ejemplo: implementación de tag paso: Crear la clase

```
package uo.dasdi.tags;

import java.io.IOException;

public class SignatureTag extends TagSupport {

    private String name;

    public String getName() {return name;}
    public void setName(String name) {this.name = name;}

    @Override
    public int doEndTag() throws JspException {
        try {
            JspWriter out = pageContext.getOut();

            out.println("--<br>" + name);
            out.println("<br> @2008 Universidad de Oviedo");
            out.println("<br>+034 985.10.4339 | FAX: +034 985.105.094");
            out.println("<br>alb@uniovi.es");
        } catch (IOException e) {
            // Ignore it
        }
        return EVAL_PAGE;
    }
```

Ejemplo: implementación de tag paso: Crear archivo TLD

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE taglib PUBLIC
    "-//Sun Microsystems, Inc.//DTD JSP Tag Library 1.2//EN"
    "http://java.sun.com/dtd/web-jsptaglibrary_1_2.dtd">

<taglib>
    <tlib-version>1.0</tlib-version>
    <jsp-version>1.2</jsp-version>
    <short-name>sig</short-name>
    <uri>uo.dasdi.signature</uri>

    <tag>
        <name>signature</name>
        <tag-class>uo.dasdi.tags.SignatureTag</tag-class>
        <body-content>empty</body-content>
        <attribute>
            <name>name</name>
        </attribute>
    </tag>
</taglib>
```

Ejemplo: implementación de tag paso: Registrar en web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
    <display-name>Prototipo DASDI</display-name>
    <description>
        Practica de la asignatura DASDI impartida en la EUITIO.
    </description>

    <taglib>
        <taglib-uri>uo.dasdi.signature</taglib-uri>
        <taglib-location>/WEB-INF/lib/tlds/signature.tld</taglib-location>
    </taglib>

    <taglib>
        <taglib-uri>http://java.sun.com/jsp/jstl/core</taglib-uri>
        <taglib-location>/WEB-INF/lib/tlds/c-1_0.tld</taglib-location>
    </taglib>

    <!-- Pagina de entrada por defecto -->
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>

</web-app>
```

Ejemplo: implementación de tag paso: <%@ taglib … en JSP

```
<%@ page contentType="text/html" pageEncoding="iso-8859-1" %>

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

<jsp:useBean id="user" class="uo.dasdi.beans.UserBean" />
<jsp:setProperty property="*" name="user" />

<HTML>
<HEAD>
<TITLE>Validación de usuario</TITLE>
</HEAD>
<BODY>

    . . . Body contents ...

<sig:signature name="Alberto MFA"/>
</BODY>
</HTML>
```