Arquitectura e Integración de Aplicaciones Empresariales

Quinta Sesión
Beans y JSPs

Universidad Autónoma Metropolitana
Casa abierta al tiempo Azcapotzalco

Dra. Maricela Bravo
Cubículo H-287-B
mari_clau_18@hotmail.com

JavaBeans en JSP

- Los JavaBeans son objetos Java que cumplen ciertas características en cuanto a su diseño.
- Se utilizan para reducir al máximo el código Java insertado en una página JSP.
- Permite separar la lógica de ejecución (en el JavaBean) de la presentación
- Se encapsula el código Java en un objeto (JavaBean) y se instancia y usa con el JSP.
- Si se usa un JavaBean en una página habrá que definir la clase correspondiente, creando los métodos set y get para los atributos definidos.
- Dentro del servlet generado se puede llamar a métodos de un JavaBean que se encarguen de realizar ciertas operaciones y el servlet muestra el resultado de las mismas.
- Ventaja del traslado de la lógica a un JavaBean
- Separación de interfaz de la implementación

jsp: useBean

jsp: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" />
```

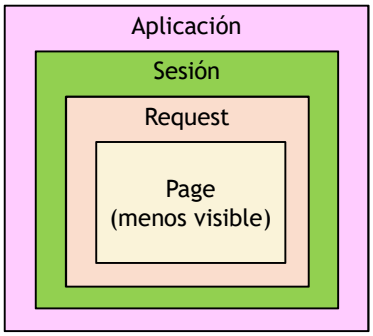
Esto normalmente significa "usa un objeto de la clase especificada por class, y se asocia a una variable con el nombre especificado por id".

Se pueden modificar sus propiedades mediante `jsp:setProperty`, o usando un scriptlet y llamando a un método de id. Para leer el valor de una propiedad se usa `jsp:getProperty`

jsp: useBean

- `id` Nombre a la variable que referenciará el bean.
- `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.
- Esta posible proporcionar un `type` y un `beanName`, y omitir el atributo `class`.

Scope del Bean



Page Scope (menos visible)

- Los Beans con scope = ***page*** son accesibles solamente en la página donde fueron creados.
- Un bean con scope ***page*** no es persistente entre requests o fuera de la página.

6

Ejemplo del Manejo de Scope

```
/* Un simple bean que cuenta visitas*/

public class Counter
{
   private int count = 1;

   public Counter() {}

   public int getCount() { return count++; }

   public void setCount(int c) { count = c; }

}
```

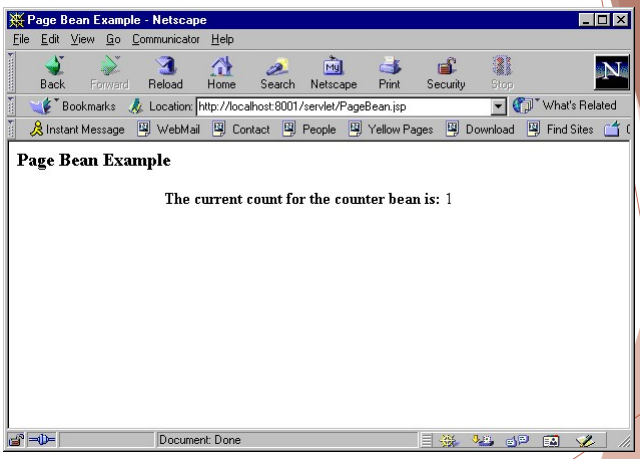
7

EjemploBean1.jsp

```
<jsp:useBean id = "ctr" scope = "request" class = "paq1.Counter" type =
"paq1.Counter"/>

<html>
  <head>
    <title>Ejemplo del manejo del scope de Beans</title>
  </head>
  <body>
    <h3>Ejemplo de scope de Bean</h3>
    <center>
      <b>La cuenta del counter bean actual es: </b>

      <jsp:expression> ctr.getCount() </jsp:expression>
    </center>
  </body>
</html>
```



La cuenta nunca cambia.

9

Request Scope

•Una página puede llamar a otra y el bean sigue disponible.

10

Modificar EjemploBean1.jsp

```
<jsp:useBean id = "ctr" scope = "request" class = "paq1.Counter" type = "paq1.Counter"/>

<html>
  <head>
    <title>Ejemplo del manejo del scope de Beans</title>
  </head>
  <body>
    <h3>Ejemplo de scope de Bean</h3>
    <center>
      <b>La cuenta del counter bean actual es: </b>

      <jsp:expression> ctr.getCount() </jsp:expression>
    </center>

     <b>Llamando a otra página ... Para ver si el bean sigue </b>
      <jsp:scriptlet> ctr.setCount(10); </jsp:scriptlet>
    </center>
    <jsp:forward page = "EjemploBean2.jsp" />

  </body>
</html>
```

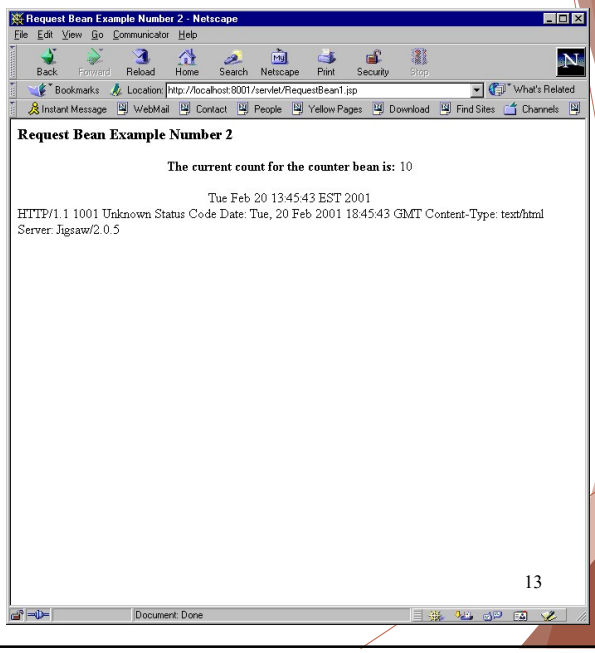
11

EjemploBean2.jsp

```
<%@ page import = "java.util.*"  %>
<jsp:useBean id = "ctr" scope = "request" class = "paq1.Counter" type =
"paq1.Counter"/>

<html>
  <head>
    <title>Ejemplo 2 del Request Bean</title>
  </head>
<body>
    <h3>Ejemplo 2 del Request Bean</h3>
    <center>
      <b>La cuenta actual del counter bean es: </b>
      <jsp:expression> ctr.getCount()  </jsp:expression>
      <p>
      <jsp:expression> new Date() </jsp:expression>
    </center>
  </body>
</html>
```

12



13

Session Scope

Los beans con scope de ***session*** son accesibles dentro de las páginas que procesan los requests de la misma sesión como en la que fue creado el bean.

El tiempo de vida de la sesión es configurable y controllable por el contenedor de servlets (Tomcat).

Cuando se utiliza el mismo nanevagor, se obtiene el bean de Sesión.

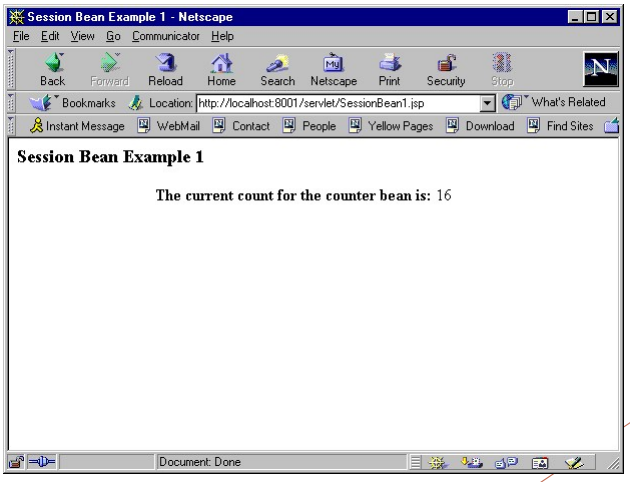
14

Session Scope

```
<jsp:useBean id = "ctr" scope = "session" class = "paq1.Counter" type =
"paq1.Counter/>
<html>
  <head>
    <title>Ejemplo de Bean de sesion</title>
  </head>
  <body>
    < h3>de Bean de sesion </h3>
    <center>
      <b>El valor actual del counter es: </b>
      <jsp:expression> ctr.getCount() </jsp:expression>
    </center>
  </body>
</html>
```

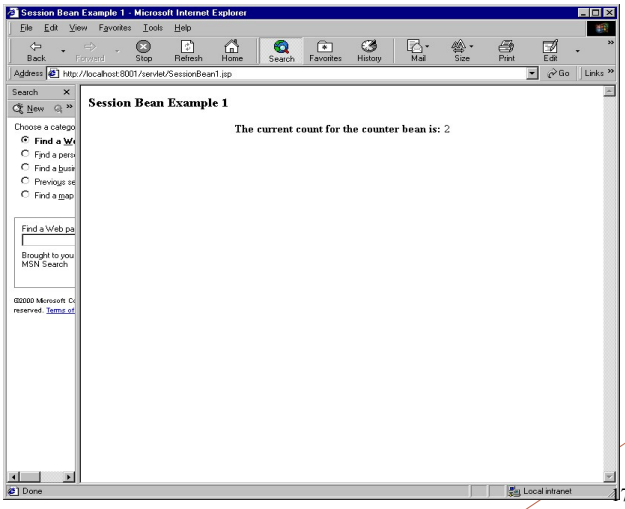
15

El contador se incrementa en cada hit.



16

Probando en otro navegador.



17

Beans de Aplicación

Un bean con el scope de aplicación tiene una disponibilidad más amplia que el bean de sesión.

Los beans de aplicación existen a lo largo de la vida del contenedor de JSPs, lo que significa que no se pierden hasta que se apaga el servidor.

Los beans de sesión están disponibles en requests subsecuentes del mismo navegador. Los beans de aplicación son compartidos por todos los usuarios.

18

Application Bean

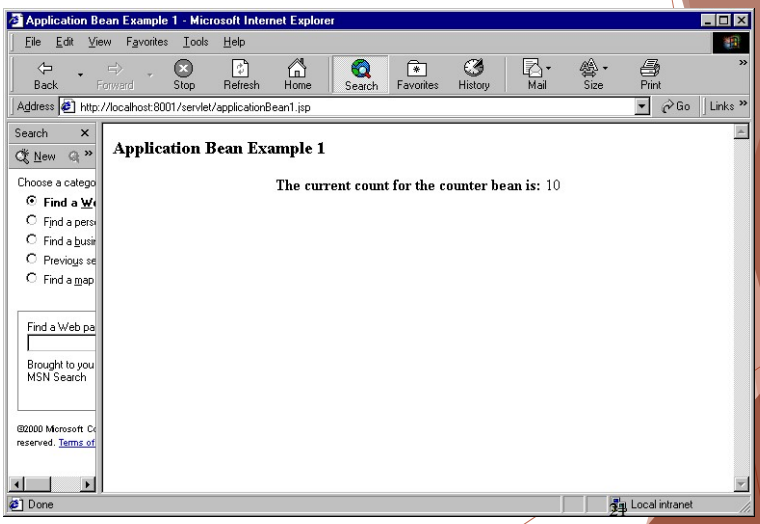
```
<jsp:useBean id = "ctr" scope = "application" class = "Counter" />
<html>
  <head>
    <title>Ejemplo de Bean de Application</title>
  </head>
  <body>
    <h3>Ejemplo de Bean de Application</h3>
    <center>
      <b>El valor actual del counter es : </b>
      <jsp:expression> ctr.getCount() </jsp:expression>
    </center>
  </body>
</html>
```

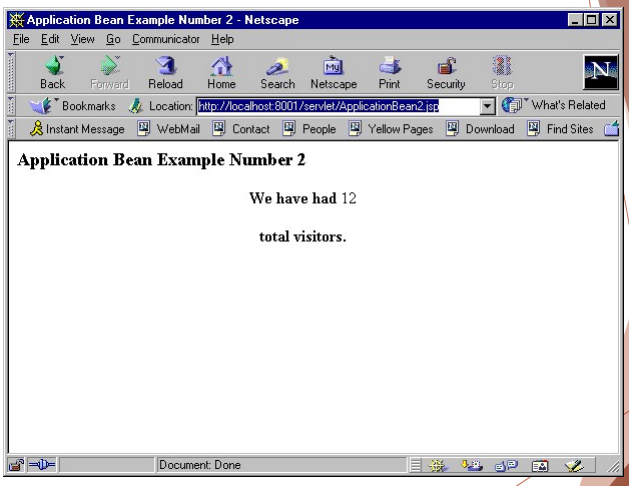
19

Application Bean

```
<%@ page import = "java.util.*"  %>
<jsp:useBean id = "ctr" scope = "application" class = "Counter" />
<html>
  <head>
    <title>Ejemplo 2 del bean de aplication</title>
  </head>
<body>
    <h3>Ejemplo 2 del bean de aplication</h3>
    <center>
      <b>Hemos recibido </b>
      <jsp:expression> ctr.getCount()  </jsp:expression>
      <p>
      <b> visitas totales. </b>
    </center>
  </body>
</html>
```

20


Application Bean Example 1 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Size Print
Address http://localhost:8001/servlet/applicationBean1.jsp
Go Links
Search
New
Application Bean Example 1
The current count for the counter bean is: 10
Done
Local intranet


Application Bean Example Number 2 - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://localhost:8001/servlet/ApplicationBean2.jsp What's Related
Instant Message WebMail Contact People Yellow Pages Download Find Sites
Application Bean Example Number 2
We have had 12
total visitors.
Document: Done

22

Acceso a las propiedades del bean

jsp: setProperty

- Para obtener valores de 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: 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: 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. Sin embargo, 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 estándar 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 proporcione 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.

Ejemplo:

- Uso de un JSP y JavaBean para el control de acceso de usuarios dentro de una aplicación, donde el usuario debe proporcionar la información de: rol, nombre, jefe y contraseña.

Crea el siguiente index.html

```
<html>    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <form action="Registro.jsp" method="post">
            <table>
                <tr><td>Ingresar nombre</td> <td><input type="text" name="nombre"/></td></tr>
                <tr><td>Ingresar apellido</td> <td><input type="text" name="apellido"/></td></tr>
                <tr><td>Ingresar puesto</td> <td><input type="text" name="puesto"/></td></tr>
                <tr><td>Ingresar jefe</td> <td><input type="text" name="jefe"/></td></tr>
                <tr><td>Ingresar contraseña</td> <td><input type="password"
    name="contrasenia"/></td></tr>
            </table>
            <input type="submit" value="Ingresar"/>
        </form>
    </body>
</html>
```

Crea el siguiente bean sesion.java

```
public class Sesion
{
    private String nombre;
    private String apellido;
    private String puesto;
    private String jefe;
    private String contrasenia;

    public String getNombre() {
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public String getApellido() {
        return apellido;
    }
    public void setApellido(String apellido) {
        this.apellido = apellido;
    }
    public String getPuesto() {
        return puesto;
    }
    public void setPuesto(String puesto) {
        this.puesto = puesto;
    }
    public String getJefe() {
        return jefe;
    }
    public void setJefe(String jefe) {
        this.jefe = jefe;
    }
    public String getContrasenia() {
        return contrasenia;
    }
    public void setContrasenia(String contrasenia) {
        this.contrasenia = contrasenia;
    }

}
```

Crea el Registro.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id = "registro" scope = "session" class = "paq1.Sesion" type = "paq1.Sesion" />
<jsp:setProperty name="registro" property="nombre" param="nombre" />
<jsp:setProperty name="registro" property="apellido" param="apellido" />
<jsp:setProperty name="registro" property="puesto" param="puesto" />
<jsp:setProperty name="registro" property="jefe" param="jefe" />
<jsp:setProperty name="registro" property="contrasenia" param="contrasenia" />
<html>
    <head><title>Registro de Usuarios</title></head>
    <body>
        <h1>Datos del usuario registrado</h1>
        <p>Nombre registrado <jsp:getProperty name="registro" property="nombre"/></p>
        <p>Apellido <jsp:getProperty name="registro" property="apellido"/></p>
        <p>Puesto que ocupa <jsp:getProperty name="registro" property="puesto"/></p>
        <p>Nombre del jefe <jsp:getProperty name="registro" property="jefe"/></p>
    </body>
</html>
```

Ejemplo del Carrito de compras