



Capa de presentación: JSP

Sesión 1:

JSP Básico

- **Introducción a los JSP. Comparación con servlets**
- **Elementos de JSP**
 - **Código Java**
 - **Objetos implícitos**
- **Directivas**
- **Acciones**

- **Introducción a los JSP. Comparación con servlets**
- **Elementos de JSP**
 - **Código Java**
 - **Objetos implícitos**
- **Directivas**
- **Acciones**

■ Código Java en páginas HTML

```
<html>
<head>
<title>Mi primera página JSP</title>
</head>
<body>
<h1> Hola, <%= request.getParameter("nombre") %>
Hoy es: <%= new java.util.Date() %> </h1>
</body>
</html>
```

■ En apariencia

- **Un JSP es HTML + Java Insertado**
- **Un Servlet es Java + HTML insertado**

■ En realidad

- **Los JSP se traducen internamente a servlets**
 - **El servidor web tiene una “plantilla de servlet”**
 - **Inserta nuestro código JSP dentro**
 - **Lo guarda en un directorio especial**
 - **Lo compila y ejecuta**
 - **En sucesivas llamadas a la página, solo hace falta ejecutar el servlet, salvo que se modifique el código del JSP ⇒ comenzar de nuevo**

- **Introducción a los JSP. Comparación con servlets**
- **Elementos de JSP**
 - **Código Java**
 - **Objetos implícitos**
- **Directivas**
- **Acciones**

■ Código Java

- Sentencias (*scriptlets*) `<% tam = 1; %>`
- Declaraciones `<%! int i = 0; %>`
- Expresiones `<%= new Date() %>`

■ Directivas: para “afinar” el *servlet* generado

■ Acciones: alteración del flujo de ejecución (normalmente)

Scriptlets



```
<%  
    Calendar ahora = Calendar.getInstance();  
    int hora = ahora.get(Calendar.HOUR_OF_DAY);  
%>  
<b> Hola mundo, <i>  
<% if ((hora>20) || (hora<6)) { %>  
    buenas noches  
<% }  
    else if ((hora>=6) && (hora<=12)) { %>  
        buenos días  
    <% }  
        else { %>  
            buenas tardes  
    <% } %> </i> </b>
```


- **Las variables o métodos declarados se insertan en el cuerpo del servlet generado \Rightarrow se conservan entre peticiones**

```
<%! private int accesos = 0; %>
```

```
<% ++accesos; %>
```

- **Se pueden sobrescribir los métodos `jspInit` y `jspDestroy` equivalentes al `init` y `destroy` de los servlets**

- Su valor se evalúa, se convierte a cadena y se imprime en el *Writer* del servlet, con un `write` o similar

```
<b>
Esta página ha sido visitada <%= visitas %> veces
Hoy es <%= new java.util.Date() %>
</b>
```

- **Variables definidas en la “plantilla” de servlet para JSP, por tanto accesibles a nuestro código**
- **Objetos**
 - **request (HttpServletRequest), response (HttpServletResponse)**
 - **out: el writer para enviar la salida al cliente**
 - **session: HttpSession**
 - **application: ServletContext**
 - **config: ServletConfig**
 - **pageContext**
 - **exception**

- **Introducción a los JSP. Comparación con servlets**
- **Elementos de JSP**
 - **Código Java**
 - **Objetos implícitos**
- **Directivas**
- **Acciones**

- **Influyen en la estructura que tendrá el *servlet* generado**

```
<%@ directiva atributo="valor" %>
```

- **Tipos**
 - **page:** usos variados
 - **include:** equivalente al `#include` de C
 - **taglib:** para usar librerías de etiquetas

- **Import (= import de Java)**

```
<%@page import="java.util.*, java.sql.*" %>
```

- **contentType (= response.setContentType())**

```
<%@page contentType="text/html" %>
```

- **isThreadSafe (= implements SingleThreadModel)**

- **session (defecto: true)**

- **buffer:** ¿Qué hay de raro en esta página JSP?
¿Cómo es que funciona?

```
<html>
<head>
  <title>Untitled</title>
</head>
<body>
<%@ page contentType="text/plain" %>
<h1> Hola </h1>
</body>
</html>
```

- Saltar a una página de error en caso de que se produzca una excepción
- En todas las páginas del sitio (menos la de error):

```
<%@page errorPage="error.jsp" %>
```

- En la página de error:

```
<%@page isErrorPage="true" %>
```

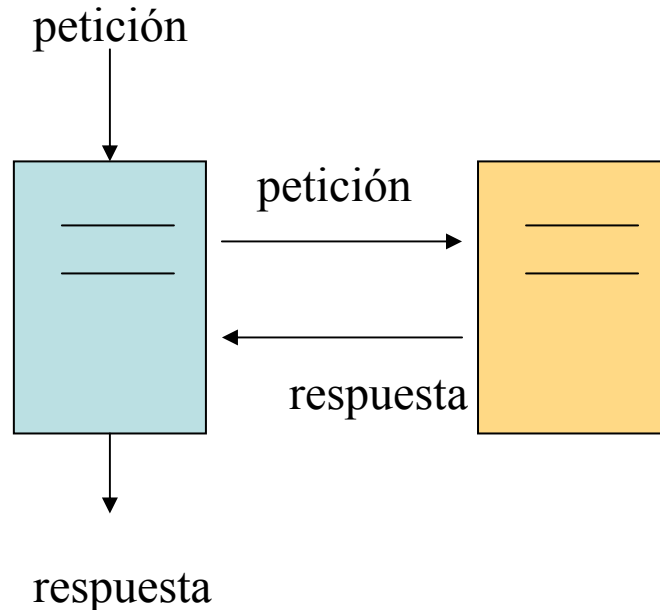
- Esto último hace accesible el objeto implícito `exception`

- **Equivalente al `#include` de C**
- **El código se incluye antes de la compilación del servlet y se compila todo junto**
- **Problema: J2EE no exige que si cambia el código incluido el servidor tenga que volver a componer el servlet**
- **Usos**
 - **Definir variables**
 - **Influir en la respuesta**

- **Introducción a los JSP. Comparación con servlets**
- **Elementos de JSP**
 - **Código Java**
 - **Objetos implícitos**
- **Directivas**
- **Acciones**

Acciones: `<jsp:include>`

- Incluye en una página la salida generada por otra



- Cambios en lo incluido no obligan a recompilar el “principal”

Acciones: <jsp:include> (II)

■ Sintaxis

```
<jsp:include page="URL relativa" flush="true | false" />
```

■ Pasarle parámetros a la página incluida

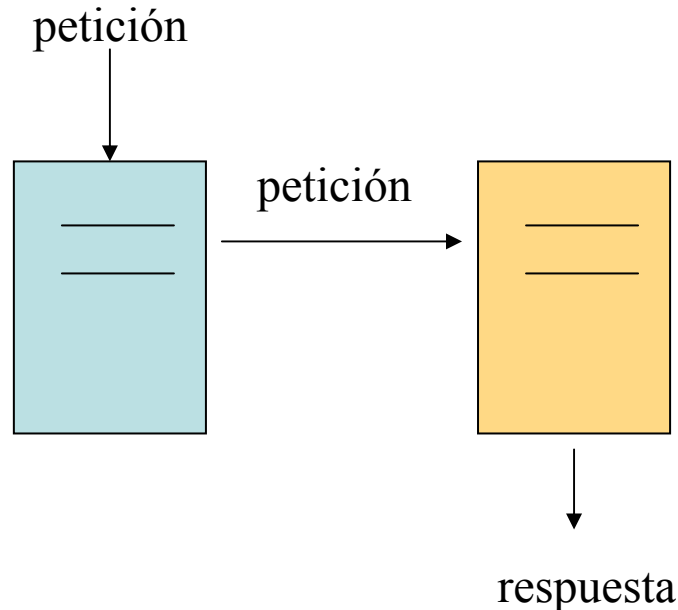
➤ Recibe el `request` (parámetros originales)

➤ Parámetros adicionales:

```
<jsp:include page="cabecera.jsp" flush="false" >  
  <jsp:param name="color" value="red"/>  
</jsp:include>
```

Acciones: `<jsp:forward>`

- Redirige la petición a otra página



- Lo que haya en el *buffer* se descarta

Acciones: <jsp:forward> (II)

■ Sintaxis

```
<jsp:forward page="URL relativa"/>
```

■ Pasarle parámetros a la página incluida: idem al <jsp:include>

- Recibe el `request` (parámetros originales)
- Parámetros adicionales:

```
<jsp:forward page="principal.jsp">  
  <jsp:param name="privilegios" value="root"/>  
</jsp:include>
```