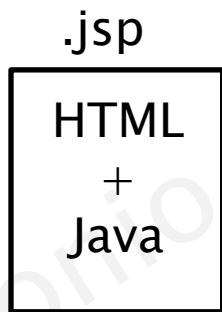


# Java Server Pages (JSP)

# Definición y características

- Componente que forma parte de una aplicación Web JavaEE
- Archivo de texto en el que se combinan bloques HTML con código Java (scriptlet) que se ejecuta en el servidor



- Adecuado para la generación de respuestas

# Código Java en JSP

➤ Se delimita por `<% y %>`. Pueden aparecer en cualquier parte de la página

➤ Puede ser:

- Scriptlet. Bloque Java que realiza alguna tarea
- Expresión. Devuelve un resultado a la página

scriptlet

```
<body>
<center>
  <%for(int i=1;i<=6;i++){ %>
    <h<%=i%>> Bienvendio a mi página</h<%=i%>>
  <%} %>
</center>
</body>
```

expresión

**Bienvendio a mi página**

**Bienvendio a mi página**

**Bienvendio a mi página**

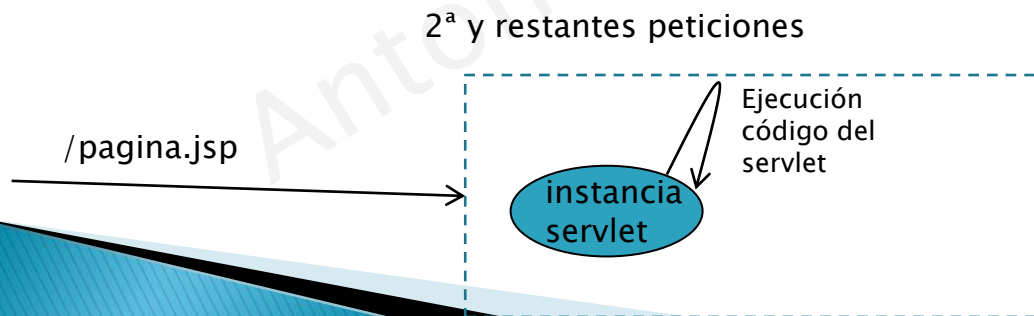
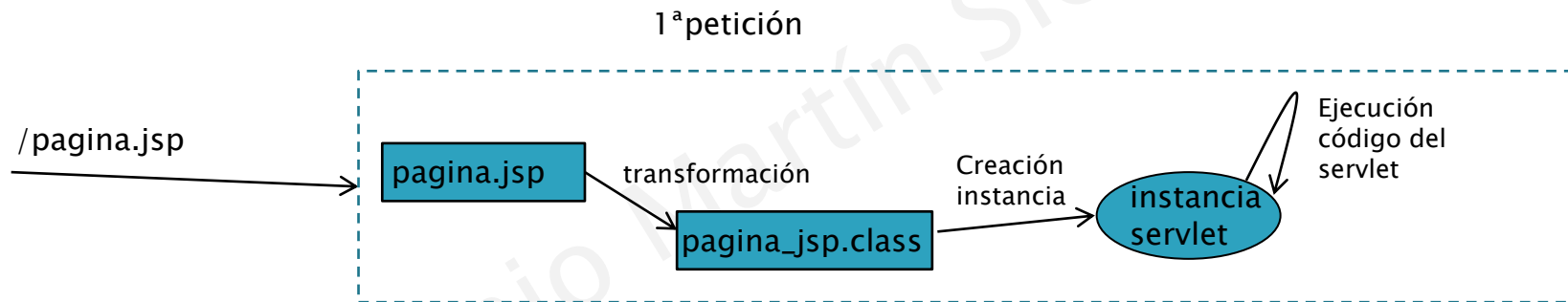
**Bienvendio a mi página**

**Bienvendio a mi página**

**Bienvendio a mi página**

# Ciclo de vida de una página JSP

- Una página JSP es transformada en un servlet cuando es solicitada por primera vez



# Componentes de una página JSP

# Objetos implícitos

➤ Dentro de un scriptlet podemos hacer uso de una serie de objetos implícitos:

- request. Instancia de HttpServletRequest
- response. Instancia de HttpServletResponse
- session. Instancia de HttpSession
- application. Instancia de ServletContext
- exception. Instancia de excepción creada
- out. Objeto PrintWriter de salida Http

# Directivas JSP

➤ Información que se suministra al servidor de aplicaciones durante la fase de transformación a servlet

➤ Sintaxis:

`<%@directiva atributo1="valor" atributo2="valor"..%>`

➤ Directivas:

- `page`. Establece propiedades generales de la página
- `include`. Incluye el contenido de algún archivo externo
- `taglib`. Permite utilizar acciones de librerías externas

# Acciones JSP

- Ejecuta una tarea habitual en la página, que lleva asociada por detrás la ejecución de un código Java.
- Sintaxis basada en XML:

```
<nombre_accion atributo1="valor" atributo2="valor".. >
```



# Directivas JSP

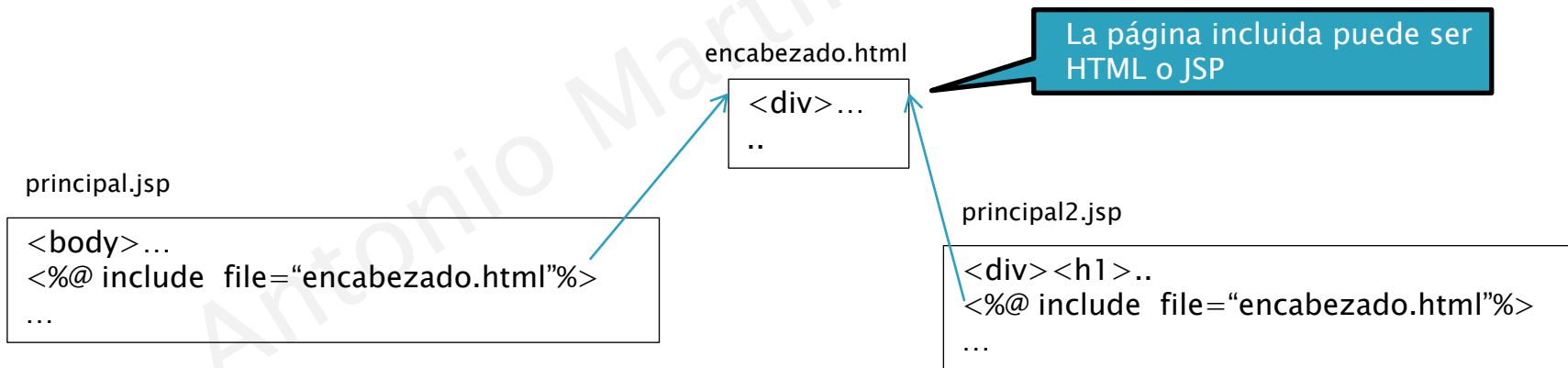
# Directiva page

➤ Proporciona información sobre la página para que sea procesada durante la transformación a servlet. Entre sus principales atributos:

- `language`. Lenguaje utilizado en el código. Por defecto, Java
- `contentType`. Tipo de contenido a generar
- `import`. Clases que deben ser importadas
- `errorPage`. Página a la que será transferido el usuario si se produce una excepción no controlada
- `isErrorPage`. Indica si es o no una página de error

# Directiva include

- Incluye dentro de la página el contenido de un archivo externo. Habitual para encabezados y pies de página
- Mediante el atributo *file* se indica el archivo a incluir:



# Directiva taglib

➤ Incorpora una librería de acciones para poder utilizarlas en la página. Dos atributos:

- uri. Identificador de la librería
- prefix. Prefijo que se debe usar delante de cada nombre de acción

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
:
<c:forEach..>
:
<c:if .. >
```

# Acciones implícitas JSP

# Definición y tipos

- Las acciones son etiquetas que representan llamadas a métodos Java, llamadas que se producen cada vez que la página es solicitada.
- JSP incorpora de forma implícita una serie de acciones, a las que se accede con el prefijo *jsp:*

- forward

- include

- param

- useBean

- setProperty

- getProperty

# forward e include

- Realizan la transferencia de la petición desde la página a otro componente de la aplicación. Equivalen, respectivamente, a los métodos *include* y *forward* del objeto `RequestDispatcher`
- Mediante el atributo *page* se indica la dirección de la página destino, que puede ser otra JSP, un HTML o incluso un servlet:

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

```
<jsp:forward page="NuevoServlet"/>
```

# Integración servlet – JSP



# Criterios de utilización

➤ En una aplicación Web se emplean habitualmente ambos componentes:

- Utilizaremos servlets cuando no haya que generar respuestas
- Se emplearán páginas JSP siempre que haya que construir dinámicamente una página

Antonio Martín Sierra


# El lenguaje de expresiones EL

# Fundamentos

- Alternativa a instrucciones Java para generar contenido en una página JSP
- Más intuitivo y sencillo que el uso de Java
- No requiere incorporar ningún elemento externo al proyecto, forma parte de la especificación JSP
- Sintaxis:

`${expresion}`

Operación en lenguaje EL que devuelve un resultado



# Objetos implícitos

➤ El lenguaje EL cuenta con una serie de objetos que exponen propiedades para acceder a datos de la aplicación:

- `param`. Acceso a los parámetros de la petición
- `requestScope`. Acceso a atributos de petición
- `sessionScope`. Acceso a atributos de sesión
- `applicationScope`. Acceso a atributos de aplicación
- `cookie`. Acceso a las cookies

# Ejemplos de uso I

## ➤ Recuperación del parámetro “email”:

Versión EL



```
${param.email}
```

Versión Java



```
<%=request.getParameter("email")%>
```

## ➤ Recuperación de la propiedad “titulo” de un JavaBean de identificador “libro” que se encuentra almacenado en un atributo de sesión:

Versión EL



```
${sessionScope.libro.titulo}
```

Versión Java



```
<%Libro lb=(Libro)session.getAttribute("libro");  
if(lb!=null){%>  
    <%=lb.getTitulo()%>  
<%}%>
```

# Ejemplos de uso II

- Recuperación del nombre del primer empleado de la lista “empleados” que se encuentra en un atributo de petición:

Versión EL 

```
${requestScope.empleados[0].nombre}
```

Versión Java 

```
<%List<Empleado> lst=(List<Empleado>)
    request.getAttribute("empelados");
if(lst!=null){%>
    <%=lst.get(0).getNombre ()%>
<%}%>
```

- Recuperación del valor de la coolie “user”:

Versión EL 

```
${cookie.user.value}
```

Versión Java 

```
(bastante largo...)
```

# Operadores EL

➤ Se emplean dentro de expresiones:

- aritméticos. +, -, \*, / y %
- relacionales. >(gt), <(lt), >=(ge), <=(le), !=(ne)
- lógicos. && (and), || (or), ! (not)
- ternario. ?:
- Operador vacío. empty. Devuelve true si es null o tiene tamaño 0 (en caso de una colección)

```
${!empty sessionScope.empleado?sessionScope.empleado.email:"el empleado no existe"}
```

# La librería de acciones JSTL



# Java Standard Tag Library

- Incluye acciones para realizar tareas habituales de lógica de programación (definición de variables, instrucciones de control,...)
- Forma parte de Java Enterprise, pero no todos los servidores la incluyen
- Se combina con expresiones EL para eliminar todo el código Java de de una página JSP

# Grupos de acciones JSTL

- Core. Grupo más importante que incluye las acciones de lógica de programación
- Formato. Contiene acciones para formateado de datos
- SQL. Acciones para enviar instrucciones SQL a la base de datos. No es muy utilizada
- XML. Contiene acciones para manipulación de documentos XML
- Function. Acciones para manipulación de cadenas de caracteres

# Utilización de JSTL en JSP

- Se deberá incorporar la librería al proyecto
- Mediante la directiva taglib, se incluye una referencia a la librería:

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

Identificador librería core

Prefijo para referirse a las acciones

# Principales acciones JSTL

# Acción set

➤ Se emplea para establecer un valor a una variable

➤ Entre sus principales atributos:

- var. Nombre o identificador asignado a la variable

- value. Valor asignado a la variable

➤ Ejemplos:

```
<c:set var="num" value="5"/>
```

```
<c:set var="result" value="${param.lado*param.lado}"/>
```



Valores: \${num}    \${result}

# Acción forEach

➤ Se emplea para recorrer un array, lista o conjunto de datos

➤ Entre sus principales atributos:

- **var.** Nombre o identificador asignado a la variable que apunta a cada elemento

- **items.** Conjunto de elementos a recorrer

```
<h1>Listado de empleados:</h1>
<ul>
  <c:forEach var="emp" items="${requestScope.empleados}">
    <li>${emp.nombre}</li>
  </c:forEach>
</ul>
```

Recorre la lista de sesión con los empleados y muestra el nombre de cada uno

# forEach. Atributo varStatus

- Apunta al objeto que representa el estado de la iteración
- A través de la propiedad *index* del objeto accedemos al índice de la iteración.

```
<h1>Listado de empleados:</h1>  
<ul>  
  <c:forEach var="emp" items="${requestScope.empleados}" varStatus="estado">  
    <li>${emp.nombre}  ${estado.index}</li>  
  </c:forEach>  
</ul>
```

Además del nombre del empleado, muestra su posición en la colección

# forEach como for estándar

➤ Se puede utilizar como una instrucción for clásica

➤ Entre sus principales atributos:

- var. Nombre o identificador de la variable de control
- begin. Valor inicial de la variable
- end. Valor final de la variable
- step. Incremento de la variable en cada iteración

```
<c:forEach var="i" begin="1" end="10" step="1">  
    ${param.num*i}<br/>  
</c:forEach>
```



# Acción if

- Procesa un conjunto de sentencias si se cumple una condición
- La condición se establece en su atributo *test*.

```
<c:if test="${not empty param.num}">  
  <c:forEach var="i" begin="1" end="10" step="1">  
    ${param.num*i}<br/>  
  </c:forEach>  
</c:if>
```

Muestra la tabla  
de multiplicar de  
un parámetro  
siempre que  
exista

# Acción choose

- Utiliza para evaluar diferentes condiciones y realizar distintas tareas según la condición cumplida
- Cada condición es evaluada mediante la sub-acción when, cuyo formato es idéntico a if
- El bloque otherwise se ejecuta si no se cumple ninguna condición

```
<c:choose>
  <c:when test="${not empty param.num}">
    <c:forEach var="i" begin="1" end="10" step="1">
      ${param.num*i}<br/>
    </c:forEach>
  </c:when>
  <c:otherwise>
    <h1>El parámetro no existe</h1>
  </c:otherwise>
</c:choose>
```

Muestra la tabla de multiplicar de un parámetro siempre que exista, sino, genera mensaje

# Utilización de JSTL en JSP

- Se deberá incorporar la librería al proyecto
- Mediante la directiva taglib, se incluye una referencia a la librería:

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

Identificador librería core

Prefijo para referirse a las acciones