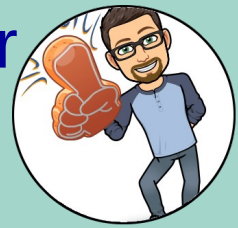




# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Objetivos de la sesión:

- **Listar datos** utilizando **JSTL** para mejorar la presentación en el navegador.
- Entender la diferencia entre **atributos** y **variables de sesión**
- Modificar la webapp para realizar **altas y bajas de datos en el listado** .



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com

Hasta ahora hemos visto como logearnos correctamente en nuestra aplicación, pero si tuviéramos una lista de alumnos en nuestra aplicación ...



¿Como podemos modificar nuestra aplicación para mostrarlos?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

· Para que nuestra aplicación pueda mostrar una lista de datos (por ejemplo alumnos) en las siguientes diapositivas realizaremos los siguientes **pasos**:

**1º Crear la clase Alumno**

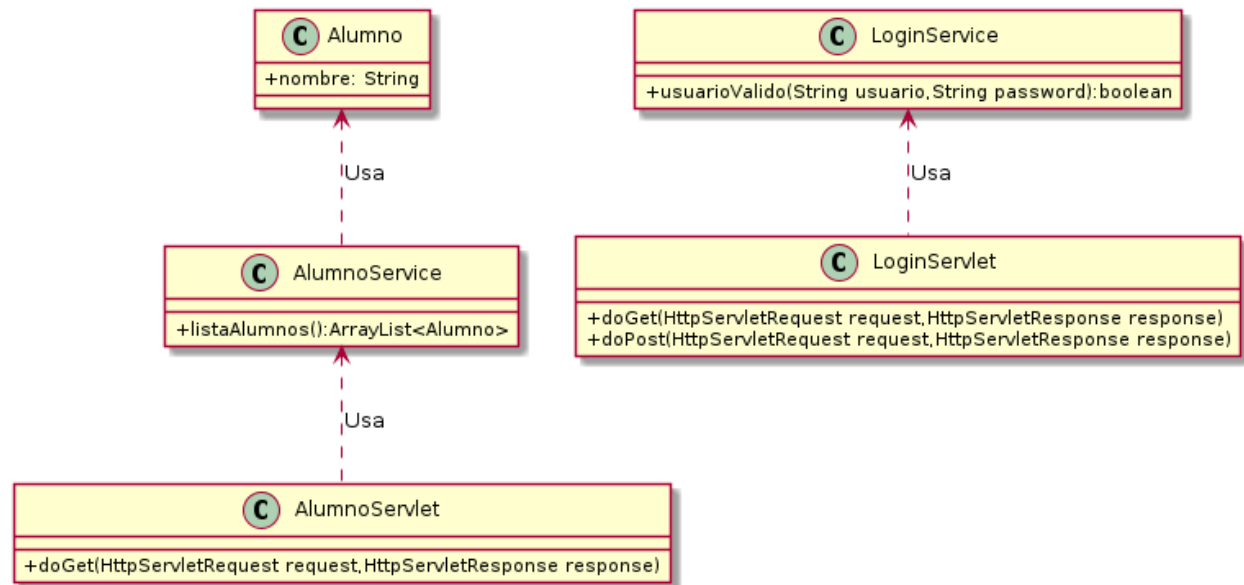
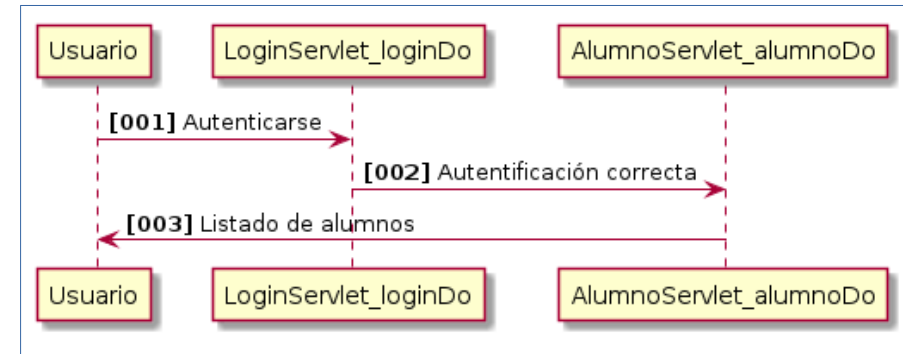
**2º Crear un servicio AlumnoService**

**3º Modificar LoginServlet para pasar la lista de Alumnos como un atributo a la página bienvenida.jsp**

**4º Refactorizar**

**5º Redirigir al nuevo servlet AlumnoServlet y alumno.jsp**

**6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos**





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

#### 1º Crear la clase Alumno

Hay que **crear una clase “Alumno”** con un atributo privado “nombre” de tipo String en el mismo package donde esta LoginServlet. Adicionalmente deberemos **añadir getters, setters, el constructor y el método toString()**.

Para los alumnos que no esten familiarizados con el Eclipse, teniendo el fichero Alumno.java abierto si pulsamos dentro con el botón derecho tenemos las opciones:

- “Source\Generate Getters and Setters...”
- “Source\Generate constructors using fields...” y
- “Source\Generate toString()...” .



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com

Listar datos:

### 2º Crear un servicio AlumnoService

Hay que crear una clase “AlumnoService” y contendrá a una lista estática de alumnos predefinida y un método listaAlumnos():

The screenshot shows an IDE with two main panels. On the left is the 'Project Explorer' showing a project named 'mi-primera-webapp-jee'. Under 'Java Resources' > 'src/main/java', there is a package 'org.profesor.joseramon.primer\_jee' containing 'Alumno.java', 'AlumnoService.java' (highlighted), 'LoginServlet.java', and 'ValidacionUsuarioService.java'. On the right is the editor showing the code for 'AlumnoService.java'.

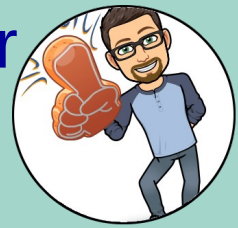
```
1 package org.profesor.joseramon.primer_jee;
2 import java.util.ArrayList;
3 import java.util.List;
4 import org.profesor.joseramon.primer_jee.Alumno;
5 public class AlumnoService {
6     private static List<Alumno> alumnos= new ArrayList<Alumno>();
7     //En una situación real aquí se realizaría un acceso a la BD para obtener
8     //los alumnos, pero en esta unidad didáctica aun no lo haremos.
9
10    //1º Para evitar tener la lista de alumnos en blanco al arrancar la app
11    //debemos crear un bloque estático para inicializar el ArrayList al principio
12    //con 3 valores: "Jose", "Pedro" y "Juan"
13    static {
14        alumnos.add(new Alumno("Jose"));
15        alumnos.add(new Alumno("Pedro"));
16        alumnos.add(new Alumno("Juan"));
17        alumnos.add(new Alumno("Maria"));
18    }
19
20    //2º Crear un método listaAlumnos() que devuelva la lista de alumnos
21
22 }
```





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com

### Listar datos:

### 3º Modificar LoginServlet para pasar la lista de Alumnos como un atributo a la página bienvenida.jsp

Hay que modificar LoginServlet para que bienvenida.jsp pueda mostrar los alumnos y modificar bienvenida.jsp como se muestra en la imagen.

Ayuda: Hará falta implementar el método toString() en Alumno para que se muestre "Alumno [nombre=" + nombre + "]".

```
bienvenida.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>Bienvenid@</title>
8 </head>
9 <body>
10  <p>Hola ${nombre}</p>
11  <p>${alumnos}</p>
12 </body>
13 </html>
```



¿ Por que bienvenida.jsp muestra la lista de alumnos simplemente pasandole la lista de alumnos con `${alumnos}` ?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

**4º Refactorizar :** En una aplicación real cada cierto tiempo debemos refactorizar para mejorar la legibilidad de nuestro código. En nuestro paquete principal de clases java empiezan a acumularse las clases, por lo que vamos a **modificar LoginServlet** para que esté dentro del subpaquete "login". También modificaremos **ValidacionUsuarioService** para que se llame **LoginService** y esté también en el paquete login y también moveremos **Alumno** y **AlumnoService** a un nuevo subpaquete "alumno". Por último cambiaremos los nombres a los 2 servicios disponibles en **LoginServlet** como en la imagen:

The screenshot shows an IDE with two main panels. The left panel is the 'Project Explorer' showing a project named 'mi-primer-webapp-jee'. Under 'Java Resources' > 'src/main/java', there are two packages: 'org.profesor.joseramon.primer\_jee.alumno' and 'org.profesor.joseramon.primer\_jee.login'. The 'login' package is expanded, showing 'LoginService.java' and 'LoginServlet.java'. The 'LoginServlet.java' file is selected and highlighted in orange. The right panel shows the code of 'LoginServlet.java'. It starts with a multi-line comment explaining that a Servlet is a Java class used to extend the capabilities of services that store applications using the request-response model. It then lists four numbered steps: 1. extends javax.servlet.http.HttpServlet, 2. @WebServlet(urlPatterns = "/login.do"), 3. doGet(HttpServletRequest request, HttpServletResponse response), and 4. ¿Como se crea la respuesta?. The code then shows the @WebServlet annotation and the class definition: 'public class LoginServlet extends HttpServlet {'. Inside the class, there are two lines of code: 'LoginService loginServicio = new LoginService();' and 'AlumnoService alumnoServicio = new AlumnoService();'.

```
25 * Un Servlet es una clase de programación Java
26 * utilizado para extender las capacidades de los servicios
27 * que almacenan aplicaciones mediante el modelo de petición (request) respuesta (response)
28 *
29
30 * 1. extends javax.servlet.http.HttpServlet
31 * 2. @WebServlet(urlPatterns = "/login.do")
32 * 3. doGet(HttpServletRequest request, HttpServletResponse response)
33 * 4. ¿Como se crea la respuesta?
34 */
35 @WebServlet(urlPatterns = "/login.do")
36 public class LoginServlet extends HttpServlet {
37     LoginService loginServicio = new LoginService();
38     AlumnoService alumnoServicio = new AlumnoService();
39 }
```



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

Listar datos:

### 5º Redirigir al nuevo servlet AlumnoServlet y alumno.jsp

Es más lógico que cuando el login sea correcto LoginServlet nos envíe a alumno.jsp (donde se mostrarán los alumnos) en vez de ir a bienvenida.jsp.

Para conseguirlo debemos **crear un nuevo servlet AlumnoServlet** en el paquete alumno cuya **url será "/alumno.do"**. Para rellenar AlumnoServlet podemos copiar y pegar instrucciones de LoginServlet y **dejar solo el servicio de alumnos y el doGet** que **añadirá el atributo "alumnos"** y **redirigirá a alumno.jsp**.

Adicionalmente deberemos de **crear alumno.jsp** (podemos copiar información de bienvenida.jsp).

Hasta ahora hemos reenviado de un servlet a una página JSP, pero ...

```
alumno.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Alumnos</title>
8 </head>
9 <body>
10 <p>${alumnos}</p>
11 </body>
12 </html>
```



¿Como podemos redirigir desde el servlet LoginServlet hasta AlumnoServlet cuando el login sea correcto?





### Listar datos:

... continuación **5º Redirigir al nuevo servlet AlumnoServlet y alumno.jsp**

Vamos a redirigir el servlet LoginServlet hasta AlumnoServlet utilizando “sendRedirect()”:

```

LoginServlet.java
36 public class LoginServlet extends HttpServlet {
37     LoginService loginServicio = new LoginService();
38     AlumnoService alumnoServicio = new AlumnoService();
39
40     @Override
41     protected void doGet(HttpServletRequest request,
42         HttpServletResponse response) throws IOException, ServletException {
43         request.getRequestDispatcher("/WEB-INF/views/login.jsp").forward(request, response);
44     }
45     @Override
46     protected void doPost(HttpServletRequest request,
47         HttpServletResponse response) throws IOException, ServletException {
48
49         String nombre = request.getParameter("nombre");
50         String password = request.getParameter("password");
51
52         if (loginServicio.usuarioValido(nombre, password)) {
53             //validación correcta: Redirigir al Servlet de alumno
54             response.sendRedirect("alumno.do");
55         } else {
56             //validación incorrecta
57             request.setAttribute("errores", "Usuario '" + nombre + "' o contraseña incorrecta");
58             request.getRequestDispatcher("/WEB-INF/views/login.jsp").forward(request, response);
59         }
60     }
61 }
    
```



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL

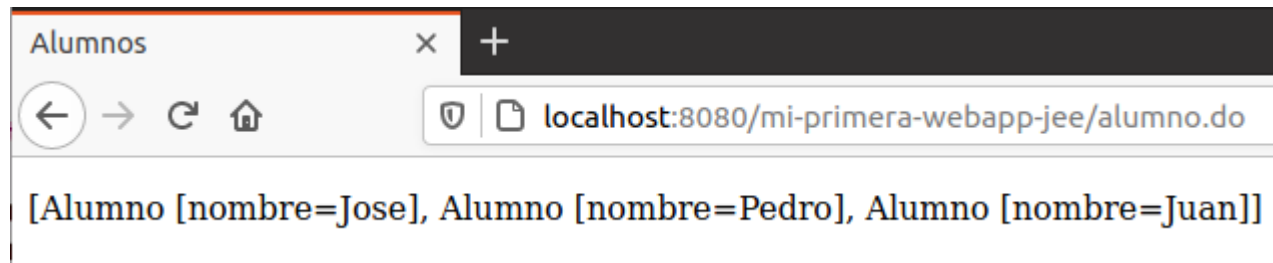


Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

... continuación **5º Redirigir al nuevo servlet AlumnoServlet y alumno.jsp**

Ahora debemos comprobar desde el navegador que cuando introducimos un login correcto nos redirige a alumno.do y nos muestra el listado:





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos: JSTL es el acrónimo de **Java Server Pages Standard Tags Library**. JSTL es una librería de tags que nos ayudará a codificar nuestras páginas JSP. Para utilizar JSTL tenemos que **añadir la librería de tags (JSTL) a la página JSP** como se puede observar en la primera línea del fichero JSP. Sin realmente utilizar los tags de JSTL **comprobemos que cambia al ejecutar la aplicación web**:

The screenshot shows an IDE with the Project Explorer on the left and the editor on the right. The Project Explorer shows a project named 'mi-primer-webapp-jee' with a 'src' folder containing 'main', 'resources', and 'webapp'. The 'webapp' folder contains a 'WEB-INF' folder, which in turn contains a 'views' folder. The 'alumno.jsp' file is located in the 'views' folder. The editor shows the content of 'alumno.jsp' with the following code:

```
1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
2 <%@ page language="java" contentType="text/html; charset=UTF-8"
3   pageEncoding="UTF-8"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8   <title>Alumnos</title>
9 </head>
10 <body>
11   <p>${alumnos}</p>
12 </body>
13 </html>
14
```



¿Por que este cambio?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

... continuación **6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos:**

El error nos aparece porque realmente aunque le hemos dicho que utilice la librería de tags JSTL en la página JSP, ***en nuestro proyecto no existe la librería porque no le hemos dicho al Maven que la incorpore.***

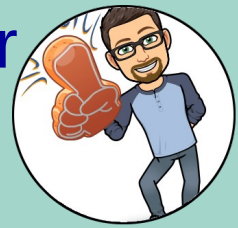






# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com

### Listar datos:

... continuación **6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos:**

Para solucionarlo **agregamos la nueva dependencia en el fichero pom.xml**. Si queremos saber que código añadir al fichero pom.xml para utilizar un artefacto y cual es la última versión de dicha dependencia podemos **teclear "maven jstl" en el buscador** del navegador y nos irá a una página de la web <https://mvnrepository.com/> donde podemos ver todas las versiones y el código a añadir en la sección de dependencias al fichero pom.xml si queremos utilizar dicho artefacto.

No nos obsesionemos con poner la última versión de la dependencia porque a veces son incompatibles con otras dependencias que ya tenemos en uso.

The screenshot shows the Maven Repository website for the JSTL 1.2 artifact. The browser address bar shows <https://mvnrepository.com/artifact/jstl/jstl/1.2>. The page includes a search bar, a graph of indexed artifacts (18.0M), and a list of popular categories. The main content area displays the JSTL 1.2 artifact details, including the date (May 12, 2006), files (pom (141 bytes), jar (404 KB)), repositories (Central, SciJava Public, Spring Lib Release), and used by (254 artifacts). The Maven tab is selected, showing the XML snippet for the dependency:

```
<!-- https://mvnrepository.com/artifact/jstl/jstl -->
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL

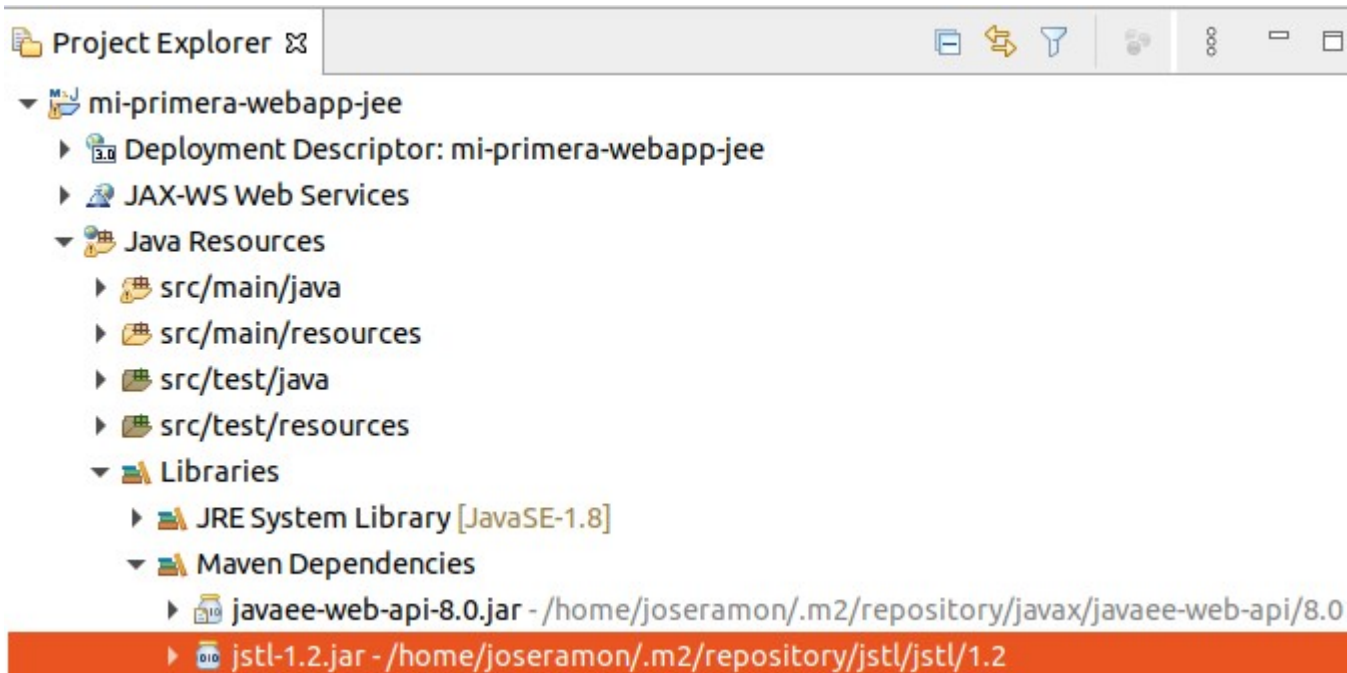


Desarrollo Web en Entorno Servidor - [joseramon.profesor@gmail.com](mailto:joseramon.profesor@gmail.com)

### Listar datos:

... continuación **6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos:**

Si lo hemos hecho bien veremos que en las **librerías de Maven** en nuestro proyecto ya aparece la **librería jstl-1.2.jar**:



¿Para que vamos a gastar esta librería?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



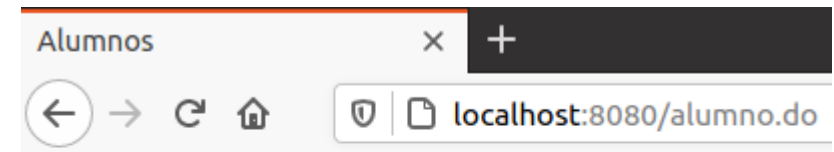
Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

... continuación **6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos:**

En HTML si queremos imprimir una lista utilizamos los tags `<ol>` y `<li>` como podemos ver en el ejemplo:

```
alumno.jsp
1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
2 <%@ page language="java" contentType="text/html; charset=UTF-8" %>
3 <% pageEncoding="UTF-8"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7 <meta charset="UTF-8">
8 <title>Alumnos</title>
9 </head>
10 <body>
11 <ol>
12 <li>Primer alumno</li>
13 <li>Segundo alumno</li>
14 <li>Tercer alumno</li>
15 </ol>
16 </body>
17 </html>
```



1. Primer alumno
2. Segundo alumno
3. Tercer alumno

***Nuestra intención es utilizar la librería para crear un bucle y poder rellenar dicha lista en base a los alumnos.***



¿Como?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



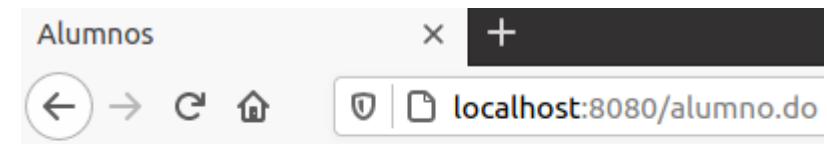
Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Listar datos:

... continuación 6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos:

Para **crear un bucle** utilizamos el **tag *forEach*** y como el prefijo de la librería le hemos dicho que es "c" (***prefix="c"***) al añadir la librería, nuestro tag realmente será ***c:forEach***. El prefijo podemos cambiarlo al declarar la librería si nos interesara. Cambiemos **alumno.jsp** y comprobemos el resultado :

```
alumno.jsp
1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
2 <%@ page language="java" contentType="text/html; charset=UTF-8"
3   pageEncoding="UTF-8"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8   <title>Alumnos</title>
9 </head>
10 <body>
11 <ol>
12   <c:forEach items="${alumnos}" var="alumno">
13     <li>${alumno}</li>
14   </c:forEach>
15 </ol>
16 </body>
17 </html>
```



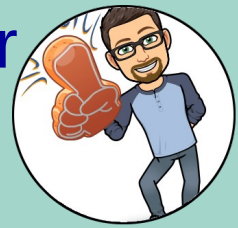
1. Alumno [nombre=Jose]
2. Alumno [nombre=Pedro]
3. Alumno [nombre=Juan]





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL

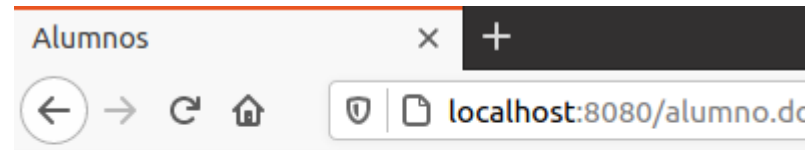


Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

Listar datos:

... continuación **6º Utilizar JSTL para mejorar el aspecto de la lista de alumnos:**

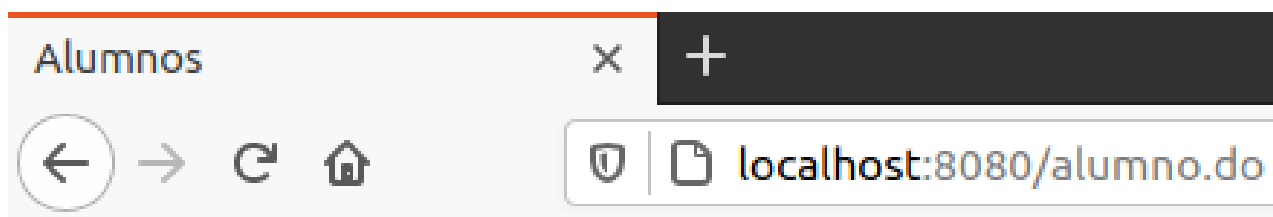
Modifica SOLO el fichero alumno.jsp para que se visualice de la siguiente manera :



1. Jose
2. Pedro
3. Juan



Ahora que ya tenemos el listado, queremos que en la página de alumnos (alumnos.jsp) aparezca un mensaje saludando al usuario como en la imagen:



Bienvenido joseramon

1. Jose
2. Pedro
3. Juan



¿Como podemos hacerlo? ¿Que fichero/s hay que modificar?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### Atributos vs Sesión:

- La primera idea es ***realizar los siguientes cambios:***
  - Modificar LoginServlet para que cuando redirige a AlumnoServlet se le pase como atributo el nombre del usuario
  - Modificar AlumnoServlet para que cuando abra alumno.jsp pasarle el atributo
  - Modificar alumno.jsp para que muestre el saludo al usuario.

Para continuar la explicación es necesario primero **realizar los cambios anteriores.**



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL

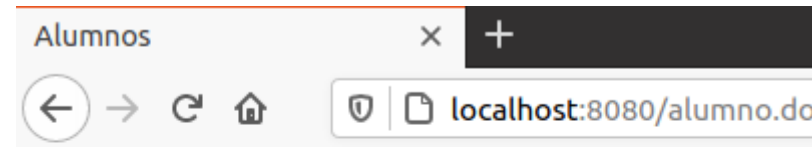
Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



### Atributos vs Sesión:

Sin embargo, una vez hechos los cambios el resultado es el siguiente:

```
alumno.jsp
1 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
2 <%@ page language="java" contentType="text/html; charset=UTF-8"
3   pageEncoding="UTF-8"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8   <title>Alumnos</title>
9 </head>
10 <body>
11 <p>Bienvenido ${nombre}</p>
12 <ol>
13   <c:forEach items="${alumnos}" var="alumno">
14     <li>${alumno.getNombre()}</li>
15   </c:forEach>
16 </ol>
17 </body>
18 </html>
```



Bienvenido

1. Jose
2. Pedro
3. Juan



¿Por que la página JSP no tiene el valor del nombre?





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Atributos vs Sesión:**

Para entender porque no ha funcionado abrimos el navegador y vamos a hacer una traza de las peticiones recibidas y enviadas al servidor:

Cuando nos pide el usuario y el login se realiza una petición GET:

The screenshot shows a web browser window with a single tab titled 'Login'. The address bar shows 'localhost:8080'. The page content includes a login form with two input fields: 'Introduzca su nombre:' containing 'joseramon' and 'Introduzca su contraseña:' containing '.....'. A 'Login' button is to the right of the password field. Below the form is the browser's developer console. The 'Red' (Network) tab is active, showing a single request. The table below summarizes the request details:

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño	
200	GET	localhost:8080	/	BrowserTabChild.jsm:102 (document)	html	490 B	347 B	1 n



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Atributos vs Sesión:**

Y cuando pulsamos en el botón “Login” estamos haciendo un POST. En dicha petición POST (doPost en LoginServlet) configuramos el atributo nombre.

Sin embargo en el doPost no reenviamos a alumno.jsp directamente, sino que redirigimos la salida al Servlet AlumnoServlet(302= redirección), con lo cual se genera otra petición de GET sobre la url alumno.do atendida por AlumnoServlet. El atributo “nombre” lo habíamos configurado para la primera petición (login.do), no para la segunda (alumno.do).

Alumnos

localhost:8080/alumno.do

Bienvenido

1. Jose
2. Pedro
3. Juan

Estado	Método	Dominio	Archivo	Iniciad
302	POST	localhost:8080	login.do	docum
200	GET	localhost:8080	alumno.do	docum



¿Como podemos pasar información de un Servlet a otro?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



... continuación **Atributos vs Sesión:**

***Si un Servlet delega en una página JSP para presentar la página HTML de respuesta se esta realizando en la misma petición***, por ello los atributos pasados a la petición (request) se pueden ver desde las páginas JSP.

Sin embargo ***si queremos comunicarnos entre 2 Servlets deberemos de utilizar otra técnica*** para hacerlo porque cuando un Servlet se redirige a otro se genera una petición distinta.

La solución pasa por ***almacenar dicha información en la sesión***. La sesión es algo que comienza cuando el usuario se logea y desaparece cuando hace logOut. Muchas veces no hace falta hacer logOut , porque existe un timeOut que permite cerrar la sesión automáticamente si se excede un tiempo (timeOut) sin hacer nada. El timeOut se utiliza por ejemplo en aplicaciones de Banca.



¿Como almacenamos la información en las sesiones en Java?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com

... continuación **Atributos vs Sesión:**

Para almacenar la información en la sesión debemos primero obtener la sesión (`request.getSession()`) y posteriormente añadirle el atributo (`.setAttribute(,)`).

Añadamos en `LoginServlet` el nombre en la sesión y comprobemos que ya funciona:

```
LoginServlet.java
1 package org.profesor.joseramon.primer_jees.login;
2
3 import java.io.IOException;
13
15 * Browser sends Http Request to Web Server
22
23 /*Java Platform, Enterprise Edition (Java EE) JEE8
35 @WebServlet(urlPatterns = "/Login.do")
36 public class LoginServlet extends HttpServlet {
37     LoginService loginServicio = new LoginService();
38     AlumnoService alumnoServicio = new AlumnoService();
39
40 @Override
41 protected void doGet(HttpServletRequest request,
42     HttpServletResponse response) throws IOException, ServletException {
43     request.getRequestDispatcher("/WEB-INF/views/login.jsp").forward(request, response);
44 }
45 @Override
46 protected void doPost(HttpServletRequest request,
47     HttpServletResponse response) throws IOException, ServletException {
48
49     String nombre = request.getParameter("nombre");
50     String password = request.getParameter("password");
51
52     if (loginServicio.usuarioValido(nombre, password)) {
53         request.getSession().setAttribute("nombre", nombre);
54         //validación correcta: Redirigir al Servlet de alumno
55         response.sendRedirect("alumno.do");
56     } else {
57         //validación incorrecta
58         request.setAttribute("errores", "Usuario " + nombre + " o contraseña incorrecta");
59         request.getRequestDispatcher("/WEB-INF/views/login.jsp").forward(request, response);
60     }
61 }
62 }
```





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com

... continuación **Atributos vs Sesión:**

Si la información está en la sesión realmente no hace falta pasar a la página JSP el nombre como atributo. Comentemos las 2 líneas en `AlumnoServlet` donde se pasa el atributo “nombre” y comprobemos si se sigue viendo el nombre:

```
AlumnoServlet.java
1 package org.profesor.joseramon.primer_je.e.alumno;
2
3 import java.io.IOException;
4
12
13 @WebServlet(urlPatterns = "/alumno.do")
14 public class AlumnoServlet extends HttpServlet {
15     AlumnoService alumnoServicio= new AlumnoService();
16
17     @Override
18     protected void doGet(HttpServletRequest request,
19         HttpServletResponse response) throws IOException, ServletException {
20         /*String nombre =request.getParameter("nombre");
21         request.setAttribute("nombre",nombre);*/
22         request.setAttribute("alumnos",alumnoServicio.listaAlumnos());
23         request.getRequestDispatcher("/WEB-INF/views/alumno.jsp").forward(request, response);
24     }
25 }
```

Ahora la pregunta de un posible alumno aventajado seria:



¿Porque no pasamos toda la información en la sesión (lista de alumnos incluida)?



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Atributos vs Sesión:**

El primer motivo es que pasar toda la información e la sesión es como hacer todas las variables globales, o sea, una muy mala práctica.

El segundo motivo es que la información de ***la sesión de un usuario la guarda el servidor en memoria***, y por lo tanto debemos guardar en la sesión el menor número de datos posible para evitar que cuando el servidor tenga muchos usuarios el sistema se colapse.



### Añadir datos a la lista:

Hasta ahora solo hemos mostrado el contenido de una lista, pero nos falta averiguar como podemos añadir y borrar información de esta lista.



¿Como podemos añadir alumnos nuevos a nuestra lista?

Alumnos x +

← → ↻ 🏠 🔒 📄 localhost:8080/alumno.do

Bienvenido joseramon

1. Jose
2. Pedro
3. Juan
4. Santiago

Añadir



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

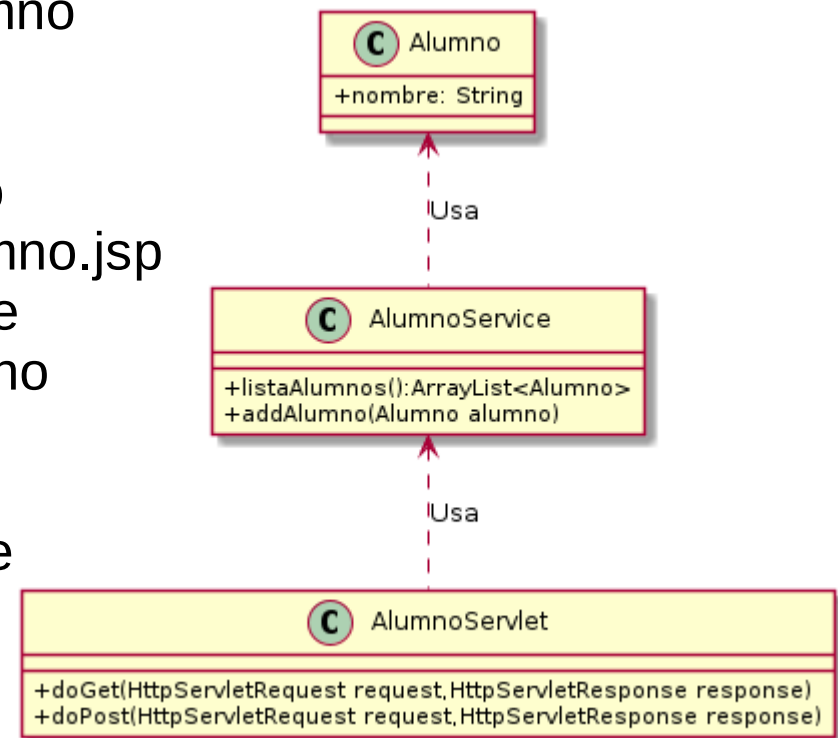
... continuación **Añadir datos a la lista:**

Para poder añadir alumnos a nuestra lista **realizaremos los siguientes pasos:**

**1º AlumnoService.java:** Crear un método nuevo “addAlumno(Alumno alumno)” que añada el alumno introducido como parámetro en la lista.

**2º AlumnoServlet.java:** Crear un método nuevo “doPost()” que antes de pasarle el testigo a alumno.jsp (podemos copiar el contenido de doGet()) realice la inserción (utilizando AlumnoService) del alumno recibido como parámetro .

**3º alumnos.jsp:** Añadir al final un formulario que pida el nombre del alumno nuevo y realice una llamada a “alumno.do”.



Realiza los pasos anteriores y comprueba que puedes añadir nuevos alumnos.



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Añadir datos a la lista:**

Tras haber añadido un nuevo alumno dale a “Recargar” la página:

¿Que problema tenemos? ¿Como podemos evitarlo?





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Añadir datos a la lista:**

Para **evitar inserciones duplicadas** deberemos **insertar el alumno** y en vez de pasar el testigo a alumno.jsp para que muestre el listado de alumnos, deberemos **redirigir de nuevo a "alumno.do"** para que sea el método doGet el que liste los alumnos.

Alumnos

← → ↻ 🏠 🔒 localhost:8080/alumno.do

Bienvenido joseramon

1. Jose
2. Pedro
3. Juan
4. Santiago

Inspector Consola Depurador Red Editor de estilos

Filtrar las URL

Estado	Método	Dominio	Archivo	Iniciador
302	POST	localhost:8080	alumno.do	document
200	GET	localhost:8080	alumno.do	document

Realiza el cambio y comprueba que ahora ya no se insertan alumnos duplicados si recargamos la página.



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

**TRUCO:** Como norma cuando queremos realizar una acción y volver a mostrar los datos una vez realizada la acción llamaremos a la página que muestra los datos con `response.sendRedirect()` en vez de cargar los datos con `request.setAttribute()` y ejecutar `request.getRequestDispatcher()`.

### Borrar datos de la lista:

Hasta ahora ya sabemos listar y añadir alumnos a nuestra lista.



¿Como podemos borrar alumnos de nuestra lista?

Alumnos

localhost:8080/alumno.do

Bienvenido joseramon

1. Jose [Borrar](#)
2. Pedro [Borrar](#)
3. Juan [Borrar](#)



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Borrar datos de la lista:**

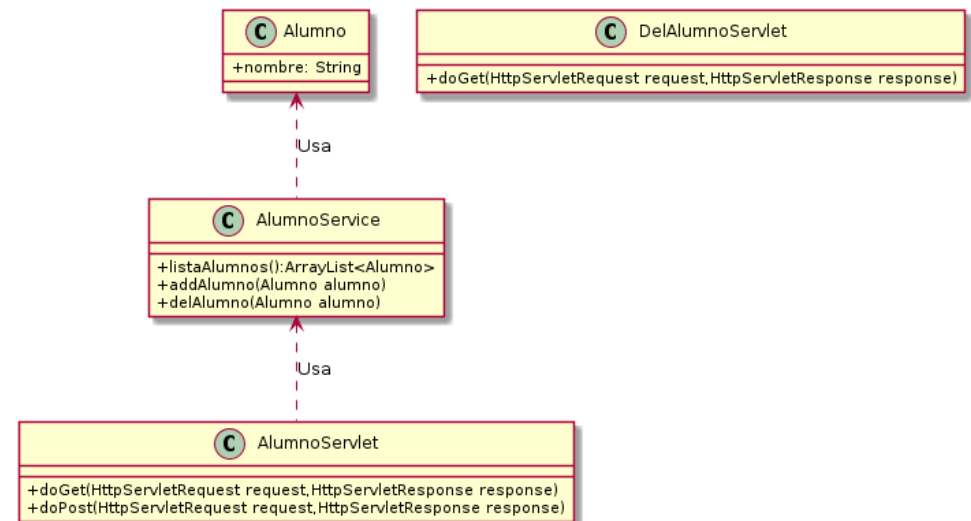
Para poder borrar alumnos en nuestra lista realizaremos los siguientes pasos:

**1º AlumnoService.java:** Crear un método nuevo “delAlumno(Alumno alumno)” que borre el alumno introducido como parámetro en la lista. Recuerda que si no tienes el método equals() en Alumno tendrás un problema para borrar ...

**2º alumnos.jsp:** Añadir junto a cada alumno un enlace a la url “del-alumno.do”. Para saber que alumno se borrará se pasará el nombre del alumno como parámetro en la url.

**3º Crear DelAlumnoServlet.java:** Para crear este servlet podemos copiar el contenido de AlumnoServlet y quitarle el doPost(). La url de este Servlet será “del-alumno.do”. En el método doGet se utilizara el servicio AlumnoService para borrar el alumno y posteriormente se redirigirá otra vez a “alumno.do”.

Realiza los pasos anteriores y comprueba que puedes borrar nuevos alumnos.





# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

... continuación **Borrar datos de la lista:**

Si lo hemos hecho bien al **borrar a Pedro** veremos que se ha generado una petición GET hacia “del-alumno.do?alumno=Pedro” que ha sido redirigida (estado 302) y posteriormente se ha realizado una petición GET hacia “alumno.do”.

Alumnos

← → ↻ 🏠 🔒 📄 localhost:8080/alumno.do

Bienvenido joseramon

1. Jose [Borrar](#)
2. Juan [Borrar](#)

Inspector Consola Depurador ↕ Red {} Editor de

🗑️ 🔍 Filtrar las URL || 🔍 🔇 Todos HTML

Estado	Método	Dominio	Archivo
302	GET	🔒 localhost:8080	del-alumno.do?alumno=Pedro
200	GET	🔒 localhost:8080	alumno.do



# UD 1: Introducción a los lenguajes de servidor

## 4.- JEE: Servlets, JSP y JSTL



Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

### EJERCICIO:

Sigue todos los pasos de los PDF y sube la aplicación final al moodle.

Para ello:

1º Haz un “Run As \Maven Clean” para dejar solo los fichero fuentes y quitar momentaneamente los necesarios para ejecutar la aplicación (dependencias).

2º Comprime la carpeta de tu aplicación y ponle como nombre al fichero comprimido UD1\_practica5\_nombreAlumno.tar.gz donde nombreAlumno es el nombre del alumno que entrega la práctica.

3º Súbela al moodle.

**IMPORTANTE:** No comprimir en RAR, porque Ubuntu no lo lee bien y en clase tenemos Ubuntu. Si tuviesemos Windows, podemos comprimir en ZIP.