
M6.UF4.A6.P2

APLICACION WEB CON JEE

Eduard Lara

INDICE

1. Introducción a aplicaciones web JEE
2. Estructura de un war
3. Patrón MVC
4. Creación proyecto JEE en Eclipse
5. Aplicación Biblioteca
6. Aplicación con base de datos

1. INTRODUCCIÓN APLICACIONES WEB

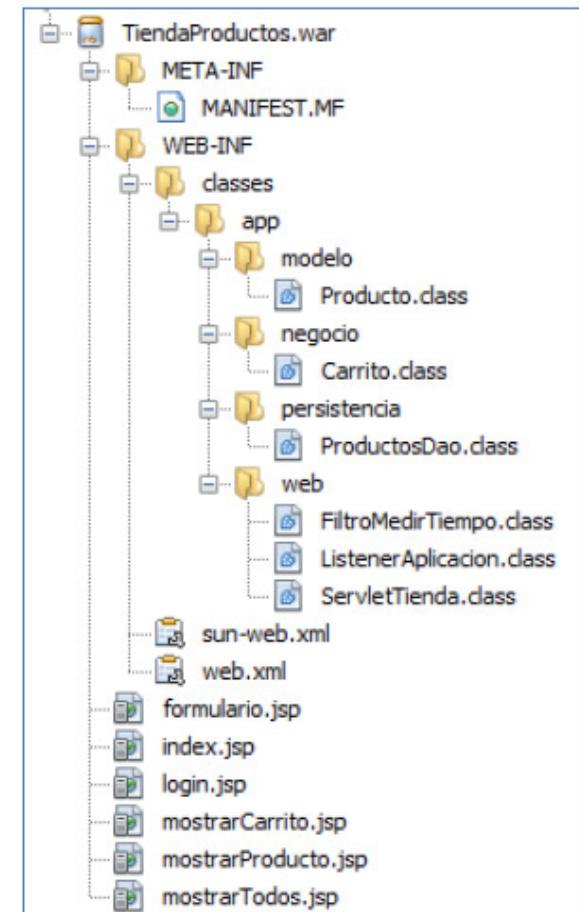
- **Una aplicación web se despliega en el contenedor Web que todo servidor de aplicaciones compatible con J2EE debe poseer.**
- **Este tipo de aplicaciones se componen de los siguientes recursos:**
 - Servlets; clases java que administra y se ejecutan en el contenedor Web.
 - JSP; páginas web compuestas de HTML, CSS, ... etc y código Java.
 - Clases Java
 - Descriptores de despliegue; archivo xml donde se configura los componentes de la aplicación.

1. INTRODUCCIÓN APLICACIONES WEB

- Para poder ejecutar este tipo de aplicaciones necesitamos los siguientes recursos fundamentales:
- JDK (Java Development Kit): Contiene todas las librerías necesarias y la máquina virtual de Java. Existe una versión para cada S.O. (Windows, Mac o Linux).
- Editor de código Java: Hay dos entornos principales: Netbeans y Eclipse. Facilitan las tareas más habituales y permite ahorrar mucho tiempo..
- Servidor de aplicaciones donde se desplegará la aplicación: Utilizaremos básicamente Tomcat Apache o GlassFish. Es el encargado de traducir nuestro código a HTML, que es lo que recibirá el usuario en su browser. .
- Un navegador web tipo Explorer, Mozilla, Crome, ...etc.

2. ESTRUCTURA DE UN WAR

- Una aplicación web se empaqueta en un modulo con extensión .war.
- Este modulo se puede desplegar directamente en el contenedor web.
- Da igual el entorno de desarrollo que utilicemos ya que la estructura de un war forma parte de la especificación J2EE y esto marca un estándar por lo cual siempre será la misma.
- Podemos ver esta estructura en la siguiente figura:



2. ESTRUCTURA DE UN WAR

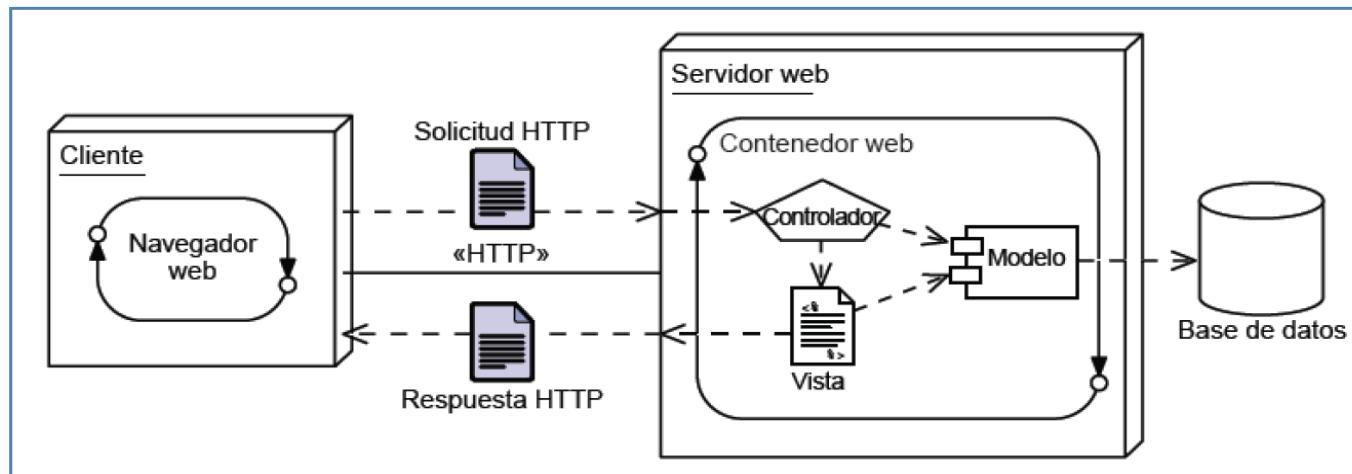
- **Carpeta META-INF;** contiene el archivo de manifiesto que utilizamos para almacenar información sobre la aplicación, control de versiones, autor, ...etc.
- **Carpeta WEB-INF;** Esta carpeta contiene:
 - los descriptores de despliegue **web.xml** y **sun-web.xml**; el primero de ellos es un estándar de J2EE por lo cual su nombre no debe variar. El segundo toma su nombre del servidor de aplicaciones que estemos utilizando.
 - carpeta **clases**; aquí encontramos todas las clases java compiladas (.class) guardando la misma estructura que los paquetes definidos.
- **Contenido web;** en esta ubicación tenemos todo el contenido web: paginas jsp, html, hojas de estilo css, archivos javascript, imagenes, ...etc.

3. PATRON MVC

Una aplicación web diseñada con la arquitectura del patrón MVC se caracteriza por lo siguiente:

- Un servlet actúa como controlador, que verifica los datos recibidos, actualiza el modelo con dichos datos y selecciona la próxima vista como respuesta.
- Una página de JSP actúa como vista. En ella se representa la respuesta HTML, recuperando los datos del modelo necesarios para generar la respuesta, y se proporcionan formularios HTML que permiten la interacción del usuario.
- Las clases Java actúan como modelo, que implementa la lógica de negocio de la aplicación web.

3. PATRON MVC

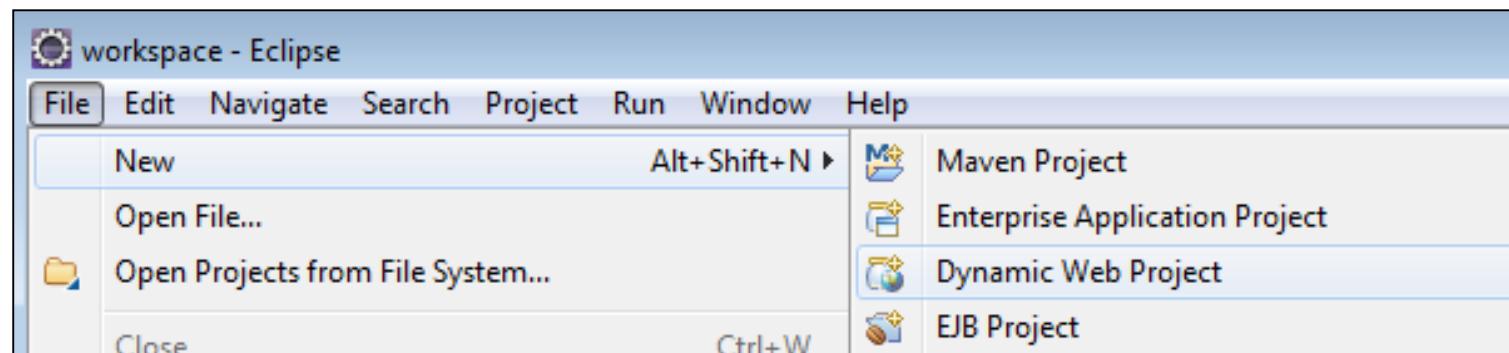


RECUERDA QUE...

- Un modulo war tiene una determinada estructura que la marca el estándar JEE.
- El patrón MVC en aplicaciones web define los siguientes componentes: El servlet actúa de controlador, las páginas jsp como vistas y las clase java como modelo.

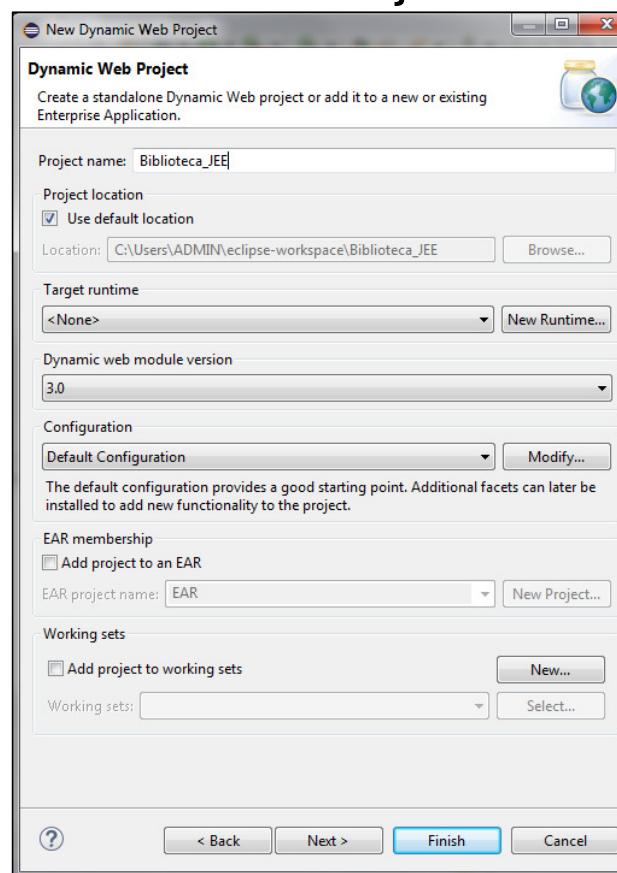
4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 1. Accedemos a File/New y seleccionamos Dynamic Web Project:



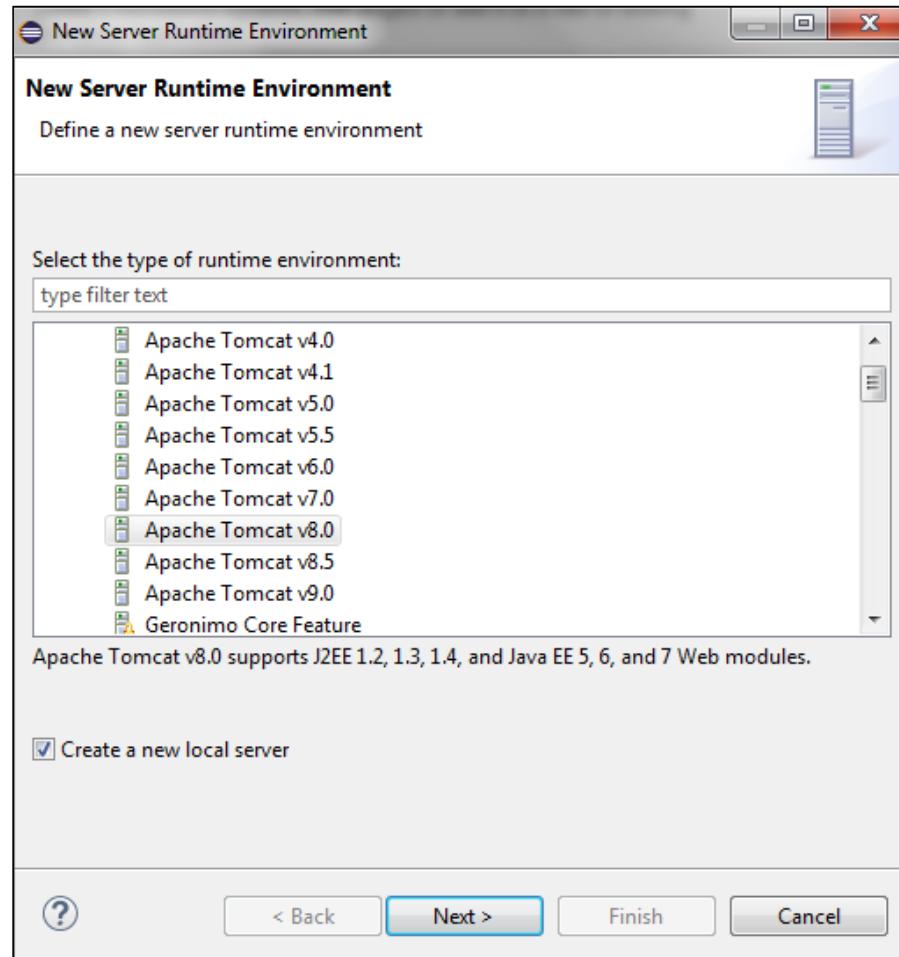
4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 2. Indicamos el nombre del proyecto y hacemos click en el botón New RunTime para seleccionar el servidor de ejecución:



4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 3. Indicaremos el servidor que queremos para nuestra aplicación:



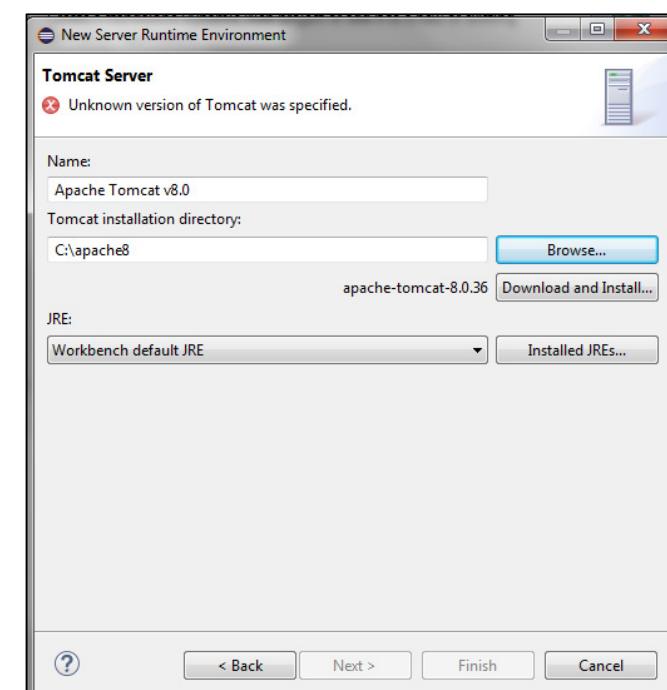
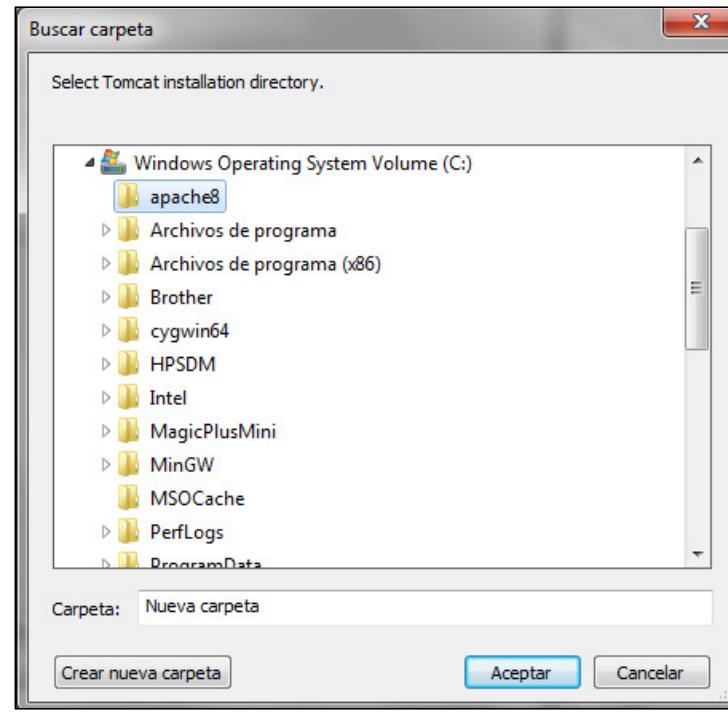
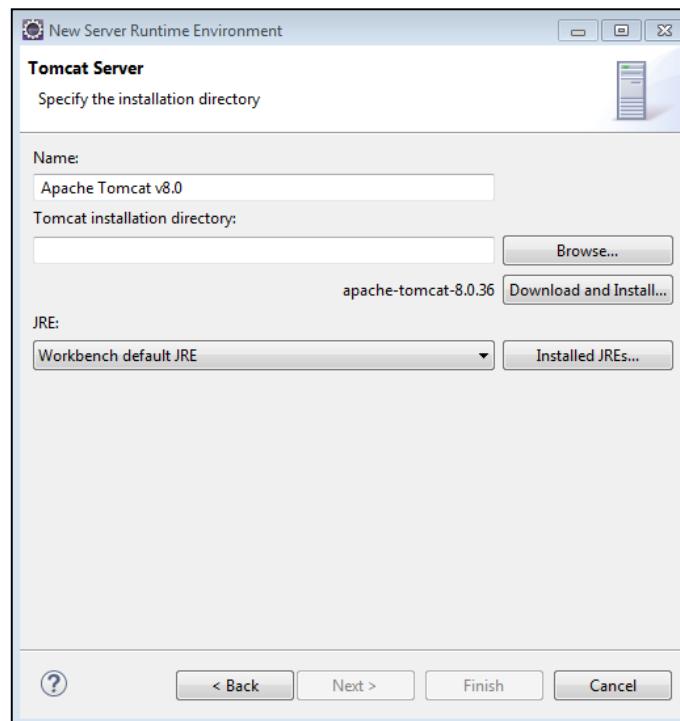
Podemos apuntar a los servidores instalados con Netbeans:

- Tomcat (en C:\Program Files\Apache Software Foundation\Apache Tomcat 8.0.27)
- Glassfish (en C:\Program Files\glassfish-4.1.1)

O realizar una nueva instalación de un servidor, por ejemplo el Apache Tomcat v8.0, que se puede instalar desde eclipse (las versiones 8.5 y 9.0 se deben de instalar por separado)

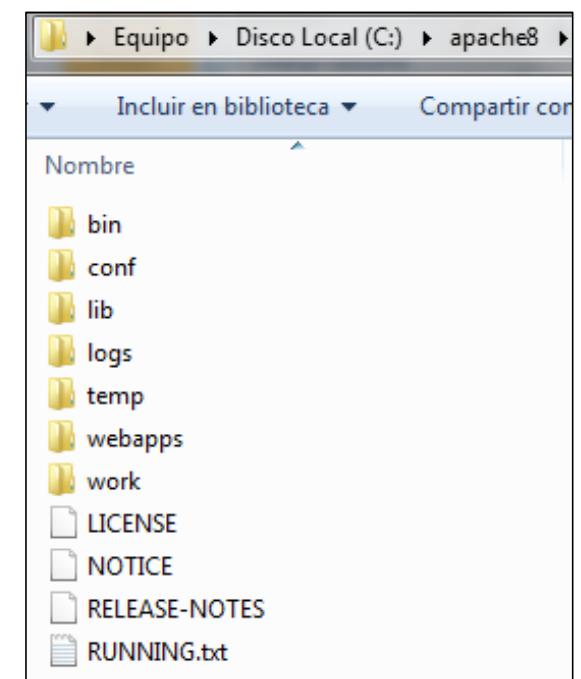
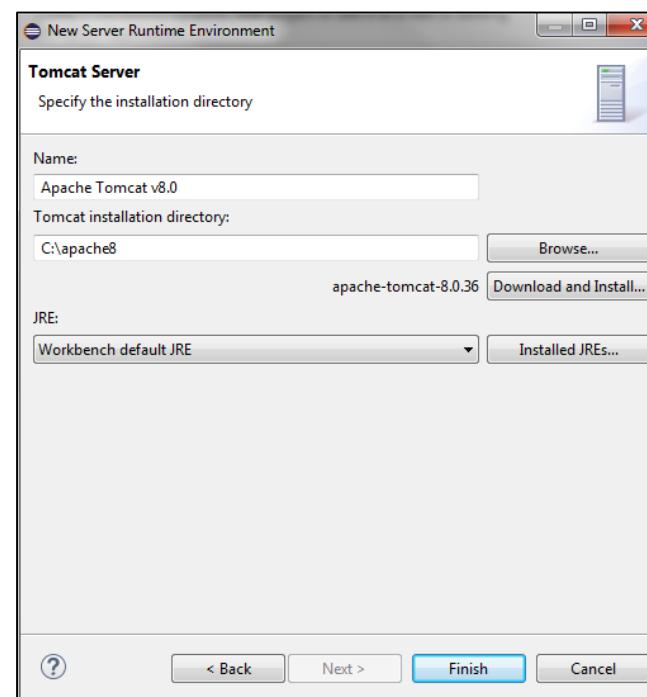
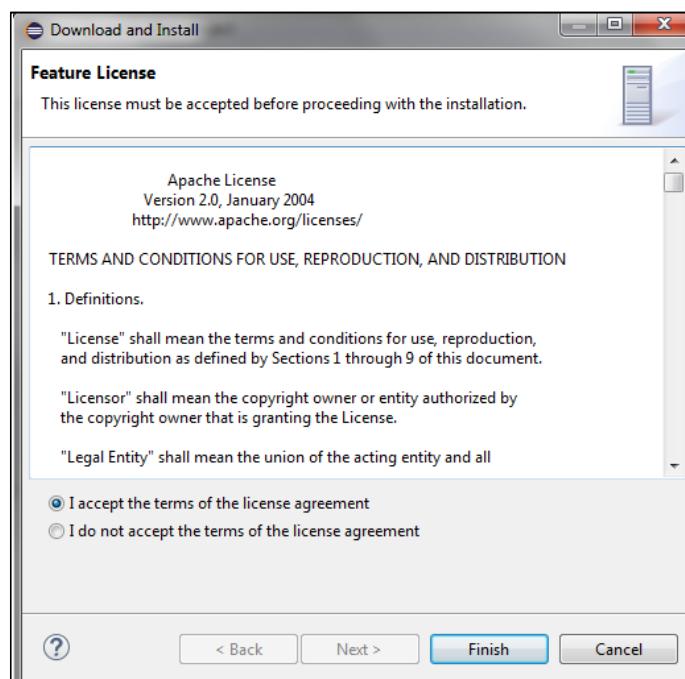
4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 4. Indicamos el path del servidor Tomcat, en la unidad c, donde creamos la carpeta “apache8”:



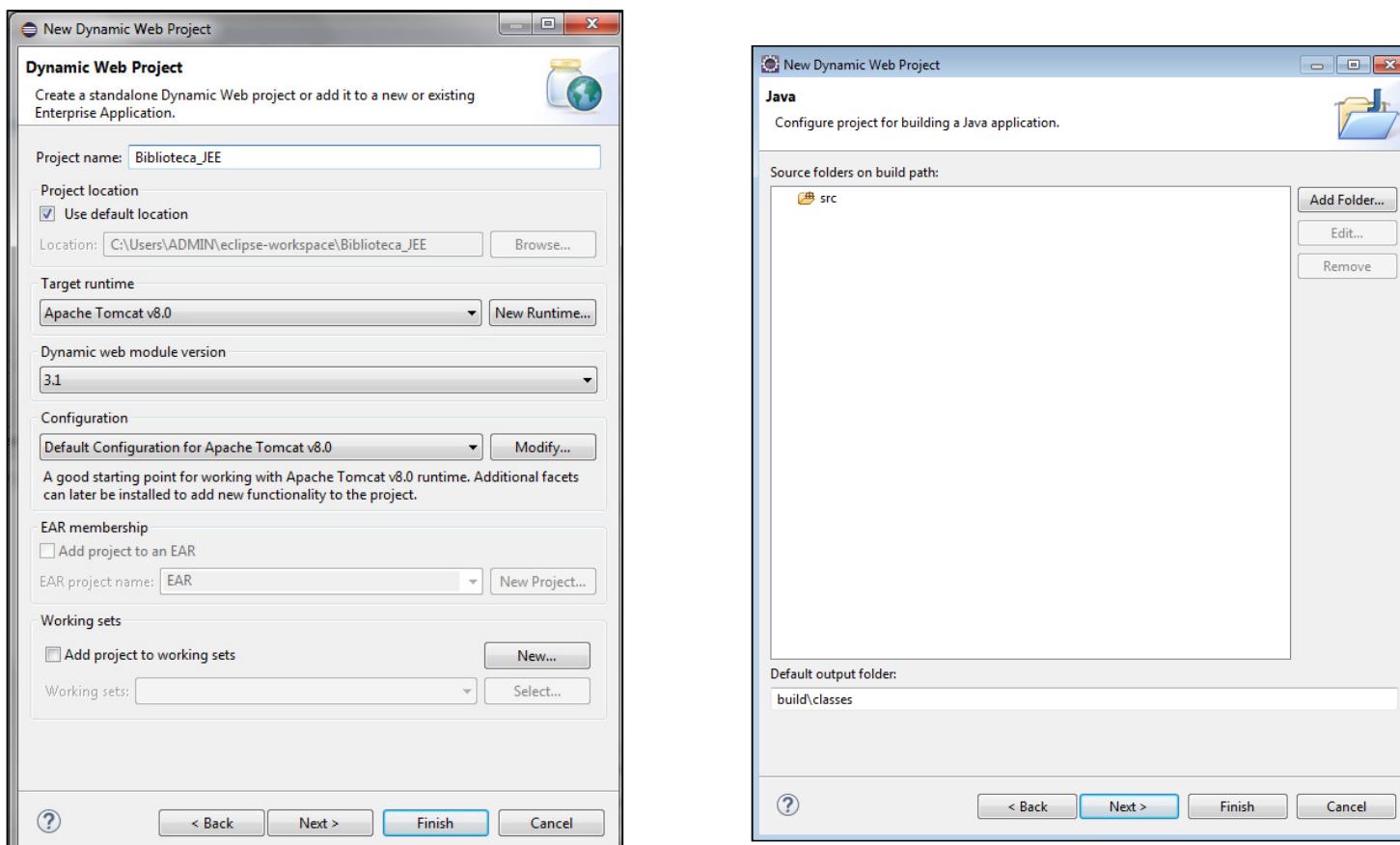
4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 5. Una vez indicada la carpeta donde vamos a instalar el servidor Tomcat v8.0, hacemos click en el botón “Download and Install”. Al final podemos ver que se ha activado el botón Finish y que en la carpeta se han instalado los ficheros del servidor:



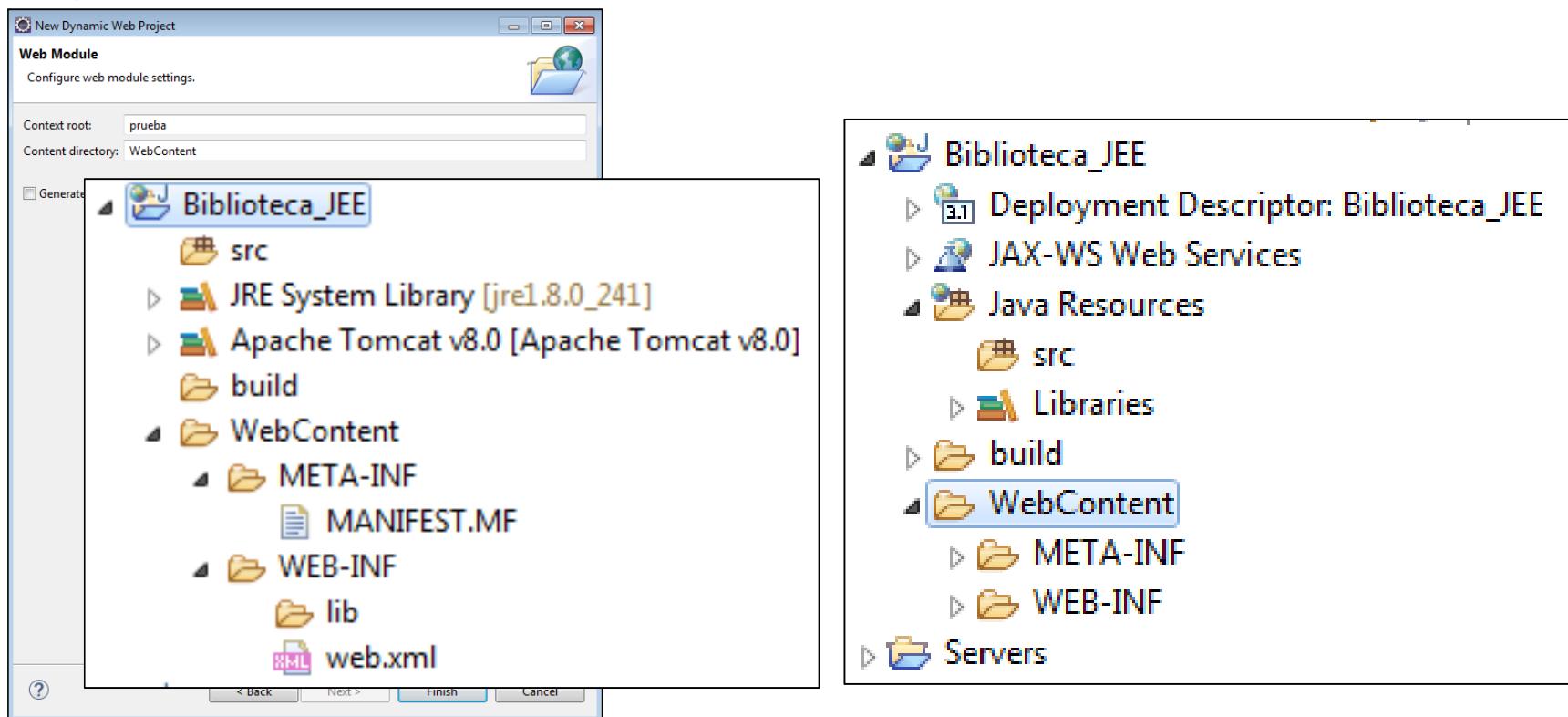
4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 6. Una vez indicado el servidor Tomcat que utilizaremos, seguimos adelante:



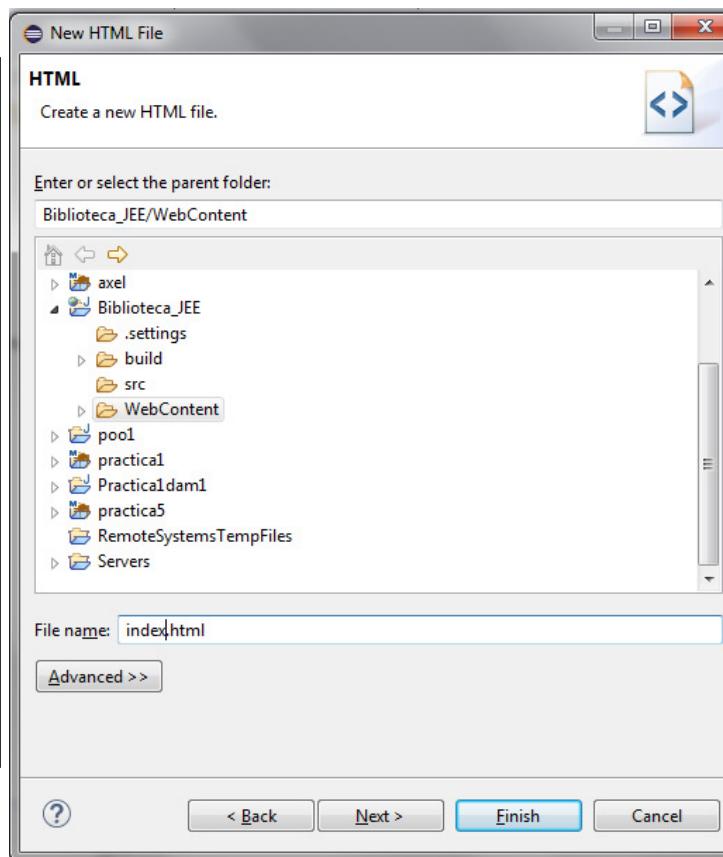
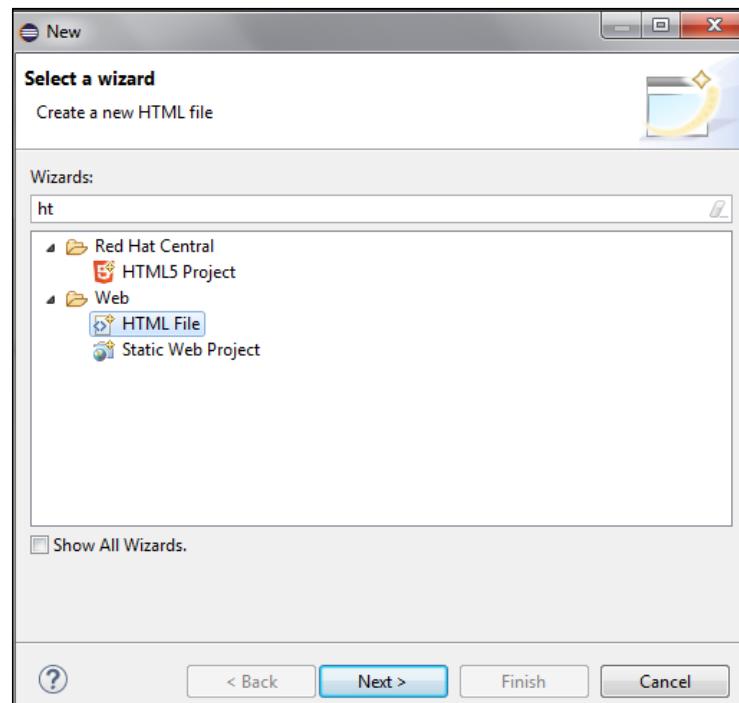
4. CREACIÓN PROYECTO J2EE EN ECLIPSE

Paso 7. Indicamos que genera el fichero descriptor web.xml y finalizamos la creación del proyecto:



5. APLICACION BIBLIOTECA

Paso 1. Para realizar la ejecución del proyecto, debemos crear un fichero **index.html** en Webcontent (eclipse por defecto no inserta ningún fichero):

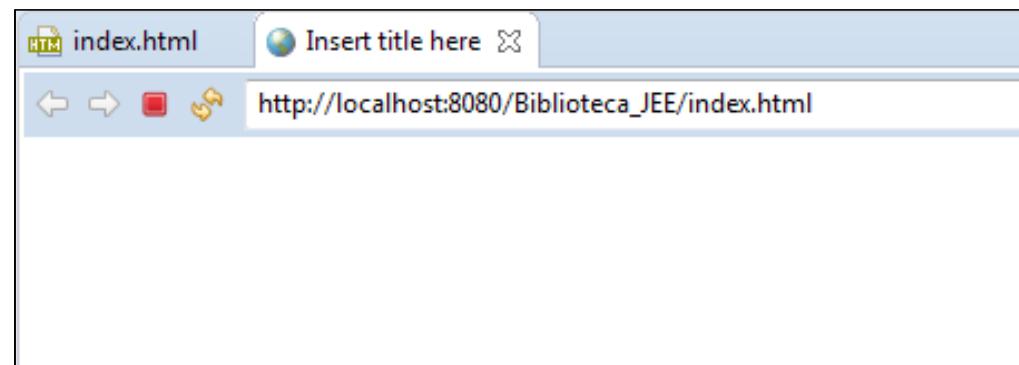
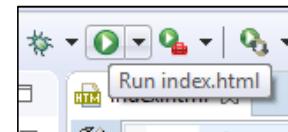
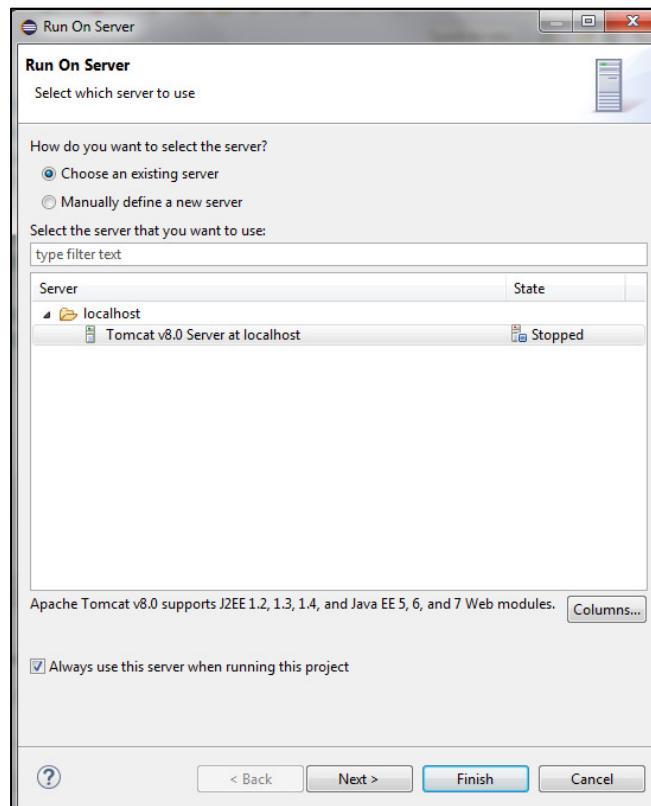


This screenshot shows the Eclipse code editor with the file 'index.html' open. The code is displayed in a syntax-highlighted format:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="ISO-8859-1">
5 <title>Insert title here</title>
6 </head>
7 <body>
8
9 </body>
10 </html>
```

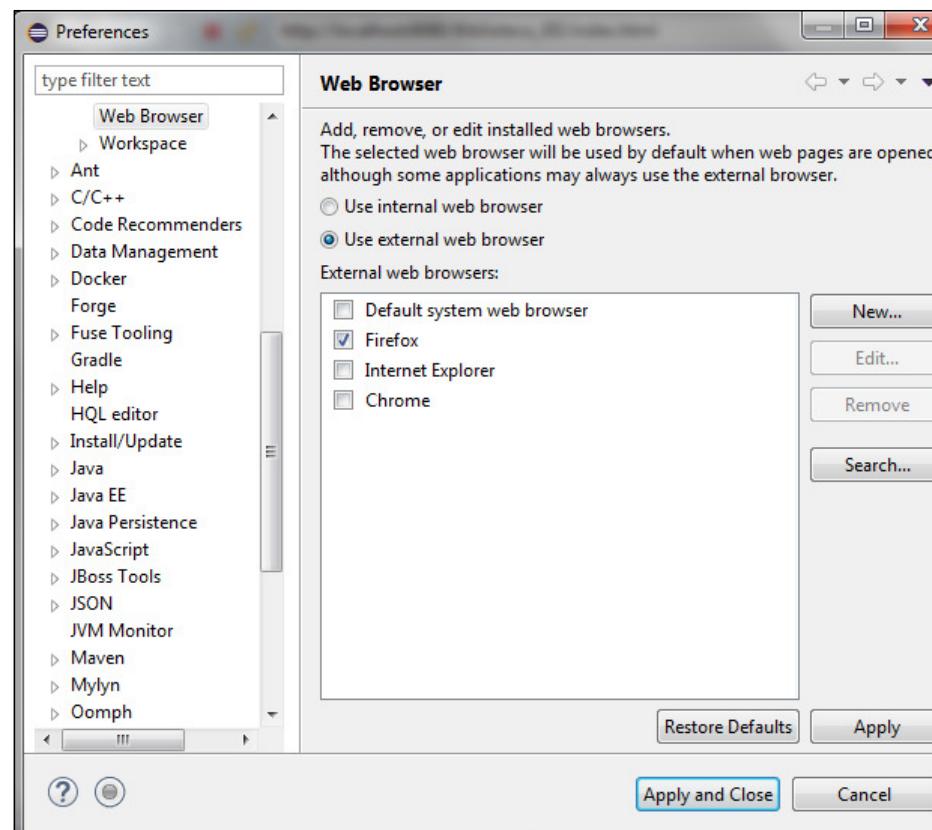
5. APPLICACION BIBLIOTECA

Paso 2. Para ejecutar el proyecto, hacer click en el botón “Run” de eclipse. Nos pide qué servidor usará la aplicación. Se recomienda hacer click en el checkbox “Always use this server...” para usar siempre el mismo servidor.



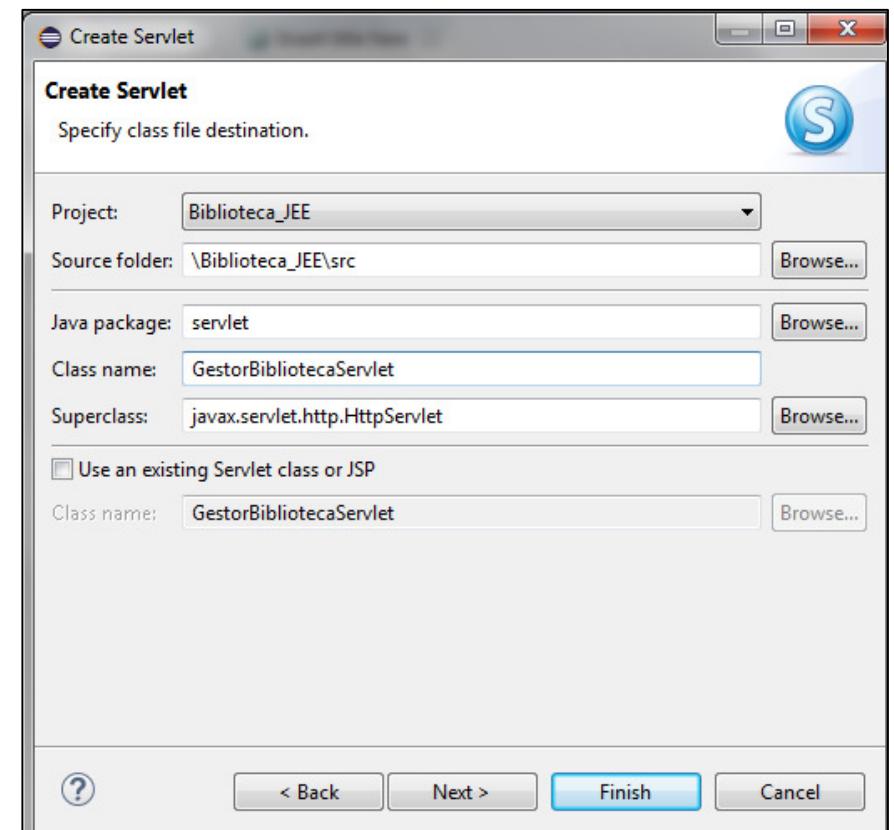
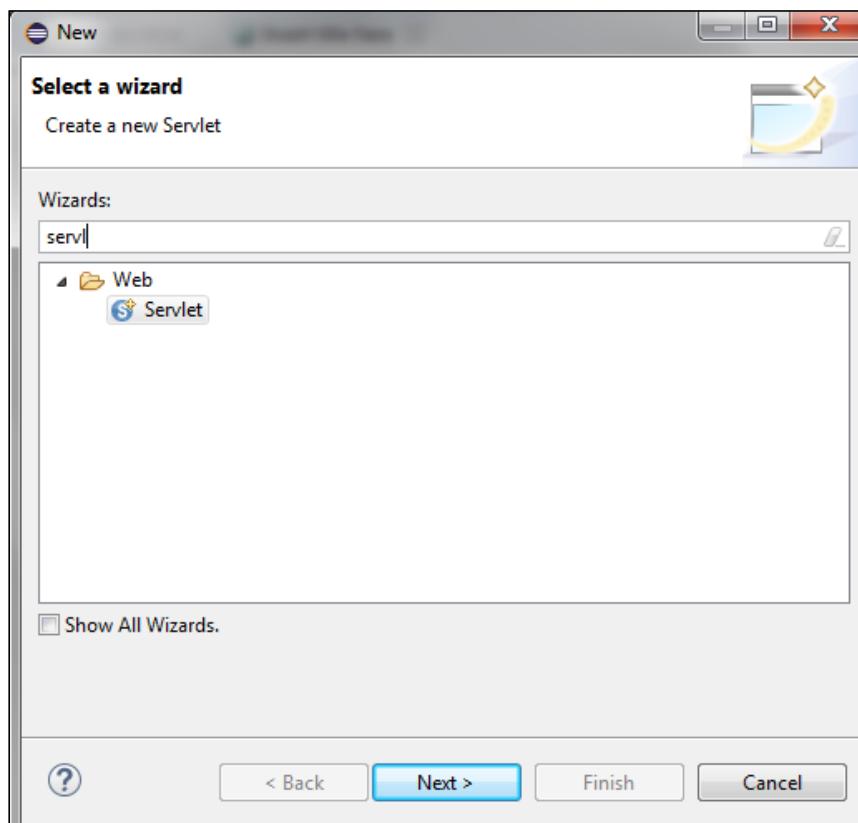
5. APPLICACION BIBLIOTECA

Paso 3. Para ejecutar el proyecto en un navegador, debemos ir a Window - Preferences - General - Web Browser.



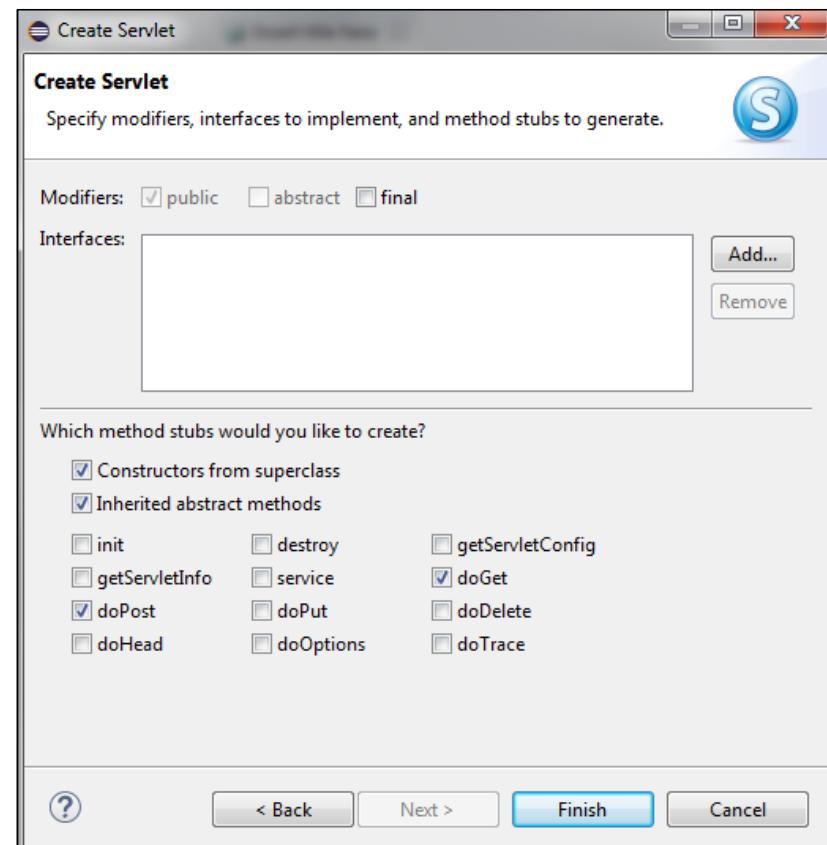
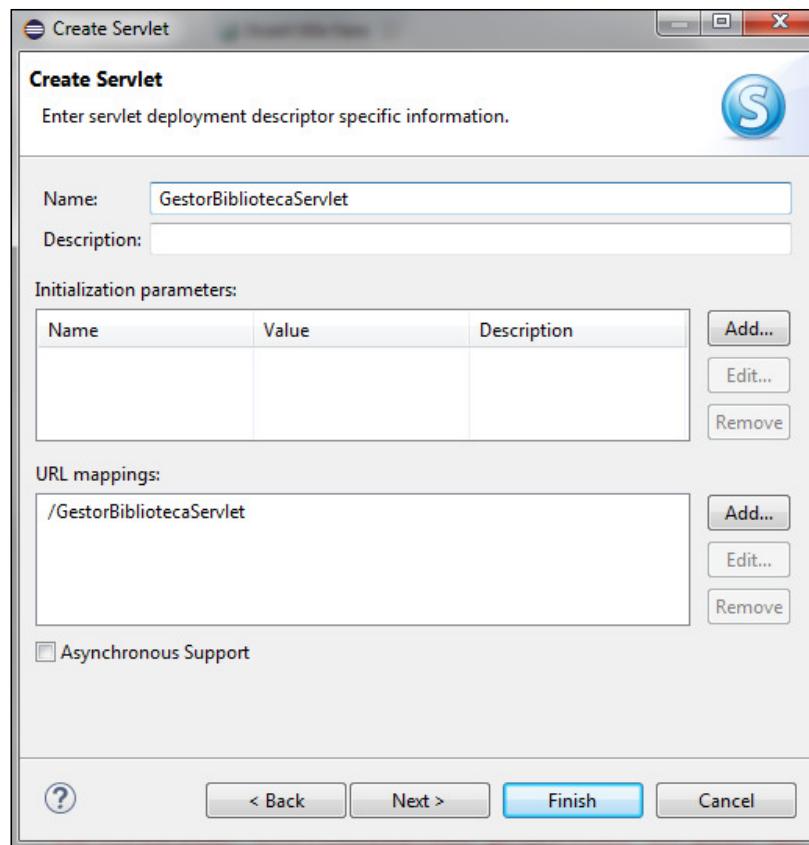
5. APLICACION BIBLIOTECA

Paso 4. Crea un Servlet al que llamaremos “GestorBibliotecaServlet”, dentro del package servlet.



5. APLICACION BIBLIOTECA

Paso 5. Se dejan los parámetros por defecto del servlet “GestorBibliotecaServlet”



5. APPLICACION BIBLIOTECA

Paso 6. Finalmente nuestro primer servlet estará ubicado de la siguiente forma dentro del proyecto JEE y contendrá el siguiente código:

The screenshot shows a Java-based IDE interface with three main components:

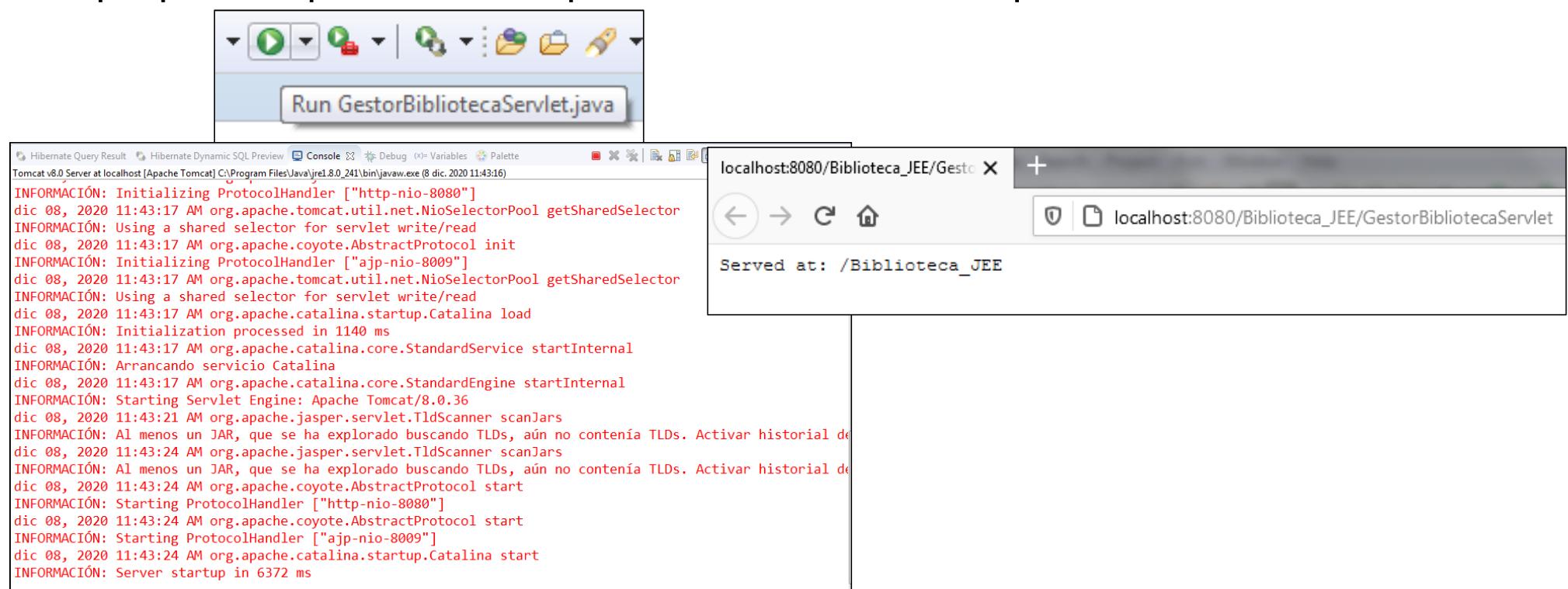
- Project Explorer (Left):** Shows the project structure for "Biblioteca_JEE". It includes a "src" folder containing a "servlet" package with a file named "GestorBibliotecaServlet.java". Other project components shown are "JRE System Library [jre1.8.0_241]", "Apache Tomcat v8.0 [Apache Tomcat v8.0]", "build", "WebContent" (with "META-INF" and "WEB-INF" subfolders), and files "index.html", "lib", and "web.xml".
- Deployment Descriptor (Bottom Left):** Shows the deployment descriptor for "Biblioteca_JEE" with sections for "Deployment Descriptor: Biblioteca_JEE", "JAX-WS Web Services", "Java Resources" (containing the "src" folder and "GestorBibliotecaServlet.java"), "Libraries", "build", "WebContent" (with "META-INF" and "WEB-INF" subfolders), and "index.html".
- Code Editor (Right):** Displays the source code for "GestorBibliotecaServlet.java". The code is as follows:

```
1 package servlet;
2
3 import java.io.IOException;
4
5 /**
6  * Servlet implementation class GestorBibliotecaServlet
7  */
8 @WebServlet("/GestorBibliotecaServlet")
9 public class GestorBibliotecaServlet extends HttpServlet {
10     private static final long serialVersionUID = 1L;
11
12     /**
13      * @see HttpServlet#HttpServlet()
14     */
15     public GestorBibliotecaServlet() {
16         super();
17         // TODO Auto-generated constructor stub
18     }
19
20     /**
21      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
22     */
23     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
24         // TODO Auto-generated method stub
25         response.getWriter().append("Served at: ").append(request.getContextPath());
26     }
27
28     /**
29      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
30     */
31     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
32         // TODO Auto-generated method stub
33         doGet(request, response);
34     }
35 }
36
37
38
39
40
41 }
```

5. APPLICACION BIBLIOTECA

Paso 7. Ejecuta la aplicación web e inicia el servidor web. Desde un navegador pon la siguiente url: http://localhost:8080/Biblioteca_JEE/GestorBibliotecaServlet

Podemos comprobar que el servlet queda configurado como un servicio web que está preparado para atender peticiones a través del protocolo HTTP.



5. APPLICACION BIBLIOTECA

Paso 8. Modifica la respuesta del servlet GestorBibliotecaServlet para que pueda dar mensajes diferentes en función del tipo de petición:

- “Hola Mundo POST!!” si recibe una petición POST
- “Hola Mundo GET!!” si recibe una petición GET.

Sigue el modelo que utiliza Eclipse en la creación por defecto de la plantilla de un servlet, el cual tiene respuestas independientes para get (doGet) y post (doPost).

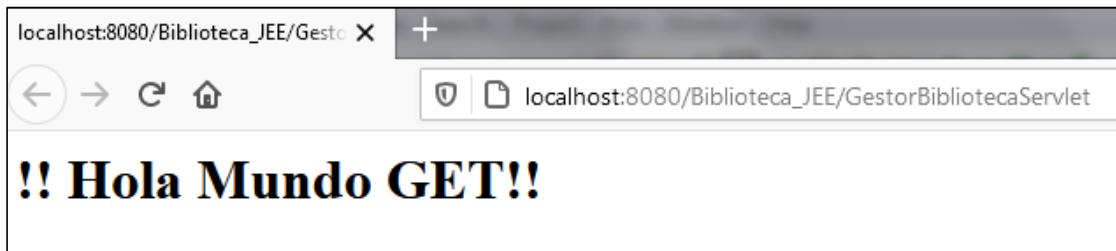
```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.getWriter().append("<h1> !! Hola Mundo GET !! </h1>");
}

protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.getWriter().append("<h1> !! Hola Mundo POST !! </h1>");
}
```

5. APPLICACION BIBLIOTECA

Paso 9. Inicia la aplicación web y desde un navegador pon la siguiente url:
http://localhost:8080/Biblioteca_J2EE/GestorBibliotecaServlet

¿Qué mensaje da el servlet? Indica dos sistemas que nos permitirían obtener los dos tipos de mensajes: GET y POST.

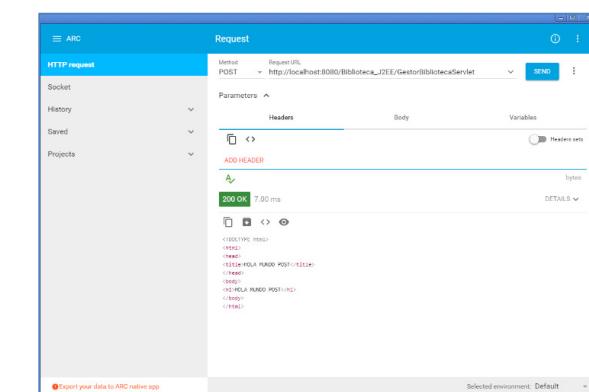
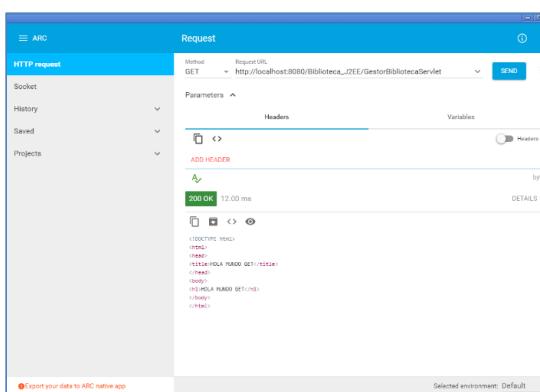
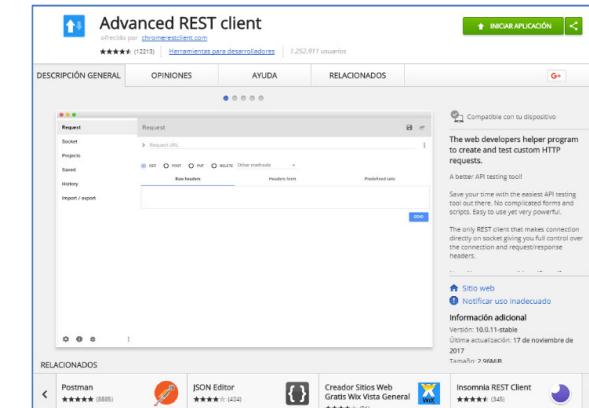
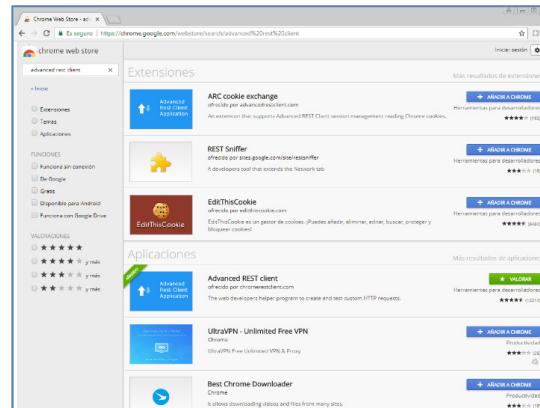


Todas las llamadas realizadas directamente con una url desde un navegador producen un mensaje tipo GET hacia el servlet.

Para conseguir realizar peticiones HTTP de los dos tipos, tanto GET como POST, se deben de hacer con formularios HTML o con algún plugin de un navegador.

5. APLICACION BIBLIOTECA

Paso 10. Podemos usar el plugin Advanced Rest Client para el navegador Chrome con el objetivo de generar peticiones tanto GET como POST gráficamente hacia nuestro servlet:



5. APPLICACION BIBLIOTECA

Paso 11. Utiliza la página web index.html para crear un formulario que contenga un único botón de submit. Este formulario debe llamar al servlet GestorBibliotecaServlet a través del método GET. Ejecuta la aplicación y llama al servlet desde el formulario.

The screenshot illustrates the development and deployment process for a Java Enterprise Edition (JEE) application named "Biblioteca_JEE".

Left Panel: An IDE interface showing the code for `index.html`. The code defines an HTML page with a title "Inicio Aplicación" and a single `<form>` element. This form uses the `GET` method and points to the URL `GestorBibliotecaServlet`. It contains a single `<input type="submit" value="Enviar Consulta">` button.

Middle Panel: A browser window titled "Inicio Aplicación" displaying the text "Método GET" and a "Enviar Consulta" button. The address bar shows the URL `localhost:8080/Biblioteca_JEE/index.html`.

Bottom Panel: Another browser window titled "localhost:8080/Biblioteca_JEE/Gesto" displaying the text "!! Hola Mundo GET !!". The address bar shows the URL `localhost:8080/Biblioteca_JEE/GestorBibliotecaServlet?`

```
index.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="ISO-8859-1">
5     <title>Inicio Aplicación</title>
6   </head>
7   <body>
8     <h1> Método GET</h1>
9     <form method="GET" action="GestorBibliotecaServlet">
10       <input type="submit" value="Enviar Consulta">
11     </form>
12   </body>
13 </html>
```

5. APPLICACION BIBLIOTECA

Paso 12. Agrega otro formulario en el archivo index.html de manera que llame al servlet GestorBibliotecaServlet a través del método POST. Ejecuta la aplicación y mira el mensaje de salida que da el servlet.

The screenshot illustrates the development process for a web application. On the left, a code editor displays the file `index.html` with the following content:

```
index.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="ISO-8859-1">
5     <title>Inicio Aplicación</title>
6   </head>
7   <body>
8     <h1> Método GET</h1>
9     <form method="GET" action="GestorBibliotecaServlet">
10       <input type=submit value="Enviar Consulta">
11     </form>
12     <h1> Método POST</h1>
13     <form method="POST" action="GestorBibliotecaServlet">
14       <input type=submit value="Enviar Consulta">
15     </form>
16
17   </body>
18 </html>
```

On the right, two browser windows are shown. The top window, titled "Inicio Aplicación", displays the page content for the GET method. It contains the heading "Método GET" and a "Enviar Consulta" button. The bottom window, also titled "Inicio Aplicación" and showing the URL `localhost:8080/Biblioteca_JEE/index.html`, displays the page content for the POST method. It contains the heading "Método POST" and a "Enviar Consulta" button. Both windows show the same basic structure with different headings and button labels. The bottom window also shows the URL `localhost:8080/Biblioteca_JEE/GestorBibliotecaServlet` in its address bar, indicating the target of the POST form submission. The overall message is "!! Hola Mundo POST !!".

5. APLICACION BIBLIOTECA

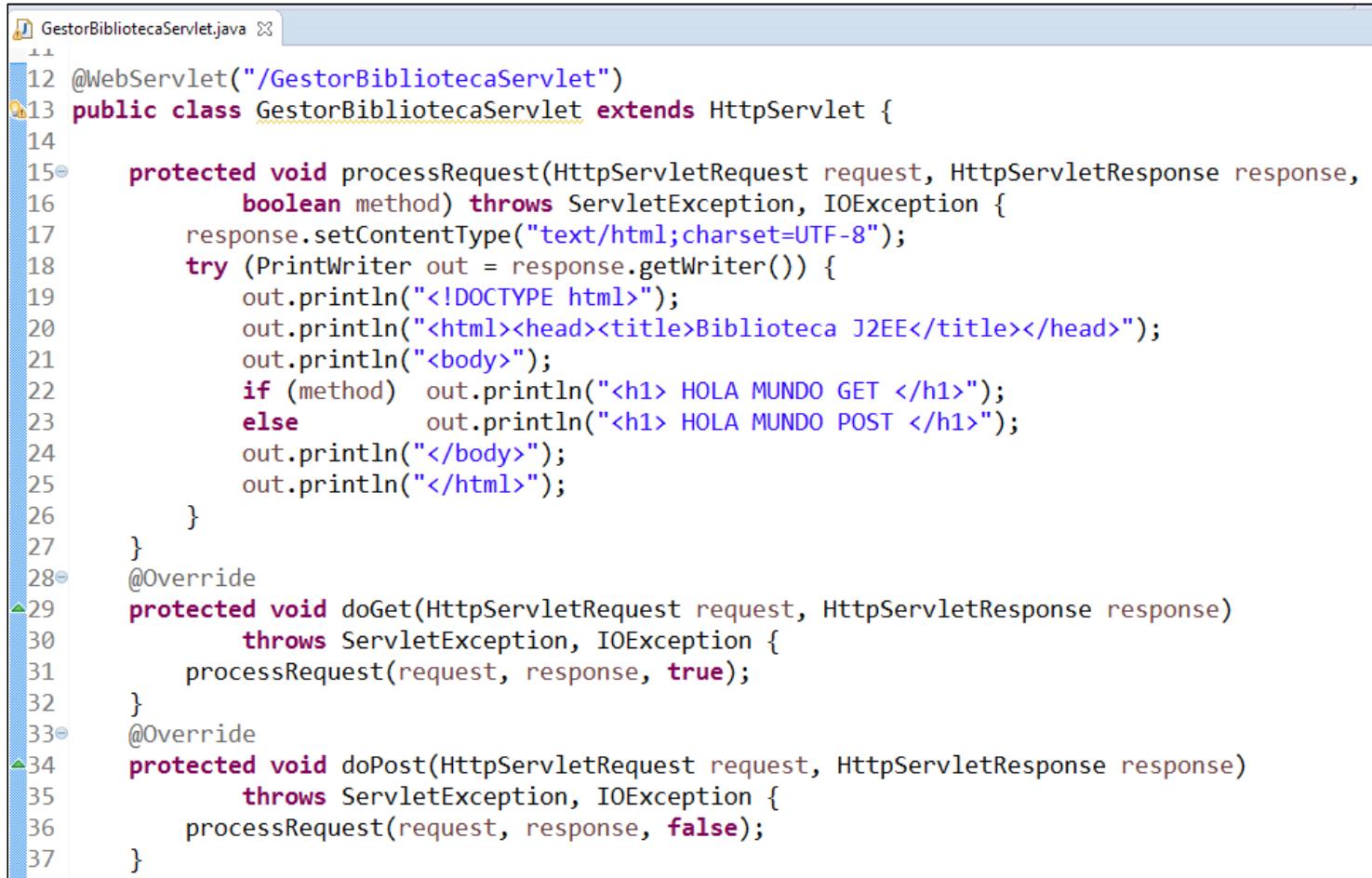
Paso 13. Unifica los métodos doGet y doPost con la función processRequest de la figura, de manera que el servlet GestorBibliotecaServlet siga haciendo lo mismo. Es el sistema utilizado por defecto en los servlets de Netbeans.

```
public class GestorBibliotecaServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Servlet NewServlet</title></head>");
            out.println("<body>");
            out.println("<h1>Servlet NewServlet at " + request.getContextPath() + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
    }
}
```

Idea: Para lograrlo se puede incluir un nuevo parámetro booleano dentro de la función processRequest.

5. APLICACION BIBLIOTECA

Paso 14. Uso de la función processRequest, con un parámetro booleano:



```
12 @WebServlet("/GestorBibliotecaServlet")
13 public class GestorBibliotecaServlet extends HttpServlet {
14
15     protected void processRequest(HttpServletRequest request, HttpServletResponse response,
16             boolean method) throws ServletException, IOException {
17         response.setContentType("text/html;charset=UTF-8");
18         try (PrintWriter out = response.getWriter()) {
19             out.println("<!DOCTYPE html>");
20             out.println("<html><head><title>Biblioteca J2EE</title></head>");
21             out.println("<body>");
22             if (method) out.println("<h1> HOLA MUNDO GET </h1>");
23             else        out.println("<h1> HOLA MUNDO POST </h1>");
24             out.println("</body>");
25             out.println("</html>");
26         }
27     }
28     @Override
29     protected void doGet(HttpServletRequest request, HttpServletResponse response)
30             throws ServletException, IOException {
31         processRequest(request, response, true);
32     }
33     @Override
34     protected void doPost(HttpServletRequest request, HttpServletResponse response)
35             throws ServletException, IOException {
36         processRequest(request, response, false);
37     }
}
```

5. APLICACION BIBLIOTECA

Paso 15. Realiza la misma operativa que en el ejercicio anterior, usando la función processRequest, pero ahora utiliza una variable privada booleana de la clase GestorBibliotecaServlet para diferenciar entre llamadas GET y POST.

```
GestorBibliotecaServlet.java X
12 @WebServlet("/GestorBibliotecaServlet")
13 public class GestorBibliotecaServlet extends HttpServlet {
14
15     private boolean method = false;
16     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
17         throws ServletException, IOException {
18         response.setContentType("text/html;charset=UTF-8");
19         try (PrintWriter out = response.getWriter()) {
20             out.println("<!DOCTYPE html>");
21             out.println("<html><head><title>Biblioteca J2EE</title></head>");
22             out.println("<body>");
23             if (method) out.println("<h1> HOLA MUNDO GET </");
24             else        out.println("<h1> HOLA MUNDO POST <");
25             out.println("</body>");
26             out.println("</html>");
27         }
28     }
}
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    method = true;
    processRequest(request, response);
}
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    method = false;
    processRequest(request, response);
}
```

5. APLICACION BIBLIOTECA

Paso 16. Divide la función processRequest en dos partes, la parte GET y la parte POST, y asigna cada parte a las funciones doGet y doPost, de forma que funcionen de forma independiente y no se tenga que utilizar variables para diferenciar entre ambas llamadas (como estaba originalmente):

```
@WebServlet("/GestorBibliotecaServlet")
public class GestorBibliotecaServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Biblioteca J2EE</title></head>");
            out.println("<body>");
            out.println("<h1> HOLA MUNDO GET </h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Biblioteca J2EE</title></head>");
            out.println("<body>");
            out.println("<h1> HOLA MUNDO POST </h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
}
```

5. APLICACION BIBLIOTECA

Paso 17. Modifica el ejercicio para que el usuario reciba el mensaje “Conectado a la BD” sólo la primera vez que acceda al Servlet. Lo haremos con la variable booleana YaIniciado.

```
@WebServlet("/GestorBibliotecaServlet")
public class GestorBibliotecaServlet extends HttpServlet {
    private boolean YaIniciado=false;
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Biblioteca J2EE</title></head>");
            out.println("<body>");
            if (!YaIniciado) {
                YaIniciado = true;
                out.println("<H2>Conectado a la base de datos");
            }
            out.println(" <h1> HOLA MUNDO GET </h1>");
            out.println(" </body>");
            out.println(" </html>");
        }
    }
}
```

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html><head><title>Biblioteca J2EE</title></head>");
        out.println("<body>");
        if (!YaIniciado) {
            YaIniciado = true;
            out.println("<H2>Conectado a la base de datos (POST)</H2>");
        }
        out.println(" <h1> HOLA MUNDO POST </h1>");
        out.println(" </body>");
        out.println(" </html>");
    }
}
```

5. APLICACION BIBLIOTECA

Paso 18. Nos interesa saber quién se conecta a nuestra biblioteca. Para ello, pasaremos el usuario y su contraseña al servlet para que ésta valide si tenemos permisos para acceder. Añade 2 campos a los formularios de la página web index.html que se llamarán usuario y password.



The screenshot shows a code editor window titled "index.html". The code is written in HTML and defines two forms: one for GET method and one for POST method. Both forms have fields for "Usuario" and "Password". The code is numbered from 1 to 21.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="ISO-8859-1">
5     <title>Inicio Aplicación</title>
6   </head>
7   <body>
8     <h1> Método GET</h1>
9     <form method="GET" action="GestorBibliotecaServlet">
10       Usuario: <input type="text" name="usuario"><br><br>
11       Password: <input type="password" name="password"><br><br>
12       <input type=submit value="Enviar Consulta">
13     </form>
14     <h1> Método POST</h1>
15     <form method="POST" action="GestorBibliotecaServlet">
16       Usuario: <input type="text" name="usuario"><br><br>
17       Password: <input type="password" name="password"><br><br>
18       <input type=submit value="Enviar Consulta">
19     </form>
20   </body>
21 </html>
```

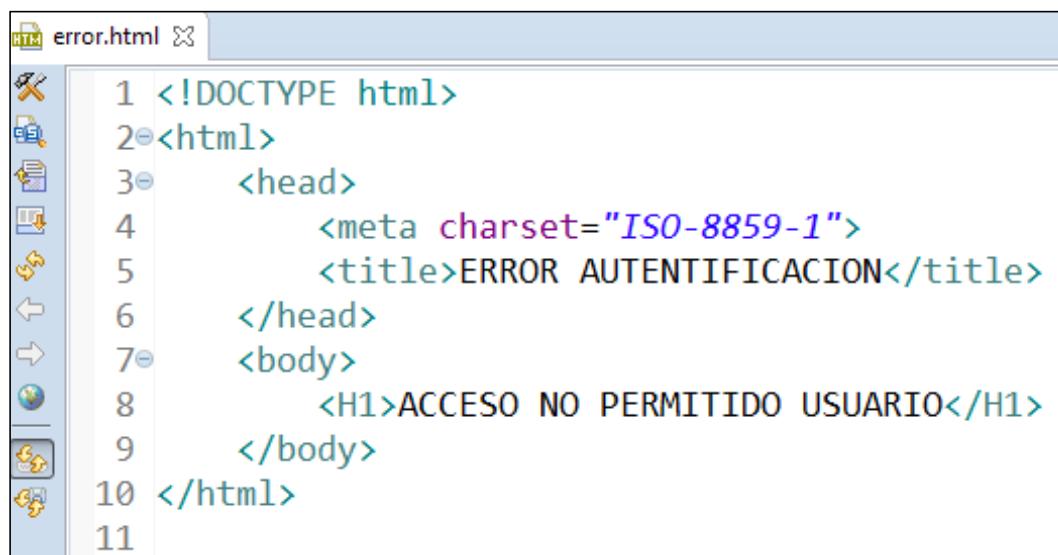
5. APLICACION BIBLIOTECA

Paso 19. El servlet GestorBibliotecaServlet tiene que recoger los dos parámetros y si son correctos contestar con “Bienvenido [usuario]”. En caso contrario debe indicar “Acceso no permitido”.

```
@WebServlet("/GestorBibliotecaServlet")
public class GestorBibliotecaServlet extends HttpServlet {
    private boolean YaIniciado=false;
    @Override
    protected void doGet(HttpServletRequest request,
                         throws ServletException, IOException {
        String usuario=request.getParameter("usuario");
        String password=request.getParameter("password");
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Biblioteca J2EE</title></head>");
            out.println("<body>");
            if (usuario.equals("pepe") && password.equals("pepe")) {
                if (!YaIniciado) {
                    YaIniciado = true;
                    out.println("<H2>Conectado a la base de datos (POST)</H2>");
                }
                out.println("<h1> HOLA MUNDO GET </h1>");
            }else out.println("<h1> ACCESO NO PERMITIDO USUARIO "+ usuario + " </h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String usuario=request.getParameter("usuario");
        String password=request.getParameter("password");
        response.setContentType("text/html; charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Biblioteca J2EE</title></head>");
            out.println("<body>");
            if (usuario.equals("pepe") && password.equals("pepe")) {
                if (!YaIniciado) {
                    YaIniciado = true;
                    out.println("<H2>Conectado a la base de datos (POST)</H2>");
                }
                out.println("<h1> HOLA MUNDO POST </h1>");
            }else out.println("<h1> ACCESO NO PERMITIDO USUARIO "+ usuario + " </h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
}
```

5. APLICACION BIBLIOTECA

Paso 20. Crea el fichero error.html, el cual será redireccionado por el servlet cuando la identificación sea incorrecta. Por lo tanto el servlet ya no dibujará directamente el resultado de la identificación errónea, sino que mostrará un fichero html donde se muestre el error.



The screenshot shows a code editor window titled "error.html". The code is a simple HTML document with the following structure:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="ISO-8859-1">
5     <title>ERROR AUTENTIFICACION</title>
6   </head>
7   <body>
8     <H1>ACCESO NO PERMITIDO USUARIO</H1>
9   </body>
10 </html>
11
```

5. APLICACION BIBLIOTECA

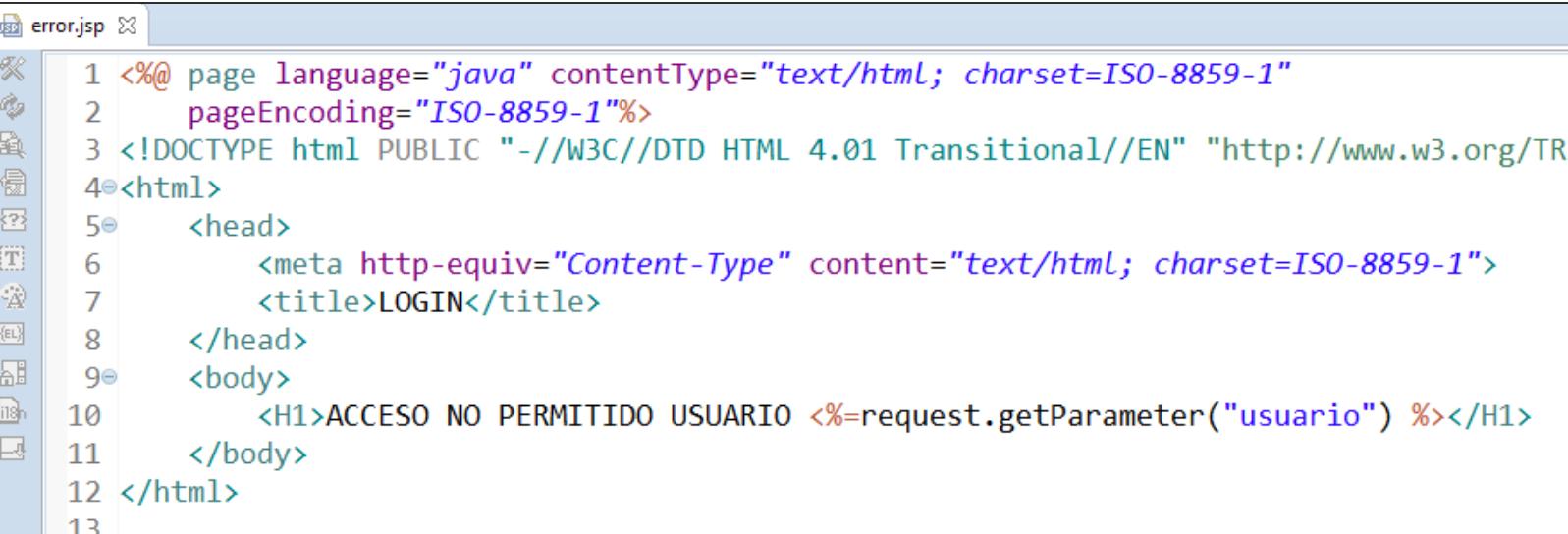
Paso 21. Utiliza la función response.sendRedirect("") en el servlet GestorBibliotecaServlet para dirigir hacia el fichero error.html.

El servlet no
puede pasar un
parámetro a un
fichero html

```
@WebServlet("/GestorBibliotecaServlet")
public class GestorBibliotecaServlet extends HttpServlet {
    private boolean YaIniciado=false;
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String usuario=request.getParameter("usuario");
        String password=request.getParameter("password");
        if (usuario.equals("pepe") && password.equals("pepe")) {
            response.setContentType("text/html;charset=UTF-8");
            try (PrintWriter out = response.getWriter()) {
                out.println("<!DOCTYPE html>");
                out.println("<html><head><title>Biblioteca J2EE</title></head>");
                out.println("<body>");
                if (!YaIniciado) {
                    YaIniciado = true;
                    out.println("<H2>Conectado a la base de datos (GET)</H2>");
                }
                out.println("<h1> HOLA MUNDO GET </h1>");
                out.println("</body>");
                out.println("</html>");
            }
        }else response.sendRedirect("error.html");
    }
}
```

5. APLICACION BIBLIOTECA

Paso 22. Crea el fichero error.jsp, que sustituirá a error.html en el redireccionado por parte del servlet GestorBibliotecaServlet cuando la identificación sea incorrecta. Le pasaremos el parámetro usuario para que lo muestre por pantalla, cosa que con un fichero html no se podía hacer.



```
error.jsp
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
4<html>
5<head>
6     <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7     <title>LOGIN</title>
8 </head>
9<body>
10    <H1>ACCESO NO PERMITIDO USUARIO <%=request.getParameter("usuario") %></H1>
11 </body>
12 </html>
13
```

5. APLICACION BIBLIOTECA

Paso 23. En GestorBibliotecaServlet realiza el redireccionado hacia el fichero error.jsp cuando la identificación sea incorrecta, pasándole el parámetro usuario para que lo muestre por pantalla (con el fichero html no se podía hacer)

```
@Override  
protected void doGet(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {  
    String usuario=request.getParameter("usuario");  
    String password=request.getParameter("password");  
    if (usuario.equals("pepe") && password.equals("pepe")) {  
        response.setContentType("text/html;charset=UTF-8");  
        try (PrintWriter out = response.getWriter()) {  
            out.println("<!DOCTYPE html>");  
            out.println("<html><head><title>Biblioteca J2EE</title></head>");  
            out.println("<body>");  
            if (!YaIniciado) {  
                YaIniciado = true;  
                out.println("<H2>Conectado a la base de datos (GET)</H2>");  
            }  
            out.println("<h1> HOLA MUNDO GET </h1>");  
            out.println("</body>");  
            out.println("</html>");  
        }  
    }else response.sendRedirect("error.jsp?usuario="+usuario);  
}
```

5. APLICACION BIBLIOTECA

Paso 24. Modifica el fichero GestorBibliotecaServlet para que redireccione y delegue su funcionalidad en caso de login correcto hacia el archivo bienvenida.jsp. Este archivo debe garantizar exactamente la misma funcionalidad del ejercicio anterior. Utilizar sendRedirect con paso de parámetros.

```
@WebServlet("/GestorBibliotecaServlet")
public class GestorBibliotecaServlet extends HttpServlet {
    private boolean YaIniciado=false;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String u;
        String p;
        if (usuario.equals("pepe") && password.equals("pepe")) {
            boolean iniciado = YaIniciado;
            if (!YaIniciado) YaIniciado = true;
            response.sendRedirect("bienvenida.jsp?usuario=" + usuario + "&iniciado=" + iniciado + "&method=POST");
        } else
            response.sendRedirect("error.html");
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String usuario=request.getParameter("usuario");
        String password=request.getParameter("password");
        if (usuario.equals("pepe") && password.equals("pepe")) {
            boolean iniciado = YaIniciado;
            if (!YaIniciado) YaIniciado = true;
            response.sendRedirect("bienvenida.jsp?usuario=" + usuario + "&iniciado=" + iniciado + "&method=POST");
        } else
            response.sendRedirect("error.html");
    }
}
```

5. APLICACION BIBLIOTECA

Paso 25. El archivo bienvenida.jsp debe de ser capaz de recoger el usuario, la variable que indica si se ha iniciado por primera vez el acceso a la base de datos y el método utilizado GET o POST.

```
bienvenida.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4 <html>
5 <head>
6 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9 <body>
10    <%String usuario=request.getParameter("usuario"); %>
11    <%String iniciado=request.getParameter("iniciado"); %>
12    <%String method=request.getParameter("method"); %>
13    <%if (iniciado.equals("false")) {%
14      <h1>Conectado a la BD</h1>
15    }%
16    <h1>BIENVENIDO USUARIO <%=usuario%> (llamada <%=method%>)</h1>
17
18 </body>
19 </html>
```

The screenshot shows the Java code for bienvenida.jsp in a code editor. The code uses JSP tags to get parameters from the request, build a string using a StringBuilder, and print it to the output. Below the code editor is a browser window displaying the result of the code execution. The URL in the browser is localhost:8080/Biblioteca_JEE/bienvenida.jsp?usuario=pepe&iniciado=true&method=GET. The page content is "BIENVENIDO USUARIO pepe (llamada GET)". There is also a form with a text input field labeled "Seleccion de Libro:" and a button labeled "Consulta Libros".

```
bienvenida.jsp x
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/ht
4 <html>
5 <head>
6 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9 <body>
10   <%
11     String usuario=request.getParameter("usuario");
12     String iniciado=request.getParameter("iniciado");
13     String method=request.getParameter("method");
14     StringBuilder pepe= new StringBuilder();
15     if (iniciado.equals("false")) {
16       pepe.append("<h1>Conectado a la BD</h1>");
17     }
18     pepe.append("<h1>BIENVENIDO USUARIO "+usuario+" (llamada "+ method + "</h1>"); 
19     out.println(pepe.toString());
20   </body>
21 </html>
```

localhost:8080/Biblioteca_JEE/bienvenida.jsp?usuario=pepe&iniciado=true&method=GET

BIENVENIDO USUARIO pepe (llamada GET)

Seleccion de Libro:

Consulta Libros

5. APLICACION BIBLIOTECA

Paso 26. Una vez estemos conectados, nos interesa mostrar una página con un listado de libros disponibles. Para ello, en el archivo de bienvenida.jsp añadiremos un formulario con un campo de texto y un botón, que nos permitirá buscar una lista de libros por título.

Este formulario debe apuntar a un nuevo servlet que llamaremos “ConsultaLibrosServlet”



```
bienvenida.jsp ✘
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.
4@<html>
5@<head>
6 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9@<body>
10    <%String usuario=request.getParameter("usuario"); %>
11    <%String iniciado=request.getParameter("iniciado"); %>
12    <%String method=request.getParameter("method"); %>
13    <%if (iniciado.equals("false")) {%>
14      <h1>Conectado a la BD</h1>
15    <%}%>
16    <h1>BIENVENIDO USUARIO <%=usuario%> (llamada <%=method%>)</h1>
17@    <form method=GET action=ConsultaLibrosServlet>
18      Seleccion de Libro: <input type="text" name="titulo"><br><br>
19      <input type="submit" value="Consulta Libros" name="submit">
20    </form>
21 </body>
22 </html>
23
```

5. APLICACION BIBLIOTECA

Paso 27. El servlet ConsultaLibrosServlet estará preparado para recibir el título solicitado por el usuario, y buscará en el fichero de texto “libros.txt” todas las entradas que contengan dicha cadena de texto. Como respuesta, enviará al navegador el listado de libros coincidentes. El contenido del fichero libros.txt es el siguiente:

Sitúa el fichero libros.txt en el directorio Webcontent de tu proyecto



```
libros.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
00000001;HARRY POTTER Y EL PRISIONERO DE AZKABÁN;J.K ROWINS;SALAMANDRA;26/9/2006 0:00:00;INFANTIL;
00000002;EL GRAN LABERINTO;FERNANDO SABATER PEREZ;ARIEL;26/9/2006 0:00:00;FICCIÓN;
00000003;ROMEO Y JULIETA;WILLIAM SHAKESPEARE;SALAMANDRA;26/9/2006 0:00:00;ROMANTICA;
00000004;LA CARTA ESFERICA;ARTURO PEREZ LOPEZ;SALAMANDRA;29/9/2006 0:00:00;FICCIÓN;
00000005;CODIGO DA VINCI;DAN BROWN;ARIEL;29/9/2006 0:00:00;FICCIÓN;
00000006;MUCHO RUIDO Y POCAS NUECES;WILLIAM SHAKESPEARE;SALAMANDRA;29/9/2006 0:00:00;ROMANTICA;
00000007;PROTOCOLO;JOSE LOPEZ MURILLO;SALAMANDRA;6/9/2006 0:00:00;SOCIAL;
00000008;LINUX;FERNANDO SABATER PEREZ;ARIEL;6/9/2006 0:00:00;INFORMÁTICA;
00000009;EL TUMULTO;H.P LOVERCRAFT;DEBATE;6/9/2006 0:00:00;CIENCIA;
00000010;PERSONAJES MITICOS;RICHARD HOLLIGHAM;DEBATE;7/9/2006 0:00:00;ENTRETENIMIENTO;
00000011;EL TIEMPO;J.K ROWINS;SALAMANDRA;7/9/2006 0:00:00;CIENCIA;
00000012;DIETAS MEDITERRANEAS;ARTURO PEREZ LOPEZ;ARIEL;16/9/2006 0:00:00;ASTRONOMIA;
00000013;ANGELES Y DEMONIOS;DAN BROWN;ARIEL;17/9/2006 0:00:00;FICCIÓN;
00000014;FORTALEZA DIGITAL;DAN BROWN;ARIEL;6/10/2006 0:00:00;FICCIÓN;
00000015;CAPITAN ALATRISTE;ARTURO PEREZ LOPEZ;ALFAGUARA;9/10/2006 0:00:00;FICCIÓN;
00000016;PIEL DE TAMBOR;ARTURO PEREZ LOPEZ;ALFAGUARA;16/10/2006 0:00:00;FICCIÓN;
00000017;TIEMPOS DE COLERA;GABRIEL GARCIA GARCIA;OVEJA NEGRA;1/9/2006 0:00:00;OCIO;
00000018;NOTICIA DE UN SECUESTRO;GABRIEL GARCIA GARCIA;ALFAGUARA;7/12/2006 0:00:00;FICCIÓN;
```

5. APPLICACION BIBLIOTECA

Paso 28. Crea el servlet ConsultaLibrosServlet. Utiliza el modelo processRequest en el servlet para unificar los métodos doGet y doPost. Utiliza la api getServletContext(). getRealPath("/") para obtener el path de este fichero.

```
@WebServlet("/ConsultaLibrosServlet")
public class ConsultaLibrosServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String titulo = request.getParameter("titulo");
        String path = getServletContext().getRealPath("/");
        File f = new File(path + "libros.txt");
        BufferedReader entrada = new BufferedReader(new FileReader(f));
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html><head><title>Servlet NewServlet</title></head>");
            out.println("<body>");
            while(entrada.ready()) {
                String linea = entrada.readLine();
                boolean presencia = linea.contains(titulo);
                if (presencia) out.println("<h3>" + linea + "</h3>");
            }
            out.println("</body>");
            out.println("</html>");
        }
    }
}
```

5. APLICACION BIBLIOTECA

Paso 29. Redirige la salida hacia el fichero consulta.jsp. Sólo obtén la lista de los libros seleccionados en un StringBuffer y pásasela al fichero jsp como parámetro:

```
@WebServlet("/ConsultaLibrosServlet")
public class ConsultaLibrosServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String titulo = request.getParameter("titulo");
        String path = getServletContext().getRealPath("/");
        File f = new File(path + "libros.txt");
        BufferedReader entrada = new BufferedReader(new FileReader(f));
        StringBuffer sb = new StringBuffer();
        while(entrada.ready()) {
            String linea = entrada.readLine();
            boolean presencia = linea.contains(titulo);
            if (presencia)
                sb.append("<h3>" + linea + "</h3>");
        }
        response.sendRedirect("consulta.jsp?lista=" + sb.toString());
    }
}
```

5. APLICACION BIBLIOTECA

Paso 30. Crea el archivo consulta.jsp. Solo debe recoger y mostrar el parámetro lista, compuesta por los libros seleccionados

The screenshot shows a Java IDE interface with two panes. The left pane displays the JSP code for 'consulta.jsp'. The right pane shows a web browser window displaying a list of books.

consulta.jsp

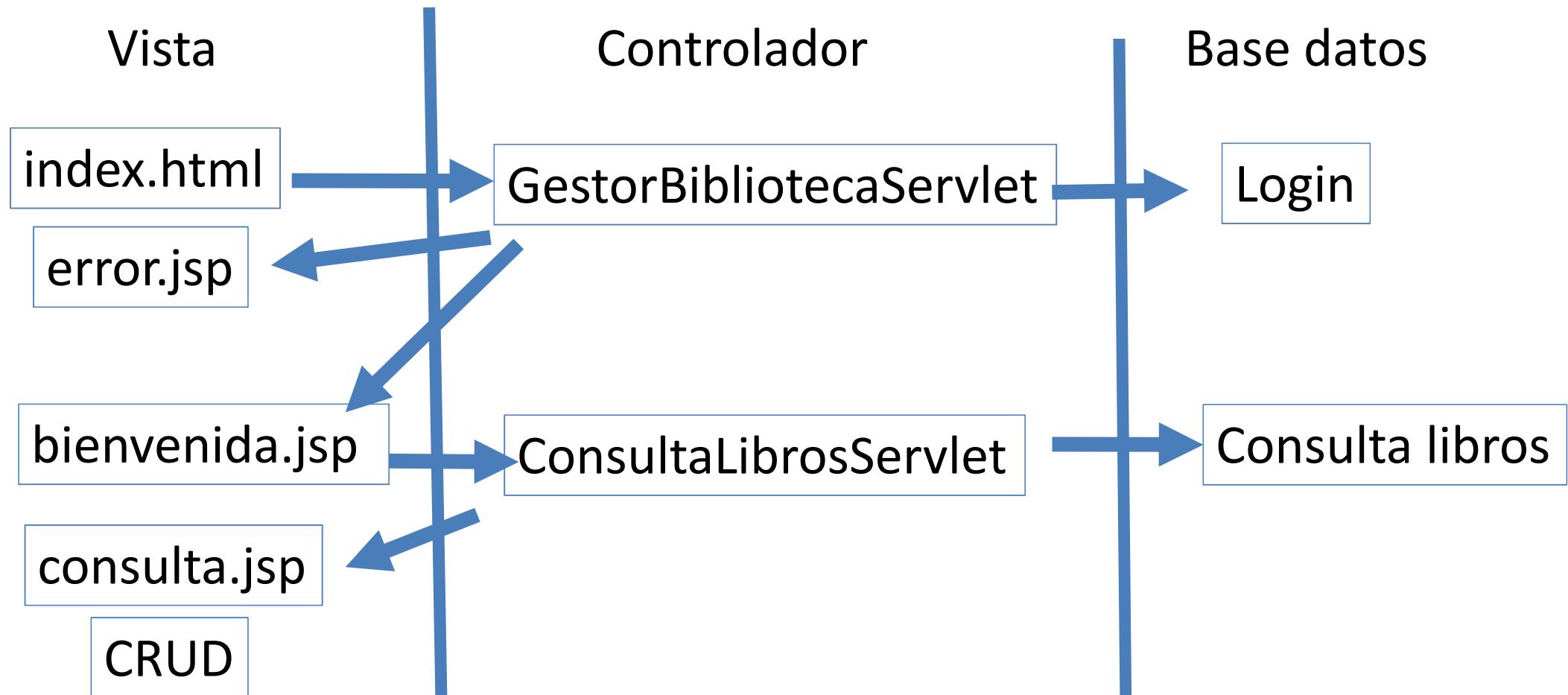
```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4 <html>
5 <head>
6 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9 <body>
10    <% String consulta = request.getParameter("lista");
11    %>
12 </body>
13 </html>
14
```

Browser Output:

localhost:8080/Biblioteca_JEE/ConsultaLibrosServlet?titulo=

00000001;HARRY POTTER Y EL PRISIONERO DE AZKABÁN;J.K ROWINS;SALAMANDRA;26/9/2006 0:00:00;INFANTIL;
00000002;EL GRAN LABERINTO;FERNANDO SABATER PEREZ;ARIEL;26/9/2006 0:00:00;FICCIÓN;
00000003;ROMEO Y JULIETA;WILLIAM SHAKESPEARE;SALAMANDRA;26/9/2006 0:00:00;ROMANTICA;
00000004;LA CARTA ESFERICA;ARTURO PEREZ LOPEZ;SALAMANDRA;29/9/2006 0:00:00;FICCIÓN;
00000005;CODIGO DA VINCI;DAN BROWN;ARIEL;29/9/2006 0:00:00;FICCIÓN;
00000006;MUCHO RUIDO Y POCAS NUECES;WILLIAM SHAKESPEARE;SALAMANDRA;29/9/2006 0:00:00;ROMANTICA;
00000007;PROTOCOLO;JOSE LOPEZ MURILLO;SALAMANDRA;6/9/2006 0:00:00;SOCIAL;
00000008;LINUX;FERNANDO SABATER PEREZ;ARIEL;6/9/2006 0:00:00;INFORMÁTICA;
00000009;EL TUMULTO;H.P LOVECRAFT;DEBATE;6/9/2006 0:00:00;CIENCIA;
00000010;PERSONAJES MITICOS;RICHARD HOLLIGHAM;DEBATE;7/9/2006 0:00:00;ENTRETENIMIENTO;
00000011;EL TIEMPO;J.K ROWINS;SALAMANDRA;7/9/2006 0:00:00;CIENCIA;

6. APLICACION CON BASE DATOS



6. APLICACION CON BASE DATOS

Paso 1. Crea una base de datos en Mysql con el nombre biblioteca_online, a partir del script Biblioteca_online.sql. Esta base de datos contendrá las siguientes tablas:

- La tabla Usuarios tendrá los campos usuario y contraseña.
- La tabla Libros tiene los campos que existen en el fichero “libros.txt”.

The screenshot shows the phpMyAdmin interface with two database tables displayed side-by-side.

Table 'libros':

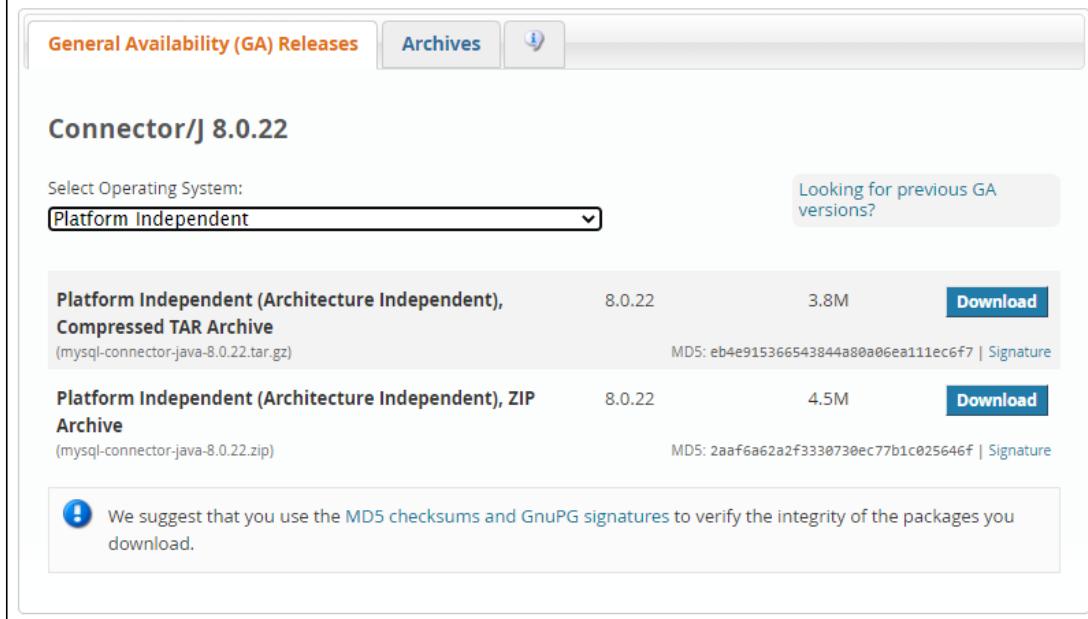
ID	Título	Autor	Editorial	Fecha	Categoría	Novedad
1	HARRY POTTER Y EL PRISIONERO DE AZKABÁN	J.K ROWLING	SALAMANDRA	2013-10-08	INFANTIL	0
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ ARIEL		2012-10-16	FICCIÓN	1
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	2014-07-18	ROMANTICA	0
4	LA CARTA ESFERICA	ARTURO PÉREZ LOPEZ	SALAMANDRA	2011-04-08	FICCION	1
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20	FICCION	0
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16	ROMANTICA	0
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21	SOCIAL	1
8	LINUX	FERNANDO SABATER PEREZ ARIEL		2012-02-02	INFORMATICA	1
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07	CIENCIA	0
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07	ENTRETENIMIENTO	1
11	EL TIEMPO	UK POW INC	SALAMANDRA	1999-11-05	CIENCIA	0
12	DIETAS MEDITERRANEAS					
13	ANGELES Y DEMONIOS					
14	FORTALEZA DIGITAL					
15	CAPITAN ALATRISTE					
16	PIEL DE TAMBOR					
17	TIEMPOS DE COLERA					
18	NOTICIA DE UN SECUESTRO					

Table 'usuarios':

ID	Usuario	Contraseña
1	admin	456
2	espai	123

6. APLICACION CON BASE DATOS

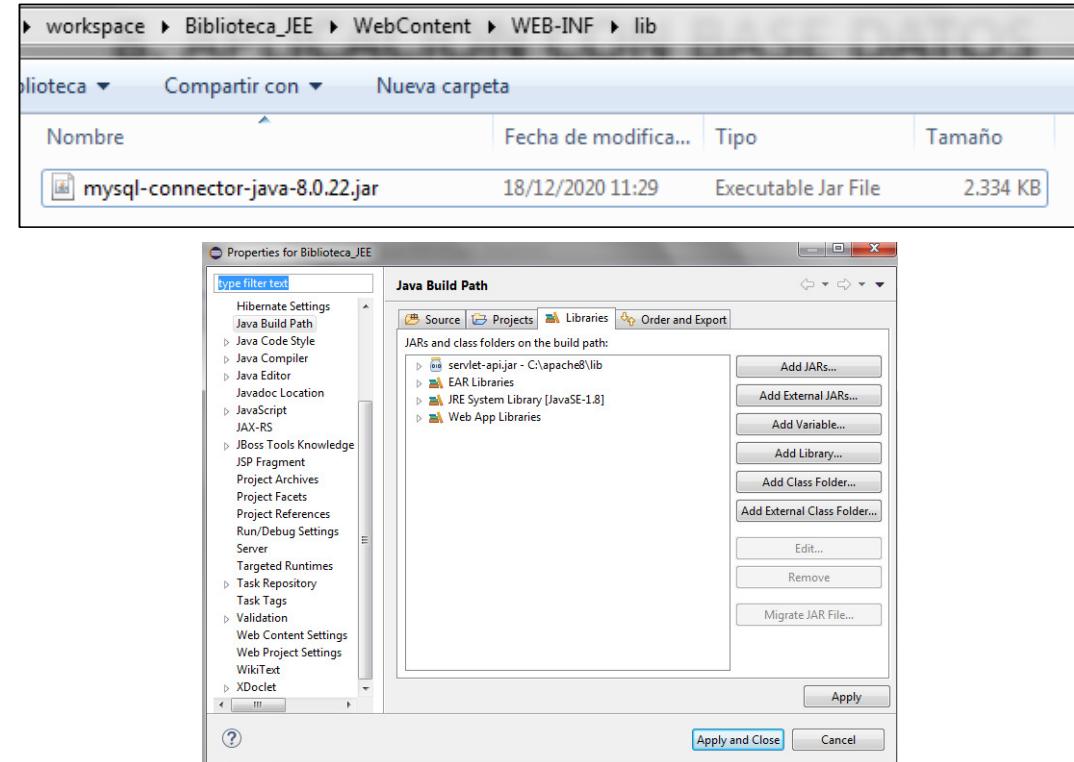
Paso 2. Con el fin de crear una conexión JDBC con la BBDD biblioteca_online, descarga el connector java-mysql (<https://dev.mysql.com/downloads/connector/j/>) y **copia el driver jar en la carpeta WEB-INF\lib del proyecto J2EE.**



The screenshot shows the MySQL Connector/J 8.0.22 download page. It features two main download links:

- Platform Independent (Architecture Independent), Compressed TAR Archive**: Size 3.8M, MD5: eb4e915366543844a80a06ea111ec6f7, Download button.
- Platform Independent (Architecture Independent), ZIP Archive**: Size 4.5M, MD5: 2aaaf6a62a2f3330730ec77b1c025646f, Download button.

A note at the bottom suggests using MD5 checksums and GnuPG signatures for integrity verification.



The screenshot shows the Eclipse IDE's Java Build Path dialog for the project "Biblioteca_JEE". The "Libraries" tab is selected, displaying the added JAR file:

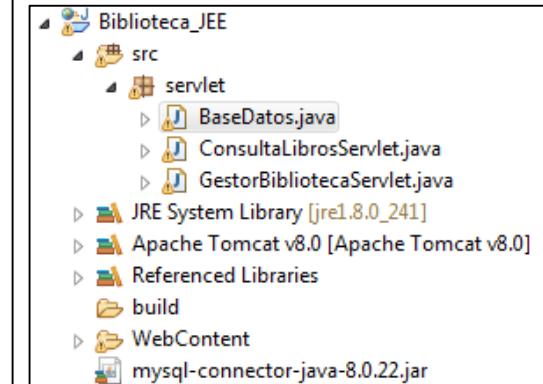
Nombre	Fecha de modifica...	Tipo	Tamaño
mysql-connector-java-8.0.22.jar	18/12/2020 11:29	Executable Jar File	2.334 KB

The dialog also includes tabs for Source, Projects, Order and Export, and buttons for Add JARs..., Add External JARs..., Add Variable..., Add Library..., Add Class Folder..., Add External Class Folder..., Edit..., Remove, Migrate JAR File..., Apply, and Cancel.

6. APLICACION CON BASE DATOS

Paso 3. Crea la clase BaseDatos.java que encapsulará todo el trabajo de base de datos de nuestra aplicación: conexión, inserción, modificación, borrado y consulta. Debe tener un objeto miembro de tipo Connection, y en su constructor debe de ser inicializado para que apunte a la base de datos biblioteca_online.

```
public class BaseDatos {  
    private Connection conexion;  
  
    public BaseDatos() {  
        try {  
            Class.forName("com.mysql.cj.jdbc.Driver");  
            String conex="jdbc:mysql://localhost:3306/biblioteca_online";  
            this.conexion = DriverManager.getConnection (conex,"root","");
        } catch (Exception e) {  
            e.printStackTrace();
        }
    }
}
```



6. APLICACION CON BASE DATOS

Paso 4. Realiza en la clase BaseDatos.java la función compruebaUsuario:
public boolean compruebaUsuario(String usuario, String password)

```
public boolean compruebaUsuario(String usuario, String password){  
    boolean check=false;  
    try {  
        Statement s = conexion.createStatement();  
        String sql = "SELECT count(*) FROM USUARIOS WHERE usuario='"+usuario+"' "  
                    + "and password='"+password+"'"';  
        s.execute(sql);  
        ResultSet rs = s.getResultSet();  
        rs.next();  
        if (rs.getInt(1)>0)  
            check=true;  
    } catch (SQLException ex) {  
        System.out.print(ex.getMessage());  
    }  
    return check;  
}
```

La utilizaremos en el servlet GestorBibliotecaServlet para hacer el login en la aplicación, de acuerdo con los usuarios introducidos en la base datos mysql

6. APLICACION CON BASE DATOS

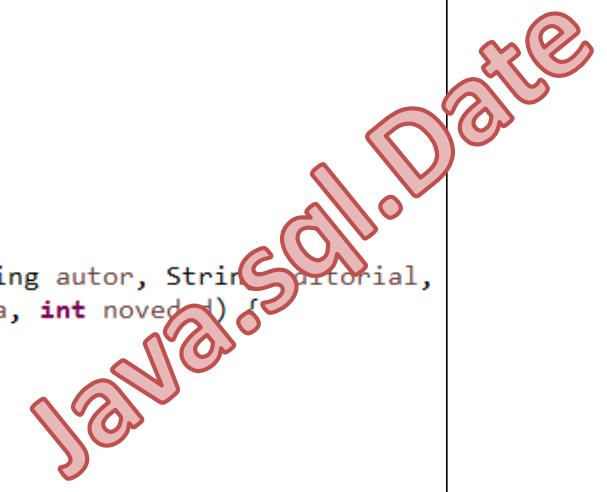
Paso 5. Reescribe GestorBibliotecaServlet usando el método comprouebaUsuario de forma que el login ahora se haga contra una base de datos:

```
@WebServlet("/GestorBibliotecaServlet")
public class GestorBibliotecaServlet extends HttpServlet {
    private boolean YaIniciado=false;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String usuario=request.getParameter("usuario"); //$_GET("");
        String password=request.getParameter("password");
        BaseDatos bd = new BaseDatos();
        //if (usuario.equals("pepe") && password.equals("pepe")) {
        if (bd.compruebaUsuario(usuario, password)){
            boolean iniciado = YaIniciado;
            if (!YaIniciado) YaIniciado = true;
            response.sendRedirect("bienvenida.jsp?usuario="+usuario+"&iniciado="+iniciado);
        }else
            response.sendRedirect("error.jsp?usuario="+usuario);
    }
}
```

6. APLICACION CON BASE DATOS

Paso 6. Crea la clase Libro utilizando los mismos atributos que las columnas de la tabla Libros:



```
1 package servlet;
2
3 import java.sql.Date;
4
5 public class Libro {
6     private int id;
7     private String titulo;
8     private String autor;
9     private String editorial;
10    private Date fecha;
11    private String categoria;
12    private int novedad;
13
14    public Libro(int id, String titulo, String autor, String editorial,
15                 Date fecha, String categoria, int novedad) {
16        this.id = id;
17        this.titulo = titulo;
18        this.autor = autor;
19        this.editorial = editorial;
20        this.fecha = fecha;
21        this.categoria = categoria;
22        this.novedad = novedad;
23    }
24    public int getId() {
25        return id;
26    }
}
```

6. APLICACION CON BASE DATOS

Paso 7. Realiza en la clase BaseDatos.java la función:

```
public ArrayList<Libro> consultaLibros(String filtro)
```

Obtiene un arrayList con los libros seleccionados según un filtro en el titulo:

```
public ArrayList<Libro> consultaLibros(String filtro){  
    ArrayList<Libro> lista =new ArrayList<Libro>();  
    try {  
        Statement s = conexion.createStatement();  
        String sql = "SELECT * FROM LIBROS WHERE TITULO LIKE '%"+filtro+"%'";  
        s.execute(sql);  
        ResultSet rs = s.getResultSet();  
        while (rs.next()) {  
            Libro libro = new Libro(rs.getInt(1), rs.getString(2), rs.getString(3),  
                                   rs.getString(4), rs.getDate(5), rs.getString(6),  
                                   rs.getInt(7));  
            lista.add(libro);  
        }  
    } catch (SQLException ex) {  
        System.out.print(ex.getMessage());  
    }  
    return lista;  
}
```

6. APLICACION CON BASE DATOS

Paso 8. Utiliza la funcion consultaLibros en el Servlet ConsultaLibrosServlet. Este debe pasar el ArrayList de la consulta a consulta.jsp como un atributo del request

```
@WebServlet("/ConsultaLibrosServlet")
public class ConsultaLibrosServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        String titulo = request.getParameter("titulo");
        BaseDatos db = new BaseDatos();
        ArrayList<Libro> libros = db.consultaLibros(titulo);
        //String path = getServletContext().getRealPath("/");
        //File f = new File(path + "libros.txt");
        //BufferedReader entrada = new BufferedReader(new FileReader(f));
        //StringBuffer sb = new StringBuffer();
        //while(entrada.ready()) {
        //    String linea = entrada.readLine();
        //    boolean presencia = linea.contains(titulo);
        //    if (presencia)
        //        sb.append("<h3>" + linea + "</h3>");
        //}
        request.setAttribute("lista", libros);
        getServletContext().getRequestDispatcher("/consulta.jsp").forward(request, response);
    }
}
```

6. APLICACION CON BASE DATOS

Paso 9. El archivo consulta.jsp debe recoger el atributo arrayList de libros y mostrarlos en una tabla html con todos los campos de la base de datos.

```
consulta.jsp ✘
9 <title>Insert title here</title>
10 </head>
11<%><body>
12     <H1>LIBROS DE LA BIBLIOTECA</H1>
13     <%ArrayList<Libro> libros = (ArrayList<Libro>)request.getAttribute("lista");%>
14
15     <table border=1>
16         <tr><h2><td>ID</td><td>TITULO</td><td>AUTOR</td><td>EDITORIAL</td><td>FECHA</td><td>CATEGORIA</td><td>NOVEDAD</h2>
17         <% for (Libro libro : libros){%
18             out.print("<tr><td>" + libro.getId() + "</td>");
19             out.print("<td>" + libro.getTitulo() + "</td>");
20             out.print("<td>" + libro.getAutor() + "</td>");
21             out.print("<td>" + libro.getEditorial() + "</td>");
22             out.print("<td>" + libro.getFecha() + "</td>");
23             out.print("<td>" + libro.getCategoría() + "</td>");
24             out.print("<td>" + libro.getNovedad() + "</td></tr>");
25         }
26     %>
27     </table>
28 </body>
29 </html>
```

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD
1	HARRY POTTER Y EL PRISIONERO DE AZKABAN	J.K ROWLING	SALAMANDRA	2013-10-08	INFANTIL	0
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	2012-10-16	FICCIÁ?N	1
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	2014-07-18	ROMANTICA	0
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	2011-04-08	FICCI?N	1
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20	FICCI?N	0
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16	ROMANTICA	0
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21	SOCIAL	1
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	2012-02-02	INFORMATICA	1
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07	CIENCIA	0
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07	ENTRETENIMIENTO	1
11	EL TIEMPO	JK ROWLING	SALAMANDRA	1999-11-05	CIENCIA	0
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	2014-09-16	COCINA	1
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	2014-09-21	FICCI?N	0
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	2011-04-19	FICCI?N	1
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	2001-04-17	HISTORICA	0
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	2009-04-14	HISTORICA	0
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	2002-08-04	HISTORICA	1
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	2013-05-13	INTRIGA	0

6. APLICACION CON BASE DATOS

Paso 10. Implementa la acción de insertar un libro en la bbdd desde un formulario en consulta.jsp, utilizando para ello el servlet GestionLibrosServlet. En consulta.jsp crea un formulario que proporcione los campos: Id, Titulo, Autor, Editorial, Fecha, Categoría y Novedad. Pon ConsultaLibrosServlet como el action del formulario.

```
consulta.jsp
11<body>
12 <H1>LIBROS DE LA BIBLIOTECA</H1>
13 <%ArrayList<Libro> libros = (ArrayList<Libro>)request.getAttribute("lista");%>
14 <table border=1>
15   <tr><h2><td>ID</td>TITULO</td>AUTOR</td>EDITORIAL</td>FECHA</td>CATEGORIA</td>NOVEDAD</td></tr>
16   <%for (Libro libro : libros){%
17     out.print("<tr><td>" + libro.getId() + "</td>");
18     out.print("<td>" + libro.getTitulo() + "</td>");
19     out.print("<td>" + libro.getAutor() + "</td>");
20     out.print("<td>" + libro.getEditorial() + "</td>");
21     out.print("<td>" + libro.getFecha() + "</td>");
22     out.print("<td>" + libro.getCategoría() + "</td>");
23     out.print("<td>" + libro.getNovedad() + "</td></tr>" );
24   } %>
25 </table>
26 <form action="GestionLibrosServlet" method="post">
27   ID: <input type="text" name="id">
28   TITULO: <input type="text" name="titulo">
29   AUTOR: <input type="text" name="autor">
30   EDITORIAL: <input type="text" name="editorial"> <br><br>
31   FECHA: <input type="text" name="fecha">
32   CATEGORIA: <input type="text" name="categoria">
33   NOVEDAD: <input type="text" name="novedad"><br><br>
34   <input type="submit" name="submit" value="Insertar Libro">
35 </form>
36 </body>
37 </html>
```

localhost:8080/Biblioteca_JEE/ConsultaLibrosServlet?titulo=

LIBROS DE LA BIBLIOTECA

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD
1	HARRY POTTER Y EL PRISIONERO DE AZKABAN	J.K ROWLING	SALAMANDRA	2013-10-08	INFANTIL	0
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	2012-10-16	FICCION	1
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	2014-07-18	ROMANTICA	0
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	2011-04-08	FICCION	1
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20	FICCION	0
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16	ROMANTICA	0
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21	SOCIAL	1
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	2012-02-02	INFORMATICA	1
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07	CIENCIA	0
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07	ENTRETENIMIENTO	1
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	1999-11-05	CIENCIA	0
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	2014-09-16	COCINA	1
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	2014-09-21	FICCION	0
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	2011-04-19	FICCION	1
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	2001-04-17	HISTORICA	0
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	2009-04-14	HISTORICA	0
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	2002-08-04	HISTORICA	1
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	2013-05-13	INTRIGA	0

ID: TITULO: AUTOR: EDITORIAL:

FECHA: CATEGORIA: NOVEDAD:

6. APLICACION CON BASE DATOS

Paso 11. Realiza en la clase BaseDatos.java una función similar a la siguiente:

```
public void insertarLibro(Libro libro)
```

```
public void insertarLibro(Libro libro){
    String query = " insert into libros (id, titulo, autor, editorial, fecha, categoria, novedad)"
        + " values (?, ?, ?, ?, ?, ?, ?)";
    try {
        PreparedStatement preparedStmt;
        preparedStmt = conexion.prepareStatement(query);
        preparedStmt.setInt (1, libro.getId());
        preparedStmt.setString (2, libro.getTitulo());
        preparedStmt.setString (3, libro.getAutor());
        preparedStmt.setString (4, libro.getEditorial());

        java.sql.Date sqlDate = new java.sql.Date(libro.getFecha().getTime());
        preparedStmt.setDate (5, sqlDate);
        preparedStmt.setString (6, libro.getCategoría());
        preparedStmt.setInt (7, libro.getNovedad());
        preparedStmt.executeUpdate();
    } catch (SQLException ex) {
        System.out.print(ex.getMessage());
    }
}
```

6. APLICACION CON BASE DATOS

Paso 12. Las diferentes llamadas al servlet ConsultaLibrosServlet se distinguirán usando el name y el value del botón submit.

```
@WebServlet("/ConsultaLibrosServlet")
public class ConsultaLibrosServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException, ParseException {

        BaseDatos db = new BaseDatos();
        String boton = request.getParameter("submit");
        String filtro="";
        if (boton.equals("Consulta Libros")){
            filtro = request.getParameter("titulo");
        }else if (boton.equals("Insertar Libro")){
            int id = Integer.parseInt(request.getParameter("id"));
            String titulo = request.getParameter("titulo");
            String autor = request.getParameter("autor");
            String editorial = request.getParameter("editorial");
            DateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");
            Date fecha = formatter.parse(request.getParameter("fecha"));
            String categoria = request.getParameter("categoria");
            int novedad = Integer.parseInt(request.getParameter("novedad"));
            Libro libro = new Libro(id, titulo, autor, editorial, fecha,categoria, novedad);
            db.insertarLibro(libro);
        }
        ArrayList<Libro> libros = db.consultaLibros(filtro);
        request.setAttribute("lista", libros);
        getServletContext().getRequestDispatcher("/consulta.jsp").forward(request, response);
    }
}
```

```
 consulta.jsp ✘
26  <form action="ConsultaLibrosServlet" method="post">
27      ID: <input type="text" name="id">
28      TITULO: <input type="text" name="titulo">
29      AUTOR: <input type="text" name="autor">
30      EDITORIAL: <input type="text" name="editorial"> <br><br>
31      FECHA: <input type="text" name="fecha">
32      CATEGORIA: <input type="text" name="categoria">
33      NOVEDAD: <input type="text" name="novedad"><br><br>
34      <input type="submit" name="submit" value="Insertar Libro">
35  </form>
36 </body>
37 </html>
```

```
 bienvenida.jsp ✘
16  <h1>BIENVENIDO USUARIO <%=usuario%> (llamada <%=method%>)</h1>
17  <form method=GET action=ConsultaLibrosServlet>
18      Seleccion de Libro: <input type="text" name="titulo"><br><br>
19      <input type="submit" value="Consulta Libros" name="submit">
20  </form>
21 </body>
22 </html>
```

6. APLICACION CON BASE DATOS

Paso 13. El proceso de inserción seria:

localhost:8080/Biblioteca_JEE/ConsultaLibrosServlet?titulo=&submit=Consulta+Libros

LIBROS DE LA BIBLIOTECA

ID	TITULO	AUTOR	EDITORIAL	FECHA
1	HARRY POTTER Y EL PRISIONERO DE AZKABAN	J.K ROWLING	SALAMANDRA	2013-10-08
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	2012-10-16
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	2014-07-18
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	2011-04-08
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	2012-02-02
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	1999-11-05
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	2014-09-16
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	2014-09-21
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	2011-04-19
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	2001-04-17
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	2009-04-14
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	2002-08-04
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	2013-05-13

ID: 19 TITULO: EL NOMBRE DE LA ROSA AUTOR: UMBERTO ECO EDITORIAL: BRUGUERA
FECHA: 16/07/2020 CATEGORIA: RELIGION NOVEDAD: 1

localhost:8080/Biblioteca_JEE/ConsultaLibrosServlet

LIBROS DE LA BIBLIOTECA

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD
1	HARRY POTTER Y EL PRISIONERO DE AZKABAN	J.K ROWLING	SALAMANDRA	2013-10-08	INFANTIL	0
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	2012-10-16	FICCION	1
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	2014-07-18	ROMANTICA	0
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	2011-04-08	FICCION	1
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20	FICCION	0
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16	ROMANTICA	0
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21	SOCIAL	1
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	2012-02-02	INFORMATICA	1
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07	CIENCIA	0
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07	ENTRETENIMIENTO	1
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	1999-11-05	CIENCIA	0
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	2014-09-16	COCINA	1
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	2014-09-21	FICCION	0
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	2011-04-19	FICCION	1
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	2001-04-17	HISTORICA	0
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	2009-04-14	HISTORICA	0
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	2002-08-04	HISTORICA	1
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	2013-05-13	INTRIGA	0
19	EL NOMBRE DE LA ROSA	UMBERTO ECO	BRUGUERA	2020-07-16	RELIGION	1

ID: TITULO: AUTOR: EDITORIAL:
FECHA: CATEGORIA: NOVEDAD:

6. APLICACION CON BASE DATOS

Paso 14. Añade a la página anterior la posibilidad de eliminar libros. En consulta.jsp agrega una nueva columna “Eliminar” formada por checkboxes a la derecha de cada entrada de la lista. Cada checkbox servirá para la posterior selección y eliminación de un libro de la base de datos.

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD	ELIMINAR
1	HARRY POTTER Y EL PRISIONERO DE AZKABAN	J.K ROWLING	SALAMANDRA	08/10/2013	INFANTIL	0	<input type="checkbox"/>
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	16/10/2012	FICCIÓN	1	<input type="checkbox"/>
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	18/07/2014	ROMANTICA	0	<input type="checkbox"/>
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	08/04/2011	FICCIÓN	1	<input type="checkbox"/>
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	20/10/2010	FICCIÓN	0	<input type="checkbox"/>
6	MUCHO RUIDO Y pocas NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	16/09/2011	ROMANTICA	0	<input type="checkbox"/>
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	21/01/2014	SOCIAL	1	<input type="checkbox"/>
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	02/02/2012	INFORMATICA	1	<input type="checkbox"/>
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	07/07/2001	CIENCIA	0	<input type="checkbox"/>
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	07/01/2013	ENTRETENIMIENTO	1	<input type="checkbox"/>
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	05/11/1999	CIENCIA	0	<input type="checkbox"/>
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	16/09/2014	COCINA	1	<input type="checkbox"/>
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	21/09/2014	FICCIÓN	0	<input type="checkbox"/>
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	19/04/2011	FICCIÓN	1	<input type="checkbox"/>
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	17/04/2001	HISTORICA	0	<input type="checkbox"/>
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	14/04/2009	HISTORICA	0	<input type="checkbox"/>
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	04/08/2002	HISTORICA	1	<input type="checkbox"/>
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	13/05/2013	INTRIGA	0	<input type="checkbox"/>

ID: TITULO: AUTOR: EDITORIAL:
FECHA: CATEGORIA: NOVEDAD:

6. APLICACION CON BASE DATOS

Paso 15. En consulta.jsp el conjunto de checkboxes Eliminar se situaran dentro de un formulario para enviar a consylta LibrosServlet

```
consulta.jsp
12     <H1>LIBROS DE LA BIBLIOTECA</H1>
13     <%ArrayList<Libro> libros = (ArrayList<Libro>)request.getAttribute("lista");%>
14
15     <form action="ConsultaLibrosServlet" method="post">
16         <table border=1>
17             <tr><h2><td>ID<td>TITULO<td>AUTOR<td>EDITORIAL<td>FECHA
18             <td>CATEGORIA<td>NOVEDAD<td>ELIMINAR</h2>
19             <%for (Libro libro : libros){%
20                 out.print("<tr><td>" + libro.getId() + "</td>");
21                 out.print("<td>" + libro.getTitulo() + "</td>");
22                 out.print("<td>" + libro.getAutor() + "</td>");
23                 out.print("<td>" + libro.getEditorial() + "</td>");
24                 out.print("<td>" + libro.getFecha() + "</td>");
25                 out.print("<td>" + libro.getCategoría() + "</td>");
26                 out.print("<td>" + libro.getNovedad() + "</td></h3>");
27                 out.print("<td><center><input type=checkbox name=eliminados value=" + libro.getId() + " /></center>");}
28             } %>
29
30         </table><br>
31         <input type="submit" name="submit" value="Eliminar Libros"><br><br>
32     </form>
```

6. APLICACION CON BASE DATOS

Paso 16. Realiza en la clase BaseDatos.java una función similar a la siguiente:

```
public void eliminarLibro (String id)
```

```
public void eliminarLibro(String id){  
    String query = " delete from libros where id="+id;  
  
    try {  
        PreparedStatement preparedStmt = conexion.prepareStatement(query);  
        preparedStmt.executeUpdate();  
    } catch (SQLException ex) {  
        System.out.print(ex.getMessage());  
    }  
}
```

6. APLICACION CON BASE DATOS

Paso 17. En el servlet GestionLibrosServlet añade el código que permita eliminar los libros seleccionados.

Después del proceso de eliminación, se debe redirigir al usuario de nuevo al listado de consulta.jsp, para que se vean los cambios realizados.

```
ConsultaLibrosServlet.java
16 @WebServlet("/ConsultaLibrosServlet")
17 public class ConsultaLibrosServlet extends HttpServlet {
18     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
19             throws ServletException, IOException, ParseException {
20
21     BaseDatos db = new BaseDatos();
22     String boton = request.getParameter("submit");
23     String filtro="";
24     if (boton.equals("Consulta Libros")){
25         filtro = request.getParameter("titulo");
26     }else if (boton.equals("Insertar Libro")){
27         int id = Integer.parseInt(request.getParameter("id"));
28         String titulo = request.getParameter("titulo");
29         String autor = request.getParameter("autor");
30         String editorial = request.getParameter("editorial");
31         DateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");
32         Date fecha = formatter.parse(request.getParameter("fecha"));
33         String categoria = request.getParameter("categoria");
34         int novedad = Integer.parseInt(request.getParameter("novedad"));
35         Libro libro = new Libro(id, titulo, autor, editorial, fecha,categoria, novedad);
36         db.insertarLibro(libro);
37     }else if (boton.equals("Eliminar Libros")){
38         String[] ids = request.getParameterValues("eliminados");
39         for(String id:ids){
40             db.eliminarLibro(id);
41         }
42     }
43     ArrayList<Libro> libros = db.consultaLibros(filtro);
44     request.setAttribute("lista", libros);
45     getServletContext().getRequestDispatcher("/consulta.jsp").forward(request, response);
46 }
```

6. APLICACION CON BASE DATOS

Paso 18. Esta sería la secuencia que se vería en la eliminación de 3 libros:

localhost:8080/Biblioteca_JEE/ConsultaLibrosServlet?titulo=&submit=Consulta+Libros

LIBROS DE LA BIBLIOTECA

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD	ELIMINAR
1	HARRY POTTER Y EL PRISIONERO DE AZKABAN	J.K ROWLING	SALAMANDRA	2013-10-08	INFANTIL	0	<input checked="" type="checkbox"/>
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	2012-10-16	FICCIÁ?N	1	<input checked="" type="checkbox"/>
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	2014-07-18	ROMANTICA	0	<input checked="" type="checkbox"/>
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	2011-04-08	FICCION	1	<input type="checkbox"/>
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20	FICCION	0	<input type="checkbox"/>
6	MUCHO RUIDO Y pocas NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16	ROMANTICA	0	<input type="checkbox"/>
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21	SOCIAL	1	<input type="checkbox"/>
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	2012-02-02	INFORMATICA	1	<input type="checkbox"/>
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07	CIENCIA	0	<input type="checkbox"/>
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07	ENTRETENIMIENTO	1	<input type="checkbox"/>
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	1999-11-05	CIENCIA	0	<input type="checkbox"/>
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	2014-09-16	COCINA	1	<input type="checkbox"/>
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	2014-09-21	FICCION	0	<input type="checkbox"/>
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	2011-04-19	FICCION	1	<input type="checkbox"/>
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	2001-04-17	HISTORICA	0	<input type="checkbox"/>
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	2009-04-14	HISTORICA	0	<input type="checkbox"/>
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	2002-08-04	HISTORICA	1	<input type="checkbox"/>
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	2013-05-13	INTRIGA	0	<input type="checkbox"/>
19	EL NOMBRE DE LA ROSA	UMBERTO ECO	BRUGUERA	2020-07-16	RELIGION	1	<input type="checkbox"/>

localhost:8080/Biblioteca_JEE/ConsultaLibrosServlet

LIBROS DE LA BIBLIOTECA

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD	ELIMINAR
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	2011-04-08	FICCION	1	<input type="checkbox"/>
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	2010-10-20	FICCION	0	<input type="checkbox"/>
6	MUCHO RUIDO Y pocas NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	2011-09-16	ROMANTICA	0	<input type="checkbox"/>
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	2014-01-21	SOCIAL	1	<input type="checkbox"/>
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	2012-02-02	INFORMATICA	1	<input type="checkbox"/>
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	2001-07-07	CIENCIA	0	<input type="checkbox"/>
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	2013-01-07	ENTRETENIMIENTO	1	<input type="checkbox"/>
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	1999-11-05	CIENCIA	0	<input type="checkbox"/>
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	2014-09-16	COCINA	1	<input type="checkbox"/>
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	2014-09-21	FICCION	0	<input type="checkbox"/>
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	2011-04-19	FICCION	1	<input type="checkbox"/>
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	2001-04-17	HISTORICA	0	<input type="checkbox"/>
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	2009-04-14	HISTORICA	0	<input type="checkbox"/>
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	2002-08-04	HISTORICA	1	<input type="checkbox"/>
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	2013-05-13	INTRIGA	0	<input type="checkbox"/>
19	EL NOMBRE DE LA ROSA	UMBERTO ECO	BRUGUERA	2020-07-16	RELIGION	1	<input type="checkbox"/>

ID: TITULO: AUTOR: EDITORIAL:
FECHA: CATEGORIA: NOVEDAD:

6. APLICACION CON BASE DATOS

Paso 19. Vamos a permitir también la edición de los libros ya existentes. Añade una nueva columna “Modificar” de checkboxes a la derecha de cada fila de libros, y un nuevo botón “Recuperar Libro”.

```
consulta.jsp ✘
12  <H1>LIBROS DE LA BIBLIOTECA</H1>
13  <%ArrayList<Libro> libros = (ArrayList<Libro>)request.getAttribute("lista");%>
14
15<form action="ConsultaLibrosServlet" method="post">
16    <table border=1>
17      <tr><h2><td>ID</td>TITULO</td>AUTOR</td>EDITORIAL</td>FECHA
18      <td>CATEGORIA</td>NOVEDAD</td>ELIMINAR</td>MODIFICAR</h2>
19      <%for (Libro libro : libros){%
20          out.print("<tr><h3><td>" + libro.getId() + "</td>");
21          out.print("<td>" + libro.getTitulo() + "</td>");
22          out.print("<td>" + libro.getAutor() + "</td>");
23          out.print("<td>" + libro.getEditorial() + "</td>");
24          out.print("<td>" + libro.getFecha() + "</td>");
25          out.print("<td>" + libro.getCategoría() + "</td>");
26          out.print("<td>" + libro.getNovedad() + "</td></h3>");
27          out.print("<td><center><input type=checkbox name=eliminados value=" + libro.getId() + " /></center>" );
28          out.print("<td><center><input type=checkbox name=recuperados value=" + libro.getId() + " /></center>" );
29      } %>
30
31    </table><br>
32    <input type="submit" name="submit" value="Eliminar Libros">
33    <input type="submit" name="submit" value="Recuperar Libro"><br><br>
34
35  </form>
```

6. APLICACION CON BASE DATOS

Paso 20. El proceso de modificar se divide en dos partes:

1º Selección del libro y recuperación en el formulario de consulta.jsp.

2º Modificación o cancelación de los datos del libro en el formulario de consulta.jsp

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD	ELIMINAR
1	HARRY POTTER Y EL PRISIONERO DE AZKABÁN	J.K ROWLING	SALAMANDRA	08/10/2013	INFANTIL	0	<input type="checkbox"/>
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	16/10/2012	FICCIÓN	1	<input type="checkbox"/>
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	18/07/2014	ROMANTICA	0	<input type="checkbox"/>
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	08/04/2011	FICCION	1	<input type="checkbox"/>
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	20/10/2010	FICCION	0	<input type="checkbox"/>
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	16/09/2011	ROMANTICA	0	<input type="checkbox"/>
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	21/01/2014	SOCIAL	1	<input type="checkbox"/>
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	02/02/2012	INFORMATICA	1	<input type="checkbox"/>
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	07/07/2001	CIENCIA	0	<input type="checkbox"/>
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	07/01/2013	ENTRETENIMIENTO	1	<input type="checkbox"/>
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	05/11/1999	CIENCIA	0	<input type="checkbox"/>
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	16/09/2014	COCINA	1	<input type="checkbox"/>
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	21/09/2014	FICCION	0	<input type="checkbox"/>
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	19/04/2011	FICCION	1	<input type="checkbox"/>
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	17/04/2001	HISTORICA	0	<input type="checkbox"/>
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	14/04/2009	HISTORICA	0	<input type="checkbox"/>
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	04/08/2002	HISTORICA	1	<input type="checkbox"/>
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	13/05/2013	INTRIGA	0	<input type="checkbox"/>
19	EL NOMBRE DE LA ROSA	HUMBERTO ECO	PLANETA	12/12/2010	RELIGION	0	<input type="checkbox"/>

ID:	TITULO:	AUTOR:	EDITORIAL:
FECHA:	CATEGORIA:	NOVEDAD:	
<input type="button" value="Insertar Libro"/>			

ID	TITULO	AUTOR	EDITORIAL	FECHA	CATEGORIA	NOVEDAD	ELIMINAR	MODIFICAR
1	HARRY POTTER Y EL PRISIONERO DE AZKABÁN	J.K ROWLING	SALAMANDRA	08/10/2013	INFANTIL	0	<input type="checkbox"/>	<input type="checkbox"/>
2	EL GRAN LABERINTO	FERNANDO SABATER PEREZ	ARIEL	16/10/2012	FICCIÓN	1	<input type="checkbox"/>	<input type="checkbox"/>
3	ROMEO Y JULIETA	WILLIAM SHAKESPEARE	SALAMANDRA	18/07/2014	ROMANTICA	0	<input type="checkbox"/>	<input type="checkbox"/>
4	LA CARTA ESFERICA	ARTURO PEREZ LOPEZ	SALAMANDRA	08/04/2011	FICCION	1	<input type="checkbox"/>	<input type="checkbox"/>
5	EL CODIGO DA VINCI	DAN BROWN	ARIEL	20/10/2010	FICCION	0	<input type="checkbox"/>	<input type="checkbox"/>
6	MUCHO RUIDO Y POCAS NUECES	WILLIAM SHAKESPEARE	SALAMANDRA	16/09/2011	ROMANTICA	0	<input type="checkbox"/>	<input type="checkbox"/>
7	PROTOCOLO	JOSE LOPEZ MURILLO	SALAMANDRA	21/01/2014	SOCIAL	1	<input type="checkbox"/>	<input type="checkbox"/>
8	LINUX	FERNANDO SABATER PEREZ	ARIEL	02/02/2012	INFORMATICA	1	<input type="checkbox"/>	<input type="checkbox"/>
9	EL TUMULTO	H.P LOVECRAFT	DEBATE	07/07/2001	CIENCIA	0	<input type="checkbox"/>	<input type="checkbox"/>
10	PERSONAJES MITICOS	RICHARD HOLLIGHAM	DEBATE	07/01/2013	ENTRETENIMIENTO	1	<input type="checkbox"/>	<input type="checkbox"/>
11	EL TIEMPO	J.K ROWLING	SALAMANDRA	05/11/1999	CIENCIA	0	<input type="checkbox"/>	<input type="checkbox"/>
12	DIETAS MEDITERRANEAS	ARTURO PEREZ LOPEZ	ARIEL	16/09/2014	COCINA	1	<input type="checkbox"/>	<input type="checkbox"/>
13	ANGELES Y DEMONIOS	DAN BROWN	ARIEL	21/09/2014	FICCION	0	<input type="checkbox"/>	<input type="checkbox"/>
14	FORTALEZA DIGITAL	DAN BROWN	ARIEL	19/04/2011	FICCION	1	<input type="checkbox"/>	<input type="checkbox"/>
15	CAPITAN ALATRISTE	ARTURO PEREZ REVERTE	ALFAGUARA	17/04/2001	HISTORICA	0	<input type="checkbox"/>	<input type="checkbox"/>
16	PIEL DE TAMBOR	ARTURO PEREZ REVERTE	ALFAGUARA	14/04/2009	HISTORICA	0	<input type="checkbox"/>	<input type="checkbox"/>
17	TIEMPOS DE COLERA	GABRIEL GARCIA	OVEJA NEGRA	04/08/2002	HISTORICA	1	<input type="checkbox"/>	<input type="checkbox"/>
18	NOTICIA DE UN SECUESTRO	GABRIEL GARCIA	ALFAGUARA	13/05/2013	INTRIGA	0	<input type="checkbox"/>	<input type="checkbox"/>
19	EL NOMBRE DE LA ROSA	HUMBERTO ECO	PLANETA	12/12/2010	RELIGION	0	<input type="checkbox"/>	<input type="checkbox"/>

ID:	TITULO:	AUTOR:	EDITORIAL:
FECHA:	CATEGORIA:	NOVEDAD:	
<input type="button" value="Actualizar Libro"/> <input type="button" value="Cancelar"/>			

6. APLICACION CON BASE DATOS

Paso 21. En consulta.jsp debemos tener en cuenta las dos situaciones: el caso normal de mostrar su formulario y el caso de modificación de libro con su formulario rellanado con los datos del libro.

```
36
37  <% Object mod = request.getAttribute("FlagModificar");
38  if (mod==null){ %>
39    <form action="GestionLibrosServlet" method="post">
40      ID: <input type="text" name="id">
41      TITULO: <input type="text" name="titulo">
42      AUTOR: <input type="text" name="autor">
43      EDITORIAL: <input type="text" name="editorial" > <br><br>
44      FECHA: <input type="text" name="fecha">
45      CATEGORIA: <input type="text" name="categoria">
46      NOVEDAD: <input type="text" name="novedad"><br><br>
47      <input type="submit" name="submit" value="Insertar Libro">
48    </form>
49  }else{
50    Libro libro = (Libro)request.getAttribute("libro");%>
51    <form action="ConsultalibrosServlet" method="post">
52      ID: <input type="text" name="id" value="<%=libro.getId()%>" readonly>
53      TITULO: <input type="text" name="titulo" value="<%=libro.getTitulo()%>">
54      AUTOR: <input type="text" name="autor" value="<%=libro.getAutor()%>">
55      EDITORIAL: <input type="text" name="editorial" value="<%=libro.getEditorial()%>"> <br><br>
56      FECHA: <input type="text" name="fecha" value="<%=libro.getFecha()%>">
57      CATEGORIA: <input type="text" name="categoria" value="<%=libro.getCategoría()%>">
58      NOVEDAD: <input type="text" name="novedad" value="<%=libro.getNovedad()%>"><br><br>
59      <input type="submit" name="submit" value="Actualiza Libro">
60      <input type="submit" name="submit" value="Cancelar">
61    </form>
62  <%}>
63  </body>
64 </html>
```



Caso
normal

Caso modificación
de libro

6. APLICACION CON BASE DATOS

Paso 22. Realiza en la clase BaseDatos.java una función similar a la siguiente:

```
public Libro recuperarLibro (String id)
```

```
public Libro recuperarLibro(String id){  
    Libro libro = null;  
    try {  
        Statement s = conexion.createStatement();  
        String sql = "SELECT * FROM LIBROS WHERE ID="+id;  
        s.execute(sql);  
        ResultSet rs = s.getResultSet();  
        rs.next();  
        libro = new Libro(rs.getInt(1),rs.getString(2),rs.getString(3), rs.getString(4),  
                          rs.getDate(5), rs.getString(6), rs.getInt(7));  
    } catch (SQLException ex) {  
        System.out.print(ex.getMessage());  
    }  
    return libro;  
}
```

6. APLICACION CON BASE DATOS

Paso 23. Realiza en la clase BaseDatos.java una función similar a la siguiente:

```
public void modificarLibro(Libro libro);
```

```
public void insertarLibro(Libro libro){  
    String query = " insert into libros (id, titulo, autor, editorial, fecha, categoria, novedad)"  
        + " values (?, ?, ?, ?, ?, ?, ?)";  
    try {  
        PreparedStatement preparedStmt;  
        preparedStmt = conexion.prepareStatement(query);  
        preparedStmt.setInt (1, libro.getId());  
        preparedStmt.setString (2, libro.getTitulo());  
        preparedStmt.setString (3, libro.getAutor());  
        preparedStmt.setString (4, libro.getEditorial());  
  
        java.sql.Date sqlDate = new java.sql.Date(libro.getFecha().getTime());  
        preparedStmt.setDate (5, sqlDate);  
        preparedStmt.setString (6, libro.getCategoría());  
        preparedStmt.setInt (7, libro.getNovedad());  
        preparedStmt.executeUpdate();  
    } catch (SQLException ex) {  
        System.out.print(ex.getMessage());  
    }  
}
```

6. APLICACION CON BASE DATOS

Paso 24. En ConsultaLibrosServlet debemos contemplar las dos opciones utilizando las dos funciones anteriormente definidas:

```
ConsultaLibrosServlet.java ✘
39     }else if (boton.equals("Eliminar Libros")){
40         String[] ids = request.getParameterValues("eliminados");
41         for(String id:ids){
42             db.eliminarLibro(id);
43         }
44     }else if (boton.equals("Recuperar Libro")){
45         String[] ids = request.getParameterValues("recuperados");
46         Libro libro=db.recuperarLibro(ids[0]);
47         request.setAttribute("libro",libro );
48         request.setAttribute("FlagModificar", 1);
49     }else if(boton.equals("Actualiza Libro")){
50         int id = Integer.parseInt(request.getParameter("id"));
51         String titulo = request.getParameter("titulo");
52         String autor = request.getParameter("autor");
53         String editorial = request.getParameter("editorial");
54
55         DateFormat format = new SimpleDateFormat("yyyyMMdd");
56         java.util.Date parsedate = format.parse(request.getParameter("fecha"));
57         java.sql.Date fecha = new java.sql.Date(parsedate.getTime());
58
59         String categoria = request.getParameter("categoria");
60         int novedad = Integer.parseInt(request.getParameter("novedad"));
61         Libro libro = new Libro(id, titulo, autor, editorial , fecha, categoria, novedad);
62         db.modificarLibro(libro);
63     }
64     ArrayList<Libro> libros = db.consultaLibros(filtro);
65     request.setAttribute("lista", libros);
66     getServletContext().getRequestDispatcher("/consulta.jsp").forward(request, response);
67 }
68 }
```