	<p align="center">UNIVERSIDAD DON BOSCO FACULTAD DE ESTUDIOS TECNOLÓGICOS</p>
<p align="center">CICLO I</p>	<p align="center">GUIA DE LABORATORIO #7 Programación Orientada a Objetos SERVLETS Y JDBC</p>

I. OBJETIVOS.

Que el estudiante

- Pueda crear Servlets con Netbeans.
- Agregue un servidor web para desarrollo de aplicaciones Cliente-Servidor
- Cree y manipule sesiones con Servlets.

II. INTRODUCCIÓN.

Un servlet es un objeto que se ejecuta en un servidor o contenedor JEE, especialmente diseñado para ofrecer contenido dinámico desde un servidor web, generalmente HTML.

Forman parte de JEE (Java Enterprise Edition), que es una ampliación de JSE (Java Standard Edition).

Un servlet implementa la interfaz `javax.servlet.Servlet` o hereda alguna de las clases más convenientes para un protocolo específico (ej: `javax.servlet.HttpServlet`). Al implementar esta interfaz el servlet es capaz de interpretar los objetos de tipo `HttpServletRequest` y `HttpServletResponse` quienes contienen la información de la página que invocó al servlet.

III. PROCEDIMIENTO.

Creación de un Servlet con Netbeans

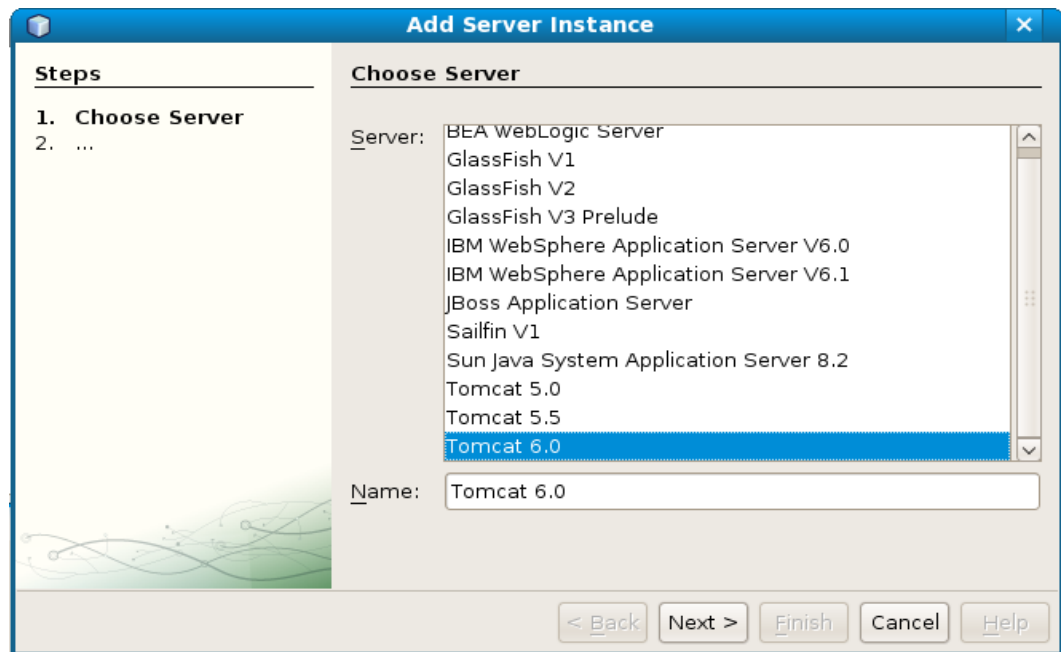
Paso 1: Agregar un contenedor Web

Existen varios servidores web tanto para desarrollo y producción. Netbeans dispone de entre sus agregados a Glassfish que fue desarrollado por Sun Microsystems (ahora perteneciente a Oracle). También se cuenta con Apache Tomcat, JBoss (proyectos de código abierto) y otras potentes opciones comerciales como lo es IBM Websphere Application Server.

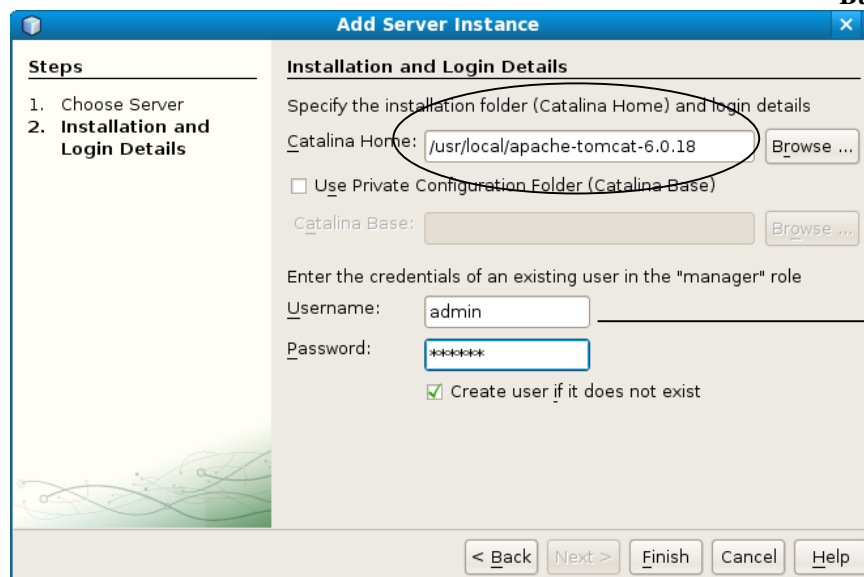
En nuestra clase podemos utilizar Apache Tomcat (<http://tomcat.apache.org/download-60.cgi>).

Nota: Seleccione el servidor adecuado en esta ventana, sea este Apache Tomcat 6.0 o 7.0

Hacer clic en la pestaña **Services**, dar clic derecho sobre la opción **Servers**, seleccionar el servidor a utilizar, para esta guía podemos utilizar Tomcat 6.0, dar click en **Next**.



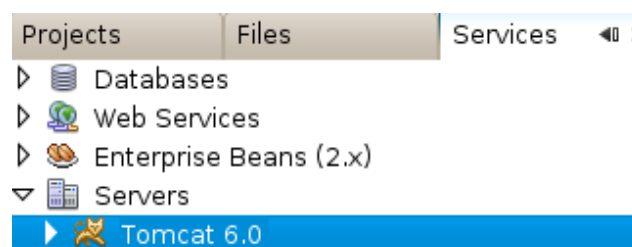
Aparecerá una pantalla como la siguiente



Buscar el directorio en el cual se encuentra el servidor.

Username: **admin**
Password: **admin**

Luego de haber ingresado todo lo necesario se habilitara el botón de **Finish**, dar click en el para que el servidor a utilizar quede agregado, así como lo muestra la siguiente figura.

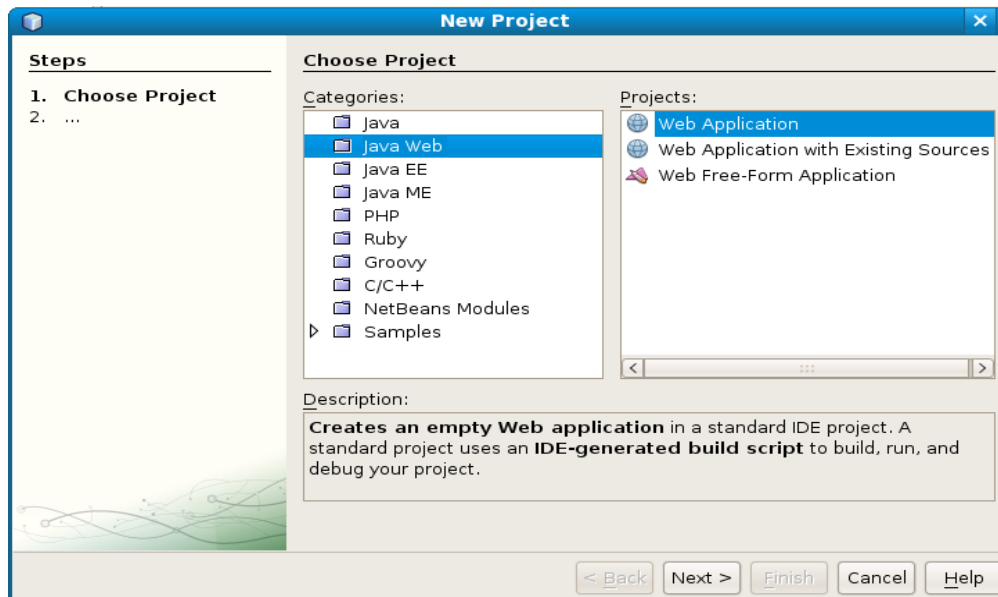


Si no cuenta con Apache Tomcat, puede disponer de Glassfish, consulte a su instructor al respecto.

Paso 2:

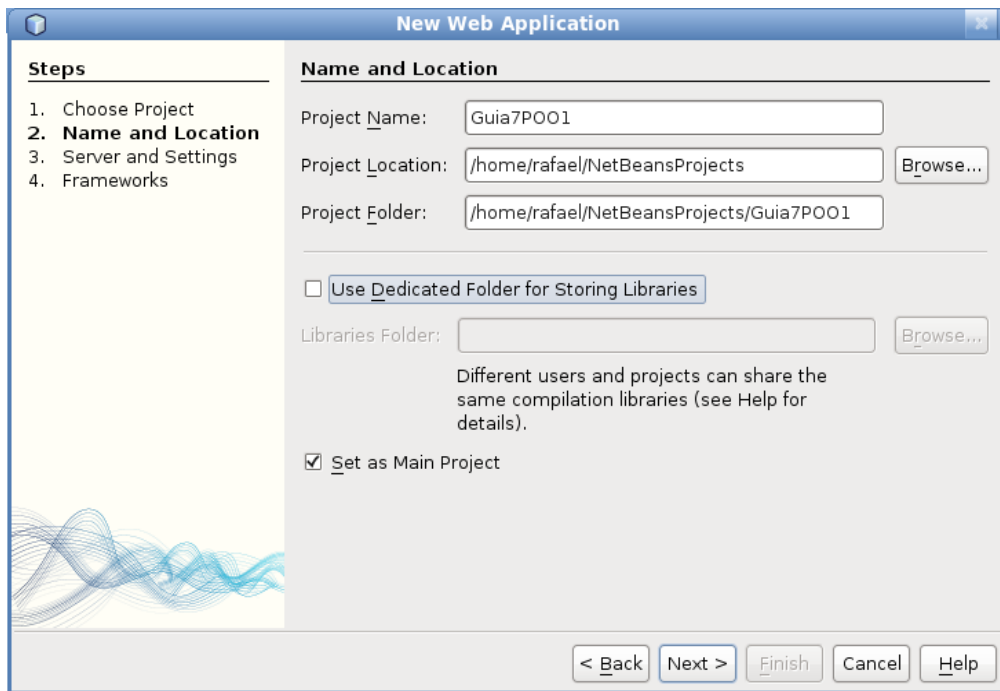
Después de haber agregado el servidor, crearemos nuestro proyecto web en el cual agregaremos nuestros servlets.

1. Iniciamos NetBeans y seleccionamos File \ New Project...
2. Se abre un diálogo que nos solicita el tipo de proyecto, seleccionamos Web Application, ver la siguiente ventana y luego dar click en **Next**.

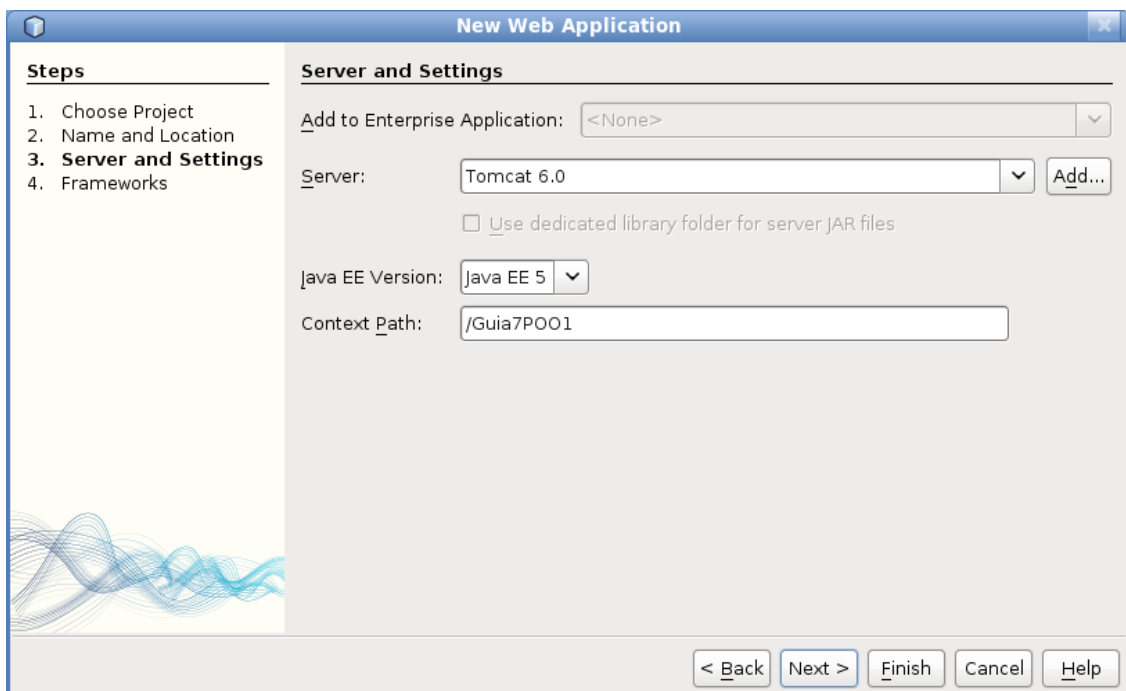


Nota: Si no dispone de Java Web, instale el plugin correspondiente.

3. Agregar al nombre del proyecto **Guia7POO1**, dar click en **Next**.



4. El siguiente paso, nos solicita el servidor web (contenedor de Servlets a utilizar). Del combo Server, seleccionamos Tomcat 6.0 o Glassfish según sea el caso.



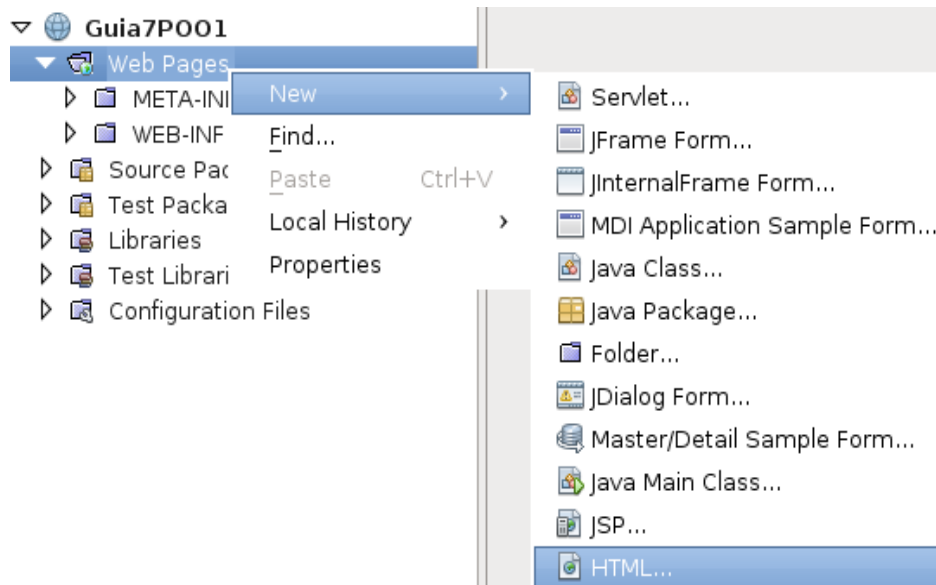
5. Finalmente, nos permite seleccionar el o los frameworks a utilizar (Spring, Struts, JSF, etc). No seleccionamos nada por ahora para este ejemplo ya que no es necesario.

NetBeans crea por su cuenta el proyecto, una estructura de directorios y dentro de la carpeta Web Pages un archivo index.jsp, que será el punto de partida de nuestra aplicación. Si bien es de extensión JSP, por ahora no escribiremos código JSP, sino simplemente un formulario HTML. En este formulario HTML definiremos en el atributo action el nombre del servlet que se ejecutará al enviar (submit) el formulario.

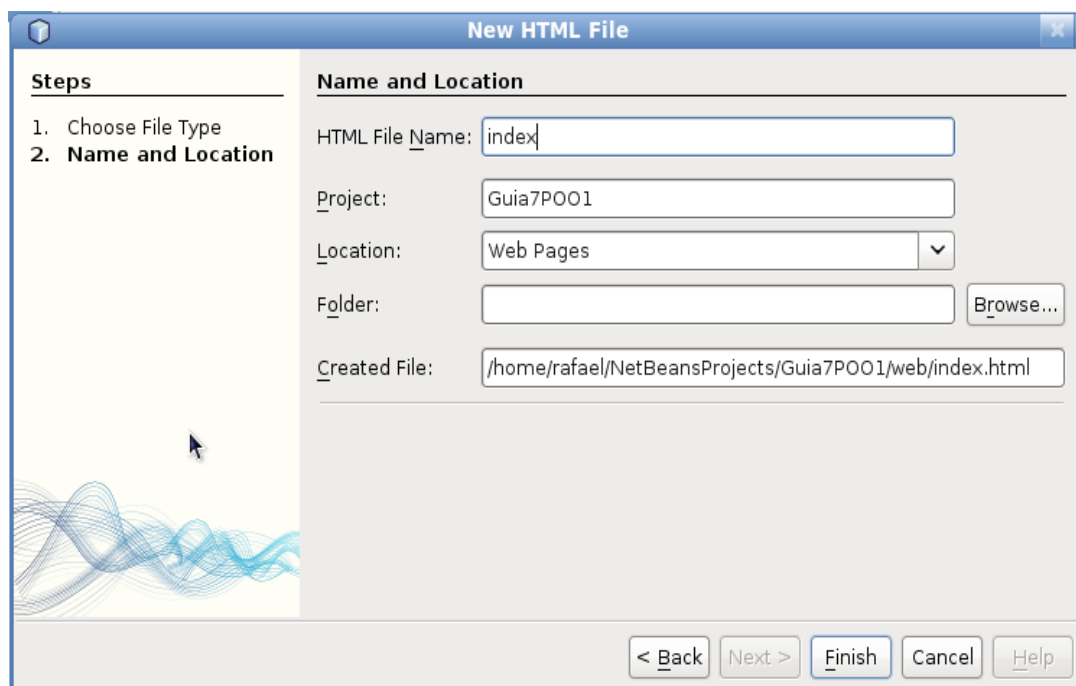
Paso 3:

Eliminación de la página index.jsp y creación de la página index.html, esto lo hacemos para que el servidor no procese la página index.jsp.

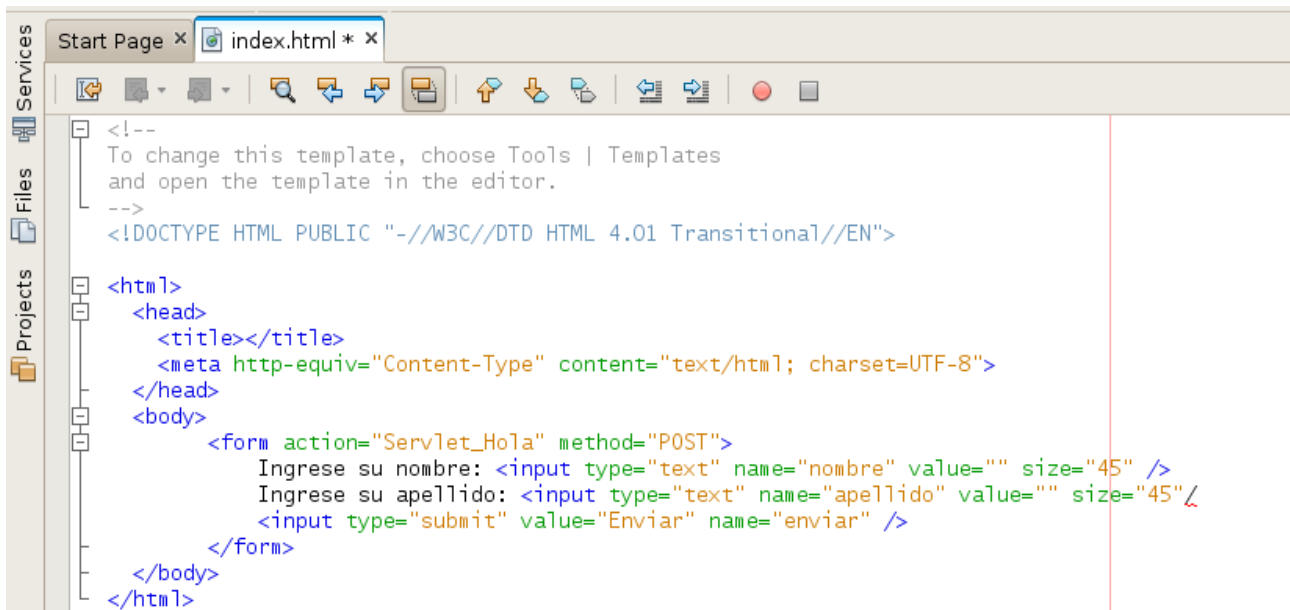
1. Situarse en la página y damos clic en la tecla suprimir.
2. Ahora crear la nueva página JSP, damos clic derecho sobre **“Web Pages”**, elegimos la opción **“New”** y seleccionamos **“HTML”** (ver la siguiente imagen).



3. Aparecerá una pantalla como la siguiente en la cual, ingresaremos como nombre de la página index, la extensión será agregada automáticamente.



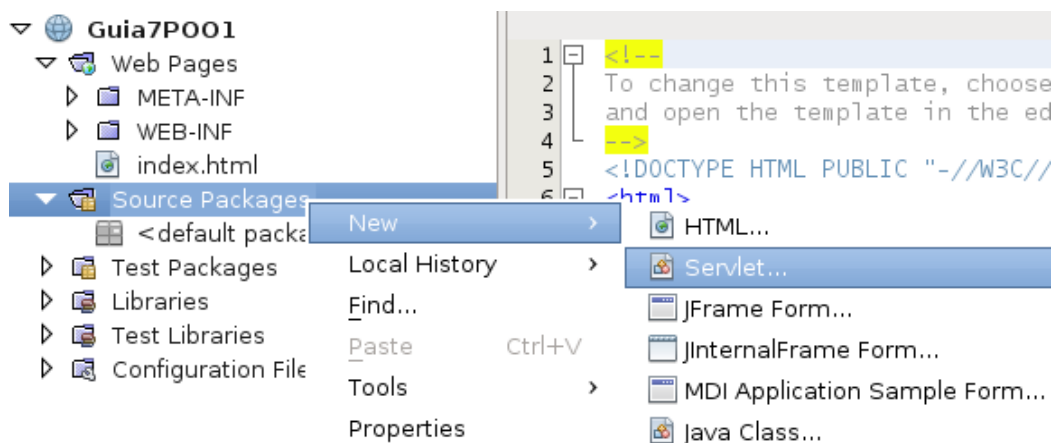
- Ahora modificaremos el código de la página index.html de tal manera que quede de la siguiente manera.



```
<!--
To change this template, choose Tools | Templates
and open the template in the editor.
-->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

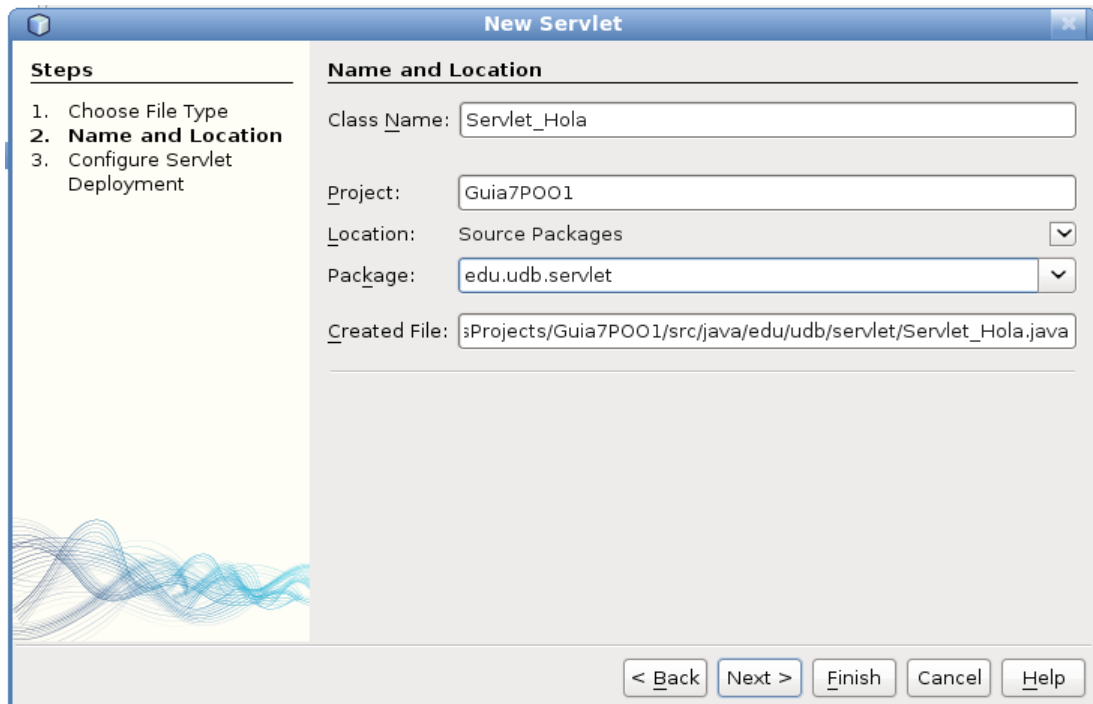
<html>
<head>
<title></title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
<form action="Servlet_Hola" method="POST">
  Ingrese su nombre: <input type="text" name="nombre" value="" size="45" />
  Ingrese su apellido: <input type="text" name="apellido" value="" size="45" />
  <input type="submit" value="Enviar" name="enviar" />
</form>
</body>
</html>
```

- Como siguiente paso, se creara el servlet, para ello, dar clic derecho en la carpeta “Source Packages”, seleccionamos New \ Servlet...



- Se abre un diálogo que nos solicita nombre y paquete del servlet.
 - En nombre, hay que ingresar el mismo nombre del atributo action del formulario creado anteriormente, pues este será el servlet que recibirá los datos enviados por el formulario HTML. En nuestro caso, según indicamos en el form: **Servlet_Hola**.
 - En paquete se puede ingresar “edu.udb.servlet”.

1. Dados el nombre del servlet y el paquete, hacemos clic sobre Finish.



Finalizado esto, automáticamente crea una clase con el nombre de servlet dado (Servlet_Hola para nosotros), que hereda de HttpServlet. Además redefine (override) algunos métodos (doGet, doPost, getServletInfo) y los rellena con un poco de código. Crea un método processRequest (invocado desde los métodos doGet y doPost) para procesar los formularios que llegan por los métodos GET y POST.

Nosotros, en este ejemplo, nos limitaremos completar con unas pocas líneas (pues la mayoría la completó automáticamente el NetBeans) en el método processRequest para que cree una salida HTML que será la respuesta del formulario enviado.

2. Ahora modificaremos un poco el servlet, para ello buscar el método “**processRequest**” y modificamos el código que se encuentra dentro del “**try**”, quedando de la siguiente manera.

```
try {  
  
    out.println("<html>");  
    out.println("<head>");  
    out.println("<title>Servlet Servlet_Hola</title>");  
    out.println("</head>");  
    out.println("<body>");  
    out.println("<h1>Resultado de Servlet_Hola</h1>");  
    out.println("<p>");  
    out.println("<b>Nombre de la persona: </b>" +  
        request.getParameter("nombre").toString()+"<br>");  
    out.println("<b>Apellido de la persona:
```

```
</b>" + request.getParameter("apellido").toString());  
out.println("</p>");  
out.println("</body>");  
out.println("</html>");  
  
}
```

Paso 4:

Proceda a correr la aplicación (proyecto) podemos hacerlo desde el menú **Run** o haciendo clic derecho en el ícono del proyecto (desde el explorador de proyectos) seleccionando el menú contextual y seleccionando **Run**.

- Al ejecutar el proyecto se abrirá el browser predeterminado con la página index.html (la que tiene el formulario):
- Si ingresamos nuestro nombre en la caja de texto y damos click en Enviar, el formulario se envía al servlet, quien se ejecuta y nos devuelve una nueva página con un dato en particular cargado dinámicamente, con los valores ingresados en el formulario del index.html.

Servlet con JDBC

Cree la siguiente base de datos (Solo ejecute las sentencias, estas tablas ya las ha trabajado en las guías 4 y 6):

Paso 1:

```
create database Guia7_POO1;  
  
use Guia7_POO1;  
  
create table Empleados  
(Codigo int primary key,  
  Nombre varchar(25),  
  Apellidos varchar(25),  
  Telefono varchar(9)  
);  
  
create table tipo_usuarios(  
  id_tipo_usuario int primary key,  
  tipo_usuario varchar(25));  
  
create table usuarios(  
  id_usuario int primary key,  
  nombres varchar(25),  
  apellidos varchar(25),  
  edad int,  
  id_tipo_usuario int,  
  usuario varchar(20),  
  password varchar(30),  
  constraint foreign key (id_tipo_usuario) references tipo_usuarios(id_tipo_usuario)  
);  
  
insert into tipo_usuarios values (0,'Tipo Usuario 1');  
insert into tipo_usuarios values (1,'Tipo Usuario 2');
```



```
insert into usuarios values (
0,'Root','No Last Name for root',1,0,'root','root'
);
insert into usuarios values (
1,'Nikola','Tesla',34,1,'tesla','corrienteAC'
);
```

Paso 2:

Para este punto crearemos una página jsp llamada **“ingresaremp.html”**, la cual contendrá el siguiente código.

```
<html>
<head>
<title>Ingresar Empleados</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
  <form action="Servlet_Ingresaremp" method="POST">
    Ingrese su codigo: <input type="text" name="codigo" value="" size="10"><br>
    Ingrese sus nombres: <input type="text" name="nombre" value="" size="45" /><br>
    Ingrese sus apellido: <input type="text" name="apellido" value="" size="45"/><br>
    Ingrese su Telefono: <input type="text" name="telefono" value="" size="10" /><br>
    <input type="submit" value="Enviar" name="enviar" /><br>
  </form>
</body>
</html>
```

Paso 3:

Ahora crea un Servlet en el paquete creado anteriormente (**edu.udb.servlet**), llamado **“Servlet_Ingresaremp”**. De este servlet solo tendrá que modificar el método **“processRequest”** y quedara de la siguiente manera (No olvide agregar el driver de **MySQL** al proyecto y además debe de importar la librería **java.sql.*** en la sección de imports)

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();

    ResultSet rs = null;
    Connection conexion = null;
    String ids=request.getParameter("codigo");
    String nombre=request.getParameter("nombre");
    String apellido=request.getParameter("apellido");
    String telefono=request.getParameter("telefono");
    try {
        //Leemos el driver de MySQL
        Class.forName("com.mysql.jdbc.Driver");

        // Se obtiene una conexión con la base de datos.
```

```

conexion = DriverManager.getConnection (
    "jdbc:mysql://localhost/Guia7_POOI", "root", "");

// Permite ejecutar sentencias SQL sin parámetros
Statement s = conexion.createStatement();
s.executeUpdate("Insert into Empleados "
    + "values("+ids+",\"" + nombre+"\", \"" + apellido+"\", \"" + telefono+"\)");

rs = s.executeQuery ("select * from Empleados");
//Decimos que nos hemos conectado
out.println("<html>");
out.println("<body>");
out.println("<h1>Datos Ingresados Exitosamente</h1>");

out.println("<table align='center' with='75%' border=1>");
out.println("<tr><th>Codigo</th><th>Nombres</th><th>Apellidos"+
    "</th><th>Telefono</th></tr>");
while (rs.next()){
    out.println("<tr><td>" + rs.getInt("Codigo") + "</td><td>" +
        rs.getString("Nombre") + "</td><td>" + rs.getString("Apellidos") + "</td><td>" +
        rs.getString("Telefono") + "</td></tr>");
}
out.println("</table>");
out.println("</body>");
out.println("</html>");

conexion.close();
}
catch (ClassNotFoundException e1) {
    //Error si no puedo leer el driver
    out.println("ERROR:No encuentro el driver de la BD: " +
        e1.getMessage());
}
catch (SQLException e2) {
    //Error SQL: login/passwd mal
    out.println("ERROR:Fallo en SQL: " + e2.getMessage());
}
finally {
    out.close();
}
}

```

Paso 4

Ahora lo único que falta es seleccionar la opción del menú contextual **Run** a la página “**ingresaremp.html**”, debe aparecer un formulario el cual debe ser llenado. Al enviar permitirá que el servlet se conecte a la base de datos y pueda insertar los valores digitados en el formulario.

Manejo de Sesiones

Paso 1:

Crear una nueva página html llamada “**login.html**” que contendrá el siguiente código:

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Login!!!</h1>
    <form action="GeneraSession" method="POST">
      Ingrese su Usuario: <input type="text" name="usuario" value="" size="45" /><br>
      Ingrese su Password: <input type="password" name="password" value=""
size="45"/><br>
      <input type="submit" value="Enviar" name="enviar" />
    </form>
  </body>
</html>
```

Como se puede observar en el **action** debe ir el nombre del servlet que genera la sesión.

Paso 2:

Crear una clase Conexión dentro del paquete edu.udb.servlet, agregar el siguiente código:

Nota: Es la misma clase Conexion de la guía 6 no es necesario que digite todo el código, incluya el archivo o copie y pegue el contenido.

```
package edu.udb.servlet;

import java.sql.*;

public class Conexion {
  private Connection conexion =null;
  private Statement s =null;
  private ResultSet rs=null;
  private String query="";

  //Constructor
  public Conexion() throws SQLException{
    try
    {
      //obtenemos el driver de para mysql
      Class.forName("com.mysql.jdbc.Driver");
      // Se obtiene una conexión con la base de datos. 2
      conexion = DriverManager.getConnection (
        "jdbc:mysql://localhost/Guia7_POO1","root", "");
      // Permite ejecutar sentencias SQL sin parámetros
    }
  }
}
```

```

        s = conexion.createStatement();
    }
    catch (ClassNotFoundException e1) {
        //Error si no puedo leer el driver de MySQL
        System.out.println("ERROR:No encuentro el driver de la BD: " +e1.getMessage());
    }
}
//Metodo que permite obtener los valores del resulset
public ResultSet getRs() {
    return rs;
}
//Metodo que permite fijar la tabla resultado de la pregunta
//SQL realizada
public void setRs(String consulta) {
    try {
        this.rs = s.executeQuery(consulta);
    } catch (SQLException e2) {
        System.out.println("ERROR:Fallo en SQL: "+e2.getMessage());
    }
}

//Metodo que recibe un sql como parametro que sea un update,insert.delete
public void setQuery(String query) throws SQLException {
    this.s.executeUpdate(query);
}

//Metodo que cierra la conexion
public void cerrarConexion() throws SQLException{
    conexion.close();
}
}

```

Paso 3:

Creamos el Servlet con el nombre “**GeneraSession**”, el paquete será el mismo utilizado en los puntos anteriores, modificamos el método **processRequest** para que quede de la siguiente manera (Nota: No olvide incluir el paquete java.sql.* también es necesario javax.servlet.http.HttpSession - puede usar fix imports-):

```

protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        String usuario=request.getParameter("usuario");
        String password=request.getParameter("password");

        try{
            Conexion con = new Conexion();

            //buscará una coincidencia (count usuario), si es correcto

```

```

//podrá loguearse
con.setRs("select count(usuario),nombres from usuarios"
+ " where usuario='"+usuario+"' and "
+ "password='"+password+"'");

ResultSet rs = con.getRs();

rs.next();

if(rs.getInt(1)==1){ //solo una coincidencia es permitida
    HttpSession session_actual=request.getSession(true);
    session_actual.setAttribute("USER", usuario);
    session_actual.setAttribute("NAME", rs.getString(2));
    response.sendRedirect("principal.jsp");
}
else{
    response.sendRedirect("login.html");
}

rs.close();
con.cerrarConexion();

}catch(SQLException e){
    out.print(e.getMessage());
}

} finally {
    out.close();
}
}

```

En este Servlet se encargada de recoger del usuario y la clave enviados desde el formulario. Una vez recibidos se almacenan en dos variables (“usuario” y “password”) de tipo String. A continuación se comparan con los valores correctos del usuario y la clave desde la base de datos. Si esta comprobación es correcta se crea un objeto de tipo session y se guarda el valor de usuario en la variable “USER” y los nombres en la variable “NAME” para la sesión mediante el método setAttribute().

A continuación y mediante la opción response.sendRedirect(**"login.html"**), se re direcciona al usuario a la página pasada por parámetro en caso no pueda acceder.

Paso 3:

Ahora crearemos la página jsp que verificar si la sesión esta activa, la página será llamada **“principal.jsp”** y de be quedar de la siguiente manera.

```

<%

HttpSession session_actual=request.getSession(false);
String usuario =(String) session_actual.getAttribute("USER");
String nombres =(String) session_actual.getAttribute("NAME");

```

```
        if (usuario==null){
            response.sendRedirect("login.html");
        }
    %>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
</head>
<body>
    <h1>Hello World!</h1>
    <h2>
        Bienvenido: (<%=usuario%>) <%=nombres%>
    </h2>
</body>
</html>
```

Finalmente puede probar loguearse con los usuarios:

Usuario: root y password: root

Usuario: tesla y Password: corrienteAC

Puede agregar más usuarios.

IV. ANALISIS DE RESULTADOS.

- Tomando de base el proyecto de periodo, realizar el mantenimiento, para poder agregar libros, categorías, debe venir completamente validado con javascript puro, no utilizar framework ni tampoco deberá utilizar html 5, para validar campos requeridos.

HOJA DE EVALUACIÓN

Carnet:	Alumno:
Fecha:	Docente:
No.:	Título de la guía:

Actividad a evaluar	Criterio a evaluar	Cumplió		Puntaje
		SI	NO	
Desarrollo	Realizó los ejemplos de guía de práctica (40%)			
	Presentó todos los problemas resueltos (20%)			
	Funcionan todos correctamente y sin errores (30%)			
	Envío la carpeta comprimida y organizada adecuadamente en subcarpetas de acuerdo al tipo de recurso (10%)			
	PROMEDIO:			
Investigación complementaria	Entregó la investigación complementaria en la fecha indicada (20%)			
	Resolvió todos los ejercicios planteados en la investigación (40%)			
	Funcionaron correctamente y sin ningún mensaje de error a nivel de consola o ejecución (40%)			
	PROMEDIO:			