	**Escuela Superior de Empresa, Ingeniería y Tecnología.** **Ingeniería Informática**
CICLO I	**GUIA DE LABORATORIO #7** **Ingeniería de Software** **Servlets y JDBC**

I. OBJETIVOS

Que el estudiante

- Pueda crear Servlets con Nebeans.
- 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.

Pero antes, Qué es un Servlet de Java?

Un Servlet es un objeto de java que pertenece a una clase que extiende de javax.servlet.http.HttpServlet, este nos permite crear aplicaciones web dinámicas, lo que quiere decir que podemos realizar consultas, insertar y eliminar datos.

Son pequeños programas (applets) escritos en Java que admiten peticiones a través del protocolo HTTP. Los servlets reciben peticiones desde un navegador web, las procesan y devuelven una respuesta al navegador, normalmente en HTML. Para realizar esto pueden utilizar las herramientas del lenguaje Java.

Qué es un contenedor de Servlets?

Un contenedor de Servlet es un programa capaz de recibir peticiones de páginas web y redireccionar estas peticiones a un objeto Servlet en específico.

Uno de los más populares es Apache Tomcat.

Cómo funcionan los contenedores de Servlets?

- ✓ El Browser pide una página al servidor HTTP que es un contenedor de Servlets.
- ✓ El contenedor de Servlets delega la petición a un Servlet en particular elegido de entre los Servlets que contiene.
- ✓ El Servlet, que es un objeto java, se encarga de generar el texto de la página web que se entrega al contenedor.
- ✓ El contenedor devuelve la página web al Browser que la solicitó.

III. PROCEDIMIENTO

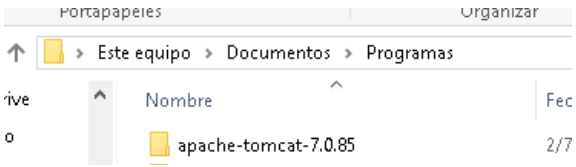
Creación de un Servlet con Netbeans

Paso 1: Descargar Tomcat 7.0

Existen varios servidores web tanto para desarrollo y producción. Netbeans dispone de entre sus agregados a Glassfish que fue desarrollado por Sun Microsystem (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.

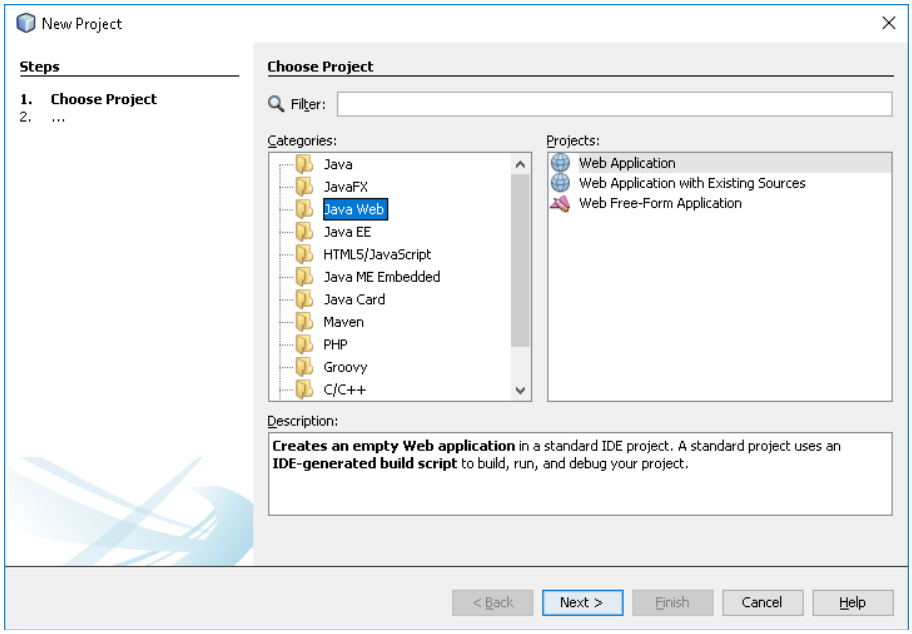
En nuestra clase podemos utilizar Apache Tomcat o superior (https://tomcat.apache.org/download-70.cgi).

Descargarlo y descomprimirlo en una carpeta

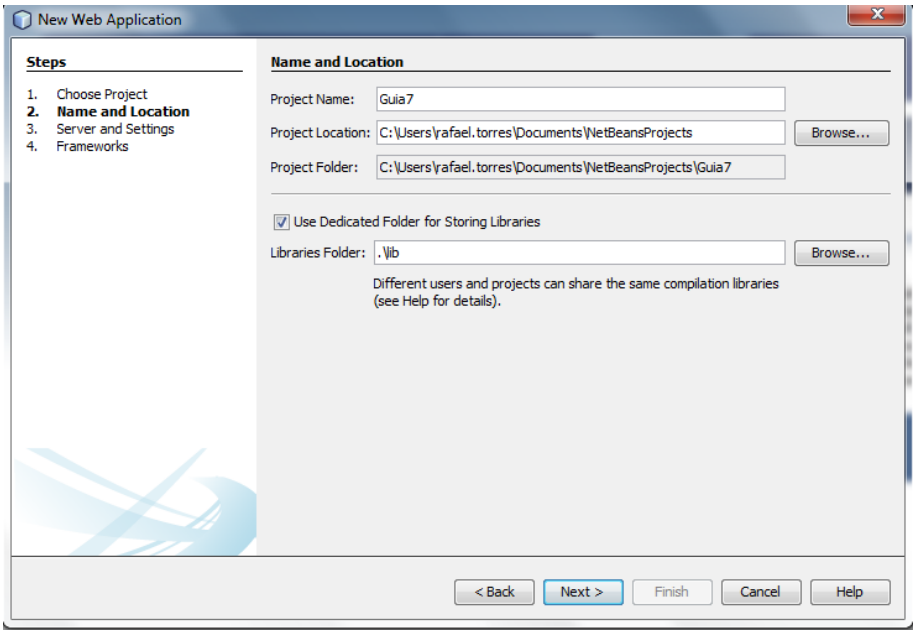


Paso 2: Crear un proyecto Java Web

1. El primer paso es generar un nuevo proyecto, **File-->New Project…**

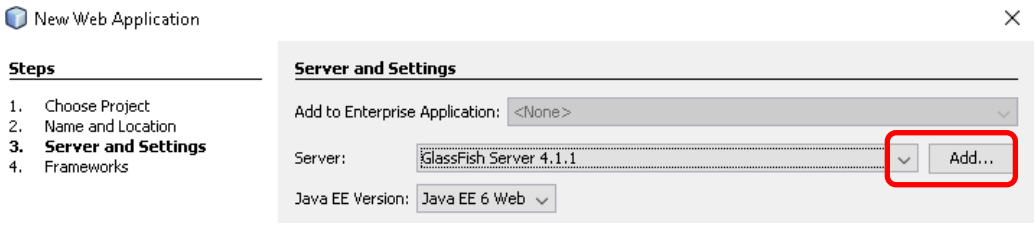


2. En la sección de categorías seleccionamos “Java Web” y en la sección de proyectos seleccionamos “Web Application”.
3. Se nombra el proyecto, en este ejemplo: “Guia7”
4. Hacer clic en el botón Next

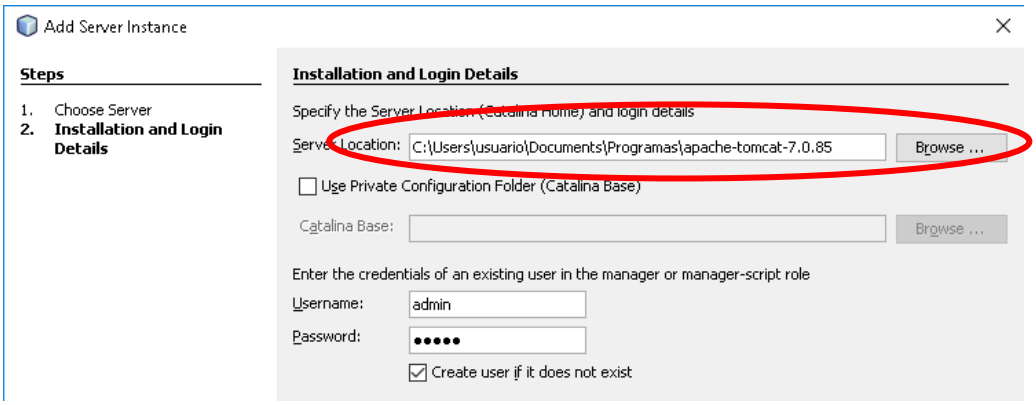


Agregar un contenedor Web

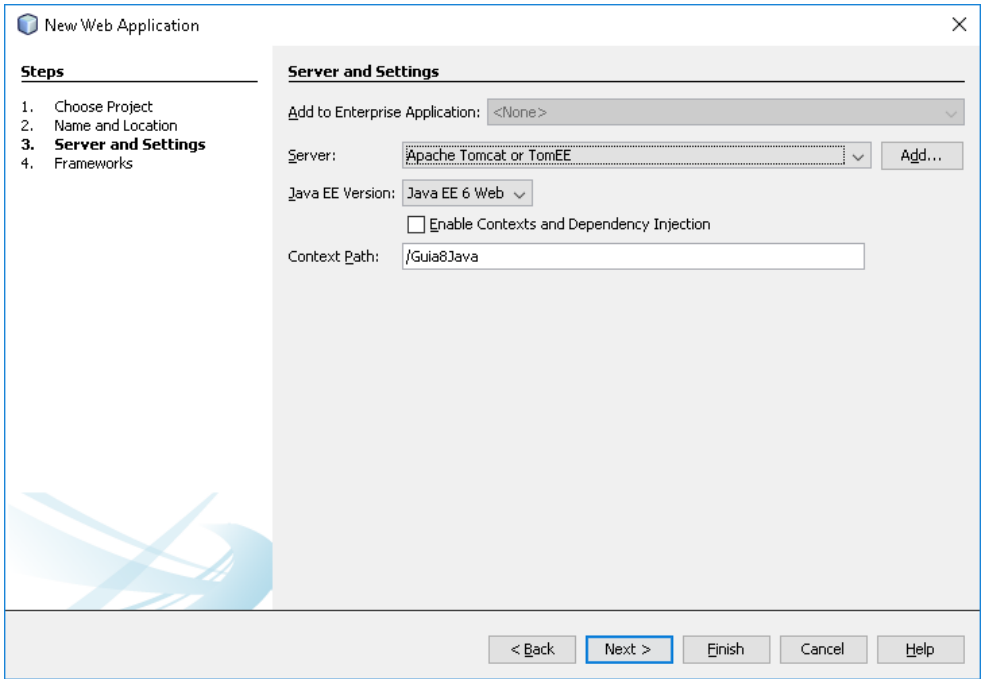
1. En la opción Server and Settings
2. Hacer clic en botón Add



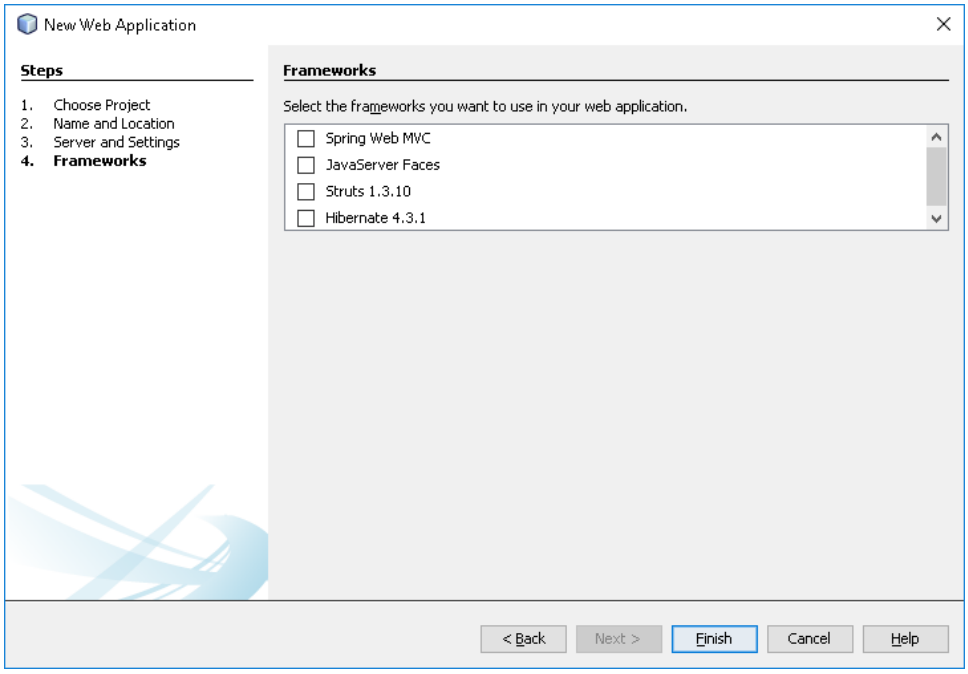
3. De la lista de servidores seleccionar ApacheTomcat or TomEE
4. Hacer clic en Next
5. Buscar la dirección de la carpeta de ApacheTomcat



6. Hacer clic en Finish
7. Se observará la siguiente ventana:



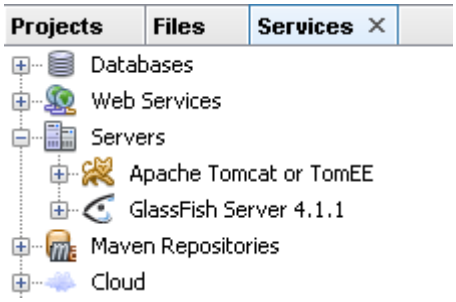
8. Hacer clic en Next
9. Finalmente, nos permite seleccionar el o los frameworks a utilizar (Spring, Struts, JSF, etc). No seleccionamos nada por ahora para este ejemplo ya que no es necesario.
   NetBeans crea por su cuenta el proyecto, una estructura de directorios y dentro de la carpeta Web Pages 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.



10. Hacer clic en Finish

11. Para verificar que el servidor a utilizar quedo agregado, verifique como se muestra en la

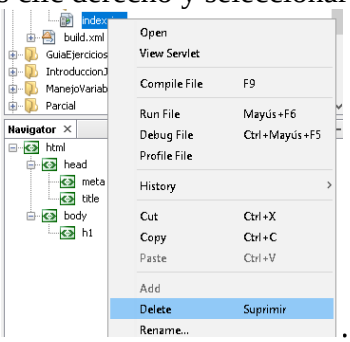
siguiente figura:



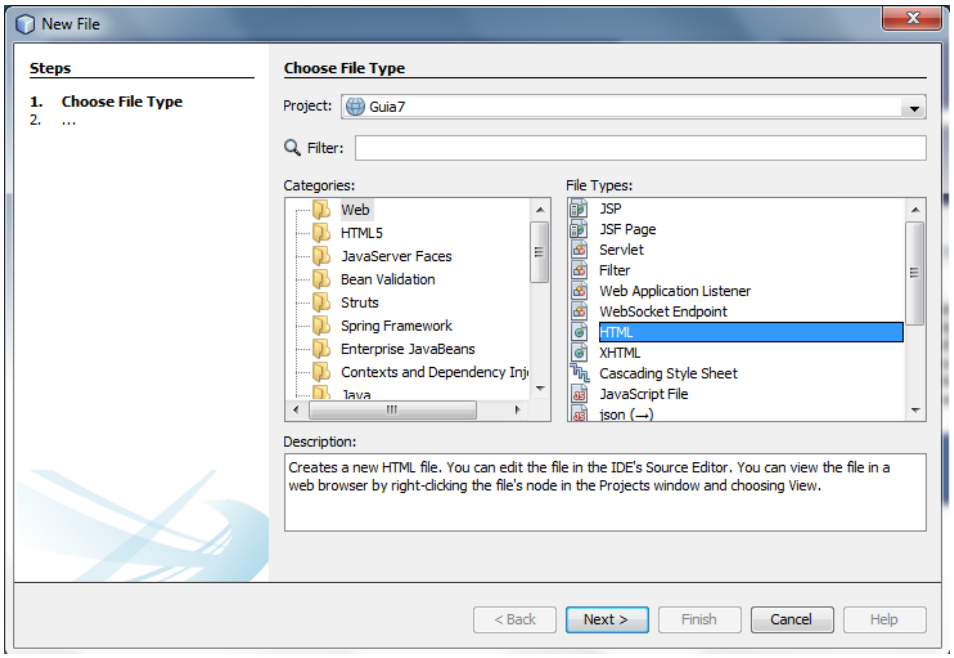
Si no cuenta con Apache Tomcat, puede disponer de Glassfish, consulte a su instructor al respecto.

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 derecho y seleccionar la opción Delete

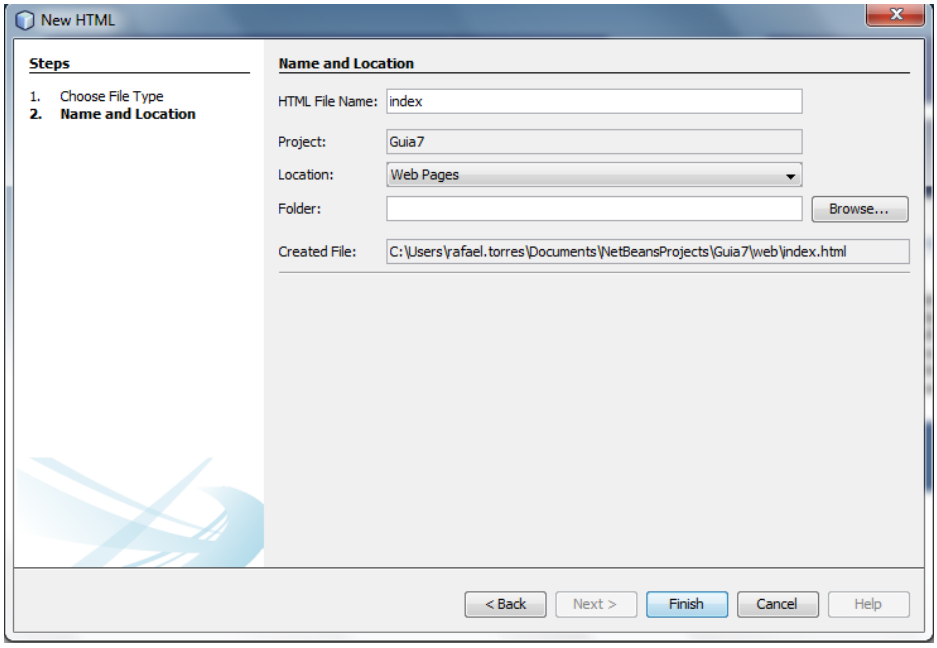


2. Hacer clic derecho sobre **"Web Pages",** elegimos la opción **"New"** y seleccionamos **"HTML"** (ver la siguiente imagen).



3. Hacer clic en Next

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

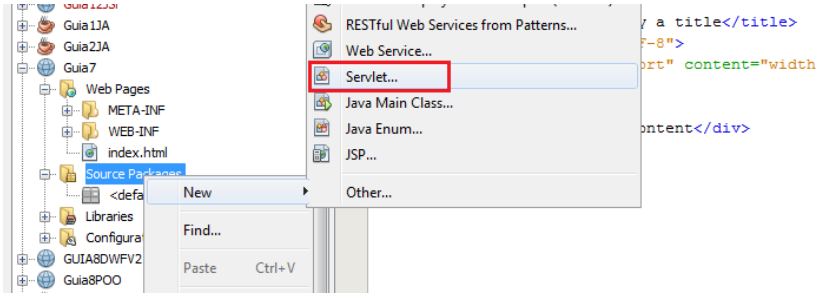


5. Hacer clic en Finish

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

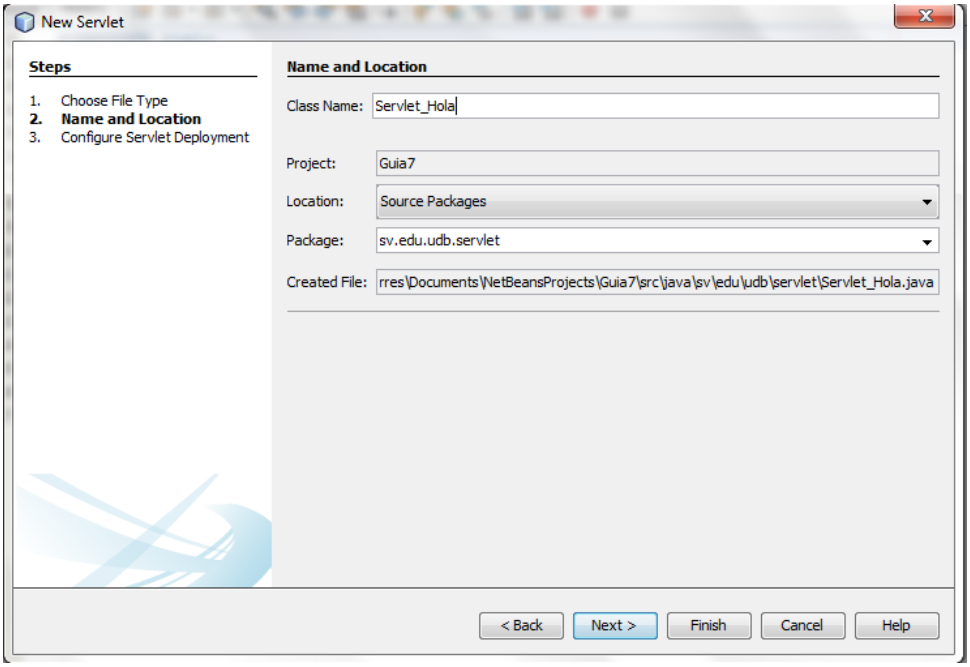
```
<html>
    <head>
        <title>Uso de Servlet</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <form action="Servlet_Hola" method="POST">
            <table border="0">
                <tbody>
                    <tr>
                        <td><label for="nombre">Ingrese su Nombre: </label></td>
                        <td><input type="text" name="nombre" id="nombre" value="" /></td>
                    </tr>
                    <tr>
                        <td><label for="apellido">Ingrese su Apellido</label></td>
                        <td><input type="text" name="apellido" id="apellido" value="" /></td>
                    </tr>
                    <tr>
                        <td colspan="2">
                            <input type="submit" value="Enviar" />
                            <input type="reset" value="Limpiar" />
                        </td>
                    </tr>
                </tbody>
            </table>

        </form>
    </body>
</html>
```

7. Como siguiente paso, se creara el servlet, para ello,dar clic derecho en la carpeta "**Source Packages**", seleccionamos **New \ Servlet...**

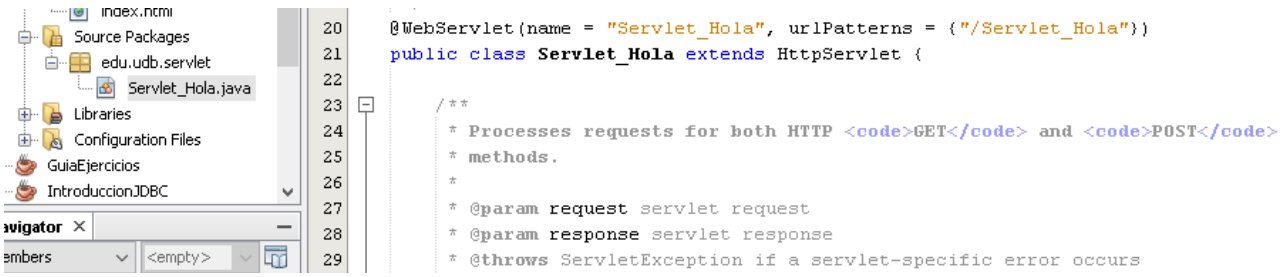


8. Se abre un diálogo que nos solicita nombre y paquete del servlet.
   - En nombre, hay que ingresar el mismo nombre del atributo action del formulario creado anteriormente, pues este será el servlet que recibirá los datos enviados por el formulario HTML. En nuestro caso, según indicamos en el form: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

9. 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.

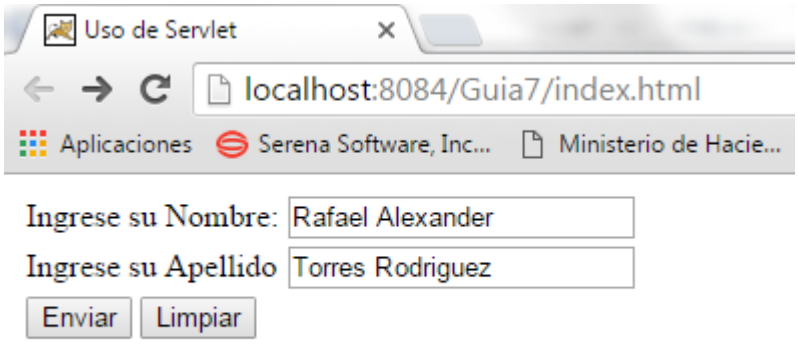
10. Ahora modificaremos un poco el servlet, para ello buscar el método **"processRequest"** y modificamos el código que se encuentra detro del **"try"**, quedando de la siguiente manera.

```
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 clic 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.

Formulario de la página index.html



Resultado del Servlet

Servlet con JDBC

Paso 1:

Cree la siguiente base de datos.

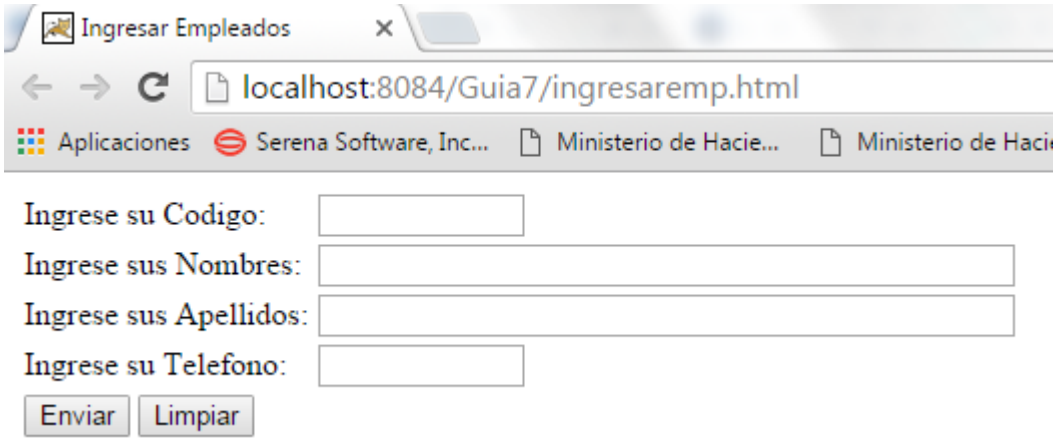
```
create database Guia7;

use Guia7;

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.

Principio del formulario

```
<html>
<head>
<title>Ingresar Empleados</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
<form action="ServletIngresarEmp" method="POST">
<table border="0">
<tbody>
<tr>
<td><label for="codigo">Ingrese su Codigo: </label></td>
<td><input type="text" name="codigo" id="codigo" value="" size="10"></td>
</tr>
<tr>
<td><label for="nombre">Ingrese sus Nombres: </label></td>
<td><input type="text" name="nombre" id="nombre" value="" size="45" /></td>
</tr>
<tr>
<td><label for="apellido">Ingrese sus Apellidos: </label></td>
<td><input type="text" name="apellido" id="apellido" value="" size="45"/></td>
</tr>
<tr>
<td><label for="telefono" >Ingrese su Telefono: </label></td>
<td><input type="text" name="telefono" id="telefono" value="" size="10" /></td>
</tr>
<tr>
<td colspan="2">
<input type="submit" value="Enviar" />
<input type="reset" value="Limpiar" />
</td>
</tr>
</tbody>
</table>
</form>
</body>
</html>
```

Al ver el formulario se debe visualizar similar al siguiente.



Paso 3:
Ahora crea un Servlet en el paquete creado anteriormente (**edu.udb.servlet)**, llamado **"ServletIngresarEmp".** 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)

```
protectedvoid 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","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 opcióndel 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:

```
package edu.udb.servlet;


import java.sql.*;

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

//Contructor
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","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
publicvoid 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
publicvoid setQuery(String query) throws SQLException {
this.s.executeUpdate(query);
    }

//Metodo que cierra la conexion
publicvoid 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-):

```
protectedvoid 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 +" group by nombres'");

            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 redirecciona 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 páginaserá 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 nuevos usuarios de tipo Jefe de Sistemas y para poder agregar nuevos casos, debe venir completamente validado con javascript puro, no utilizar framework ni tampoco deberá utilizar html 5, para validar campos requeridos.