AUT03_02. Actividad BD Empleados

1. Objetivos

Los objetivos de esta actividad son:

- Repasar métodos del ciclo de vida de Servlet
- Repasar formularios y peticiones de tipo GET/POST
- Conocer los fundamentos de JDBC

2. Crear aplicación Empleados

Se creará una aplicación Maven de tipo Web usando Java Enterprise Edition 7 y Wildfly con nombre AUT03_02_ApellidoNombre (proyecto). Subir AUT03_02_Apellido1Apellido2Nombre.zip

2.1 Crear formulario de búsqueda HTML

Será una página **index.html** con un formulario de búsqueda. En este caso con método **GET**.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Buscador</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <h1>Buscador de empleados</h1>
        <form action="/EmpleadosApellidosNombre/buscadorEmpleados" method="GET">

            <fieldset>
                <legend>Datos empleado</legend>
                <label for="txtNombre">Nombre</label>
                <input id="txtNombre" type="text" name="nameToSearch"/>
                <input type="submit" value="Enviar"/>
```

```
            </fieldset>

        </form>

    </body>

</html>
```

2.2 Fichero .properties de configuración de la conexión a base de datos

Se creará un fichero **empleados.properties** que se ubicará en la carpeta raíz de "source packages". Su contenido será el siguiente:

```
database.driver=com.mysql.jdbc.Driver

database.url=jdbc:mysql://localhost:3306/employees?
profileSQL=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&server
Timezone=UTC

database.user=2daw

database.password=2daw

database.pageSize=10
```

- database.driver. Nombre cualificado de la clase Java del conector a la base de datos.
- database.url. Cadena de conexión a la base de datos.
- database.user. Usuario de la base de datos.
- database.password. Contraseña del usuario de la base de datos.
- database.pageSize. Número máximo de registros por consulta.

2.3 Web component de tipo servlet que realizará las búsquedas de empleados

Se creará un servlet, mapeado en la URL **/buscadorempleados**, cuyo nombre de clase plenamente cualificado será: **es.cifpcm.empleados.buscador.web.BuscadorEmpleadosServlet**.

Este servlet, definirá un parámetro de inicialización que será usado en la conexión a la base de datos.

i. app.config → empleados

Esto quiere decir que tendremos un fichero empleados.properties con la configuración de la aplicación.

El servlet, en el método init, leerá las propiedades del fichero de configuración, usando un ResourceBundle. En el método init cargará la clase del driver, mediante Class.forName utilizando la propiedad driver.classname del fichero properties referido.

Ejemplo:

```
@Override
```

```
public void init(ServletConfig config) throws ServletException {
    super.init(config);

    try {
        String configBundleName = config.getInitParameter("app.config");
        ResourceBundle rb = ResourceBundle.getBundle(configBundleName);
        this.dbUrl = rb.getString("database.url");
        this.dbUser = rb.getString("database.user");
        this.dbPassword = rb.getString("database.password");
        this.dbPageSize = rb.getString("database.pageSize") == null
                                                    ?   DEFAULT_PAGESIZE   :
Integer.parseInt(rb.getString("database.pageSize"));
        String driverClassName = rb.getString("database.driver");

        Class.forName(driverClassName);
    } catch (ClassNotFoundException ex) {
          Logger.getLogger(BuscadorEmpleadosServlet.class.getName()).log(Level.SEVERE,
null, ex);
    }
}
```

2.4 Clase Java de tipo (POJO) para almacenar datos de empleados

POJO es un acrónimo que significa **P**lain **O**ld **J**ava **O**bject. Hace referencia a una clase Java muy simple, que únicamente contiene datos, en forma de propiedades de instancia y simplemente métodos get/set para acceder o modificar las propiedades.

```
public class Employee {

    private Integer empNo;
    private Date birthDate;
    private String firstName;
    private String lastName;
```

```
    private Character gender;

    private Date hireDate;


    public Employee() {

    }


    public Integer getEmpNo() {

        return empNo;

    }


    public void setEmpNo(Integer empNo) {

        this.empNo = empNo;

    }


    public Date getBirthDate() {

        return birthDate;

    }

...
```

2.5 Código del servlet

El servlet realizará lo siguiente:

1. El servlet en este primer paso recogerá el valor del campo nameToSearch, aunque no lo utilizará por ahora en la consulta SQL.
2. Obtendrá una conexión del DriverManager, utilizando la cadena de conexión JDBC, así como el nombre de usuario y contraseña de empleados.properties.
3. A partir de la conexión creará un objeto Statement.
4. A través del método setMaxRows del objeto Statement se establecerá el número máximo de registros devueltos por consulta.
5. Se ejecutará el statement con la siguiente consulta SQL: “SELECT * FROM employees”, asígnándose el resultado a un objeto de tipo ResultSet.

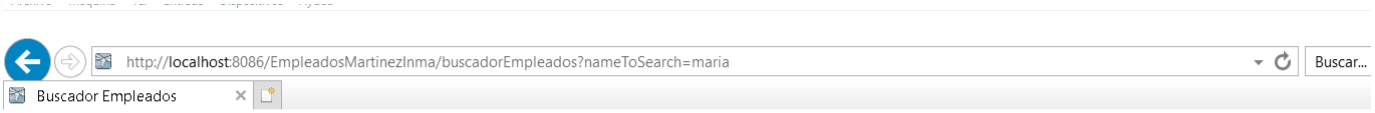
6. Se recorrerá el resultset obteniendo los datos de cada empleado y almacenando los empleados en una colección de tipo ArrayList.
7. Se cerrarán objetos de base de datos.
8. Se devolverá un HTML de respuesta, mostrando en una tabla los datos de los empleados.

   NOTA: Se utilizará una hoja de estilos CSS.

2.6 Ejemplo de petición y respuesta

2.6.1 Página de búsqueda

2.6.2 Página de resultados



Servlet BuscadorEmpleadosServlet at /EmpleadosMartinezInma

Usted está buscando a maria

Id Cliente	Nombre	Apellido
10002	Bezalel	Simmel
10003	Parto	Bamford
10004	Chirstian	Koblick
10005	Kyoichi	Maliniak
10006	Anneke	Preusig
10007	Tzvetan	Zielinski
10008	Saniya	Kalloufi
10009	Sumant	Peac
10010	Duangkaew	Piveteau