



**INSTITUTO POLITÉCNICO NACIONAL  
ESCUELA SUPERIOR DE CÓMPUTO**



**ALUMNO:** Meza Vargas Brandon David.

**PRÁCTICA:** Práctica No. 7.

**TEMA:** JDBC.

**OPCIÓN:** Opción 1, Insertar Perros en una base de datos.

**FECHA:** 07-01-21

**GRUPO:** 2CM1

**MATERIA:** Programación Orientada a Objetos

## **INTRODUCCIÓN**

Java Database Connectivity (JDBC) es una interfase de acceso a bases de datos estándar SQL que proporciona un acceso uniforme a una gran variedad de bases de datos relacionales. JDBC también proporciona una base común para la construcción de herramientas y utilidades de alto nivel.

Para usar JDBC con un sistema gestor de base de datos en particular, es necesario disponer del driver JDBC apropiado que haga de intermediario entre ésta y JDBC. Dependiendo de varios factores, este driver puede estar escrito en Java puro, o ser una mezcla de Java y métodos nativos JNI (Java Native Interface).

## **DESARROLLO**

Para esta práctica se escogió la opción numero 1, la cual consiste en insertar perros en una base de datos, se insertará su nombre, raza, edad y género.

Primeramente, creamos la interfaz gráfica, la cual consta de 4 campos de entrada con sus labels correspondientes para ingresar los datos que se almacenaran en la base de datos, esta parte no se explicará con mas detalle pues es una interfaz sencilla cuyos elementos ya se han explicado en prácticas anteriores, el código lo vemos en la figura 1;

```

public Practica7() {
    super("Insertar perros a BD");

    n= new JLabel("Nombre: ");
    r= new JLabel("Raza: ");
    e= new JLabel("Edad: ");
    g= new JLabel("Genero: ");

    no=new JTextField();
    ra= new JTextField();
    ed=new JTextField();
    ge= new JTextField();

    con= new JButton("Conectar");
    ins= new JButton("Insertar");
    con.addActionListener(this);
    ins.addActionListener(this);
    setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    setSize(450,300);
    setVisible(true);
    init();
}

void init() {
    setLayout(null);
    n.setBounds(5,15,60,10); add(n);
    no.setBounds(80,10,50,20); add(no);
    r.setBounds(5,55,50,10); add(r);
    ra.setBounds(80,50,50,20); add(ra);
    e.setBounds(5,95,50,10); add(e);
    ed.setBounds(80,90,50,20); add(ed);
    g.setBounds(5,135,50,10); add(g);
    ge.setBounds(80,130,50,20); add(ge);
    con.setBounds(5, 180, 90,20); add(con);
    ins.setBounds(120, 180, 90,20); add(ins);
}

```

Figura 1. Interfaz.

La parte importante de esta práctica viene en el método actionPerformed, en la figura 2 podemos ver un if diciendo que, si el botón presionado fue el de conectar, se hará la conexión con la base de datos, la conexión se hace dentro de un try por que puede arrojar una excepción. Para crear la conexión, usamos el método getConnection de la clase DriverManager, en donde ponemos la url, el usuario y la contraseña y para evitar problemas, removemos el ActionListener al botón de conectar y se lo agregamos al de insertar.

```

if(btn==con){
    try {
        cone= DriverManager.getConnection("jdbc:mysql://localhost:3306/perros", "root", "");
        con.removeActionListener(this);
        con.addActionListener(this);
        ins.addActionListener(this);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

Figura 2. Conexión con la base de datos.

Si el botón presionado es el de insertar crearemos un statement para insertar los datos ingresados en los campos de entrada en la base de datos, esto se logra con `executeUpdate`, lo podemos ver en la figura 3;

```
else{
    nombre= no.getText();
    raza=ra.getText();
    edad=Integer.parseInt(ed.getText());
    genero=ge.getText();

    try {
        st=cone.createStatement();
        st.executeUpdate("Insert into perro (nombre, raza, edad, genero) Values ('"+nombre+"', '"+raza+"', '"+edad+"', '"+genero+"')");
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Figura 3. Inserción en la base de datos.

Ahora veamos el funcionamiento del programa, una vez tengamos nuestra base de datos creada y la tabla donde se insertarán los datos, podemos correr el programa e insertar los datos.

En la figura 4, vemos que nuestra tabla está vacía

```
MariaDB [perros]> select * from perro;
Empty set (0.000 sec)
```

Figura 4. Tabla vacía.

En la figura 5 vemos los datos que se insertarán en la tabla desde nuestro programa.

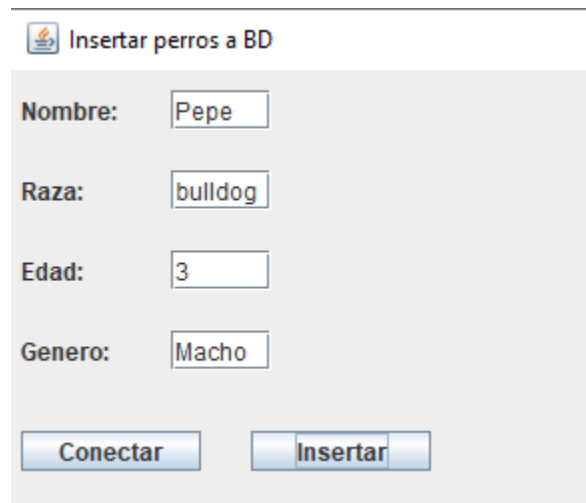


Figura 5. Datos a insertar.

Al conectarnos y presionar el botón de insertar, se agregan estos datos a la tabla de nuestra base de datos, esto lo vemos en la figura 6;

```

MariaDB [perros]> select * from perro;
+-----+-----+-----+-----+
| nombre | raza   | edad | genero |
+-----+-----+-----+-----+
| Pepe   | bulldog | 3    | Macho  |
+-----+-----+-----+-----+
1 row in set (0.000 sec)

```

Figura 6. Datos insertados en la tabla.

En las dos figuras siguientes vemos otra inserción en nuestra tabla;

Insertar perros a BD

Nombre:

Raza:

Edad:

Genero:

Figura 7. Datos a insertar.

```

MariaDB [perros]> select * from perro;
+-----+-----+-----+-----+
| nombre | raza      | edad | genero |
+-----+-----+-----+-----+
| Pepe   | bulldog   | 3    | Macho  |
| Robert | Chihuahua | 5    | Hembra |
+-----+-----+-----+-----+
2 rows in set (0.000 sec)

```

Figura 8. Datos insertados.

## CONCLUSIÓN

Como vimos, gracias al driver JDBC, Java consigue construir un código, llamémosle neutro, que le permite conectarse a cualquier tipo de base de datos, realizando operaciones sobre ellas desde nuestra aplicación, en este caso fue insertar perros a una base de datos.