Conexión MySQL con Java Eclipse.

José Alfredo Castro Hernández

alfredo.castro9708@gmail.com

Universidad MEZE

Programación III

08 de Mayo de 2019

Resumen

En el presente documento se tratará el tema sobre la instalación del componente JDBC en el lenguaje de programación Java Eclipse para realizar una conexión a una base de datos alojada en el software MySQL.

Keywords: *Base de datos, Eclipse, Conexión,*

Introducción

Actualmente y desde hace tiempo las bases de datos han sido de suma importancia para cualquier empresa o institución, logrando almacenar una impresionante cantidad información a diario, pues son una herramienta indispensable remplazando completamente a los archiveros tradicionales. Existen dos tipos de bases de datos, relacionales(SQL) y no relacionales(NoSQL). Con esta tecnología se ha logrado mejorar la accesibilidad de los datos en ese momento, evitando buscar en papeles por minutos e incluso horas, con esto mismo mejorando la productividad e integridad de los datos.

Requisitos antes de la instalación.

1. Java Develpement Kit v8.0.2010.9

2. MySQL Workbeanch v8.0.15

3. Java Eclipse 2019-3

4. Tener una base de datos ya creada con algunos campos.

5. Un nuevo proyecto de Eclipse.

Bien, ahora vamos a descargar el driver para su respectiva instalación.

1. Para descargar el driver JDBC deberemos de ir a la siguiente ***URL:http://dev.mysql.com/downloads/connector/j/***, que nos llevar´a a la página oficial de MySQL para adquirirlo. Bajamos un poco para elegir la plataforma y seleccionamos ***Platform Independent.***

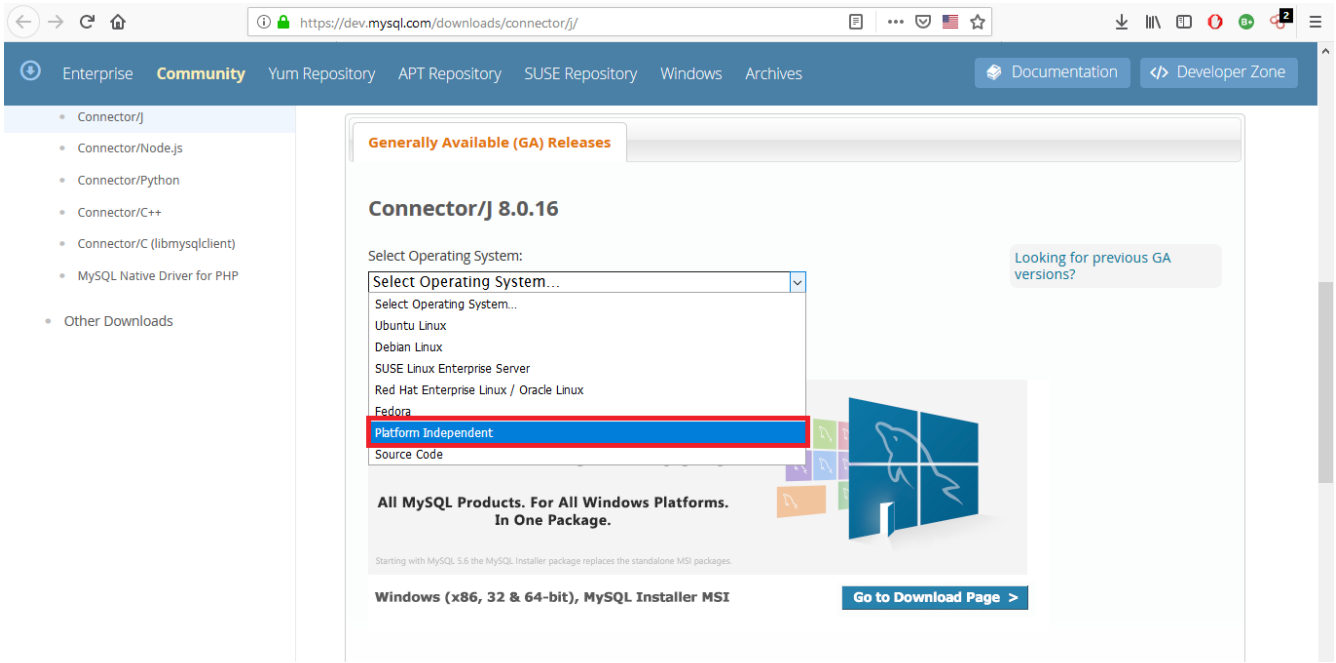


Fig 1: Página de descarga del driver.

2. Ahora se mostrara en pantalla dos opciones, nosotros elegiremos la segunda opción.

Fig 2: Continuar con la descarga.

3. Si no existen problemas, podremos pulsar sobre la opción ”No thanks, just start mydownload.” y comenzar´a nuestra descarga automáticamente. Si por el contrario no pasa esto, crearemos una cuenta o nos iniciamos sesión y ya podremos proceder a dicha descarga.

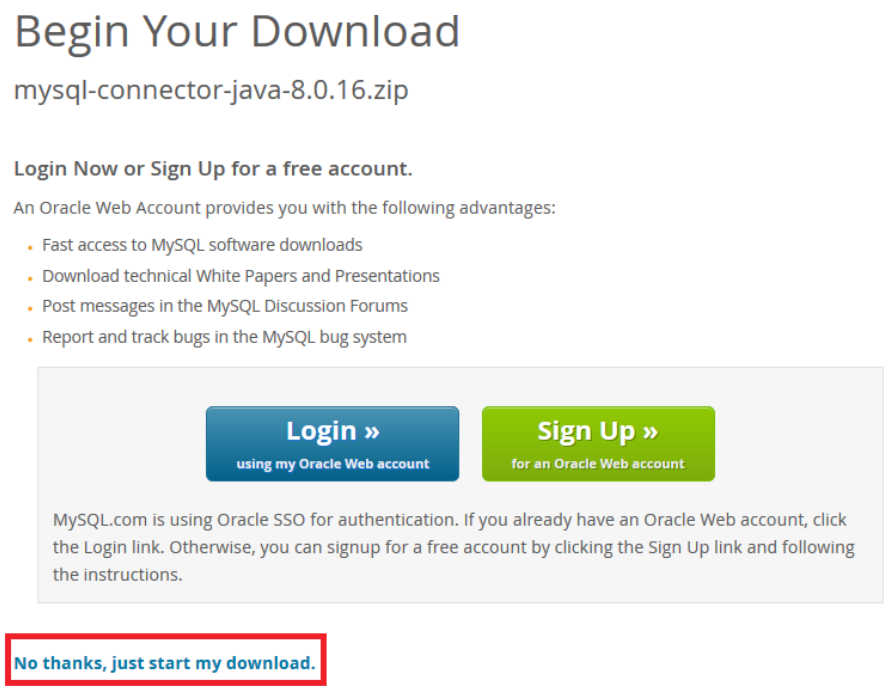


Fig 3: Paso final para descargar el Driver.

4. Ahora añadiremos el conector a Eclipse.

4.1. Pulsamos las propiedades del proyecto en el cual necesitemos el conector y damos click derecho y buscamos **Properties**.

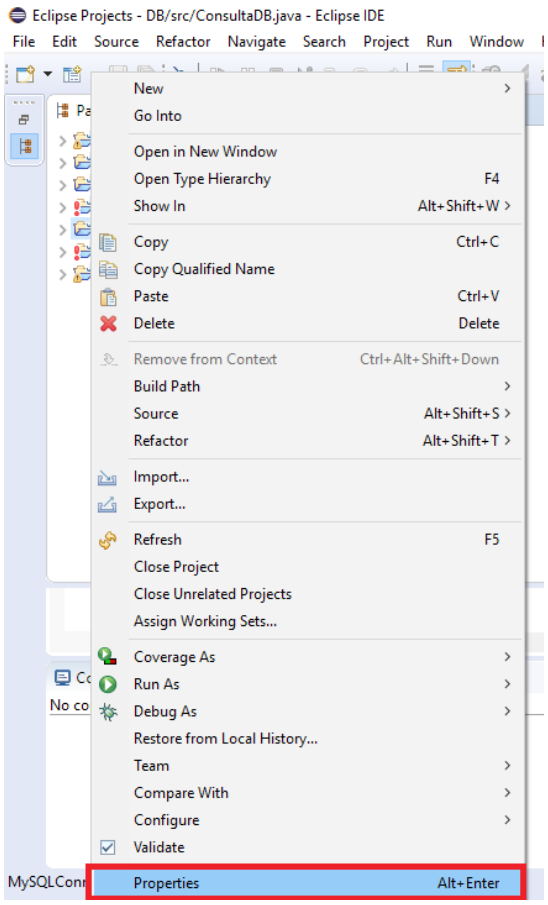


Fig 4: Buscamos la opción Properties.

5. Aparecerá un menú y buscamos **Java Build Path.**

5.1. Luego buscamos **Libraries.**

5.2. Posteriormente damos click en **Add External JAR.**

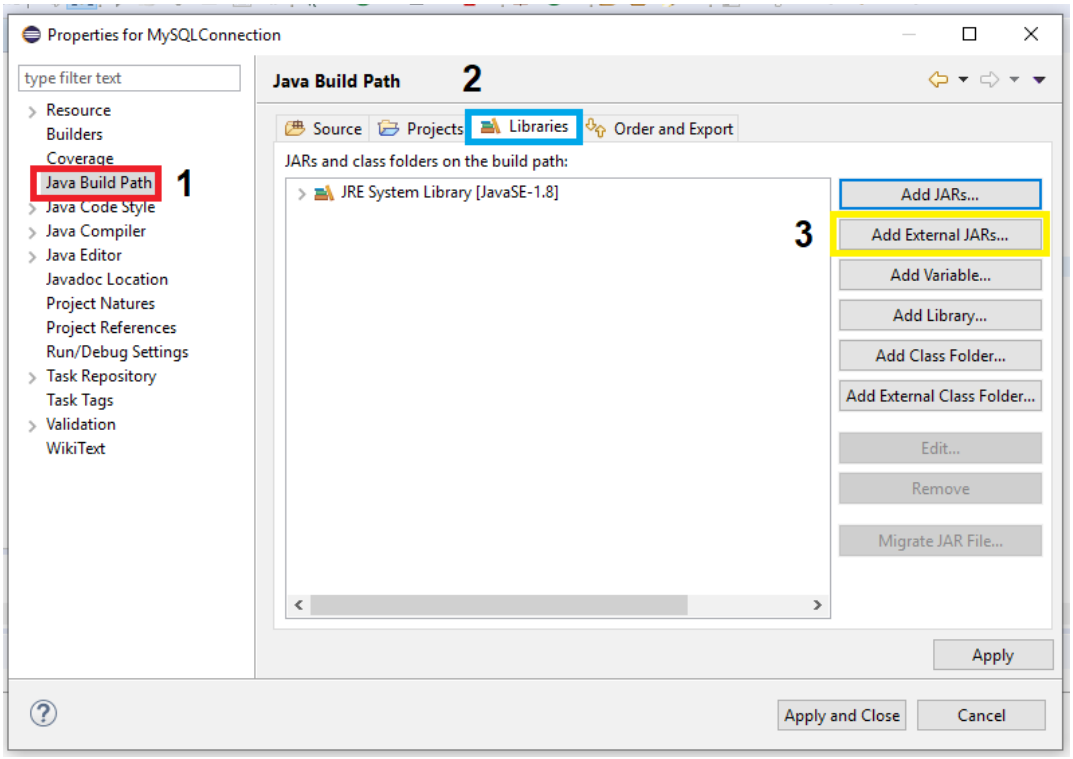


Fig 4: Menú propiedades del proyecto.

6. Se mostrará en pantalla el explorador de archivos de Windows. Buscamos dónde guardamos el archivo que descargamos anteriormente. En mi caso en la carpeta Descargas. Después damos click en Abrir y se agregará la librería.

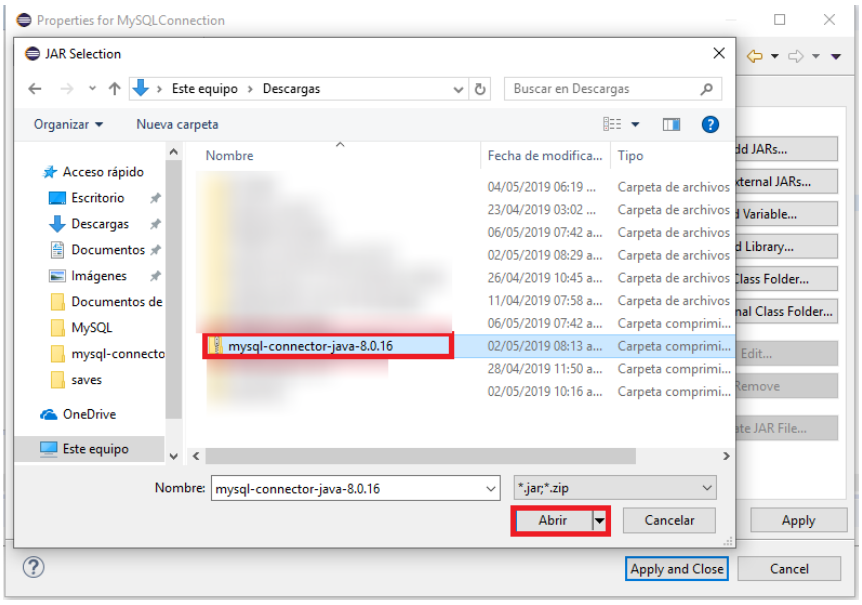


Fig 5: Explorador de Archivos de Windows.

7. Al hacer esto, ya se habrá agregado la librería al proyecto y procederemos a escribir el código correspondiente para realizar la conexión a la base de datos de MySQL

8. Importamos todas las librerías de SQL con import java.sql.*; esto para no tener problemas a la hora de estar programando y no buscar que librería se necesita en especifico para tal tarea.

```
import java.sql.*;
```

9. Creamos una instancia dentro de nuestra `public class “Nombre de su DB”`

```
public  ConnectionDB()
```

```
 public class ConnectionDB {


  }
```

Esta instancia se utilizará para escribir todo el código de conexión, consulta y muestra de resultados.

10. Ahora usaremos un `try  catch` para los posibles errores de de MySQL.

```
 try{

      }
         catch (Exception e) {

        }
```

11. Lo primero que debemos hacer es utilizar el driver que acabamos de descargar e importar al proyecto y lo registramos.

```
  DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver());
```

12. Después establecemos la conexión con la base de datos usamos ***`getConnection`*****()**

```
Connection conexion = DriverManager.getConnection ("jdbc:mysql://localhost/
Nombre de su DB","Su usuario", "Su contraseña");
```

El primer parámetro del método **getConnection()** es un **String** que contiene la URL de la base de datos:

jdbc: mysql porque estamos utilizando un driver jdbc para MySQL, que es el que nos hemos bajado.
localhost: porque el servidor de base de datos, en mi caso, está en el mismo ordenador en el que voy a correr el prorama java. Aquí puede ponerse una IP o un nombre de máquina que esté en la red.

Nombre de la DB: es el nombre de la base de datos que he creado dentro de MySQL, para poder conectarnos.
Usuario: Nombre de su usuario cuando inicia en MySQL.
Contraseña: Contraseña usada para iniciar sesión en MySQL, en caso de que no use se deja en blanco entre las comillas.

13. Para realizar una consulta, insertar datos, modificar, etc. Se usa la clase `Statement`

```
Statement state = conexion.createStatement();
```

createStatement()

Es un objeto del tipo de una clase que implementa la interfaz Statement, y provee la infraestructura para ejecutar sentencias SQL sobre una conexión con una base de datos.

El **Statement** obtenido tiene un método **executeQuery()**. Este método sirve para realizar una consulta a base de datos y usamos **Resultset**.

```
ResultSet res = state.executeQuery ("SELECT * FROM Contacto");
```

Se utiliza **"SELECT * FROM Contacto"** siendo **Contacto** el nombre de mi tabla, remplace la tabla por la suya.

El resultado devuelve el método como un **Resultset.**

14. El **ResultSet** contiene dentro los registros obtenidos de la base de datos. Inicialmente, tal cual nos lo devuelve el **Statement.executeQuery()**.

Se usa un **while,** para que vaya buscando en los registros de la base de datos, los métodos usados son **getInt()**, **getString(),** etc nos van devolviendo los valores de los campos de dicho registro.

```
while (res.next()) {

  System.out.println (res.getInt ("ID") + " " + res.getString (2)+ " " +
res.getString(3)  +  "  "+  res.getString(4)+"  "  +  res.getInt(5)+  "
"+res.getString(6));

}
```

15. Se cierra la conexión a la base de datos, junto con el **catch**.

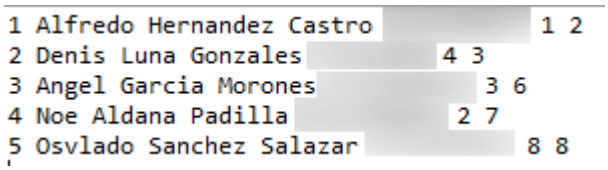
```
conexion.close();
```

```
catch (Exception e) {

  e.printStackTrace();

}
```

Código completo.

```
import java.sql.*;

public class ConnectionDB {
    public  ConnectionDB()
    {
        try
        {
 DriverManager.registerDriver(newcom.mysql.cj.jdbc.Driver());

Connection conexion = DriverManager.getConnection ("jdbc:mysql://localhost/Nombre de la
DB","Usuario", "");
            Statement state = conexion.createStatement();

            ResultSet res = state.executeQuery ("SELECT * FROM Contacto");

            while (res.next())
            {
System.out.println (res.getInt ("ID") + " " + res.getString (2)+ " " + res.getString(3) +
" "+ res.getString(4)+" " + res.getInt(5)+ " "+res.getString(6));
            }

            conexion.close();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
    public static void main(String[] args)
    {
        new  ConnectionDB();
    }
}
```

16. Ahora ejecutaremos el código para realizar la consulta y esto es lo que nos debe de mostrar en la consola.



Conclusiones.

Las conexiones a una base de dato, para mí, no son nada fáciles,siempre he tenido algunos problemas con esta parte en la programación, pero eso no quiere decir que lo deje de hacer por algunas complicaciones, al contrario, me hacer querer seguir intentando y aprender más sobre el tema, y mejorar poco a poco en el mismo. En este trabajo si tuve mis complicaciones, sin embargo pude obtener el resultado que quise.

Referencias

[1] JDBC: Conexión con Base de Datos
http://labojava.blogspot.com/2012/05/jdbc-conexion-con-base-de-datos.html

[2] Download Connector/J
https://dev.mysql.com/downloads/connector/j/

[3] Ejemplo básico de MySQL con Java
nhttp://www.chuidiang.org/java/mysql/EjemploJava.php