JDBC Connection in Java

Before establishing a connection between your Java Program(Front end) and Database (back end). The database we should learn what precisely a JDBC is and why it came to existence.

What is JDBC ?
JDBC is an acronym for Java Database Connectivity. It's an advancement for ODBC (Open Database Connectivity). JDBC is an standard API specification developed in order to move data from frontend to backend. This API consists of classes and interfaces written in Java. It basically acts as an interface (not the one we use in Java) or channel between your Java program and databases i.e it establishes a link between the two so that a programmer could send data from Java code and store it in the database for future use.

Why JDBC came into existence ?
As previously told JDBC is an advancement for ODBC, ODBC being platform dependent had a lot of drawbacks. ODBC API was written in C,C++, Python, Core Java and as we know above languages (except Java and some part of Python)are platform dependent . Therefore to remove dependence, JDBC was developed by database vendor which consisted of classes and interfaces written in Java.

To get JDBC Jar and maven dependency

https://mvnrepository.com/artifact/mysql/mysql-connector-java/8.0.12

Steps for connectivity between Java program and database

Step 1. Loading the Driver
To begin with, you first need load the driver or register it before using it in the program . Registration is to be done once in your program. You can register a driver in one of two ways mentioned below :

- **Class.forName() :** Here we load the driver's class file into memory at the runtime. No need of using new or creation of object .The following example uses Class.forName() to load the Oracle driver –
```
Class.forName("oracle.jdbc.driver.OracleDriver");
```
- **DriverManager.registerDriver():** DriverManager is a Java inbuilt class with a static member register. Here we call the constructor of the driver class at compile time . The following example uses DriverManager.registerDriver()to register the Oracle driver –
```
DriverManager.registerDriver(new
oracle.jdbc.driver.OracleDriver())
```

Step 2. Create the connections
After loading the driver, establish connections using :

```
Connection connect = DriverManager.getConnection(url,user,password)
```

user – username from which your sql command prompt can be accessed("root")
password – password from which your sql command prompt can be accessed.(your password)
connect: is a reference to Connection interface.
url : Uniform Resource Locator. It can be created as follows:

```
String url =
"jdbc:mysql://localhost:3306/maruf?useSSL=false&allowPublicKeyRetrieval=true"
```

Where oracle is the database used "@localhost" is the IP Address where database is stored, "3306" is the port number and "maruf". All 3 parameters above are of String type and are to be declared by programmer before calling the function. Use of this can be referred from final code.

Step 3. Create a statement
Once a connection is established you can interact with the database. The JDBCStatement, CallableStatement, and PreparedStatement interfaces define the methods that enable you to send SQL commands and receive data from your database. Use of JDBC Statement is as follows:

```
Statement st = connect.createStatement();
```

Here, "connect" is a reference to Connection interface used in previous step .

PIIT Video: https://video.piit.us/index.php/player/selenium/12-04-2018-ny-in-class-wd-sel-5th-class
https://docs.oracle.com/javase/8/docs/api/java/sql/package-summary.html