**JDBC Connection in Java**

**What is JDBC ?**  
JDBC is an acronym for Java 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.

**Steps for connectivity between Java program and database**

**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())

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

Connection con = DriverManager.getConnection(url,user,password)

**user** – username from which your sql command prompt can be accessed.  
**password** – password from which your sql command prompt can be accessed.

**con:** is a reference to Connection interface.  
**url** : Uniform Resource Locator. It can be created as follows:

String url = “ jdbc:oracle:thin:@localhost:1521:xe”

Where oracle is the database used, thin is the driver used , @localhost is the IP Address where database is stored, 1521 is the port number and xe is the service provider. 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.

**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 = con.createStatement();

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

**4. Execute the query**  
Now comes the most important part i.e executing the query. Query here is an SQL Query . Now we know we can have multiple types of queries. Some of them are as follows:

* Query for updating / inserting table in a database.
* Query for retrieving data .

The executeQuery() method of Statement interface is used to execute queries of retrieving values from the database. This method returns the object of ResultSet that can be used to get all the records of a table.  
The executeUpdate(sql query) method of Statement interface is used to execute queries of updating/inserting .

Example:

int m = st.executeUpdate(sql);

if (m==1)

System.out.println("inserted successfully : "+sql);

else

System.out.println("insertion failed");

Here sql is sql query of the type String

**5.Close the connections**  
So finally we have sent the data to the specified location and now we are at the verge of completion of our task .  
By closing connection, objects of Statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection.

Example : con.close();

importjava.sql.\*;

importjava.util.\*;

class Main

{

public static void main(String a[])

{

//Creating the connection

String url = "jdbc:oracle:thin:@localhost:1521:xe";

String user = "system";

String pass = "12345";

//Entering the data

Scanner k = new Scanner(System.in);

System.out.println("enter name");

String name = k.next();

System.out.println("enter roll no");

int roll = k.nextInt();

System.out.println("enter class");

String cls = k.next();

//Inserting data using SQL query

String sql = "insert into student1 values('"+name+"',"+roll+",'"+cls+"')";

Connection con=null;

try

{

DriverManager.registerDriver(new oracle.jdbc.OracleDriver());

//Reference to connection interface

con = DriverManager.getConnection(url,user,pass);

Statement st = con.createStatement();

int m = st.executeUpdate(sql);

if (m == 1)

System.out.println("inserted successfully : "+sql);

else

System.out.println("insertion failed");

con.close();

}

catch(Exception ex)

{

System.err.println(ex);

}

}

}