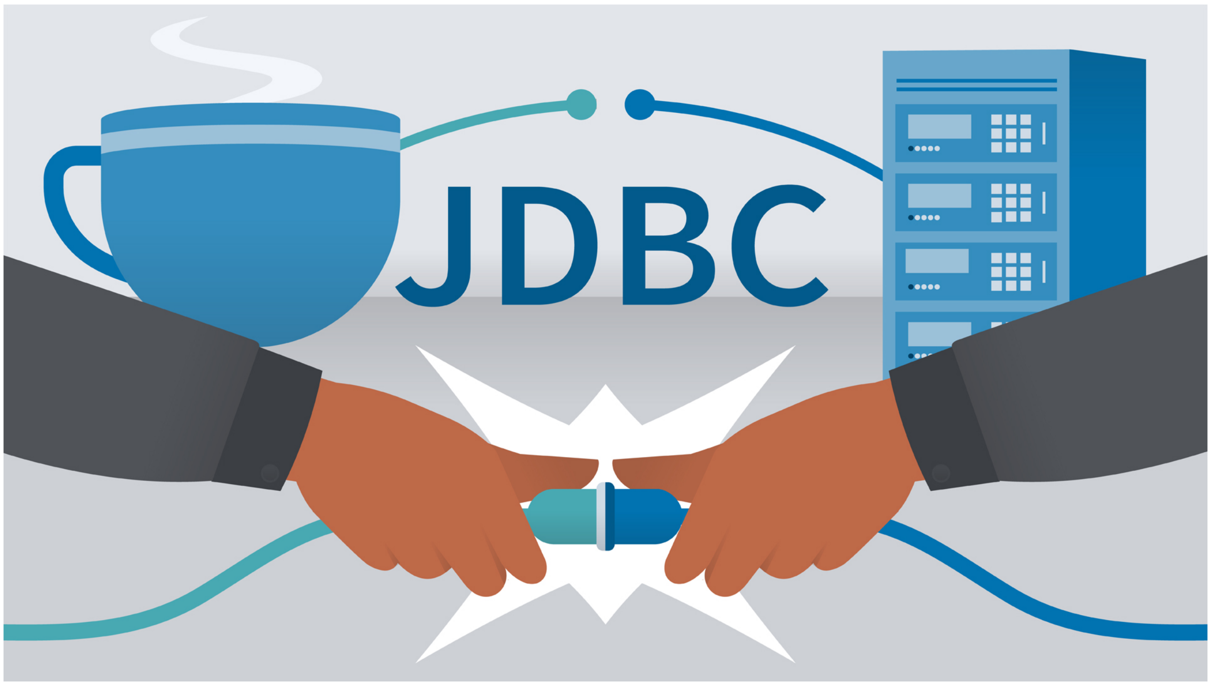
JDBC Day 1: Intro, Need, Steps to connect Java with DBMS, Connection)

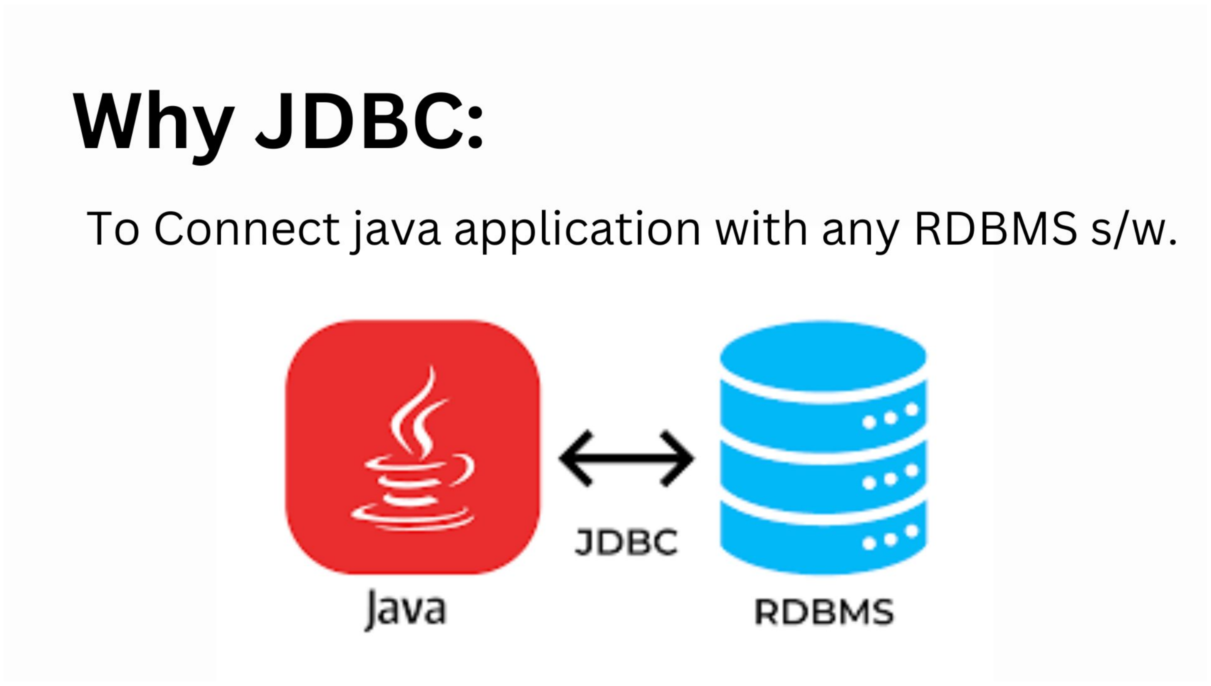


JDBC

JDBC stands for **Java Database Connectivity**. JDBC is a Java API to connect and execute the query with the database. JDBC API uses JDBC drivers to connect with the database.

We can use JDBC API to access tabular data stored in any relational database. By the help of JDBC API, we can save, update, delete and fetch data from the database.

The Java Database Connectivity (JDBC) API provides universal data access from the Java programming language. Using the JDBC API, you can access virtually any data source, from relational databases to spreadsheets and flat files. JDBC technology also provides a common base on which tools and alternate interfaces can be built.

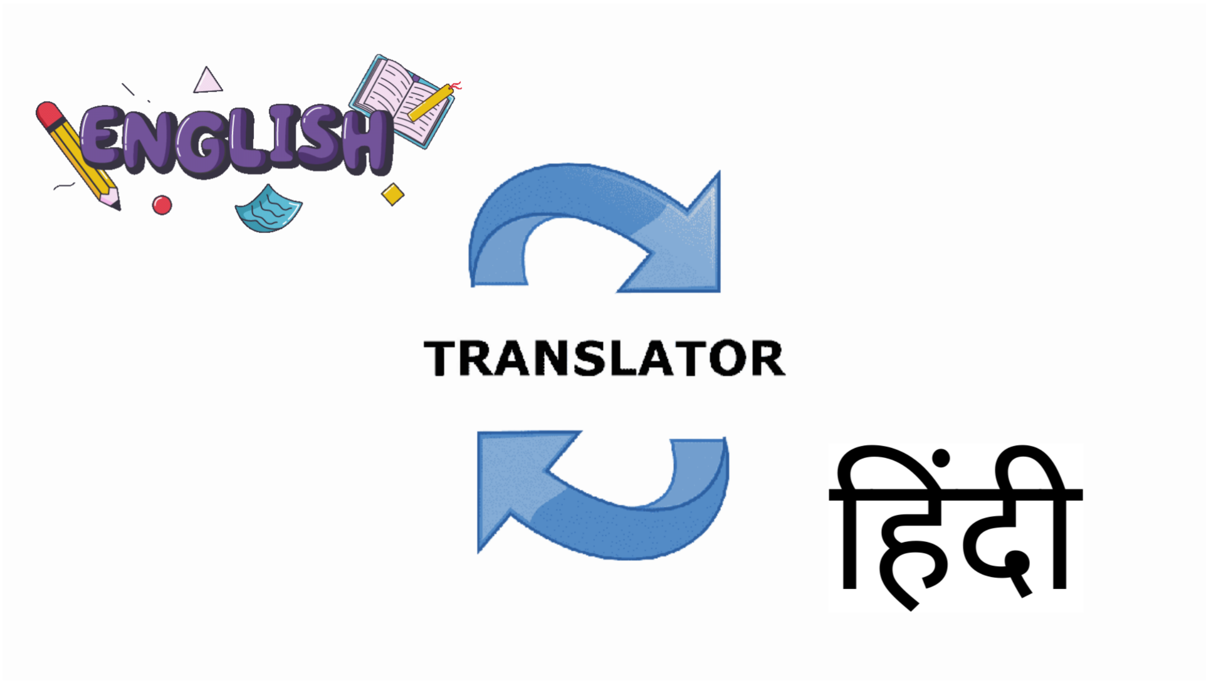


JDBC Driver

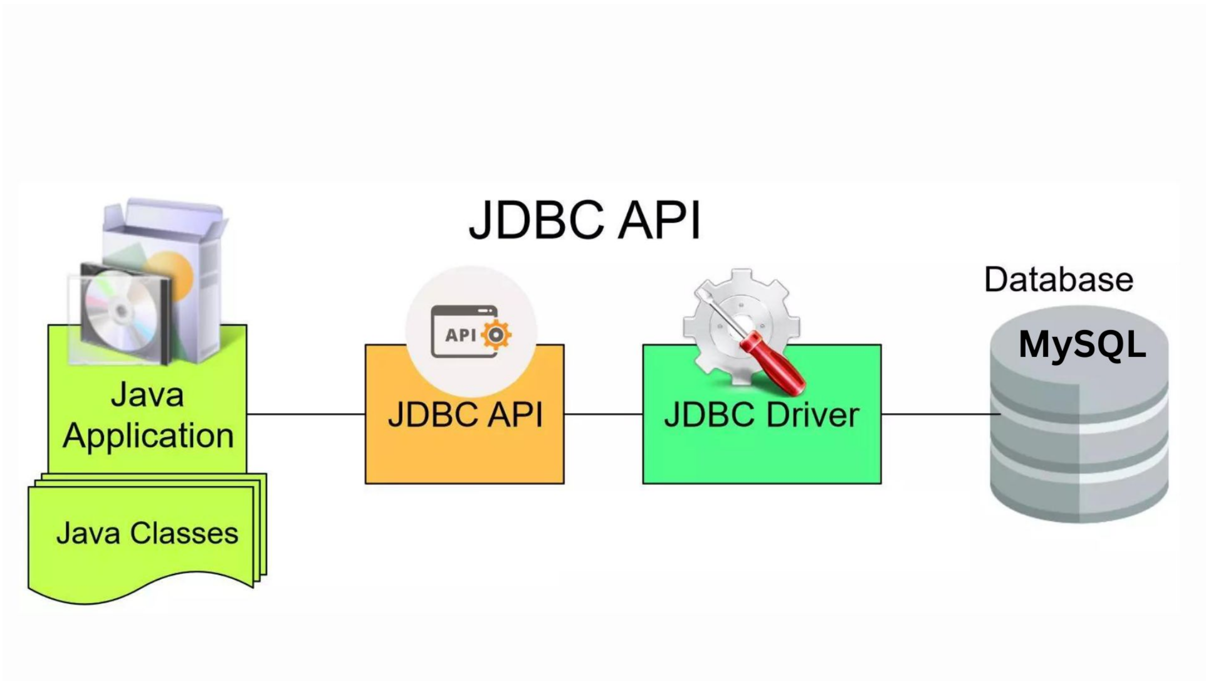
JDBC Driver is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers:

Essentially, a JDBC driver makes it possible to do three things: Establish a connection with a data source. Send queries and update statements to the data source.

Real Life Example:



Similarly…



There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:

1) Register the driver class

forName()

Syntax of forName() method

1. **public static void** forName(String className)**throws** ClassNotFoundException

Note: Since JDBC 4.0, explicitly registering the driver is optional. We just need to put vender's Jar in the classpath, and then JDBC driver manager can detect and load the driver automatically.

Example to register the OracleDriver class

Here, Java program is loading oracle driver to esteblish database connection.

1. Class.forName("oracle.jdbc.driver.OracleDriver");

Example to register the Mysql driver class:

Class.forName("com.mysql.cj.jdbc.Driver");

2) Create the connection object

getConnection()

Syntax of getConnection() method

1) **public static** Connection getConnection(String url)**throws** SQLException

2) **public static** Connection getConnection(String url,String name,String password) **throws** SQLException

---

3) Create the Statement object

Syntax of createStatement() method

1. **public** Statement createStatement()**throws** SQLException

---

4) Execute the query

Syntax of executeQuery() method

1. **public** ResultSet executeQuery(String sql)**throws** SQLException

---

5) Close the connection object

Syntax of close() method

1. **public void** close()**throws** SQLException

Note: Since Java 7, JDBC has ability to use try-with-resources statement to automatically close resources of type Connection, ResultSet, and Statement.

It avoids explicit connection closing step.

Connecting Java with MySQL

Let's first create a table in the mysql database, but before creating table, we need to create database first.

```
create database jdbcdb;

use jdbcdb;

create table student(roll int,name varchar(40),address varchar(40),mobile varchar(10));
```

Without Configure Build Path with mysql jar file:

```
package com.masai;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class Demo {
public static void main(String[] args) throws ClassNotFoundException {

  Class.forName("com.mysql.cj.jdbc.Driver");

  String url = "jdbc:mysql://localhost:3306/jdbcdb";

  try {
    Connection con = DriverManager.getConnection(url,"root","root");

      System.out.println(con);

    con.close();

  } catch (SQLException e) {
    e.printStackTrace();
  }

 }
}
```

Problem: Above code will throw **ClassNotFoundException(),** Because we have not configured the

class path with mysql driver jar file.

JAR
https://dev.mysql.com/downloads/connector/j/
select platform independent
go to **Platform Independent (Architecture Independent), ZIP Archive**
Or just go to https://dev.mysql.com/downloads/file/?id=513221

To import jar file in your Eclipse IDE, follow the steps given below.

1. Right click on your project
2. Select Build Path
3. Click on Configure Build Path
4. Click on Libraries and select Add External JARs
5. Select the jar file from the required folder
6. Click and Apply and Ok

After Configured Build Path with mysql jar file:

```
package com.masai;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class Demo {
public static void main(String[] args) throws ClassNotFoundException {
Class.forName("com.mysql.cj.jdbc.Driver");

String url = "jdbc:mysql://localhost:3306/jdbcdb";

try {
  Connection con = DriverManager.getConnection(url,"root","root");

    System.out.println(con);

  con.close();

} catch (SQLException e) {
  e.printStackTrace();
}
}
}

//Output: com.mysql.cj.jdbc.ConnectionImpl@5890e879
```

WE Problem:

Insert record into the student table:

```
package com.masai;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class Demo {
public static void main(String[] args) throws ClassNotFoundException {

  Class.forName("com.mysql.cj.jdbc.Driver");

  String url = "jdbc:mysql://localhost:3306/jdbcdb";

  try {
    Connection con = DriverManager.getConnection(url,"root","root");

      Statement st = con.createStatement();

      int row =   st.executeUpdate("insert into student values(23,'Rakesh','Patna','56
88349232')");

      if(row>0) {
        System.out.println("Student detail has inserted..");
      }else {
        System.out.println("Student detail has not inserted..");
      }

    con.close();

  } catch (SQLException e) {
    e.printStackTrace();
  }

   }
}
```

You Problem:

Create student table inside the database and insert some record and then grace marks by 500 to whose marks is less than 500.

References:

https://docs.oracle.com/javase/7/docs/api/java/sql/ResultSet.html
https://www.javatpoint.com/java-jdbc

https://www.baeldung.com/java-statement-preparedstatement

https://www.roseindia.net/tutorial/java/jdbc/dataaccessobjectdesignpattern.html

https://www.oracle.com/java/technologies/data-access-object.html