<Java とデータベース(MySQL)を接続する>

◆Java とデータベースを接続するとは ここでは、これまで学んできたデータベースである MySQL と Java を接続します。 Java は直接データベースに接続することができません。 Java は Java 言語、MySQL は SQL を使って実行するので、命令する言葉が違っています。 そこで、必要となるのがドライバーというツールです。

◆JDBC ドライバーとは データベースと接続をするための工具箱のことです。 この中には Java とデータベースを繋げる道具が入っています。

◆DBConnector Java とデータベースを繋げるために自分たちで作るファイルです。 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Java プログラミング

DBConnector

データベース

このファイルを作ります JDBC ドライバー

ネット上から 「mysql-connector-java」をダウンロードします

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Google Chrome を開く。 「mysql connector j」と検索する。

MySQL :: Download Connector/J https://dev.mysql.com/downloads/connector/j/5.1.html

画面下の「Platform Independent (Architecture Independent), ZIP Archive」と書かれ ている”Download”ボタンを押す

画面下の No thanks, just start my download.をクリック

「mysql-connector-java-5.x.xx.zip」がダウンロードされる

Windows エクスプローラを開く

ダウンロードフォルダを開く

「mysql-connector-java-5.x.xx.zip」を展開する

展開した「mysql-connector-java-5.x.xx-bin」フォルダを開く

この中の「mysql-connector-java-5.x.xx-bin.jar」をこれから使ってゆきます。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Eclipse を使ってプログラミング>

Java プロジェクト「TestDB」を作成する。

パッケージ・エクスプローラーの空白部分を右クリック->新規->Java プロジェクトを選択 します。 「新規 Java プロジェクト」画面が表示されます。 プロジェクト名:TestDB 実行環境 JRE の使用:JavaSE-1.8 「次へ」ボタンを押下します。 「ライブラリー」タブを選択。 「外部 JAR の追加」ボタンを押下。 先ほどの mysql-connector-java-x.x.xx-bin.jar を選択して、「開く」ボタンを押下しま す。 「完了」ボタンを押下します。

Java プロジェクトフォルダ(TestDB)を開きます。 src フォルダを右クリックします。 新規->クラスを選択します。 「新規 Java クラス」画面が表示されます。 名前:ChkLoadJdbc

□「public static void main(String[] args)」にチェックをつける。

□「コメントの生成」にチェックをつける。 「完了」ボタンを押下します。 -------------------------------------------------------------------------------- 以下をプログラミングしましょう。 --------------------------------------------------------------------------------

public class ChkLoadJdbc {

public static void main(String[] args) throws InstantiationException, IllegalAccessException { String msg = ""; try {

Class.forName("com.mysql.jdbc.Driver").newInstance(); msg = "ドライバのロードに成功しました"; } catch (ClassNotFoundException e){

msg = "ドライバのロードに失敗しました"; } System.out.println(msg); } }

-------------------------------------------------------------------------------- 実行しましょう。 -------------------------------------------------------------------------------- JDBC ドライバを正常に呼び出していれば、「ドライバのロードに成功しました」、失敗し ていた場合は、「ドライバのロードに失敗しました」と表示されます。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Eclipse を使ってプログラミング>

Java プロジェクト「TestDB2」を作成する。

パッケージ・エクスプローラーの空白部分を右クリック->新規->Java プロジェクトを選択 します。 「新規 Java プロジェクト」画面が表示されます。 プロジェクト名:TestDB2 実行環境 JRE の使用:JavaSE-1.8 「次へ」ボタンを押下します。 「ライブラリー」タブを選択。 「外部 JAR の追加」ボタンを押下。 先ほどの mysql-connector-java-x.x.xx-bin.jar を選択して、「開く」ボタンを押下しま す。 「完了」ボタンを押下します。

Java プロジェクトフォルダ(TestDB2)を右クリックします。 新規->フォルダを選択します。 「sql」と入力して「完了」ボタンを押下。 sql フォルダを右クリック

「その他」を選択 ウィザードに「sql」と入力 SQL ファイルを選択 「testdb.sql」と入力。 「完了」ボタンを押下。

-------------------------------------------------------------------------------- testdb.sql に以下をプログラミングしましょう。 -------------------------------------------------------------------------------- drop database if exists testdb; create database testdb; use testdb; create table test\_table( user\_id int, user\_name varchar(255), password varchar(255) );

insert into test\_table values(1,”taro”,”123”); insert into test\_table values(2,”jiro”,”456”); insert into test\_table values(3,”hanako”,”789”);

<SQL ファイルを使って、データベースを作成する>

※初めて操作する場合と、2回目以降の場合では方法が変わります。

(初めて操作する場合)

SQL ファイルを選択して、

1.右クリック

2.SQL ファイルの実行

3.データベース・サーバー・タイプ:MySql\_5.1 を選択してください。

接続プロファイル名:「作成」ボタンを押下してください。

4.MySQL を選択し、名前欄に接続するデータベース名を入力してください。

「次へ」ボタンを押下します。

5.ドライバー:右から 2 番目の四角に+がついたマークを選択し、 MySQL JDBC ドライバ

ー MySQL 5.1 を選択してください。 「Jar リスト」のタブを選択してください。 ドライバー欄に mysql-connector-java-5.1.0-bin.jar があるが サンプルとしてダミー 設定されています。 その為、適切なものに置き換えます。 まず、「JAR/Zip の除去」ボタンを押下し削除します。 「JAR/Zip の追加」のボタンを押下します。

6.mysql-connector-java-x.x.xx-bin.jar を選択し、「開く」を押下します。

7.データベース:データベース名を入力します。

8.URL:jdbc:mysql://localhost:3306/database を jdbc:mysql://localhost:3306/mysql に変更します。

これは自分の PC に作成されたデータベースを指定することで、接続を実現する為です。

9.パスワード:mysql と入力します。

「次へ」ボタンを押下します。

10.「次へ」ボタンを押下する。

11.前の画面にて入力した情報が表示されます。

「完了」ボタンを押下します。

(2回目以降の場合)

SQL ファイルを選択して、

1.右クリック

2.SQL ファイルの実行

3.接続プロファイル名をプルダウンして、作成した「新規 MySQL」を選択する

4.データベース名をプルダウンして、作成したデータベースを選択する

5.OK ボタンを押す

-------------------------------------------------------------------------------- 上記の操作でデータベースが作成されたことを確認しましょう。 --------------------------------------------------------------------------------

1.コマンドプロンプトを開く(cmd) 2.MySQL にログインする。

mysql -u root -p mysql 3.show databases; 4.testdb が存在することを確認する。 5.use testdb; 6.show tables; 7.select \* from test\_table; 8.test\_table に情報が登録されていることを確認する。

※これで、以下のデータベースとテーブルが作成できています。

データベース名:testdb テーブル名 :test\_table テーブル定義 : Field Type Null Key Default Extra user\_id int(11) YES NULL user\_name varchar(255 ) YES NULL Password varchar(255 ) YES NULL

user\_id user\_name password

1 taro 123

2 jiro 456

3 hanako 789

-------------------------------------------------------------------------------- DBConnector を作成する。 --------------------------------------------------------------------------------

src フォルダを右クリックします。 新規->クラスを選択します。 「新規 Java クラス」画面が表示されます。 名前:DBConnector

□「コメントの生成」にチェックをつける。 「完了」ボタンを押下します。 -------------------------------------------------------------------------------- 以下をプログラミングしましょう。 -------------------------------------------------------------------------------- import java.sql.DriverManager;

import java.sql.SQLException;

import com.mysql.jdbc.Connection;

/\*\*

\* <p>MySQL に接続する為のユーティリティクラスです。<br> \* ルートアカウントにて DB に接続されます。</p> \*/ public class DBConnector {

/\*\*

\* JDBC ドライバー名 \*/ private static String driverName = "com.mysql.jdbc.Driver"; /\*\*

\* データベース接続 URL \*/ private static String url = "jdbc:mysql://localhost/testdb?autoReconnect=true&useSSL=false";

/\*\*

\* データベース接続ユーザ名 \*/ private static String user = "root"; /\*\*

\* データベース接続パスワード \*/ private static String password = "mysql";

public Connection getConnection() { Connection con = null; try{ Class.forName(driverName); con = (Connection) DriverManager.getConnection(url,user,password); } catch (ClassNotFoundException e) { e.printStackTrace() ; } catch (SQLException e) { e.printStackTrace() ; } return con ; } }

<TestUserDAO を作成する>

import import import import java.sql.Connection; import import import import java.sql.PreparedStatement; import import import import java.sql.ResultSet; import import import import java.sql.SQLException;

public public public public class class class class TestUserDAO {

String name = ""; String password = ""; public public public public void void void void select(String name,String password) {

DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="select \* from test\_table where user\_name=? and password=?"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ps.setString(1, name); ps.setString (2, password); ResultSet rsrsrsrs=ps.executeQuery(); ifififif (rs.next()) {

System.out out out out.println(rs.getString("user\_name")); System.out out out out.println(rs.getString("password")); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ; } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } } }

<Test を作成してインスタンス化および実行する>

public class Test {

public static void main(String[] args) { TestUserDAO dao = new TestUserDAO();

dao.select("taro", "123"); } }

<TestUserDAO に以下のプログラムを追加してみましょう。>

public public public public void void void void selectAll() {

DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="select \* from test\_table"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ResultSet rsrsrsrs=ps.executeQuery(); while (rs.next()) {

System.out out out out.println(rs.getString("user\_name")); System.out out out out.println(rs.getString("password")); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ;

} catch catch catch catch (SQLException e ) {

e.printStackTrace(); } }

<Test に以下のプログラムを追加して実行してみましょう。>

dao.selectAll();

<TestUserDAO に以下のプログラムを追加してみましょう。>

public public public public void void void void selectByName(String name) {

DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="select \* from test\_table where user\_name=?"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ps.setString(1, name); ResultSet rsrsrsrs=ps.executeQuery(); while (rs.next()) {

System.out out out out.println(rs.getString("user\_name")); System.out out out out.println(rs.getString("password")); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ; } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } }

<Test に以下のプログラムを追加して実行してみましょう。>

dao.selectByName("taro");

<TestUserDAO に以下のプログラムを追加してみましょう。>

public public public public void void void void selectByPassword(String password) { DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="select \* from test\_table where password=?"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ps.setString (1, password); ResultSet rsrsrsrs=ps.executeQuery(); while (rs.next()) {

System.out out out out.println(rs.getString("user\_name")); System.out out out out.println(rs.getString("password")); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ; } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } }

<Test に以下のプログラムを追加して実行してみましょう。>

dao.selectByPassword("123");

<TestUserDAO に以下のプログラムを追加してみましょう。>

public public public public void void void void updateUserNameByUserName(String oldName,String newName) {

DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="update test\_table set user\_name=? where user\_name=?"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ps.setString(1, newName); ps.setString (2, oldName); int int int int i=ps.executeUpdate(); ifififif (i>0) {

System.out out out out.println(i + "件更新されました");

}else else else else{

System.out out out out.println("該当するデータはありませんでした"); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ; } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } }

<Test に以下のプログラムを追加して実行してみましょう。>

dao.updateUserNameByUserName("taro", "saburo");

<TestUserDAO に以下のプログラムを追加してみましょう。>

public public public public void void void void insert(int int int int user\_id,String name, String password) {

DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="insert into test\_table values(?,?,?)"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ps.setInt(1, user\_id); ps.setString(2, name); ps.setString (3, password); int int int int i=ps.executeUpdate(); ifififif (i>0) {

System.out out out out.println(i + "件登録されました"); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ; } catch catch catch catch (SQLException e ) {

e.printStackTrace(); }

}

<Test に以下のプログラムを追加して実行してみましょう。>

dao.insert(4, "shiro", "012");

<TestUserDAO に以下のプログラムを追加してみましょう。>

public public public public void void void void delete(String name) {

DBConnector dbdbdbdb = new new new new DBConnector(); Connection con con con con = db.getConnection();

String sql sql sql sql ="delete from test\_table where user\_name=?"; try try try try {

PreparedStatement pspspsps = con.prepareStatement(sql); ps.setString(1, name); int int int int i=ps.executeUpdate(); ifififif (i>0) {

System.out out out out.println(i + "件削除されました"); } } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } try try try try {

con.close() ; } catch catch catch catch (SQLException e ) {

e.printStackTrace(); } }

<Test に以下のプログラムを追加して実行してみましょう。>

dao.delete("shiro");