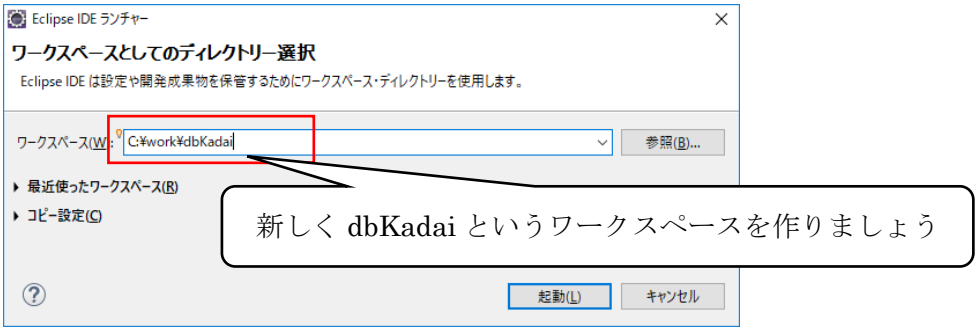
# DBKadai1　解説プリント

STEP1.まず、動かしてみよう

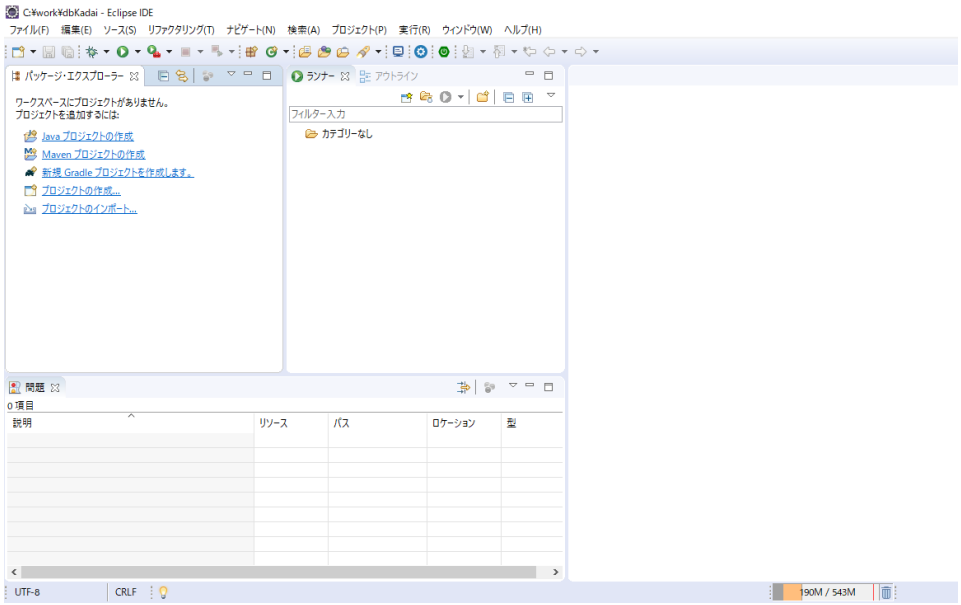
プロジェクトをインポートして動かしてみましょう。

〇プロジェクトをインポートするには、以下の手順で行います。

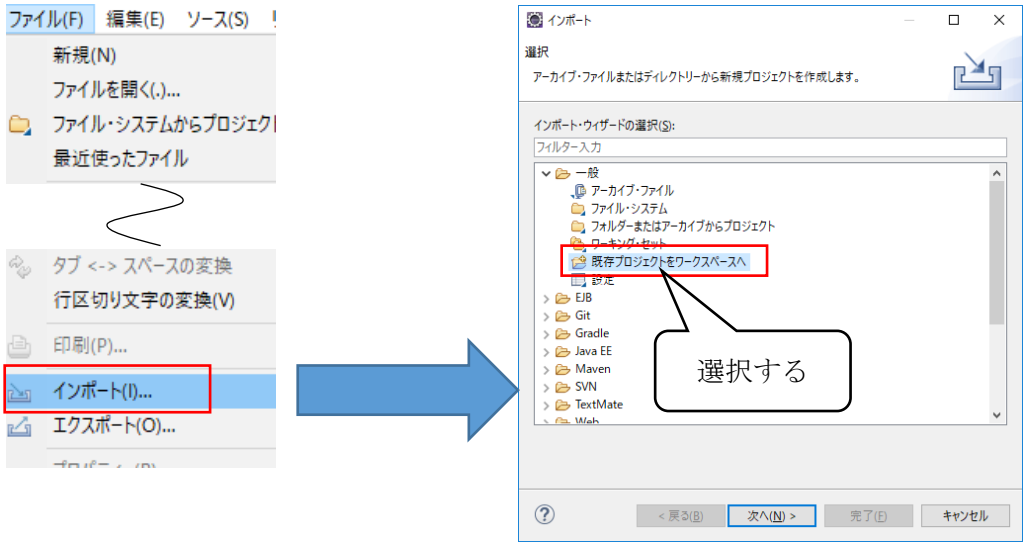
１．からのワークスペースを作る



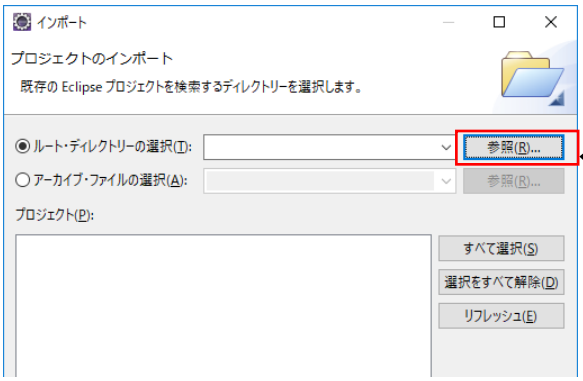

空っぽのワークスペースが開けば OK



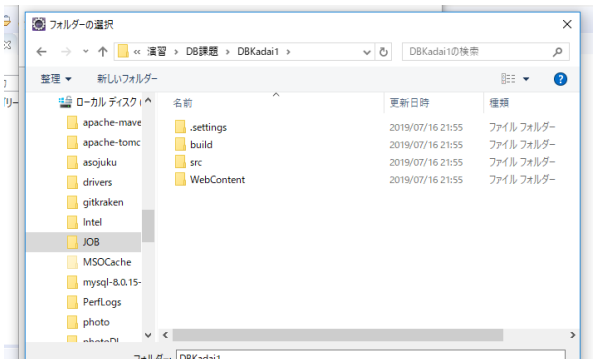

２．「ファイル」－「インポート」－「一般」－「既存のプロジェクトをワークスペースへ」

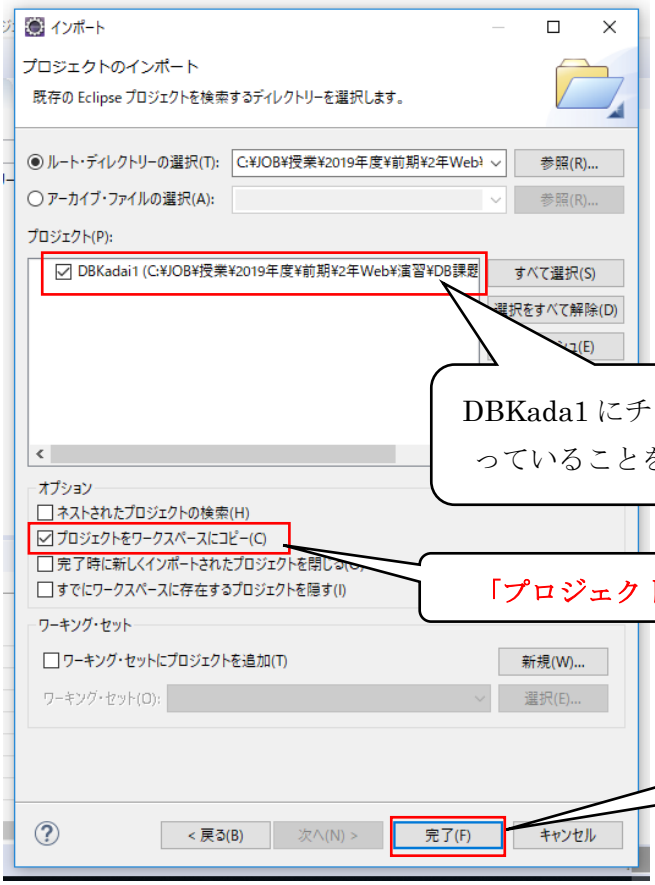


「ルートディレクトリの選択」でダウンロードして解凍した DBKadai1 のフォルダを選択する




クリックして、DBKada1 フォルダを選択する




DBKada1 にチェックが入っていることを確認する

「プロジェクトをワークスペースにコピー」にチェック

完了をクリック

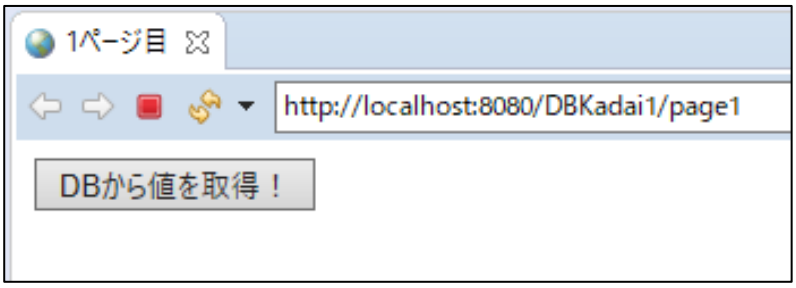
以上でプロジェクトのインポートは完了です。

**※プロジェクトにエラーが出る場合は、プロジェクトの「ビルドパス」－「ビルドパスの構成」を確認して（アンバインド済み）とついているライブラリを「除去」し、新しくライブラリを追加する**

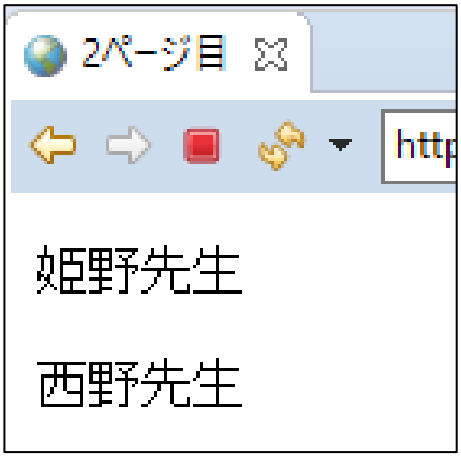
〇プログラムを動かそう！

XAMP（MAMP）をつかって、MySQL を起動して、Page1Servlet から軌道をかけてください。

↓のような動きになれば OK です



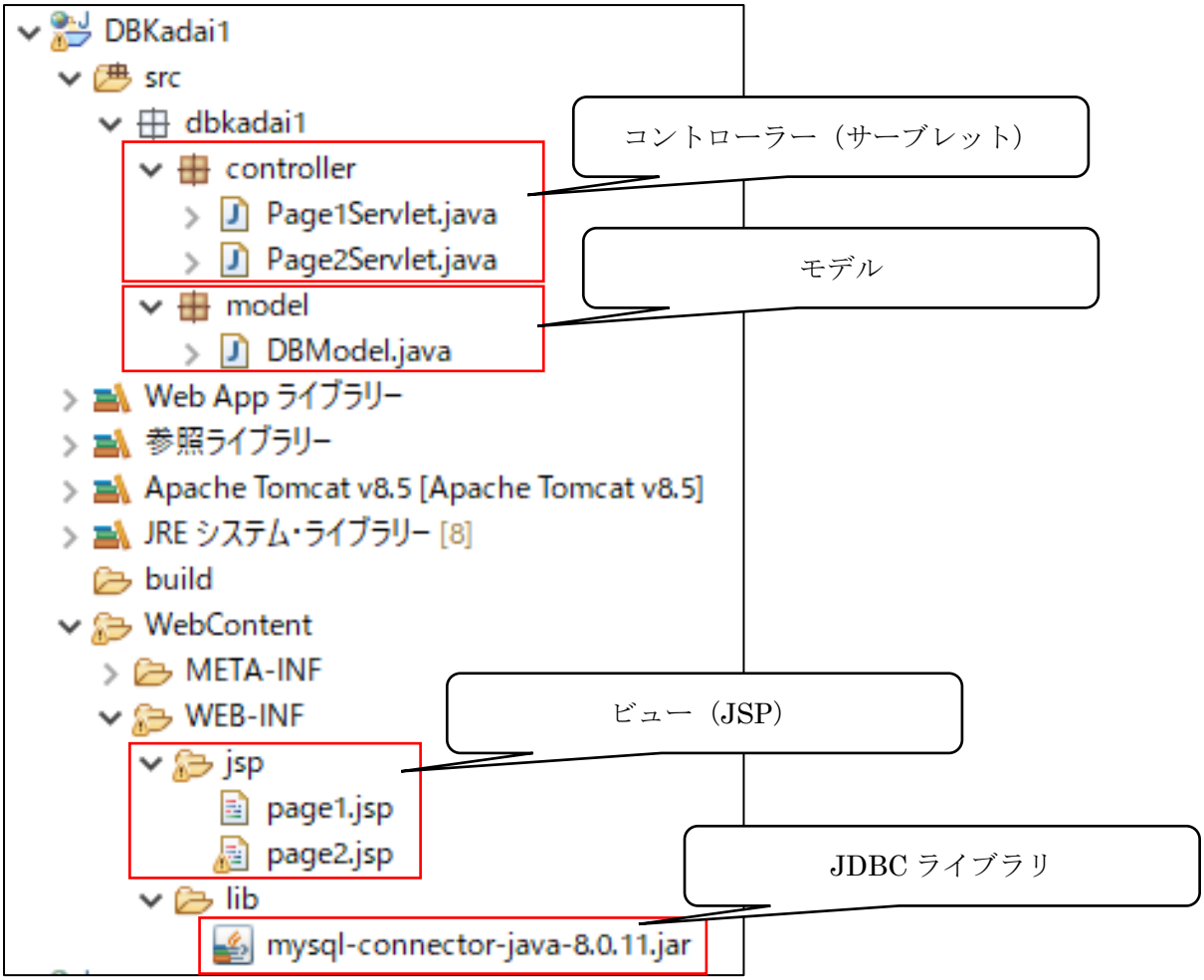

1 ページ目




2 ページ目

DB から「name」列（名前）を全件取ってきて表示しています

STEP2.プロジェクトを確認しよう

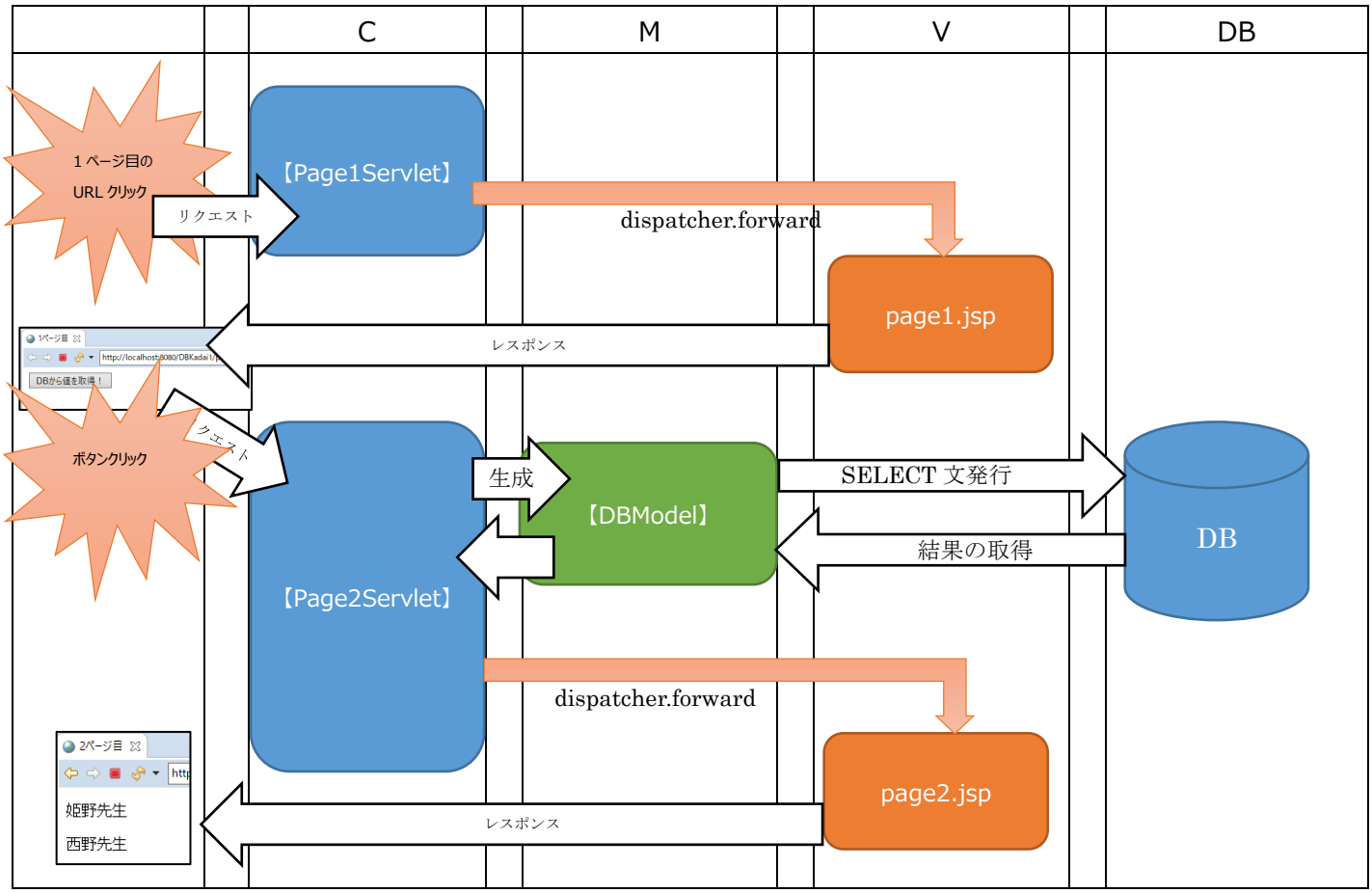



各ファイルの説明

| ファイル名（クラス名） | 説明 | MVC |
|---|---|---|
| Page1Servlet | 1 ページ目の表示を行うサーブレット<br>特にやることはないので画面転送のみ | C |
| Page2Servlet | 2 ページ目を表示するサーブレット<br>DB から値を取得するモデルを呼び出して、結果を戻り値で受け取り、リクエストスコープにセットする | C |
| DBModel | DB に接続して SQL を発行し DB から値を取得する<br>このプロジェクトのメインの処理 | M |
| page1.jsp | 1 ページ目の表示。ボタンを表示しているだけ | V |
| page2.jsp | 2 ページ目の表示。Page2Servlet でセットした値を取得して表示する | V |
| mysql-connector-java-8.0.11.jar | JDBC ドライバ<br>これがないと DB に接続できない。<br>WEB-INF の中にある lib フォルダに置く | — |

STEP3.プログラムの流れを理解する

プログラムの流れは以下のようになります




今までの MVC モデルと流れは変わりませんが、DB へのアクセスは Mode が行うことに注意してください。

ひつこいようですが、**一番メインの処理は Model が行う**と覚えておきましょう。

**サーブレットはあくまで旗振り**のみ。**View は結果の表示のみ**を行うことを意識してください。

STEP4.プログラムを見てみよう

Page1Servlet

ここは画面表示のみなので特に説明することはありません。

```java
package dbkadai1.controller;

import java.io.IOException;

@WebServlet("/page1")
public class Page1Servlet extends HttpServlet {


    @Override
    protected void doGet(HttpServletRequest req,HttpServletResponse res)throws ServletException,IOException{

        ////////////////////////////////////
        //画面遷移
        RequestDispatcher dis=req.getRequestDispatcher("WEB-INF/jsp/page1.jsp");
        dis.forward(req, res);
    }

}
```

page1.jsp

ここも特に解説することはないです。

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http:/
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>1ページ目</title>
</head>
<body>
<form action="page2" method="GET">
    <input type="submit" value="DBから値を取得！">
</form>
</body>
</html>
```

Page2Servlet

```java
package dbkadai1.controller;

import java.io.IOException;

@WebServlet("/page2")
public class Page2Servlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request,HttpServletResponse response)
                                    throws ServletException,IOException{


        ////////////////////////////////////
        //データベースからユーザー名を取得する
        DBModel dbModel = new DBModel();
        List<String> list = dbModel.getMailList();

        ////////////////////////////////////
        //リクエストスコープに取得した情報を設定する
        request.setAttribute("list", list);

        ////////////////////////////////////
        //画面遷移
        RequestDispatcher dis=request.getRequestDispatcher("WEB-INF/jsp/page2.jsp");
        dis.forward(request, response);
    }
}
```

DBMoldel の呼び出し
DB の情報を List という形で
受け取っている

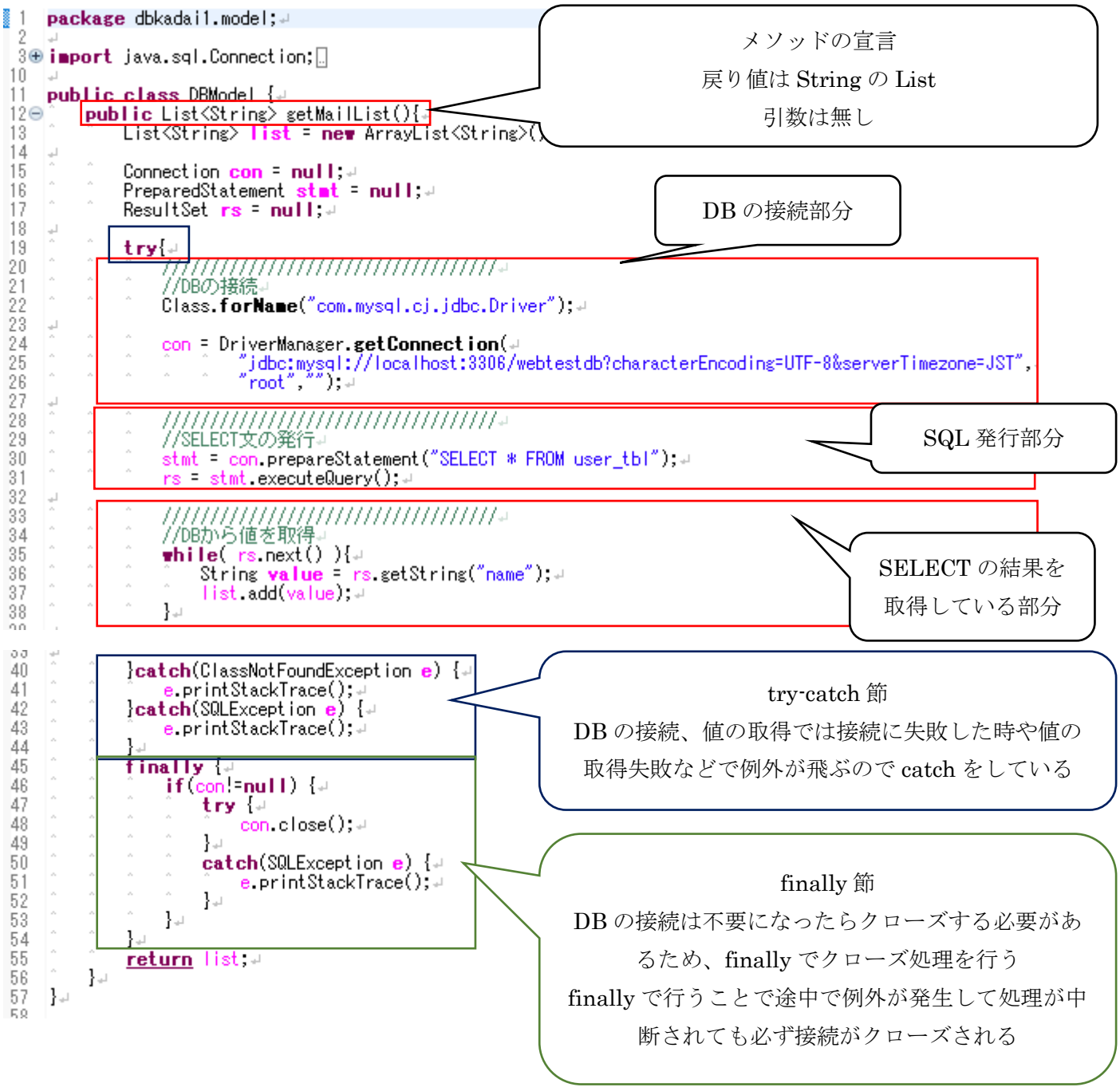
Model から受け取った List を
リクエストスコープにセットする

※List について

List クラスは、配列と同じように情報をまとめて持つことができます。今回の場合、DB から名前を複数件取ってきているので、List クラスを用いて複数件の名前をひと塊として扱っています。

List クラスの宣言は特殊です。<>の間に「何のリストか？」を書きます。
今回は、名前つまり「文字列」のリストなので <> の中が String になっています。
これを「String の List」という言い方をします。

DBModel




※List について 2

List 自体はインターフェースであるため、インスタンスを作成することができない（インターフェースを忘れた人は GitHub の資料の中にある「Java 入門編まとめ.pdf」を見てみよう）。

なのでインスタンス（new する）を作成するときは、List の実装クラスを指定する必要がある。

よくつかう List の実装クラスは ArrayList なので、まずは List のインスタンスを作るときは ArrayList を使うと覚えておこう！

※まずは↓を List のインスタンスを作るときの「ひな型」として覚えておこう

```
List<String> list = new ArrayList<String>();
```

String の List
という意味

List の変数名

ArrayList のコンストラクタ呼び出し
<String>の部分は List<String>と必ず合わせる

それぞれの処理をピックアップ

〇DB の接続

JDBC のクラスを DriverManager に登録している
理屈は結構むつかしいので「なんか知らんけど書かかんといけん」で OK

```
Class.forName("com.mysql.cj.jdbc.Driver");
con = DriverManager.getConnection(
    "jdbc:mysql://localhost:3306/webtestdb?characterEncoding=UTF-8&serverTimezone=JST",
    "root","");
```

DB 接続の処理を行い接続オブジェクト Connection クラスのインスタンスを取得している
SQL を発行するためには、まず DB の接続が必要。

getConnection の引数について

第 1 引数は「接続文字列」と言われる最も重要な文字列です。

ここは気にしなくていい
接続するときの文字コードやタイムゾーンの設定をしている

第 1 引数である以下について詳しく説明

```
"jdbc:mysql://localhost:3306/webtestdb?characterEncoding=UTF-8&serverTimezone=JST",
```

MySQL の JDBC で接続してくださいという意味

接続先：ポート番号 localhost:3306 は『**自分自身の MySQL にポート番号 3306 を通して接続する**』という意味

接続するデータベースの名前


第 2 引数、第 3 引数

ユーザー名とパスワードを指定します

例の場合は

```
"root","";
```

なので、ユーザー名が「root」でパスワードは無し。

もし、MySQL のパスワードを変更しているのであれば、第 3 引数は変更してあげてください。

※もし DB に接続できない人が居たら 「MySQL が起動しているか」「ユーザー名パスワードが間違っていないか」を見てみよう

〇SELECT の発行

con（Connection オブジェクト）の prepareStament メソッドを使って SQL 文をセットします。セットした SQL 文は PreparedStatement というクラスのインスタンスに格納されます

```
stmt = con.prepareStatement("SELECT * FROM user_tbl");
rs = stmt.executeQuery();
```

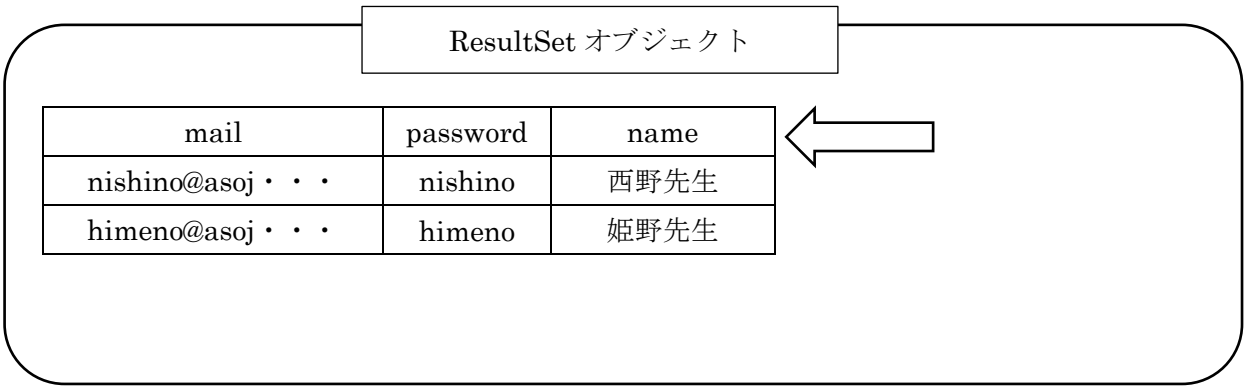
PreparedStatement のインスタンスが持っている executeQuery を実行することで SQL 文が発行されます。
結果は ResultSet クラスのインスタンスという形で受け取ります

【ポイント】
**SQL 文を発行するときは、1 手順でいきなり発行するのではなく、PreparedStatement を使ってセットした後に executeQuery で発行するという 2 手順踏むことを覚えておこう！**

〇DB からの値取得

SQL の SELECT 結果は ResultSet オブジェクトに格納されています。

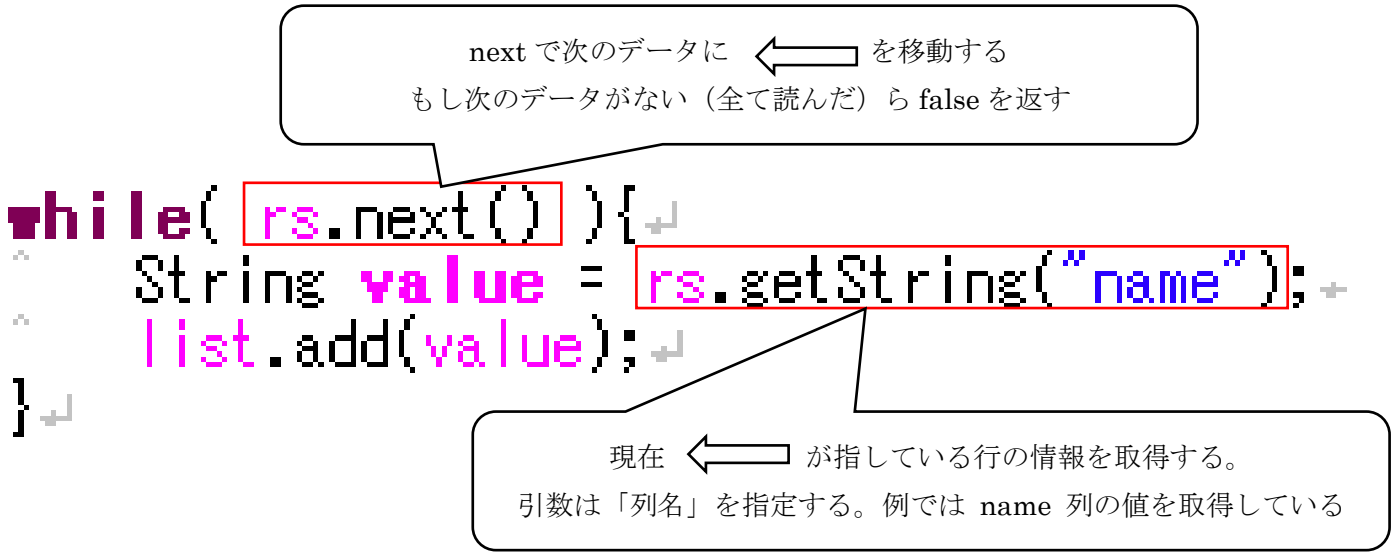
イメージとしては以下のような感じです




このように ResultSet オブジェクトの中には SELECT した結果が全て入っています

⇦は、現在何件目を読もうとしているのかを指しており、**取得直後は「0 件目」を指している**

ことに注意しましょう。




※List について 3

List に項目を追加するときは add メソッドを使う。

注意点としては add するオブジェクトは宣言の時に <> で指定したものと同じにしなければならない点。

今回は List<String>と宣言しているので、add できるのは String のオブジェクト（String 変数）のみ。

※getString について

ResultSet から値を取得する getString は DB の型によってメソッドを変更する必要があります。

主に以下のようなものがあります

| DB の型 | ResultSet の取得メソッド |
|---|---|
| 文字列（VARCHAR） | getString |
| 数値（INT、INTEGER など） | getInt |
| 日付（DATE、TIMESTAMP） | getTimestamp |

page2.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="java.util.List" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>2ページ目</title>
</head>
<body>
<%
    List<String> list = (List<String>)request.getAttribute("list");
%>
<% for( String name : list){ %>
    <p><%= name %>
<% } %>
</body>
</html>
```

リストはインポートが必要なことに注意しよう

getAttribute を使ってサーブレットでセットした値を取得するキャストしなくてはいけないことに注意

ループをしてリストの中身のデータを表示している

※拡張 for 文について

List や配列の中身を表示するときによく使われるループに「拡張 for 文」というのがあります。

List や配列の 1 件 1 件の中身を受け取る変数を宣言します
この場合 String のリストなので String の変数を宣言しています。もし Integer のリストなら Integer の変数を宣言します

```java
List<String> list = new ArrayList<String>();

list.add("aaaaa");
list.add("bbbbb");
list.add("ccccc");

for(String str : list ) {
    System.out.println(str);
}
```

ここには List は配列のインスタンスが入った変数を指定します

List にある分ループします（この場合 3 回）。
1 回目のループでは str に”aaaaa”が 2 回目のループでは str に”bbbbb”が 3 回目のループでは str に”ccccc”が入ります

以上でソースコードの解説は終わりです。

DB の接続、SQL の発行、取得をしっかり押さえましょう。ついでに List や拡張 for 文の使い方も覚えよう！

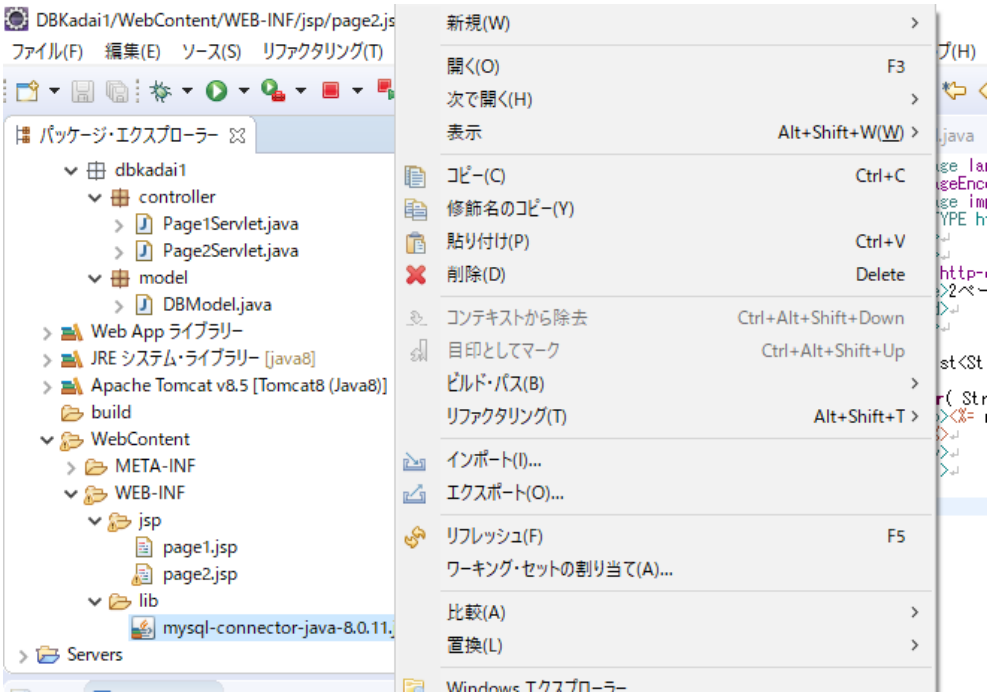
最後に、演習課題を行うときの注意点！

# 「JDBC ドライバは lib フォルダに置くだけだはだめです」

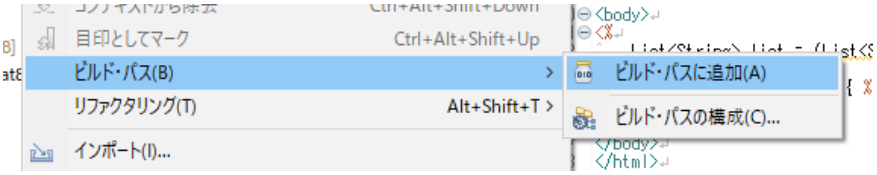
JDBC ドライバ（mysql-connector-java-8.0.11.jar）は、WEB-INF の lib フォルダにコピーするだけではだめです。
「ビルドパスに追加」してあげる必要があります。

ビルドパスに追加する手順です。

１． JDBC ドライバを選択して右クリックする

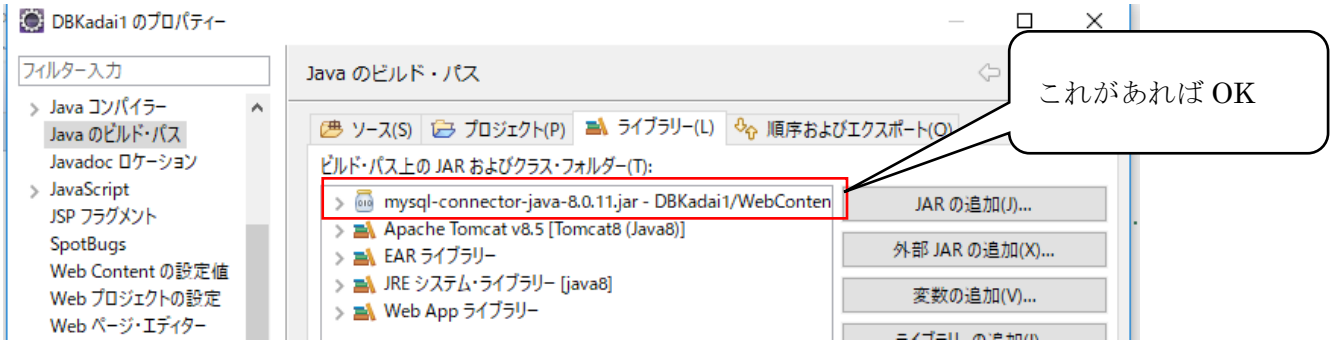



「ビルド・パス」－「ビルドパスに追加」を選択する




以上で OK（簡単だぜ！）

念のため、右クリックー「ビルドパス」―「ビルドパスの構成」でちゃんと追加されているかを確認しましょう。



注意点２

例外が発生したときは・・・・

```
The last packet sent successfully to the server was 0 milliseconds ago. The driver has not received any packets from the server.
    at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
    at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
    at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
    at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
    at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:59)
    at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:103)
    at com.mysql.cj.exceptions.ExceptionFactory.createException(ExceptionFactory.java:149)
    at com.mysql.cj.exceptions.ExceptionFactory.createCommunicationsException(ExceptionFactory.java:165)
    at com.mysql.cj.protocol.a.NativeSocketConnection.connect(NativeSocketConnection.java:92)
    at com.mysql.cj.NativeSession.connect(NativeSession.java:152)
    at com.mysql.cj.jdbc.ConnectionImpl.connectOneTryOnly(ConnectionImpl.java:982)
    at com.mysql.cj.jdbc.ConnectionImpl.createNewIO(ConnectionImpl.java:852)
    ... 31 more
Caused by: java.net.ConnectException: Connection refused: connect
    at java.net.DualStackPlainSocketImpl.connect0(Native Method)
    at java.net.DualStackPlainSocketImpl.socketConnect(DualStackPlainSocketImpl.java:79)
    at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:350)
    at java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:206)
    at java.net.AbstractPlainSocketImpl.connect(AbstractPlainSocketImpl.java:188)
    at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:172)
    at java.net.SocksSocketImpl.connect(SocksSocketImpl.java:392)
    at java.net.Socket.connect(Socket.java:589)
    at com.mysql.cj.protocol.StandardSocketFactory.connect(StandardSocketFactory.java:173)
    at com.mysql.cj.protocol.a.NativeSocketConnection.connect(NativeSocketConnection.java:66)
    ... 34 more
```

このように「**Connection refused**」と出ている場合は接続に失敗しています。**MySQL が起動しているかをチェック**してみてください。

```
java.sql.SQLException: Access denied for user 'root'@'localhost' (using password: YES)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:127)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:95)
    at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:122)
    at com.mysql.cj.jdbc.ConnectionImpl.createNewIO(ConnectionImpl.java:862)
    at com.mysql.cj.jdbc.ConnectionImpl.<init>(ConnectionImpl.java:444)
    at com.mysql.cj.jdbc.ConnectionImpl.getInstance(ConnectionImpl.java:230)
    at com.mysql.cj.jdbc.NonRegisteringDriver.connect(NonRegisteringDriver.java:226)
    at java.sql.DriverManager.getConnection(DriverManager.java:664)
    at java.sql.DriverManager.getConnection(DriverManager.java:247)
    at dbkadai1.model.DBModel.getMailList(DBModel.java:24)
    at dbkadai1.controller.Page2Servlet.doGet(Page2Servlet.java:26)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:635)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:231)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:193)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:199)
    at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:96)
    at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:493)
    at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:137)
    at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:81)
    at org.apache.catalina.valves.AbstractAccessLogValve.invoke(AbstractAccessLogValve.java:660)
    at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:87)
    at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:343)
    at org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:798)
    at org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:66)
    at org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:806)
    at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1498)
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
    at java.lang.Thread.run(Thread.java:748)
```

「**Access denied**」はアクセス拒否。プログラムの getConnection で指定しているユーザー名とパスワードが不一致の場合が多いので、そこを確認しましょう。

```
java.sql.SQLSyntaxErrorException: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right sy
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:118)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:95)
    at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:122)
    at com.mysql.cj.jdbc.ClientPreparedStatement.executeInternal(ClientPreparedStatement.java:960)
    at com.mysql.cj.jdbc.ClientPreparedStatement.executeQuery(ClientPreparedStatement.java:1019)
    at dbkadai1.model.DBModel.getMailList(DBModel.java:31)
    at dbkadai1.controller.Page2Servlet.doGet(Page2Servlet.java:26)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:635)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:231)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:193)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:199)
    at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:96)
    at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:493)
    at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:137)
    at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:81)
    at org.apache.catalina.valves.AbstractAccessLogValve.invoke(AbstractAccessLogValve.java:660)
    at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:87)
    at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:343)
    at org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:798)
    at org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:66)
    at org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:806)
    at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1498)
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
    at java.lang.Thread.run(Thread.java:748)
```

「**SQLSyntaxErrorException**」と出ているときは SQL 文が間違っています。

prepareStatement で指定している **SQL 文に誤りが無いかを確認**しましょう。

**SQL 文中に全角文字が入ってても NG ですよ！！！！**

```
java.sql.SQLException: Column 'namea' not found.
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:127)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:95)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:87)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:61)
    at com.mysql.cj.jdbc.result.ResultSetImpl.findColumn(ResultSetImpl.java:572)
    at com.mysql.cj.jdbc.result.ResultSetImpl.getString(ResultSetImpl.java:890)
    at dbkadai1.model.DBModel.getMailList(DBModel.java:36)
    at dbkadai1.controller.Page2Servlet.doGet(Page2Servlet.java:26)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:635)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:231)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:193)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:199)
    at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:96)
    at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:493)
    at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:137)
    at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:81)
    at org.apache.catalina.valves.AbstractAccessLogValve.invoke(AbstractAccessLogValve.java:660)
    at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:87)
    at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:343)
    at org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:798)
    at org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:66)
    at org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:806)
    at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1498)
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
    at java.lang.Thread.run(Thread.java:748)
```

「**Column XXXX not found**」と出るときは、SELECT の結果取得のところで getString の引数で渡している列名がおかしい（rs.getString(“xxxx”)←この「xxxx」の部分）場合、もしくは prepareStatement で指定している SQL 文で存在しない列名を書いている場合が多いです。そこをチェックしてみよう！

```
java.lang.ClassNotFoundException: com.mysql.cj.jdbc.Drivera
    at org.apache.catalina.loader.WebappClassLoaderBase.loadClass(WebappClassLoaderBase.java:1364)
    at org.apache.catalina.loader.WebappClassLoaderBase.loadClass(WebappClassLoaderBase.java:1185)
    at java.lang.Class.forName0(Native Method)
    at java.lang.Class.forName(Class.java:264)
    at dbkadai1.model.DBModel.getMailList(DBModel.java:22)
    at dbkadai1.controller.Page2Servlet.doGet(Page2Servlet.java:26)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:635)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:231)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:193)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:166)
    at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:199)
    at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:96)
    at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:493)
    at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:137)
    at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:81)
    at org.apache.catalina.valves.AbstractAccessLogValve.invoke(AbstractAccessLogValve.java:660)
    at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:87)
    at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:343)
    at org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:798)
    at org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:66)
    at org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:806)
    at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1498)
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
    at java.lang.Thread.run(Thread.java:748)
```

「ClassNotFountException:com.mysql.cj.Driver」と出ているようなときは Class.forName で指定しているドライバ名が間違っている場合が多いので、そこをチェックしてみよう

## この資料と、サンプルソースを見て課題に取り組んでみましょう！