

<Servlet を使ったプログラム>

<動的 Web プロジェクトを作成する>

※Web サイトは「動的 Web プロジェクト」で作成してゆく。

[Eclipse の起動]

Eclipse を起動する。

画面表示を「JavaEE」 パースペクティブにする。

[新規動的 Web プロジェクトの作成]

(手順)

プロジェクト・エクスプローラー・ペインの空白部分を右クリックする。

新規 -> その他 -> Web -> 動的 Web プロジェクトの順でメニュー選択する。

「次へ」 ボタンを押下する。

「プロジェクト名」を入力する。

※今回、プロジェクト名は HelloServlet と入力する。

「デフォルト・ロケーションを使用」をチェックする。

「ターゲット・ランタイム」にて Tomcat v.8.0 を選択する。

(確認)

(1) 「動的 web モジュールバージョン」が自動設定されていること。

(2) 「構成」がデフォルト構成にて設定されていること。

「次へ」 ボタンを押下する。

(3) 「ビルド・パス上のソース・フォルダ」にて src が表示されていること。

(4) 「デフォルト出力フォルダー」にて build\classes が表示されていること。

「次へ」 ボタンを押下する。

(5) 「コンテキスト・ルート」にて作成した動的 Web プロジェクト名 (今回は HelloServlet) が表示されていること。

(6) 「コンテンツ・ディレクトリー」にて WebContent が表示されていること。

(7) 「web.xml デプロイメント記述子の生成」を必ずチェックする。

「完了」を押下する。

(8) プロジェクト・エクスプローラー・ペインに今回作成した動的 Web プロジェクトのフォルダが作成されていること。

(9) メニュー -> プロジェクト -> プロパティの順に選択する。

リソース を選択する。

テキスト・ファイルのエンコードを確認する。

※UTF-8 にて設定されていること。

HelloServlet

＜JSP ファイルの作成＞

JSP ファイル「index.jsp」を作成しましょう。

WebContent フォルダを右クリック -> 新規 -> JSP ファイルの順に選択する。

「ファイル名」欄に index.jsp と記述する。
「完了」ボタンを押下する。

JSP をプログラミングしましょう。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HelloServlet</title>
</head>
<body>
<input type="button" value="HelloServlet" onClick="location.href='HelloServlet'">
</body>
</html>
```

<サーブレットの作成>

サーブレットを作成しましょう。

src フォルダを右クリックする。
新規をクリックする。
サーブレットをクリックする。
「クラス名」に **HelloServlet** と入力する。
次へボタンをクリックする。
「名前」に **HelloServlet** と表示されていることを確認する。
次へボタンをクリックする。
doGet と **doPost** にチェックがついていることを確認する。
完了ボタンをクリックする。

サーブレットをプログラミングしましょう。
※今回は、doPost メソッド部分を削除してプログラミングしてゆきます。
※本来は/** */のようなコメントをつけるべきですが、今回はこれも削除してプログラミングしてゆきます。(実際の処理を見やすくするため)

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/HelloServlet")
public class HelloServlet extends HttpServlet {

    public HelloServlet() {
        super();
    }

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

```
        PrintWriter out = response.getWriter();
        out.println("<HTML>");
        out.println("<BODY>");
        out.println("<H3>Hello Servlet!</H3>");
        out.println("</BODY>");
        out.println("</HTML>");
    }
}
```

<web.xml の記述>

web.xml は画面を表示する為の設定ファイル。

※必要に応じて画面中央あたりの「ソース」タブをクリックする。

WebContent¥WEB-INF のweb.xml をプログラミングしましょう。

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
    <display-name>HelloServlet</display-name>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>

    <servlet>
        <servlet-name>Hello</servlet-name>
        <servlet-class>HelloServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Hello</servlet-name>
        <url-pattern>/HelloServlet/HelloServlet</url-pattern>
    </servlet-mapping>

</web-app>
```

※<servlet-name>はサーブレット（Java のプログラム）と実際に表示する際の紐づけに使用しています。

ここで入力したものが、紐づけのキーワードとなります。

入力する内容は英数字であれば問題ありません。

※<servlet-class>はサーブレット名をプログラムします。作成した Java ファイル名を入力します。

※<url-pattern>は<servlet-class>にプログラムしたサーブレットを実際に呼び出す URL の一部を入力します。

（例）

http://localhost:8080/xxxxx/xxxxx

この場合、

/xxxxx/xxxxx の部分を入力します

HelloServlet を実行しましょう。

プロジェクトフォルダ「HelloServlet」を右クリックする。

実行をクリックする。

「サーバーで実行」をクリックする。

実行するサーバーを選択する。（今回は Tomcat8.0）

「次へ」をクリックする。

使用可能欄→Eclipse 内で作成された動的 Web プロジェクトのリスト

構成済み →実際に動かしたい動的 Web プロジェクト

「追加」「削除」「すべて追加」「すべて除去」ボタンを使って、実際に動かしたい動的 Web プロジェクトを構成済みに含めるように調整しておく。

「完了」ボタンをクリック。

WelcomeServlet

index.jsp の<body>タグにプログラムを追加しましょう。

```
<input type="button" value="問い合わせ" onClick="location.href='welcome.jsp'">
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HelloServlet</title>
</head>
<body>
<input type="button" value="HelloServlet" onClick="location.href='HelloServlet'">
<input type="button" value="WelcomeServlet" onClick="location.href='welcome.jsp'">
</body>
</html>
```

JSP ファイル「welcome.jsp」を作成しましょう。

WebContent フォルダを右クリック → 新規 → JSP ファイルの順に選択する。

「ファイル名」欄に welcome.jsp と記述する。

「完了」ボタンを押下する。

JSP ファイル「welcome.jsp」にプログラミングしましょう。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
```

```
<head>
<meta charset="UTF-8">
<title>WELCOME</title>
</head>
<body>
名前とパスワードを入力してください。
<form method="post" action="WelcomeServlet">
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
</body></html>
```

<サーブレットの作成>

サーブレットを作成しましょう。

src フォルダを右クリックする。
新規をクリックする。
サーブレットをクリックする。
「クラス名」に **WelcomeServlet** と入力する。
次へボタンをクリックする。
「名前」に **WelcomeServlet** と表示されていることを確認する。
次へボタンをクリックする。
doGet と **doPost** にチェックがついていることを確認する。
完了ボタンをクリックする。

サーブレットをプログラミングしましょう。
※今回は、doGet メソッド部分を削除してプログラミングしてゆきます。
※本来は/***/のようなコメントをつけるべきですが、今回はこれも削除してプログラミングしてゆきます。(実際の処理を見やすくするため)

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
```

```

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/WelcomeServlet")

public class WelcomeServlet extends HttpServlet {

    public WelcomeServlet() {
        super();
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        request.setCharacterEncoding("UTF-8");
        response.setContentType("text/html; charset=UTF-8");
        String username = request.getParameter("username");

        System.out.println(username);

        PrintWriter out=response.getWriter();
        out.println("<html><head></head><body><br>"+username+"さん、ようこそ！
</body></html>");
    }
}

```

<web.xml の記述>

web.xml は画面を表示する為の設定ファイル。
※必要に応じて画面中央あたりの「ソース」タブをクリックする。

WebContent¥WEB-INF のweb.xml をプログラミングしましょう。

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
    <display-name>HelloServlet</display-name>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>

```



```
<servlet>
  <servlet-name>Hello</servlet-name>
  <servlet-class>HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Hello</servlet-name>
  <url-pattern>/HelloServlet/HelloServlet</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>Welcome</servlet-name>
  <servlet-class>WelcomeServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Welcome</servlet-name>
  <url-pattern>/HelloServlet/WelcomeServlet</url-pattern>
</servlet-mapping>

</web-app>
```

HelloServletを実行しましょう。

プロジェクトフォルダ「HelloServlet」を右クリックする。
実行をクリックする。
「サーバーで実行」をクリックする。
実行するサーバーを選択する。(今回は Tomcat8.0)
「次へ」をクリックする。

使用可能欄→Eclipse 内で作成された動的 Web プロジェクトのリスト
構成済み →実際に動かしたい動的 Web プロジェクト

「追加」「削除」「すべて追加」「すべて除去」ボタンを使って、実際に動かしたい動的 Web プロジェクトを構成済みに含めるように調整しておく。

「完了」ボタンをクリック。

InquiryServlet

index.jsp の<body>タグにプログラムを追加しましょう。

```
<input type="button" value="問い合わせ" onClick="location.href='inquiry.jsp'">
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HelloServlet</title>
</head>
<body>
<input type="button" value="HelloServlet" onClick="location.href='HelloServlet'">
<input type="button" value="WelcomeServlet" onClick="location.href='welcome.jsp'">
<input type="button" value="問い合わせ" onClick="location.href='inquiry.jsp'">
</body>
</html>
```

JSP ファイル「inquiry.jsp」を作成しましょう。

WebContent フォルダを右クリック -> 新規 -> JSP ファイルの順に選択する。

「ファイル名」欄に inquiry.jsp と記述する。

「完了」ボタンを押下する。

JSP ファイル「inquiry.jsp」にプログラミングしましょう。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
```

```
<html>
<head>
<meta charset="UTF-8">
<title>お問い合わせフォーム</title>
</head>
<body>
  <form method="post" action="InquiryServlet">
    名前:<br> <input type="text" name="name"><br>
    お問い合わせの種類:<br> <select name="qtype">
      <option value="company">会社について</option>
      <option value="product">製品について</option>
      <option value="support">アフターサポートについて</option>
    </select> <br> お問い合わせ内容:<br>
    <textarea name="body">
  </textarea>
  <br> <input type="submit" value="登録">
</form>
</body>
</html>
```

<サーブレットの作成>

サーブレットを作成しましょう。

src フォルダを右クリックする。
新規をクリックする。
サーブレットをクリックする。
「クラス名」に **InquiryServlet** と入力する。
次へボタンをクリックする。
「名前」に **InquiryServlet** と表示されていることを確認する。
次へボタンをクリックする。
doGet と **doPost** にチェックがついていることを確認する。
完了ボタンをクリックする。

サーブレットをプログラミングしましょう。
※今回は、doGet メソッド部分を削除してプログラミングしてゆきます。
※本来は/***/のようなコメントをつけるべきですが、今回はこれも削除してプログラミングしてゆきます。(実際の処理を見やすくするため)

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/InquiryServlet")
public class InquiryServlet extends HttpServlet {

    public InquiryServlet() {
        super();
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        request.setCharacterEncoding("UTF-8");
        response.setContentType("text/html; charset=UTF-8");
        String name = request.getParameter("name");

        System.out.println(name);

        PrintWriter out=response.getWriter();
        out.println("<html><head></head><body><br>"+name+"さん、お問合せありがとうございました
</body></html>");
    }
}
```

<web.xml の記述>

WebContent¥WEB-INF のweb.xml をプログラミングしましょう。

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
    <display-name>HelloServlet</display-name>
```

```
<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>

<servlet>
  <servlet-name>Hello</servlet-name>
  <servlet-class>HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Hello</servlet-name>
  <url-pattern>/HelloServlet/HelloServlet</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>Welcome</servlet-name>
  <servlet-class>WelcomeServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Welcome</servlet-name>
  <url-pattern>/HelloServlet/WelcomeServlet</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>Inquiry</servlet-name>
  <servlet-class>InquiryServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Inquiry</servlet-name>
  <url-pattern>/HelloServlet/InquiryServlet</url-pattern>
</servlet-mapping>

</web-app>
```

HelloServlet を実行しまし ょ う。

TestServlet

index.jsp の<body>タグにプログラムを追加しましょう。

GET 通信

```
<form method="get" action="TestServlet">
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
```

POST 通信

```
<form method="post" action="TestServlet">
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HelloServlet</title>
</head>
<body>
<input type="button" value="HelloServlet" onClick="location.href='HelloServlet'">
<input type="button" value="WelcomeServlet" onClick="location.href='welcome.jsp'">
<input type="button" value="問い合わせ" onClick="location.href='inquiry.jsp'">
GET 通信
<form method="get" action="TestServlet">
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
POST 通信
<form method="post" action="TestServlet">
```

```
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
</body>
</html>
```

<サーブレットの作成>

サーブレットを作成しましょう。

src フォルダを右クリックする。
新規をクリックする。
サーブレットをクリックする。
「クラス名」に **TestServlet** と入力する。
次へボタンをクリックする。
「名前」に **TestServlet** と表示されていることを確認する。
次へボタンをクリックする。
doGet と **doPost** にチェックがついていることを確認する。
完了ボタンをクリックする。

サーブレットをプログラミングしましょう。
※本来は/** */のようなコメントをつけるべきですが、今回はこれも削除してプログラミングしてゆきます。(実際の処理を見やすくするため)

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/TestServlet")
public class TestServlet extends HttpServlet {

    public TestServlet() {
```

```

        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        request.setCharacterEncoding("UTF-8");
        response.setContentType("text/html; charset=UTF-8");

        String username=request.getParameter("username");
        String password=request.getParameter("password");
        System.out.println(username);
        System.out.println(password);

        PrintWriter out=response.getWriter();

        out.println("<html><head></head><body><br>" + username + "<br>" + password + "</body></html>");
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        request.setCharacterEncoding("UTF-8");
        response.setContentType("text/html; charset=UTF-8");

        String username=request.getParameter("username");
        String password=request.getParameter("password");
        System.out.println(username);
        System.out.println(password);

        PrintWriter out=response.getWriter();

        out.println("<html><head></head><body><br>" + username + "<br>" + password + "</body></html>");
    }
}

```

<web.xml の記述>

WebContent¥WEB-INF のweb.xml をプログラミングしましょう。

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

```



```
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
  <display-name>HelloServlet</display-name>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>Hello</servlet-name>
    <servlet-class>HelloServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Hello</servlet-name>
    <url-pattern>/HelloServlet/HelloServlet</url-pattern>
  </servlet-mapping>

  <servlet>
    <servlet-name>Welcome</servlet-name>
    <servlet-class>WelcomeServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Welcome</servlet-name>
    <url-pattern>/HelloServlet/WelcomeServlet</url-pattern>
  </servlet-mapping>

  <servlet>
    <servlet-name>Inquiry</servlet-name>
    <servlet-class>InquiryServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Inquiry</servlet-name>
    <url-pattern>/HelloServlet/InquiryServlet</url-pattern>
  </servlet-mapping>

  <servlet>
    <servlet-name>Test</servlet-name>
    <servlet-class>TestServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Test</servlet-name>
    <url-pattern>/HelloServlet/TestServlet</url-pattern>
```

```
</servlet-mapping>
```

```
</web-app>
```

HelloServlet を実行しましょう。

GET 通信と POST 通信でそれぞれ入力と送信ボタンを押して、アドレス（URL）の違いを確認してみましょう。

※GET 通信を使った場合には、送信される情報が URL に表示されます。
※POST 通信を使った場合には、送信される情報は URL に表示されない。

MySQLServlet

※ここでは、mysql-connector-java-x.x.xx-bin.jar を使ってゆきます。

<JDBC 用 jar ファイルの保存>

WebContent¥WEB-INF¥lib に mysql-connector-java-x.x.xx-bin.jar を貼り付けます。
(これで動的 Web プロジェクトでは、Java-DB 接続のプログラミングができます。)

<SQL ファイルの作成>

動的 Web プロジェクトフォルダ (HelloServlet) 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
<code>user_id</code>	<code>int(11)</code>	YES		NULL	
<code>user_name</code>	<code>varchar(255)</code>	YES		NULL	
<code>Password</code>	<code>varchar(255)</code>	YES		NULL	

user_id	user_name	password
1	taro	123
2	jiro	456
3	hanako	789

index.jsp の<body>タグにプログラムを追加しましょう。

```
<input type="button" value="MySQLServlet" onClick="location.href='MySQLServlet'">
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HelloServlet</title>
</head>
<body>
<input type="button" value="HelloServlet" onClick="location.href='HelloServlet'">
<input type="button" value="WelcomeServlet" onClick="location.href='welcome.jsp'">
<input type="button" value="問い合わせ" onClick="location.href='inquiry.jsp'">
<input type="button" value="MySQLServlet" onClick="location.href='MySQLServlet'">
GET 通信
<form method="get" action="TestServlet">
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
POST 通信
<form method="post" action="TestServlet">
<input type="text" name="username">
<input type="password" name="password">
<input type="submit" value="送信">
</form>
</body>
</html>
```

<サーブレットの作成>

サーブレットを作成しましょう。

src フォルダを右クリックする。
新規をクリックする。
サーブレットをクリックする。
「クラス名」に **MySQLServlet** と入力する。
次へボタンをクリックする。
「名前」に **MySQLServlet** と表示されていることを確認する。
次へボタンをクリックする。
doGet と doPost にチェックがついていることを確認する。
完了ボタンをクリックする。

サーブレットをプログラミングしましょう。
※今回は、doPost メソッド部分を削除してプログラミングしてゆきます。
※本来は/***/のようなコメントをつけるべきですが、今回はこれも削除してプログラミングしてゆきます。(実際の処理を見やすくするため)

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/MySQLServlet")
public class MySQLServlet extends HttpServlet {

    public MySQLServlet() {
        super();
    }
}
```

```

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    request.setCharacterEncoding("UTF-8");
    response.setContentType("text/html; charset=UTF-8");

    PrintWriter out = response.getWriter();

    out.println("<html>");
    out.println("<head>");
    out.println("<title>データベーステスト</title>");
    out.println("</head>");
    out.println("<body>");

    Connection conn = null;
    String url = "jdbc:mysql://localhost/testdb";
    String user = "root";
    String password = "mysql";

    try {
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        conn = DriverManager.getConnection(url, user, password);

        Statement stmt = conn.createStatement();
        String sql = "SELECT * FROM test_table";
        ResultSet rs = stmt.executeQuery(sql);

        while(rs.next()) {
            int userId = rs.getInt("user_id");
            String userName = rs.getString("user_name");
            String userPassword = rs.getString("password");
            out.println("<p>");
            out.println("ユーザー ID:" + userId + ", ユーザー名:" + userName + ", パスワード:" +
userPassword);
            out.println("</p>");
        }

        rs.close();
        stmt.close();
    } catch (ClassNotFoundException e) {
        out.println("ClassNotFoundException:" + e.getMessage());
    } catch (SQLException e) {

```



```

        out.println("SQLException:" + e.getMessage());
    } catch (Exception e) {
        out.println("Exception:" + e.getMessage());
    } finally {
        try {
            if (conn != null) {
                conn.close();
            }
        } catch (SQLException e) {
            out.println("SQLException:" + e.getMessage());
        }
    }
}

out.println("</body>");
out.println("</html>");
}
}

```

<web.xml の記述>

WebContent¥WEB-INF のweb.xml をプログラミングしましょう。

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
    <display-name>HelloServlet</display-name>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>

    <servlet>
        <servlet-name>Hello</servlet-name>
        <servlet-class>HelloServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Hello</servlet-name>
        <url-pattern>/HelloServlet/HelloServlet</url-pattern>
    </servlet-mapping>

    <servlet>

```

```
<servlet-name>Welcome</servlet-name>
<servlet-class>WelcomeServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Welcome</servlet-name>
  <url-pattern>/HelloServlet/WelcomeServlet</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>Inquiry</servlet-name>
  <servlet-class>InquiryServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Inquiry</servlet-name>
  <url-pattern>/HelloServlet/InquiryServlet</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>Test</servlet-name>
  <servlet-class>TestServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Test</servlet-name>
  <url-pattern>/HelloServlet/TestServlet</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>MySQL</servlet-name>
  <servlet-class>MySQLServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>MySQL</servlet-name>
  <url-pattern>/HelloServlet/MySQLServlet</url-pattern>
</servlet-mapping>

</web-app>
```

HelloServlet.を実行しましょう。
