

MVC のサンプル

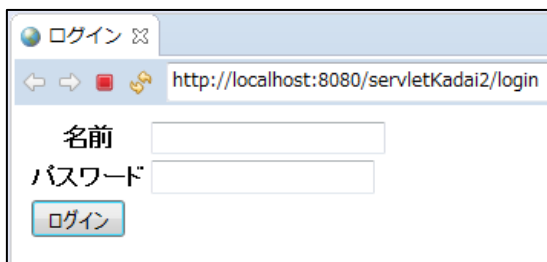
以前つくった、なんちゃってログイン画面を MVC で書き直してみよう！

完成品

○1 ページ目

U R L : <http://localhost:8080/servletKadai3/login>

画面：



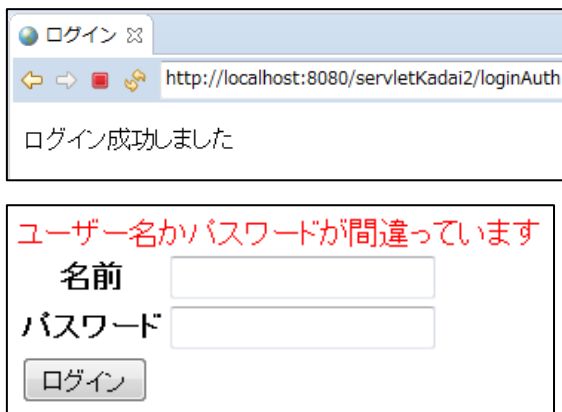
The screenshot shows a web browser window with the title "ログイン". The address bar displays "http://localhost:8080/servletKadai2/login". The main content area contains two text input fields labeled "名前" (Name) and "パスワード" (Password), and a blue button labeled "ログイン" (Login).

仕様：名前とパスワードの初期状態は空っぽ。send ボタンをクリックすると
2 ページ目へ画面遷移する。なお 2 ページ目の通信方法は P O S T 通信とする。

○2 ページ目

U R L : <http://localhost:8080/servletKadai3/loginAuth>

画面：



The first screenshot shows a web browser window with the title "ログイン". The address bar displays "http://localhost:8080/servletKadai2/loginAuth". The main content area displays the message "ログイン成功しました" (Login successful).
The second screenshot shows the same login page with a red error message at the top: "ユーザー名かパスワードが間違っています" (Username or password is incorrect). Below the message are the "名前" (Name) and "パスワード" (Password) input fields and the "ログイン" (Login) button.

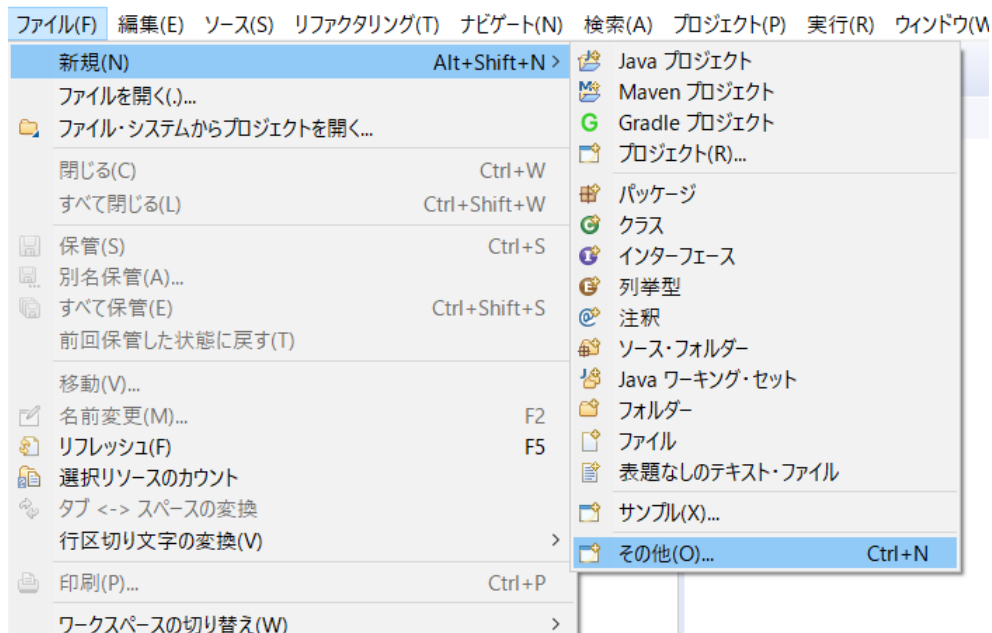
仕様：1 ページ目に入力された値が
名前=nishino、パスワード=1111 の場合は画面に「ログイン成功しました」
を表示する。それ以外の場合はログイン画面に戻り
「ユーザー名かパスワードが間違っています」を表示する

1. プロジェクトを作成する

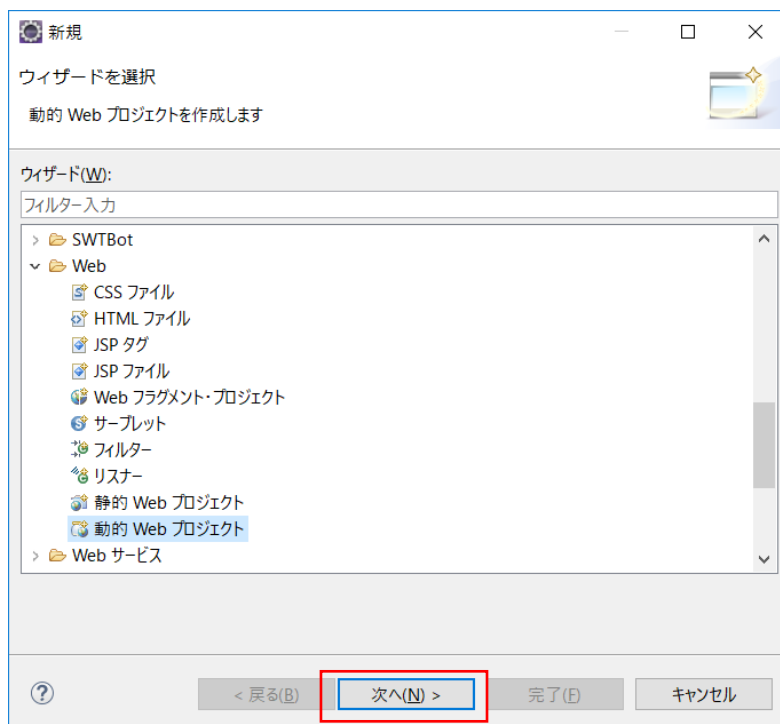
※ワークスペースは以前作った、「mvcsample」にしましょう。

※以前あった「3_EclipseWeb プロジェクト作成.pdf」と同じ手順です。

「ファイル」－「その他」を選択する



「動的プロジェクト」を選び、次へをクリックします



プロジェクト名を入力します。「mvcsample」にしましょう
入力したら完了をクリックします

新規動的 Web プロジェクト

動的 Web プロジェクト
スタンドアロン動的 Web プロジェクトを作成、または新規、既存のエンタープライズ・アプリケーションに追加します。

プロジェクト名(M):

プロジェクトのロケーション
☒ デフォルト・ロケーションを使用(D)
ロケーション(L): C:\work\servletsample2\sample4

ターゲット・ランタイム(U)
Tomcat8 (Java8)

動的 web モジュールバージョン(V)
3.1

構成(C)
Tomcat8 (Java8) デフォルト構成

Tomcat8 (Java8) ランタイムを操作するための良い出発点です。後で新しい機能をプロジェクトに追加するために追加ファセットをインストールすることができます

EAR メンバーシップ

2. 1 ページ目のコントローラと JSP を記述する

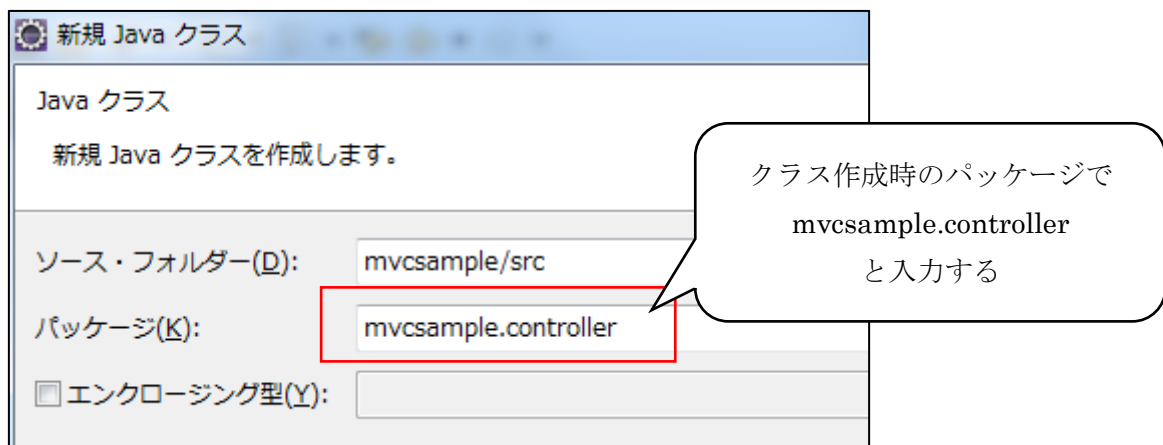
1 ページ目のコントローラと JSP を記述します。

まずはサーブレットを作ります。ここではサーブレットの記述内容のみ記載します。

サーブレットの作り方を忘れた人は「サーブレットサンプル.pdf」を参考にして下さい

○LoginController を作り、以下のような記述をします。

【注意】 パッケージは `mvcsample.controller` にしましょう！



```
LoginController.java
1 package mvcsample.controller;
2
3 import java.io.IOException;
4
5 import javax.servlet.RequestDispatcher;
6 import javax.servlet.ServletException;
7 import javax.servlet.annotation.WebServlet;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 @WebServlet("/login")
13 public class LoginController extends HttpServlet {
14
15     @Override
16     protected void doGet(HttpServletRequest request, HttpServletResponse response)
17         throws ServletException, IOException {
18
19         RequestDispatcher dispatcher =
20             request.getRequestDispatcher("WEB-INF/jsp/login_view.jsp");
21         dispatcher.forward(request, response);
22     }
23 }
```

○次に login_view.jsp を作り、以下のように記述します。

login_view.jsp は、先日同様、WebContent\WEB-INF の直下に jsp フォルダを作成し、jsp フォルダに作成します。

```
login_view.jsp
1  <%@ page language="java" contentType="text/html; charset=UTF-8"%>
2  <% pageEncoding="UTF-8"%>
3  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.
4  </html>
5  <head>
6  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"%>
7  <title>ログイン</title>
8  </head>
9  <body>
10 <%
11     String errorMsg = (String)request.getAttribute("errorMsg");
12 %>
13 <% if( errorMsg != null ){ %>
14     <font color="red"><%=errorMsg %></font>
15 <% } %>
16 <form action="loginAuth" method="POST">
17     <table>
18     <tr>
19         <th>名前</th>
20         <td><input type="text" name="account" /></td>
21     </tr>
22     <tr>
23         <th>パスワード</th>
24         <td><input type="password" name="password" /></td>
25     </tr>
26     <tr>
27         <td colspan="2"><input type="submit" value="ログイン" /></td>
28     </tr>
29     </table>
30 </form>
31 </body>
32 </html>
```

2. 2 ページ目のコントローラとモデル、JSP を記述する

2 ページ目のコントローラとモデル、JSP を記述します。

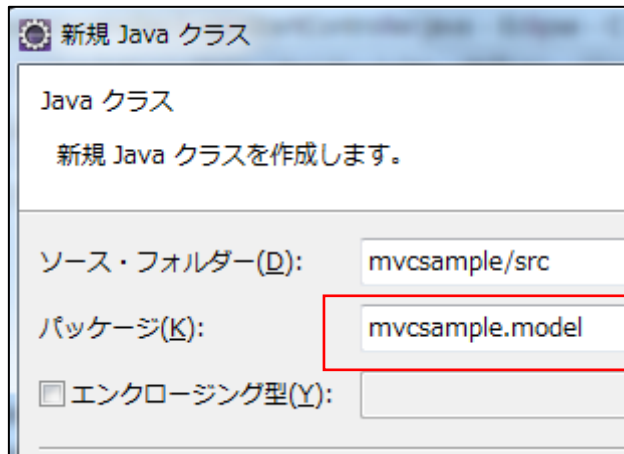
○LoginStartController を作り、以下のような記述をします。

【注意】 パッケージは **mvcsample.controller** にしましょう！

```
LoginStartController.java
1 package mvcsample.controller;
2
3 import java.io.IOException;
4
13 @WebServlet("/loginAuth")
14 public class LoginStartController extends HttpServlet {
15
16     @Override
17     protected void doPost(HttpServletRequest request, HttpServletResponse response)
18         throws ServletException, IOException {
19
20         //値を取得
21         String account = request.getParameter("account");
22         String password = request.getParameter("password");
23
24         //★モデルのインスタンスを生成する★
25         LoginModel loginModel = new LoginModel();
26
27         //パスワードと名前を比較して、結果によってメッセージを変更する
28         String message = "";
29         String forward = "";
30
31         //★モデルクラスのloginメソッドを呼び出して結果を受け取る★
32         boolean result = loginModel.login(account, password);
33         if (result) {
34             message = "ログイン成功しました";
35             forward = "WEB-INF/jsp/result_view.jsp";
36             //メッセージを結果画面へ送る
37             request.setAttribute("msg", message);
38         } else {
39             message = "ログイン失敗しました";
40             forward = "WEB-INF/jsp/login_view.jsp";
41             //メッセージを結果画面へ送る
42             request.setAttribute("errorMsg", "ユーザー名かパスワードが間違っています");
43         }
44
45         RequestDispatcher dispatcher =
46             request.getRequestDispatcher(forward);
47         dispatcher.forward(request, response);
48     }
49 }
50
51
```

○次にモデルを作ります。もではログイン用のモデルで、クラス名は `LoginModel` としましょう。

【注意】 パッケージは `mvcsample.model` にしましょう！



クラス作成時のパッケージで
`mvcsample.model`
と入力する

`LoginModel` の実装は以下の通りです。

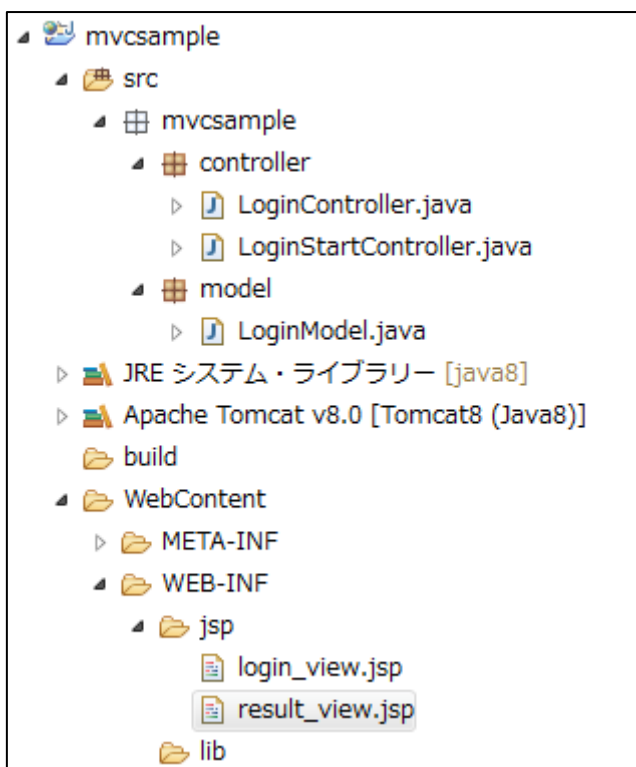
```
1 package mvcsample.model;
2
3 /**
4  * ログイン用のモデルクラス
5  *
6  * @author nishino
7  */
8
9 public class LoginModel {
10
11     /**
12      * ログイン処理を行い結果を返す
13      *
14      * @param account アカウント
15      * @param password パスワード
16      * @return ログイン結果
17      */
18     public boolean login(String account, String password) {
19         boolean result = false;
20
21         //ログイン処理（アカウントとパスワードのチェック）を行う
22         if (account.equals("nishino") && password.equals("1111")) {
23             result = true;
24         } else {
25             result = false;
26         }
27
28         return result;
29     }
30 }
31
```

○次に result_view.jsp を作り、以下のように記述します。

result_view.jsp も同様に、WebContent\WEB-INF の直下の jsp フォルダに作成します。

```
result_view.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"%>
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http:
4 <html>
5 <head>
6 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"%>
7 <title>ログイン</title>
8 </head>
9 <body>
10 <%
11   String message = (String)request.getAttribute("msg");
12   %>
13 <p><%=message %></p>
14 </body>
15 </html>
```

ここまでで、パッケージエクスプローラーは以下のような表示になっているはずです。



4. 実行する

実行するには

Page1Servlet.java を選択して、右クリック－「実行」－「1 サーバーで実行」を選択する