ECサイト初級

ログイン機能の作成

3時間目

作業目次

作成機能一覧

1. ログイン認証機能

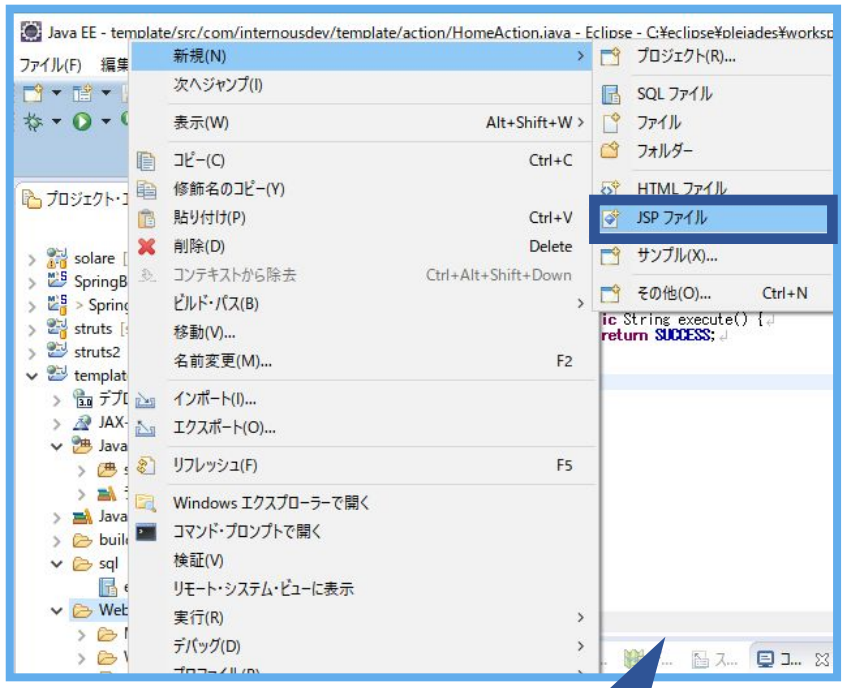
2. ユーザー登録機能

3. 商品購入機能

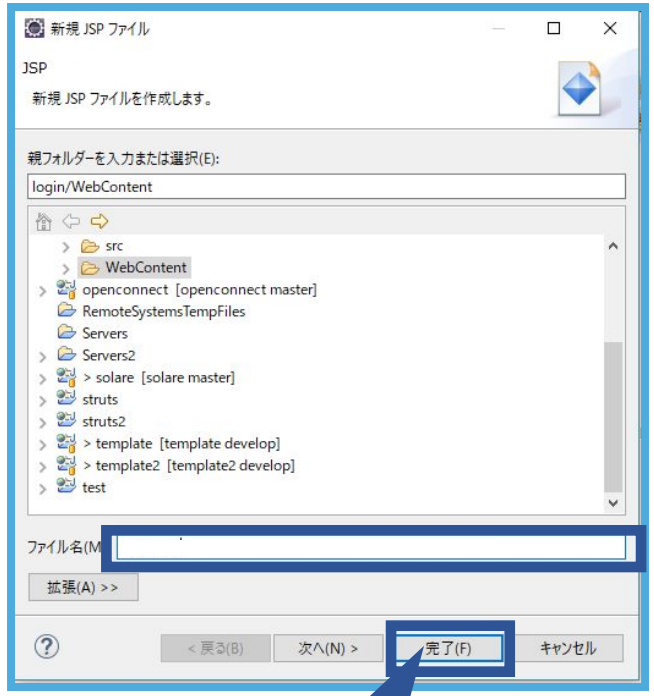
4. 購入履歴機能

JSPファイルの作成

1 home.jsp



①「プロジェクト」「WebContent」を選択し、「右クリック」「新規」「JSPファイル」を選択します。



②「名前(M):」欄に「home.jsp」を入力し、完了ボタンをクリックします。

home.jspの作成

③ 以下の内容を写経します。

home.jsp(jspファイル)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
        pageEncoding="UTF-8"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html>
<html>
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <meta http-equiv="Content-Style-Type" content="text/css" />
        <meta http-equiv="Content-Script-Type" content="text/javascript" />
        <meta http-equiv="imagetoolbar" content="no" />
        <meta name="description" content="" />
        <meta name="keywords" content="" />
        <title>Home画面</title>

        <style type="text/css">
        /* ========TAG LAYOUT========= */
```

次へ続きます。

home.jspの作成

前の続きです。

```
body {
   margin:0;
   padding:0;
   line-height:1.6;
   letter-spacing:1px;
   font-family:Verdana, Helvetica, sans-serif;
   font-size:12px;
   color:#333;
   background:#fff;
}

table {
   text-align:center;
   margin:0 auto;
}
/* =========ID LAYOUT========= */
#top {
   width:780px;
   margin:30px auto;
```

次へ続きます。

home.jspの作成

前の続きです。

```
    border:1px solid #333;
}

#header {
    width: 100%;
    height: 80px;
    background-color: black;
}

#main {
    width: 100%;
    height: 500px;
    text-align: center;
}
#footer {
    width: 100%;
    height: 80px;
    background-color: black;
    clear:both;
}
```

次へ続きます。

home.jspの作成

前の続きです。

```
        </style>
</head>
<body>
        <div id="header">
                <div id="pr">
                </div>
        </div>
        <div id="main">
                <div id="top">
                        <p>Home</p>
                </div>
                <div>
                        <s:form action="HomeAction">
                                <s:submit value="商品購入"/>
                        </s:form>
                </div>
        </div>
        <div id="footer">
                <div id="pr">
```

次へ続きます。

home.jspの作成

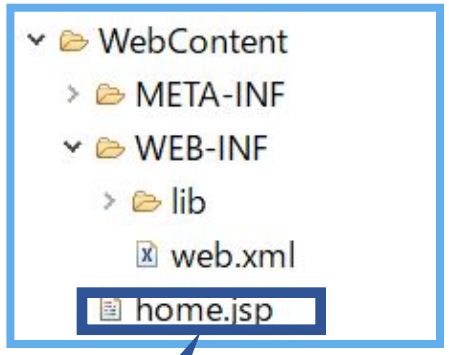
前の続きです。

```
                </div>
            </div>
</body>
</html>
```

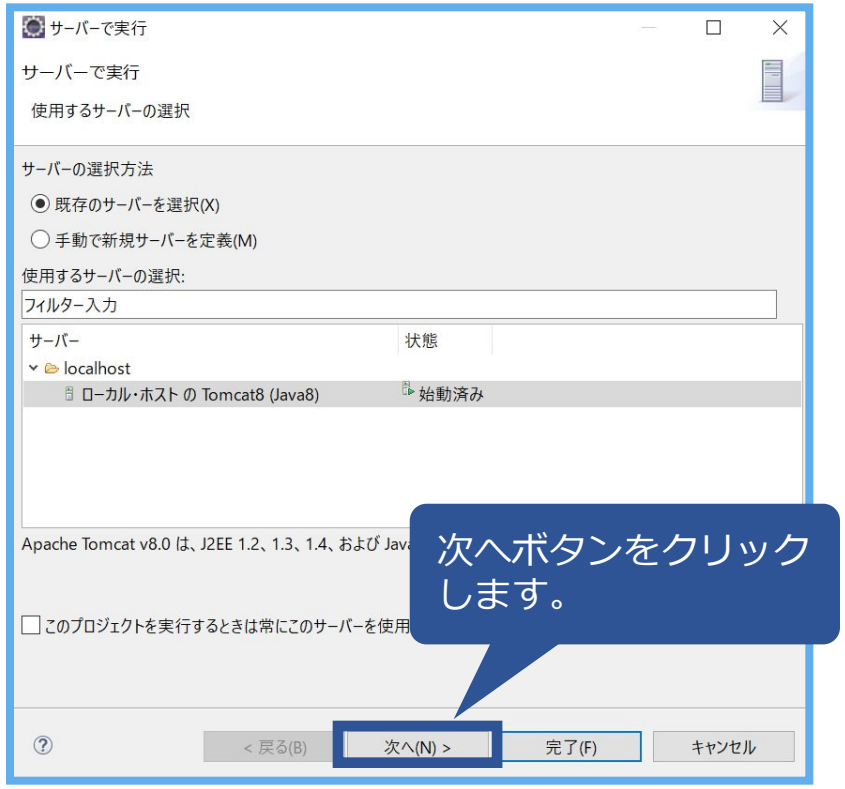
jspファイルの実行方法

解説

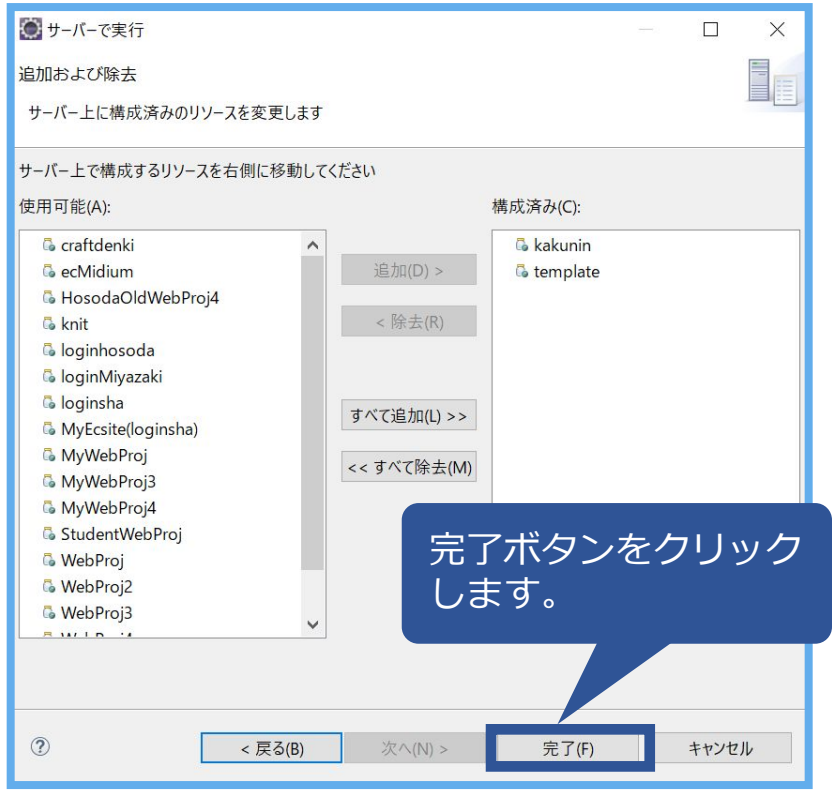
必要なActionファイルを記述していない段階でjspファイルの実行結果を確かめるには、jspファイルだけを単独で実行します。
※ templateプロジェクト自体を実行すると、 必要なActionファイルが無い場合、途中でエラーになってしまいます。



該当のjspファイルを右クリックします。



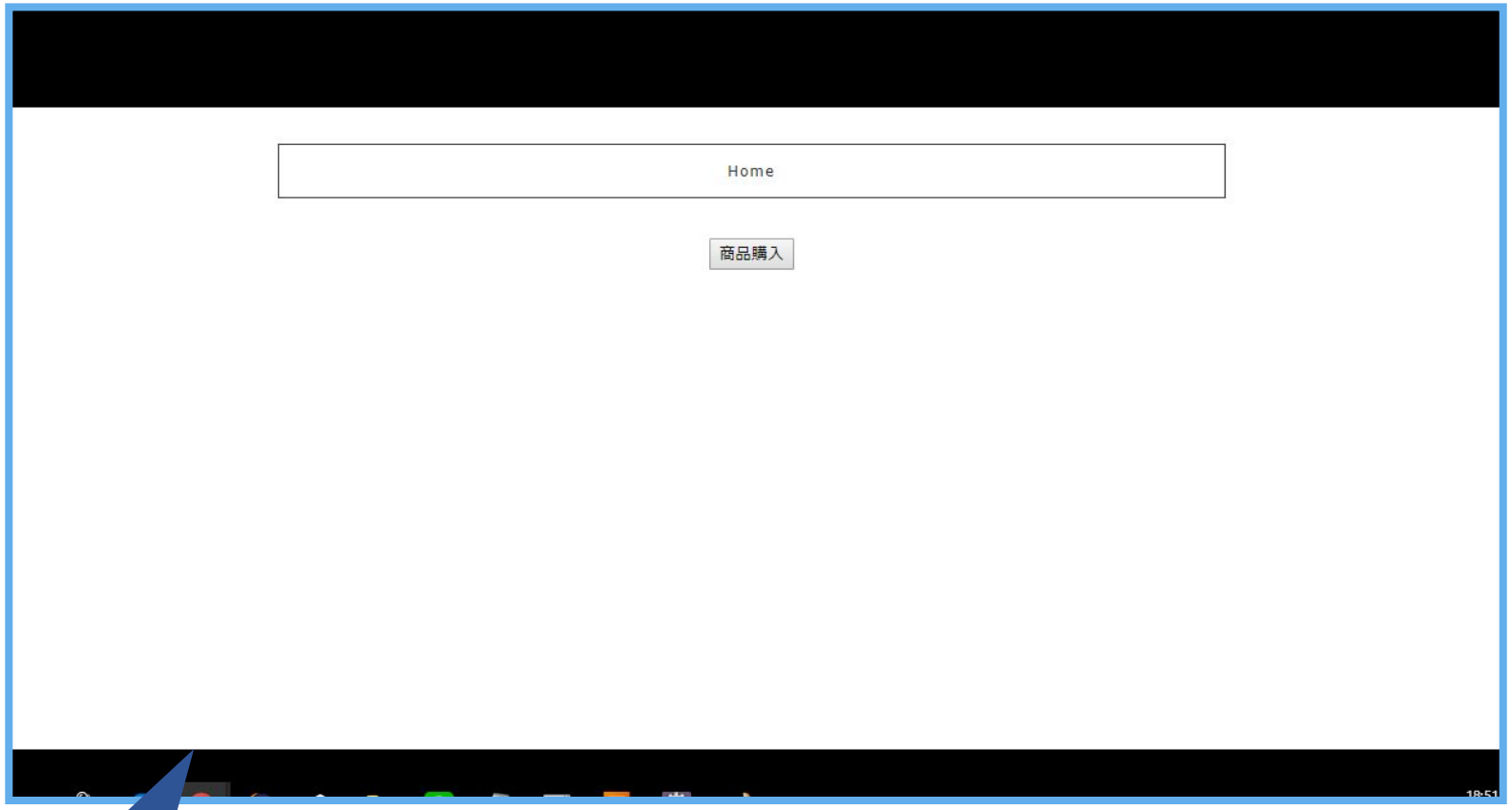
次へボタンをクリックします。



完了ボタンをクリックします。

home.jspの作成

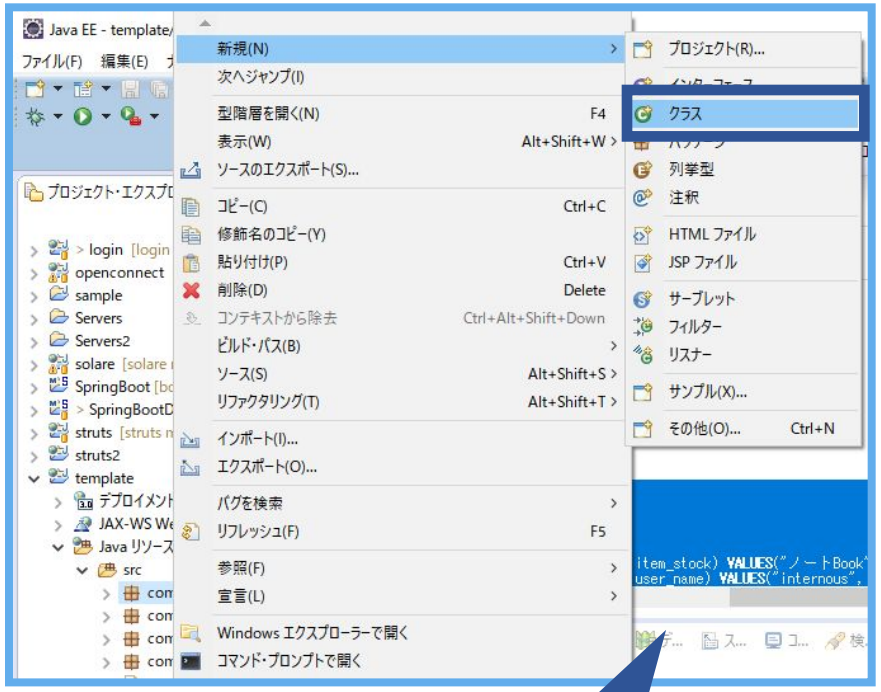
home.jspの画面



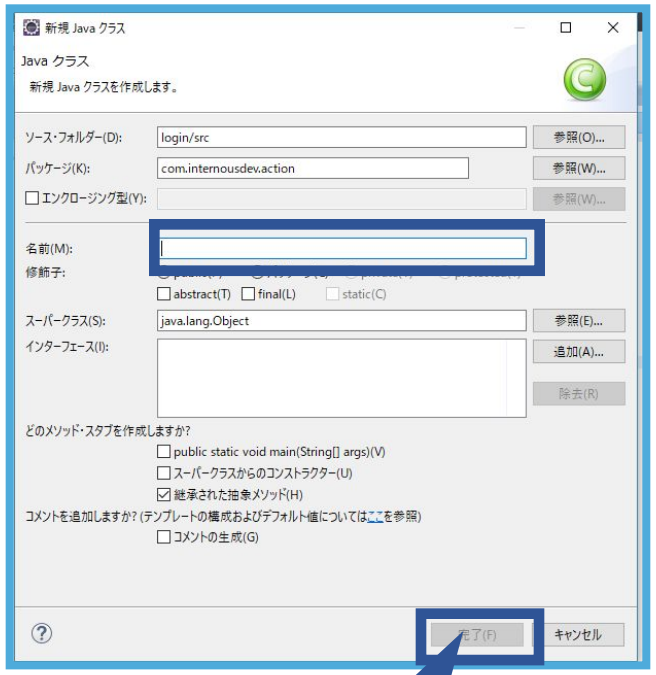
上記の画面が表示できれば成功です。

HomeAction.javaの作成

2 HomeAction.java



① 「com.internousdev.template.action」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「HomeAction」を入力し、完了ボタンをクリックします。

HomeAction.javaの作成

③ 以下の内容を写経します。

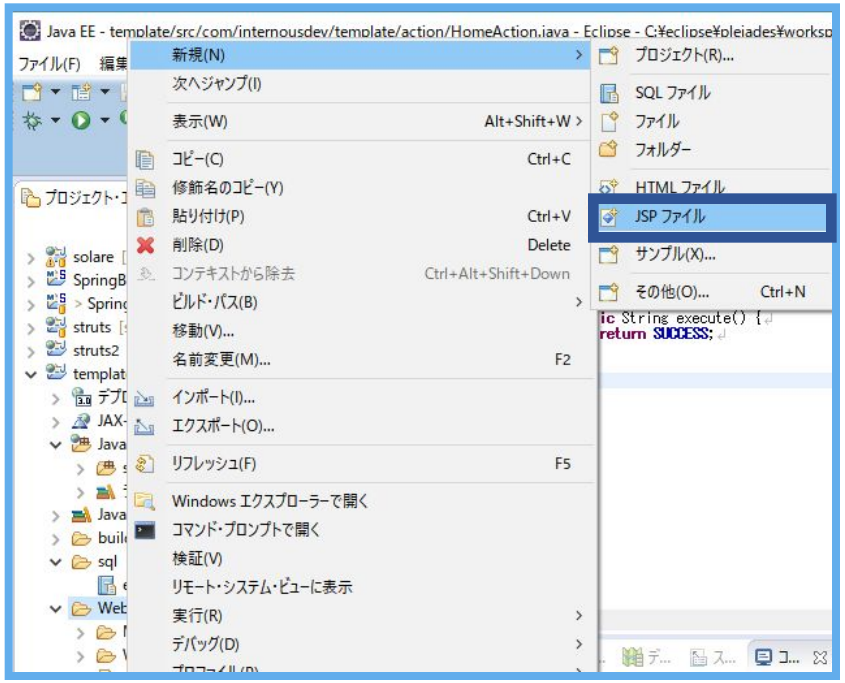
HomeAction(javaファイル)

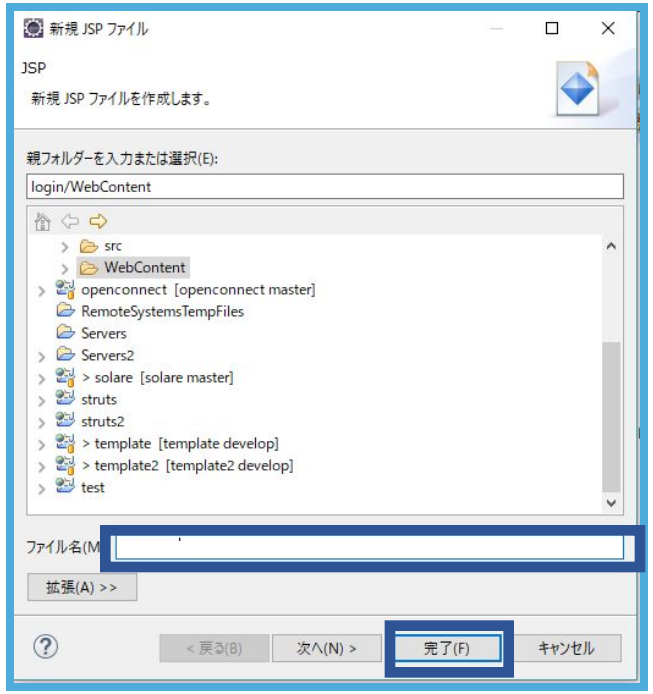
```
package com.internousdev.template.action;

import com.opensymphony.xwork2.ActionSupport;

public class HomeAction extends ActionSupport {
        public String execute() {
                return SUCCESS;
        }
}
```

「excecute」メソッドを実行した後は
"success"文字列を返します。

login.jspの作成

3 login.jsp





① 「プロジェクト」「WebContent」を選択し、「右クリック」「新規」「JSPファイル」を選択します。

② 「名前(M):」欄に「login.jsp」を入力し、完了ボタンをクリックします。

login.jspの作成

③ 以下の内容を写経します。

login.jsp(jspファイル)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
        pageEncoding="UTF-8"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html>
<html>
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <meta http-equiv="Content-Style-Type" content="text/css" />
        <meta http-equiv="Content-Script-Type" content="text/javascript" />
        <meta http-equiv="imagetoolbar" content="no" />
        <meta name="description" content="" />
        <meta name="keywords" content="" />
        <title>Login画面</title>

        <style type="text/css">
        /* ========TAG LAYOUT========= */
```

次へ続きます。

login.jspの作成

前の続きです。

```
body {
   margin:0;
   padding:0;
   line-height:1.6;
   letter-spacing:1px;
   font-family:Verdana, Helvetica, sans-serif;
   font-size:12px;
   color:#333;
   background:#fff;
}

table {
   text-align:center;
   margin:0 auto;
}
/* =========ID LAYOUT========= */
#top {
   width:780px;
   margin:30px auto;
```

次へ続きます。

login.jspの作成

前の続きです。

```
    border:1px solid #333;
}

#header {
    width: 100%;
    height: 80px;
    background-color: black;
}

#main {
    width: 100%;
    height: 500px;
    text-align: center;
}
#footer {
    width: 100%;
    height: 80px;
    background-color: black;
    clear:both;
}
```

次へ続きます。

login.jspの作成

前の続きです。

```
        </style>
</head>
<body>
        <div id="header">
                <div id="pr">
                </div>
        </div>
        <div id="main">
                <div id="top">
                        <p>Login</p>
                </div>
                <div>
                        <h3>商品を購入する際にはログインをお願いします。</h3>
                        <s:form action="LoginAction">
                                <s:textfield name="loginUserId"/>
                                <s:password name="loginPassword"/>
                                <s:submit value="ログイン"/>
                        </s:form>
                        <br/>
```

次へ続きます。

login.jspの作成

前の続きです。

```
                        <div>
                            <span>新規ユーザー登録は
                                <a href='<s:url action="UserCreateAction" />'>こちら</a>
                            </span>
                        </div>
                </div>
        </div>

        <div id="footer">
                <div id="pr">
                </div>
        </div>
</body>
</html>
```

login.jspの作成

解説

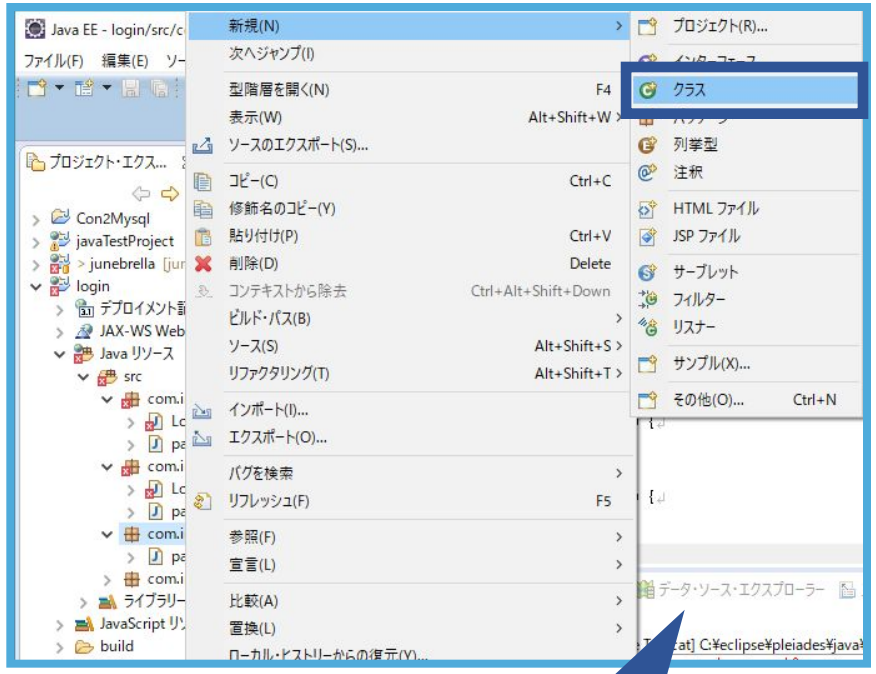
必要なActionファイルを記述していない段階でjspファイルの実行結果を確かめるには、jspファイルだけを単独で実行します。
※ 「template」プロジェクト自体を実行すると、 必要なActionファイルが無い場合、途中でエラーになってしまいます。

login.jspの画面

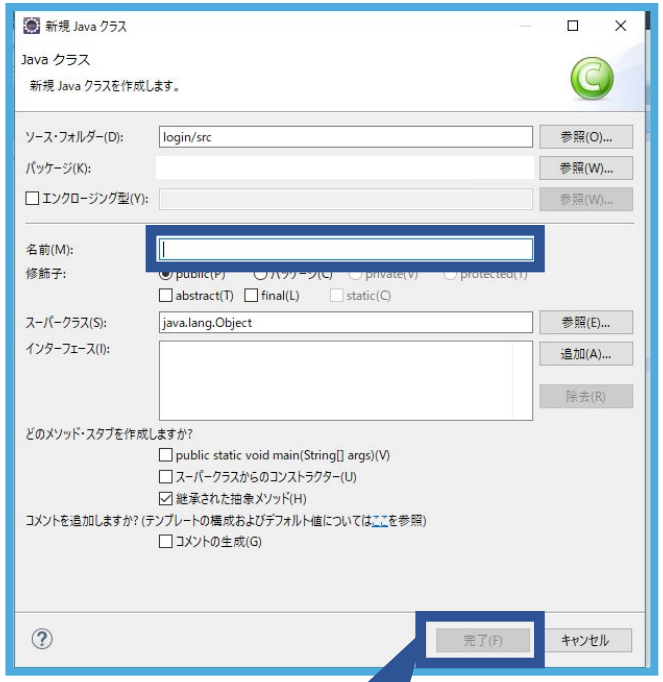
上記の画面が表示できれば成功です。

LoginDTO.javaの作成

4 LoginDTO.java



① 「com.internousdev.template.dto」を選択し、クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「LoginDTO」を入力し、完了ボタンをクリックします。

LoginDTO.javaの作成

③ 以下の内容を写経します。

LoginDTO(javaファイル)

```
package com.internousdev.template.dto;

public class LoginDTO {

        private String loginId;
        private String loginPassword;
        private String userName;
        private boolean loginFlg = false;

        public String getLoginId() {
                return loginId;
        }

        public void setLoginId(String loginId) {
                this.loginId = loginId;
        }
```

次へ続きます。

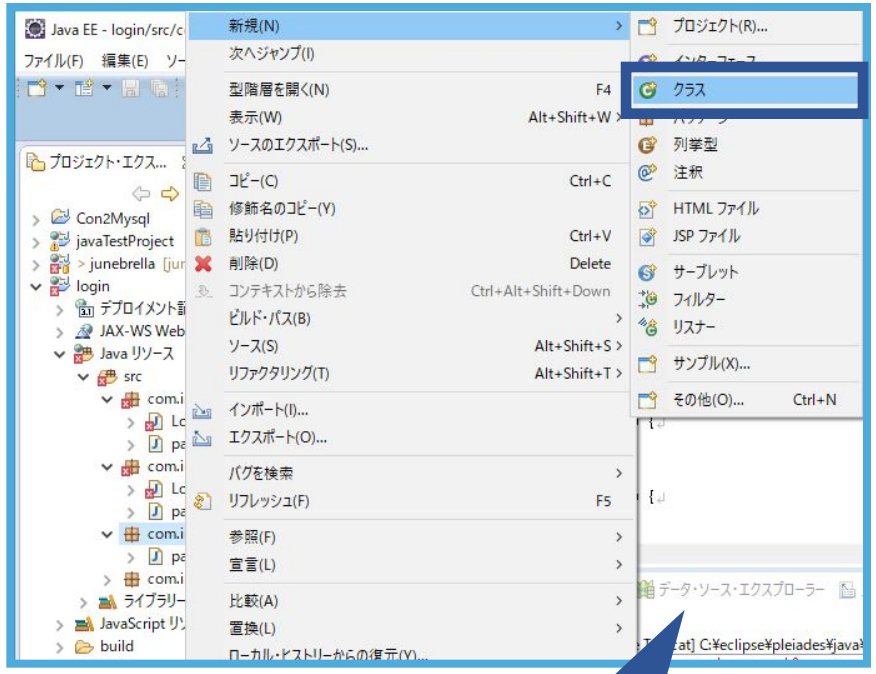
LoginDTO.javaの作成

前の続きです。

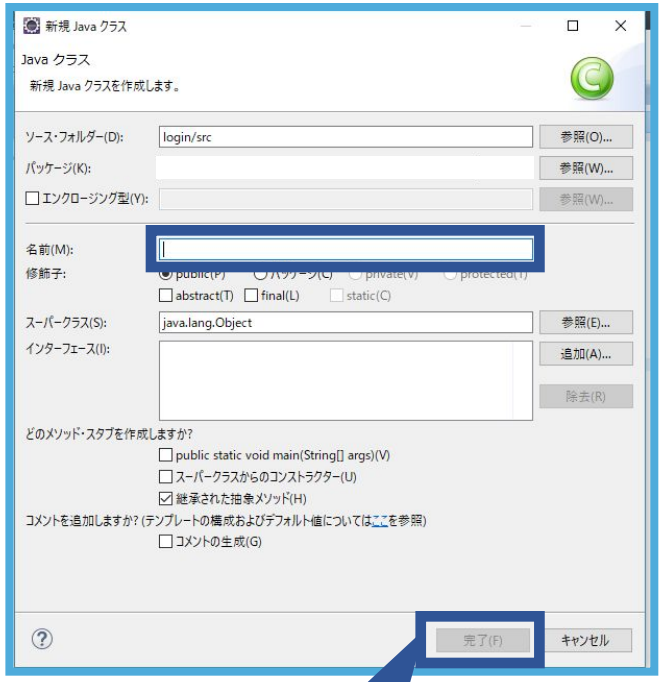
```
        public String getLoginPassword() {
                return loginPassword;
        }
        public void setLoginPassword(String loginPassword) {
                this.loginPassword = loginPassword;
        }
        public String getUserName() {
                return userName;
        }
        public void setUserName(String userName) {
                this.userName = userName;
        }
        public boolean getLoginFlg() {
                return loginFlg;
        }
        public void setLoginFlg(boolean loginFlg) {
                this.loginFlg = loginFlg;
        }
}
```

LoginDAO.javaの作成

5 LoginDAO.java



① 「com.internousdev.template.dao」を選択し、右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「LoginDAO」を入力し、完了ボタンをクリックします。

LoginDAO.javaの作成

③ 以下の内容を写経します。

LoginDAO(javaファイル)

```
package com.internousdev.template.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import com.internousdev.template.dto.LoginDTO;
import com.internousdev.template.util.DBConnector;

public class LoginDAO {

        public LoginDTO getLoginUserInfo(String loginUserId, String loginPassword) {

                DBConnector dbConnector = new DBConnector();
                Connection connection = dbConnector.getConnection();
                LoginDTO loginDTO = new LoginDTO();
                String sql = "SELECT * FROM login_user_transaction where login_id = ? AND
login_pass = ?";
```

メソッド名は処理内容を分かりやすくするために「getLoginUserInfo」にします。

次へ続きます。

LoginDAO.javaの作成

前の続きです。

```
try {
        PreparedStatement preparedStatement = connection.prepareStatement(sql);

        preparedStatement.setString(1, loginUserId);
        preparedStatement.setString(2, loginPassword);

        ResultSet resultSet = preparedStatement.executeQuery();

        if(resultSet.next()) {
                loginDTO.setLoginId(resultSet.getString("login_id"));
                loginDTO.setLoginPassword(resultSet.getString("login_pass"));
                loginDTO.setUserName(resultSet.getString("user_name"));

                if(!(resultSet.getString("login_id").equals(null))) {
                        loginDTO.setLoginFlg(true);
                }
```

次へ続きます。

LoginDAO.javaの作成

前の続きです。

```
                    }
                } catch(Exception e) {
                    e.printStackTrace();
                }
                return loginDTO;
        }

}
```

PreparedStatementの説明

セキュリティを考慮し、javaでは
PreparedStatementを利用します。

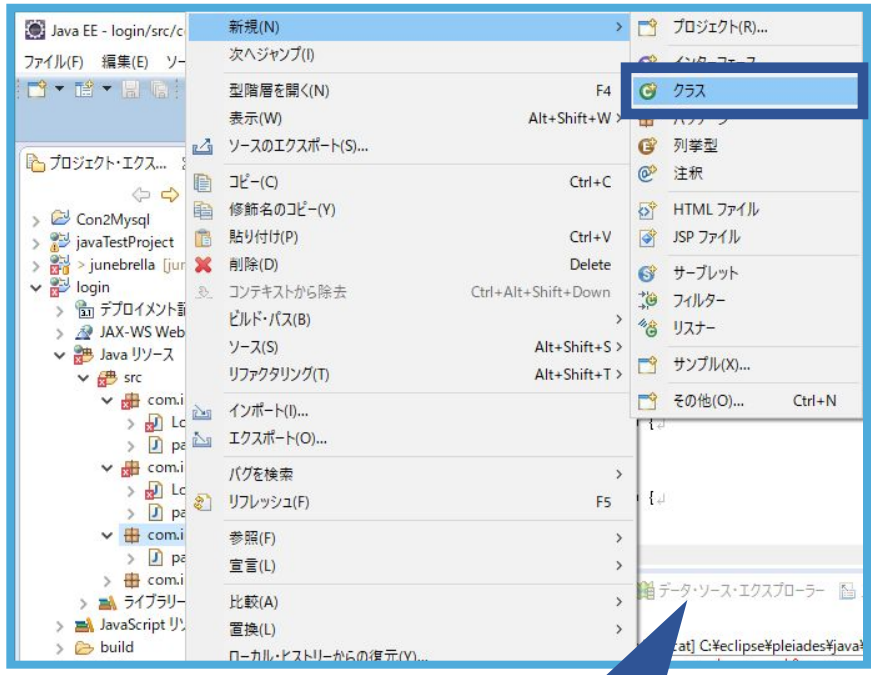
```
PreparedStatement preparedStatement = connection.prepareStatement(sql);

preparedStatement.setString(1, loginUserId);
preparedStatement.setString(2, loginPassword);
```

SQL文の「？」パラメータに指定した
値を挿入することができます。

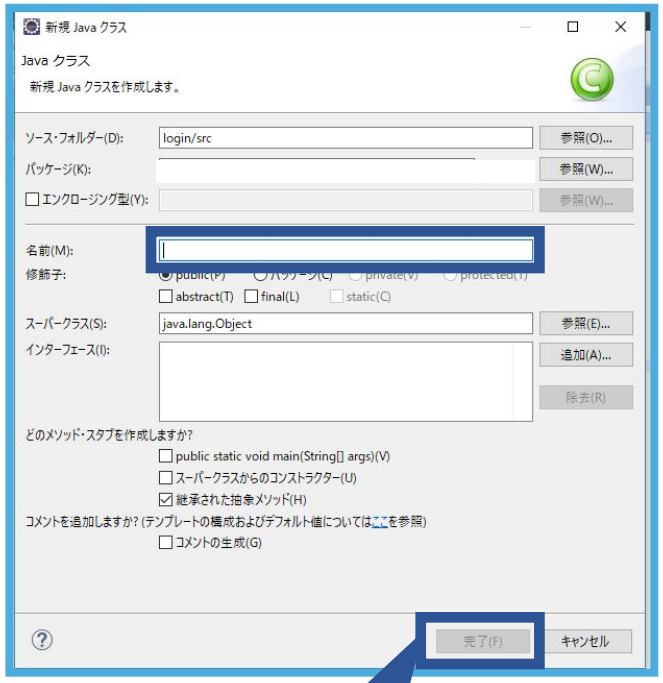
```
String sql = "SELECT * FROM login_user_transaction where login_id = ? AND login_pass = ?";
```

BuyItemDTOの作成

6 DTOクラス



① 「com.internousdev.template.dto」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「BuyItemDTO」を入力し、完了ボタンをクリックします。

BuyItemDTOの作成

BuyItemDTO(javaファイル)

③ 以下の内容を写経します。

```
package com.internousdev.template.dto;

public class BuyItemDTO {
        private int id;
        private String itemName;
        private String itemPrice;

        public String getItemName() {
                return itemName;
        }

        public void setItemName(String itemName) {
                this.itemName = itemName;
        }

        public String getItemPrice() {
                return itemPrice;
        }
```

テーブルカラムに対応したフィールド変数を宣言します。

フィールド変数に対応したGetter Setterを定義します。

次へ続きます。

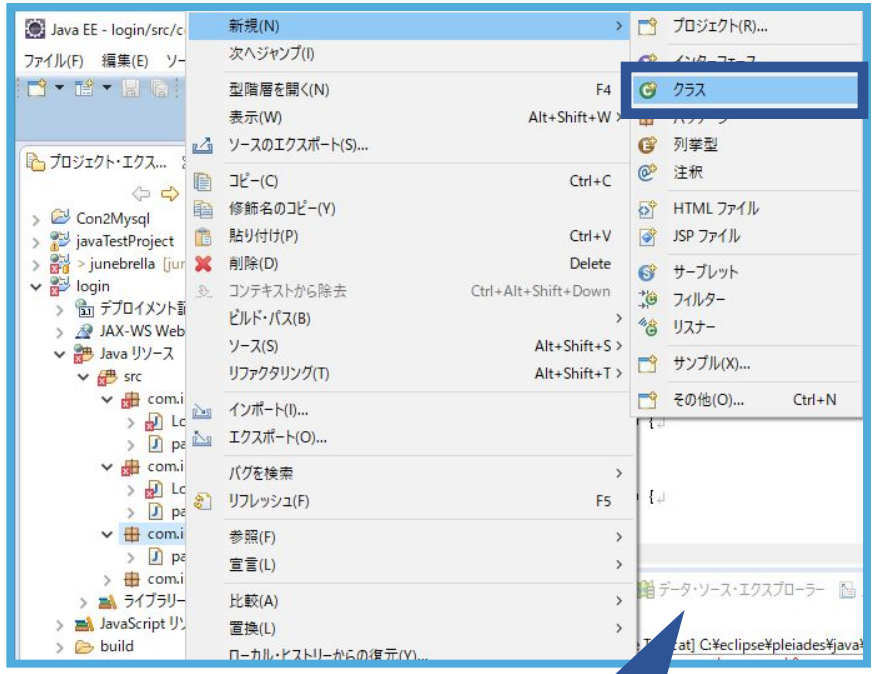
BuyItemDTOの作成

前の続きです。

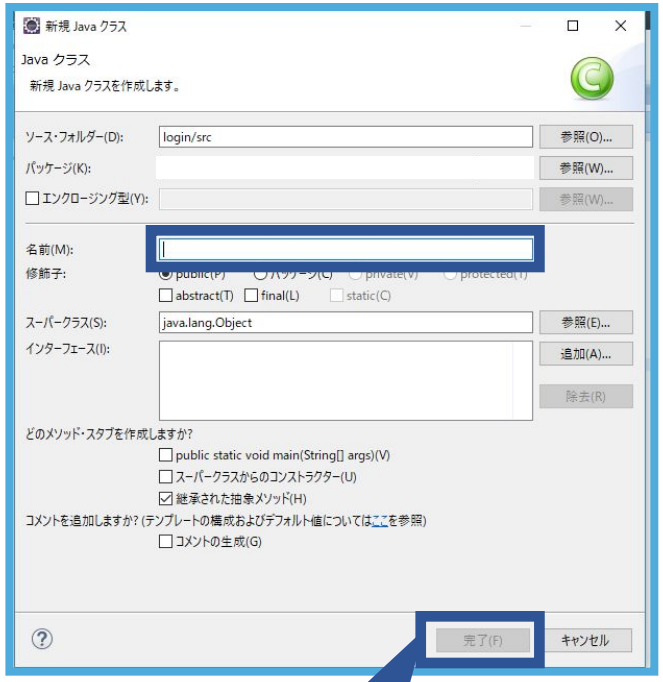
```
	public void setItemPrice(String itemPrice) {
		this.itemPrice = itemPrice;
	}

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}
}
```

BuyItemDAO.javaの作成

7 BuyItemDAO.java



① 「com.internousdev.template.dao」を選択し、右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「BuyItemDAO」を入力し、完了ボタンをクリックします。

BuyItemDAO.javaの作成

③ 以下の内容を写経します。

BuyItemDAO(javaファイル)

```
package com.internousdev.template.dao;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import com.internousdev.template.dto.BuyItemDTO;
import com.internousdev.template.util.DBConnector;

public class BuyItemDAO {

        public BuyItemDTO getBuyItemInfo() {
                DBConnector dbConnector = new DBConnector();
                Connection connection = dbConnector.getConnection();
                BuyItemDTO buyItemDTO = new BuyItemDTO();

                String sql = "SELECT id, item_name, item_price FROM item_info_transaction";
```

メソッド名は処理内容を分かりやすくするために「getBuyItemInfo」にします。

次へ続きます。

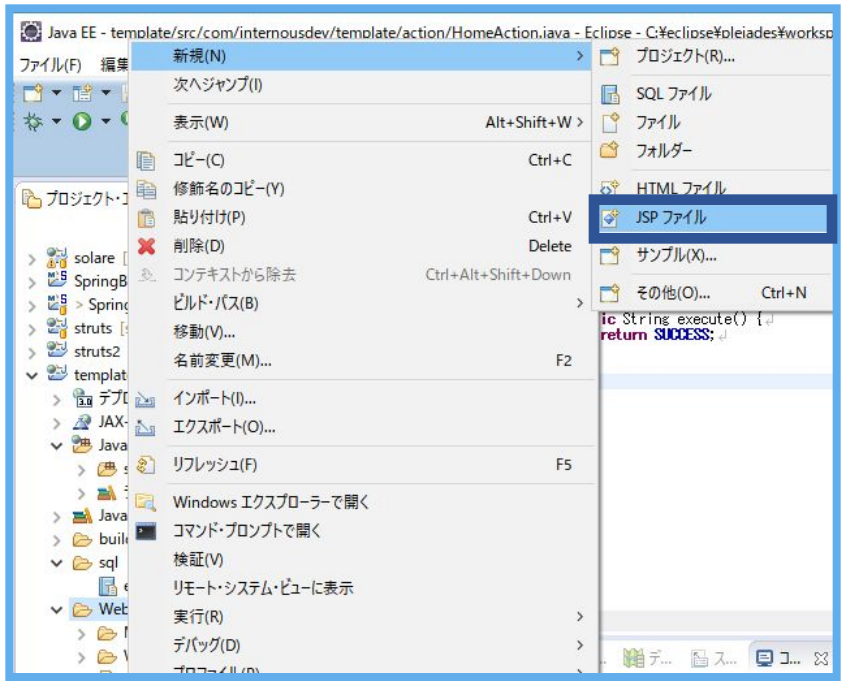
BuyItemDAO.javaの作成

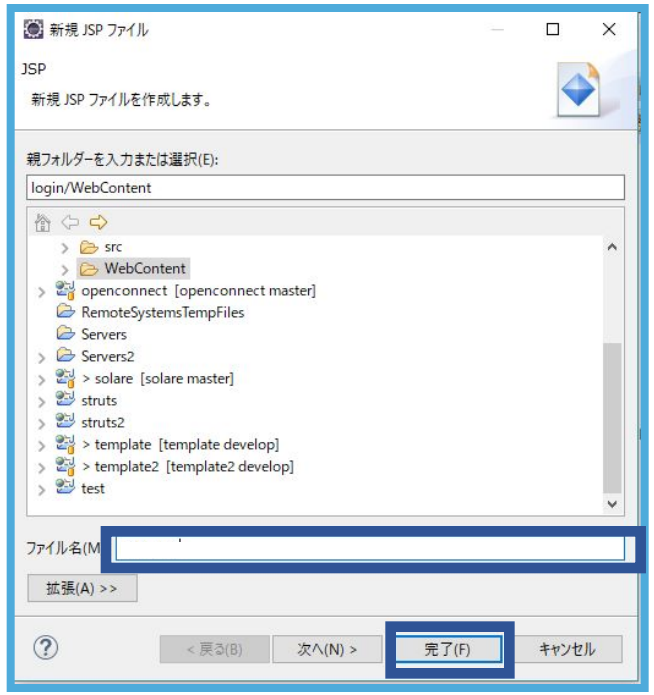
前の続きです。

```
                try {
                        PreparedStatement preparedStatement = connection.prepareStatement(sql);
                        ResultSet resultSet = preparedStatement.executeQuery();

                        if(resultSet.next()) {
                                buyItemDTO.setId(resultSet.getInt("id"));
                                buyItemDTO.setItemName(resultSet.getString("item_name"));
                                buyItemDTO.setItemPrice(resultSet.getString("item_price"));
                        }

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

                return buyItemDTO;
        }

}
```

buyItem.jspの作成

8 buyItem.jsp





① 「プロジェクト」「WebContent」を選択し、「右クリック」「新規」「JSPファイル」を選択します。

② 「名前(M):」欄に「buyItem.jsp」を入力し、完了ボタンをクリックします。

buyItem.jspの作成

③ 以下の内容を写経します。

buyItem.jsp(jspファイル)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
        pageEncoding="UTF-8"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html>
<html>
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <meta http-equiv="Content-Style-Type" content="text/css" />
        <meta http-equiv="Content-Script-Type" content="text/javascript" />
        <meta http-equiv="imagetoolbar" content="no" />
        <meta name="description" content="" />
        <meta name="keywords" content="" />
        <title>buyItem画面</title>

        <style type="text/css">
        /* ========TAG LAYOUT========= */
```

次へ続きます。

buyItem.jspの作成

前の続きです。

```
body {
   margin:0;
   padding:0;
   line-height:1.6;
   letter-spacing:1px;
   font-family:Verdana, Helvetica, sans-serif;
   font-size:12px;
   color:#333;
   background:#fff;
}

table {
   text-align:center;
   margin:0 auto;
}
/* =========ID LAYOUT========= */
#top {
   width:780px;
   margin:30px auto;
```

次へ続きます。

buyItem.jspの作成

前の続きです。

```
    border:1px solid #333;
}

#header {
    width: 100%;
    height: 80px;
    background-color: black;
}

#main {
    width: 100%;
    height: 500px;
    text-align: center;
}
#footer {
    width: 100%;
    height: 80px;
    background-color: black;
    clear:both;
}
```

次へ続きます。

buyItem.jspの作成

前の続きです。

```
        </style>
</head>
<body>
        <div id="header">
                <div id="pr">
                </div>
        </div>
        <div id="main">
                <div id="top">
                        <p>BuyItem</p>
                </div>
                <div>
                <s:form action="BuyItemAction">
                        <table>
                                <tr>
                                        <td>
                                                <span>商品名</span>
                                        </td>
```

次へ続きます。

buyItem.jspの作成

前の続きです。

```
                <td>
                        <s:property value="session.buyItem_name" />
                </td>
        </tr>
        <tr>
                <td>
                        <span>値段</span>
                </td>
                <td>
                <s:property value="session.buyItem_price" />
                        <span>円</span>
                </td>
        </tr>
        <tr>
                <td>
                        <span>在庫</span>
                </td>
                <td>
                        <select name="stock">
```

次へ続きます。

buyItem.jspの作成

前の続きです。

```
                                        <option value="1" selected="selected">1</option>
                                        <option value="2">2</option>
                                        <option value="3">3</option>
                                        <option value="4">4</option>
                                        <option value="5">5</option>
                                    </select>
                                </td>
                            </tr>
                            <tr>
                                <td>
                                    <span>支払い方法</span>
                                </td>
                                <td>
                                    <input type="radio" name="pay" value="1"
checked="checked">現金払い
                                    <input type="radio" name="pay" value="2">クレ
ジットカード
                                </td>
                            </tr>
                            <tr>
```

次へ続きます。

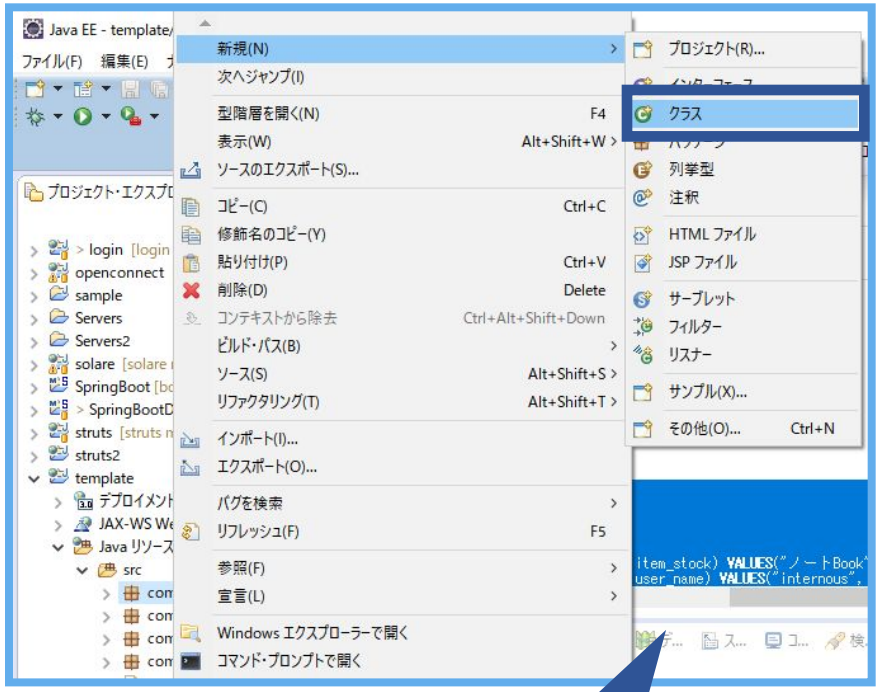
buyItem.jspの作成

前の続きです。

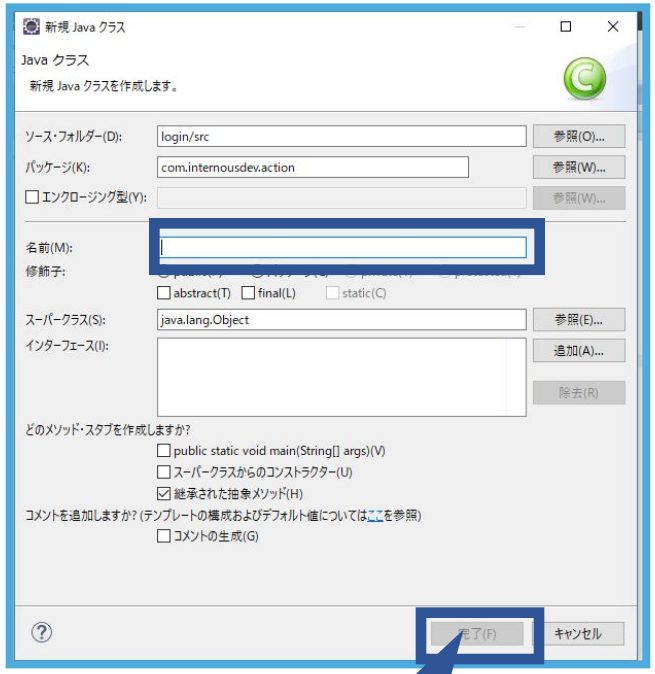
```
                                        <td>
                                                <s:submit value="購入" />
                                        </td>
                                </tr>
                        </table>
                </s:form>
                        <div>
                                <span>前画面に戻る場合は</span>
                                <a href='<s:url action="HomeAction" />'>こちら</a>
                        </div>
                </div>
        </div>
        <div id="footer">
                <div id="pr">
                </div>
        </div>
</body>
</html>
```

LoginAction.javaの作成

9 LoginAction.java



① 「com.internousdev.template.action」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「LoginAction」を入力し、完了ボタンをクリックします。

LoginAction.javaの作成

③ 以下の内容を写経します。

LoginAction (javaファイル)

```
package com.internousdev.template.action;
import java.util.Map;
import org.apache.struts2.interceptor.SessionAware;
import com.internousdev.template.dao.BuyItemDAO;
import com.internousdev.template.dao.LoginDAO;
import com.internousdev.template.dto.BuyItemDTO;
import com.internousdev.template.dto.LoginDTO;
import com.opensymphony.xwork2.ActionSupport;

public class LoginAction extends ActionSupport implements SessionAware {

        private String loginUserId;
        private String loginPassword;
        private String result;
        private Map<String, Object> session;
```

LoginActionの作成

前の続きです。

```
public String execute() {
        LoginDAO loginDAO = new LoginDAO();
        LoginDTO loginDTO = new LoginDTO();
        BuyItemDAO buyItemDAO = new BuyItemDAO();


        result = ERROR;
        loginDTO = loginDAO.getLoginUserInfo(loginUserId, loginPassword);
        session.put("loginUser", loginDTO);

        if(((LoginDTO) session.get("loginUser")).getLoginFlg()) {
                result = SUCCESS;
                BuyItemDTO buyItemDTO = buyItemDAO.getBuyItemInfo();

                session.put("login_user_id",loginDTO.getLoginId());
                session.put("id", buyItemDTO.getId());
                session.put("buyItem_name", buyItemDTO.getItemName());
                session.put("buyItem_price", buyItemDTO.getItemPrice());
```

入力値からユーザー情報の検索を行います。
ログイン認証が成功した場合、次の画面で
「商品情報」が必要なため商品情報を取得します。

次へ続きます。

LoginAction.javaの作成

前の続きです。

```
                return result;

            }
            return result;
    }


    public String getLoginUserId() {
            return loginUserId;
    }

    public void setLoginUserId(String loginUserId) {
            this.loginUserId = loginUserId;
    }

    public String getLoginPassword() {
            return loginPassword;
    }
```

次へ続きます。

LoginAction.javaの作成

前の続きです。

```
    public void setLoginPassword(String loginPassword) {
            this.loginPassword = loginPassword;
    }

    public Map<String, Object> getSession() {
            return session;
    }

    @Override
    public void setSession(Map<String, Object> session) {
            this.session = session;
    }



}
```

補足 この後、struts.xmlに**HomeAction**／**LoginAction**の<action>タグ部分を書き足します。（2時間目10～11ページを参照）

ログイン認証機能の確認

buyItem.jsp／home.jspの画面

正しいID・パスワードを入力した場合

間違ったID・パスワードを入力した場合

templateプロジェクトを実行し、userCreate画面の登録ボタンから上記2パターンの画面が表示できれば成功です。