時計在庫管理システム-EE8,PostgreSQL

office・M
2024年9月21日 18:07

JavaEE8（JSP・Servlet）の環境でWebアプリケーションの開発方法を学ぶ講座をシリーズで提供しています。今回は、『時計在庫検索システム』から発展させ、時計店の在庫状況を管理するシステムを作成します。具体的には検索機能に加えて販売と入荷の機能を追加します。チーム開発での役割分担を想定しています。

2024年9月よりECLIPSEのバージョンを最新版（Version: 2024-06 (4.32.0)）に変更しました。

目次

開発概要

『時計在庫検索システム』から環境は引き続きます。よってJavaEE8アプリケーション・サーバ（Tomcat）とデータベース・サーバ（PostgreSQL）を利用して開発を行うという前提は変わりません。開発ツールには統合開発環境のEclipseを用いることにします。DBの設計も変更ありません。

具体的には時計店の在庫を検索した結果の表示内に［販売］と［入荷］のボタンを付加し在庫数を増減させる処理を実装します。

開発方針

Webアプリケーションの開発環境には図１のようにEclipse2024を用います。

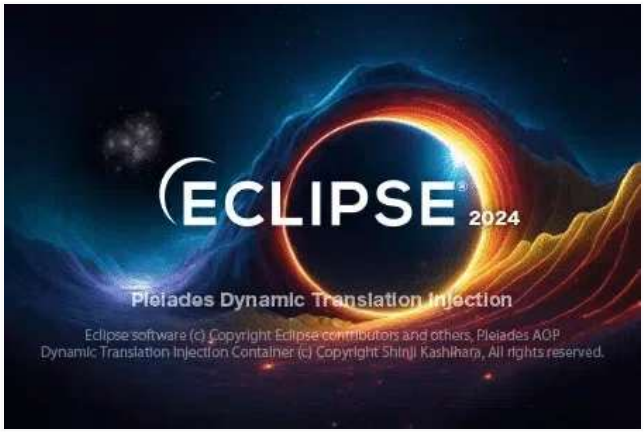

図1．開発環境 Eclipse2024

Spring等のフレームワークは使用しません。これは、MVCモデルにおけるhttpプロトコルの処理の実装を直に学んでいただきたいためです。

開発に必要な最小限の設計ドキュメント提示します。この情報をもとに、自分で試行錯誤しながら実装してみてください。

DB設計

論理設計

『時計在庫検索システム』から変更ありません。

物理設計

『時計在庫検索システム』から変更ありません。

外部設計

『時計在庫検索システム』から変更ありません。
・接続url：**http://localhost:8080/zaiko/Inventory**

『時計在庫検索システム』の基本仕様に加えて、在庫検索結果後の表示を図２のように変更します。
［販売］ボタンで在庫数を１減少、［入荷］ボタンで在庫数を１増加させます。

図2．在庫管理システムの検索結果画面

内部設計

クラスの連携は図３のようになります。黄色部分は検索時のボタンの追加、橙色部分は販売時のボタンの追加、青色部分は入荷時のボタンの追加となります。

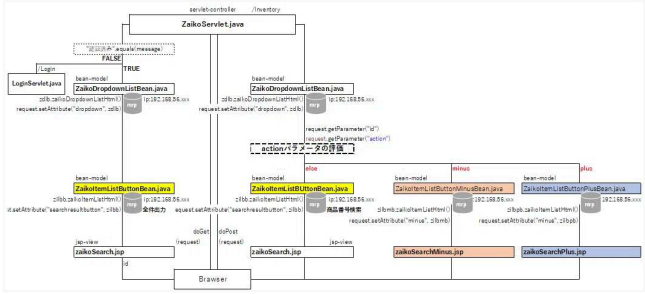

図3．クラス連携図

ZaikoItemListButtonBean.classの仕様とクラス図

（仕様）
・データベースから商品番号をキーに商品情報のリストを作成します。
・コンストラクタの引数で商品番号とPostgreSQLサーバのIPとDB名を指定して実体化します。
・zaikoItemListHtml()メソッドの実行によりDBから商品情報を取得し、昇順の並べかえを行ってテーブル整形されたhtml文を生成します。
・その際、シリーズごとの出力や全件出力にも対応させます。
・さらに、時計のイメージ画像の出力にも対応させます。
（時計の画像はimages-watchフォルダ内から読み込みます）
・**formタグで販売ボタンと入荷ボタンを表示します。**
・生成されたhtml文はresultHtmlフィールドに格納されます。
・利用側のクラスはgetResultHtml()メソッドで商品リスト（html文）を取得します。

（クラス図）

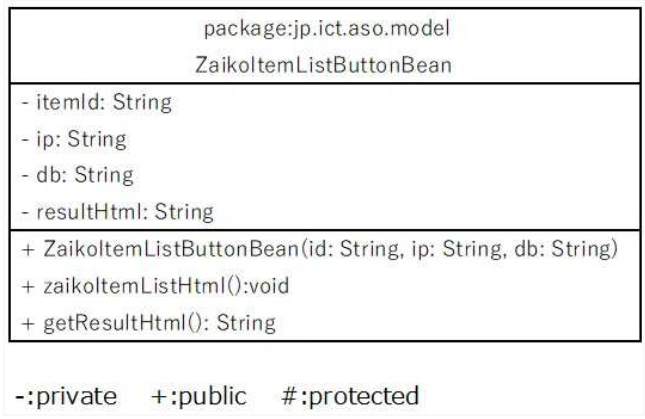

図4．ZaikoItemListButtonBeanクラス図

ZaikoItemListButtonMinusBean.classの仕様とクラス図

（仕様）
・データベースから商品番号をキーに商品情報のリストを作成します。
・コンストラクタの引数で商品番号とPostgreSQLサーバのIPとDB名を指定して実体化します。
・zaikoItemListHtml()メソッドの実行によりDBから商品情報を取得し**在庫数を１減少**させ、結果をテーブル整形されたhtml文として生成します。
・さらに、時計のイメージ画像の出力にも対応させます。
（時計の画像はimages-watchフォルダ内から読み込みます）
・**formタグで販売ボタンを表示します。**
・生成されたhtml文はresultHtmlフィールドに格納されます。
・利用側のクラスはgetResultHtml()メソッドで商品リスト（html文）を取得します。

（クラス図）

図5．ZaikoItemListButtonMinusBeanクラス図

ZaikoItemListButtonPlusBean.classの仕様とクラス図

（仕様）
・データベースから商品番号をキーに商品情報のリストを作成します。
・コンストラクタの引数で商品番号とPostgreSQLサーバのIPとDB名を指定して実体化します。
・zaikoItemListHtml()メソッドの実行によりDBから商品情報を取得し**在庫数を１増加**させ、結果をテーブル整形されたhtml文として生成します。
・さらに、時計のイメージ画像の出力にも対応させます。
（時計の画像はimages-watchフォルダ内から読み込みます）
・**formタグで入荷ボタンを表示します。**
・生成されたhtml文はresultHtmlフィールドに格納されます。
・利用側のクラスはgetResultHtml()メソッドで商品リスト（html文）を取得します。

（クラス図）

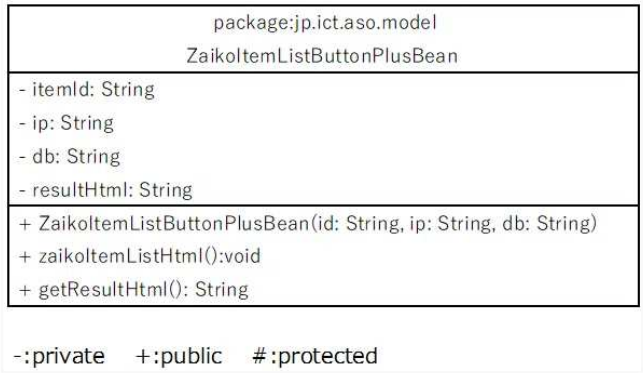

図6．ZaikoItemListButtonPlusBeanクラス図

実装準備

PostgreSQLのjdbcドライバの登録

『時計在庫検索システム』から変更ありません。

ハッシュ算出用ライブラリの登録

『時計在庫検索システム』から変更ありません。

実装手順１（ログイン処理）

『時計在庫検索システム』から変更ありません。

実行確認１（ログイン処理）

『時計在庫検索システム』から変更ありません。

実装手順２（在庫検索処理）

１．DAO用JavaBeansクラスを作成します

Eclipseパッケージ・エクスプローラより
zaikoプロジェクトを右クリック→新規→クラス

→以下の内容で作成する

・ZaikoItemListButtonBean.java
　パッケージ：jp.ict.aso.model
　クラス名：ZaikoItemListButtonBean
　ソースコード：以下のヒントと図４を参考に考えましょう！

【ヒント】

```
public void zaikoItemListHtml(){
    String id;
    String name;
    int zaiko;
    String location;
try{
//[1]JDBCドライバのロード
  Class.forName("org.postgresql.Driver");
//「2」RDBへの接続
  String url="jdbc:postgresql://"+ip+":5432/"+db;
  Connection connection =
          DriverManager.getConnection(url,"postgres","postgres");
//[3]SQL文のコンテナ作成
  Statement stmt = connection.createStatement();
//[4]SQL文を実行する
//executeUpdateは処理結果の行数が返ってきます
//更新後の在庫情報を取得
  ResultSet rs = stmt.executeQuery
    ("select ITEM_ID,ITEM_NAME,ZAIKO_NUM,LOCATION from T_ZAIKO where ITEM_ID like '"
                        + itemId + "' order by ITEM_ID");
//[5]RDBの検索結果を取り出す
  resultHtml=resultHtml+"<table border=1><tr>";
  resultHtml=resultHtml+"<tr><th>No.</th><th>商品ID</th><th>商品イメージ</th>"
                          + "<th>商品名</th>";
  resultHtml=resultHtml+"<th>在庫数</th><th>保管棚</th></tr>";
  int i=0;
  while( rs.next() ){
   i++;
   id = rs.getString("ITEM_ID");
   name = rs.getString("ITEM_NAME" );
   zaiko = rs.getInt( "ZAIKO_NUM" );
   location = rs.getString( "LOCATION" );
   resultHtml=resultHtml+"<tr><td><center>"+i+".</center><br>";
   resultHtml=resultHtml+"<form method=\"POST\" action=\"Inventory?action=minus\">"
                       + "<input type=\"hidden\" name=\"id\" value=\""+id+"\">";
   resultHtml=resultHtml+"<input type=\"submit\" value=\"　販　売　\"></form>";
   resultHtml=resultHtml+"<form method=\"POST\" action=\"Inventory?action=plus\">"
                       + "<input type=\"hidden\" name=\"id\" value=\""+id+"\">";
   resultHtml=resultHtml+"<input type=\"submit\" value=\"　入　荷　\"></form></td>";
   resultHtml=resultHtml+"<td>"+id+"</td>";
   resultHtml=resultHtml+"<td><img src='watch-images/"+id+".png' width=100></td>";
   resultHtml=resultHtml+"<td>"+name+"</td>";
   resultHtml=resultHtml+"<td>"+zaiko+"</td>";
```

```
    resultHtml=resultHtml+"<td>"+location+"</td></tr>";
   }
   resultHtml=resultHtml+"</table>";

//[6]データベースから切断する
   stmt.close();
   connection.close();

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

Eclipseパッケージ・エクスプローラより
zaikoプロジェクトを右クリック→新規→クラス
→以下の内容で作成する

・ZaikoItemListButtonMinusBean.java
　パッケージ：jp.ict.aso.model
　クラス名：ZaikoItemListButtonMinusBean
　ソースコード：以下のヒントと図５を参考に考えましょう！

```
【ヒント】
public void zaikoItemListHtml(){
  String id;
  String name;
  int zaiko;
  String location;
  try{
//[1]JDBCドライバのロード
    Class.forName("org.postgresql.Driver");
//「2」RDBへの接続
    String url="jdbc:postgresql://"+ip+":5432/"+db;
    Connection connection =
    DriverManager.getConnection(url,"postgres","postgres");
//[3]SQL文のコンテナ作成
    Statement stmt = connection.createStatement();
//[4]SQL文を実行する
//executeUpdateは処理結果の行数が返ってきます
    int zaikoNum = stmt.executeUpdate
    ("update T_ZAIKO set ZAIKO_NUM = ZAIKO_NUM-1 where ITEM_ID = '" + itemId + "'");
//更新後の在庫情報を取得
    ResultSet rs = stmt.executeQuery
    ("select ITEM_ID,ITEM_NAME,ZAIKO_NUM,LOCATION from T_ZAIKO where ITEM_ID like '"
```

```
            + itemId + "' order by ITEM_ID");
        //[5]RDBの検索結果を取り出す
            resultHtml=resultHtml+"<table border=1><tr>";
            resultHtml=resultHtml+"<tr><th>No.</th><th>商品ID</th><th>商品イメージ</th>"
            + "<th>商品名</th>";
            resultHtml=resultHtml+"<th>在庫数</th><th>保管棚</th></tr>";
            int i=0;
            while( rs.next() ){
                i++;
                id = rs.getString("ITEM_ID");
                name = rs.getString("ITEM_NAME" );
                zaiko = rs.getInt( "ZAIKO_NUM" );
                location = rs.getString( "LOCATION" );
                resultHtml=resultHtml+"<form method="POST" action="Inventory?action=minus">";
                resultHtml=resultHtml+"<tr><td><center>"+i+".</center><br>"
                + "<input type="hidden" name="id" value=""+id+"">";
                resultHtml=resultHtml+"<input type="submit" value="  販　売  "></td>";
                resultHtml=resultHtml+"<td>"+id+"</td>";
                resultHtml=resultHtml+"<td><img src='watch-images/"+id+".png' width=100></td>";
                resultHtml=resultHtml+"<td>"+name+"</td>";
                resultHtml=resultHtml+"<td>"+zaiko+"</td>";
                resultHtml=resultHtml+"<td>"+location+"</td></tr></form>";
            }
            resultHtml=resultHtml+"</table>";
        //[6]データベースから切断する
            stmt.close();
            connection.close();
          }catch ( Exception e) {
            e.printStackTrace();
          }
        }
```

Eclipseパッケージ・エクスプローラより
zaikoプロジェクトを右クリック→新規→クラス
→以下の内容で作成する

・ZaikoItemListButtonPlusBean.java
　パッケージ：jp.ict.aso.model
　クラス名：ZaikoItemListButtonPlusBean
　ソースコード：ZaikoItemListButtonMinusBeanのヒントと図６を参考に
　　考えましょう！

２．リクエストコントロール用Servletクラスを変更します

Eclipseパッケージ・エクスプローラより
zaikoプロジェクトを右クリック→新規→その他→Web→サーブレット
→以下の内容で作成する

・ZaikoServlet.java
　パッケージ：jp.ict.aso.controller
　クラス名：ZaikoServlet
　ソースコード：以下のヒントを参考に考えましょう
　　（見えないところは補完しましょう）
※アノテーションは**/Inventory**

【ヒント】

```
protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
//httpセッションが認証済みか確認
     HttpSession session=request.getSession();
     String message=(String)session.getAttribute("ok");
     if("認証済み".equals(message)) {
        // ドロップダウンリストの実体化（IP,DB名の初期値をプロパティに設定）
        ZaikoDropdownListBean zdlb = new ZaikoDropdownListBean("192.168.56.200","mrp");
        // ドロップダウンリストを作成
        zdlb.zaikoDropdownListHtml();
        // リクエストスコープに保存（スコープ名：dropdown）
        request.setAttribute("dropdown", zdlb);

        // 更新ボタン付き検索結果リストの実体化（商品番号の初期値、IP、DB名をプロパティに設定）
        ZaikoItemListButtonBean zilbb = new ZaikoItemListButtonBean("192.168.56.200","mrp");
        zilbb.setItemId("%");
        // 在庫リストを作成
        zilbb.zaikoItemListHtml();
        // リクエストスコープに保存（スコープ名：searchresultbutton）
        request.setAttribute("searchresultbutton", zilbb);

        // 入出力用JSPにフォワード
        RequestDispatcher dispatcher =
            request.getRequestDispatcher
                 ("/WEB-INF/jsp/zaikoSearch.jsp");
        dispatcher.forward(request, response);
     }else {
        // 認証されていない場合はアカウント入力に戻す
        RequestDispatcher dispatcher =
            request.getRequestDispatcher
                 ("Login");
        dispatcher.forward(request, response);
     }
    }
    protected void doPost(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        // ドロップダウンリストの実体化（IP,DB名の初期値をプロパティに設定）
        ZaikoDropdownListBean zdlb = new ZaikoDropdownListBean("192.168.56.200","mrp");
```

```
// ドロップダウンリストを作成
zdlb.zaikoDropdownListHtml();
// リクエストスコープに保存（スコープ名：dropdown）
request.setAttribute("dropdown", zdlb);

// リクエストパラメータ（入力値）の取得（パラメータ名：id）
request.setCharacterEncoding("UTF-8");
String id = request.getParameter("id");

// サーブレットクラスの動作を決定する「action」の値をリクエストパラメータから取得
String action = request.getParameter("action");
if(action.equals("minus")){    //actionがマイナスのとき

    // 更新ボタン付き検索結果リストの実体化（商品番号の初期値、IP、DB名をプロパティに設定）
    ZaikoItemListButtonMinusBean zilbmb = new ZaikoItemListButtonMinusBean("192.168.
    zilbmb.setItemId(id);
    // 在庫リストを作成
    zilbmb.zaikoItemListHtml();
    // リクエストスコープに保存（スコープ名：minus）
    request.setAttribute("minus", zilbmb);

    // 入出力用JSPにフォワード
    RequestDispatcher dispatcher =
        request.getRequestDispatcher
            ("/WEB-INF/jsp/zaikoSearchMinus.jsp");
    dispatcher.forward(request, response);

}else if(action.equals("plus")){
    // 更新ボタン付き検索結果リストの実体化（商品番号の初期値、IP、DB名をプロパティに設定）
    ZaikoItemListButtonPlusBean zilbpb = new ZaikoItemListButtonPlusBean("192.168.56
    zilbpb.setItemId(id);
    // 在庫リストを作成
    zilbpb.zaikoItemListHtml();
    // リクエストスコープに保存（スコープ名：plus）
    request.setAttribute("plus", zilbpb);

    // 入出力用JSPにフォワード
    RequestDispatcher dispatcher =
        request.getRequestDispatcher
            ("/WEB-INF/jsp/zaikoSearchPlus.jsp");
    dispatcher.forward(request, response);

}else {
    // 更新ボタン付き検索結果リストの実体化（商品番号の初期値、IP、DB名をプロパティに設定）
    ZaikoItemListButtonBean zilbb = new ZaikoItemListButtonBean("192.168.56.200","mr
    zilbb.setItemId(id);
    // 在庫リストを作成
    zilbb.zaikoItemListHtml();
    // リクエストスコープに保存（スコープ名：searchresultbutton）
    request.setAttribute("searchresultbutton", zilbb);

    // 入出力用JSPにフォワード
    RequestDispatcher dispatcher =
        request.getRequestDispatcher
            ("/WEB-INF/jsp/zaikoSearch.jsp");
    dispatcher.forward(request, response);
}
```

```
    }
```

３．入力・検索結果リスト出力用画面のJSPファイルを作成します

Eclipseパッケージ・エクスプローラより
zaikoプロジェクトを右クリック→新規→その他→Web→JSPファイル
→以下の内容で作成する

・zaikoSearchMinus.jsp
　保存場所：zaiko/src/main/webapp/**WEB-INF/jsp ← 注意！**
　ファイル名：zaikoSearchMinus.jsp
　ソースコード：以下のヒントを参考に考えましょう！
　　（見えないところは補完しましょう）

【ヒント】

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="jp.ict.aso.model.*,java.util.*"%>
<%
ZaikoDropdownListBean zdlb=(ZaikoDropdownListBean)request.getAttribute("dropdown");
ZaikoItemListButtonMinusBean zilbmb=(ZaikoItemListButtonMinusBean)request.getAttribute("minu
String message=(String)session.getAttribute("ok");
if(!"認証済み".equals(message)){
    response.sendRedirect("Login"); // ログインページにリダイレクト
}

// ログアウトボタンが押されたらセッションを切ってログインへ
if (request.getParameter("logout") != null) {
    message="";
    session.invalidate();
    response.sendRedirect("Login"); // ログインページにリダイレクト
}
%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>商品在庫取得</title>
    <link rel="stylesheet" type="text/css" href="zaiko.css">
</head>
<body><center>
    <!-- ヘッダー -->
    <div class="section-header">
```

```
            <div class="left-content">
                <img src="watch-images/lady.png" width="80" alt="Logo Image">
                <div class="text-content">
                    <h1>在庫管理システム</h1>
                    <p><font face="Verdana" size="5"><b>Inventory Search</b></font></p>
                </div>
            </div>
        </div>
        <!-- メインコンテンツ -->
        <section>
             <hr width="360">
            <table border="1">
                <tr>
                    <form action="Inventory?action=view" method="post">
                        <td>商品ID:
                            <select name="id">
                                <%= zdlb.getResultHtml() %>
                            </select>
                        </td>
                        <td><input type="submit" value="   在 庫 状&nbs
                    </form>
                    <td><%= message %></td>
                    <td>
                        <form method="post">
                            <input type="submit" name="logout" value="ログアウト">
                        </form>
                    </td>
                </tr>
            </table>
            <%= zilbmb.getResultHtml() %>
        </section>
        <!-- フッター -->
        <hr width="360">
        <footer>
            <p>&copy; 2024 在庫管理システム. All Rights Reserved.</p>
            <p>お問い合わせ: info@slowlife.halfmoon.jp</p>
        </footer>
    </body>
    </html>
```

Eclipseパッケージ・エクスプローラより
zaikoプロジェクトを右クリック→新規→その他→Web→JSPファイル
→以下の内容で作成する

・zaikoSearchPlus.jsp
　保存場所：zaiko/src/main/webapp/**WEB-INF/jsp** ← **注意！**

ファイル名：zaikoSearchPlus.jsp
ソースコード：zaikoSearchMinus.jspのヒントを参考に考えましょう！

実行確認２（在庫検索処理）

・ログインシステム（LoginServlet.java）を実行します。正常に認証されたとき在庫検索画面が表示されるか確認します。
・販売ボタン、入荷ボタンで在庫数が増減するか確認します。

図7．時計在庫管理システム