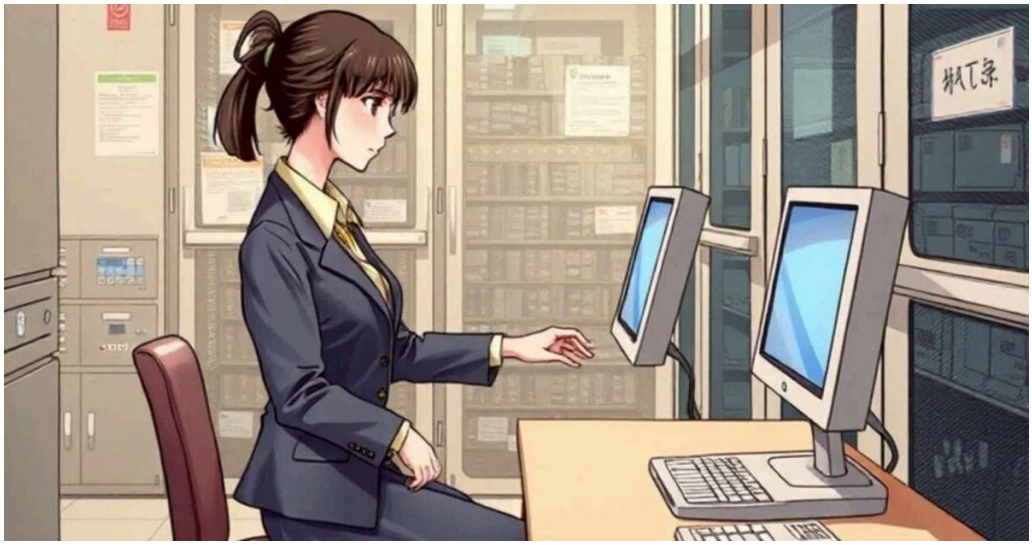


Swingによるデスクトップアプリケーション開発（商品在庫検索）-JavaSE1.8

office・M
2024年11月9日 12:16

¥1,000 •••

Java8のSwing環境でデスクトップ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WindowBuilder（Swingデザイナー）を使ってPostgreSQLデータベースに接続して商品の在庫を検索するシステムを作成します。

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

外部設計

（仕様）
・初期起動時（図１参照）にクリック可能なボタンは「ログイン」と「リセット」だけです。
・サーバのIPアドレス、データベース名はデフォルトでの設定値を表示します。
・リセットボタンのクリックで利用者電話番号とパスワードの入力領域がクリアされます。
・認証用に登録済みの利用者電話番号とパスワードを入力しログインボタンのクリックで認証が成功すると、「在庫状況取得」と「ログアウト」のボタンが使用可能になり商品在庫の検索が可能になります（図２参照）。
・この時、「ログイン」と「リセット」ボタンは使用不可となります。
・認証できないときは「認証不可」のメッセージを赤色で表示します。
・「ログアウト」ボタンをクリックすることでデータベースの接続を断ち「在庫状況取得」と「ログアウト」のボタンを使用不可にし、「利用者電話番号」「パスワード」「検索結果」の各領域をクリアして、「ログイン」と「リセット」のボタンを使用可能にします。

図1．初期起動画面

図2．認証完了後在庫検索画面

内部設計

DB設計

「時計在庫検索システム-EE8,PostgreSQL」で定義されたDB設計を用います。当該Webページの認証方式、論理設計、物理設計、を参照して実装してください。
また、認証用のT_USERテーブルも同様に当該Webページを参照して実装してください。

処理ロジック

DBに接続して在庫検索を行う業務ロジック（**DataAccessObject**）を実装します。詳細は「時計在庫検索システム-EE8,PostgreSQL」で作成したクラスファイルの仕様と実装を参照してください。作成するクラスファイル（Bean）のクラス図は以下のようになります。

LoginCheckBean.classの仕様とクラス図

（仕様）
・データベースからTEL番号をキーにハッシュ化されたパスワードを取得します（ハッシュ化されたパスワードは事前にDBに登録されています）。
・コンストラクタの引数でキーとなる電話番号とPostgreSQLサーバのIPとDB名を指定して実体化します。
・コンストラクタの実行により対象となる電話番号のパスワードが取得されます。電話番号が登録されていないときにはパスワードは空となります。
・利用側のクラスはgetPassword()メソッドでパスワードを取得します。

（クラス図）

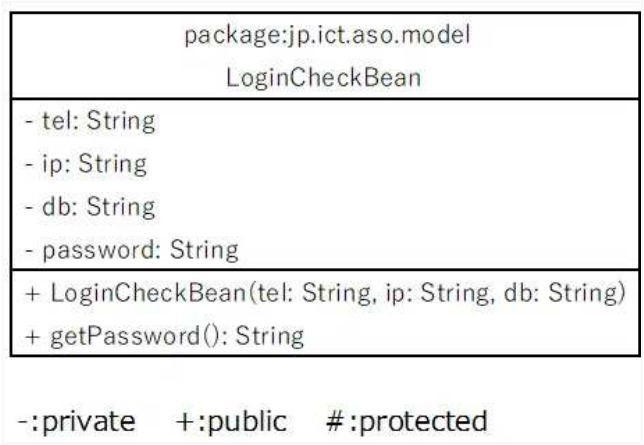


図3．LoginCheckBeanクラス図

ZaikoItemListBean.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）
・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）

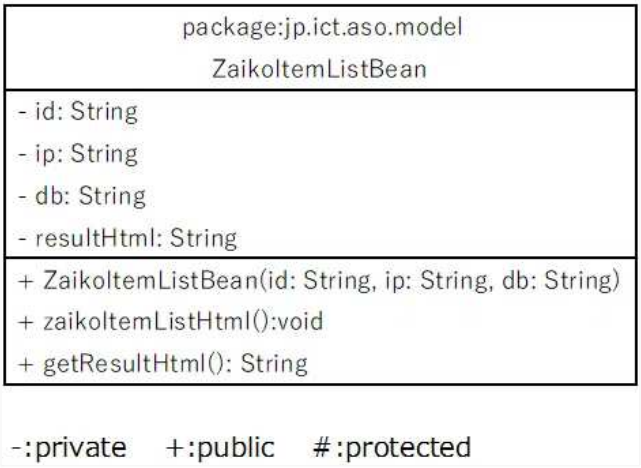


図4．ZaikoItemListBeanクラス図

実装準備

プロジェクトの作成

Eclipseのメニューバーより
ファイル→新規→Javaプロジェクト→「SwingZaiko」プロジェクトを作成
→図５の内容で設定する

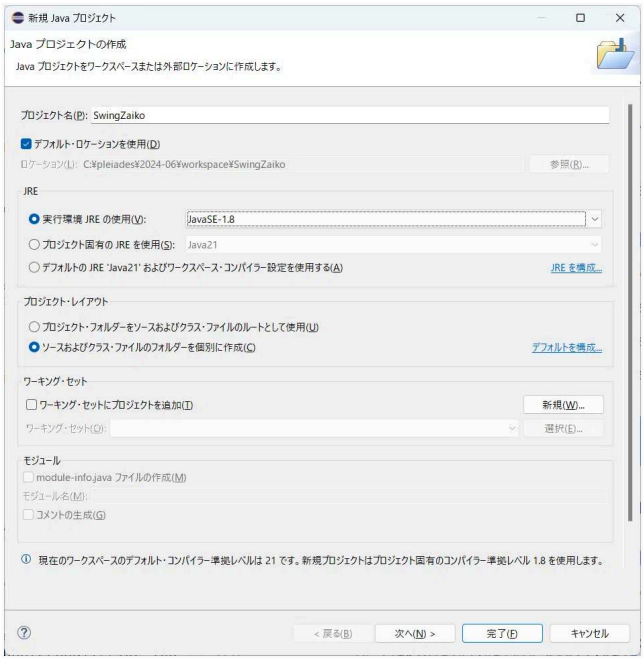


図5．SwingZaikoプロジェクト設定

※作成済みであればこの処理は必要ありません。

以下画面のスクリーンショットはライトテーマで取得します。
（ライトテーマの設定方法）
Eclipseのメニューバーより
　ウィンドウ → 設定 → 一般 → 外観 → ルック＆フィール → ライト
　→ 適用して閉じる → Eclipseの再起動がかかります

外部ファイルの利用準備

(1)jarファイルの準備をします
以下Windows環境を想定しています。
事前に「**提供Library**」フォルダを作成しておきます。
（例　c:￥提供Library）

図６のように提供Libraryフォルダ内に以下のの**２つのjarファイル**を保存しておきます。
・**ハッシュライブラリ：commons-codec-1.15.jar**
・**jdbcドライバ：org.postgresql.driver.jar**

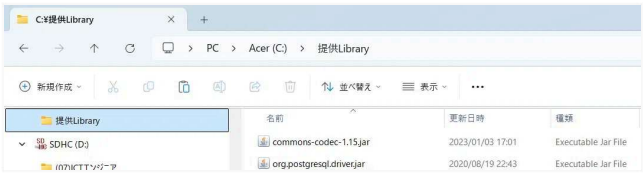


図6．Hashライブラリとjdbcドライバの保存

ビルド・パスに追加

(1)jarファイルを登録します
２つのjarファイルをビルド・パスに追加します。

Eclipseパッケージ・エクスプローラより
SwingZaikoプロジェクトを右クリック→ビルド・パス
→ビルド・パスの構成→「ライブラリ」タブ→外部JARの追加
→Ctrlキーを押しながら**commons-codec-1.15.jar**と**org.postgresql.driver.jar**をクリック→「開く」をクリックします
→最後に「適用して閉じる」をクリック

※**Eclipse2024の場合** → SwingZaikoプロジェクトを右クリック→プロパティ
　→Javaのビルド・パスで設定します。

※図７のように一度設定されていれば再度設定する必要はありません。

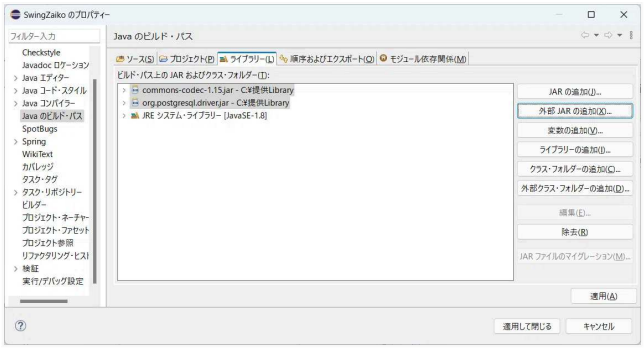


図7．外部JARファイルをビルド・パスに追加

実装

ひな形の作成

WindowBuilderを用いてSwingアプリケーションのスケルトン（骨格）を自動生成させます。

Eclipseパッケージ・エクスプローラより
SwingZaikoプロジェクトを右クリック→新規→その他
→ WindowBuilder → Swingデザイナー → JFrameを選択 → 次へ

以下の内容で作成
　　パッケージ：jp.ict.aso.swing
　　名前：Inventory

GUI実装

自動生成されたプログラム（スケルトン）からGUIのデザインを実装します。

パレットと構造（コンポーネント、プロパティ）のViewを利用するのがコツです。デザインイメージは設定反映の参考としてとらえた方が良いでしょう。

とりあえず図８のようにGUIを実装します。

図8．GUI（部品）の実装

イベント実装

ソースを変更し仕様を実装します。

以下の２つのクラスについては、「時計在庫検索システム-EE8,PostgreSQL」で作成したクラスを使用します。
ただしJavaのバージョンが違うため、今回のプロジェクト内で**リコンパイル**します。
また、画像を表示するクラスについてはイメージリソースの指定方法がWebアプリとデスクトップアプリでは違うため、一部ソースを変更します。

・LoginCheckBean.java：変更なしでリコンパイル
・ZaikoItemListBean.java：画像指定方法を変更してリコンパイル
（変更内容）

```
resultHtml = resultHtml+"<td><img src='"+ getClass().getResource("/watch-images/"
          + id + ".png").toString()+ "' width=100></td>";
```

※あわせてテーブルのセル幅も広めに変えましょう（`width=100` 程度で大丈夫です）

商品画像ファイルの配置

srcフォルダの直下に商品画像ファイルが入ったwatch-imagesのフォルダを置きます。画像のファイル名は商品IDとなります。

ここまでのファイル・ディレクトリ構成（図９）を確認してください。

図9．ファイル・ディレクトリ構成

実行確認

不具合などを解消して完成です。GUIの調整も行います。
ちなみに、商品の選択はシリーズ単位で大丈夫です。コンボボックス内にAシリーズ、Bシリーズ、Cシリーズ、Dシリーズの区分で設定しましょう。

図10．商品在庫管理システム在庫検索画面

単独起動

実行可能JARファイル

①せっかくですので、単独で起動できるアプリケーションにエクスポートしましょう。Java1.8以上のJREの環境がWindowsのPCにインストールされていればダブルクリックで起動できます。

Eclipseパッケージ・エクスプローラより
SwingZaikoプロジェクトを右クリック → エクスポート
→ Java → 実行可能JARファイル → 次へ
→ 以下のように設定する → 完了

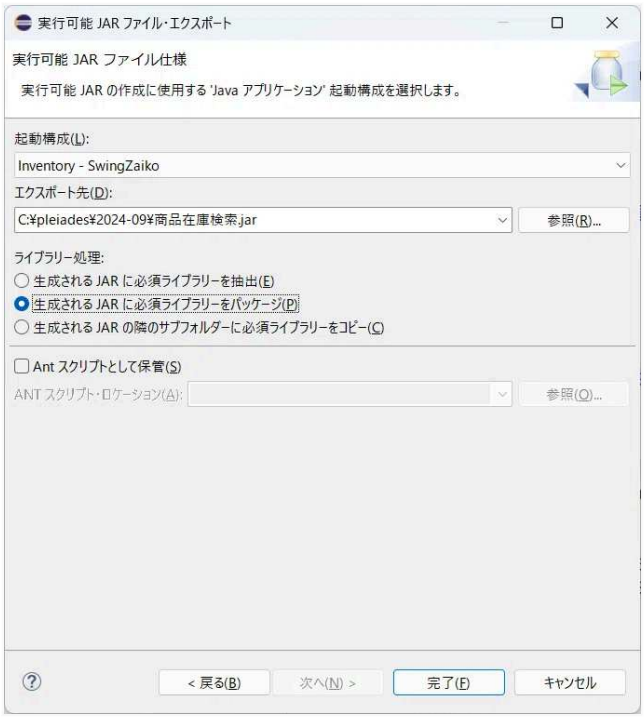


図11．実行可能JARファイルの設定

以下のような警告が出る場合がありますが、気にしません。

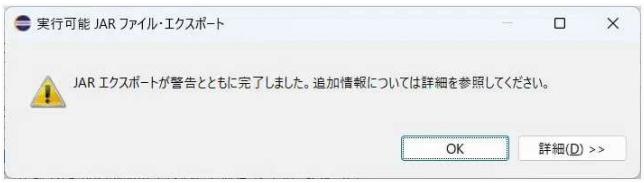


図12．警告ダイアログ

作成されたjarファイルをダブルクリックするとアプリが起動します。

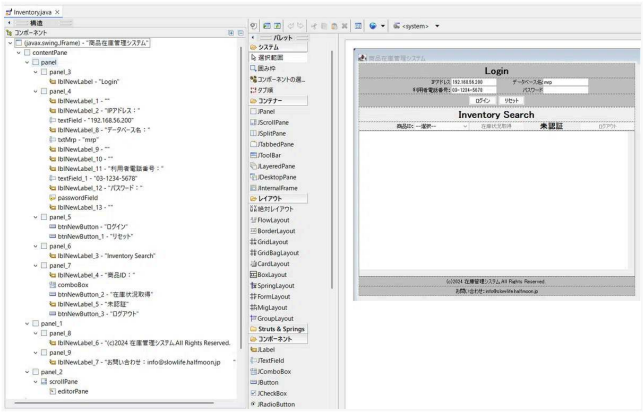
図13．実行可能JARファイル

最後に

今回は、GUIデザイン部分の設定の詳細は提示していません。また、実装変更部分のソースコード例も載せていません。自由に工夫してみてください！

ヒント

GUIコンポーネント設計



Inventory.java

```
package jp.ict.aso.swing;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
```

```java
import javax.swing.AbstractAction;
import javax.swing.Action;
import javax.swing.BoxLayout;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JEditorPane;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JScrollPane;
import javax.swing.JTextField;
import javax.swing.ScrollPaneConstants;
import javax.swing.SwingConstants;
import javax.swing.border.EmptyBorder;

import org.apache.commons.codec.digest.DigestUtils;

import jp.ict.aso.model.LoginCheckBean;
import jp.ict.aso.model.ZaikoItemListBean;

public class Inventory extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField textField;
    private JTextField txtMrp;
    private JTextField textField_1;
    private JPasswordField passwordField;
    private final Action action = new SwingAction();
    private final Action action_1 = new SwingAction_1();
    private final Action action_2 = new SwingAction_2();
    private final Action action_3 = new SwingAction_3();

    JLabel lblNewLabel_5;   //koko認証済みラベル
    JEditorPane editorPane; //koko
    JButton btnNewButton;   //koko
    JButton btnNewButton_1; //koko
    JButton btnNewButton_2; //koko
    JButton btnNewButton_3; //koko
    JComboBox comboBox;     //koko

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Inventory frame = new Inventory();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
```

```java
/**
 * Create the frame.
 */
public Inventory() {
    setTitle("商品在庫管理システム");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 680, 600);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(contentPane);
    contentPane.setLayout(new BorderLayout(0, 0));

    JPanel panel = new JPanel();
    contentPane.add(panel, BorderLayout.NORTH);
    panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));

    JPanel panel_3 = new JPanel();
    panel_3.setBackground(new Color(192, 192, 192));
    panel.add(panel_3);

    JLabel lblNewLabel = new JLabel("Login");
    lblNewLabel.setFont(new Font("MS UI Gothic", Font.BOLD, 24));
    panel_3.add(lblNewLabel);

    JPanel panel_4 = new JPanel();
    panel_4.setBackground(new Color(192, 192, 192));
    panel.add(panel_4);
    panel_4.setLayout(new GridLayout(0, 6, 0, 0));

    JLabel lblNewLabel_1 = new JLabel("");
    lblNewLabel_1.setHorizontalAlignment(SwingConstants.RIGHT);
    lblNewLabel_1.setFont(new Font("MS UI Gothic", Font.PLAIN, 12));
    panel_4.add(lblNewLabel_1);

    JLabel lblNewLabel_2 = new JLabel("IPアドレス：");
    lblNewLabel_2.setHorizontalAlignment(SwingConstants.RIGHT);
    panel_4.add(lblNewLabel_2);

    textField = new JTextField();
    textField.setText("192.168.56.200");
    panel_4.add(textField);
    textField.setColumns(10);

    JLabel lblNewLabel_8 = new JLabel("データベース名：");
    lblNewLabel_8.setHorizontalAlignment(SwingConstants.RIGHT);
    panel_4.add(lblNewLabel_8);

    txtMrp = new JTextField();
    txtMrp.setText("mrp");
    panel_4.add(txtMrp);
    txtMrp.setColumns(10);

    JLabel lblNewLabel_9 = new JLabel("");
    panel_4.add(lblNewLabel_9);

    JLabel lblNewLabel_10 = new JLabel("");
```

```
panel_4.add(lblNewLabel_10);

JLabel lblNewLabel_11 = new JLabel("利用者電話番号：");
lblNewLabel_11.setHorizontalAlignment(SwingConstants.RIGHT);
panel_4.add(lblNewLabel_11);

textField_1 = new JTextField();
textField_1.setText("03-1234-5678");
panel_4.add(textField_1);
textField_1.setColumns(10);

JLabel lblNewLabel_12 = new JLabel("パスワード：");
lblNewLabel_12.setHorizontalAlignment(SwingConstants.RIGHT);
panel_4.add(lblNewLabel_12);

passwordField = new JPasswordField();
panel_4.add(passwordField);

JLabel lblNewLabel_13 = new JLabel("");
panel_4.add(lblNewLabel_13);

JPanel panel_5 = new JPanel();
panel_5.setBackground(new Color(192, 192, 192));
panel.add(panel_5);

btnNewButton = new JButton("ログイン");  //koko
btnNewButton.setAction(action);
panel_5.add(btnNewButton);

btnNewButton_1 = new JButton("リセット");    //koko
btnNewButton_1.setAction(action_1);
panel_5.add(btnNewButton_1);

JPanel panel_6 = new JPanel();
panel.add(panel_6);

JLabel lblNewLabel_3 = new JLabel("Inventory Search");
lblNewLabel_3.setFont(new Font("MS UI Gothic", Font.BOLD, 24));
panel_6.add(lblNewLabel_3);

JPanel panel_7 = new JPanel();
panel.add(panel_7);
panel_7.setLayout(new GridLayout(0, 5, 0, 0));

JLabel lblNewLabel_4 = new JLabel("商品ID：");
lblNewLabel_4.setHorizontalAlignment(SwingConstants.RIGHT);
panel_7.add(lblNewLabel_4);

comboBox = new JComboBox(); //koko
comboBox.setModel(new DefaultComboBoxModel(new String[]
    {"--選択--", "Aシリーズ", "Bシリーズ", "Cシリーズ", "Dシリーズ"}));
panel_7.add(comboBox);

btnNewButton_2 = new JButton("在庫状況取得");    //koko
btnNewButton_2.setAction(action_2);
panel_7.add(btnNewButton_2);

lblNewLabel_5 = new JLabel("未認証");    //koko
```

```
        lblNewLabel_5.setFont(new Font("MS UI Gothic", Font.BOLD, 18));
        lblNewLabel_5.setHorizontalAlignment(SwingConstants.CENTER);
        panel_7.add(lblNewLabel_5);

        btnNewButton_3 = new JButton("ログアウト");  //koko
        btnNewButton_3.setAction(action_3);
        panel_7.add(btnNewButton_3);

        JPanel panel_1 = new JPanel();
        contentPane.add(panel_1, BorderLayout.SOUTH);
        panel_1.setLayout(new BoxLayout(panel_1, BoxLayout.Y_AXIS));

        JPanel panel_8 = new JPanel();
        panel_8.setBackground(new Color(192, 192, 192));
        panel_1.add(panel_8);

        JLabel lblNewLabel_6 = new
                JLabel("(c)2024 在庫管理システム.All Rights Reserved.      ");
        panel_8.add(lblNewLabel_6);

        JPanel panel_9 = new JPanel();
        panel_9.setBackground(new Color(192, 192, 192));
        panel_1.add(panel_9);

        JLabel lblNewLabel_7 = new
                JLabel("お問い合わせ：info@slowlife.halfmoon.jp    ");
        panel_9.add(lblNewLabel_7);

        JPanel panel_2 = new JPanel();
        contentPane.add(panel_2, BorderLayout.CENTER);
        panel_2.setLayout(new BoxLayout(panel_2, BoxLayout.X_AXIS));

        JScrollPane scrollPane = new JScrollPane();
        scrollPane.setVerticalScrollBarPolicy
                (ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS);
        scrollPane.setHorizontalScrollBarPolicy
                (ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS);
        panel_2.add(scrollPane);

        editorPane = new JEditorPane();
        editorPane.setContentType("text/html");
        scrollPane.setViewportView(editorPane);

           //初期状態のボタンの使用可否
           btnNewButton.setEnabled(true);
           btnNewButton_1.setEnabled(true);
           btnNewButton_2.setEnabled(false);
           btnNewButton_3.setEnabled(false);

    }

    private class SwingAction extends AbstractAction {
        public SwingAction() {
            putValue(NAME, "ログイン");
            putValue(SHORT_DESCRIPTION, "データベースにログインします");
        }
        public void actionPerformed(ActionEvent e) {
```

```
            // DB接続パラメータの設定
            String ip = textField.getText();
            String db = txtMrp.getText();
            String tel = textField_1.getText();
            // 入力パスワードのハッシュ化
            char[] password = passwordField.getPassword();
            String pass = new String(password);
            String hashPass = DigestUtils.sha256Hex(pass);  //ハッシュ値を求める

             // DB接続・ユーザ検索・ハッシュパスワード取得
                LoginCheckBean lcb = new LoginCheckBean(tel,ip,db);
             // リクエストパスワードとDB取得パスワードのチェック
              if(hashPass.equals(lcb.getPassword())) {
                // 認証OKなら認証済みメッセージを設定
                String message="認証済み";
                lblNewLabel_5.setText(message);
                lblNewLabel_5.setForeground(Color.GREEN);
                   //ボタンの使用可否を検索モードに変更
                   btnNewButton.setEnabled(false);
                   btnNewButton_1.setEnabled(false);
                   btnNewButton_2.setEnabled(true);
                   btnNewButton_3.setEnabled(true);

              }else  {
                 // 認証NGならエラーメッセージを設定
                 String message="認証不可";
                 lblNewLabel_5.setText(message);
                 lblNewLabel_5.setForeground(Color.RED);

                   //ボタンの使用可否を認証モードに変更
                   btnNewButton.setEnabled(true);
                   btnNewButton_1.setEnabled(true);
                   btnNewButton_2.setEnabled(false);
                   btnNewButton_3.setEnabled(false);
              }
        }
    }
    private class SwingAction_1 extends AbstractAction {
        public SwingAction_1() {
            putValue(NAME, "リセット");
            putValue(SHORT_DESCRIPTION, "ログイン情報と在庫検索結果をクリアします");
        }
        public void actionPerformed(ActionEvent e) {
            editorPane.setText(null);
            textField_1.setText(null);
            passwordField.setText(null);

        }
    }
    private class SwingAction_2 extends AbstractAction {
        public SwingAction_2() {
            putValue(NAME, "在庫状況取得");
            putValue(SHORT_DESCRIPTION, "商品の在庫状況を検索します");
        }
        public void actionPerformed(ActionEvent e) {
            // DB接続パラメータの設定
            String ip = textField.getText();
            String db = txtMrp.getText();
```

```
                String id = "";
                // コンボボックスの取得
                String itemId=comboBox.getSelectedItem().toString();
                if("Aシリーズ".equals(itemId)) {
                    id="A%";
                }
                if("Bシリーズ".equals(itemId)) {
                    id="B%";
                }
                if("Cシリーズ".equals(itemId)) {
                    id="C%";
                }
                if("Dシリーズ".equals(itemId)) {
                    id="D%";
                }
                // 検索結果リストの実体化（商品番号、IP、DB名をプロパティに設定）
                ZaikoItemListBean zilb = new ZaikoItemListBean(id,ip,db);
                // 在庫リストを作成
                zilb.zaikoItemListHtml();
                // 表示領域に貼り付け
                editorPane.setText(zilb.getResultHtml());

            }
        }
        private class SwingAction_3 extends AbstractAction {
            public SwingAction_3() {
                putValue(NAME, "ログアウト");
                putValue(SHORT_DESCRIPTION, "データベース接続を断ちます");
            }
            public void actionPerformed(ActionEvent e) {
                   //ボタンの使用可否を認証モードに変更
                   btnNewButton.setEnabled(true);
                   btnNewButton_1.setEnabled(true);
                   btnNewButton_2.setEnabled(false);
                   btnNewButton_3.setEnabled(false);

                   editorPane.setText(null);
                   textField_1.setText(null);
                   passwordField.setText(null);

                   String message="未認証";
                   lblNewLabel_5.setText(message);
                   lblNewLabel_5.setForeground(Color.BLACK);
            }
        }
    }
```

LoginCheckBean.java

```
    package jp.ict.aso.model;

    import java.io.Serializable;
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class LoginCheckBean implements Serializable {
      private String tel;
      private String ip;
      private String db;
      private String password;

   public LoginCheckBean() {
   }
   public LoginCheckBean(String tel,String ip,String db) {
       this.tel=tel;
       this.ip=ip;
       this.db=db;
   try{
       /**
       * ●業務処理セクション
       */
       //[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
       ResultSet rs = stmt.executeQuery
                ("select TEL,PASSWORD from T_USER where TEL = '" + tel + "'");
       //[5]RDBの検索結果を取り出す
       while( rs.next() ){
             tel = rs.getString("TEL" );
             password = rs.getString("PASSWORD" );
       }
       //[6]データベースから切断する
       stmt.close();
       connection.close();

       }catch ( Exception e) {
         System.out.println("SQL実行エラー:LoginCheckBean");
         e.printStackTrace();
       }
    }
   public String getPassword() {
         return password;
   }
}
```

ZaikoItemListBean.java

```
package jp.ict.aso.model;
import java.io.Serializable;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

import javax.swing.ImageIcon;

public class ZaikoItemListBean implements Serializable{
  private String id;
  private String ip;
  private String db;
  private String resultHtml;

public ZaikoItemListBean(String id,String ip,String db){
  this.id=id;
  this.ip=ip;
  this.db=db;
  resultHtml="";
}

public void zaikoItemListHtml(){
      String name;
      int zaiko;
      String location;
      ImageIcon image;
try{
/**
* ●業務処理セクション
*/
//[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
ResultSet rs = stmt.executeQuery
  ("select ITEM_ID,ITEM_NAME,ZAIKO_NUM,LOCATION from T_ZAIKO where ITEM_ID like '"
                  + id + "' 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 width=100>No.</th><th width=100>商品ID</th>"
                    + "<th width=100>商品イメージ</th>"
                    + "<th width=100>商品名</th>"
                    +"<th width=100>在庫数</th><th width=100>保管棚</th></tr>";
int no=0;
  while( rs.next() ){
   no++;
   id = rs.getString("ITEM_ID");
   name = rs.getString("ITEM_NAME" );
   zaiko = rs.getInt( "ZAIKO_NUM" );
   location = rs.getString( "LOCATION" );
   resultHtml=resultHtml+"<tr><td>"+no+"</td>";
```

```
    resultHtml=resultHtml+"<td>"+id+"</td>";

    resultHtml += "<td><img src='"
               + getClass().getResource("/watch-images/" + id + ".png").toString()
               + "' width=100></td>";

    resultHtml=resultHtml+"<td>"+name+"</td>";
    resultHtml=resultHtml+"<td>"+zaiko+"</td>";
    resultHtml=resultHtml+"<td>"+location+"</td></tr>";
   }
resultHtml=resultHtml+"</table>";
System.out.println(resultHtml);
//[6]データベースから切断する
stmt.close();
connection.close();

}catch ( Exception e) {
   e.printStackTrace();
   }
}
public String getResultHtml() {
   return resultHtml;
   }
}
```