

# LAB2: SWING INTRODUCTION

## MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

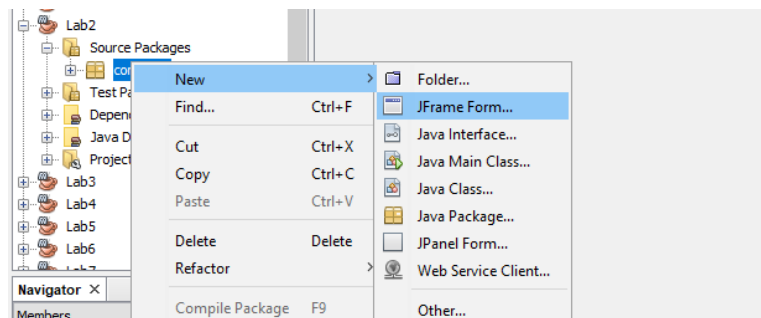
- ✓ Thiết kế được giao diện
- ✓ Xử lý sự kiện
- ✓ Lập trình được với giao diện

## Hướng dẫn thiết kế giao diện swing

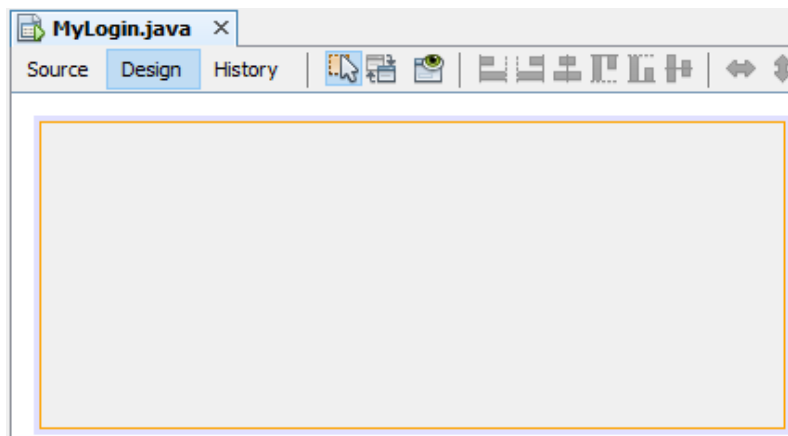
### HD1: Tạo project swing

- ✓ **Bước 1:** Tạo 1 project mới tên Lab2.
- ✓ **Bước 2:** Click phải package -> New -> JFrame Form.

Class Name đặt tên: “MyLogin” -> Finish để tạo mới một JFrame với giao diện đồ họa

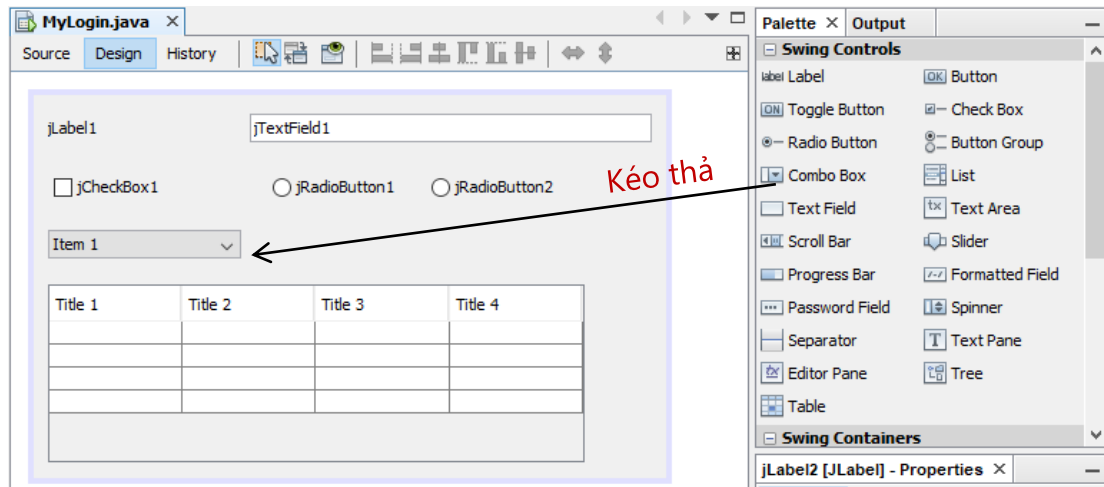


- ✓ **Bước 3:** Chú ý 2 Tab Source và Design. Tab *source* để chuyển sang *giao diện code*, Tab *design* để hiện thị *giao diện đồ họa* kéo thả.



✓ **Bước 4:** Thêm các control

- Chọn Tab Design.
- Trong cửa sổ Palette kéo điều khiển (Label, Text Field, Check Box, Radio Button + Button Group, Combo Box, Table, Button) vào giao diện.  
(Nếu không thấy cửa sổ Palette thì chọn Window -> IDE Tools -> Palette)



- Trong cửa sổ Properties chỉnh lại thuộc tính cho các điều khiển.

Các thuộc tính thường dùng cho từng loại điều khiển:

1. Label (Nhãn)

- ❖ Text: nội dung hiển thị
- ❖ Name: tên của nhãn, nên bắt đầu bởi **lbl**

2. Text Field (Ô nhập)

- ❖ Text: nội dung ô nhập
- ❖ Name: tên của ô nhập, nên bắt đầu bởi **txt**

3. Text Area (Ô nhập cho phép chứa nhiều dòng):

Thuộc tính tương tự Text Field

4. Password Field (Ô nhập mật khẩu): Thuộc tính tương tự Text Field

5. Check Box (Hộp kiểm)

- ❖ Text: Nhãn đính kèm
- ❖ Selected: Trạng thái
- ❖ Name: Tên, nên bắt đầu bởi **chk**

6. Button Group (Nhóm các điều kiện): Thuộc loại other components, không hiển thị trên giao diện

- ❖ Name: Tên, nên bắt đầu bởi **grp**

7. Radio Button (Lựa chọn)

- ❖ Text: Nhãn đính kèm

- ❖ Selected: Trạng thái

- ❖ Name: Tên, nên bắt đầu bởi **rdo**

- ❖ ButtonGroup: nhóm mà radio thuộc vào. Mỗi radio chỉ có thể chọn một group.

8. Combo Box (Hộp chọn)

- ❖ Model: chứa danh sách dữ liệu

- ❖ SelectedIndex: vị trí mục được chọn

- ❖ SelectedItem: dữ liệu mục chọn

- ❖ Name: tên, nên bắt đầu bởi **cbo**

9. Table (Bảng)

- ❖ Model: dữ liệu bảng

- ❖ SelectionMode: chế độ chọn các hàng

- ❖ RowHeight: chiều cao mỗi hàng

- ❖ Name: Tên, nên bắt đầu bởi **tbl**

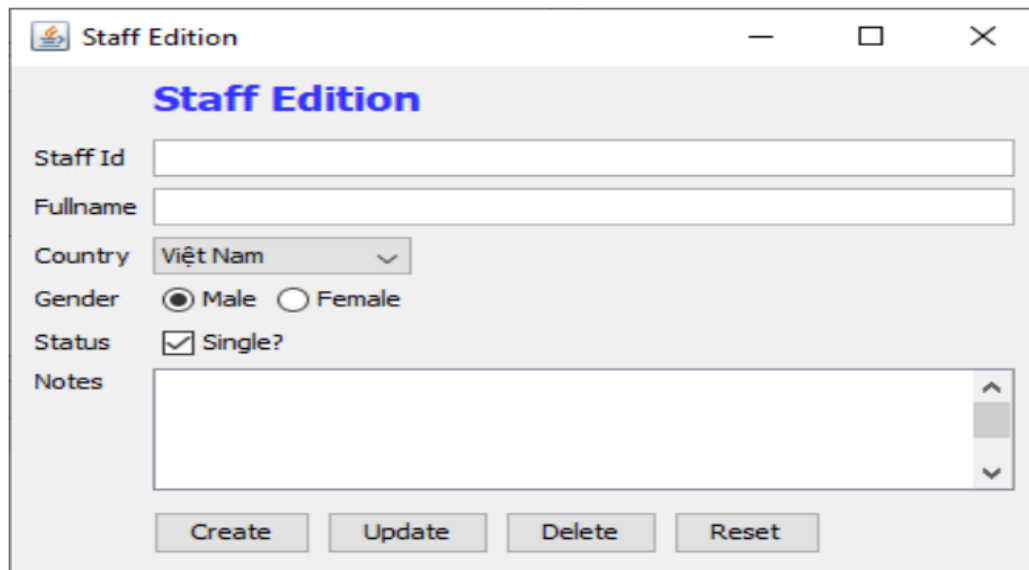
10. Button (Nút nhấn)

- ❖ Text: nhãn của nút

- ❖ Name: tên của nút, nên bắt đầu bởi **btn**

**Dựa vào HD1 thực hiện thiết kế các giao diện sau**

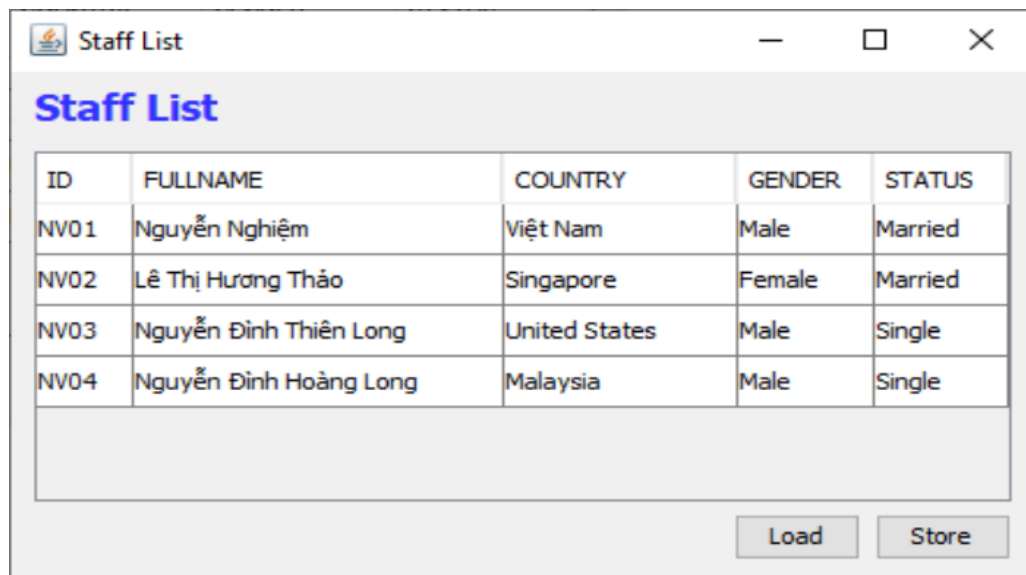
**Bài 1 (2 điểm)** Thiết kế giao diện quản lý nhân viên như hai hình



Hình 1: Form nhập thông tin nhân viên

Lưu ý để làm 2 radio Male và Female loại trừ lẫn nhau thì ta kéo vào 1 control tên Button Group. Sau khi kéo trên giao diện ta không thấy nhưng kiểm tra trong mã lại có 1 đối tượng tên `buttonGroup1` → đổi name `grpGender`.

Quay về giao diện, chọn radio Male, chỉnh thuộc tính `buttonGroup` là `grpGender`. Làm tương tự cho radio Female.



ID	FULLNAME	COUNTRY	GENDER	STATUS
NV01	Nguyễn Nghiêm	Việt Nam	Male	Married
NV02	Lê Thị Hương Thảo	Singapore	Female	Married
NV03	Nguyễn Đình Thiên Long	United States	Male	Single
NV04	Nguyễn Đình Hoàng Long	Malaysia	Male	Single

Hình 2: Danh sách nhân viên

Control	Name	Description
TextField	txtId	Staff Id
TextField	txtFullname	Fullname
ComboBox	cboCountry	Country
RadioButton	rdoMale	Gender
RadioButton	rdoFemale	
ButtonGroup	grpGender	
CheckBox	chkStatus	Single?
TextArea	txtNotes	Notes
JTable	tblStaffList	Staff List

### TableModel của Staff List

tblStaffList [JTable] - model

Set **tblStaffList's model** property using: Table model customizer

Table Model

Table Settings Default Values

Specify Title and Column Types Here:

Column	Title	Type	Editable
1	ID	Object	<input type="checkbox"/>
2	Fullname	Object	<input checked="" type="checkbox"/>
3	Country	Object	<input checked="" type="checkbox"/>
4	Gender	Object	<input checked="" type="checkbox"/>
5	Status	Object	<input checked="" type="checkbox"/>

Rows: 4 Columns: 5

OK Reset to Default Cancel Help

tblStaffList [JTable] - model

Set **tblStaffList's model** property using: Table model customizer

Table Model

Table Settings Default Values

Default Table Values:

ID	Fullname	Country	Gender	Status
NV01	Tran Tuan Hai	Viet Nam	Male	Single
NV02	Nguyen Thi Thu Thuy	Singapore	Female	Married
NV03	Le Thanh Tung	Thailand	Male	Married
NV04	Ngô Thị Hồng Hạnh	Viet Nam	Female	Single

Columns: Insert Delete Move Left Move Right

Rows: Insert Delete Move Up Move Down

Rows: 4 Columns: 5

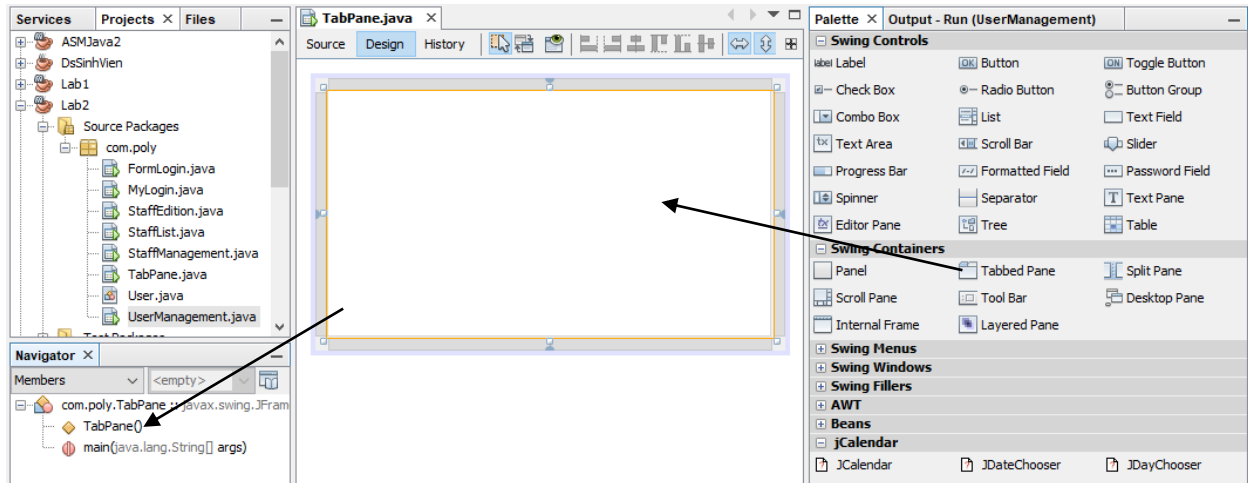
OK Reset to Default Cancel Help

## HD2: tổ chức giao diện với TabPane

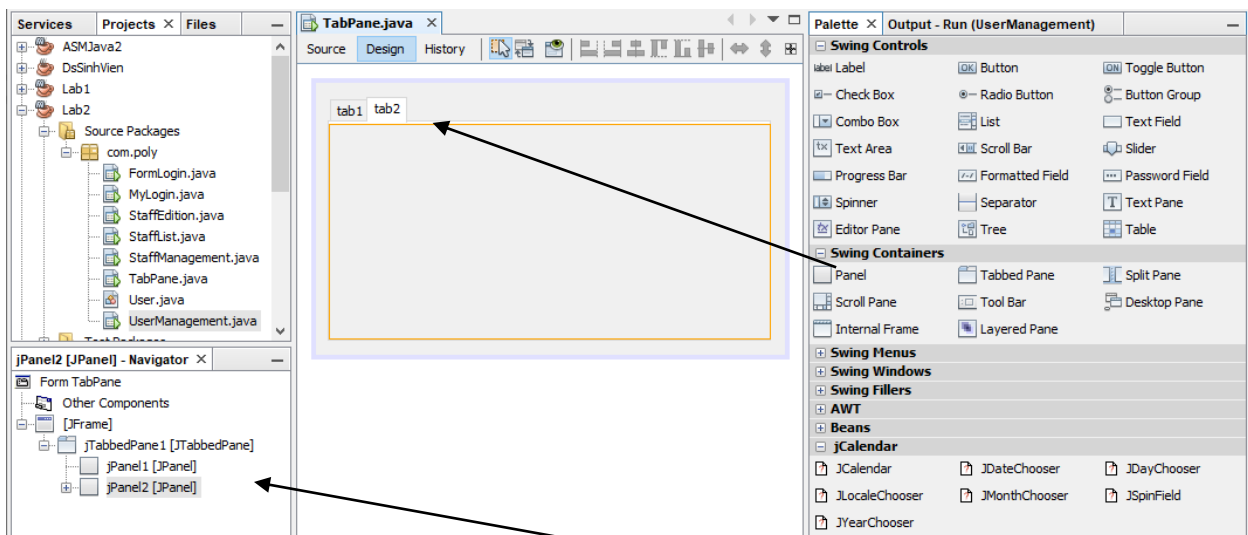
✓ **Bước 1:** Click phải package -> New -> JFrame Form.

Class Name đặt tên: “TabPane” -> Finish

✓ **Bước 2:** Kéo thả Tabbed Pane vào giao diện.



✓ **Bước 3:** Kéo thả Panel vào TabPane đã tạo.

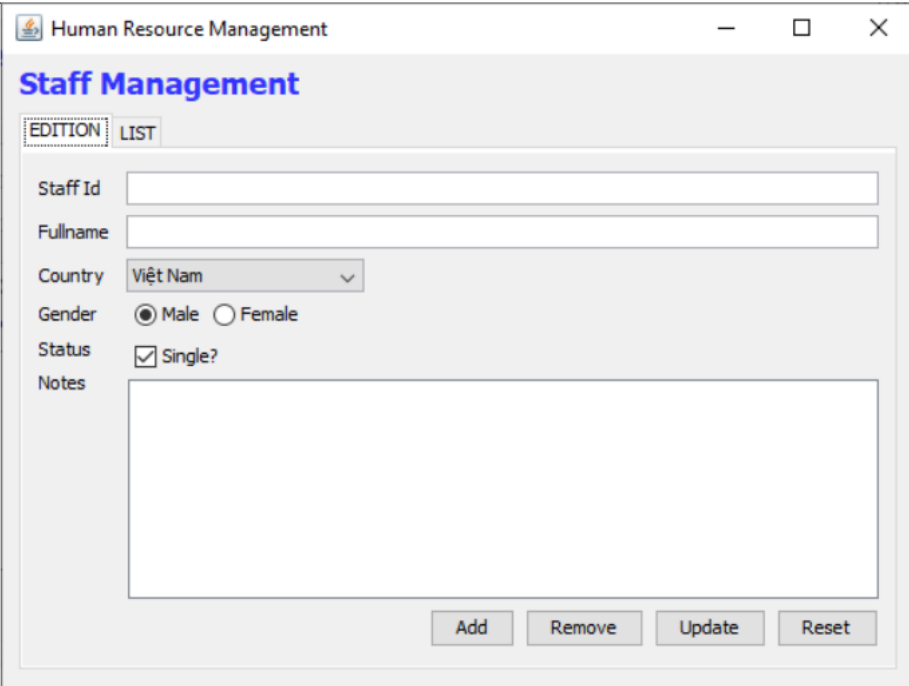


Lưu ý: Tất cả các JPanel phải nằm trong cùng 1 JTabbedPane. Nếu chưa thấy đúng thì dùng chuột kéo JPanel thả vào trong JtabbedPane là được (Thao tác tại hộp thoại Navigator).

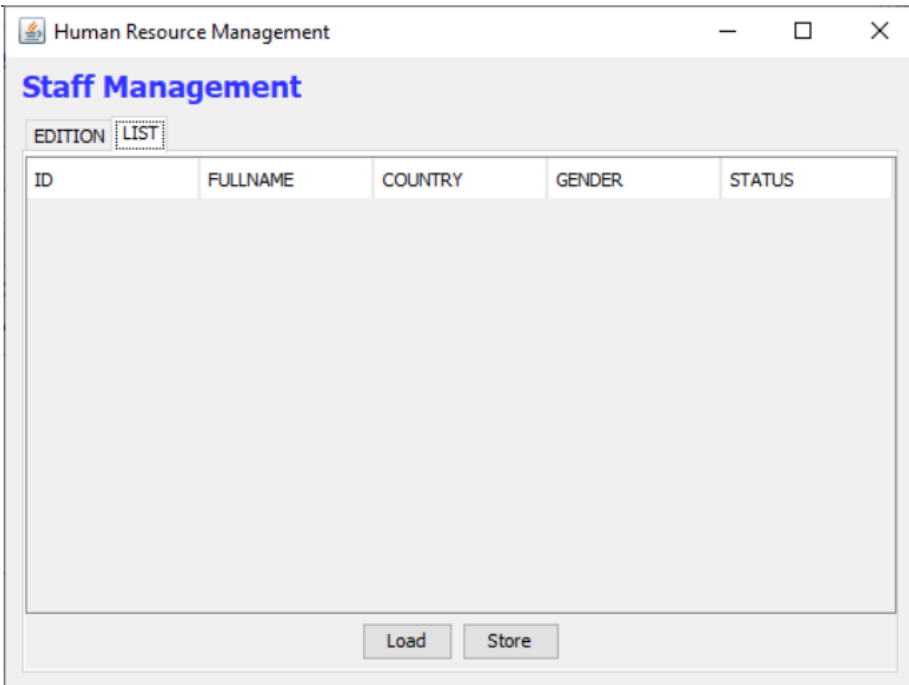
✓ **Bước 4:** Tạo giao diện trong các Tab của TabPane tương tự như các giao diện trước.

**Dựa vào HD2 thực hiện thiết kế các giao diện sau**

**Bài 2 (2 điểm):** Thiết kế giao diện cho của sổ quản lý nhân viên được tổ chức với TabPane như 2 hình sau:



Hình 3: Tab nhập thông tin nhân viên



Hình 4: Tab chứa danh sách sinh viên

**HD3: Bẫy các sự kiện theo yêu cầu, tổ chức viết và gọi các hàm xử lý nghiệp vụ**

Chọn **Tab Source** để xem source code và tổ chức viết các hàm người dùng.

- ✓ Ta thấy class `MyLogin` kế thừa từ `javax.swing.JFrame`. Lớp này là một `JFrame` có giao diện đồ họa, `MyLogin` kế thừa từ `JFrame` nên có được giao diện đồ họa.

```

6   package com.poly;
7
8   /**
9    *
10   * @author NgaHTH4
11   */
12   public class MyLogin extends javax.swing.JFrame {
13
14       /**
15        * Creates new form MyLogin
16        */
17       public MyLogin() {
18           initComponents();
19       }
20

```

- ✓ Trên NetBeans chú ý phần “generated code” màu xám, phần này sẽ tự phát sinh khi ta thiết kế kéo thả, ta không viết code vào phần xám này được.

```

30   // <editor-fold defaultstate="collapsed" desc="Generated Code">
31   private void initComponents() {
32
33       lblLogin = new javax.swing.JLabel();
34       jLabel1 = new javax.swing.JLabel();
35       jLabel2 = new javax.swing.JLabel();
36       txtPassword = new javax.swing.JTextField();
37       txtUsername = new javax.swing.JTextField();
38       chkRemember = new javax.swing.JCheckBox();
39       btnLogin = new javax.swing.JButton();
40       btnReset = new javax.swing.JButton();
41
42       setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
43       setTitle("Login");
44
45       lblLogin.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
46       lblLogin.setForeground(new java.awt.Color(0, 0, 153));
47       lblLogin.setText("Login Form");
48
49       jLabel1.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
50       jLabel1.setForeground(new java.awt.Color(0, 0, 51));
51       jLabel1.setText("Username");

```



- ✓ Trong hàm main có nội dung như hình. Khi click phải run file MyLogin.java, trong code của hàm main tạo MyLogin và hiển thị nó lên.

```

114  /**
115   * @param args the command line arguments
116   */
117  public static void main(String args[]) {
118      /* Set the Nimbus look and feel */
119      Look and feel setting code (optional)
120
121
122      /* Create and display the form */
123      java.awt.EventQueue.invokeLater(new Runnable() {
124          public void run() {
125              new MyLogin().setVisible(true);
126          }
127      });
128  }

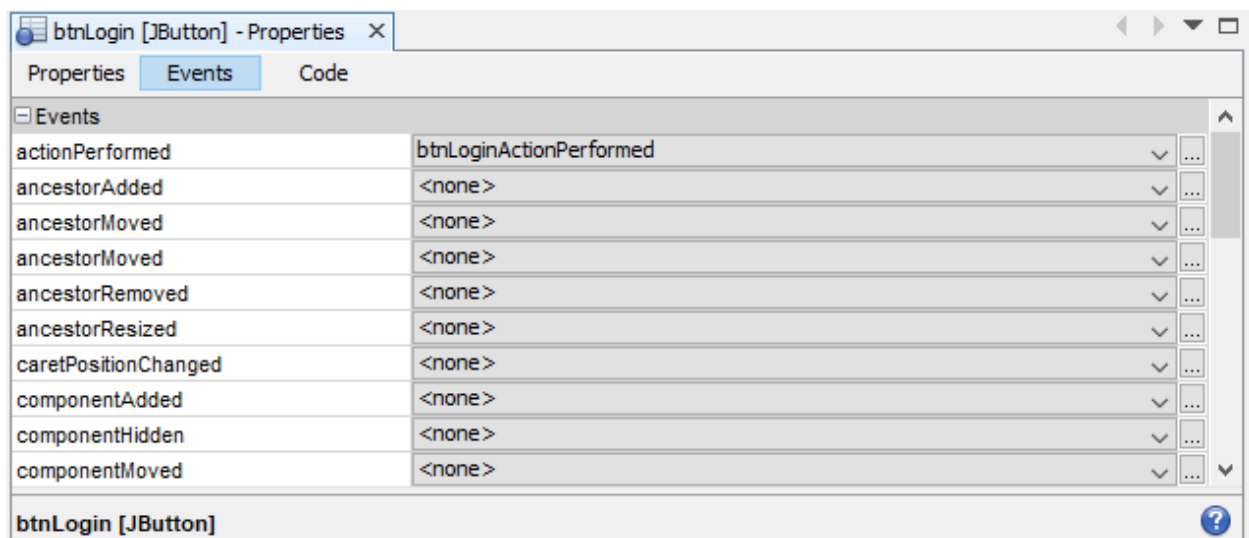
```

Event (sự kiện) là một sự thay đổi trong trạng thái của đối tượng, chẳng hạn như sự kiện mô tả sự thay đổi trong trạng thái của source. Các sự kiện được tạo ra là do tương tác của người dùng với các thành phần UI. Ví dụ như việc nhấn vào một nút button, di chuyển chuột, nhập ký tự thông qua bàn phím, ...

Bắt (bẫy) sự kiện là tương ứng với một sự kiện ta sẽ gọi đến 1 hàm để viết phần xử lý cho sự kiện đó.

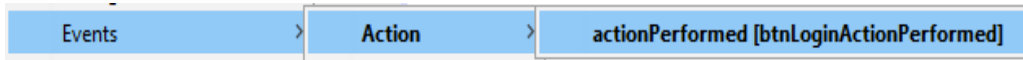
Cửa sổ Events hiển thị các sự kiện có thể thực hiện cho control ta đang chọn.

Trong cửa sổ Properties, chọn qua tab Events ta thấy các sự kiện của **btnLogin**



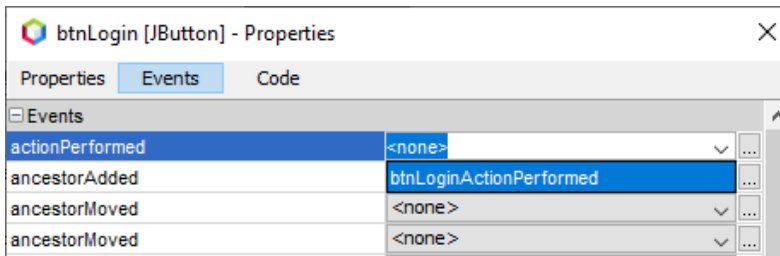
Các cách bắt sự kiện ActionPerformed (ví dụ Nút Login - btnLogin)

1. Double Click tại điều khiển Login
2. Right Click tại điều khiển → Events → Action → actionPerformed



3. Mở tại Events của cửa sổ Properties

Tại sự kiện actionPerformed → <none> → btnLoginActionPerformed



Ta có hàm được phát sinh:

```
129 private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {
130     // TODO add your handling code here:
131 }
```

Code mã xử lý vào sau dòng // TODO add your handling code here:

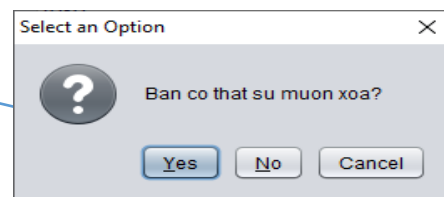
Các hàm xử lý nên viết tập trung ở cuối file để dễ quản lý.

#### HD4: Giới thiệu hộp thoại thông báo

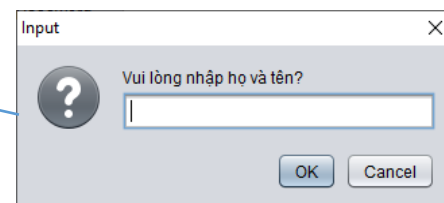
✓ JOptionPane là một dạng cửa sổ được sử dụng để tạo các hộp thoại thông báo.

Có 3 dạng thông báo thường dùng

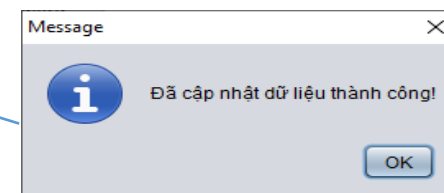
1. Xác nhận – Confirm



2. Nhập – Input



3. Thông báo – Message



## PHẦN II

### Bài 3 (2 điểm)

Lập trình cho form đăng nhập theo yêu cầu sau:

Hình 5: Form đăng nhập

Name	Event	Yêu cầu nghiệp vụ
btnAdd	Action	<ul style="list-style-type: none"> <li>• Không để trống username và password</li> <li>• Đăng nhập đúng khi                         <ul style="list-style-type: none"> <li>○ Username là fpt</li> <li>○ Password là polytechnic</li> </ul> </li> <li>• Nếu có tích vào Remember me thì đưa ra thông báo “Tài khoản của bạn đã được ghi nhớ”</li> <li>• Đưa ra thông báo phù hợp</li> </ul>
btnReset	Action	<ul style="list-style-type: none"> <li>• Xóa trắng username và password</li> <li>• Tích sẵn lên Remember me</li> </ul>

### HƯỚNG DẪN:

- ✓ Đọc ghi các thuộc tính Text của control với `x.getText()` và `x.setText(String)`
- ✓ Đọc ghi trạng thái của checkbox với `x.isSelected()` và `x.setSelected(boolean)`
- ✓ Hiện thị hộp thoại thông báo với  
`JOptionPane.showMessageDialog(this, “Thông báo của bạn”)`

Code **Button Login** với Username: fpt và Password: polytechnic

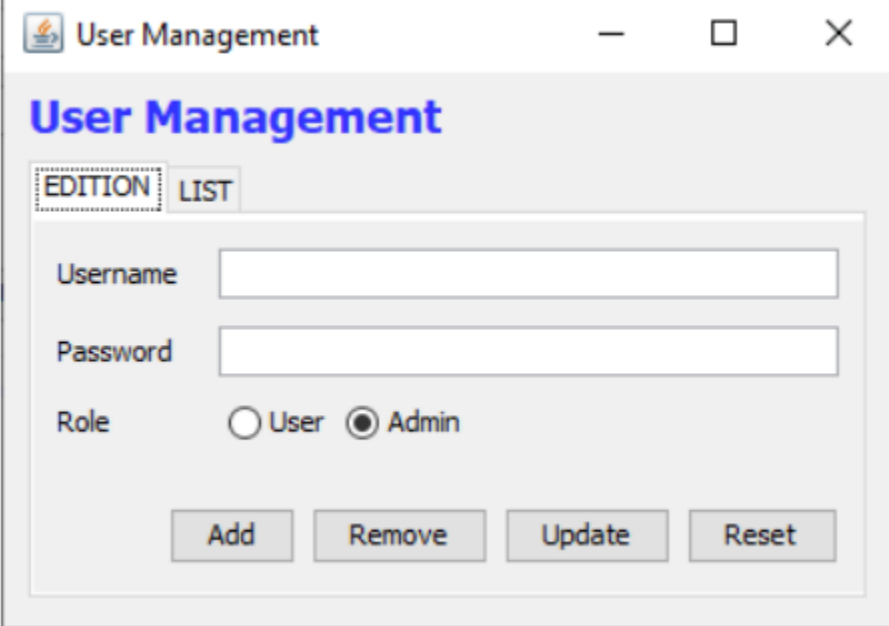
```
153 private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {  
154     // TODO add your handling code here:  
155     if (txtUsername.getText().equals("")) {  
156         JOptionPane.showMessageDialog(this, "Username is not null!");  
157     } else if (txtPassword.getText().equals("")) {  
158         JOptionPane.showMessageDialog(this, "Password is not null!");  
159     } else if (txtUsername.getText().equalsIgnoreCase("fpt") &&  
160         txtPassword.getText().equalsIgnoreCase("polytechnic")) {  
161         JOptionPane.showMessageDialog(this, "Login Successful!");  
162         if (chkRemember.isSelected()) {  
163             JOptionPane.showMessageDialog(this, "Your account is remember.");  
164         }  
165     } else {  
166         JOptionPane.showMessageDialog(this, "Usernaeme or Password not found. Login Fail!");  
167     }  
168 }  
169 }  
170
```

### **Button Reset**

```
171 private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  
172     // TODO add your handling code here:  
173     txtUsername.setText("");  
174     txtPassword.setText("");  
175     chkRemember.setSelected(true);  
176 }  
177
```

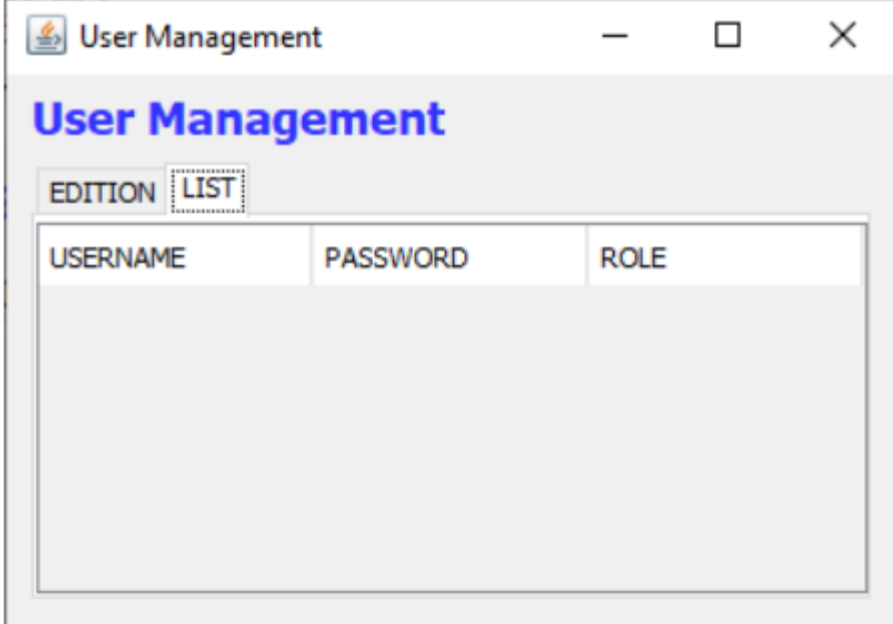
**Bài 4 (2 điểm)**

Lập trình quản lý người sử dụng theo mô tả sau.



The image shows a Java Swing window titled "User Management". It has a standard title bar with minimize, maximize, and close buttons. The window content features a blue title "User Management" at the top. Below the title are two tabs: "EDITION" (which is selected and highlighted with a dashed border) and "LIST". The "EDITION" tab contains a form with three input fields: "Username", "Password", and "Role". The "Role" field has two radio buttons, "User" and "Admin", with "Admin" being selected. At the bottom of the form are four buttons: "Add", "Remove", "Update", and "Reset".

Hình 6: Form nhập thông tin người sử dụng



The image shows the same "User Management" window, but with the "LIST" tab selected. The "EDITION" tab is now disabled. The "LIST" tab displays a table with three columns: "USERNAME", "PASSWORD", and "ROLE". The table is currently empty, showing only the header row.

Hình 7: Danh sách người sử dụng

Control	Sự kiện	Yêu cầu thực hiện
JFrame	WindowOpened	+ Đổ List<User> vào bảng
tblUserList	MouseClicked	+ Hiển thị thông tin của user được chọn lên form
btnReset	Action	+ Xóa trắng form
btnAdd	Action	+ Thêm user mới vào List<User> + Hiển thị lại bảng danh sách + Xóa trắng form + Đưa ra thông báo
btnUpdate	Action	+ Cập nhật user đang xem + Hiển thị lại bảng danh sách + Đưa ra thông báo
btnRemove	Action	+ Xóa user đang xem + Hiển thị lại bảng danh sách + Xóa trắng form + Đưa ra thông báo

### HƯỚNG DẪN:

1. Tạo lớp User với các thuộc tính sau (nên encapsulate)

- ✓ Username
- ✓ Password
- ✓ Role

```
package com.poly;

/**
 *
 * @author NgaHTH4
 */
public class User {

    private String username;
    private String password;
    private String role;
```

Insert code các phương thức Getter/Setter.

Insert code 2 Constructor có đủ 3 tham số và không tham số

## 2. Viết mã tập trung ở cuối file để dễ quản lý

✓ Khai báo List<User> để chứa danh sách user

```
342 List<User> list = new ArrayList<>();
```

✓ void fillTable() // đổ List<User> vào Model

```
344 void fillTable() {
345     DefaultTableModel model = (DefaultTableModel) tblUserList.getModel();
346     model.setRowCount(0);
347     for (User u : list) {
348         Object[] row = {u.getUsername(), u.getRole()};
349         model.addRow(row);
350     }
351 }
```

### Giải thích:

Thêm hàng vào model bằng cách dùng for duyệt từng phần tử của ArrayList.

Lấy từng phần tử trong ArrayList, tách dữ liệu username, role và thêm mảng row, cuối cùng add row vào model như 1 hàng dữ liệu.

Lệnh setRowCount(0) để xóa sạch dữ liệu trong model trước khi đổ dữ liệu để tránh trường hợp ta gọi hàm fillTable nhiều lần thì dữ liệu sẽ bị trùng lặp lên nhiều lần.

✓ User readForm() // Tạo đối tượng user với dữ liệu form

```
353 User readForm() {
354     String user = txtUsername.getText();
355     String pass = pwfPassword.getText();
356     String role = "";
357     if (rdoAdmin.isSelected()) {
358         role = "Admin";
359     } else {
360         role = "User";
361     }
362     User u = new User(user, pass, role);
363     return u;
364 }
```

### Giải thích:

Đọc dữ liệu username và password được nhập trên form bằng hàm getText().

Dựa vào radio Admin hay User được chọn thông qua hàm isSelected() để gán quyền cho role

Sau khi thu thập dữ liệu xong ta tạo ra một user mới từ dữ liệu và trả về cho hàm

✓ **void writeForm(User)** // Hiển thị thông tin từ user lên form

```

371 void writeForm(User u) {
372
373     txtUsername.setText(u.getUsername());
374     pwfPassword.setText(u.getPassword());
375     if (u.getRole().equals("Admin")) {
376         rdoAdmin.setSelected(true);
377     } else {
378         rdoUser.setSelected(true);
379     }
380 }

```

#### Giải thích:

Gán thông tin đọc được từ user được truyền vào, đưa lên các TextField username và password qua hàm setText().

Thể hiện radio Admin hay User được chọn thông qua hàm setSelected() dựa vào role

✓ **void addUser()** // Tạo user từ form và thêm vào List<User>

```

382 void addUser() {
383     User u = readForm();
384     list.add(u);
385     fillTable();
386     clearForm();
387     JOptionPane.showMessageDialog(this, "Them mot user thanh cong!");
388     txtUsername.requestFocus();
389 }

```

#### Giải thích:

Tạo một user từ thông tin trên màn hình đọc bằng hàm readForm().

Thêm user vào list. Đổ dữ liệu lại cho table. Xóa trắng các textField tên Form.

Xuất hộp thoại thông báo thêm thành công

Đưa con trỏ đứng tại textField txtUsername.

✓ **void removeUser()** // Xóa user đang xem khỏi List<User>

```

391 void removeUser() {
392     int ind = tblUserList.getSelectedRow();
393     list.remove(ind);
394     fillTable();
395     JOptionPane.showMessageDialog(this, "Xoa User thanh cong!");
396 }

```

#### Giải thích:

Xác định dòng được chọn trên table. Xóa user trong list tại vị trí đã xác định.

Đổ dữ liệu lại cho table. Xuất hộp thoại thông báo xóa thành công



✓ void updateUser() // Thay thế user đang xem với thông tin mới

```

404 void updateUser() {
405     int ind = tblUserList.getSelectedRow();
406     User u = readForm();
407     list.set(ind, u);
408     fillTable();
409     JOptionPane.showMessageDialog(this, "Cập nhật thông tin user thành công!");
410 }

```

#### Giải thích:

Xác định dòng được chọn trên table. Đọc dữ liệu trên form

Cập nhật user trong list tại vị trí đã xác định. Đổ dữ liệu lại cho table.

Xuất hộp thoại thông báo cập nhật thành công

✓ void clearForm() // Xóa trắng form

```

366 void clearForm() {
367     writeForm(new User());
368     txtUsername.requestFocus();
369 }

```

#### Giải thích:

Ghi lên form thông tin user mới (dữ liệu trắng)

Đưa con trỏ đúng tại textField txtUsername.

✓ void selectUser() // Thể hiện dòng được chọn ra form

```

419 private void selectUser() {
420     int ind = tblUserList.getSelectedRow();
421     if (ind < 0) {
422         return;
423     }
424     User u = list.get(ind);
425     writeForm(u);
426     tab.setSelectedIndex(0);
427 }

```

#### Giải thích:

Xác định dòng được chọn trên table. Nếu chưa được chọn thì không làm gì cả

Lấy user tại dòng được chọn ghi lên form.

Chuyển chọn sang tab Staff Edition

3. Bẫy các sự kiện theo yêu cầu mô tả trên và gọi các hàm đã viết

- ✓ Bổ sung lệnh `setLocationRelativeTo(null);` → để form hiển thị giữa màn hình.

```
22 public UserManagement() {
23     initComponents();
24     setLocationRelativeTo(null);
25 }
```

- ✓ Khi chọn nút **Add**

```
259 private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
260     // TODO add your handling code here:
261     addUser();
262 }
```

- ✓ Khi chọn nút **Remove**

```
254 private void btnRemoveActionPerformed(java.awt.event.ActionEvent evt) {
255     // TODO add your handling code here:
256     removeUser();
257 }
```

- ✓ Khi chọn nút **Update**

```
269 private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt) {
270     // TODO add your handling code here:
271     updateUser();
272 }
```

- ✓ Khi chọn nút **Reset**

```
279 private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {
280     // TODO add your handling code here:
281     this.clearForm();
282 }
```

- ✓ Khi mở form, con trỏ đặt tại `txtUsername`

```
274 private void formWindowOpened(java.awt.event.WindowEvent evt) {
275     // TODO add your handling code here:
276     txtUsername.requestFocus();
277 }
```

- ✓ Khi click **chọn 1 dòng trong Staff List**, sẽ hiển thị thông tin dòng được chọn lên Tab Staff Edition

```
264 private void tblUserListMouseClicked(java.awt.event.MouseEvent evt) {
265     // TODO add your handling code here:
266     selectUser();
267 }
```

**Bài 5 (2 điểm)**

Giảng viên cho thêm.

1. Bổ sung hàm `boolean isValidated()`
  - Hàm trả về true nếu nhập đủ username và password
  - Ngược lại ô nhập nào bỏ trống sẽ thông báo phải nhập dữ liệu cho ô đó đồng thời trả về giá trị false.
2. Sửa mã các nút Add, Update, Delete chỉ thực hiện khi validate thành công, ngược lại thông báo hành động không thực hiện được.
3. Bổ sung lệnh cho nút Delete có hiển thị hộp thoại confirm và chỉ xóa khi người dùng chọn OK.

Chúc các bạn làm bài thật tốt.

HẾT.