Session 5

Lists and Panes

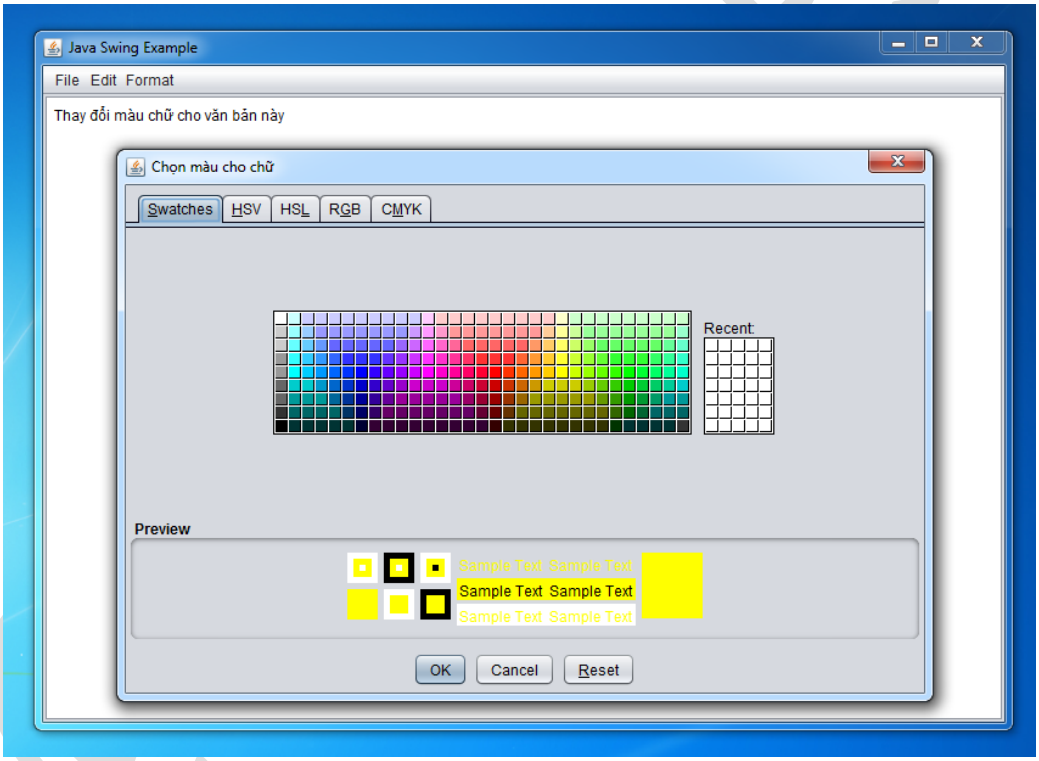
Phần I - Thực hiện trong 120 phút

1. *Mục tiêu*

- Hiểu và sử dụng JColorChooser.
- Hiểu và sử dụng JList.
- Hiểu và sử dụng JCombobox
- Hiểu và sử dụng JSplistPane.
- Hiểu và sử dụng JTabbedPane.

2. *Thực hiện*

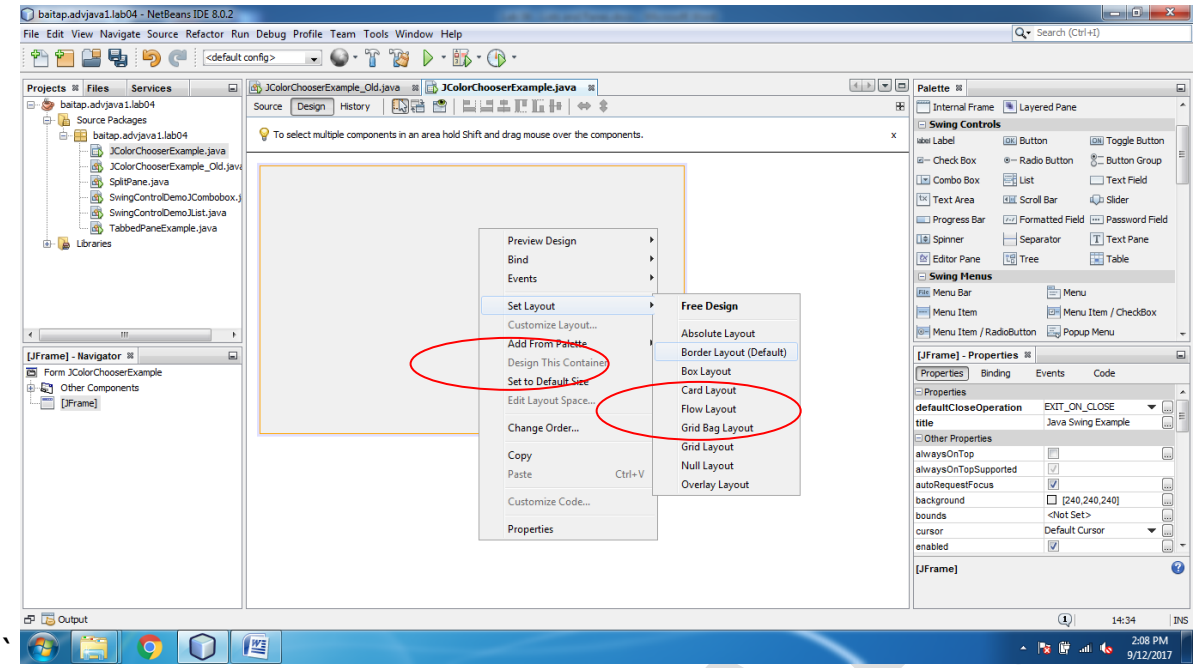
Bài thực hành 1: Viết ứng dụng cho phép lựa chọn màu sắc.



Bước 1: Tạo một JFrame Form đặt tên là "JColorChooserExample"

Đặt tiêu đề cho form là "Java Swing Example"

Bước 2: Click chuột phải vào form chọn "Set Layout", chọn "Border Layout"



Sau đó kéo thả một JTextArea vào trong form (JTextArea sẽ được mở rộng kích thước bằng với kích thước của form - kể cả khi phóng to hay thu nhỏ).

Click chuột phải vào JTextArea chọn "Change Variable Name..." rồi đặt tên cho JTextArea này là "txaCompose".

Ấn phím F2 (hoặc nháy đúp chuột vào txaCompose) rồi nhập vào dòng chữ

" Thay đổi màu chữ cho văn bản này "

Bước 3: Gắp một Menu Bar vào form, sau đó gắp một Menu vào sau menu Edit rồi đặt tên "Format"

Gắp tiếp một Menu đặt vào menu Format (Sẽ được 1 menu con có mũi tên sang phải), đổi tên thành "Color"

Gắp một Menu Item vào menu "Color" và đặt tên "ForeColor"

Gắp một Menu Item vào menu "Color" và đặt tên "BackColor"

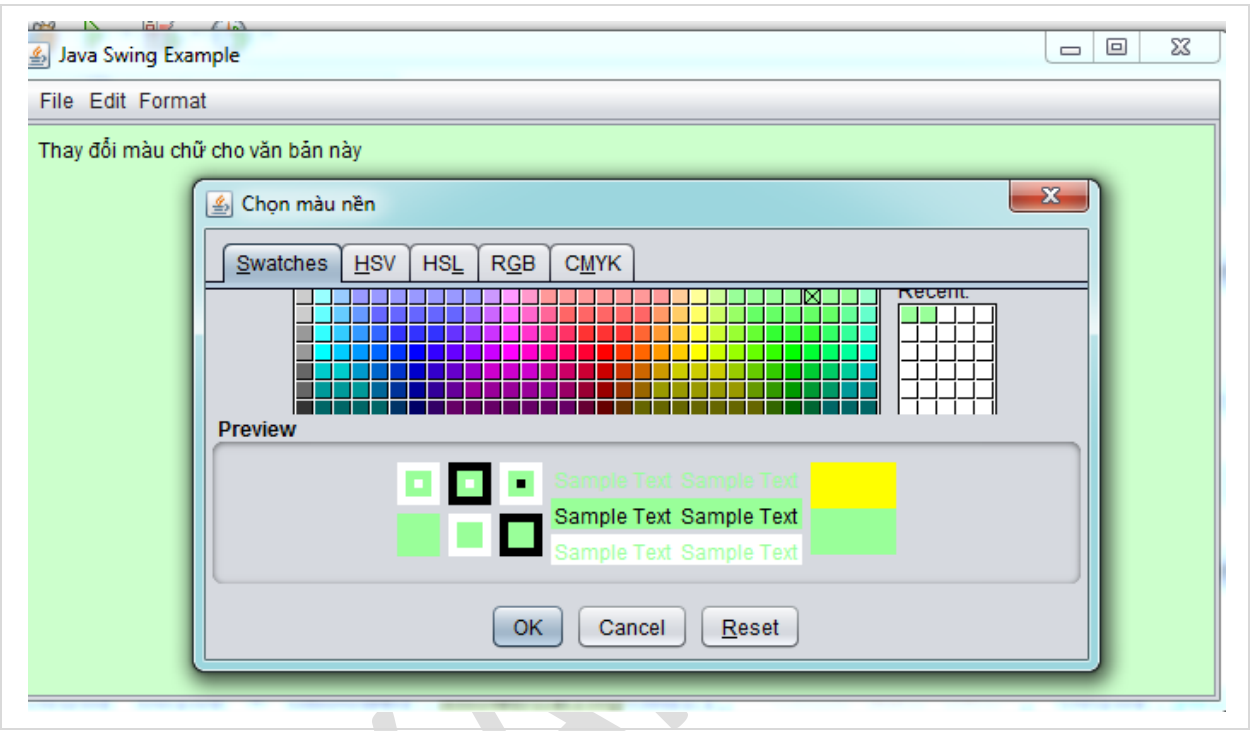
Bước 4: Nháy đúp chuột vào ForeColor và thêm code sau vào:

```
private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        JColorChooser chooser = new JColorChooser();
        Color color = chooser.showDialog(null, "Chọn màu cho chữ", Color.yellow);
        txaCompose.setForeground(color);
    }
```

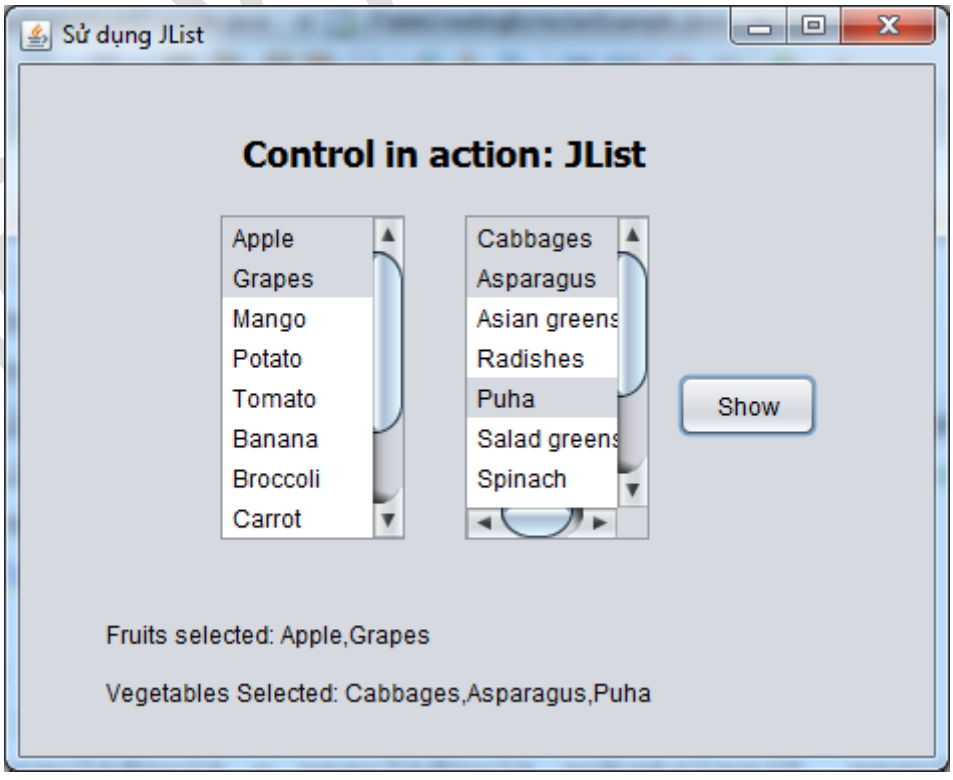
Bước 5: Nháy đúp chuột vào BackColor và thêm code sau:

```
private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        JColorChooser chooser = new JColorChooser();
        Color color = chooser.showDialog(null, "Chọn màu cho chữ", Color.yellow);
        txaCompose.setBackground(color);
    }
```

Bước 6: Chạy chương trình.



Bài thực hành 2: Viết ứng dụng sử dụng JList như hình.



Bước 1: Tạo một JFrame Form đặt tên "SwingControlDemoJList".

- Đặt tiêu đề của form là " Sử dụng JList"

- Gắp một JLabel vào form sửa nội dung hiển thị thành "Control in action: JList", click vào nút chọn font trong cửa sổ Properties để thay đổi font cho label này. Font style chọn là "Bold", Font size chọn là 18.

- Gắp một JList vào form để nhập danh sách các loại quả. Click phải vào JList đặt tên là "lsFruits", click vào mục "model" trong cửa sổ Properties để nhập trực tiếp dữ liệu cho JList này (Có thể sử dụng DefaultListModel để điều khiển dữ liệu thêm bớt cho JList).

- Gắp một JList khác vào form để nhập danh sách các loại rau. Thực hiện tương tự như JList phía trên. Đặt tên JList này là "lsVegetables"

- Gắp một JButton vào ngay sau JList "lsVegetables" như yêu cầu, đặt nội dung hiển thị cho JButton này là "Show".

- Gắp một JLabel đặt xuống cuối form, kéo rộng chiều dài JLabel này ra và đặt tên là "lblShowFruit".

- Gắp một JLabel khác đặt xuống ngay sau lblShowFruit và đặt tên "lblShowVegetable".

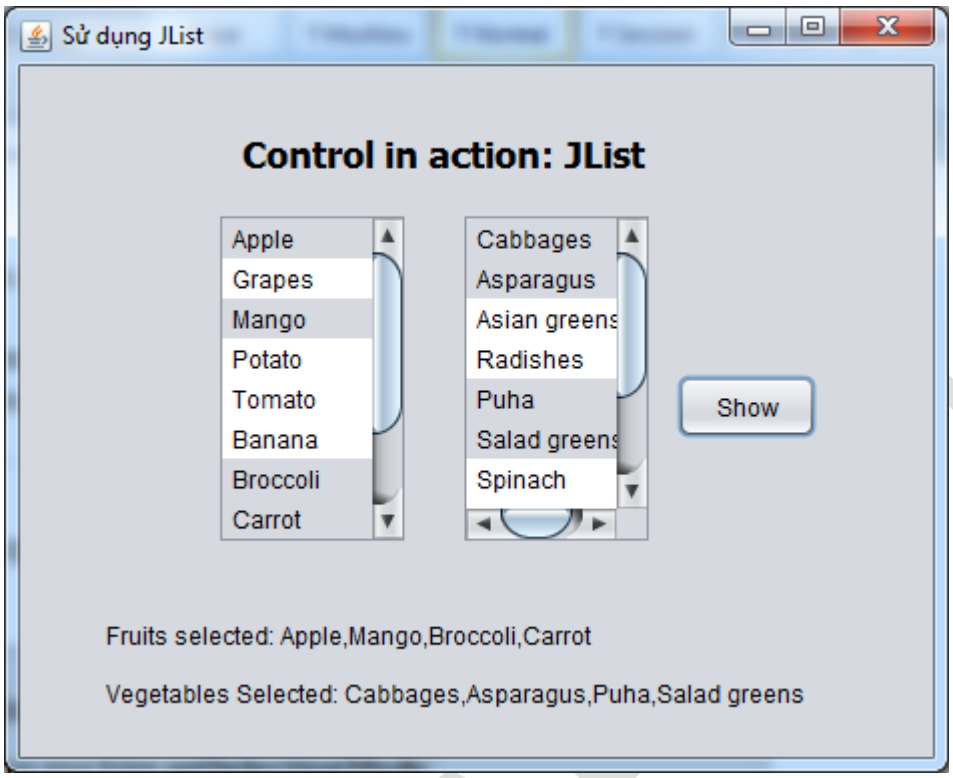
- Mặc định JList đã cho phép lựa chọn nhiều đối tượng cùng lúc với nhau.

Bước 2: Nháy đúp chuột vào button "Show" và viết code sau:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String resultFruit = "";
        String resultVege = "";
        List<String> listFruitSelected = lsFruits.getSelectedValuesList();
        List<String> listVegeSelected = lsVegetables.getSelectedValuesList();
        resultFruit = "Fruits selected: ";
        for (String f : listFruitSelected) {
            resultFruit +=f+",";
        }
        //loại bỏ kí tự dấu phẩy ở cuối chuỗi
        resultFruit = resultFruit.substring(0, resultFruit.length()-1);

        resultVege +="Vegetables Selected: ";
        for (String v : listVegeSelected) {
            resultVege += v+",";
        }
        resultVege = resultVege.substring(0, resultVege.length()-1);

        lblShowFruit.setText(resultFruit);
        lblShowVegetable.setText(resultVege);
```

```
    }
```

Bước 3: Chạy thử và xem kết quả



Lưu ý: Jlist có thể cài đặt lựa chọn 1 đối tượng hoặc nhiều đối tượng cùng lúc là do cài đặt thuộc tính thông qua hàm **setSelectionMode**:

1. fruitList.setSelectionMode(ListSelectionModel.**SINGLE_SELECTION**);
2. vegList.setSelectionMode(ListSelectionModel.**MULTIPLE_INTERVAL_SELECTION**);

Bài thực hành 3: Viết ứng dụng Java có giao diện như hình:

Bước 1: Tạo một JFrame Form đặt tên là " SwingControlDemoJCombobox"

- Đặt tiêu đề form là "Sử dụng JComboBox"

- Gắp một JLabel vào form sửa nội dung hiển thị thành "Control in action: JComboBox", click vào nút chọn font trong cửa sổ Properties để thay đổi font cho label này. Font style chọn là "Bold", Font size chọn là 18.

- Gắp một JComboBox vào form, kéo rộng chiều dài ra và click chuột phải vào để đặt tên thành " cboProvinces".

- Trong phần model ở cửa sổ Properties, click chuột vào button "..." và thay đổi nội dung của JComboBox thành tên các tỉnh thành của Việt Nam.

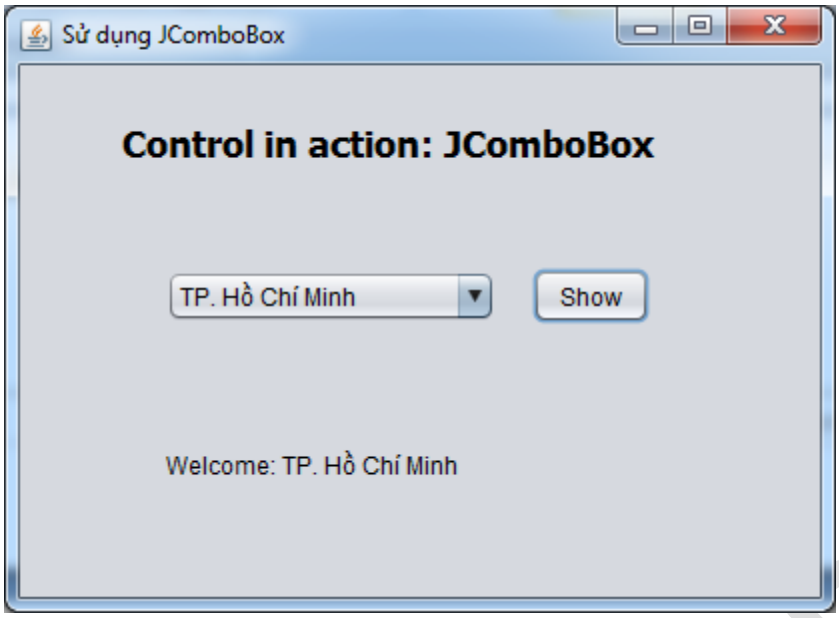
- Kéo thả một nút nhấn (JButton) và trong form ngay sau JComboBox và đổi tên thành "Show".

- Kéo thả một JLabel vào cuối form để hiển thị kết quả, xóa text hiển thị và đặt tên biến là " lblResult".

Bước 2: Nháy đúp chuột vào nút nhấn "Show" và viết code sau

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String province = (String) cboProvinces.getSelectedItem();
        lblResult.setText("You are from: "+province);
    }
```

Bước 3: Chạy và xem kết quả



- Mở rộng: JComboBox có thể nhận dữ liệu là kiểu Object, hãy xem đoạn code dưới đây để sử dụng dữ liệu của JComboBox là dữ liệu dạng Object

Tạo class MonHoc

```
// Class MonHoc lưu trữ dữ liệu của môn học
private class MonHoc {

        int id;
       String name;

public MonHoc(int id, String name) {
          this.id = id;
          this.name = name;
       }

/**
        * HàmtoStringsẽcótácdụnggửidữliệuhiểnthịlêncombobox
*/
       @Override
public String toString() {
              return   name;
       }
   }
```

Khai báo đối tượng của lớp DefaultComboBoxModel để đưa dữ liệu vào cho JComboBox:

```
package sem2.demo.advanced.controls;
```

```
import javax.swing.DefaultComboBoxModel;

/**
 *
 * @author HAITHANH
 */
public class SwingControlDemoJCombobox_Object extends javax.swing.JFrame
{

    /**
     * Creates new form SwingControlDemoJCombobox
     */
    DefaultComboBoxModel<MonHoc> comboBoxModel;

    public SwingControlDemoJCombobox_Object() {
        initComponents();

        comboBoxModel = new DefaultComboBoxModel<>();
        comboBoxModel.addElement(new MonHoc(1, "Java"));
        comboBoxModel.addElement(new MonHoc(2, "Android"));
        comboBoxModel.addElement(new MonHoc(3, "PHP"));
        comboBoxModel.addElement(new MonHoc(4, ".NET"));
        comboBoxModel.addElement(new MonHoc(5, "HTML 5"));
        comboBoxModel.addElement(new MonHoc(6, "CSS"));
        cboMonHoc.setModel(comboBoxModel);
    }

    /**
     * This method is called from within the construct
form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        cboMonHoc = new javax.swing.JComboBox();
        jButton1 = new javax.swing.JButton();
        lblResult = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Sử dụng JComboBox");

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
```

Khai báo đối tượng để đưa dữ liệu vào JComboBox

Lệnh trong constructor để khởi tạo và gán dữ liệu cho ComboBox

```
        jLabel1.setText("Control in action: JComboBox");

        jButton1.setText("Show");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(51, 51, 51)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 276,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(73, 73, 73)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(lblResult,
javax.swing.GroupLayout.PREFERRED_SIZE, 217,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(cboMonHoc,
javax.swing.GroupLayout.PREFERRED_SIZE, 165,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(18, 18, 18)
                                .addComponent(jButton1)))))
                .addContainerGap(73, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(28, 28, 28)
                .addComponent(jLabel1)
                .addGap(51, 51, 51)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cboMonHoc,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

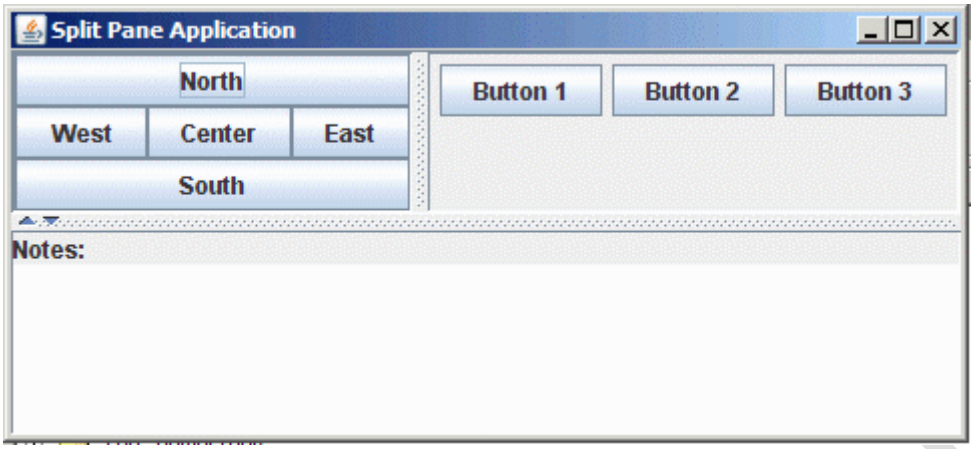
```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton1))
                .addGap(60, 60, 60)
                .addComponent(lblResult, javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(56, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        MonHoc mon = (MonHoc)cboMonHoc.getSelectedItem();
        lblResult.setText(mon.id+" - "+mon.name);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default
look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(SwingControlDemoJCombobox_Object.class.getNa
me()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(SwingControlDemoJCombobox_Object.class.getNa
me()).log(java.util.logging.Level.SEVERE, null, ex);
```

Lệnh xử lý hiển thị

```
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(SwingControlDemoJCombobox_Object.class.getNa
me()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(SwingControlDemoJCombobox_Object.class.getNa
me()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new SwingControlDemoJCombobox_Object().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JComboBox cboMonHoc;
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel lblResult;
    // End of variables declaration
}
```

Bước 6: Xem kết quả.

Bài thực hành 4: Viết ứng dụng sử dụng JSplitPane như hình.



Bước 1: Tạo một JFrame Form đặt tên "SplitPaneDemo".

- Thay đổi kích thước của form cho phù hợp.
- Đặt tiêu đề của form là "Split Pane Application"
- Click chuột phải vào form chọn "Set Layout" / "Border Layout"

Bước 2: Gắp một Split Pane vào trong form

- Chọn thuộc tính "orientation" trong cửa sổ Properties là "VERTICAL_SPLIT".
- Đặt giá trị "dividerLocation" là 200.
- Kéo chiều cao của form cho cân xứng 2 nửa theo Split Pane
- Đặt 2 JPanel vào 2 nửa của Split Pane, rồi thiết lập lại "dividerLocation" của SplitPane thành 200 (nếu nó không thay đổi thì đặt là 201, rồi lại đặt lại 200)

Bước 3: Click chuột phải vào JPanel ở nửa trên của JSplit Pane và chọn "Set Layout" chọn "Border Layout".

- Kéo thả một JSplit Pane khác vào JPanel này và thiết lập "dividerLocation" sao cho nó chia thành 2 nửa bằng nhau.
- Kéo thả 2 JPanel vào 2 nửa của JSplit Pane mới thêm vào này và thiết lập lại "dividerLocation" để hai nửa đó kích thước bằng nhau.

Bước 4: Ở góc trái trên click chuột phải chọn "Set Layout" và chọn "Border Layout"

- Kéo thả một JButton vào và chữ tên hiển thị là "CENTER"
- Gắp tiếp 1 JButton khác vào chữ tên hiển thị là "SOURTH"
- Gắp 1 JButton tiếp theo sang bên trái đặt chữ hiển thị là "WEST"
- Gắp 1 JButton vào bên phải đặt chữ hiển thị là "EAST"
- Gắp 1 JButton vào trên cùng đặt chữ hiển thị là "NORTH"

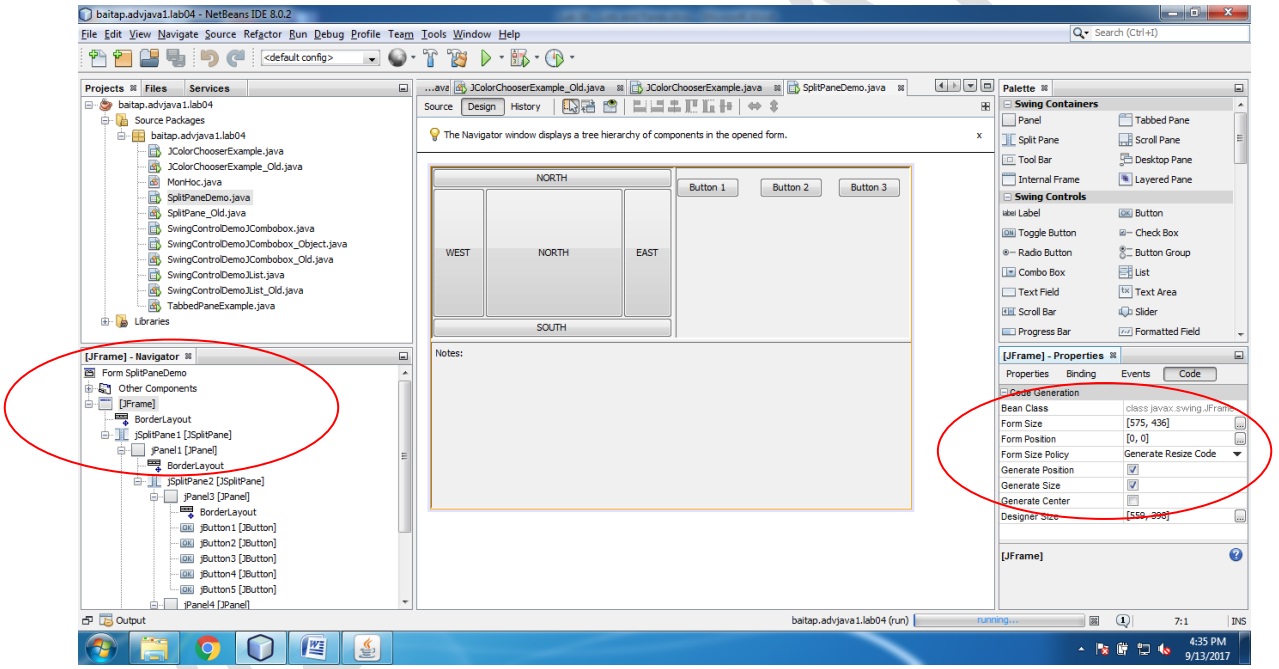
Bước 5: Ở góc phải trên của JSplit Pane.

- Đặt 3 JButton vào và đặt nội dung hiển thị "Button 1", "Button 2", "Button 3".

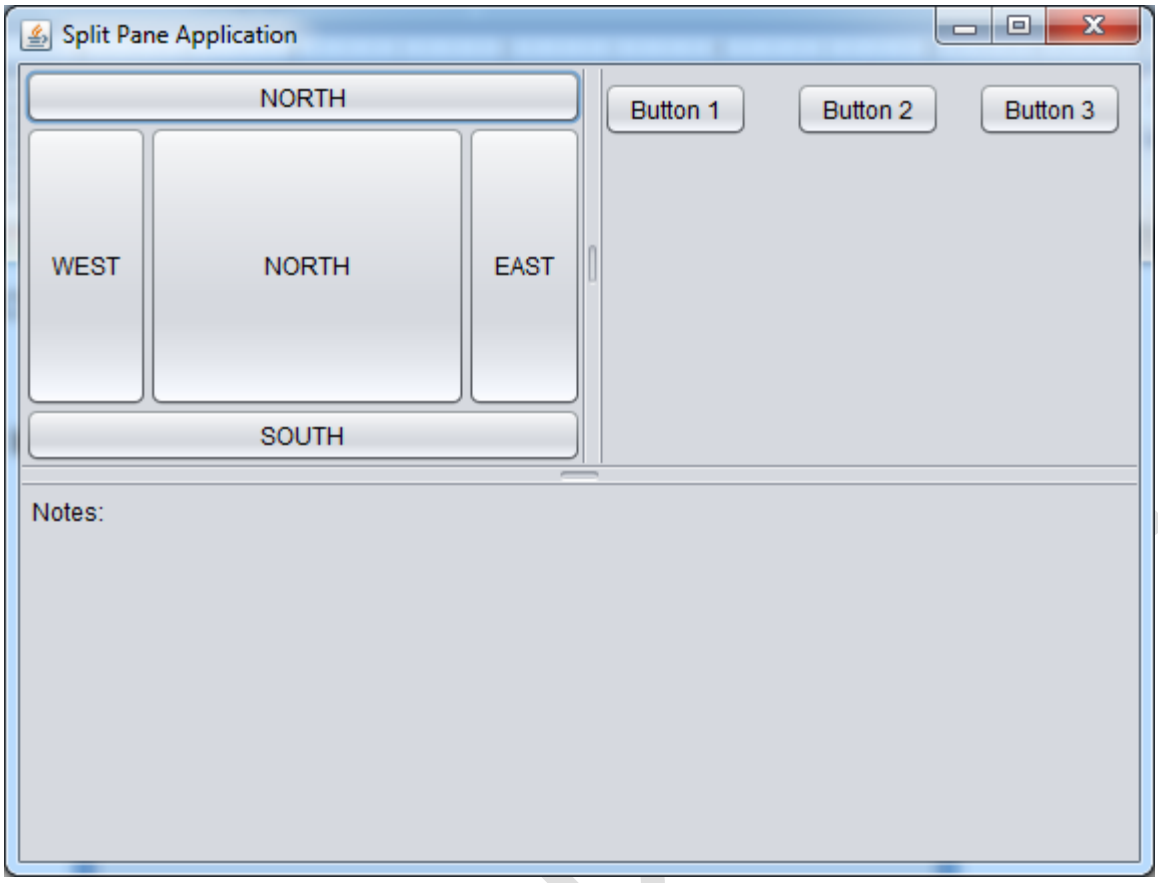
Bước 6: Ở SPlit Pane phía dưới

- Click chuột phải vào JPanel của nó rồi chọn "Set Layout" / "Flow Layout"
- Click chuột trái vào "FlowLayout" ở góc trái dưới của cửa sổ màn hình (ở mục "Navigator"), chọn "Alignment" là "LEFT".
- Kéo thả một JLabel vào thì nó sẽ được căn chỉnh sang phía bên trái, sửa chữ hiển thị thành "Notes".

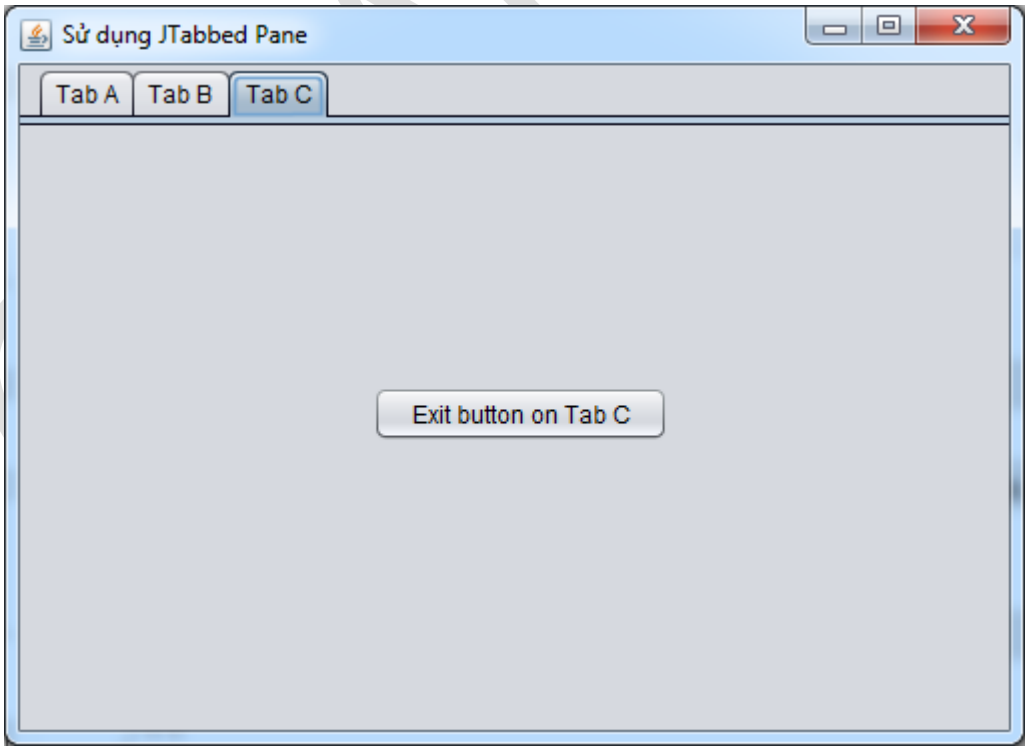
Bước 7: Để JFrame Form không thay đổi kích thước khi hiển thị lên, click chuột vào JFrame (ở cửa sổ Navigator trong giao diện Design) rồi chọn sang tab "code" trong cửa sổ Properties, sau đó chọn "Form Size Policy" đổi thành "Generate Resize Code".



Bước 8: Chạy thử và xem kết quả



Bài thực hành 5: Viết ứng dụng sử dụng JTabbedPane như hình.



Bước 1: Tạo một JFrame Form đặt tên "TabbedPaneExample"

- Thay đổi kích thước form cho phù hợp

- Đặt tiêu đề form là "Sử dụng JTabbed Pane"
- Click chuột phải vào form chọn "Set Layout" / "Border Layout".

Bước 2: Gắp một JTabbed Pane đặt vào form

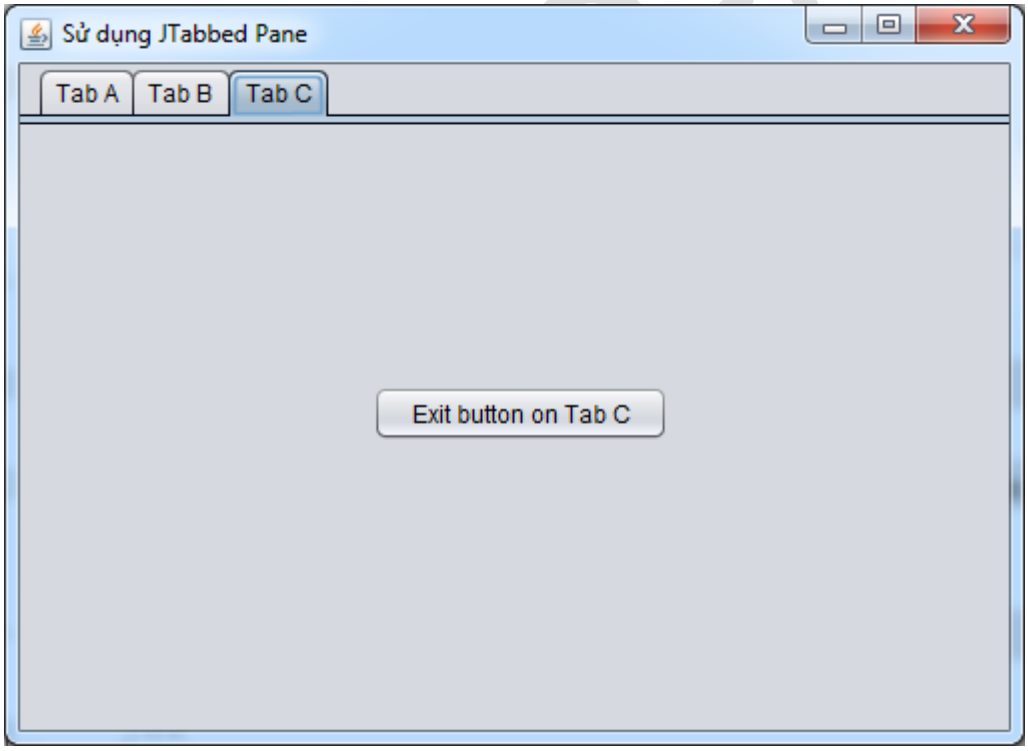
- Gắp một JPanel đặt vào form tiếp theo thì nó sẽ được đặt thành các tab của JTabbed Pane, ấn phím F2 và đổi tên tab này thành "Tab A"
- Tiếp tục đặt các JPanel khác vào thành các tab "Tab B", "Tab C". Khi đặt vào phải lựa sao cho đường khung của JPanel to bằng form thì nó mới tạo thành tab được.

Bước 3: Đặt một JButton vào "Tab C" và đặt tên hiển thị "Exit button on Tab C".

- Nháy đúp vào button này và viết code sau:

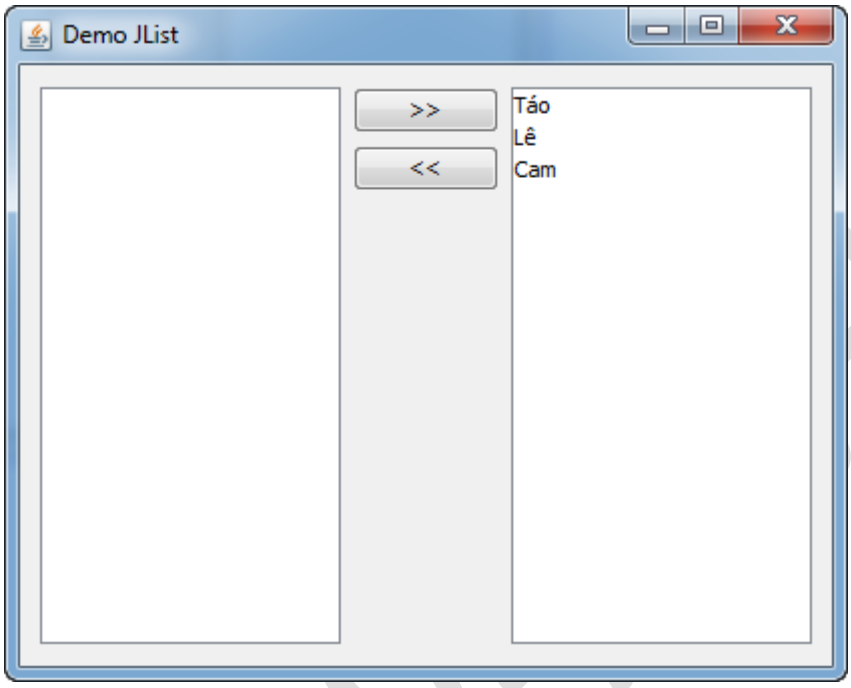
```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dispose();
}
```

Bước 4: Chạy thử chương trình và xem kết quả.

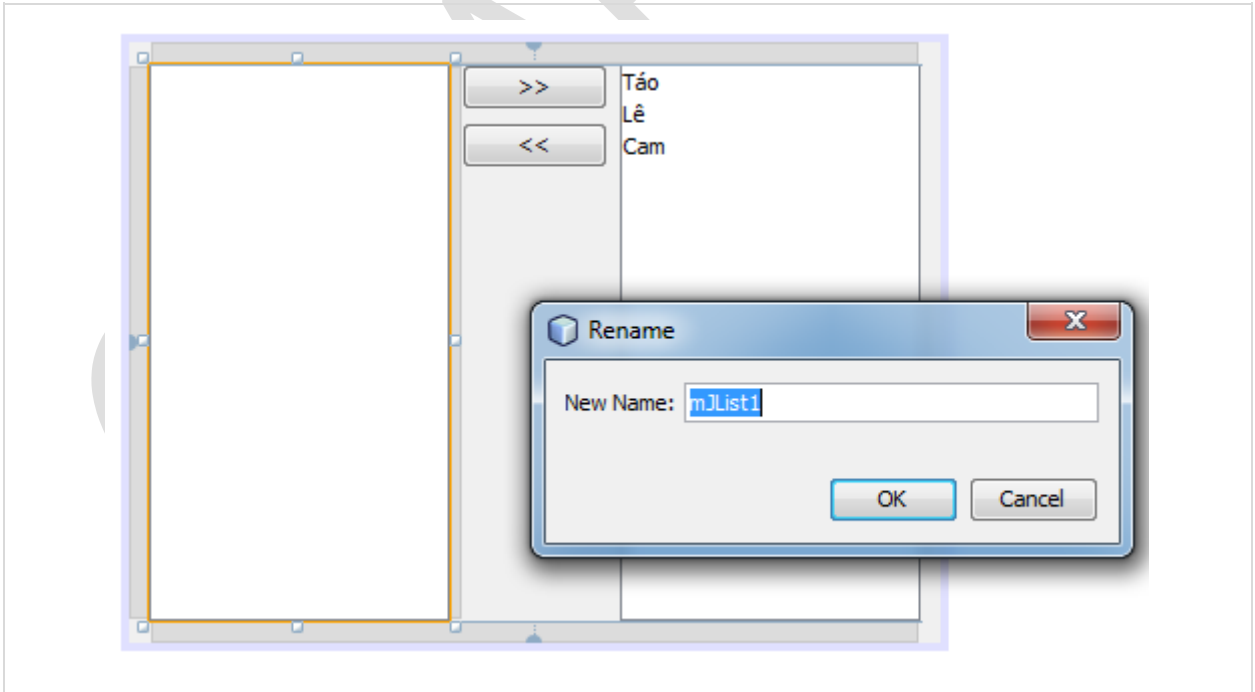


*** Lưu ý:** Chúng ta có thể trình bày các form hoặc bảng hiển thị trong các tab của form này giống như khi chúng ta tạo form hoặc bảng hiển thị trong các form như trước đây.

Bài thực hành 6: Tạo Frame trong đó có chứa hai JList như hình, yêu cầu khi click vào nút >> sẽ chuyển các item (đang chọn) từ List bên trái sang bên phải, khi click nút << thì chuyển item (đang chọn từ List bên phải về bên trái).



Bước 1: Dựng giao diện như yêu cầu, đặt tên biến theo chuẩn.



Bước 2: Khai báo 2 Model cho 2 Jlist, viết hàm khởi tạo dữ liệu cho model và gán cho JList.

```
import javax.swing.ListSelectionModel;
```

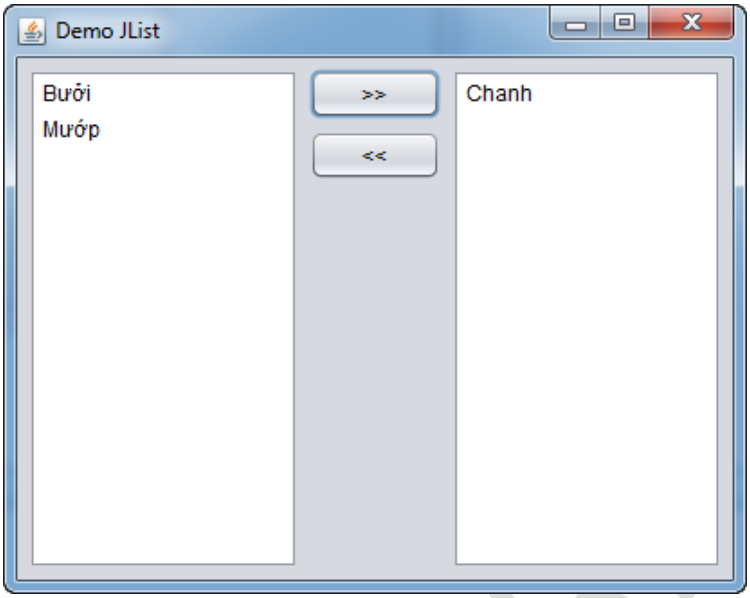
```
/**
 *
 * @authorminhvt
*/
public class DemoJList extends javax.swing.JFrame {

    DefaultListModel<String> model1, model2;

/**
     * CreatesnewformDemoJList
*/
public DemoJList() {
        initComponents();
        initJListFruit();
    }

private void initJListFruit() {
model1 = new DefaultListModel<>();
model2 = new DefaultListModel<>();

model1.addElement("Bưởi");
model1.addElement("Chanh");
model1.addElement("Mướp");

mJList1.setModel(model1);
mJList1.setSelectedIndex(1);
mJList1.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

mJList2.setModel(model2);
    }
.....
```

Bước 3: Viết hàm xử lý sự kiện chuyển item trên List như yêu cầu.

```
private void jButton1ActionPerformed(java.awt.event.ActionEventevt){
        String value = mJList1.getSelectedValue();
model2.addElement(value);                    // Thêm và List bên trái
model1.remove(mJList1.getSelectedIndex());  // Xóa item List bên phải
}

private void jButton2ActionPerformed(java.awt.event.ActionEventevt){
int selected[] = mJList2.getSelectedIndices(); // Lấy về các item được chọn
for (int i = 0; i < selected.length; i++) {
            String valueRemove = model2.remove(selected[i]);
model1.addElement(valueRemove);
        }
```

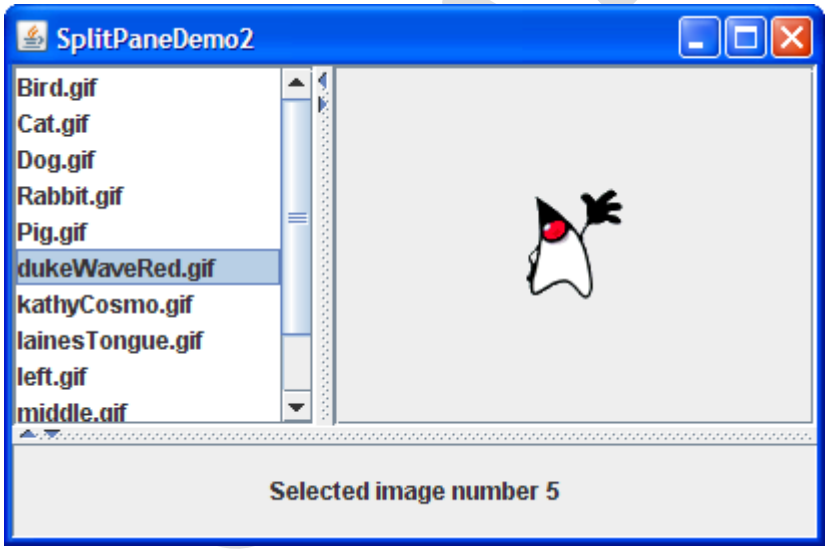
```
    }
```

Bước 4: Xem kết quả.

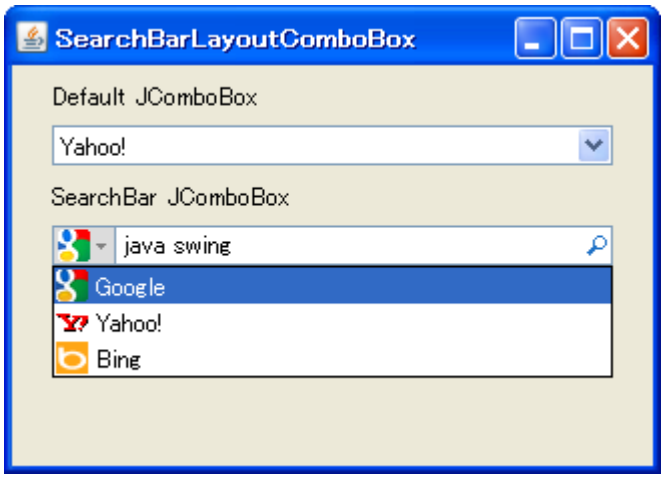


Phần II - Bài tập tự làm

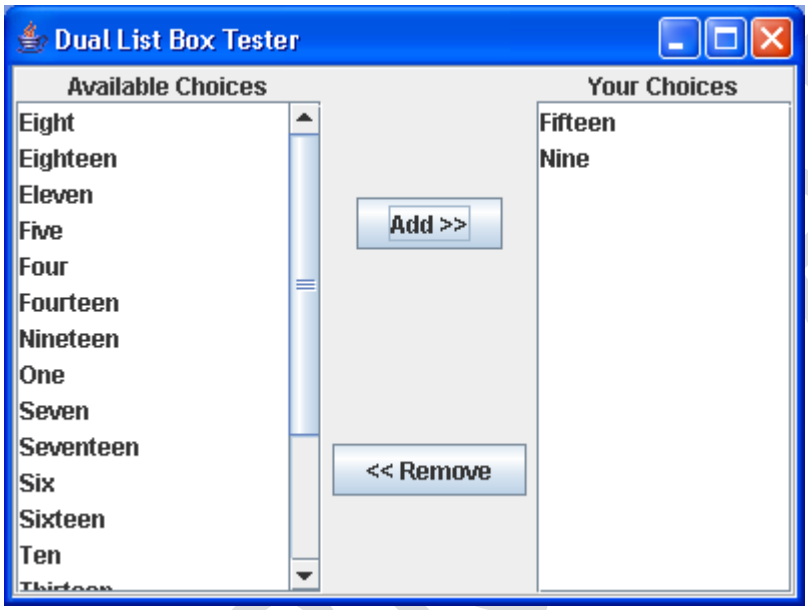
1. Tạo ứng dụng sử dụng Split Pane như hình:



2. Tạo ứng dụng sử dụng ComboBox như hình: (xem lại bài tập này, được thêm gợi ý)



3. Tạo ứng dụng sử dụng JList như hình:



Khi ấn add sẽ thêm item (chọn 1 hoặc nhiều) được lựa chọn ở bên trái và thêm sang bên phải, khi bấm chọn bên phải rồi nhấn Remove thì sẽ xóa.