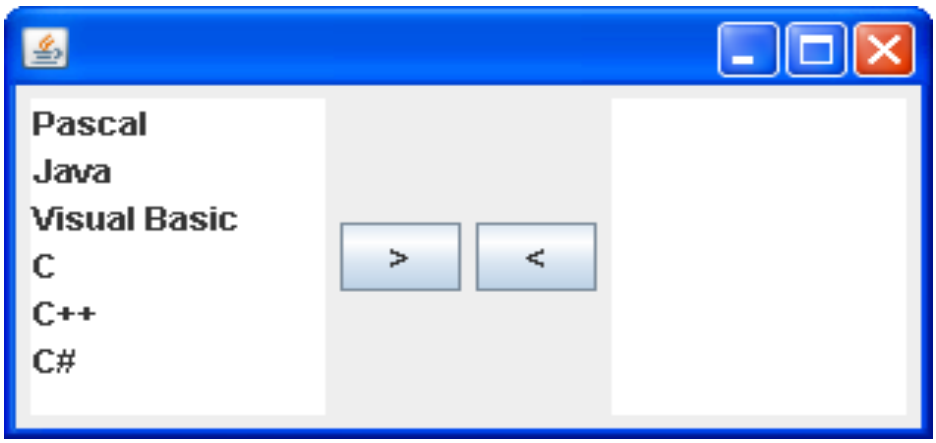
MỤC TIÊU

- ✓ Nắm vững kỹ thuật gửi email qua gmail
- ✓ Biết cách queue email để tránh nghẽn

BÀI 1: Thiết kế giao diện như hình dưới



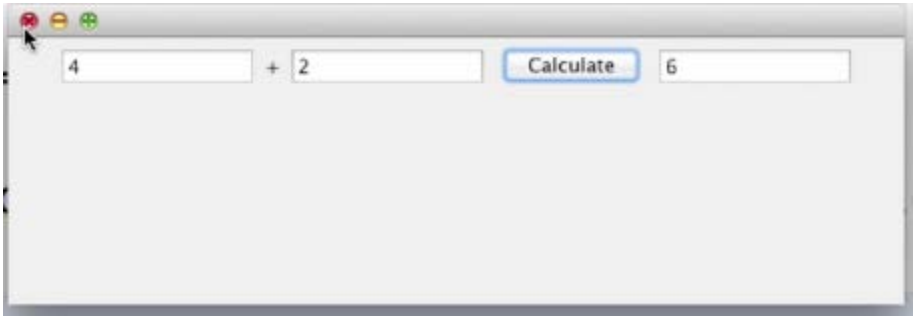
- Model: Mô hình dữ liệu cho JList
- View: Giao diện hiển thị các mục được chọn
- Controller: Xử lý sự kiện khi một nút được nhấn

Hướng dẫn thực hiện

```
1  package mvc;
2  import java.awt.*;
3  import java.awt.event.*;
4  import javax.swing.*;
5  public class JListExample implements ActionListener {
6      private JList lst1,lst2;
7      private DefaultListModel dt1,dt2;
8      private JButton bt1,bt2;
9      public JListExample(){
10         JFrame jf = new JFrame();
11         jf.getContentPane().setLayout(new FlowLayout());
12         String dsMonHoc[]= {"Pascal","Java","Visual
13                                                 Basic","C","C++","C#"};
14         dt1 = new DefaultListModel();
15         dt2 = new DefaultListModel();
16         for (int i =0;i<dsMonHoc.length;i++){
17             dt1.addElement(dsMonHoc[i]);
18         }
19     lst1 = new JList(dt1);
20         lst2 = new JList(dt2);
21         bt1 = new JButton(">");
22         bt1.addActionListener(this);
23         bt2 = new JButton("<");
24         bt2.addActionListener(this);
25         jf.getContentPane().add(lst1);
26         jf.getContentPane().add(bt1);
27         jf.getContentPane().add(bt2);
28         jf.getContentPane().add(lst2);
29         jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
30         jf.setVisible(true);
31         lst1.setPreferredSize(new Dimension(100,120));
32         lst2.setPreferredSize(new Dimension(100,120));
33         jf.pack();
34     }
```

```
35 // Phuong thuc xu ly su kien khi mot nut duoc nhan
36 public void actionPerformed(ActionEvent e) {
37     if (e.getSource()==bt1){
38         Object s1 = lst1.getSelectedValue();
39         dt2.addElement(s1);
40         dt1.removeElement(s1);
41     }else{
42         Object s2 = lst2.getSelectedValue();
43         dt1.addElement(s2);
44         dt2.removeElement(s2);
45     }
46
47 }
48 public static void main(String[] args) {
49     new JListExample();
50 }
51
52 }
```

Bài 2 : Lập trình một máy tính đơn giản như sau



Sử dụng mô hình MVC

Hướng dẫn thực hiện

CalculatorModel.java

```
01 // The Model performs all the calculations needed
02 // and that is it. It doesn't know the View
03 // exists
04
05 public class CalculatorModel {
06
07     // Holds the value of the sum of the numbers
08     // entered in the view
09
10     private int calculationValue;
11
12     public void addTwoNumbers(int firstNumber, int secondNumber){
13
14         calculationValue = firstNumber + secondNumber;
15
16     }
17
18     public int getCalculationValue(){
19
20         return calculationValue;
21
22     }
23
24 }
```

CalculatorView.java

```java
// This is the View
// Its only job is to display what the user sees
// It performs no calculations, but instead passes
// information entered by the user to whomever needs
// it.

import java.awt.event.ActionListener;

import javax.swing.*;

public class CalculatorView extends JFrame{

    private JTextField firstNumber  = new JTextField(10);
    private JLabel additionLabel = new JLabel("+");
    private JTextField secondNumber = new JTextField(10);
    private JButton calculateButton = new JButton("Calculate");
    private JTextField calcSolution = new JTextField(10);

    CalculatorView(){

        // Sets up the view and adds the components

        JPanel calcPanel = new JPanel();

        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setSize(600, 200);

        calcPanel.add(firstNumber);
        calcPanel.add(additionLabel);
        calcPanel.add(secondNumber);
        calcPanel.add(calculateButton);
        calcPanel.add(calcSolution);

        this.add(calcPanel);

        // End of setting up the components --------

    }

    public int getFirstNumber(){

        return Integer.parseInt(firstNumber.getText());

    }

    public int getSecondNumber(){

        return Integer.parseInt(secondNumber.getText());

    }
```

```
50      }
51
52      public int getCalcSolution(){
53
54          return Integer.parseInt(calcSolution.getText());
55
56      }
57
58      public void setCalcSolution(int solution){
59
60          calcSolution.setText(Integer.toString(solution));
61
62      }
63
64      // If the calculateButton is clicked execute a method
65      // in the Controller named actionPerformed
66
67      void addCalculateListener(ActionListener listenForCalcButton){
68
69          calculateButton.addActionListener(listenForCalcButton);
70
71      }
72
73      // Open a popup that contains the error message passed
74
75      void displayErrorMessage(String errorMessage){
76
77          JOptionPane.showMessageDialog(this, errorMessage);
78
79      }
80
81  }
```

CalculatorController.java

```java
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

// The Controller coordinates interactions
// between the View and Model

public class CalculatorController {

    private CalculatorView theView;
    private CalculatorModel theModel;

    public CalculatorController(CalculatorView theView, CalculatorModel theModel) {
        this.theView = theView;
        this.theModel = theModel;

        // Tell the View that when ever the calculate button
        // is clicked to execute the actionPerformed method
        // in the CalculateListener inner class

        this.theView.addCalculateListener(new CalculateListener());
    }

    class CalculateListener implements ActionListener{

        public void actionPerformed(ActionEvent e) {

            int firstNumber, secondNumber = 0;

            // Surround interactions with the view with
            // a try block in case numbers weren't
            // properly entered

            try{

                firstNumber = theView.getFirstNumber();
                secondNumber = theView.getSecondNumber();

                theModel.addTwoNumbers(firstNumber, secondNumber);

                theView.setCalcSolution(theModel.getCalculationValue());

            }

            catch(NumberFormatException ex){

                System.out.println(ex);

                theView.displayErrorMessage("You Need to Enter 2 Integers");

            }

        }

    }
```

MVCCalculator.java

```java
public class MVCCalculator {

    public static void main(String[] args) {

        CalculatorView theView = new CalculatorView();

        CalculatorModel theModel = new CalculatorModel();

        CalculatorController theController = new CalculatorController(theView,theModel);

        theView.setVisible(true);

    }
}
```

BAI 3: Giáo viên cho thêm