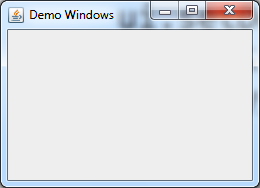
**MODULE** **- SWING**

**Mục đích:**

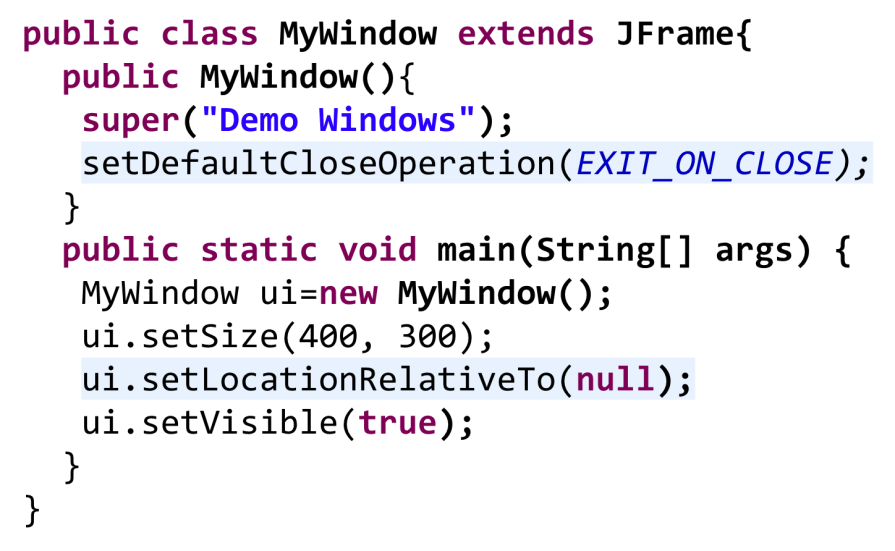
Ôn tập Graphic User Interface,giúp các sinh viên hiểu được LayoutManager, Common Control, Event, DialogBox, Advanced Control. Sinh viên phải thực hiện tốt Module 1 để ứng dụng cho các Module tiếp theo, đặc biệt là phần kết nối cơ sở dữ liệu.

**Bài tập 1:** Thực hành cách hiển thị cửa sổ Windows trong Java

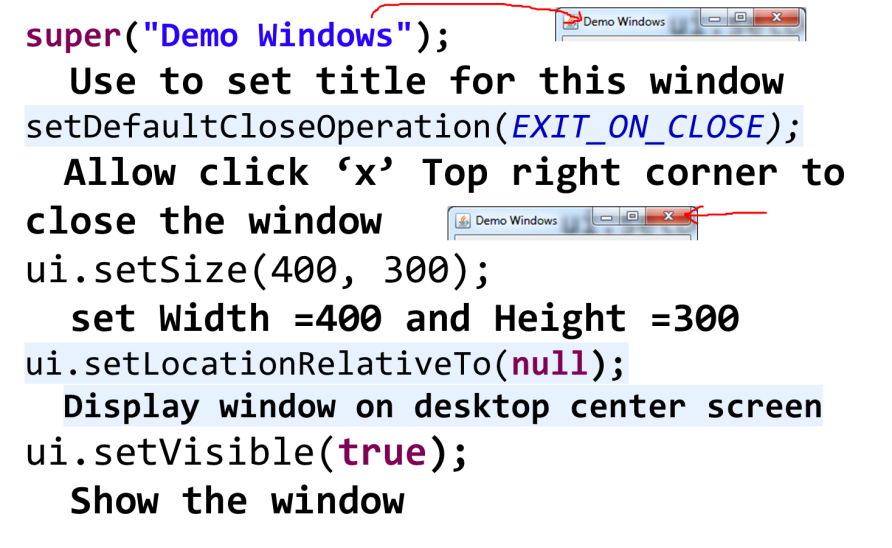


Hãy hiển thị cửa sổ trên, yêu cầu viết class kế thừa từ JFrame

**Hướng dẫn:**

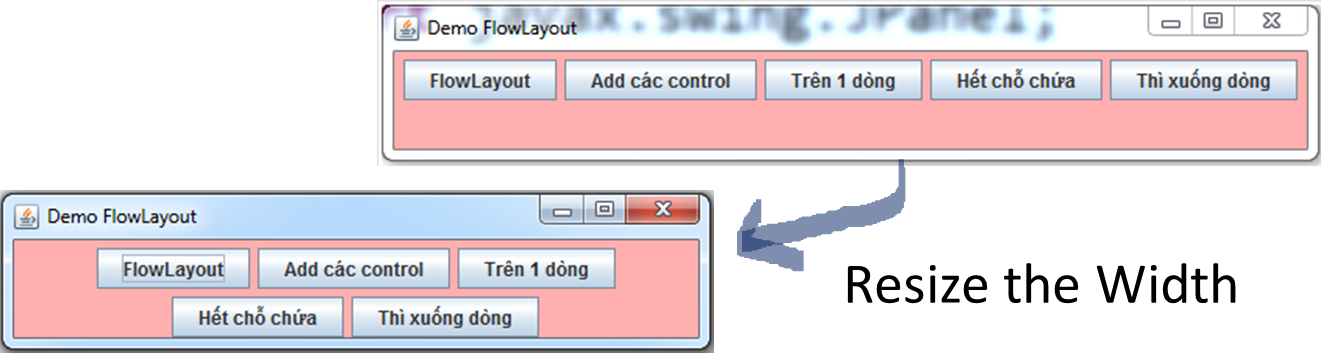


**Giải thích:**

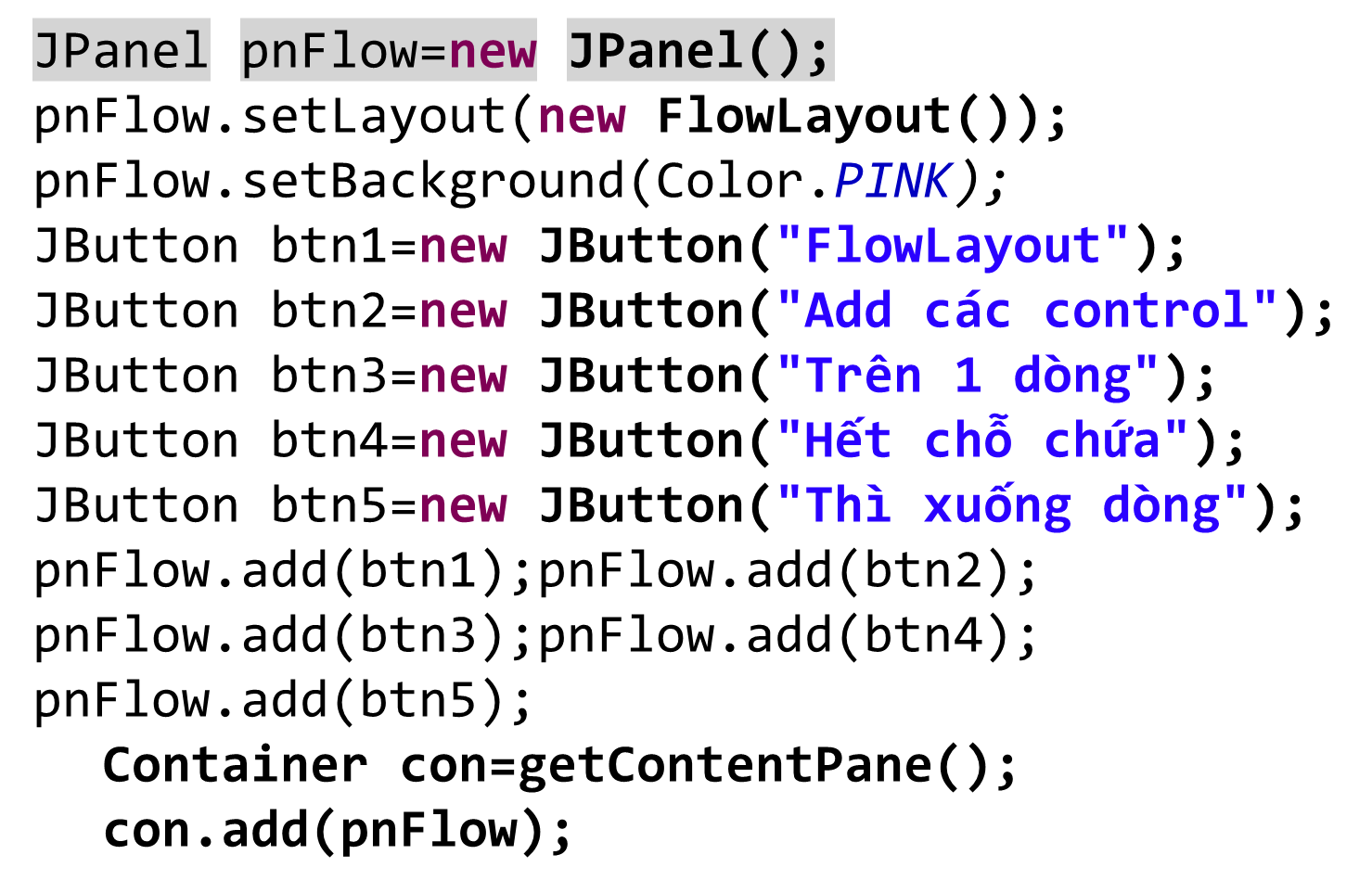
****

**Bài tập 2: Thực hành về FlowLayout**

FlowLayout cho phép add các control trên cùng một dòng, khi nào hết chỗ chứa nó sẽ tự động xuống dòng, ta cũng có thể điều chỉnh hướng xuất hiện của control. Mặc định khi một JPanel được khởi tạo thì bản thân lớp chứa này sẽ có kiểu Layout là FlowLayout.

****

**Hướng dẫn:**

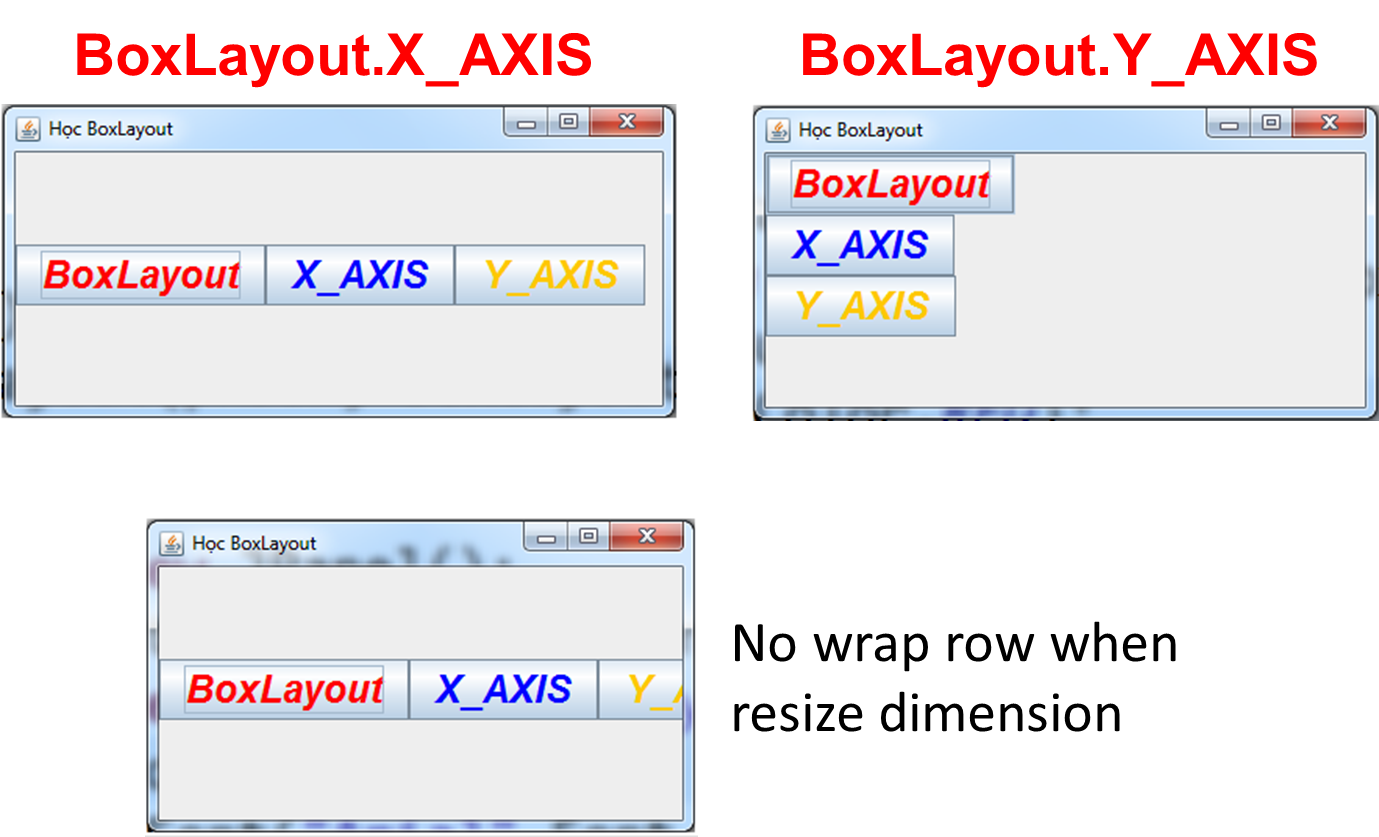
****

**Bài tập 3:** Thực hành về **BoxLayout**

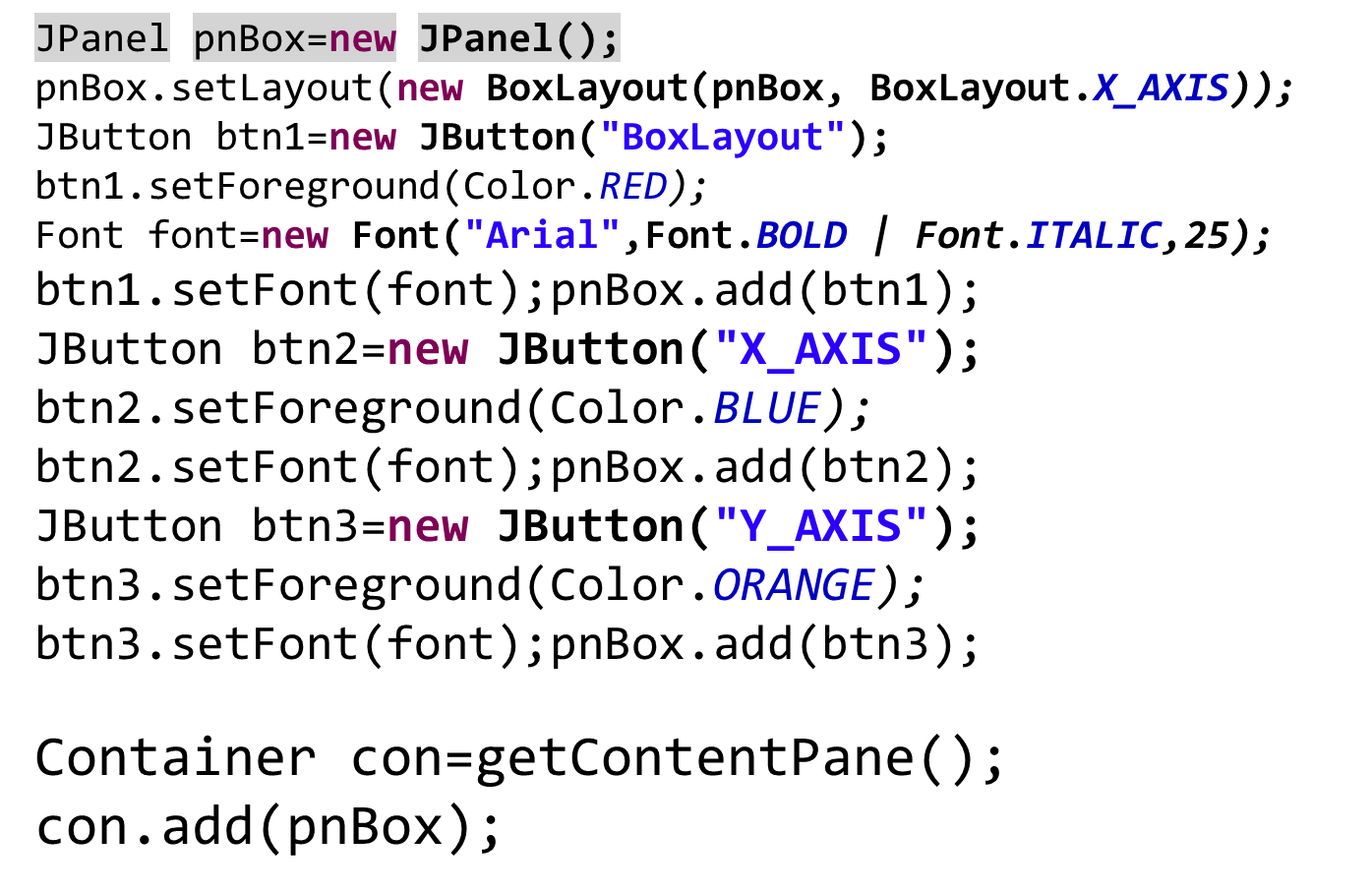
BoxLayout cho phép add các control theo dòng hoặc cột, tại mỗi vị trí add nó chỉ chấp nhận 1 control, do đó muốn xuất hiện nhiều control tại một vị trí thì bạn nên add vị trí đó là 1 JPanel rồi sau đó add các control khác vào JPanel này.

**BoxLayout.X\_AXIS :** Cho phép add các control theo hướng từ trái qua phải.

**BoxLayout.Y\_AXIS** **:** Cho phép add các control theo hướng từ trên xuống dưới.

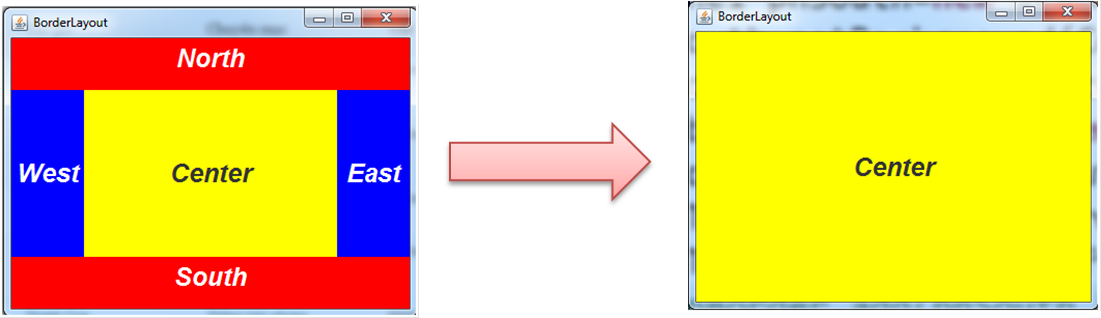
**BoxLayout** sẽ không tự động xuống dòng khi hết chỗ chứa, tức là các control sẽ bị che khuất nếu như thiếu không gian chứa nó.****

**Hướng dẫn:**

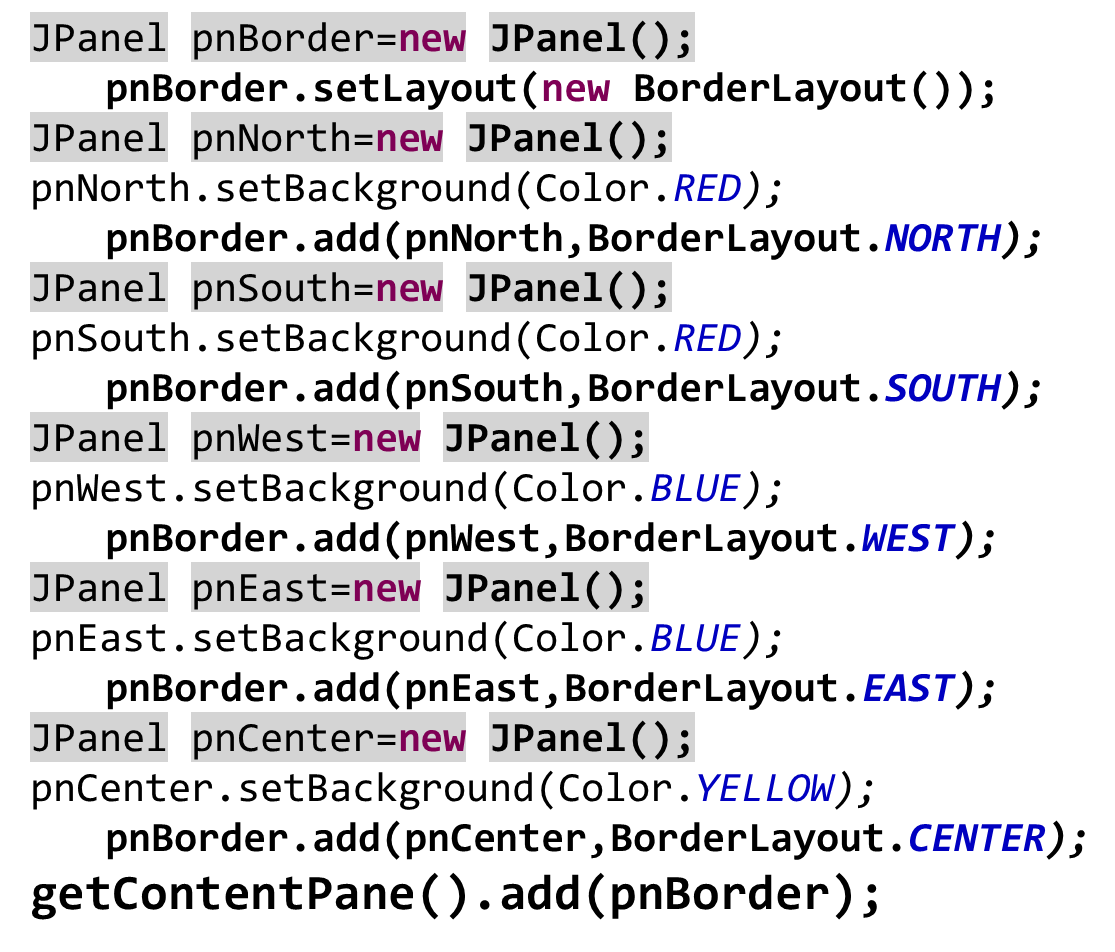
****

**Bài tập 4:** Thực hành về **BorderLayout**

BorderLayout giúp chúng ta hiển thị các control theo 5 vùng: North, South, West, East, Center

****

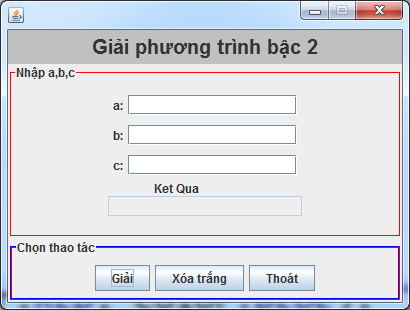
Nếu như không có 4 vùng : North, West, South, East. Thì vùng Center sẽ tràn đầy cửa sổ, thông thường khi đưa các control JTable, JTree, ListView, JScrollpane… ta thường đưa vào vùng Center để nó có thể tự co giãn theo kích thước cửa sổ giúp giao diện đẹp hơn.



**Bài tập 5:** Thực hành về các control căn bản

* **JButton**
* **JLabel**
* **JTextField**
* **JTextArea**
* **ButtonGroup & JRadioButton**
* **JCheckBox**
* **JComboBox**
* **JList**

Thiết kế giao diện để giải phương trình bậc 2:

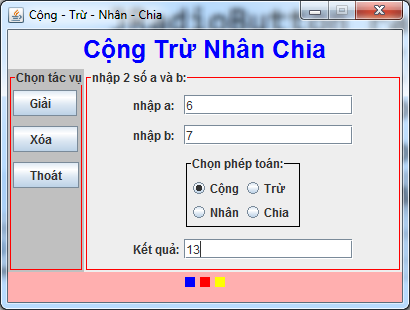


Hướng dẫn: Sinh viên phải xác định Layout Manager trước, ta cũng có thể kế hợp các Layout để thiết kế giao diện, đặt tên control theo yêu cầu bên dưới

|  |  |  |
| --- | --- | --- |
| Tên Control | Tên Biến Control | Mô tả |
| JTextField | txtSoa | Dùng để nhập giá trị cho a |
| JTextField | txtSob | Dùng để nhập giá trị cho b |
| JTextField | txtSoc | Dùng để nhập giá trị cho c |
| JTextField | txtKetqua | Dùng để hiển thị kết quả |
| JButton | btnGiai | Viết lệnh để giải phương trình |
| JButton | btnXoaTrang | Xóa toàn bộ dữ liệu trong ô dl |
| JButton | btnThoat | Viết lệnh thoát chương trình |
| JLabel | lblTieuDe | Giải Phương Trình Bậc 2 |

**Bài tập 6:** thiết kế giao diện để thực hiện các phép toán : **‘+’ ‘-’ ‘\*’ ‘:’**

Thiết kế giao diện như hình bên dưới:



Khi bấm nút Giải thì tùy thuộc vào phép toán được chọn mà kết quả thực hiện khác nhau.

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** javax.swing.border.\*;

**public** **class** CongTruNhanChiaUI **extends** JFrame {

**private** **static** **final** **long** *serialVersionUID* = 1L;

**public** CongTruNhanChiaUI(String title)

{

setTitle(title);

}

**public** **void** doShow()

{

setSize(400, 300);

setLocationRelativeTo(**null**);

setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);

addControl();

setResizable(**false**);

setVisible(**true**);

}

**public** **void** addControl()

{

JPanel pnBorder=**new** JPanel();

pnBorder.setLayout(**new** BorderLayout());

JPanel pnNorth=**new** JPanel();

JLabel lblTitle=**new** JLabel("Cộng Trừ Nhân Chia");

pnNorth.add(lblTitle);

pnBorder.add(pnNorth,BorderLayout.*NORTH*);

lblTitle.setForeground(Color.*BLUE*);

Font ft=**new** Font("arial", Font.*BOLD*, 25);

lblTitle.setFont(ft);

JPanel pnWest=**new** JPanel();

pnWest.setLayout(**new** BoxLayout(pnWest, BoxLayout.*Y\_AXIS*));

JButton btnGiai=**new** JButton("Giải ");

JButton btnXoa=**new** JButton("Xóa ");

JButton btnThoat=**new** JButton("Thoát");

pnWest.add(btnGiai);

pnWest.add(Box.*createVerticalStrut*(10));

pnWest.add(btnXoa);

pnWest.add(Box.*createVerticalStrut*(10));

pnWest.add(btnThoat);

pnBorder.add(pnWest,BorderLayout.*WEST*);

pnWest.setBackground(Color.*LIGHT\_GRAY*);

Border southborder

=BorderFactory.*createLineBorder*(Color.*RED*);

TitledBorder southTitleBorder=

**new** TitledBorder(southborder, "Chọn tác vụ");

pnWest.setBorder(southTitleBorder);

JPanel pnSouth=**new** JPanel();

pnSouth.setPreferredSize(**new** Dimension(0, 30));

pnSouth.setBackground(Color.*PINK*);

JPanel pns1=**new** JPanel();

pns1.setBackground(Color.*BLUE*);

pnSouth.add(pns1);

JPanel pns2=**new** JPanel();

pns2.setBackground(Color.*RED*);

pnSouth.add(pns2);

JPanel pns3=**new** JPanel();

pns3.setBackground(Color.*YELLOW*);

pnSouth.add(pns3);

pnBorder.add(pnSouth,BorderLayout.*SOUTH*);

JPanel pnCenter=**new** JPanel();

pnCenter.setLayout(**new** BoxLayout(pnCenter, BoxLayout.*Y\_AXIS*));

pnBorder.add(pnCenter,BorderLayout.*CENTER*);

Border centerborder

=BorderFactory.*createLineBorder*(Color.*RED*);

TitledBorder centerTitleBorder=

**new** TitledBorder(centerborder, "nhập 2 số a và b:");

pnCenter.setBorder(centerTitleBorder);

JPanel pna=**new** JPanel();

JLabel lbla=**new** JLabel("nhập a:");

**final** JTextField txta=**new** JTextField(15);

pna.add(lbla);

pna.add(txta);

pnCenter.add(pna);

JPanel pnb=**new** JPanel();

JLabel lblb=**new** JLabel("nhập b:");

**final** JTextField txtb=**new** JTextField(15);

pnb.add(lblb);

pnb.add(txtb);

pnCenter.add(pnb);

JPanel pnc=**new** JPanel();

JPanel pnpheptoan=**new** JPanel();

pnpheptoan.setLayout(**new** GridLayout(2, 2));

pnpheptoan.setBorder(**new** TitledBorder(BorderFactory.*createLineBorder*(Color.*BLACK*),"Chọn phép toán:"));

**final** JRadioButton radCong=**new** JRadioButton("Cộng");

pnpheptoan.add(radCong);

**final** JRadioButton radTru=**new** JRadioButton("Trừ");

pnpheptoan.add(radTru);

**final** JRadioButton radNhan=**new** JRadioButton("Nhân");

pnpheptoan.add(radNhan);

**final** JRadioButton radChia=**new** JRadioButton("Chia");

pnpheptoan.add(radChia);

ButtonGroup group=**new** ButtonGroup();

group.add(radCong);group.add(radTru);

group.add(radNhan);group.add(radChia);

pnc.add(pnpheptoan);

pnCenter.add(pnc);

JPanel pnkq=**new** JPanel();

JLabel lblkq=**new** JLabel("Kết quả:");

**final** JTextField txtkq=**new** JTextField(15);

pnkq.add(lblkq);

pnkq.add(txtkq);

pnCenter.add(pnkq);

lbla.setPreferredSize(lblkq.getPreferredSize());

lblb.setPreferredSize(lblkq.getPreferredSize());

btnThoat.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

**int** ret=JOptionPane.*showConfirmDialog*(**null**, "Muốn thoát hả?", "Thoát", JOptionPane.*YES\_NO\_OPTION*);

**if**(ret==JOptionPane.*YES\_OPTION*)

System.*exit*(0);

}

});

btnXoa.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

txta.setText("");

txtb.setText("");

txtkq.setText("");

txta.requestFocus();

}

});

btnGiai.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

String sa=txta.getText();

**int** a=0,b=0;

**try**

{

a=Integer.*parseInt*(sa);

}

**catch**(Exception ex)

{

JOptionPane.*showMessageDialog*(**null**, "Nhập sai định dạng!");

txta.selectAll();

txta.requestFocus();

**return**;

}

String sb=txtb.getText();

**try**

{

b=Integer.*parseInt*(sb);

}

**catch**(Exception ex)

{

JOptionPane.*showMessageDialog*(**null**, "Nhập sai định dạng!");

txtb.selectAll();

txtb.requestFocus();

**return**;

}

**double** kq=0;

**if**(radCong.isSelected())

{

kq=(a+b);

}

**else** **if**(radTru.isSelected())

{

kq=(a-b);

}

**else** **if**(radNhan.isSelected())

{

kq=(a\*b);

}

**else**

{

kq=a\*1.0/b\*1.0;

}

txtkq.setText(kq+"");

}

});

Container con=getContentPane();

con.add(pnBorder);

}

**public** **static** **void** main(String[] args) {

CongTruNhanChiaUI ui=**new** CongTruNhanChiaUI("Cộng - Trừ - Nhân - Chia");

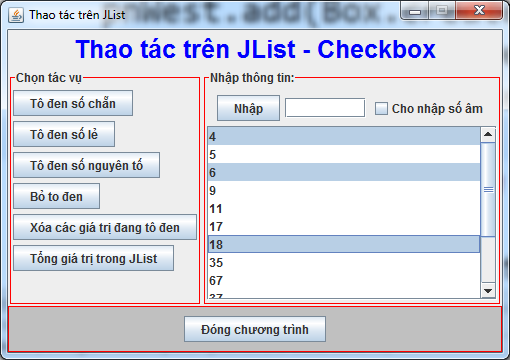
ui.doShow();

}

}

**Bài tập 7:** Thao tác trên JList – Jcheckbox

Thiết kế giao diện như hình bên dưới và thực hiện các thao tác theo yêu cầu:

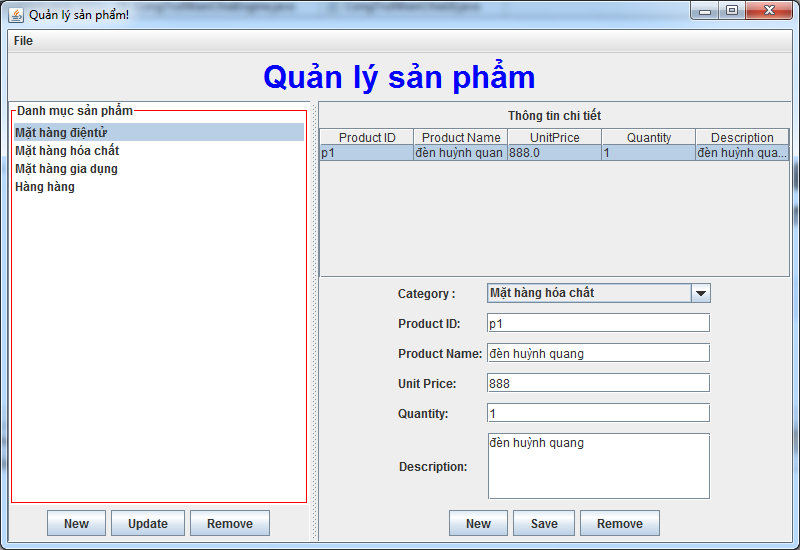


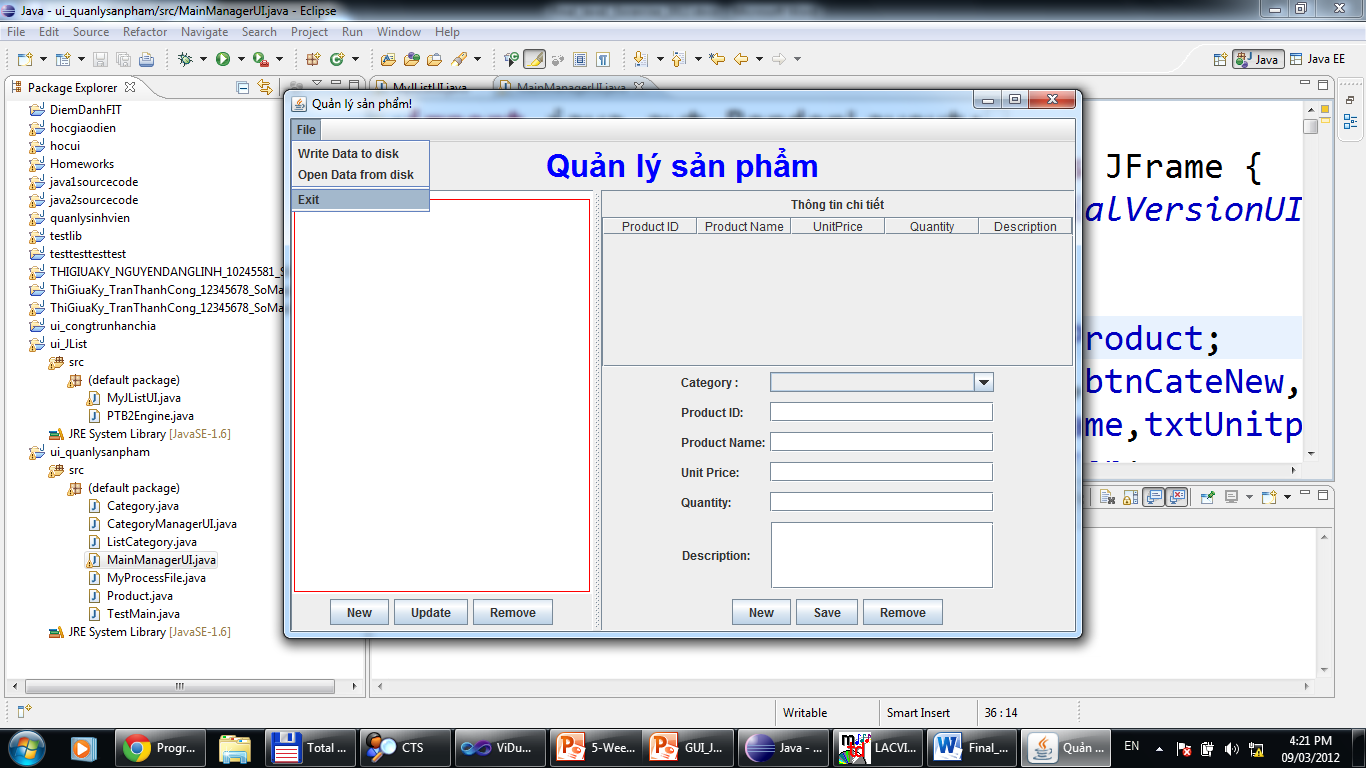
* Chương trình cho phép nhập vào các số nguyên từ giao diện trong phần nhập thông tin, Khi người sử nhập giá trị vào JTextField và click nút “Nhập” thì sẽ cập nhập dữ liệu xuống JList, Nếu checked vào “Cho nhập số âm” thì các số âm mới được phép đưa vào JList còn không thì thông báo lỗi.
* Ô Chọn tác vụ, sinh viên phải thực hiện toàn bộ các yêu cầu
* Nút Đóng chương trình: sẽ hiển thị thông báo hỏi người sử dụng có muốn đóng hay không.

**Bài tập 8:** Viết chương trình quản lý sản phẩm

Yêu cầu chức năng: Cho phép nhập/ xuất danh mục, danh sách sản phẩm

* Cho phép cập nhật thông tin
* Cho phép lưu / đọc danh mục sản phẩm
* Yêu cầu sử dụng JMenuBar, JList, JTable, JCombobox, …





Menu **Write Data to disk** dùng để lưu dữ liệu xuống ổ cứng

Menu **Open Data from disk** để đọc dữ liệ từ ổ cứng

Menu **Exit** dùng để thoát chương trình

**Cách lưu/ đọc đối tượng trên ổ cứng:**

* Tất cả các class phải **implements** **Serializable**:
  + public class **Sanpham** **implements** Serializable{…}
  + public class **DanhMucSanPham** **implements** Serializable{…}
* Viết một class **MyFile** có 2 phương thức:
  + Lưu đối tượng:

**public** **static** **void** luuDoiTuong(Object obj, String fileName)

{

**try**

{

FileOutputStream fOut=**new** FileOutputStream(fileName);

ObjectOutputStream out=**new** ObjectOutputStream(fOut);

out.writeObject(obj);

out.close();

}

**catch**(Exception ex)

{

ex.printStackTrace();

}

}

* + Đọc đối tượng

**public** **static** Object docDoiTuong(String fileName)

{

**try**

{

FileInputStream fIn=**new** FileInputStream(fileName);

ObjectInputStream in=**new** ObjectInputStream(fIn);

Object o=in.readObject();

in.close();

**return** o;

}

**catch**(Exception ex)

{

ex.printStackTrace();

}

**return** **null**;

}

* Trong testMain:

DanhMucSanPham dsDienTu=(DanhMucSanPham )MyFile.*docDoiTuong*("luuluu.data");

**if**(dsDienTu!=**null**)

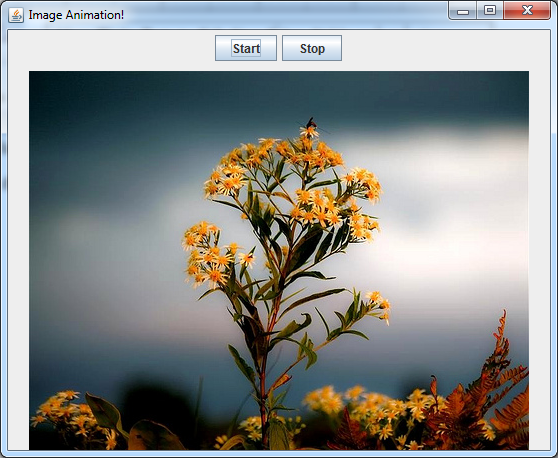
System.*out*.println(dsDienTu);

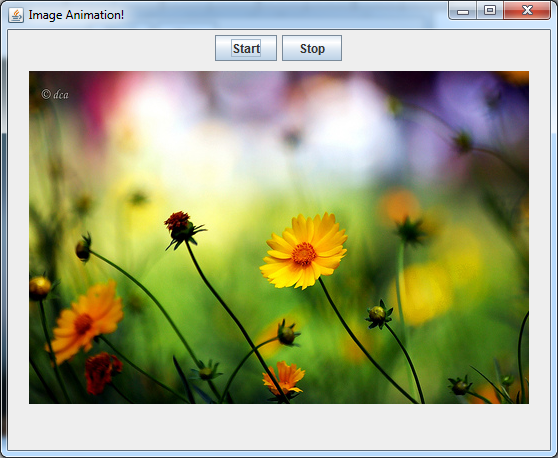
MyFile.*luuDoiTuong*(dsDienTu, "luuluu.data");

**Bài tập 9:** Thực hành về Timer class (\*)

Dùng class Timer để thiết kế ứng dụng **ImageAnimation**.

Giao diện sẽ có 2 JButton: Start và Stop. Khi bấm Start chương trình sẽ hiển thị hình ảnh tuần tự trong mảng 10 hình ảnh có sẵn. Bấm Stop để tạm dừng duyệt hình ảnh. Xem hình yêu cầu





**Hướng dẫn:** Dùng CardLayout và Timer

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**public** **class** ImageAnimation **extends** JFrame{

**private** **static** **final** **long** *serialVersionUID* = 1L;

Timer timer;

**private** **int** pos=1;

**public** ImageAnimation(String title)

{

**super**(title);

timer=**new** Timer(500, **null**);

}

**public** **void** doShow()

{

setSize(500,550);

setLocationRelativeTo(**null**);

setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);

addControl();

setVisible(**true**);

}

**public** **void** addControl()

{

JPanel pnBorder=**new** JPanel();

pnBorder.setLayout(**new** BorderLayout());

JPanel pnNorth=**new** JPanel();

JButton btnStart=**new** JButton("Start");

JButton btnStop=**new** JButton("Stop");

pnNorth.add(btnStart);

pnNorth.add(btnStop);

pnBorder.add(pnNorth,BorderLayout.*NORTH*);

**final** JPanel pnCenter=**new** JPanel();

pnCenter.setLayout(**new** CardLayout());

pnBorder.add(pnCenter,BorderLayout.*CENTER*);

pnCenter.setBackground(Color.*RED*);

JPanel []pnArr=**new** JPanel[10];

addImage(pnCenter,pnArr);

showImage(pnCenter,"card1");

btnStart.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

timer.start();

timer.addActionListener(**new** TimerPanel(pnCenter));

}

});

btnStop.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

timer.stop();

}

});

Container con=getContentPane();

con.add(pnBorder);

}

**private** **void** addImage(JPanel pnCenter,JPanel []pnArr)

{

**for**(**int** i=0;i<pnArr.length;i++)

{

pnArr[i]=**new** JPanel();

JLabel lbl=**new** JLabel();

ImageIcon icon=**new** ImageIcon("E:\\hoa\\"+i+".jpg");

lbl.setIcon(icon);

pnArr[i].add(lbl);

pnCenter.add(pnArr[i],"card"+i);

}

}

**public** **void** showImage(JPanel pn,String cardName)

{

CardLayout cl=(CardLayout)pn.getLayout();

cl.show(pn, cardName);

}

**private** **class** TimerPanel **implements** ActionListener

{

JPanel pn=**null**;

**public** TimerPanel(JPanel pn) {

**this**.pn=pn;

}

**public** **void** actionPerformed(ActionEvent arg0) {

showImage(pn,"card"+pos);

pos++;

**if**(pos>=10)

pos=1;

}

}

**public** **static** **void** main(String[] args) {;

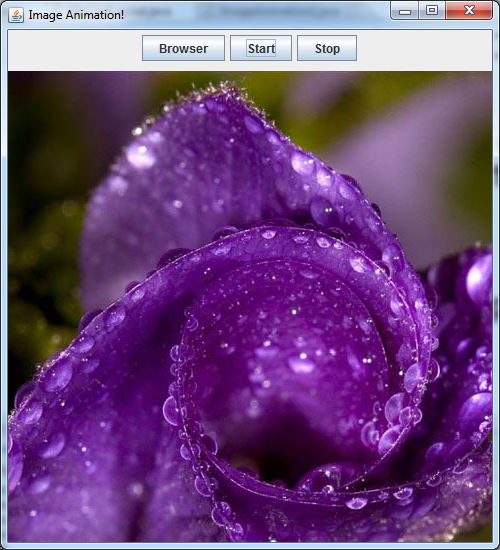
ImageAnimation imgUi=**new** ImageAnimation("Image Animation!");

imgUi.doShow();

}

}

**Bài tập 10:** Cải tiến bài tập 10. Chương trình sẽ cho phép đọc danh sách các hình ảnh bất kỳ trong ổ đĩa. (\*)



**Hướng dẫn:**

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.io.File;

**import** javax.swing.\*;

**public** **class** ImageAnimation2 **extends** JFrame{

**private** **static** **final** **long** *serialVersionUID* = 1L;

Timer timer;

**private** **int** pos=0;

**public** ImageAnimation2(String title)

{

**super**(title);

timer=**new** Timer(500, **null**);

}

**public** **void** doShow()

{

setSize(500,550);

setLocationRelativeTo(**null**);

setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);

addControl();

setVisible(**true**);

}

**public** **void** addControl()

{

JPanel pnBorder=**new** JPanel();

pnBorder.setLayout(**new** BorderLayout());

JPanel pnNorth=**new** JPanel();

JButton btnBrowser=**new** JButton("Browser");

JButton btnStart=**new** JButton("Start");

JButton btnStop=**new** JButton("Stop");

pnNorth.add(btnBrowser);

pnNorth.add(btnStart);

pnNorth.add(btnStop);

pnBorder.add(pnNorth,BorderLayout.*NORTH*);

**final** JPanel pnCenter=**new** JPanel();

pnCenter.setLayout(**new** CardLayout());

pnBorder.add(pnCenter,BorderLayout.*CENTER*);

pnCenter.setBackground(Color.*RED*);

btnStart.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

timer.start();

timer.addActionListener(**new** TimerPanel(pnCenter));

}

});

btnStop.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

timer.stop();

}

});

btnBrowser.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

JFileChooser jfc=**new** JFileChooser();

jfc.setMultiSelectionEnabled(**true**);

**if**(jfc.showOpenDialog(**null**)==JFileChooser.*APPROVE\_OPTION*)

{

File []files=jfc.getSelectedFiles();

**for**(**int** i=0;i< files.length;i++)

{

File f=files[i];

ImageIcon icon=**new** ImageIcon(f.getPath());

JPanel pn=**new** JPanel();

JLabel lbl=**new** JLabel(icon);

pn.add(lbl);

pnCenter.add(pn,"card"+i);

}

showImage(pnCenter, "card0");

}

}

});

Container con=getContentPane();

con.add(pnBorder);

}

**public** **void** showImage(JPanel pn,String cardName)

{

CardLayout cl=(CardLayout)pn.getLayout();

cl.show(pn, cardName);

}

**private** **class** TimerPanel **implements** ActionListener

{

JPanel pn=**null**;

**public** TimerPanel(JPanel pn) {

**this**.pn=pn;

}

**public** **void** actionPerformed(ActionEvent arg0) {

showImage(pn,"card"+pos);

pos++;

**if**(pos>=pn.getComponentCount())

pos=0;

}

}

**public** **static** **void** main(String[] args) {;

ImageAnimation2 imgUi=**new** ImageAnimation2("Image Animation!");

imgUi.doShow();

}

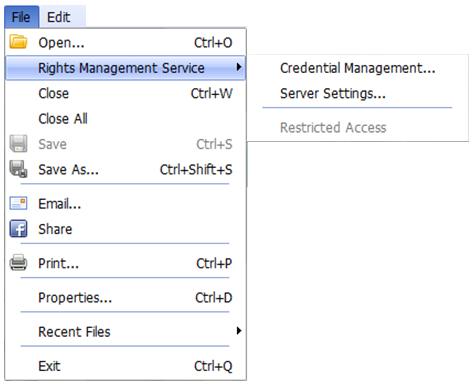
}

**Bài tập 11:** Thực hành về tạo Menu. Yêu cầu thiết kế Menu theo hình sau, ứng với mỗi menu item sinh viên hãy cài đặt coding để hiển thị thông báo là đang chọn menu nào. (\*)

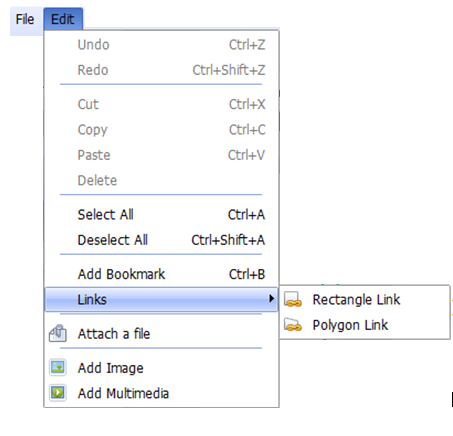
**Hướng dẫn:** JMenuBar🡺JMenu🡺JMenuItem. Phải biết kết hợp các class này.

MenuBar sẽ add Menu, Menu sẽ add MenuItem, rồi gọi setJMenuBar(menuBar);

Yêu cầu giả lập Menu giống như chương trình Foxit Reader:



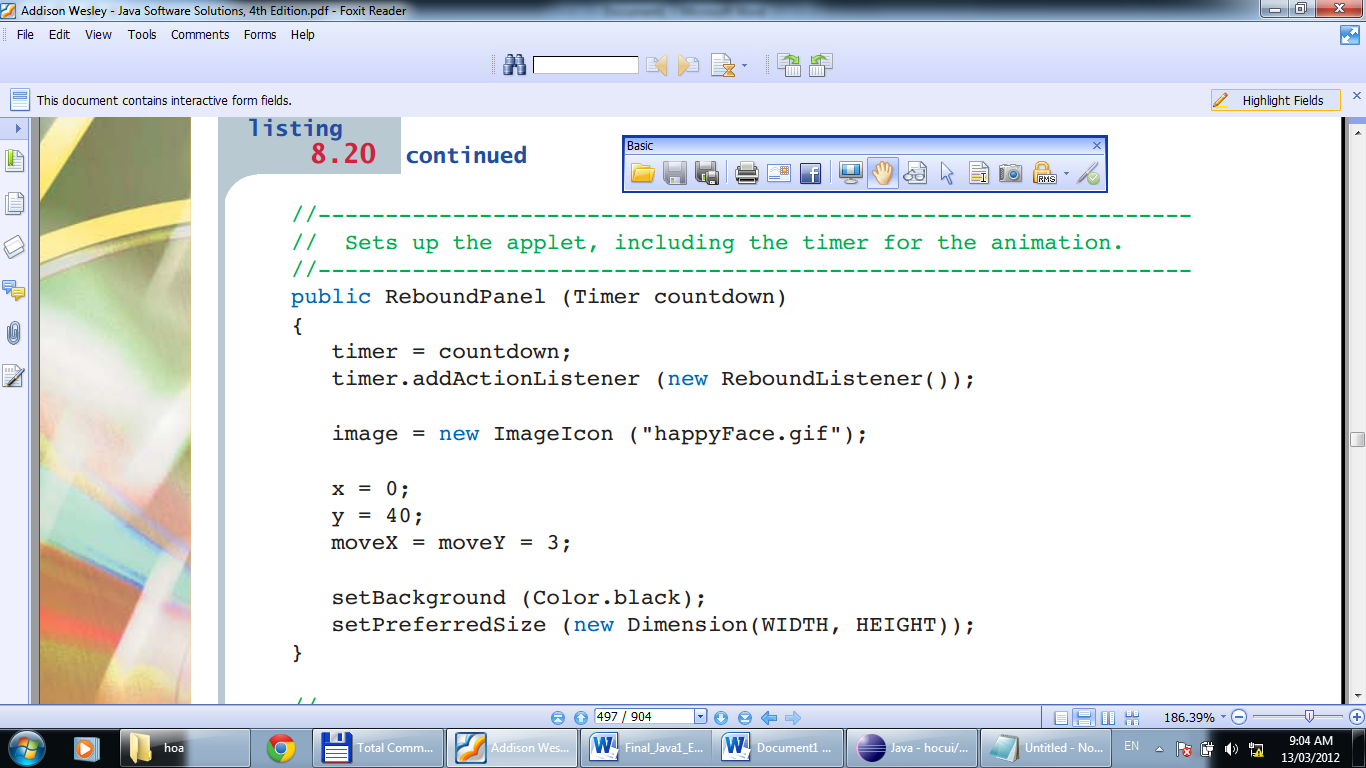
Menu File có giao diện như trên



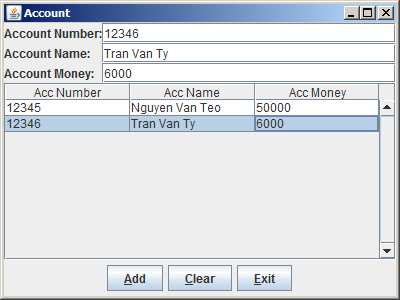
Menu Edit có giao diện như trên

**Bài tập 12:** Thực hành về JToolBar, tương tự như câu 12, giả lập Toolbar của chương trình Foxit Reader, ứng với mỗi lệnh trên JToolBar, sinh viên hãy xuấtt thông báo đang sử dụng chức năng nào. (\*)

Hướng dẫn: tạo các JButton rồi add vào JToolBar



**Bài tập 13:** Thiết kế giao diện như hình bên dưới - JTable: (\*)



**package** baitap13;

**import** java.awt.BorderLayout;

**import** java.awt.Container;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** javax.swing.BoxLayout;

**import** javax.swing.JButton;

**import** javax.swing.JFrame;

**import** javax.swing.JLabel;

**import** javax.swing.JOptionPane;

**import** javax.swing.JPanel;

**import** javax.swing.JScrollPane;

**import** javax.swing.JTable;

**import** javax.swing.JTextField;

**import** javax.swing.table.DefaultTableModel;

**public** **class** AccUI {

JFrame myFrame=**new** JFrame("Account");

JTextField txtAccNumber;

JTextField txtAccName;

JTextField txtAccMoney;

JTable tblAcc;

DefaultTableModel tblModelAcc;

JButton btnAdd,btnClear,btnExit;

**public** AccUI()

{

JPanel pnAccNumber=**new** JPanel();

pnAccNumber.setLayout(**new** BoxLayout(pnAccNumber, BoxLayout.*X\_AXIS*));

JLabel lblAccNumber=**new** JLabel("Account Number:");

pnAccNumber.add(lblAccNumber);

txtAccNumber=**new** JTextField(15);

pnAccNumber.add(txtAccNumber);

JPanel pnAccName=**new** JPanel();

pnAccName.setLayout(**new** BoxLayout(pnAccName, BoxLayout.*X\_AXIS*));

JLabel lblAccName=**new** JLabel("Account Name:");

pnAccName.add(lblAccName);

txtAccName=**new** JTextField(15);

pnAccName.add(txtAccName);

JPanel pnAccMoney=**new** JPanel();

pnAccMoney.setLayout(**new** BoxLayout(pnAccMoney, BoxLayout.*X\_AXIS*));

JLabel lblAccMoney=**new** JLabel("Account Money:");

pnAccMoney.add(lblAccMoney);

txtAccMoney=**new** JTextField(15);

pnAccMoney.add(txtAccMoney);

Container con=myFrame.getContentPane();

con.setLayout(**new** BorderLayout());

JPanel pInfor=**new** JPanel();

pInfor.setLayout(**new** BoxLayout(pInfor, BoxLayout.*Y\_AXIS*));

pInfor.add(pnAccNumber);

pInfor.add(pnAccName);

pInfor.add(pnAccMoney);

con.add(pInfor,BorderLayout.*NORTH*);

tblModelAcc=**new** DefaultTableModel();

tblModelAcc.addColumn("Acc Number");

tblModelAcc.addColumn("Acc Name");

tblModelAcc.addColumn("Acc Money");

tblAcc=**new** JTable(tblModelAcc);

JScrollPane sc=**new** JScrollPane(tblAcc,JScrollPane.*VERTICAL\_SCROLLBAR\_ALWAYS*,JScrollPane.*HORIZONTAL\_SCROLLBAR\_AS\_NEEDED*);

con.add(sc,BorderLayout.*CENTER*);

JPanel pnButton=**new** JPanel();

btnAdd=**new** JButton("Add");

btnAdd.setMnemonic('A');

btnClear=**new** JButton("Clear");

btnClear.setMnemonic('C');

btnExit=**new** JButton("Exit");

btnExit.setMnemonic('E');

pnButton.add(btnAdd);

pnButton.add(btnClear);

pnButton.add(btnExit);

con.add(pnButton,BorderLayout.*SOUTH*);

lblAccName.setPreferredSize(lblAccNumber.getPreferredSize());

lblAccMoney.setPreferredSize(lblAccNumber.getPreferredSize());

addEvents();

}

**public** **void** addEvents()

{

btnAdd.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

String arr[]={txtAccNumber.getText(),txtAccName.getText(),txtAccMoney.getText()};

tblModelAcc.addRow(arr);

}

});

btnExit.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

**int** ret=JOptionPane.*showConfirmDialog*(**null**, "Are you sure you want to exit?", "Exit!", JOptionPane.*YES\_NO\_OPTION*);

**if**(ret==JOptionPane.*YES\_OPTION*)

System.*exit*(0);

}

});

}

**public** **void** doShow()

{

myFrame.setSize(400, 300);

myFrame.setLocationRelativeTo(**null**);

myFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);

myFrame.setVisible(**true**);

}

}

**package** baitap13;

**public** **class** TestAccount {

/\*\*

\* **@param** args

\*/

**public** **static** **void** main(String[] args) {

AccUI ui=**new** AccUI();

ui.doShow();

}

}