

Bài 04:

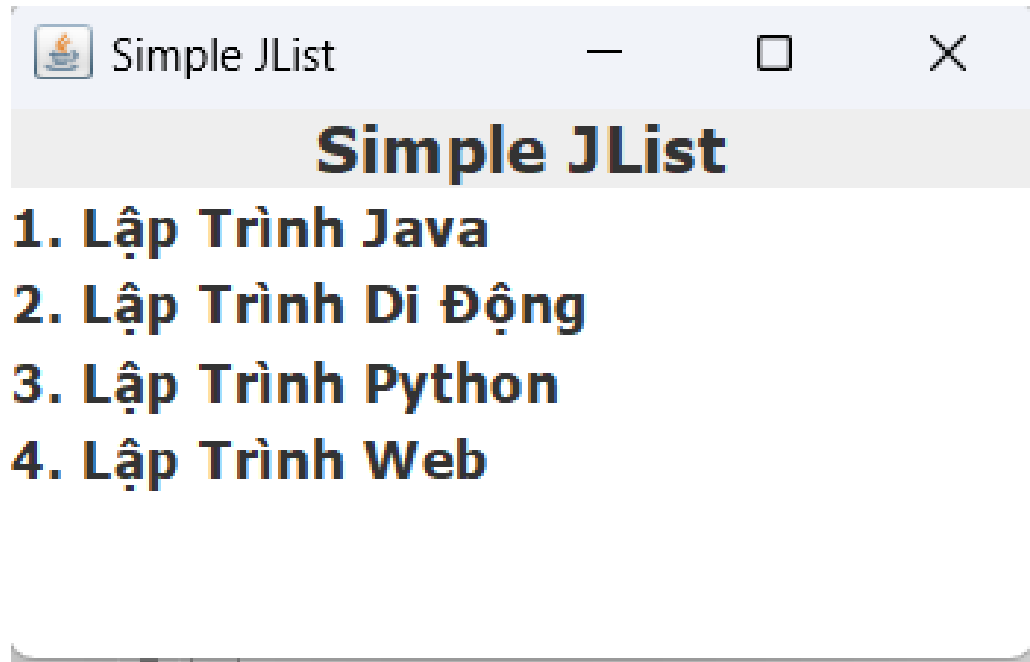
CÁC LOẠI LAYOUT MANAGER

THƯỜNG DÙNG (Tiếp theo)

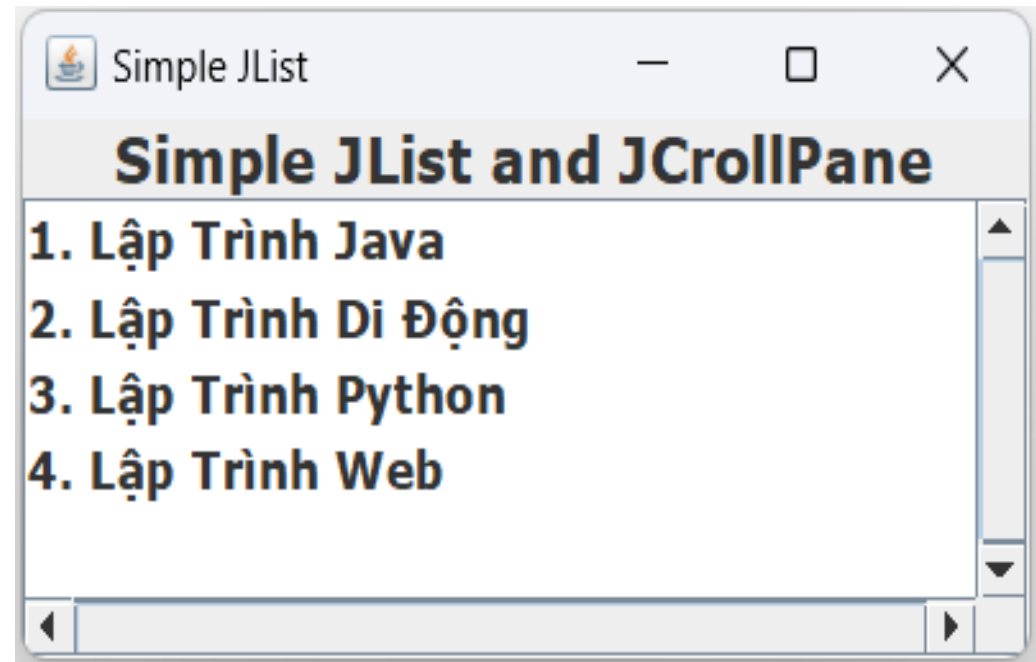
Giảng Viên: ThS. Giang Hào Côn

4.1/ JList

a) Giới Thiệu



b) Một số tính năng của JList



Sinh viên nhận xét ?

4.1/ JList

1./ **Tạo Jlist:** có 2 cách **tạo trực tiếp** và **tạo từ mô hình dữ liệu**

```
// Tạo JList trực tiếp
JList<String> list = new JList<>(new String[]{"Item 1", "Item 2", "Item 3"});

// Tạo JList sử dụng mô hình dữ liệu
DefaultListModel<String> model = new DefaultListModel<>();
model.addElement("Item 1");
model.addElement("Item 2");
model.addElement("Item 3");
JList<String> list = new JList<>(model);
```

4.1/ JList

2./ Màu nền và màu văn bản:

```
1 list.setBackground(Color.WHITE); // Đặt màu nền  
2 list.setForeground(Color.BLACK); // Đặt màu văn bản
```

3./ Font chữ:

```
list.setFont(new Font("Arial", Font.PLAIN, 12)); // Đặt font Arial, kiểu chữ thường, cỡ chữ 12
```

4./ Đường viền

```
list.setBorder(BorderFactory.createLineBorder(Color.GRAY)); // Đặt đường viền màu xám
```

4.1/ JList

5./ Xử lý sự kiện với JList: Nghe sự kiện khi mục được chọn

```
list.addListSelectionListener(new ListSelectionListener() {  
    public void valueChanged(ListSelectionEvent e) {  
        // Xử lý sự kiện khi mục được chọn  
        // Lấy mục được chọn thông qua phương thức getSelectedValue()  
        Object selectedValue = list.getSelectedValue();  
        // ...  
    }  
});
```

4.1/ JList

5./ Xử lý sự kiện với JList: Nghe sự kiện khi chuột phải được nhấp vào mục

```
list.addMouseListener(new MouseAdapter() {  
    public void mouseClicked(MouseEvent e) {  
        if (e.getButton() == MouseEvent.BUTTON3) {  
            // Xử lý sự kiện khi chuột phải được nhấp vào mục  
            // Lấy mục được nhấp thông qua phương thức locationToIndex()  
            int index = list.locationToIndex(e.getPoint());  
            Object clickedValue = list.getModel().getElementAt(index);  
            // ...  
        }  
    }  
});
```

4.1/ JList

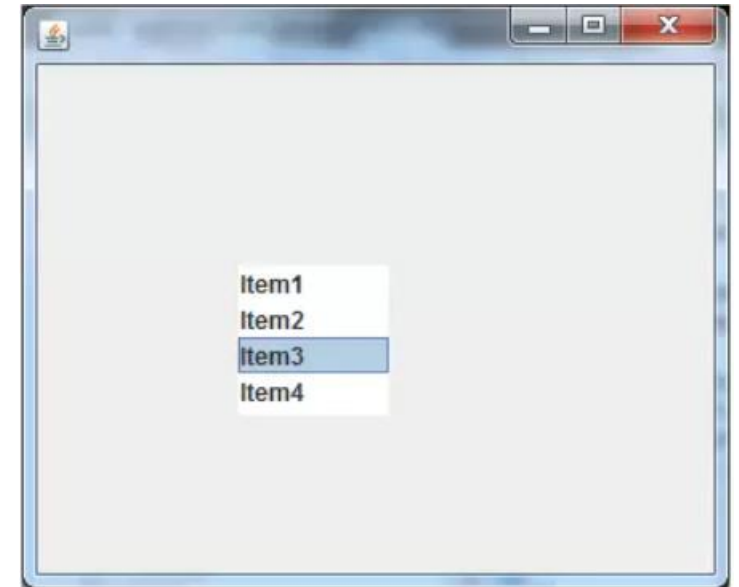
5./ Xử lý sự kiện với JList: Nghe sự kiện khi mục được kéo thả:

```
list.setTransferHandler(new TransferHandler() {  
    // Triển khai các phương thức của TransferHandler  
    // để xử lý sự kiện kéo thả  
    // ...  
});
```

Ứng dụng của JList trong giao diện người dùng

4.1/JList – Ví dụ 01

```
import javax.swing.*;
public class ListExample
{
    ListExample(){
        JFrame f= new JFrame();
        DefaultListModel<String> l1 = new DefaultListModel<>();
        l1.addElement("Item1");
        l1.addElement("Item2");
        l1.addElement("Item3");
        l1.addElement("Item4");
        JList<String> list = new JList<>(l1);
        list.setBounds(100,100, 75,75);
        f.add(list);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
    }
    public static void main(String args[])
    {
        new ListExample();
    }
}
```



4.1/JList – Ví dụ 02

```
import javax.swing.*;
import java.awt.event.*;
public class ListExample
{
    ListExample(){
        JFrame f= new JFrame();
        final JLabel label = new JLabel();
        label.setSize(500,100);
        JButton b=new JButton("Show");
        b.setBounds(200,150,80,30);
        final DefaultListModel<String> l1 = new DefaultListModel<>();
        l1.addElement("C");
        l1.addElement("C++");
        l1.addElement("Java");
        l1.addElement("PHP");
        final JList<String> list1 = new JList<>(l1);
        list1.setBounds(100,100, 75,75);
    }
}
```

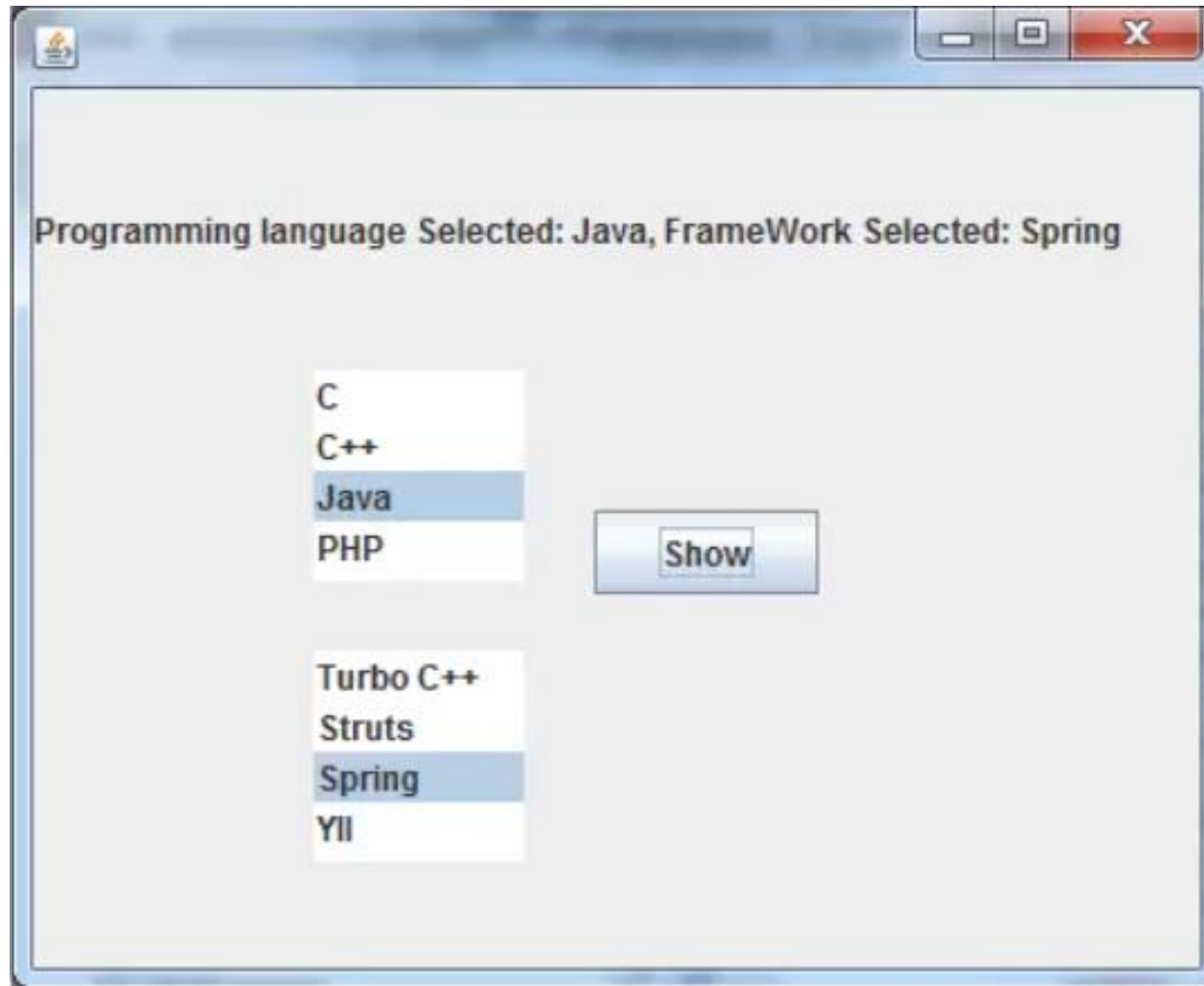
4.1/JList – Ví dụ 02

```
DefaultListModel<String> l2 = new DefaultListModel<>();  
l2.addElement("Turbo C++");  
l2.addElement("Struts");  
l2.addElement("Spring");  
l2.addElement("YII");  
final JList<String> list2 = new JList<>(l2);  
list2.setBounds(100, 200, 75, 75);  
f.add(list1); f.add(list2); f.add(b); f.add(label);  
f.setSize(450, 450);  
f.setLayout(null);  
f.setVisible(true);
```


4.1/JList – Ví dụ 02

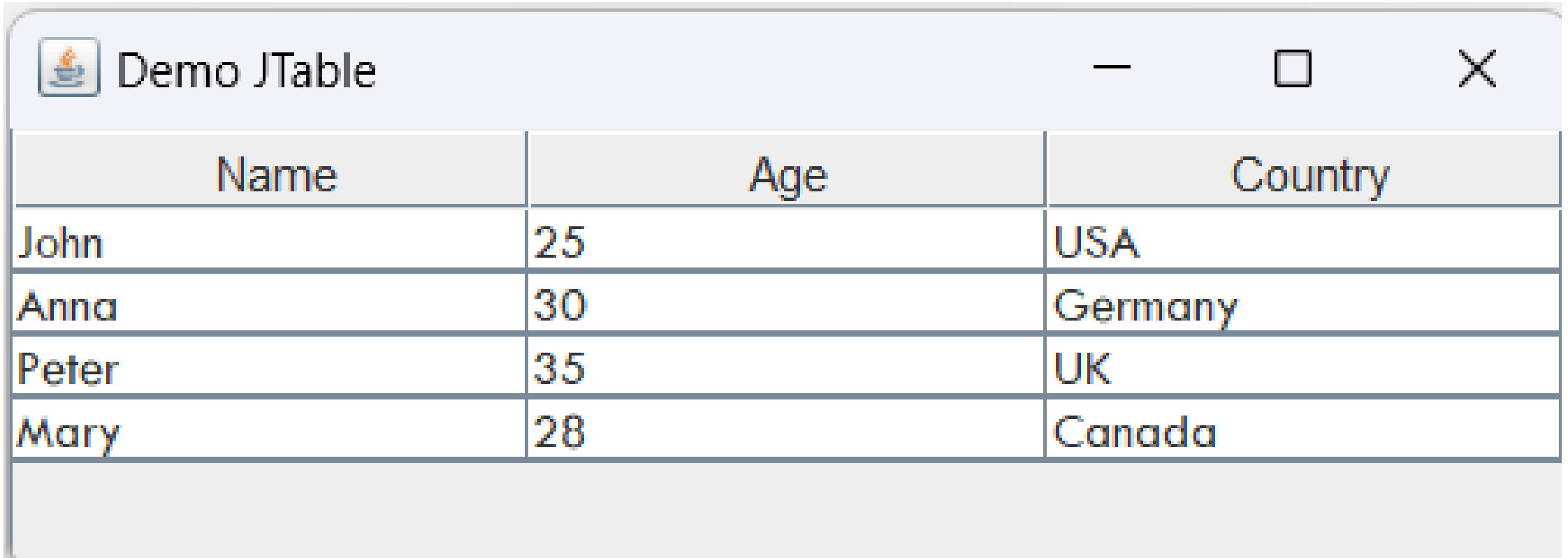
```
b.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        String data = "";  
        if (list1.getSelectedIndex() != -1) {  
            data = "Programming language Selected: " + list1.getSelectedValue();  
            label.setText(data);  
        }  
        if(list2.getSelectedIndex() != -1){  
            data += ", Framework Selected: ";  
            for(Object frame :list2.getSelectedValues()){  
                data += frame + " ";  
            }  
        }  
        label.setText(data);  
    }  
});  
}  
public static void main(String args[])  
{  
    new ListExample();  
}}
```

4.1/JList – Ví dụ 02



4.2/ JTable

1./ Giới Thiệu



Name	Age	Country
John	25	USA
Anna	30	Germany
Peter	35	UK
Mary	28	Canada

4.2/ JTable

2./ Tạo JTable đơn giản

Bước 01: Tạo một đối tượng `DefaultTableModel` để lưu trữ dữ liệu của bảng:

```
DefaultTableModel model = new DefaultTableModel();
```

Bước 02: Thêm các cột vào bảng bằng phương thức `addColumn()`:

```
model.addColumn("Column 1");  
model.addColumn("Column 2");  
model.addColumn("Column 3");
```

4.2/ JTable

2./ Tạo JTable đơn giản

Bước 03: Thêm các dòng dữ liệu vào bảng bằng phương thức `addRow()`:

```
model.addRow(new Object[]{"Data 1", "Data 2", "Data 3"});  
model.addRow(new Object[]{"Data 4", "Data 5", "Data 6"});
```

Bước 04: Tạo một đối tượng `JTable` và gán `model` dữ liệu cho nó:

```
JTable table = new JTable(model);
```


4.2/ JTable

2./ Tạo JTable đơn giản

Bước 05: Đặt `JTable` vào một `JScrollPane` để cho phép cuộn.

```
JScrollPane scrollPane = new JScrollPane(table);
```

Bước 06: Thêm `JScrollPane` vào giao diện của ứng dụng:.

```
frame.add(scrollPane);
```

4.2/ JTable

3./ Tùy chỉnh màu sắc nền và văn bản trong ô của JTable:

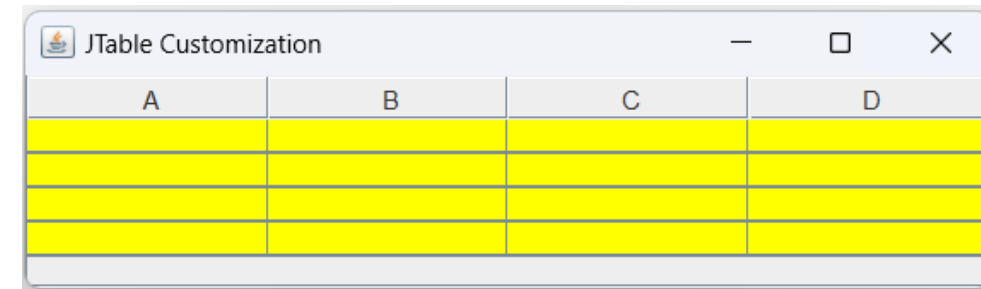
```
import javax.swing.*;
import javax.swing.table.DefaultTableCellRenderer;
import java.awt.*;

public class JTableCustomization {
    public static void main(String[] args) {
        JTable table = new JTable(4, 4);

        // Tạo một đối tượng renderer để tùy chỉnh màu sắc và văn bản
        DefaultTableCellRenderer renderer = new DefaultTableCellRenderer();
        renderer.setBackground(Color.YELLOW);
        renderer.setForeground(Color.BLUE);
        renderer.setFont(new Font("Arial", Font.BOLD, 14));

        // Áp dụng renderer cho tất cả các ô trong JTable
        table.setDefaultRenderer(Object.class, renderer);

        JFrame frame = new JFrame("JTable Customization");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.add(new JScrollPane(table));
        frame.pack();
        frame.setVisible(true);
    }
}
```



A	B	C	D

4.2/ JTable

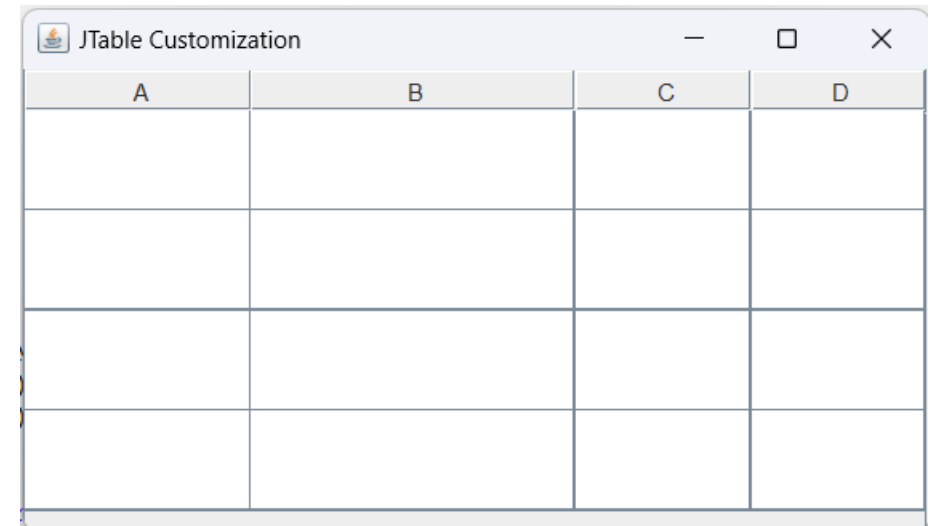
4./ Tùy chỉnh độ rộng cột và chiều cao hàng của JTable:

```
import javax.swing.*;
import javax.swing.table.TableColumnModel;

public class JTableWidth {
    public static void main(String[] args) {
        JTable table = new JTable(4, 4);

        // Tạo đối tượng TableColumnModel để tùy chỉnh
        // độ rộng cột và chiều cao hàng
        TableColumnModel columnModel = table.getColumnModel();
        columnModel.getColumn(0).setPreferredWidth(100);
        columnModel.getColumn(1).setPreferredWidth(150);
        table.setRowHeight(50);

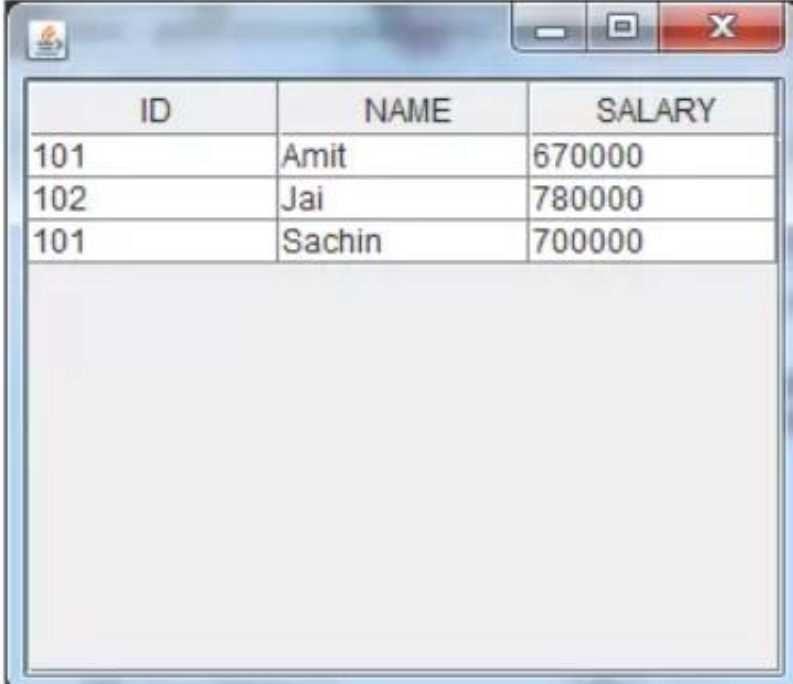
        JFrame frame = new JFrame("JTable Customization");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.add(new JScrollPane(table));
        frame.pack();
        frame.setVisible(true);
    }
}
```



A	B	C	D

4.2/ Jtable – Ví dụ 01

```
import javax.swing.*;
public class TableExample {
    JFrame f;
    TableExample(){
        f=new JFrame();
        String data[][]={ {"101","Amit","670000"},
                           {"102","Jai","780000"},
                           {"101","Sachin","700000"}};
        String column[]={"ID","NAME","SALARY"};
        JTable jt=new JTable(data,column);
        jt.setBounds(30,40,200,300);
        JScrollPane sp=new JScrollPane(jt);
        f.add(sp);
        f.setSize(300,400);
        f.setVisible(true);
    }
    public static void main(String[] args) {
        new TableExample();
    }
}
```

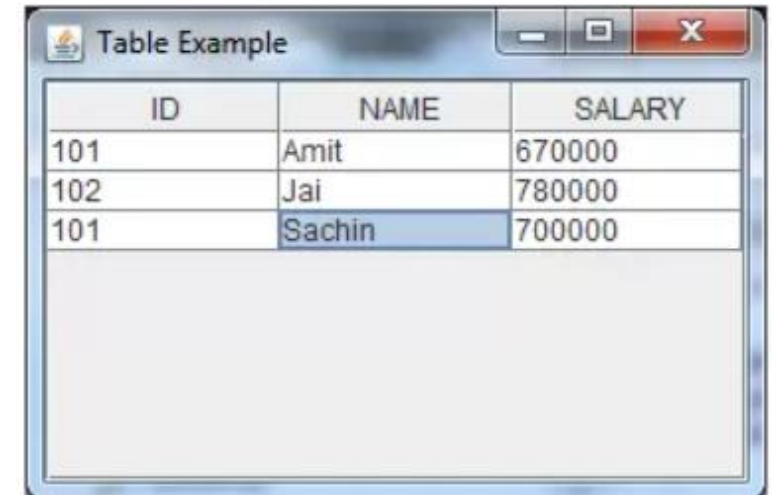


ID	NAME	SALARY
101	Amit	670000
102	Jai	780000
101	Sachin	700000

4.2/ Jtable – Ví dụ 02 Java JTable với ListSelectionListener

```
import javax.swing.*;
import javax.swing.event.*;
public class TableExample {
    public static void main(String[] a) {
        JFrame f = new JFrame("Table Example");
        String data[][] = { {"101", "Amit", "670000"},
                            {"102", "Jai", "780000"},
                            {"101", "Sachin", "700000"} };

        String column[] = {"ID", "NAME", "SALARY"};
        final JTable jt = new JTable(data, column);
        jt.setCellSelectionEnabled(true);
        ListSelectionModel select = jt.getSelectionModel();
        select.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        select.addListSelectionListener(new ListSelectionListener() {
            public void valueChanged(ListSelectionEvent e) {
                String Data = null;
                int[] row = jt.getSelectedRows();
                int[] columns = jt.getSelectedColumns();
                for (int i = 0; i < row.length; i++) {
                    for (int j = 0; j < columns.length; j++) {
                        Data = (String) jt.getValueAt(row[i], columns[j]);
                    }
                }
                System.out.println("Table element selected is: " + Data);
            }
        });
        JScrollPane sp = new JScrollPane(jt);
        f.add(sp);
        f.setSize(300, 200);
        f.setVisible(true);
    }
}
```



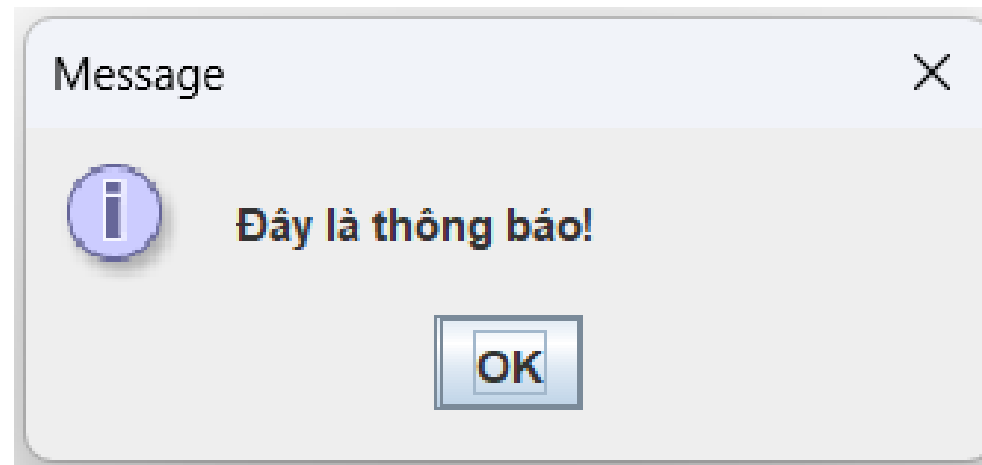
ID	NAME	SALARY
101	Amit	670000
102	Jai	780000
101	Sachin	700000

4.3/ JOptionPane

a./ Khái niệm cơ bản

a.1./Hiển thị Thông Báo Đơn Giản

```
JOptionPane.showMessageDialog(null, "Đây là thông báo!");
```

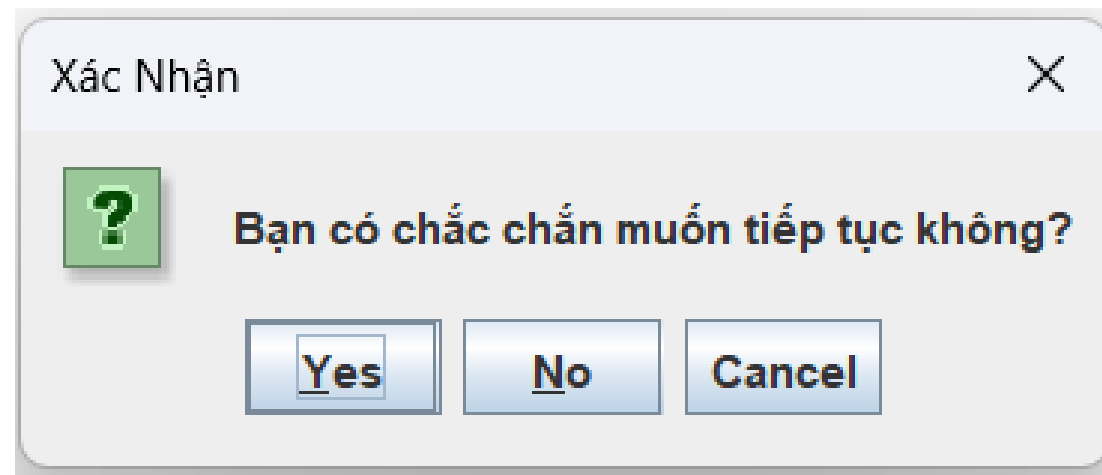


4.3/ JOptionPane

a./ Khái niệm cơ bản

a.2./ Cửa Sổ Xác Nhận (Confirm Dialog):

```
int response = JOptionPane.showConfirmDialog(null, "Bạn có chắc chắn muốn tiếp tục không?",  
                                             "Xác Nhận", JOptionPane.YES_NO_CANCEL_OPTION);
```

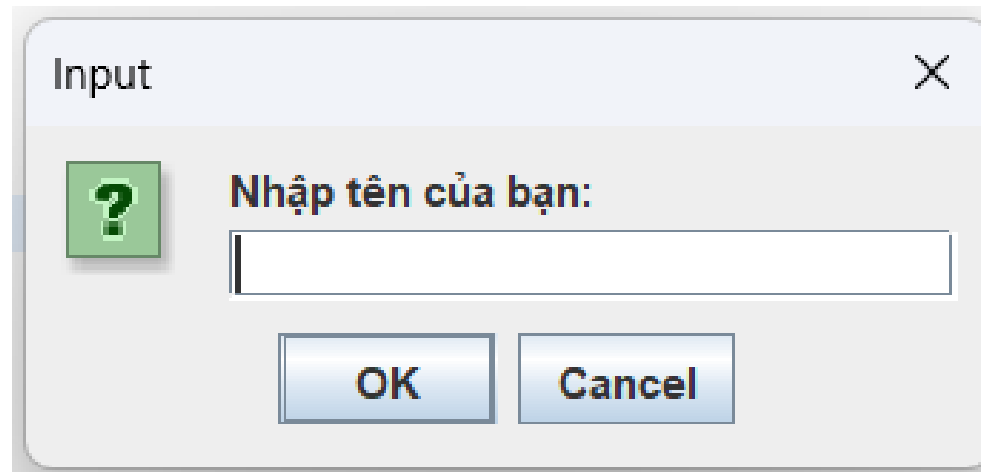


4.3/ JOptionPane

a./ Khái niệm cơ bản

a.3./Cửa Sổ Nhập Liệu (Input Dialog):

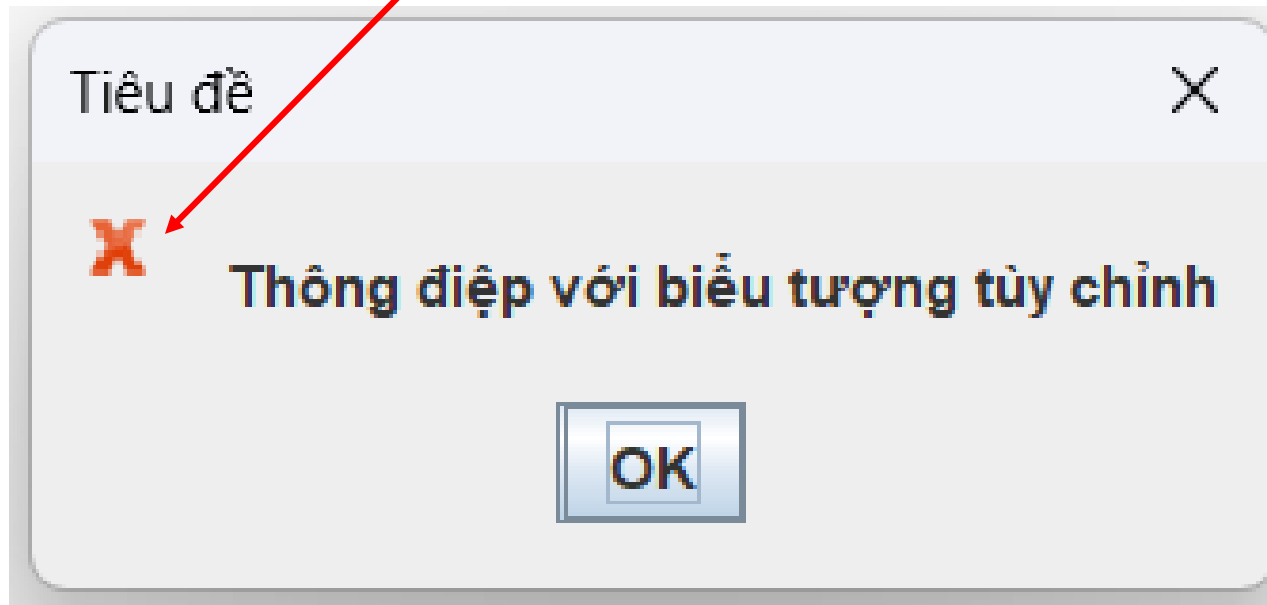
```
String name = JOptionPane.showInputDialog(null, "Nhập tên của bạn:");
```



4.3/ JOptionPane

b./ Tùy chỉnh Icon

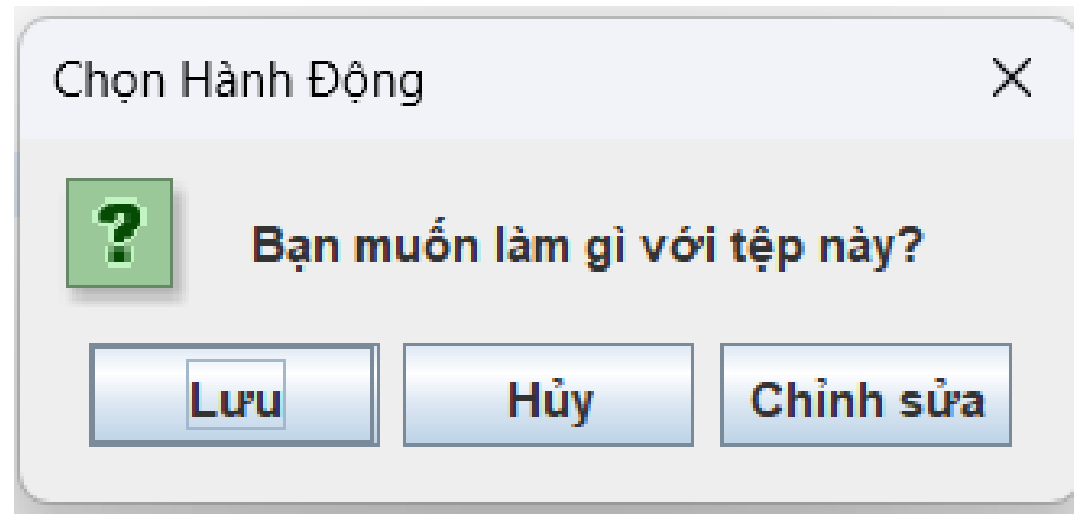
```
ImageIcon customIcon = new ImageIcon("Hinh/btnThoat.png");  
JOptionPane.showMessageDialog(null, "Thông điệp với biểu tượng tùy chỉnh", "Tiêu đề",  
JOptionPane.INFORMATION_MESSAGE, customIcon);
```



4.3/ JOptionPane

c./ Tùy Chọn Nút Bấm

```
Object[] options = {"Lưu", "Hủy", "Chỉnh sửa"};  
int response = JOptionPane.showOptionDialog(null, "Bạn muốn làm gì với tệp này?", "Chọn Hành Động",  
JOptionPane.DEFAULT_OPTION, JOptionPane.QUESTION_MESSAGE, null, options, options[0]);
```



4.3/ JOptionPane

d./ Xử Lý Sự Kiện với showConfirmDialog

```
int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn lưu thay đổi không?", "Xác nhận",
JOptionPane.YES_NO_CANCEL_OPTION);

if (result == JOptionPane.YES_OPTION) {
    // Người dùng đã chọn "Có", thực hiện hành động tương ứng
    // Ví dụ: Lưu dữ liệu
} else if (result == JOptionPane.NO_OPTION) {
    // Người dùng đã chọn "Không", thực hiện hành động tương ứng
    // Ví dụ: Không lưu và thoát
} else if (result == JOptionPane.CANCEL_OPTION) {
    // Người dùng đã chọn "Hủy", thực hiện hành động tương ứng
    // Ví dụ: Hủy bỏ thay đổi
}
```

4.3/ JOptionPane

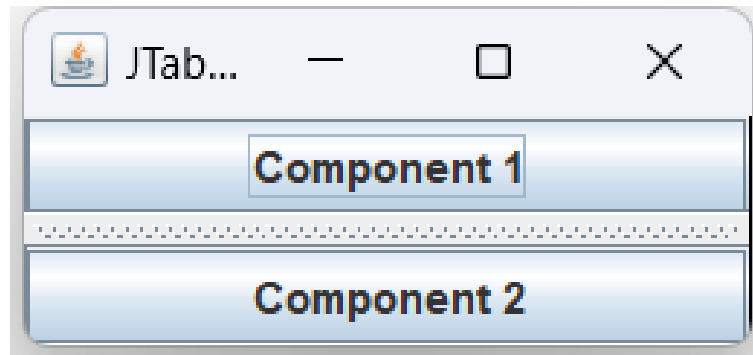
e./ Xử Lý Sự Kiện với showDialog

```
String input = JOptionPane.showInputDialog(null, "Nhập tên của bạn:");  
if (input != null) {  
    // Người dùng đã nhập dữ liệu và chọn "OK"  
    // Ví dụ: Hiển thị tên đã nhập  
    JOptionPane.showMessageDialog(null, "Tên của bạn là: " + input);  
} else {  
    // Người dùng đã chọn "Hủy" hoặc đóng cửa sổ nhập liệu  
    // Ví dụ: Xử lý trường hợp không có dữ liệu nhập  
}
```

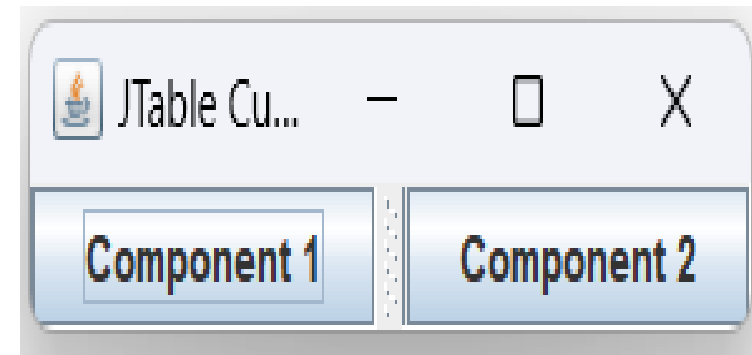
4.4/ Jsplif Pane

a./ Giới thiệu

VERTICAL_SPLIT




HORIZONTAL_SPLIT

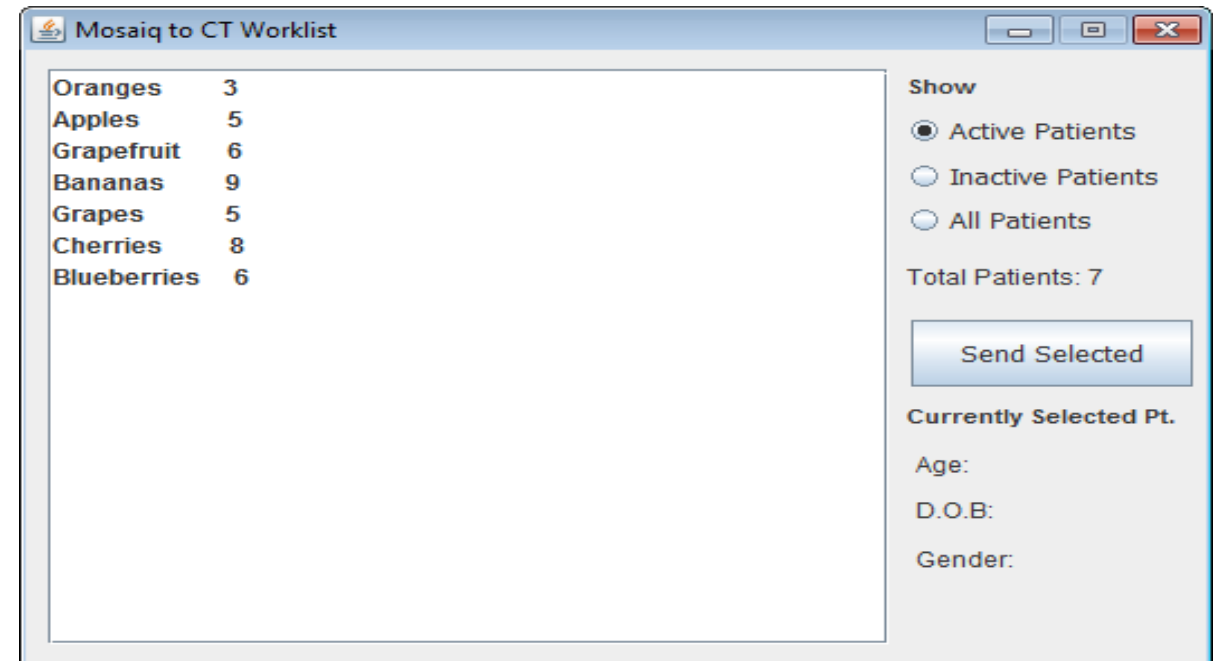


Câu hỏi thảo luận

- 1) Trình Bày ứng dụng của Jlist, Jtable và JSplitPane trong giao diện người dùng.
- 2) Viết Code cho giao diện a và b như sau:



The screenshot shows a Java Swing window titled "Quản lý sách - Trần Minh". The window contains a form titled "Quản Lý Sách" with three input fields: "Mã Sách" (book ID) with the value "CNTT1", "Tên Sách" (book name) with the value "Java Programming", and "Năm sản xuất" (year of production) with the value "2012". To the right of these fields are four buttons: "Sắp Xếp" (Sort), "Thêm" (Add), "Hủy" (Cancel), and "Xóa" (Delete). At the bottom left, there are navigation buttons: "<<", "1/6", and ">>". At the bottom right, there is a "Thoát" (Exit) button.



The screenshot shows a Java Swing window titled "Mosaik to CT Worklist". The window contains a list of fruits and their counts: Oranges (3), Apples (5), Grapefruit (6), Bananas (9), Grapes (5), Cherries (8), and Blueberries (6). To the right of the list are three radio buttons for "Show": "Active Patients" (selected), "Inactive Patients", and "All Patients". Below these is the text "Total Patients: 7" and a "Send Selected" button. At the bottom right, there are labels for "Currently Selected Pt.", "Age:", "D.O.B:", and "Gender:".