

CJ Lecture-79

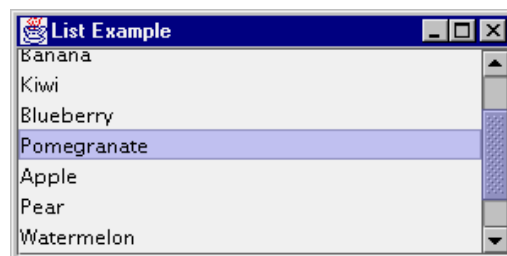
Topic: JList

A component that displays a list of objects and allows the user to select one or more items.

There is one major difference between List and JList. JList doesn't directly support scrolling.

You need to place the JList within a JScrollPane object, and let it deal with the scrolling.

```
public class ListPanel extends JPanel {  
    String label [] = {"Cranberry", "Orange",  
        "Banana", "Kiwi", "Blueberry",  
        "Pomegranate", "Apple", "Pear",  
        "Watermelon", "Raspberry", "Snozberry"  
    };  
    public ListPanel() {  
        setLayout (new BorderLayout());  
        JList list = new JList(label);  
        JScrollPane pane = new JScrollPane(list);  
        add(pane, BorderLayout.CENTER);  
    }  
}
```






Selecting Items in a List

By default, a list selection model allows any combination of items to be selected at a time.

You can specify a different selection mode by calling the **setSelectionMode** method on the list.

The following table describes the three list selection modes:

Mode	Description
SINGLE_SELECTION 	Only one item can be selected at a time. When the user selects an item, any previously selected item is deselected first.
SINGLE_INTERVAL_SELECTION 	Multiple, contiguous items can be selected. When the user begins a new selection range, any previously selected items are deselected first.
MULTIPLE_INTERVAL_SELECTION 	The default. Any combination of items can be selected. The user must explicitly deselect items.

Constructors

JList()

Constructs a JList with an empty, read-only, model.

JList(E[] listData)

Constructs a JList that displays the elements in the specified array.

JList(Vector <E> listData)

Constructs a JList that displays the elements in the specified Vector.

Methods

void	addListSelectionListener(ListSelectionListener listener) Adds a listener to the list, to be notified each time a change to the selection occurs; the preferred way of listening for selection state changes.
void	clearSelection() Clears the selection.
int	getSelectedIndex() Returns the smallest selected cell index.
int[]	getSelectedIndices() Returns an array of all of the selected indices, in increasing order.
<u>E</u>	getSelectedValue() Returns the value for the smallest selected cell index.
Object[]	getSelectedValues() Returns an array of selected values.
int	getSelectionMode() Returns the current selection mode for the list.
boolean	isSelectedIndex(int index) Returns true if the specified index is selected, else false.
Boolean	isSelectionEmpty() Returns true if nothing is selected, else false.
void	setSelectedIndex(int index) Selects a single cell.
void	setSelectedIndices(int[] indices) Changes the selection to be the set of indices specified by the given array.
void	setSelectionMode(int selectionMode) Sets the selection mode for the list. Constants for slectionMode are as follows: ListSelectionModel.SINGLE_SELECTION ListSelectionModel.SINGLE_INTERVAL_SELECTION ListSelectionModel.MULTIPLE_INTERVAL_SELECTION

Program to demonstrate use of JList

```
import java.awt.*;
import javax.swing.*;
import javax.swing.event.*;

class ListTest extends JFrame implements ListSelectionListener
{
    String []items = {"PD","CD","HDD","CPU","KB","MB","Mouse",
                     "Plotter","Printer","Scanner"};

    JList<String> list = new JList<String>(items);
    JLabel label = new JLabel();

    ListTest()
    {
        setSize(600,500);
        setTitle("List Box Test");

        JPanel center = new JPanel();
        JScrollPane sp = new JScrollPane(list);
        list.setVisibleRowCount(5);
        center.add(sp);
        add(center);
        list.setSelectionMode(ListSelectionModel.SINGLE_INTERVAL_SELECTION);
        list.addListSelectionListener(this);

        add(label,"North");
    }
    public void valueChanged(ListSelectionEvent e)
    {
        Object values[] = list.getSelectedValues();
        String s = "Selected items are ";
        for(int i=0;i<values.length;i++)
            s=s+values[i]+ " ";
        label.setText(s);
    }
    public static void main(String arg[])
    {
        ListTest f = new ListTest();
        f.setDefaultCloseOperation(3);
    }
}
```

```
        f.setVisible(true);  
    }  
}
```

Output:

