Using the JFC JList Class

You will note that the previous example used the AWT List class. This is all quite straightforward, but suppose you would like to rewrite the program using the Java Foundation Classes (JFC or "Swing"). Most of the methods you use for creating and manipulating the user interface remain the same. However, the JFC JList class is markedly different from the AWT List class. In fact, because the JList class was designed to represent far more complex kinds of lists, there are virtually no methods in common between the classes.

AWT List Class	JFC JList Class
`add(String);`	na
`remove(String)`	na
`String[] getSelectedItems()`	`Object[] getSelectedValues()`

Both classes have quite a number of other methods, and almost none of them are closely correlated. However, since we have already written the program once and make use of two different list boxes, writing an adapter to make the JList class look like the List class will provide a rapid solution to our problem.

The JList class is a window container that has an array, Vector, or other ListModel class associated with it. It is this ListModel that actually contains and manipulates the data. Further, the JList class does not contain a scroll bar but instead relies on being inserted in the viewport of the JScrollPane class. Data in the JList class and its associated ListModel are not limited to strings; they may be almost any kind of objects, as long as you provide the cell drawing routine for them. This makes it possible to have list boxes with pictures illustrating each choice in the list.

In this case, we are going to create a class that only emulates the List class, which here needs only the three methods shown in the previous table.

We can define the needed methods as an interface and then make sure that the class that we create implements those methods.

```
public interface awtList {
     public void      add(String s);
     public void      remove(String s);
     public String[]  getSelectedItems();
}
```

Interfaces are important in Java because Java does not allow multiple inheritance as C++ does. So you cannot create a class that inherits from JList and List at the same time. However, you can create a class that inherits from one class