This is shown in Figure 31.2.

Figure 31.2 A JList with the selected item displayed in the JTextField.

Changing a List Display Dynamically

For a list to change dynamically while your program is running, the problem is somewhat more involved because the JList displays the data that it is initially loaded with and does not update the display unless you tell it to. One simple way to accomplish this is to use the *setListData* method of JList to keep passing it a new version of the updated Vector after you change it. In the JListDemo.java program, the contents of the top text field are added to the list each time that the Add button is clicked. All of this place in the *action* routine for that button.

```
public void actionPerformed(ActionEvent e)      {
    dlist.addElement(text.getText());    //add text from the field
    list.setListData(dlist);        //send a new Vector to the list
    list.repaint();                 //and tell it to redraw
}
```

One drawback to this solution is that you are passing the entire Vector to the list each time and it must update its entire contents each time rather than updating only the portion that changed. This brings us to the underlying ListModel that contains the data that the JList displays.

When you create a JList using an array or a Vector, the JList automatically creates a ListModel object that contains those data. Each ListModel object is a concrete subclass of the AbstractListModel class that defines the following methods:

```
void fireContentsChanged(Object source, int index0, int index1);
void fireIntervalAdded(Object source, int index0, int index1);
void fireIntervalRemoved(Object source, int index0, int index1);
```