Just as the ListModel class does for JList objects, the TableModel class holds and manipulates the data and notifies the JTable whenever it changes. Thus the JTable is an Observer pattern, operating on the TableModel data.

Cell Renderers

Each cell in a table is rendered by a cell renderer. The default renderer is a JLabel, which may be used for all of the data in several columns. Thus these cell renderers can be thought of as Flyweight pattern implementations. The JTable class chooses the renderer according to the object's type, as discussed previously. However, you easily can change to a different renderer, such as one that uses another color, or to another visual interface.

Cell renderers are registered by type of data,

```
table.setDefaultRenderer(String.class, new ourRenderer());
```

and each renderer is passed the object, selected mode, row, and column using the only required public method.

```
public Component getTableCellRendererComponent(JTable jt,
    Object value, boolean isSelected,
    boolean hasFocus, int row, int column)
```

One common way to implement a cell renderer is to extend the JLabel type, catch each rendering request within the renderer, and return a properly configured JLabel object, usually the renderer itself. The following renderer displays cell (1, 1) in boldface and the remaining cells in plain black type:

```
public class ourRenderer extends JLabel
    implements TableCellRenderer {
        Font bold, plain;
        public ourRenderer() {
            super();
            setOpaque(true);
            setBackground(Color.white);
            bold  = new Font("SansSerif", Font.BOLD, 12);
            plain = new Font("SansSerif", Font.PLAIN, 12);
            setFont(plain);
        }
        public Component getTableCellRendererComponent(JTable jt,
                Object value, boolean isSelected,
                boolean hasFocus, int row, int column) {
        setText((String)value);
        if (row == 1 && column == 1) {
            setFont(bold);
            setForeground(Color.red);
        } else {
            setFont(plain);
```