

构建基于 JAVA 渲染器的用户界面

夏明俊、王万意、王奇

上海远程教育集团

前言

在 Internet 飞速发展的今天，由于 Internet 的信息更新及时快捷，基于浏览器的客户端软件安装，联入方便等等优点，越来越多的企业开始将眼光聚集到基于 B/S 架构的 WEB 应用程序上。其中 J2EE 应用更以其强大的跨平台移植功能深受用户青睐。随着 B/S 结构软件的复杂性不断增加，用户对于 CLIENT 端软件的要求也越来越高。对于界面丰富的浏览器而言，一些以 JAVA APPLET 实现客户端功能，在界面的实现方面就显得相对单调了许多。

本文将阐述如何使用 JAVA SWING 包中的渲染器和编辑器来实现 APPLET 界面中 CELL 元素的个性定制化

1. JAVA 渲染器 Renderer 简介

在 Sun 的官方网站上对 Editor 和 Renderer 的解释是这样的：

Renderer:

Instead, a single cell renderer is generally used to draw all of the cells that contain the same type of data. You can think of the renderer as a configurable ink stamp that the table uses to stamp appropriately formatted data onto each cell. When the user starts to edit a cell's data, a cell editor takes over the cell, controlling the cell's editing behavior.

即在 Cell 中对于不同的数据以不同的格式进行，相当于在 MVC 中根据不同的 Model 选择不同的 View, Renderer 就是给你这个选择的权利。通过 Renderer 你可以在 Table 和 Tree 的 Cell 元素中显示单选框，下拉菜单等多种形式

默认的几种数据的显示方式如下：

Boolean — rendered with a check box.

Number — rendered by a right-aligned label.

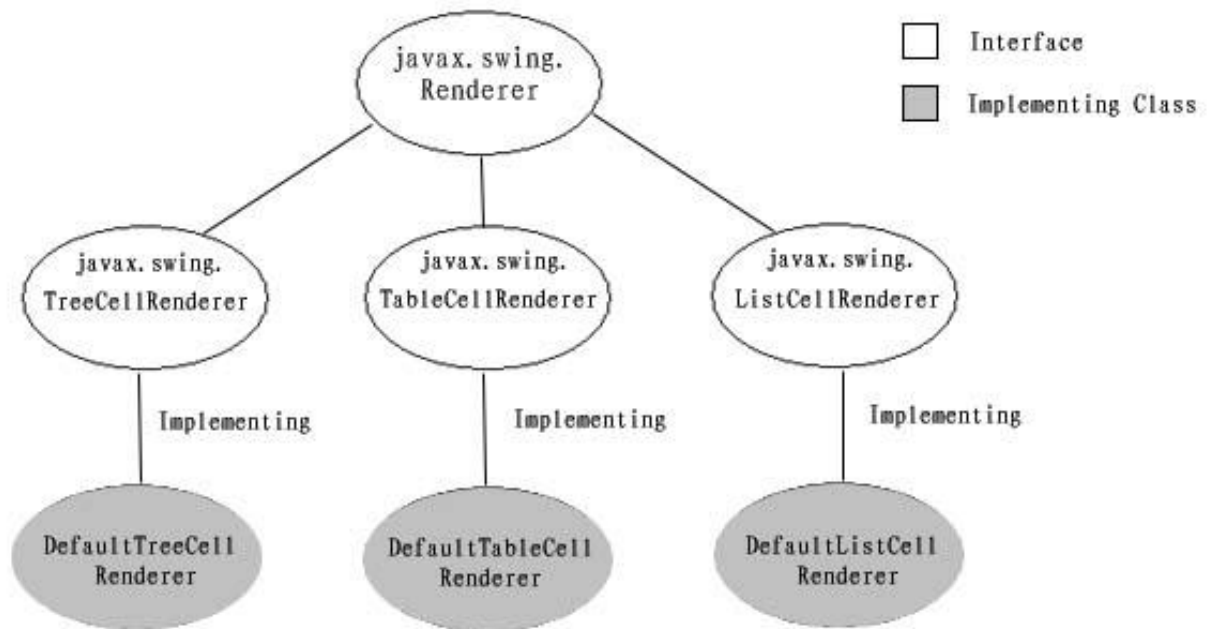
Double, Float — same as Number, but the object-to-text translation is performed by a NumberFormat instance (using the default number format for the current locale).

Date — rendered by a label, with the object-to-text translation performed by a DateFormat instance (using a short style for the date and time).

ImageIcon, Icon — rendered by a centered label.

Object — rendered by a label that displays the object's string value.

2. JAVA SWING 包中渲染器接口的结构



实现了渲染器的三个接口

Interface javax.swing.ListCellRenderer

对于 LIST 的 Cell 元素进行渲染的接口

返回经过渲染的组件的方法: `public Component getListCellRendererComponent(`
`JList list, ----- 需要进行渲染的列表`
`Object value, ----- 需要渲染的单元的值`
`int index, ----- Cell 的 index`
`boolean isSelected, ----- 是否被选中`
`boolean cellHasFocus ----- 是否获得焦点`
`)`

Implementing Class: DefaultListCellRenderer

Interface javax.swing.TableCellRenderer

对于 TABLE 的 Cell 元素进行渲染的接口

返回经过渲染的组件的方法: `public Component getTableCellRendererComponent(`
`JTable table, ----- 需要进行渲染的表格`
`Object value, ----- 需要渲染的单元的值`
`boolean isSelected, ----- 是否被选中`
`boolean hasFocus, ----- 是否获得焦点`
`int row, ----- 表格的行号`
`int column ----- 表格的列号`
`)`

Implementing Class: DefaultTableCellRenderer

Interface javax.swing.TreeCellRenderer

对于 TREE 的 Cell 元素进行渲染的接口

返回经过渲染的组件的方法: `public Component getTreeCellRendererComponent(`
`JTree tree, ----- 需要进行渲染的树`

Object value, ----- 需要渲染的单元的值
boolean selected, ----- 是否被选中
boolean expanded, ----- 是否可以扩展
boolean leaf, ----- 是否是叶 Node
int row, ----- 需要进行渲染的行号
boolean hasFocus ----- 是否获得焦点

3. JAVA 渲染器的实现方法

下面以在 JTREE 中渲染 NODE 为例，阐释 JAVA 渲染器的使用方法

首先编写一个要渲染的类来实现 TreeCellRenderer 接口。根据要渲染的对象，该类可以从 JCheckBox 类或者 JComboBox 类继承，以便继承它们的特性

```
public class CheckBoxTreeCellRenderer extends JCheckBox implements  
TreeCellRenderer
```

然后重写 getTreeCellRendererComponent()的方法，用以描述要渲染的节点的样式。

```
public Component getTreeCellRendererComponent(JTree tree, Object value, boolean  
selected, boolean expanded, boolean leaf, int row, boolean hasFocus)
```

在实现了 CheckBoxTreeCellRenderer 类之后，在 JTree 中找到 setCellRenderer(TreeCellRenderer x)方法，将其设置为刚才编写的 CheckBoxTreeCellRenderer 类。该方法会将 Jtree 中的每一个 Cell 元素（即树上的每个节点）都设置成渲染器接口类描述的样式。

```
tree.setCellRenderer(new CheckBoxTreeCellRenderer())
```

这样就可以将刚才定义的，实现渲染器接口的类定义到了树之中，运行程序之后 Jtree 的每一个节点都会以渲染后的样式展示

对于 TableCellRenderer, ListCellRenderer 的使用，方法基本和 TreeCellRenderer 相同。最主要是对于 getXXCellRendererComponent(....)方法的编写。该方法定义被要渲染的 Cell 的

具体样式。

4. 结束语

JAVA 提供的渲染器可以实现多种界面的渲染功能，比如使 Jtree 如同资源管理器一般展示节点。改变了用 JAVA 实现出的用户界面比较单调的缺点，使用 JAVA 设计出的用户界面更加个性化，具有更强大的功能。

参考文献:

[1] SUN Java™ 2 Platform Std. Ed. v1.4.2 API Specification

[2] JAVA SWING 程序设计

[3] Rendering cells in Swing's JTable component

[4] <http://www-106.ibm.com/developerworks/java/library/j-jt>