## 怎么做一个自定义的ComboBox

### 1.组合框显示字符串

要在组合框里显示一些字符串非常简单，只需要向组合框中放入字符串就可以了

String[] petStrings = { "Bird", "Cat", "Dog", "Rabbit", "Pig" };

JComboBox petList = new JComboBox(petStrings);

### 2.获得用户所选内容

ActonListener listener = new ActionListener(){ //定义一个动作监听器匿名内部类对象

public void actionPerformed(ActionEvent e){

JComboBox cb = (JComboBox)e.getSource(); //获得发生事件的组合框对象

String petName = (String)cb.getSelectedItem();//获得用户所选字符串

updateLabel(petName);//将字符串更新到某个界面组件上

}

};

petList.addActionListener(listener);//将监听器注册到组合框上

### 3.显示图像

需要用到ListCellRenderer

JComboBox petList = new JComboBox(intArray);

. . .

ComboBoxRenderer renderer = new ComboBoxRenderer(); //一个自定义的能显示定制内容的组件

renderer.setPreferredSize(new Dimension(200, 130));//设置最佳尺寸

petList.setRenderer(renderer);//让渲染组件起作用

petList.setMaximumRowCount(3);//设置最大显示多少行

#### 3.1自定义组件

class ComboBoxRenderer extends JLabel //继承标签类

implements ListCellRenderer {//实现ListCellRenderer接口

. . .

public ComboBoxRenderer() {//构造函数

setOpaque(true); //设置透明

setHorizontalAlignment(CENTER);//设置本标签水平居中对齐

setVerticalAlignment(CENTER);//设置本标签垂直居中对齐

}

/\*

\* This method finds the image and text corresponding

\* to the selected value and returns the label, set up

\* to display the text and image.

实现接口，本方法寻找图片和文本，在本标签上渲染出来，并返回本标签。

参数说明：

* JList list — 一个JList对象，用于展示要显示的值，在这个例子里，List中每一个item的背景色和前景色的具体值，是采集的本List的背景色和前景色。
* Object value —要被渲染的值.这个例子里是一个整数.
* int index — 被渲染的值所在位置.
* boolean isSelected — 是否被用户选中，可以根据这个参数来变换显示效果.
* boolean cellHasFocus — 当前值是否获得了焦点，可以根据这个参数来变换显示效果.

\*/

public Component getListCellRendererComponent(

JList list,

Object value,

int index,

boolean isSelected,

boolean cellHasFocus) {

//获取被选中的索引位置，由于参数中的index不是总是有效的，所以使用value

int selectedIndex = ((Integer)value).intValue();

if (isSelected) {//重点，这里是根据用户选中与否，给本标签设置背景色和前景色

setBackground(list.getSelectionBackground());//想让自己的标签显示什么背景色，可以直接指定，不一定非要采集List

setForeground(list.getSelectionForeground());

} else {

setBackground(list.getBackground());

setForeground(list.getForeground());

}

//Set the icon and text. If icon was null, say so.

ImageIcon icon = images[selectedIndex];//这是获取了图片

String pet = petStrings[selectedIndex];//这是获取了文字

setIcon(icon);//在这里将图片放到本标签上

if (icon != null) {

setText(pet);//这里把文字放到标签上

setFont(list.getFont());

} else {

setUhOhText(pet + " (no image available)",

list.getFont());

}

return this;//返回本标签

}

. . .

}

### 4.只显示色块

#### 4.1 定义颜色数组

java.awt.Color[] colors = {Color.red,Color.green,Color.gray};

#### 4.2让渲染组件起作用

JComboBox petList = new JComboBox(colors);

. . .

ComboBoxRenderer renderer = new ComboBoxRenderer(); //一个自定义的能显示定制内容的组件

renderer.setPreferredSize(new Dimension(200, 130));//设置最佳尺寸

petList.setRenderer(renderer);//让渲染组件起作用

petList.setMaximumRowCount(3);//设置最大显示多少行

#### 4.3定义自己的组件

使用画图等工具软件，提前制作一些纯色块的图片，这里假设命名为red.png,green.png和gray.png，将以上文件复制到当前项目文件夹中，稍后会做界面上显示这些图片。

redgreengray

class ComboBoxRenderer extends JLabel implements ListCellRenderer{

//准备 一个map，用于保存提前准备好的图片

public HashMap<Color,Icon> icons = new HashMap<>();

. . .

public ComboBoxRenderer() {//构造函数

setOpaque(true); //设置透明

//setHorizontalAlignment(CENTER);//设置本标签水平居中对齐

//setVerticalAlignment(CENTER);//设置本标签垂直居中对齐

setPreferredSize(new Dimension(100, 20));//设置最佳尺寸

icons.put(Color.green,new ImageIcon("green.png") );//在map中放入提前准备好图片

icons.put(Color.red,new ImageIcon("red.png")); //用Color对象作为索引

icons.put(Color.gray,new ImageIcon("gray.png")); //用Color对象作为索引

}

public Component getListCellRendererComponent(

JList list,

Object value,

int index,

boolean isSelected,

boolean cellHasFocus) {

//获取被选中的索引位置，由于参数中的index不是总是有效的，所以使用value

//value就是被选择的对象,colors中的元素

Color color = (Color) value;//将value转换成Color

setIcon(icons.get(color)); //根据color，从map中提取相应图片，设置背景色

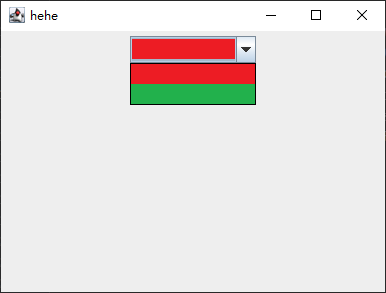
//setText(...);//不需要显示文字

return this;

}

......

}



#### 4.4获取用户选中的元素

ActonListener listener = new ActionListener(){ //定义一个动作监听器匿名内部类对象

public void actionPerformed(ActionEvent e){

JComboBox cb = (JComboBox)e.getSource(); //获得发生事件的组合框对象

Color color = (Color)cb.getSelectedItem();//获得用户所选value,由于构造送的是Color数组，就可以直接获得Color数组

registeColorChanged(color);//将改变过的颜色注册到某个地方，例如控制部分提供一个这样的函数，接收该颜色，并通知画板更新底色

}

};

petList.addActionListener(listener);//将监听器注册到组合框上