



DeepL

订阅DeepL Pro以翻译大型文件。

欲了解更多信息，请访问www.DeepL.com/pro。

JC2002 Java 程序设计

第 8 天：事件驱动编程（人工智能、计算机科学与技术）

JC2002 Java 程序设计

第 8 天，第 1 课时：事件监听器和事件处理程序

参考文献和学习目标

- 今天的课程主要基于 Oracle 文档：
 - <https://docs.oracle.com/en/java/javase/11/docs/api/java.desktop/java/awt/AWTEvent.html>
 - <https://docs.oracle.com/en/java/javase/11/docs/api/java.desktop/java/awt/event/AWTEventListener.html>
- 今天的课程结束后，您应该能够
 - 解释 Java 中事件驱动编程的概念
 - 使用事件侦听器实现交互式图形用户界面元素
 - 为不同的应用程序选择适当的事件监听器

事件监听

- JAVA API 提供了多个 *事件监听器*，可用于响应事件，例如由 swing 组件触发的事件：
 - **ActionListener**：监听由可点击组件（如按钮、组合框、菜单项、切换按钮等）触发的事件。
 - **ItemListener**：监听由实现 ItemSelectable 接口的组件（如复选框、复选菜单项、组合框等）触发的事件。
 - **WindowListener**：监听某些窗口活动后触发的事件，如打开或关闭窗口、聚焦或散焦窗口、最大化窗口等。

行动听众

- 每当执行一个操作时，就会发生一个操作事件。
用户（如点击按钮）
- 要创建动作监听器对象，需要声明一个实现 **ActionListener** 接口的类或扩展一个实现 **ActionListener** 接口的类
 - 通常使用组件的 **addActionListener()** 方法将动作监听器分配给图形用户界面组件；但是，实现细节取决于组件类
 - 当动作事件发生时，程序会执行动作监听器的方法 **actionPerformed()**：您可以在该方法中实现所需的功能

带动作监听器的按钮示例 (1)

```
1  import javax.swing.*;
2  import javax.swing.border.*;
3  导入 java.awt.BorderLayout.BorderLayout.BorderLayout.BorderLayout.BorderLayout;
4  import java.awt.event.*;
5  公共类 ButtonExample2 {
6      protected static JButton b1, b2, b3;
7      protected static JLabel questionLabel, responseLabel;
8      static class ActionHandler implements ActionListener {
9          public void actionPerformed(ActionEvent e) {
10              if(e.getActionCommand().equals("happy")) {
11                  responseLabel.setText("Glad to hear you are happy!");
12              } else if(e.getActionCommand().equals("neutral")) {
13                  responseLabel.setText("感谢您的反馈! ");
14              } else if(e.getActionCommand().equals("unhappy")) {
```

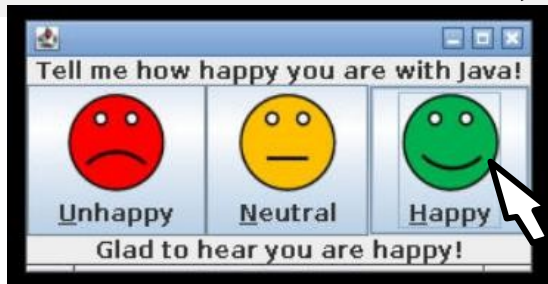
嵌套类（类内类）用于
处理事件

```
15         responseLabel.setText("Sorry to hear you are not happy.");
16     }
17 }
18 }
```

执行规定动作

带动作监听器的按钮示例 (2)

```
...  
25      ...  
      ActionListener actionHandler = new ActionListener();  
...  
51      ...  
52      b1.setActionCommand("unhappy");  
53      b1.addActionListener(actionHandler);  
54      b2.setActionCommand("neutral");  
55      b2.addActionListener(actionHandler);  
56      b3.setActionCommand("happy");  
      b3.addActionListener(actionHandler);  
...      ...
```



设置操作命令和
指定动作监听器
对象到按钮

项目监听器

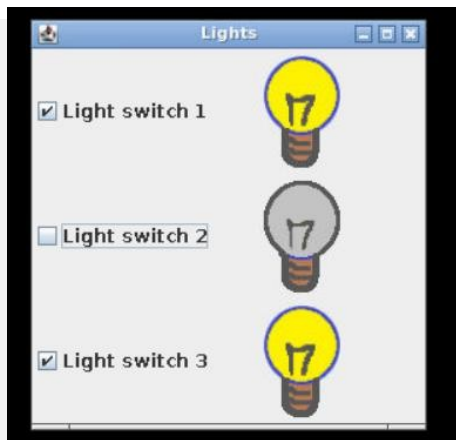
- 点击复选框时会发生项目事件
- 要创建一个项目监听器对象，需要声明一个实现 **ItemListener** 接口的类或扩展一个实现 **ItemListener** 接口的类
 - 当项目事件发生时，程序会执行方法 **itemStateChanged(ItemEvent e)**
 - 您可以通过调用 **ItemEvent** 方法 **getStateChange()** 来测试项目是否已选中或取消选中，该方法可以是 **ItemEvent.SELECTED** 或

ItemEvent.DESELECTED

- 您可以使用 **getSource()** 方法获取事件源按钮

带有项目监听器的复选框示例

```
...
3
4 import java.awt.event.*;
...
公共类 CheckBoxExample2 {
22
...
23     ItemListener itemHandler = new ItemListener() {
24         public void itemStateChanged(ItemEvent e) {
25             for(int i=0; i<3; i++) {
26                 if(e.getSource()==cb[i]) {
27                     if(e.getStateChange() == ItemEvent.SELECTED) {
28                         lights[i].setIcon(lightOnIcon);
29                     } else {
30                         lights[i].setIcon(lightOffIcon);
31                     }
32                 }
            }
        }
    }
}
```



...

```

    }
};
for(int i=0; i<3; i++)
{
    cb[i].addItemListener(
        itemStateChangeListener);
}

```

匿名类实现
itemStateChangeListener

ged() 被定义为项目监听器

多个事件监听器

- 在前面的示例中，我们使用了一个嵌套类或匿名类作为事件监听器
 - 只需要简单功能时也能正常工作
- 还可以为不同组件实现多个事件监听器

单选按钮事件

提交按钮事件



带事件的单选按钮示例 (1)

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

```
submit.setEnabled(true);
```

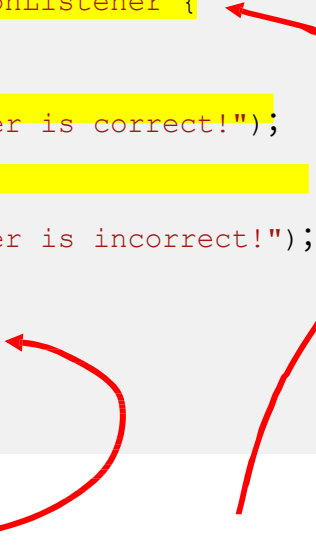
组件移至此处，以便事件监听器可以访问它们



单选按钮的 ActionListener（仅启用提交按钮，以防该按钮尚未启用）

带事件的单选按钮示例 (2)

```
18 protected static class SubmitListener implements ActionListener {
19     public void actionPerformed(ActionEvent e) {
20         if(rb[1].isSelected()) {
21             JOptionPane.showMessageDialog(frame, "Your answer is correct!");
22             否则 {
23                 JOptionPane.showMessageDialog(frame, "Your answer is incorrect!");
24             }
25             frame.dispatchEvent(new WindowEvent(frame,
26                                     WindowEvent.WINDOW_CLOSING));
27         }
28     }
29     protected static SubmitListener submitListener;
30     protected static RadioListener radioListener;
```



显示信息对话框后关闭
框架并退出程序

提交按钮的 ActionListener (检查答
案是否正确)

带事件的单选按钮示例 (3)

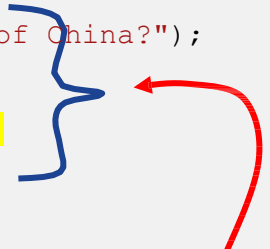
```
31 public static void main(String[] args) {  
32     frame = new JFrame();  
33     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
34     panel = new JPanel();  
35     BorderLayout boxlayout = new BorderLayout(panel, BorderLayout.Y_AXIS);  
36     panel.setLayout(boxlayout);  
37     question = new JLabel("What is the capital of China?");  
38     panel.add(question);  
39     submit = new JButton("Submit your answer");  
40     submit.setEnabled(false);  
41     submitListener = new SubmitListener()  
42     ;  
43     submit.addActionListener(submitListener);  
44     rb = new JRadioButton[4];  
45     rb[0] = 新的 JRadioButton ("上海");
```

初始化窗口
和布局

```
rb[1] = 新的 JRadioButton ("北京") ;  
rb[2] = 新的 JRadioButton ("广州") ;  
rb[3] = 新的 JRadioButton ("重庆") ;
```

带事件的单选按钮示例 (4)

```
31 public static void main(String[] args) {
32     frame = new JFrame();
33     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
34     panel = new JPanel();
35     BorderLayout boxlayout = new BorderLayout(panel, BorderLayout.Y_AXIS);
36     panel.setLayout(boxlayout);
37     question = new JLabel("What is the capital of China?");
38     panel.add(question);
39     submit = new JButton("Submit your answer");
40     submit.setEnabled(false);
41     submitListener = new SubmitListener()
42     ;
43     submit.addActionListener(submitListener);
44     rb = new JRadioButton[4];
45     rb[0] = 新的 JRadioButton ("上海");
```



```
rb[1] = 新的 JRadioButton ("北京") ;  
rb[2] = 新的 JRadioButton ("广州") ;  
rb[3] = 新的 JRadioButton ("重庆") ;
```

带事件的单选按钮示例 (5)

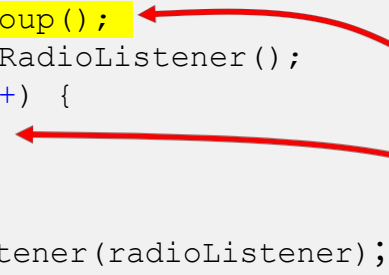
```
31 public static void main(String[] args) {  
32     frame = new JFrame();  
33     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
34     panel = new JPanel("Quiz");  
35     BoxLayout boxlayout = new BoxLayout(panel, BoxLayout.Y_AXIS);  
36     panel.setLayout(boxlayout);  
37     question = new JLabel("What is the capital of China?");  
38     panel.add(question);  
39     submit = new JButton("Submit your answer");  
40     submit.setEnabled(false);  
41     submitListener = new SubmitListener();  
    submit.addActionListener(submitListener);
```

创建单选按钮

```
43     rb = new JRadioButton[4];
44     rb[0] = new JRadioButton("Shanghai");
45     rb[1] = new JRadioButton("Beijing");
46     rb[2] = new JRadioButton("Guangzhou");
47     rb[3] = new JRadioButton("Chongqing");
```

带事件的单选按钮示例 (6)

```
48 group = new ButtonGroup();
49 radioListener = new RadioListener();
50 for(int i=0; i<4; i++) {
51     group.add(rb[i]);
52     panel.add(rb[i]);
53     rb[i].addActionListener(radioListener);
54 }
55 panel.add(submit);
56 frame.add(panel);
57 frame.pack();
58 }
59 frame.setVisible(true);
60 }
```



为按钮组指定单选按钮

带事件的单选按钮示例 (7)

```
48     group = new ButtonGroup();
49     radioListener = new RadioListener();
50     for(int i=0; i<4; i++) {
51         group.add(rb[i]);
52         panel.add(rb[i]);
53         rb[i].addActionListener(radioListener);
54     }
55     panel.add(submit);
56     frame.add(panel);
57     frame.pack();
58     frame.setVisible(true);
59 }
60 }
```

为动作监听器指定单
选按钮

带事件的单选按钮示例 (8)

```
48     group = new ButtonGroup();
49     radioListener = new RadioListener();
50     for(int i=0; i<4; i++) {
51         group.add(rb[i]);
52         panel.add(rb[i]);
53     }
54     rb[i].addActionListener(radioListener);
55 }
56 panel.add(submit);
57 frame.add(panel);
58 frame.pack();
59 }
60 frame.setVisible(true);
```

在面板上添加单选按钮
并最终确定视图



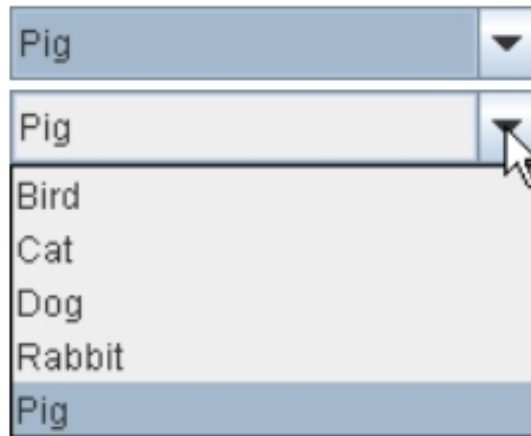
有问题或意见？

JC2002 Java 程序设计

第 8 天，第 2 课：更多事件处理程序示例

组合框JComboBox

- 组合框可让用户选择
从下拉列表中选择多个选项
 - 在 Swing 中，用
JComboBox 组件
- 组合框事件是 `ActionEvents`
- 组合框的两种主要形式
不可编辑和可编辑



参考资料：

组合框示例：视图初始化

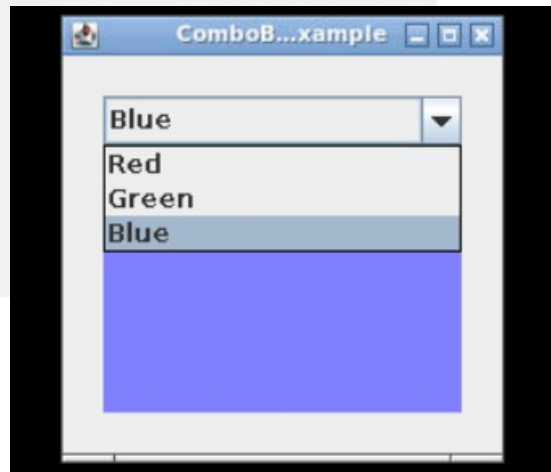
```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4  public class ComboBoxExample extends JPanel
5      实现 ActionListener {
6      JLabel 图片;
7      public ComboBoxExample() {
8          super(new BorderLayout());
9          String[] colorStrings = { "红", "绿", "蓝" };
10         JComboBox<String> colorList = new JComboBox<>(colorStrings);
11         colorList.setSelectedIndex(0);
12         colorList.addActionListener(this);
13         picture = new JLabel();
14         picture.setOpaque(true);
```

```
15     picture.setBackground(new Color(255, 128, 128));
16     picture.setBorder(BorderFactory.createEmptyBorder(10,0,0,0));
17     picture.setPreferredSize(new Dimension(177, 122+10));
18     add(colorList, BorderLayout.PAGE_START);
19     添加 (图片, BorderLayout.PAGE_END) ;
20     setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
21 }
```

组合框示例：事件监听器

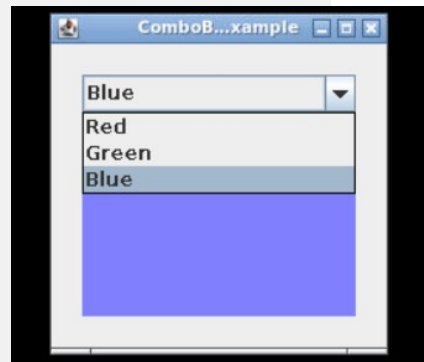
```
22
23 public void actionPerformed(ActionEvent e) {
24     JComboBox cb = (JComboBox)e.getSource();
25     字符串 colorName = (String)cb.getSelectedItem();
26     if(colorName.equals("Red"))
27         picture.setBackground(new Color(255, 128, 128));
28     else if(colorName.equals("Green"))
29         picture.setBackground(new Color(128, 255, 128));
30     else if(colorName.equals("Blue"))
31         picture.setBackground(new Color(128, 128, 255));
32 }
```

- 组合框事件监听器
- 更改标签颜色



组合框示例：入口点

```
33
34 私人静态 void createAndShowGUI() {
35      JFrame frame = new JFrame("ComboBoxExample");
36      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
37      JComponent newContentPane = new ComboBoxExample();
38      newContentPane.setOpaque(true);
39      frame.setContentPane(newContentPane);
40      frame.pack();
41      frame.setVisible(true);
42  }
43
44
45  public static void main(String[] args) {
46      javax.swing.SwingUtilities.invokeLater(new Runnable() {
47          public void run() {
48              createAndShowGUI();
49          }
50      });
51  }
```



滑块JSlider

- 滑块可用于提供有最小值和最大值限制的数值输入
 - 在 Swing 中，使用 **JS 滑块** 组件
 - JSlider 事件是 **ChangeEvent** 变化监听器监听的对象



参考资料：<https://docs.oracle.com/javase/tutorial/uiswing/components/slider.html>

滑块示例：视图初始化

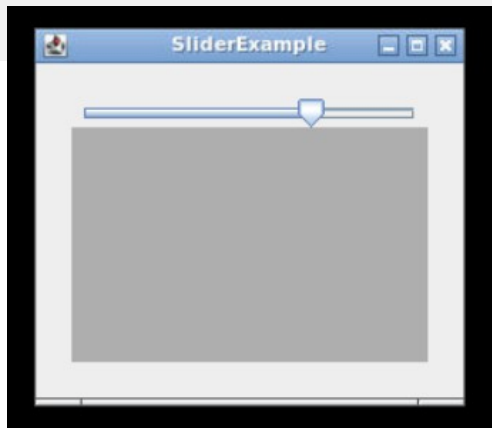
```
1  import java.awt.*;
2  import javax.swing.event.*;
3  import javax.swing.*;
4  公共类 SliderExample 扩展 JPanel
5                                  实现 ChangeListener {
6      JLabel 图片;
7      public SliderExample() {
8          super(new BorderLayout());
9          JSlider brightness = new JSlider(JSlider.HORIZONTAL, 0, 255, 128);
10         brightness.addChangeListener(this);
11         picture = new JLabel();
12         picture.setOpaque(true);
13         picture.setBackground(new Color(128, 128, 128));
```

```
14     picture.setBorder(BorderFactory.createEmptyBorder(10,0,0,0));
15     picture.setPreferredSize(new Dimension(177, 122+10));
16     添加（亮度, BorderLayout.PAGE_START) ;
17     添加（图片, BorderLayout.PAGE_END) ;
18     setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
19 }
```

滑块示例：事件监听器

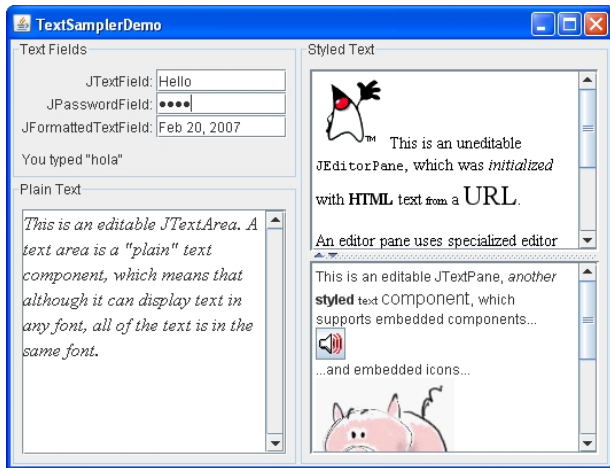
```
20  
21 public void stateChanged(ChangeEvent e) {  
22     JSlider slider = (JSlider)e.getSource();  
23     int value = (int)slider.getValue();  
24     picture.setBackground(new Color(value,value,value));  
25     ...  
    ...  
}
```

- 滑块事件监听器
(变更事件)
- 改变标签亮度



文本字段

- 文本组件，如 JTextArea 和 JTextPane 都可以编辑
 - 必要时，可使用方法 **getText()** 获取编辑过的文本
 - 您可以为文本组件添加 ActionListener，但通常没有必要



参考文献：<https://docs.oracle.com/javase/tutorial/uiswing/components/textfield.html>

<https://docs.oracle.com/javase/tutorial/uiswing/components/editorpane.html>

带交互功能的文本字段示例

```
...
4 public class TextFieldExample extends JPanel
5                                     实现 ActionListener {
6
7     JTextField textField;
8     JButton submitButton;
9     JLabel output;
10
11     public TextFieldExample() {
12         super(new BorderLayout(10,10));
13         textField = new JTextField(30);
14         submitButton = new JButton("Submit
15 text");
16         submitButton.addActionListener(this);
17         output = new JLabel(" "); add(textField,
18 BorderLayout.PAGE_START);
19 add(submitButton, BorderLayout.CENTER);
20 add(output, BorderLayout.PAGE_END);
21
22         setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
23     }
24     public void actionPerformed(ActionEvent e) {
25         output.setText(textField.getText());
26     }
27 }
```

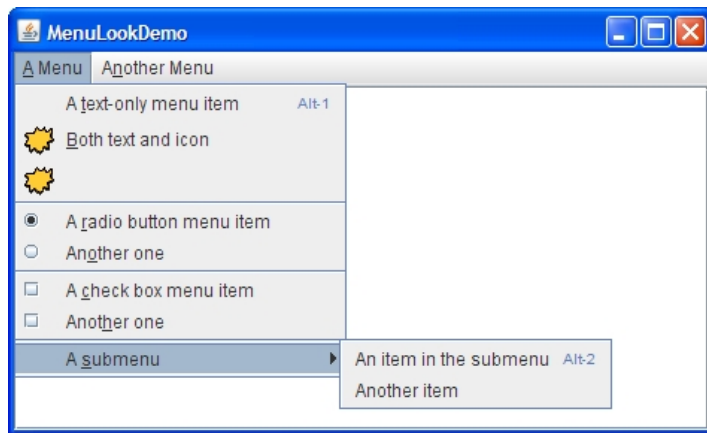


段中写入的文本

输出
文本
字

为菜单项添加功能

- 昨天，我们演示了如何为菜单栏添加项目
 - 还可以在菜单中使用图标、单选按钮和复选框
- 我们可以使用 ActionListener 来处理来自菜单项的事件，就像来自按钮的事件一样



参考资料

<https://docs.oracle.com/javase/tutorial/uiswing/components/menu.html>

菜单示例：视图初始化器

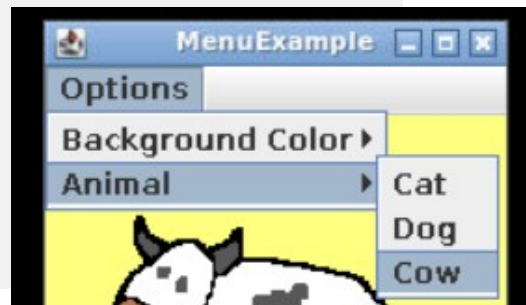
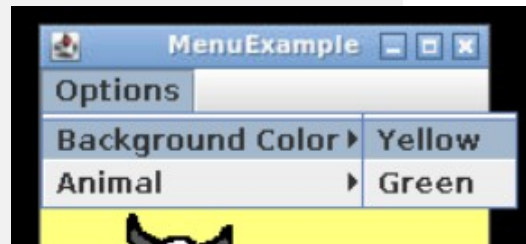
```
1  import javax.swing.*;
2  import java.awt.*;
3
4  import java.awt.event.*;
5  类 MenuExample2 实现 ActionListener {
6      JMenu menu, submenu1, submenu2;
7      JMenuItem i1, i2, i3, i4, i5;
8
9      JLabel label;
10
11     ImageIcon catIcon、 dogIcon、 cowIcon;
12     MenuExample2() {
13         JFrame f= new JFrame("MenuExample");
14         f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
15         JMenuBar mb=new JMenuBar();
16
17         menu=new JMenu("Options");
18         submenu1=new JMenu("Background Color");
```

初始化菜单
和子菜单

```
submenu2=new JMenu("Animal");
```

菜单示例：创建菜单项

```
16         i1=new JMenuItem("Yellow");
17         i2=new JMenuItem("Green");
18         i3=new JMenuItem("Cat");
19         i4=new JMenuItem("Dog");
20         i5=new JMenuItem("Cow");
21         submenu1.add(i1);
22         submenu1.add(i2);
23         submenu2.add(i3);
24         submenu2.add(i4);
25         submenu2.add(i5);
26         menu.add(submenu1);
27         menu.add(submenu2);
28
29         mb.add(menu);
30         f.setJMenuBar(mb);
```



菜单示例：最终确定视图

```
31     label = new JLabel();
32     catIcon = new ImageIcon("cat.png");
33     dogIcon = new ImageIcon("dog.png");
34     cowIcon = new ImageIcon("cow.png");
35     label.setOpaque(true);
36     label.setBackground(new Color(255,255,255));
37     label.setPreferredSize(new Dimension(200, 200));
38     label.setIcon(catIcon);
39     f.getContentPane().add(label, BorderLayout.CENTER);
40
41     i1.addActionListener(this);
42     i2.addActionListener(this);
43
44
```



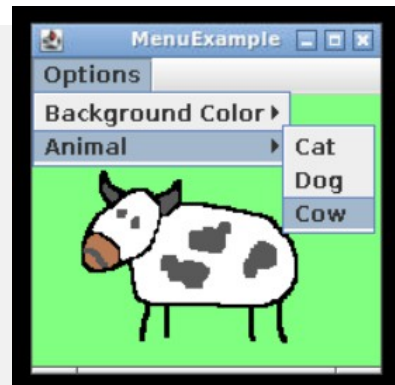
A diagram consisting of a yellow rectangle with a red border. A red arrow originates from the right side of this rectangle and points horizontally to the left, ending at the `BorderLayout.CENTER` parameter in line 39 of the code. A blue curly brace is positioned to the left of the code, spanning from line 37 to line 39, grouping the `label.setPreferredSize` and `label.setIcon` lines with the `add` line.

```
46         i3.addActionListener(this);  
47         i4.addActionListener(this);  
48         i5.addActionListener(this)  
49     }  
;  
  
f.setSize(200,200);  
f.setVisible(true);
```

指定行动
监听器
菜单项

菜单示例：动作监听器

```
50 public void actionPerformed(ActionEvent e) {  
51     if((JMenuItem)e.getSource()==i1)  
52         label.setBackground(new Color(255, 255, 128));  
53     else if((JMenuItem)e.getSource()==i2)  
54         label.setBackground(new Color(128, 255, 128));  
55     else if((JMenuItem)e.getSource()==i3)  
56         label.setIcon(catIcon);  
57     else if((JMenuItem)e.getSource()==i4)  
58         label.setIcon(dogIcon);  
59     else if((JMenuItem)e.getSource()==i5)  
60         label.setIcon(cowIcon);  
61     }  
62 }  
63 public static void main(String args[]){  
64     new MenuExample2();  
65 }
```



子菜单 1：更改
背景颜色

子菜单 2：更改动

更多阅读

- Swing 中定义了大量的图形用户界面组件、布局等，我们只介绍了其中最重要的几种
- 有关更多示例，请参阅 Java 官方文档：
 - 组件：
<https://docs.oracle.com/javase/tutorial/uiswing/components/index>
 - 事件监听器：
<https://docs.oracle.com/javase/tutorial/uiswing/events/index.html>

摘要

- 在 Java 中，交互式图形用户界面组件是通过事件实现接口 `EventListener` 或其子接口的监听器
 - 可为图形用户界面组件分配事件侦听器，以便对用户操作（如单击按钮或选择菜单项）做出反应
- 实现事件侦听器的正确方法取决于组成部分
 - 例如，按钮和菜单项使用 `ActionListener` 接口，滑块使用 `ChangeListener` 接口

- 有关具体组件的详细信息，请参阅文档

有问题或意见？