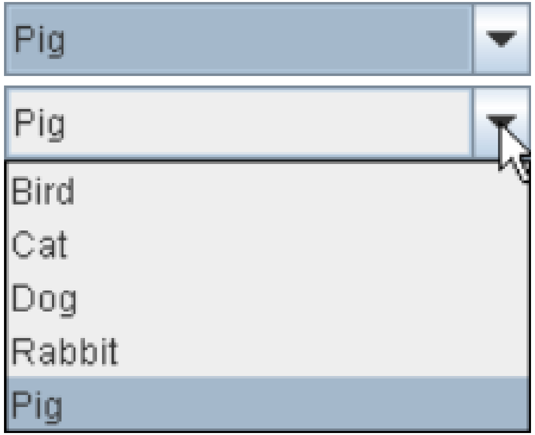
1495 UNIVERSITY OF ABERDEEN

JC2002 Java Programming

Lecture 24: More examples of event handlers

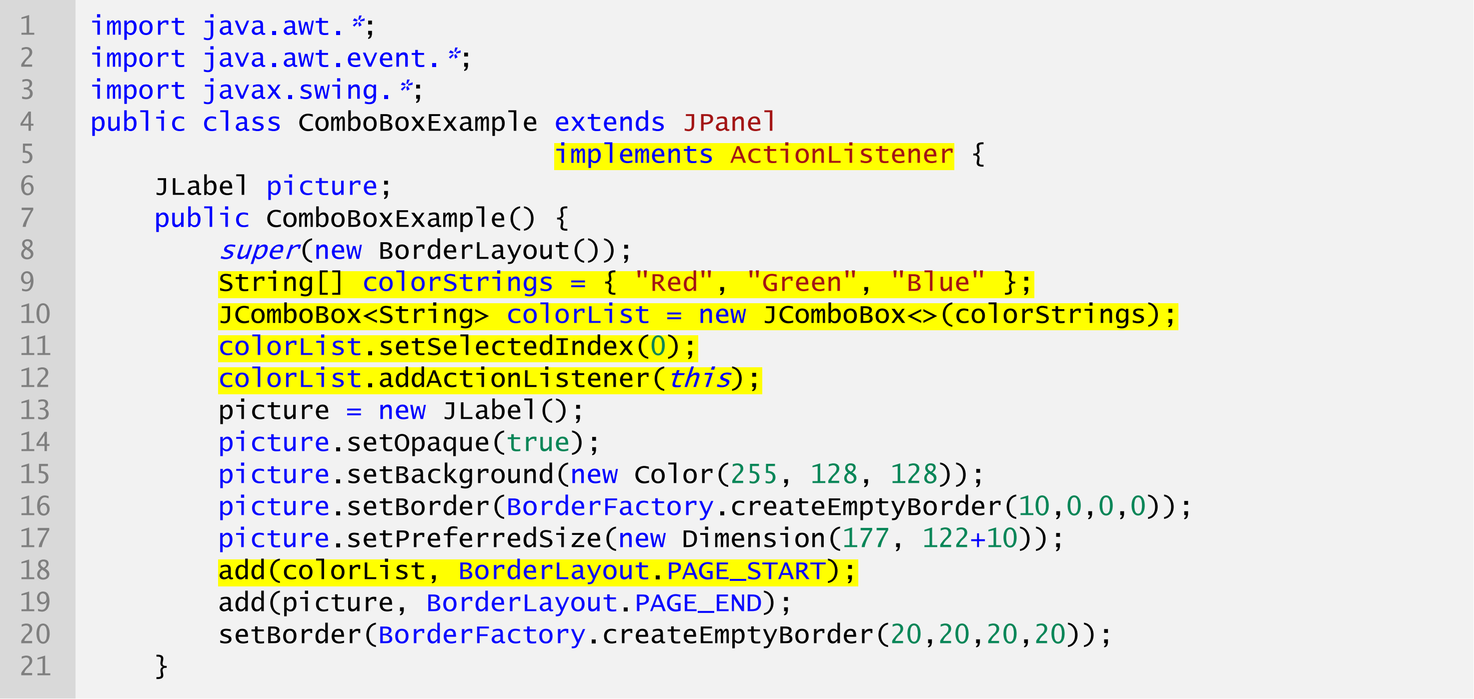
Combo boxes: JComboBox

- Combo boxes let the user choose one of several choices from a dropdown list
  - In Swing, combo boxes created with **`JComboBox`** component
  - Combo box events are **`ActionEvent`**s
- Two main forms of combo boxes: uneditable and editable



Reference:
https://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html

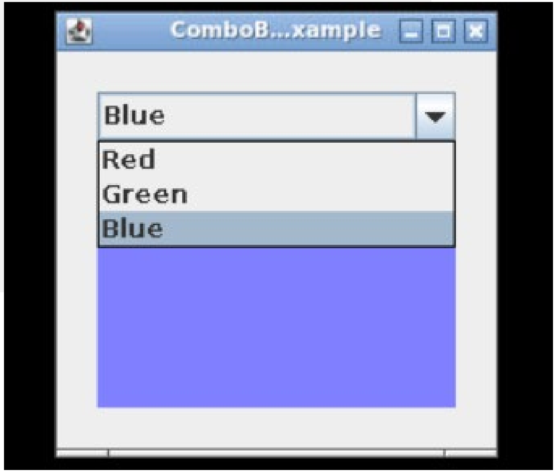
Combo box example: view initialisation



```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class ComboBoxExample extends JPanel
                             implements ActionListener {
    JLabel picture;
    public ComboBoxExample() {
        super(new BorderLayout());
        String[] colorStrings = { "Red", "Green", "Blue" };
        JComboBox<String> colorList = new JComboBox<>(colorStrings);
        colorList.setSelectedIndex(0);
        colorList.addActionListener(this);
        picture = new JLabel();
        picture.setOpaque(true);
        picture.setBackground(new Color(255, 128, 128));
        picture.setBorder(BorderFactory.createEmptyBorder(10,0,0,0));
        picture.setPreferredSize(new Dimension(177, 122+10));
        add(colorList, BorderLayout.PAGE_START);
        add(picture, BorderLayout.PAGE_END);
        setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
    }
```

Combo box example: event listener

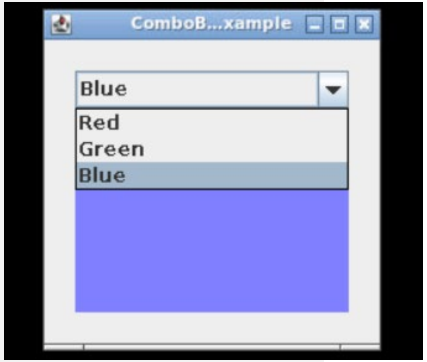
```java
public void actionPerformed(ActionEvent e) {
  JComboBox cb = (JComboBox)e.getSource();
  String colorName = (String)cb.getSelectedItem();
  if(colorName.equals("Red"))
    picture.setBackground(new Color(255, 128, 128));
  else if(colorName.equals("Green"))
    picture.setBackground(new Color(128, 255, 128));
  else if(colorName.equals("Blue"))
    picture.setBackground(new Color(128, 128, 255));
}
```



- Listener for the combo box events
- Changes color of the label

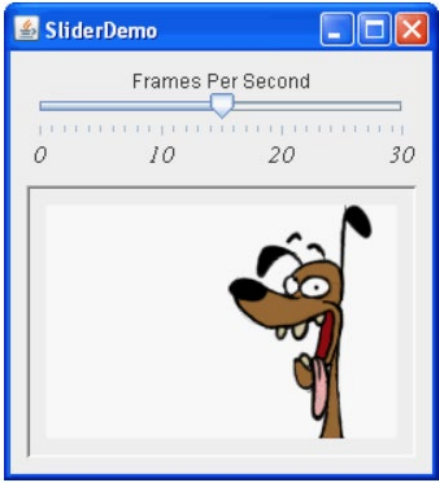
Combo box example: entry point

```java

    private static void createAndShowGUI() {
        JFrame frame = new JFrame("ComboBoxExample");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JComponent newContentPane = new ComboBoxExample();
        newContentPane.setOpaque(true);
        frame.setContentPane(newContentPane);
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```



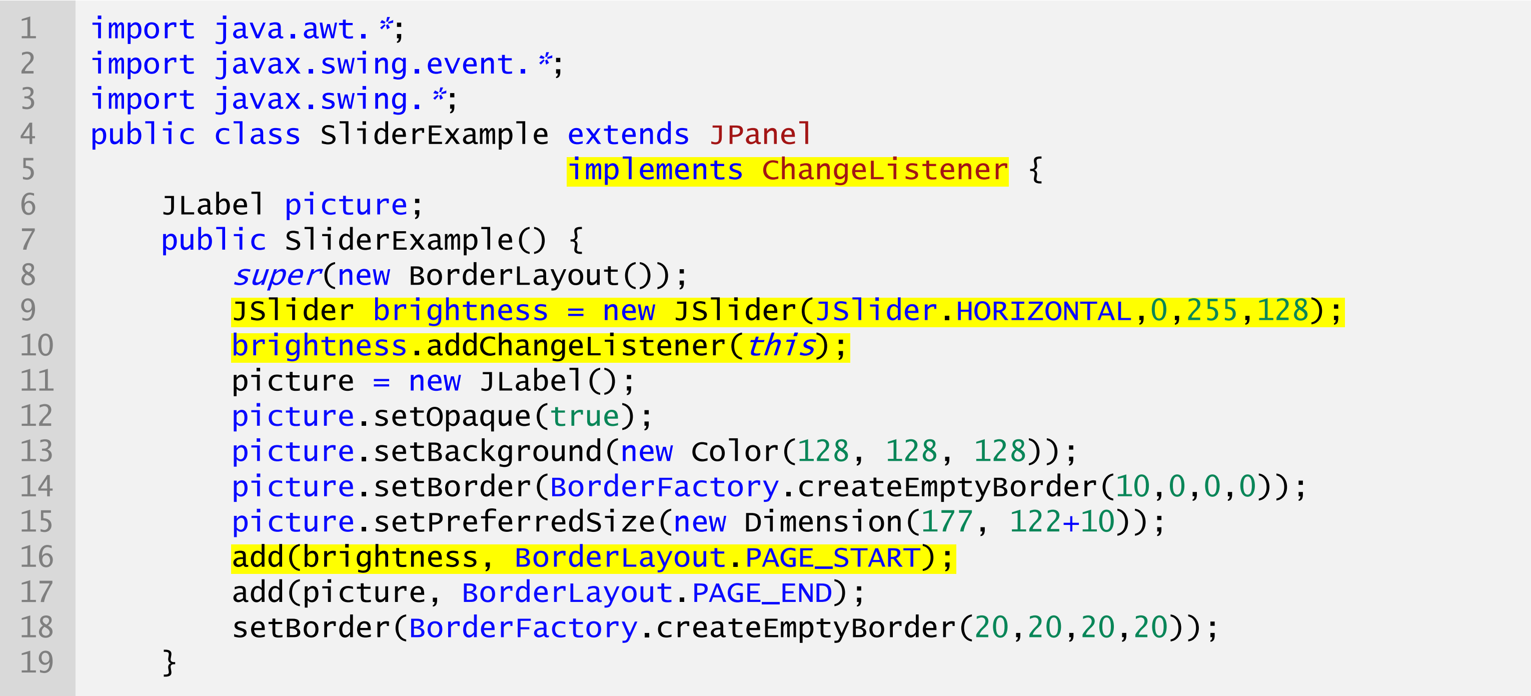
Sliders: JSlider

- Sliders can be used to give numerical input bounded with minimum and maximum value
  - In Swing, sliders created using **`JSlider`** component
  - `JSlider` events are **`ChangeEvent`** objects listened by **`ChangeListener`**



Reference:
https://docs.oracle.com/javase/tutorial/uiswing/components/slider.html

Slider example: view initialisation

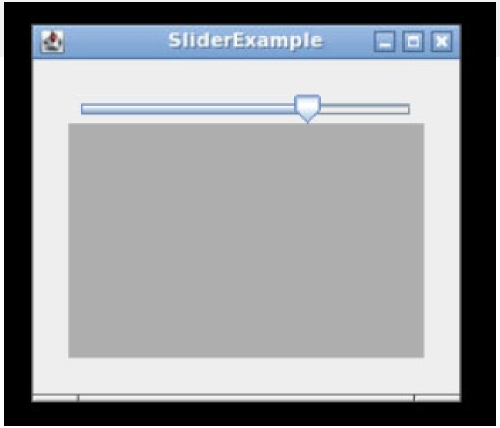


```
import java.awt.*;
import javax.swing.event.*;
import javax.swing.*;
public class SliderExample extends JPanel
                           implements ChangeListener {
    JLabel picture;
    public SliderExample() {
        super(new BorderLayout());
        JSlider brightness = new JSlider(JSlider.HORIZONTAL,0,255,128);
        brightness.addChangeListener(this);
        picture = new JLabel();
        picture.setOpaque(true);
        picture.setBackground(new Color(128, 128, 128));
        picture.setBorder(BorderFactory.createEmptyBorder(10,0,0,0));
        picture.setPreferredSize(new Dimension(177, 122+10));
        add(brightness, BorderLayout.PAGE_START);
        add(picture, BorderLayout.PAGE_END);
        setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
    }
```

Slider example: event listener

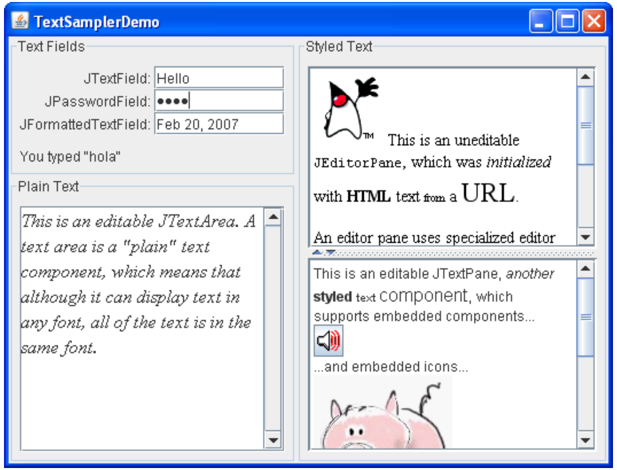
```

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

- Listener for the slider events (ChangeEvent)
- Changes brightness of the label



Text fields

- Text components, such as `JTextArea` and `JTextPane`, can be editable
  - Method **`getText()`** can be used to get the edited text when needed
  - You can add an `actionListener` to a text component, but it is not usually necessary



JTextField: Hello

JPasswordField: ••••

JFormattedTextField: Feb 20, 2007

You typed "hola"

This is an editable JTextArea. A text area is a "plain" text component, which means that although it can display text in any font, all of the text is in the same font.

This is an uneditable `JEditorPane`, which was *initialized* with **HTML** text from a URL.

An editor pane uses specialized editor

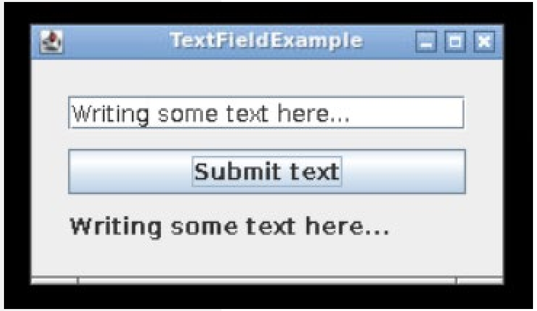
This is an editable JTextPane, *another* **styled** text component, which supports embedded components...

...and embedded icons...

References:
https://docs.oracle.com/javase/tutorial/uiswing/components/textfield.html
https://docs.oracle.com/javase/tutorial/uiswing/components/editorpane.html

Text field example with interaction

```
...
public class TextFieldExample extends JPanel
                             implements ActionListener {
    JTextField textField;
    JButton submitButton;
    JLabel output;
    public TextFieldExample() {
        super(new BorderLayout(10,10));
        textField = new JTextField(30);
        submitButton = new JButton("Submit text");
        submitButton.addActionListener(this);
        output = new JLabel("  ");
        add(textField, BorderLayout.PAGE_START);
        add(submitButton, BorderLayout.CENTER);
        add(output, BorderLayout.PAGE_END);
        setBorder(BorderFactory.createEmptyBorder(20,20,20,20));
    }
    public void actionPerformed(ActionEvent e) {
        output.setText(textField.getText());
    }
    ...
```



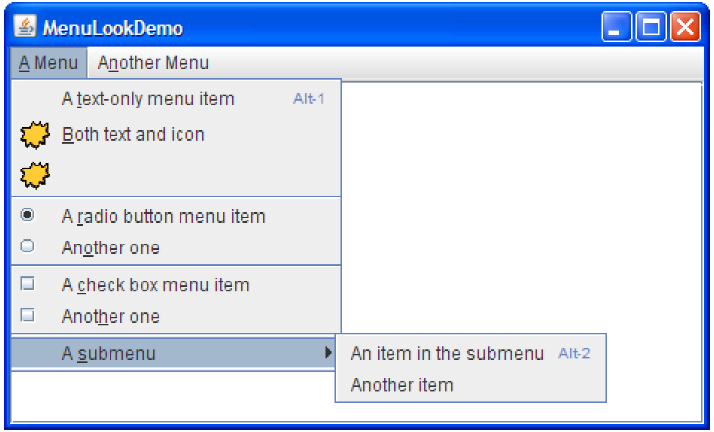
Writing some text here...

Writing some text here...

Outputs the text written in text field

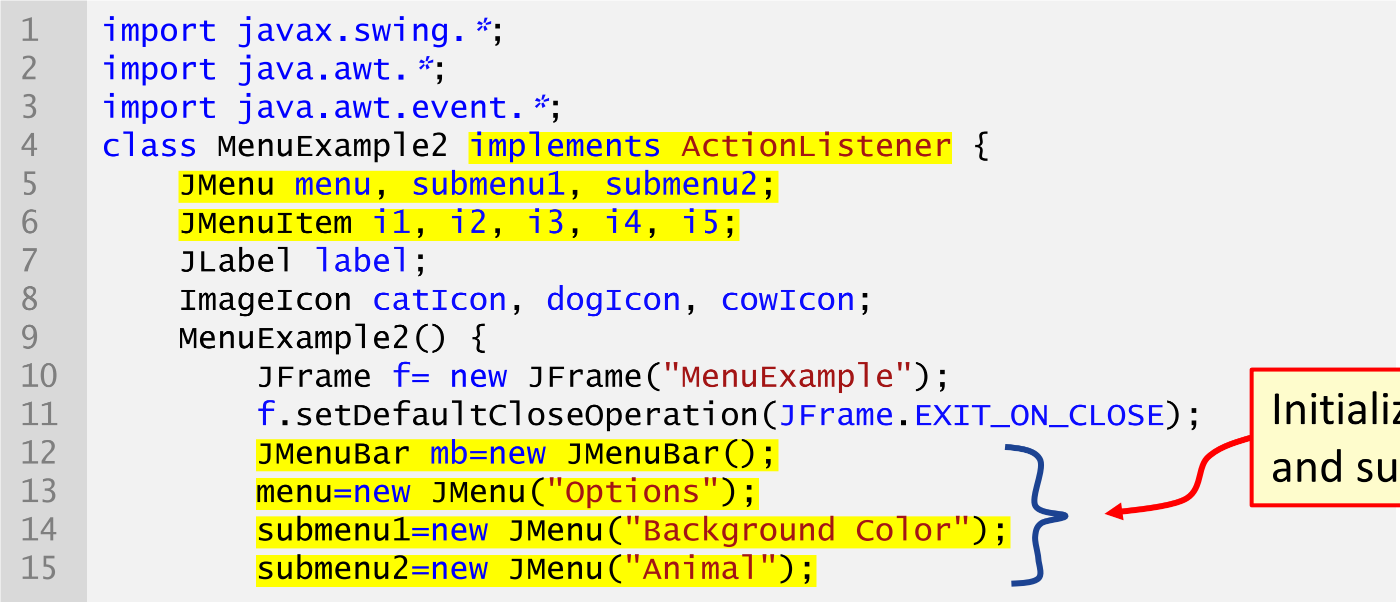
Adding functionality to menu items

- Yesterday, we demonstrated how to add items to the menu bar
  - It is also possible to use icons, radio buttons and check boxes in menus
- We can use `ActionListener` to handle events from menu items, just like from buttons



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

Menu example: view initialiser

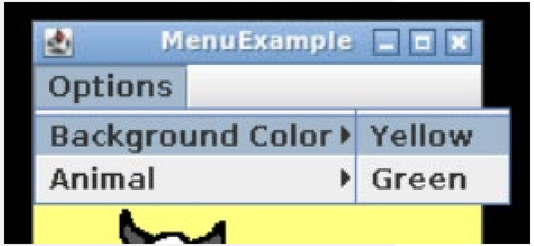


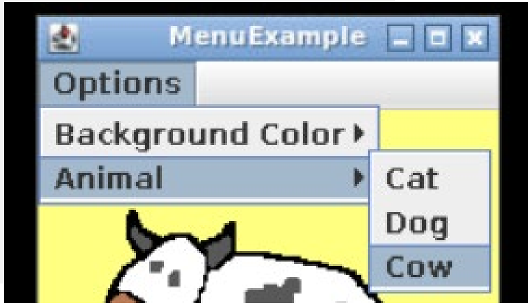
```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class MenuExample2 implements ActionListener {
    JMenu menu, submenu1, submenu2;
    JMenuItem i1, i2, i3, i4, i5;
    JLabel label;
    ImageIcon catIcon, dogIcon, cowIcon;
    MenuExample2() {
        JFrame f= new JFrame("MenuExample");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JMenuBar mb=new JMenuBar();
        menu=new JMenu("Options");
        submenu1=new JMenu("Background Color");
        submenu2=new JMenu("Animal");
```

Initialize menu and submenus

Menu example: create menu items

```
        i1=new JMenuItem("Yellow");
        i2=new JMenuItem("Green");
        i3=new JMenuItem("Cat");
        i4=new JMenuItem("Dog");
        i5=new JMenuItem("Cow");
        submenu1.add(i1);
        submenu1.add(i2);
        submenu2.add(i3);
        submenu2.add(i4);
        submenu2.add(i5);
        menu.add(submenu1);
        menu.add(submenu2);
        mb.add(menu);
        f.setJMenuBar(mb);
```





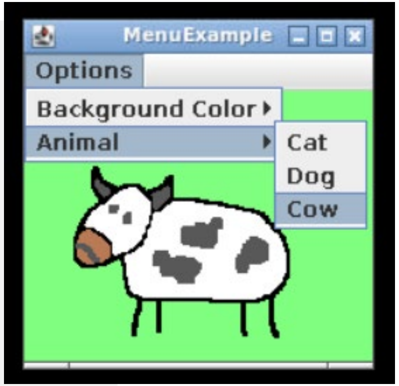
Menu example: finalise view

```
        label = new JLabel();
        catIcon = new ImageIcon("cat.png");
        dogIcon = new ImageIcon("dog.png");
        cowIcon = new ImageIcon("cow.png");
        label.setOpaque(true);
        label.setBackground(new Color(255,255,255));
        label.setPreferredSize(new Dimension(200, 200));
        label.setIcon(catIcon);
        f.getContentPane().add(label, BorderLayout.CENTER);

        i1.addActionListener(this);
        i2.addActionListener(this);
        i3.addActionListener(this);
        i4.addActionListener(this);
        i5.addActionListener(this);

        f.setSize(200,200);
        f.setVisible(true);
    }
```

Assign action listener to the menu items

Menu example: action listener

```java
  public void actionPerformed(ActionEvent e) {
    if((JMenuItem)e.getSource()==i1)
      label.setBackground(new Color(255, 255, 128));
    else if((JMenuItem)e.getSource()==i2)
      label.setBackground(new Color(128, 255, 128));
    else if((JMenuItem)e.getSource()==i3)
      label.setIcon(catIcon);
    else if((JMenuItem)e.getSource()==i4)
      label.setIcon(dogIcon);
    else if((JMenuItem)e.getSource()==i5)
      label.setIcon(cowIcon);
  }
  public static void main(String args[]){
    new MenuExample2();
  }
}
```



Submenu 1: change background colour

Submenu 2: change animal icon

Further reading

- There is an enormous number of GUI components, layouts etc. defined in Swing, and we have only covered some of the most important ones
- For more examples, see the official Java documentation:
  - Components: https://docs.oracle.com/javase/tutorial/uiswing/components/index
  - Event listeners: https://docs.oracle.com/javase/tutorial/uiswing/events/index.html

Summary

- In Java, interactive GUI components are implemented using event listeners implementing interface `EventListener` or its subinterface
  - Event listeners can be assigned to GUI components to react to user actions, such as clicking a button or selecting a menu item
- The proper way to implement event listeners depends on the component
  - For example, buttons and menu items use `ActionListener` interface, sliders use `ChangeListener` interface
  - Refer to the documentation for more details about specific components

Questions, comments?