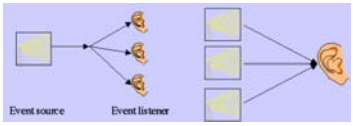
2D Spiele Programmieren in Java

Teil 4: Interaktionen

Dr. Katja Wegner
Dr. Ursula Rost

Spielerinteraktion

- Event listener
  - wird einer Komponente hinzugefügt
  - wartet auf Ereignisse von dieser Komponente:
    - Mausklicks
    - Tastatureingaben
  - aufgrund dieser Ereignisse Anweisungen ausführen



Listeners

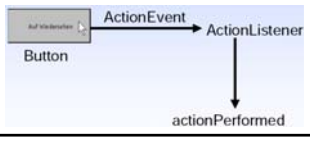
Aktion, die das Ereignis auslöst	Listener Typ
Benutzer drückt Knopf oder Return nach der Eingabe in ein Textfeld oder nach Auswahl eines Menüpunktes	*ActionListener*
Benutzer schließt ein Fenster	*WindowListener*
Benutzer drückt Mausknopf während der Mauszeiger sich über einer Komponente befindet	*MouseListener*
Benutzer bewegt die Maus über eine Komponente	*MouseMotionListener*
Komponente wird angezeigt	*ComponentListener*
Komponente bekommt Tastaturfokus	*FocusListener*
Auswahl einer Tabelle oder Liste wird geändert	*ListSelectionListener*

Listeners (2)

Komponente	Event	Listener
Window	WindowEvent	WindowListener
TextField, TextArea	TextEvent	TextListener
Checkbox, Choice, List	ItemEvent	ItemListener
Button, TextField, List, MenuItem	ActionEvent	ActionListener
Component	ComponentEvent, FocusEvent, KeyEvent, MouseEvent	ComponentListener, FocusListener, KeyListener, MouseListener, MouseMotionListener

Listeners implementieren

- `java.awt.event`
- Um Listener zu nutzen, entsprechendes Interface und **alle** seine Methoden implementieren
- Durch `add???Listener()` der Komponente hinzufügen



Beispiel

```
private class MyButtonListener implements ActionListener {
        String s="";

        public MyButtonListener() { }

        public void actionPerformed(ActionEvent e) {
                String entry = e.getActionCommand();

                if (entry.equals("but 1"))
                        s = "button 1";
                else if (entry.equals("but 2"))
                        s = "button 2";

                System.out.println(s);
        }
}
```

Beispiel 2 (1/2)

- WindowListener als Adapter implementieren
  - Nur die benötigten Methoden implementieren

```
import java.awt.event.*;

public void initFrame() {
        ...
        this.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
        this.addWindowListener(new WindowsAdapter()  {
                public void windowClosed(WindowEvent e) {
                        System.exit(0);
                }
        });
}
```

Beispiel 2 (2/2)

```
import javax.swing.*;
public class MyGui extends JFrame {
   public MyGui() {
      super(''Name meiner GUI'');
   }
   public void initFrame() {
          Container contentPane = getContentPane();

          JButton b1 = new JButton(''Button 1'');

          MyButtonListener myButLis = new MyButtonListener();
          b1.addActionListener(myButLis);

          contentPane.add(b1);
   }
   ...
}
```

Übung 4.1

- Fügen Sie einen `KeyListener` hinzu. Die dazugehörige Methode soll prüfen, ob **`Escape (e.getKeyCode() == KeyEvent.VK_ESCAPE)`** gedrückt wurde und wenn ja, die Variable `gameRunning` auf `false` setzen.

KeyListener:
- keyPressed(KeyEvent e): Key wird gedrückt
- keyReleased(KeyEvent e): Key wird losgelassen
- keyTyped(KeyEvent e): Key wurde gedrückt und wieder losgelassen

In MukaGame() einfügen: `requestFocus();`

Übung 4.2

- Erweitern Sie den `KeyListener` so, dass die Spielfigur Muka über Cursortasten steuerbar ist.
- Kommentieren Sie dazu die `move()` Methode in der MukaEntity Klasse aus, aber bitte nicht löschen.

Übung 4.3

- Fügen Sie den `MouseListener` der `MukaGame` Klasse hinzu und schreiben Sie eine Methode, die nach einem Mausklick einen Richtungswechsel bei dem Gegenspieler verursacht.

MouseListener:
- mouseClicked(MouseEvent e): Maus geklickt
- mouseEntered(MouseEvent e): Maus kommt innerhalb des Applet Frames
- mouseExited(MouseEvent e): Maus verlässt Applet Frame
- mousePressed(MouseEvent e): Mausbutton ist gedrückt
- mouseReleased(MouseEvent e): Mausbutton wird wieder losgelassen