

Ugur Özbesnili (Gruppe 06)

Gui

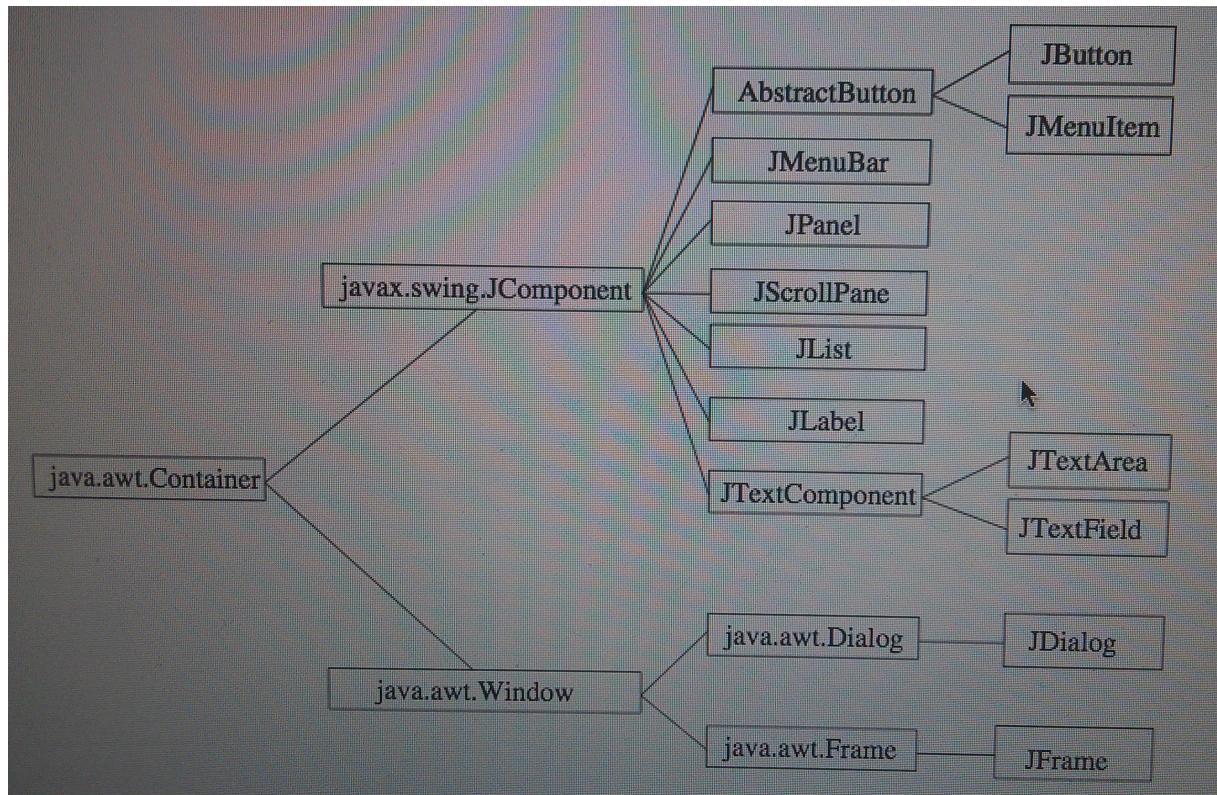
Swing

- Swing setzt auf dem AWT auf

Unterschied zwischen Swing und AWT

AWT	Swing
- schwergewichtige Komponente	- leichtgewichtige Komponente
- Schnittmenge aller, in den versch. Betriebssystemen vorhandenen, grafischen Komponenten	- Realisiert MVC-Architektur
- Layout zur Laufzeit	

Swing-Komponenten



Eine Gui mit Swing erstellen

1. Layout festlegen

```
package swingdemo;

import javax.swing.*;
import java.awt.BorderLayout;
import java.awt.event.*;

public class MyFrame extends JFrame implements ActionListener {
    private JTextArea textArea = new JTextArea();

    public MyFrame() {
        this.setTitle("SwingDemo Frame");
        //-----
        //set the layout to BorderLayout
        //-----
        this.getContentPane().setLayout(new BorderLayout());
        ...
    }
}
```

2. Grafische Komponente festlegen

```
//-----
//creating graphical components
//-----
textArea.setText("");
JScrollPane scrollPane = new JScrollPane(textArea);

JButton clearTextButton = new JButton("Clear TextArea");
JButton appendTextButton = new JButton("Append Text");
JButton setTextButton = new JButton("Set Text");

 JPanel panel = new JPanel();
panel.add(setTextButton);
panel.add	appendTextButton);
panel.add(clearTextButton);

this.getContentPane().add(scrollPane, BorderLayout.CENTER);
this.getContentPane().add(panel, BorderLayout.SOUTH);
```

3. Eventhandling festlegen

```
//-----
//set the event handling
//-----
setTextButton.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e){
        textArea.setText("Ein wenig Text");
    }
});

appendTextButton.addActionListener(
    new AppendTextActionListener());
clearTextButton.addActionListener(this);

this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
/*this.addWindowListener(new WindowAdapter(){
    public void windowClosing(WindowEvent e){
        System.exit(0);
}});*/
private class AppendTextActionListener
    implements ActionListener{
    public void actionPerformed(ActionEvent e){
        textArea.append("Noch mehr Text");
    }
}

public void actionPerformed(ActionEvent e){
    textArea.setText("");
}

public static void main(String[] args){
    MyFrame frame = new MyFrame();
    frame.setSize(400,500);
    frame.setVisible(true);
    //frame.pack();frame.show();
}
```

OpenGL

- Open Graphics Library
- Eine standardisierte Graphikbibliothek
- Mehrere hundert Funktionen, um 2D und 3D Rasterisierung mit einem Z-Bufferalgorithmus durchzuführen
- Eine kleine Sammlung geometrischer Primitive – Punkte, Zeilen, Polygone, Bilder und Bitübersichten

3D

- Schatten- und Spiegeleffekte muss gehen nicht mit OpenGL
- Deshalb nicht geeignet für photorealistische Bilder
- Aber wegen der hohen Geschwindigkeit ideal für Virtual-Reality-Anwendungen
- Oder für 3D-Modellierungsprogramme
- Durch OpenGl kann der Nutzer schattierte Körper sehen anstatt Drahtgittermodelle
- Im Profi-Bereich unumstrittener Standard

120 Funktionen

- 3D-Verschiebungen, Rotationen und Skalierungen
- acht Lichtquellen
- Definition von Umgebungshelligkeit, Glanz- und Diffus-Lichtanteil für jede Quelle
- Durchsichtigkeit und Reflektivität der einzelnen Lichtkomponenten als Materialeigenschaft
- Texturen aus beliebigen, farbigen Bildern
- Umsetzung von TrueType-Fonts in 3D-Schriften
- Doppelpufferung zur Erzeugung flüssiger Animationen

Quellen:

Philipp Meier – Institut für Informatik Ludwig-Maximilians-Universität München

[http://www.pst.informatik.uni-](http://www.pst.informatik.uni-muenchen.de/lehre/WS0304/progprakt/material/woche3/swingintro.pdf)

[muenchen.de/lehre/WS0304/progprakt/material/woche3/swingintro.pdf](http://www.it.fht-esslingen.de/~schmidt/vorlesungen/vr/seminar/ws9899/opengl.html)

<http://www.it.fht-esslingen.de/~schmidt/vorlesungen/vr/seminar/ws9899/opengl.html>