Hochschule Harz	FB Automatisierung und Informatik
Programmierung2	Dipl.-Inf., Dipl.-Ing. (FH) M. Wilhelm
Tutorial 03:	„Programmierung 2“ für MI / WI Thema: **Swing und GridBagLayout**

Versuchsziele

Kenntnisse in der Anwendung von:

- Erstellen einer Swing-Anwendung mit
  - JSpinner
  - GridBagLayout

Tutorial03: Swing: Erstellen eines JFrames mit GridBagLayout

In dieser Aufgabe soll das unten abgebildete Fenster erstellt werden.

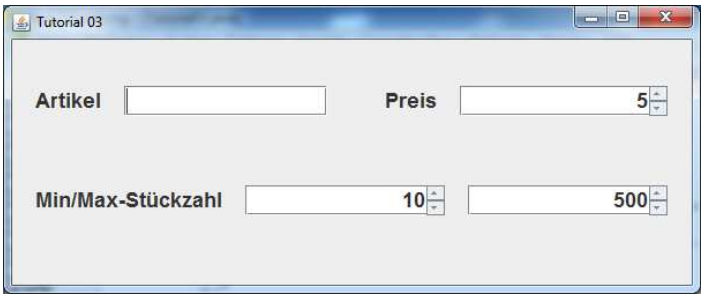


Abbildung 1 Musterlösung

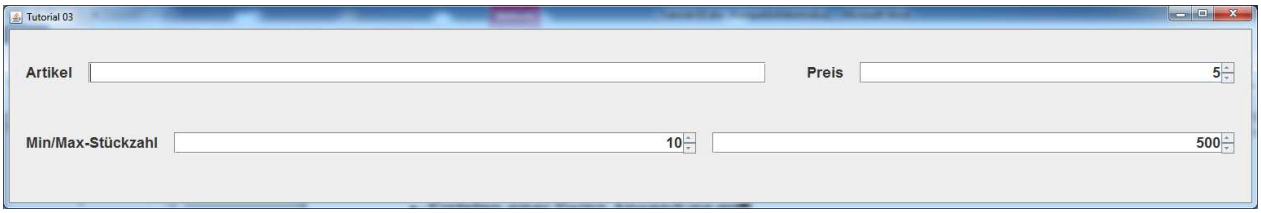


Abbildung 2 Musterlösung mit größerer Breite

Hinweis:

- Der Artikel hat einen „Prozentwert“ von 70%.
- Der Preis hat einen „Prozentwert“ von 30%.
- Die unteren JSpinner haben jeweils einen „Prozentwert“ von 50%.
- Bitte beachten Sie die unterschiedliche Spaltenzuordnung des ersten JSpinners in der zweiten Zeile.

1. Teilaufaufgaben: Projekt erstellen

- Erstellen Sie ein neues Eclipse–Projekt:
  - Projektname: Tutorial03
  - Klassenname: JFrame03
- Erstellen Sie eine neue Klasse
  - Menü File, Eintrag New, Eintrag class
  - Name: Tutorial03
- Erstellen Sie aus dem vorgegebene JFrame-Rahmen folgendes Programm:
  - Drei JLabel-Elemente
  - Ein JTextField als Editorzeile
  - Ein JSpinner mit double-Werten
  - Zwei JSpinner mit int-Werten

2. Teilaufaufgaben: GridBagLayout erstellen

- Fügen Sie die einzelnen GUI-Elemente in das JFrame
    - `JLabel lArtikel`
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill
    - `JTextField artikel`
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill
    - `JLabel lPreis`
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill
    - JSpinner preis
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill
    - `JLabel lPreis`
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill
    - `JSpinner anzmin`
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill
    - `JSpinner anzmax`
        - gridx
        - gridy
        - gridwidth
        - weightx
        - anchor
        - fill

- JSpinner preis
    - Datentyp double
        - Minimum: 0.0
        - Maximum: 1000.0
        - Defaultwert: 5.0
        - Step: 0.5
- JSpinner anzmin
    - Datentyp int
        - Minimum: 0
        - Maximum: 1000
        - Defaultwert: 10
        - Step: 1
- JSpinner anzmax
    - Datentyp int
        - Minimum: 0
        - Maximum: 1000
        - Defaultwert: 500
        - Step: 1

Hinweise:

- Sie können beliebig viele zusätzliche JPanel einfügen.
- Mit der unteren Anweisung wird das Panel farbig dargestellt.
    - `setBackground(Color.RED);`

Rahmen:

```
//Titel:            3. Tutorial, Programmierung 2
//Version:          1,0
//Copyright:        Copyright (c) 2013
//Autor:            M. Wilhelm
//Organisation:     HS-Harz
//Beschreibung:     GridBagLayout mit JSpinner


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

import java.util.*;
import java.io.*;


public class Tutorial03 extends JFrame {

 JLabel lArtikel = new JLabel("Artikel");
 JTextField artikel = new JTextField();
 JLabel lPreis = new JLabel("Preis");
 JSpinner preis;  // erzeugt in setGUI
 JLabel lMinMax = new JLabel("Min/Max-Stückzahl");
 JSpinner anzmin; // erzeugt in setGUI
 JSpinner anzmax; // erzeugt in setGUI


  public Tutorial03() {
    setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
    setGUI();
  }

  private void setGUI() {
    setSize(600, 250);
    setTitle("Tutorial 03");

    this.getContentPane().setLayout( new GridBagLayout() );

/*    this.getContentPane().add(label1 ,
        new GridBagConstraints(
```

```
             0, 0, 2, 2, 0.0, 0.0
            ,GridBagConstraints.WEST,
            GridBagConstraints.NONE,
            new Insets(0,0,0,0), 0, 0));
*/
    // hier fehlt Code


 // Nummern-Spinner für Werte von 1    default, min, max, step
/*
    SpinnerNumberModel model1 = new SpinnerNumberModel( 5.0, 0.0, 10.0, 0.4 );
    preis = new JSpinner( model1 );
*/

   //panelTwo.setBackground(Color.RED);

   // setFont( this.getContentPane() );
  } // setGUI



  public static void main(String[] args) {
    Tutorial03 frame = new Tutorial03();
    frame.setVisible(true);
 }

}  // Tutorial03
```

Anhang:

Klasse GridBagContraints:

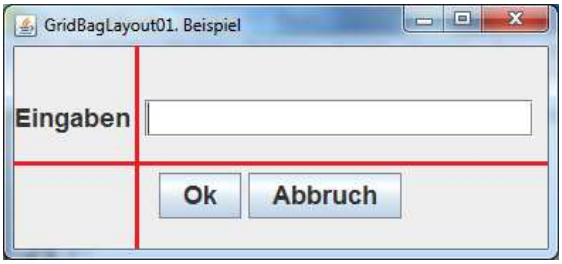
- int gridx — Element in der gridx-te Spalte
- int gridy — Element in der gridy-te Zeile
- int gridwidth — Anzahl der Spalten, die das Element benötigt
- int gridheight — Anzahl der Zeilen, die das Element benötigt
- double weightx — Verteilung von Platzänderung (Prozentual)
- double weighty — Verteilung zusätzlichen Platzes (Prozentual)
- int anchor — Verknüpfung (NORTHEAST, EAST, SOUTHEAST, CENTER, SOUTH, SOUTHWEST, WEST, NORTH, NORTHWEST)
- int fill — Ausfüllen (NONE, HORIZONTAL, BOTH, VERTICAL)
- Insets insets — äußerer Rand, **Top, Left, Bottom, Right**
- int ipadx — vergrößert das GUI-Element, Breite des GUI-Elements
- int ipady — vergrößert das GUI-Element, Höhe des GUI-Elements

Beispiel:

```
 this.getContentPane().add(jLabel1,
     new GridBagConstraints(0, 0, 1, 1, 0.0, 0.0,GridBagConstraints.WEST,
             GridBagConstraints.NONE, new Insets(4,20,4,0), 00, 0));

 this.getContentPane().add(tName,
     new GridBagConstraints(1, 0, 1, 1, 1.0, 0.0,GridBagConstraints.WEST,
             GridBagConstraints.HORIZONTAL, new Insets(4,0,4,20), 00, 0));

JPanel panelBn = new JPanel();
panelBn.setLayout( new FlowLayout() );
panelBn.add(bnOk);
panelBn.add(bnEsc);
this.getContentPane().add(panelBn,
     new GridBagConstraints(0, 1, 2, 1, 0.0, 0.0,GridBagConstraints.CENTER,
            GridBagConstraints. HORIZONTAL, new Insets(4,0,4,0), 0, 0));
```



Beispielcode für JSpinner

Folgender Programmausschnitt zeigt mehrere Arten von JSpinner an.

```
JSpinner spInteger;
                                        // Aktueller Wert, Min, Max, Increment
spInteger = new JSpinner( new SpinnerNumberModel(0,  0, 100, 1)   );
```

```
JSpinner spDouble;
                                        // Aktueller Wert, Min, Max, Increment
spDouble = new JSpinner( new SpinnerNumberModel(5.5,  0, 100, 0.5)   );
```

setzen eines Wertes in einem JSpinner:

```
SpinnerInt.setValue(  new Integer( 123 )  );
SpinnerDouble.setValue(  new Double ( 55.6 )  );
```

holen eines Wertes aus einen Integer-JSpinner:

```
Integer I = (Integer) Spinner1.getValue(   );
int k = I.intValue();
```

oder

```
int k = ( (Integer) Spinner1.getValue() ).intValue();
```

holen eines Wertes aus einen Double-JSpinner:

```
Double D = (Double) Spinner2.getValue(   );
double f = D.doubleValue();
```

oder

```
double f = ( (Double) Spinner2.getValue() ).doubleValue();
```

Menü-Erstellung

Variablendeklaration:

- private JMenuBar menuBar1; // wird ins JFrame eingefügt
- private JMenu mainFile; // kennzeichnet ein Hauptmenü
- private JMenuItem mnClose; // kennzeichnet ein Untermenü

Menü erzeugen:

- menuBar1 = new JMenuBar();
- mainFile = new JMenu("Datei");
- mnClose = new JMenuItem("Schließen");
- Action-Evenmt ist identisch mit einem Schalter
- mainFile.add(mnClose); // der Menüeintrag wird in das Hauptmenü eingefügt
- menuBar1.add(mainFile); // das Hauptmenü wird in den MenüBar eingefügt
- this.setJMenuBar(menuBar1); // einfügen des MenüBar in das JFrame

Schriftart setzen

Aufruf in setGUI mit: setFont(this.getContentPane());

```
private void setFont(Container parent) {
   System.out.println(""+parent.getComponentCount()  );
   for (int i=0; i<parent.getComponentCount(); i++ ) {
      Component c = parent.getComponent(i);
      if (c instanceof JPanel ) {
         JPanel  pn = (JPanel ) c;
         setFont(pn);
      }
     else {
         c.setFont( new Font("Arial", Font.BOLD,18)  );
     }
   }
 }

private void setFont(JComponent parent) {
   // System.out.println(""+parent.getCount()  );
   for (int i=0; i<parent.getComponentCount(); i++ ) {
      Component c = parent.getComponent(i);
      if (c instanceof JPanel ) {
        // setFont(c);
      }
     else {
         c.setFont( new Font("Arial", Font.BOLD,18)  );
     }
   }
 }
```