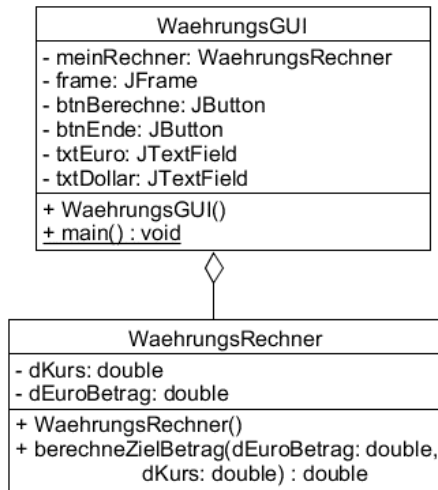


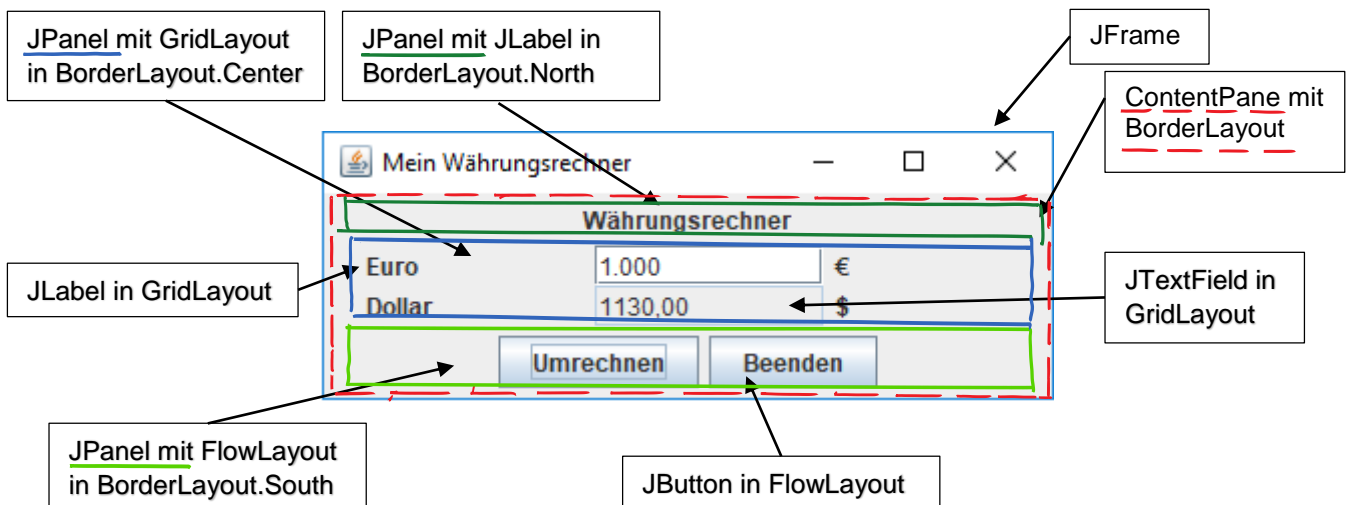
Einfache GUI Anwendung

Ziel: Erstellung einer einfachen GUI Anwendung: Kennenlernen des prinzipiellen Aufbaus eines JFrame mit JPanel, LayoutManagern und verschiedenen Dialogelementen.

1.1. Die zu erstellende Anwendung hat folgendes Klassendiagramm:



1.2. Die GUI der Anwendung hat folgende Struktur:



- Erstellen Sie zunächst die Klasse `WaehrungsGUI` und zum Testen der GUI die Klasse `WaehrungsStart`.
- Erstellen Sie eine `MouseListener` Klasse (abgeleitet von der `MouseAdapter` Klasse) für den „Beenden“ Button als innere Klasse von `WaehrungGUI`. In dieser Klasse muss nur die `mouseClicked()` Methode implementiert werden. Wenn der Button geklickt wird, soll das Programm mit dem Befehl `System.exit(0)` beendet werden.
- Erstellen Sie eine `MouseListener` Klasse (abgeleitet von der `MouseAdapter` Klasse) für den „Umrechnen“ Button als innere Klasse von `WaehrungGUI`. In dieser Klasse muss nur die `mouseClicked()` Methode implementiert werden. Wenn der Button geklickt wird, geben Sie zum Testen zunächst nur einen Text im unteren `JTextField` aus.

1.6. Implementieren Sie im selben Package jetzt noch die Fachkonzept-Klasse

WaehrungsRechner.

1.7. Jetzt soll die `mouseClicked()` Methode von 1.5 abgeändert werden.

- aus dem Euro-Textfeld wird der String mit `getText()` ausgelesen.
- der String muss in einen double Wert konvertiert werden.
- dann kann mit einem Objekt der Fachkonzeptklasse der Zielbetrag berechnet werden.
- das Ergebnis wird mit `String.format()` für die Ausgabe im Dollar-Textfeld formatiert.

1.8. Wenn man das Programm testet, stellt man fest, dass es bei einer Fehleingabe im Euro-Textfeld möglicherweise mit einer Exception abstürzt. Aus diesem Grund ist es sinnvoller für dieses Textfeld statt der Klasse `JTextField` die Klasse`JFormattedTextField` zu verwenden:

- ändern Sie die Verweisvariable von `JTextField` in `JFormattedTextField` ab
- erzeugen Sie das `JFormattedTextField` mit:
`new JFormattedTextField(NumberFormat.getNumberInstance())`
→ In diesem Textfeld können jetzt nur noch Zahlen eingegeben werden
- Der Wert aus dem Textfeld kann mit folgendem Befehl geholt und konvertiert werden:
`((Number)txtEuro.getValue()).doubleValue()`
- Testen Sie das geänderte Programm.

1.9. Eine weitere Verbesserung des Programms bestünde darin, dass die Umrechnung nicht nur dann ausgeführt wird, wenn man den Button klickt, sondern auch dann, wenn man im Euro-Textfeld die <Enter> Taste drückt. Hierfür gibt es selbstverständlich verschiedene Lösungen, eine sieht folgendermaßen aus:

Statt den beiden `MouseListener` Klassen wird eine `ActionListener` Klasse für die Ereignisbehandlung verwendet:

- Kommentieren Sie alles aus, was mit der `MouseListener` Ereignisbehandlung zu tun hatte.
- Definieren Sie als innere Klasse eine `ActionListener` Klasse die das `ActionListener` Interface implementiert:

```
private class BtnActionListener implements ActionListener
{
    public void actionPerformed(ActionEvent e)
    {
        if (e.getSource().equals(btnEnde))
        {
            System.exit(0);
        }
        else
        {
            // hier die Berechnung durchführen
        }
    }
}
```

- Registrieren Sie den `ActionListener` jetzt sowohl auf die beiden Buttons als auch auf das Euro-Textfeld und testen Sie, ob das geänderte Programm, die Anforderung erfüllt.

```
public class WaehrungsRechner
{
    private double dKurs;
    private double dEuroBetrag;

    public WaehrungsRechner()
    {
        this.dEuroBetrag = 1;
        this.dKurs = 1.1;
    }

    public double berechneZielBetrag(double dEuroBetrag, double dKurs)
    {
        this.dEuroBetrag = dEuroBetrag;
        this.dKurs = dKurs;
        return dEuroBetrag * dKurs;
    }

    public double getdKurs()
    {
        return this.dKurs;
    }

    public double getdEuroBetrag()
    {
        return this.dEuroBetrag;
    }
}

* WaehrungsGUI.java
package guiApps;
import java.awt.*;
* @author stk
public class WaehrungsGUI
{
    private WaehrungsRechner meinRechner; // Verweis auf Fachkonzept Objekt
    // GUI Komponenten nur dann zu Attributen machen, wenn nötig
    private JFrame frame; // Verweis auf Fenster Objekt
    private JButton btnBerechne;
    private JButton btnEnde;
    private JFormattedTextField txtEuro;
    private JTextField txtDollar;

    public static void main(String[] args)
    {
        // ein Objekt der neuen GUI-Klasse erzeugen
        WaehrungsGUI fenster = new WaehrungsGUI();
        // Darstellung des Fensters in der optimalen Grösse
        fenster.frame.pack();
        // Darstellung des Fensters auf dem Bildschirm
        fenster.frame.setVisible(true);
    }
}
```

```

public WaehrungsGUI()
{
    meinRechner = new WaehrungsRechner();
    this.frame = new JFrame("Mein Währungsrechner");
    // Wenn das Fenster geschlossen wird, soll auch das Programm enden
    this.frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    // In contentPane werden die Steuerelemente angelegt
    Container contentPane = frame.getContentPane();
    contentPane.setLayout(new BorderLayout());
    // Panel für den Titel
    JPanel pnlTitel = new JPanel(); // Standard ist FlowLayout
    JLabel lblTitel = new JLabel("Währungsrechner");
    pnlTitel.add(lblTitel);
    // Panel für die Textfelder und Labels
    JPanel pnlEinAusgabe = new JPanel();
    pnlEinAusgabe.setLayout(new GridLayout(0, 3));
    JLabel lblEuro = new JLabel("Euro");
    JLabel lblDollar = new JLabel("Dollar");
    JLabel lblEuroEinheit = new JLabel(" €");
    JLabel lblDollarEinheit = new JLabel(" $");
    this.txtEuro = new JFormattedTextField(NumberFormat.getNumberInstance());
    this.txtEuro.setValue(Double.valueOf(1));
    this.txtDollar = new JTextField(10);
    this.txtDollar.setEditable(false);
    pnlEinAusgabe.add(lblEuro);
    pnlEinAusgabe.add(this.txtEuro);
    pnlEinAusgabe.add(lblEuroEinheit);
    pnlEinAusgabe.add(lblDollar);
    pnlEinAusgabe.add(this.txtDollar);
    pnlEinAusgabe.add(lblDollarEinheit);
    // Panel für die Buttons
    JPanel pnlButtons = new JPanel(new FlowLayout());
    this.btnBerechne = new JButton("Umrechnen");
    this.btnEnde = new JButton("Beenden");
    pnlButtons.add(this.btnBerechne);
    pnlButtons.add(this.btnEnde);
    // Platzieren der Panels auf der Inhaltsebene (contentPane)
    contentPane.add(pnlTitel, BorderLayout.NORTH);
    contentPane.add(new JLabel(" "), BorderLayout.WEST);
    contentPane.add(pnlEinAusgabe, BorderLayout.CENTER);
    contentPane.add(pnlButtons, BorderLayout.SOUTH);

    // Auskommentiert Lösung für A1.4 / A1.5
    // Listener Objekt erzeugen und auf die Buttons registrieren
    // BtnBerechneListener btnBerechneListener = new BtnBerechneListener();
    // this.btnBerechne.addMouseListener(btnBerechneListener);
    // //btnBerechneListener.berechne();
    // this.btnEnde.addMouseListener(new BtnEndeListener());
    // Geänderte Lösung A1.9
    BtnActionListener btnActionListener = new BtnActionListener();
    this.btnEnde.addActionListener(btnActionListener);
    this.btnBerechne.addActionListener(btnActionListener);
    this.txtEuro.addActionListener(btnActionListener);
    berechne();
}

```



```
private void berechne()
{
    double dEuro = ((Number)txtEuro.getValue()).doubleValue();
    //double dEuro = Double.parseDouble(txtEuro.getText());

    double dDollar;

    dDollar = meinRechner.berechneZielBetrag(dEuro, 0.88);

    txtDollar.setText(String.format("%.2f", dDollar));
}

private class BtnBerechneListener extends MouseAdapter
{
    public void mouseClicked(MouseEvent e)
    {
        berechne();
    }
}

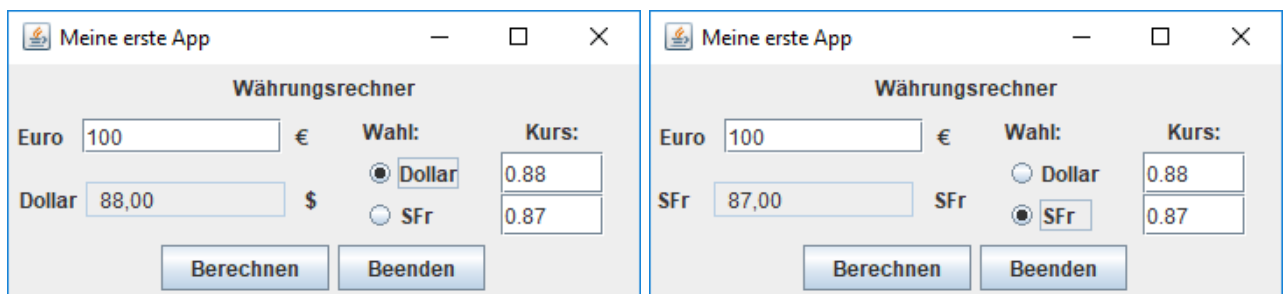
private class BtnEndeListener extends MouseAdapter
{
    public void mouseClicked(MouseEvent e)
    {
        System.exit(0);
    }
}

private class BtnActionListener implements ActionListener
{
    @Override
    public void actionPerformed(ActionEvent e)
    {
        if (e.getSource().equals(btnEnde))
        {
            System.exit(0);
        }
        else
        {
            berechne();
        }
    }
}
}
```

Arbeiten mit dem Eclipse Window Builder

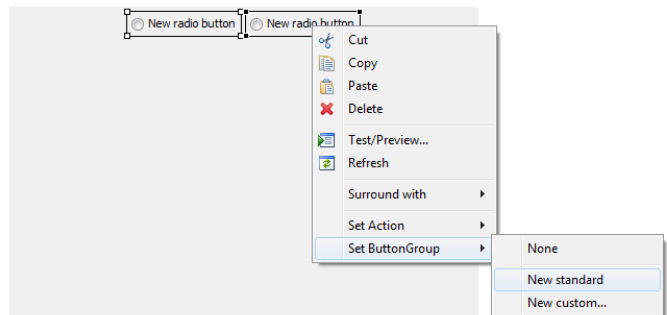
Es gibt verschiedene Tools um grafische Oberflächen mit Java produktiver und einfacher zu erstellen. Eclipse bietet hierfür den Window Builder (und weitere Alternativen).

- 2.1. Um sich mit dem Window Builder ein wenig vertraut zu machen, soll zunächst der Währungsrechner aus Aufgabe 1 jetzt mit dem Window Builder erstellt werden. Die Fachkonzeptklasse kann wiederverwendet werden.
- 2.2. Legen Sie eine Kopie der Lösung von 2.1 an und ergänzen Sie diese entsprechend dem nachfolgenden Bild.



Wie zu erkennen ist, soll sich bei der Änderung der Devisenwahl auch die Beschriftungen bei der Ergebnisausgabe entsprechend anpassen.

- Für den Bereich „Devisenauswahl“ wurde eine extra JPanel angelegt, dass sich im BorderLayout.EAST befindet und selbst ein GridLayout enthält.
- Die RadioButton müssen zu einer Gruppe zusammengefügt werden, so dass immer nur eine Option wählbar ist.
- In der Endversion soll bei jeder Optionsänderung die Berechnung automatisch ausgeführt werden, ohne dass der Benutzer den „Berechnen“ Knopf anklicken muss.



Zur Lösung der Aufgabe wird die Lösung von 2.1 erweitert. Wie in der Aufgabe beschrieben werden die RadioButtons hinzugefügt und zu einer Gruppe zusammengefügt. Einige Komponenten müssen als Attribute zur Verfügung stehen. Falls der Window Builder diese Komponenten zunächst nur im Konstruktor bzw. in der initialize() Methode definiert, dann muss die Verweisvariable möglicherweise zu den Attributen verschoben werden. (Achtung: die Variablennamen stimmen vermutlich nicht mit Ihrer GUI überein!)

```
private JFrame frmMeineErsteApp;
private JFormattedTextField txtEuro;
private JTextField txtDollar;
private JLabel lblZielName;
private JLabel lblZielEinheit;
private WaehrungsRechner meinRechner = new WaehrungsRechner(0.88, 100);
private final ButtonGroup buttonGroup = new ButtonGroup();
private JRadioButton rdbtnDollar;
private JRadioButton rdbtnSFr;
private JTextField txtDollarKurs;
private JTextField txtSFrKurs;
```

Die Methode berechne() muss so angepasst werden, dass geprüft wird, welcher RadioButton ausgewählt ist:

```
private void berechne()
{
    double dEuro = ((Number)txtEuro.getValue()).doubleValue();
    double dKurs;
    //double dEuro = Double.parseDouble(txtEuro.getText());

    double dZielBetrag;

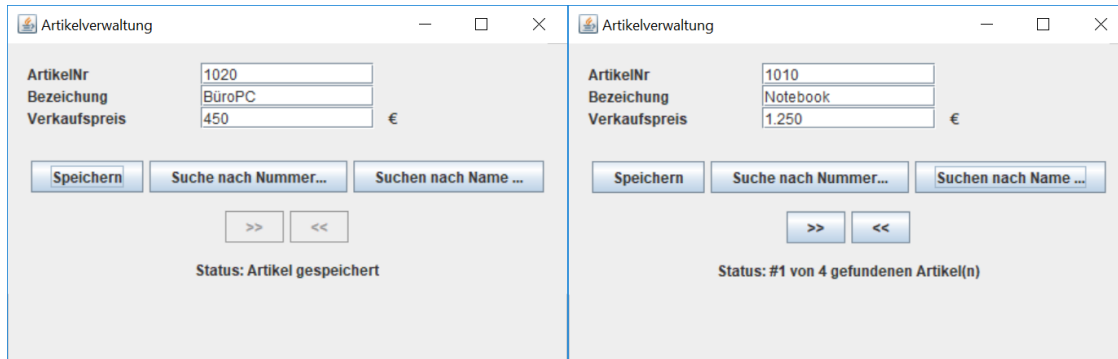
    if (rdbtnDollar.isSelected())
    {
        lblZielName.setText("Dollar");
        lblZielEinheit.setText(" $ ");
        dKurs = Double.parseDouble(txtDollarKurs.getText());
    }
    else
    {
        lblZielName.setText("SFr ");
        lblZielEinheit.setText(" SFr");
        dKurs = Double.parseDouble(txtSFrKurs.getText());
    }

    dZielBetrag = meinRechner.berechneZielBetrag(dEuro, dKurs);

    txtDollar.setText(String.format("%7.2f", dZielBetrag));
}
```

- 2.3. Im Aufgabenblatt „A_Java_Arrays“ in Aufgabe 3.1 haben wir ein Programm zur Erfassung und Verwaltung von Artikel Daten erstellt (siehe UML Klassendiagramm dort). In Aufgabe 1 von Aufgabenblatte „A_JavaCollections“ haben wir die Container-Klasse dieses Programms noch verbessert. Für dieses Programm soll jetzt statt der bisherigen Konsoloberfläche eine GUI bereitgestellt werden (s.u.).

Die GUI soll im wesentlichen die selbe Funktionalität ermöglichen, wie die Konsolanwendung. Es können Artikel erfasst werden, Artikel anhand ihrer eindeutigen ArtikelNr und anhand einer nicht eindeutigen Bezeichnung gesucht werden.



- Mit der Speichern-Schaltfläche wird ein neuer Artikel erfasst und in der Container-Klasse abgespeichert. Die Vor- und Zurück-Schaltflächen sind dabei deaktiviert.
- Mit Klick auf „Suchen nach Nummer...“ öffnet sich ein kleines Dialogfenster zur Eingabe der gewünschten ArtikelNr und sofern ein passender Artikel gefunden wird, werden dessen Daten angezeigt.
Um nebenstehenden Dialog anzuzeigen und dessen Rückgabewert entgegenzunehmen wird folgender Befehl verwendet:



```
sNummer = (String) JOptionPane.showInputDialog("Bitte ArtikelNr  
eingeben:");
```

Die Vor- und Zurück-Schaltflächen bleiben auch in diesem Fall deaktiviert.

- Mit Klick auf „Suchen nach Name...“ öffnet sich ebenfalls ein entsprechendes Dialogfenster zur Eingabe der gesuchten Artikelbezeichnung. In diesem Fall kann es mehrere Artikel mit derselben Bezeichnung als Ergebnis geben. Falls mehr als ein Artikel gefunden wurde, werden die Vor- und Zurück-Schaltflächen aktiviert und mit jedem Klick darauf wird der nächste bzw. vorherige Artikel angezeigt. Das kann so gemacht werden, dass man nach dem letzten Artikel wieder zum ersten gelangt und umgekehrt.

Vorgehensweise:

- Erstellen Sie ein neues Package und kopieren Sie die Klassen von Moodle in das Package („KlassenArtikelContainer“).
- Erstellen Sie eine GUI-Klasse mit Hilfe des Eclipse Window Builder.
- Die Inhaltsebene enthält ein GridLayout mit 1 Spalte.
- Zeile 1 enthält ein JPanel mit einem GridLayout mit 3 Spalten für den Ein-/Ausgabebereich.
- Zeile 2 enthält ein JPanel für den Button-Bereich bestehend aus zwei JPanel für die obere und untere Button-Zeile.
Zeile 3 enthält nur ein JPanel mit JLabel für die Statusmeldungen
- Für die Ereignisverarbeitung bietet sich ein ActionListener in einer inneren Klasse an. Hier wird mit getSource() jeweils festgestellt von welcher Komponente das Ereignis ausgelöst wurde und dann können in einer entsprechenden Methode die notwendigen Aktionen ausgeführt werden.

Lösung: GUI mit Window Builder entsprechend der Anleitung erstellen. Es ist wichtig, den Komponenten sprechende Namen zu vergeben. Einige Komponenten müssen als Attribute zur Verfügung stehen. Falls der Window Builder diese Komponenten zunächst nur im Konstruktor bzw. in der initialize() Methode definiert, dann muss die Verweisvariable möglicherweise zu den Attributen verschoben werden.

```
public class AuftragsVerwContGUI
{
    // Attribute der GUI Klasse
    private ArtikelContainer artikelliste = new ArtikelContainer();
    private Artikel[] aArtikelGefunden = null;
    private int iAktIndex;
    private JFrame frmArtikelverwaltung;
    private JTextField txtArtikelNr;
    private JTextField txtBezeichnung;
    private JFormattedTextField fxtPreis;
    private JButton btnSpeichern;
    private JButton btnSuchenNr;
    private JButton btnSuchenName;
    private JButton btnVorwaerts;
    private JButton btnRueckwaerts;
    private JLabel lblStatus;

    // Action Listener registrieren
    txtArtikelNr.addActionListener(new BtnActionListener());
    txtBezeichnung.addActionListener(new BtnActionListener());
    fxtPreis.addActionListener(new BtnActionListener());
    btnSpeichern.addActionListener(new BtnActionListener());
    btnSuchenNr.addActionListener(new BtnActionListener());
    btnSuchenName.addActionListener(new BtnActionListener());
    btnVorwaerts.addActionListener(new BtnActionListener());
    btnRueckwaerts.addActionListener(new BtnActionListener());
}

private class BtnActionListener implements ActionListener
{
    // Innere Klasse für den Action Listener
    @Override
    public void actionPerformed(ActionEvent e)
    {
        if (e.getSource() == btnSpeichern)
        {
            speichern();
        }
        else if (e.getSource() == btnSuchenNr)
        {
            suchenNachNr();
        }
        else if (e.getSource() == btnSuchenName)
        {
            suchenNachName();
        }
        else if (e.getSource() == btnVorwaerts)
        {
            zeigenVorwaerts();
        }
        else if (e.getSource() == btnRueckwaerts)
        {
            zeigenRueckwaerts();
        }
    }
}
```

```

private void speichern()
{
    Artikel einArtikel;
    int iArtikelNr = Integer.parseInt(txtArtikelNr.getText());
    double dEuro = ((Number)fxtPreis.getValue()).doubleValue();
    einArtikel = new Artikel(iArtikelNr, txtBezeichnung.getText(), dEuro);
    artikelliste.speichereArtikel(einArtikel);
    lblStatus.setText("Status: Artikel gespeichert");
    btnVorwaerts.setEnabled(false);
    btnRueckwaerts.setEnabled(false);
}

private void suchenNachNr()
{
    String sNummer;
    Artikel einArtikel;

    sNummer = (String)JOptionPane.showInputDialog("Bitte ArtikelNr eingeben:");

    einArtikel = artikelliste.sucheArtikelNachNr(Integer.parseInt(sNummer));
    if (einArtikel != null)
    {
        txtArtikelNr.setText(sNummer);
        txtBezeichnung.setText(einArtikel.getSBezeichnung());
        fxtPreis.setValue(einArtikel.getDVerkaufsPreis());
        // txtPreis.setText(String.format("%.2f", einArtikel.getDVerkaufsPreis()));
        lblStatus.setText("Status: Artikel " + sNummer + " gefunden");
        btnVorwaerts.setEnabled(false);
        btnRueckwaerts.setEnabled(false);
    }
}

private void suchenNachName()
{
    String sArtikelBez;
    Artikel einArtikel;

    sArtikelBez = (String)JOptionPane.showInputDialog("Bitte Artikelbezeichnung eingeben:");

    aArtikelGefunden = artikelliste.sucheArtikelNachBezeichnung(sArtikelBez);
    if (aArtikelGefunden != null)
    {
        iAktIndex = 0;
        txtArtikelNr.setText(Integer.toString(aArtikelGefunden[iAktIndex].getINr()));
        txtBezeichnung.setText(aArtikelGefunden[iAktIndex].getSBezeichnung());
        fxtPreis.setValue(aArtikelGefunden[iAktIndex].getDVerkaufsPreis());
        lblStatus.setText("Status: #" + (iAktIndex+1) + " von " + aArtikelGefunden.length
            + " gefundenen Artikel(n)");
        if (aArtikelGefunden.length > 1)
        {
            btnVorwaerts.setEnabled(true);
            btnRueckwaerts.setEnabled(true);
        }
    }
}

```

```

private void zeigenVorwaerts()
{
    if (iAktIndex < aArtikelGefunden.length-1)
    {
        iAktIndex++;
    }
    else
    {
        iAktIndex = 0;
    }

    txtArtikelNr.setText(Integer.toString(aArtikelGefunden[iAktIndex].getINr()));
    txtBezeichnung.setText(aArtikelGefunden[iAktIndex].getSBezeichnung());
    fxtPreis.setValue(aArtikelGefunden[iAktIndex].getDVerkaufsPreis());
    lblStatus.setText("Status: #" + (iAktIndex+1) + " von " + aArtikelGefunden.length
        + " gefundenen Artikel(n)");
}

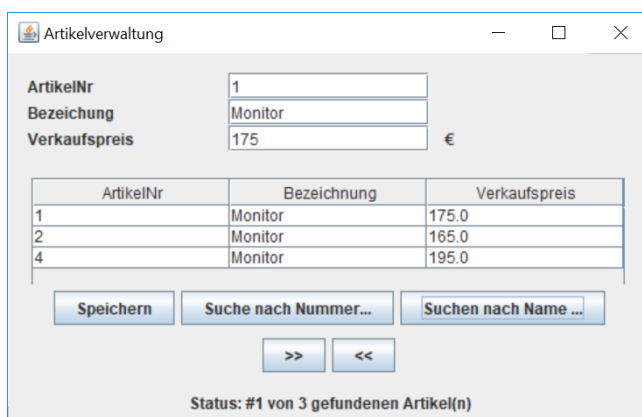
private void zeigenRueckwaerts()
{
    if (iAktIndex > 0)
    {
        iAktIndex--;
    }
    else
    {
        iAktIndex = aArtikelGefunden.length-1;
    }

    txtArtikelNr.setText(Integer.toString(aArtikelGefunden[iAktIndex].getINr()));
    txtBezeichnung.setText(aArtikelGefunden[iAktIndex].getSBezeichnung());
    fxtPreis.setValue(aArtikelGefunden[iAktIndex].getDVerkaufsPreis());
    lblStatus.setText("Status: #" + (iAktIndex+1) + " von " + aArtikelGefunden.length
        + " gefundenen Artikel(n)");
}
}

```

2.4. Zusatzaufgabe für ganz Schnelle:

Erweitern Sie das Programm um eine Ausgabetabelle für den Fall, dass bei der „Suchen nach Name...“ mehr als ein Artikel gefunden wird.



| ArtikelNr | Bezeichnung | Verkaufspreis |
|-----------|-------------|---------------|
| 1 | Monitor | 175.0 |
| 2 | Monitor | 165.0 |
| 4 | Monitor | 195.0 |

- Falls danach „Speichern“ oder „Suchen nach Nummer..“ ausgeführt wird, soll die Tabelle wieder entfernt werden.
- Eine Tabelle, die nur Werte anzeigen soll, diese aber vom Benutzer nicht verändert werden können, ist relativ einfach zu erstellen. In Moodle liegt die Klasse