Erstellung von Benutzerschnittstellen in der Programmiersprache Java

Java Abstract Windowing Toolkit (AWT)

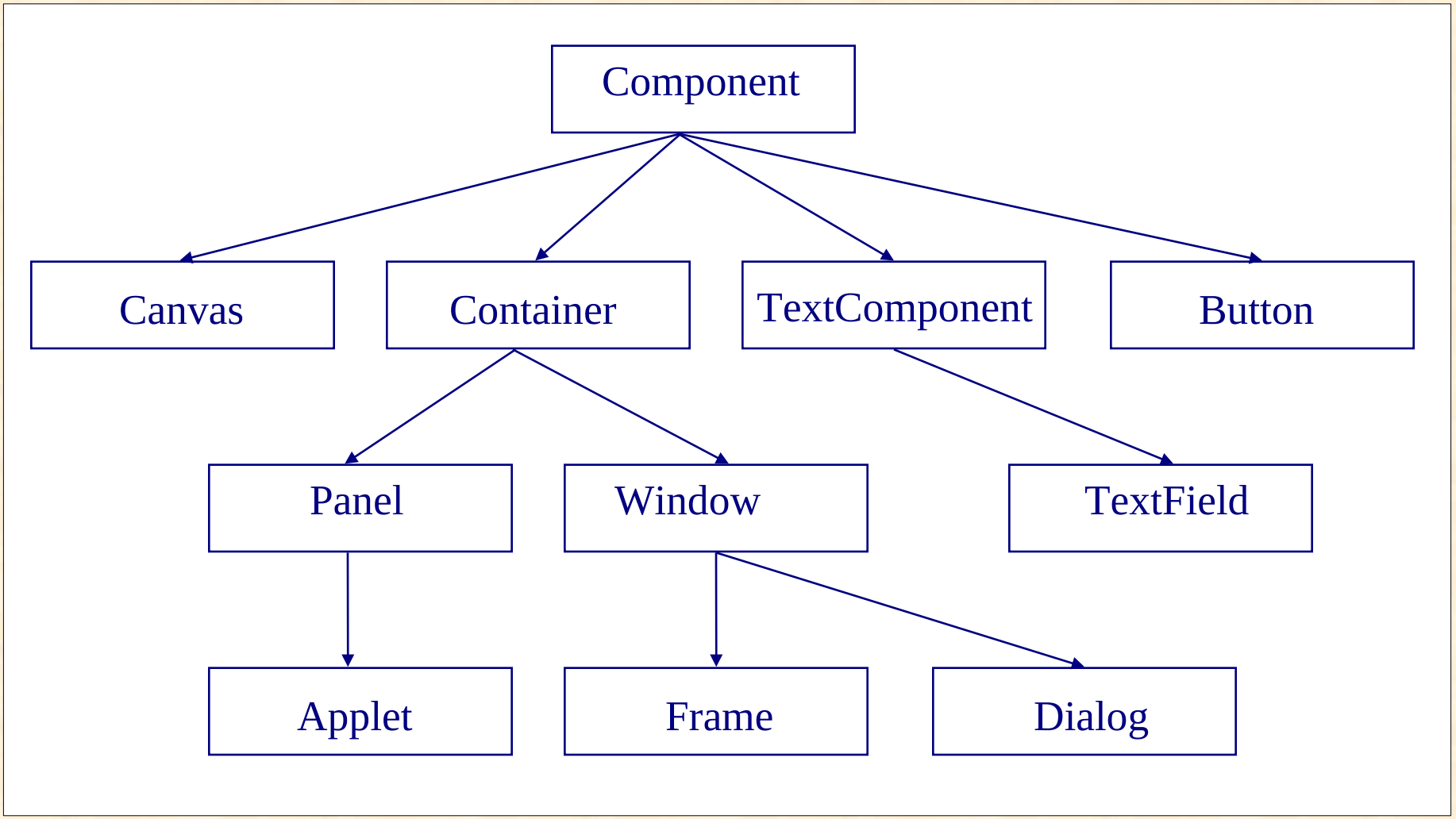
- Elemente der Benutzeroberfläche: Fenster, Menüs, Schaltfläche, Kontrollfelder, Textfelder, Bildlaufleisten und Listenfelder
- Container, die Elemente der Benutzeroberfläche und andere Container enthalten
- Mechanismen zum Anlegen von Komponenten, um eine plattformunabhängige Benutzeroberfläche zu entwickeln
- Handhabung von System- und Benutzerereignissen

- Alternativ: Benutzeroberfläche kann mit Klassen der Swing-Bibliothek im Paket javax.swing gestaltet werden:
  - Klassen muss J vorangestellt werden (JLabel, JButton, etc.)
  - höhere Funktionalität und Plattformunabhängigkeit
  - aber: Positionierung der Komponenten und Handhabung von Ereignissen wie bei der AWT

Container und Komponenten

- graphische Oberflächen bestehen aus Komponenten
  - einfache Komponenten
  - zusammengesetzte Komponenten (Container)
- einfache Komponenten des AWT
  - Button, Canvas, Label, List, TextArea, TextField, CheckBox, CheckBoxGroup, Choice, Scrollbar, MenuBar und PopupMenu
- zusammengesetzte Komponenten des AWT (Container)
  - Panel, ScrollPane, Window, Frame und Dialog
- Container können wiederum Container enthalten
  - hierarchischer Aufbau der grafischen Oberflächen möglich
- Container als auch einfache Komponenten sind von der Klasse java.awt.Component abgeleitet

AWT-Klassenhierarchie



Methoden der Klasse Component

- getParent() - liefert den Container, in dem sich die Komponente befindet
- setForeground(Color) - setzt Vordergrundfarbe derKomponente
- setBackground(Color) - setzt Hintergrundfarbe der Komponente
- setSize(int, int) - legt die Größe der Komponente fest
- getSize() - liefert die momentane Größe der Komponente
- paint(Graphics) - zeichnet die Komponente
- setEnabled(boolean) - selektierbar bzw. nicht selektierbar machen
- setFont() - Setzen des Schriftfonts
- setVisible(false) - Verbergen der Komponente
- setVisible(true)- Zeigen der Komponente

Grundlagen: Java-Applikationen

- Java-Applikationen benutzen als Container oft einen Frame
  - Konstruktion eines Frames

    ```
    Frame myFrame = new Frame(´´A simple Frame´´);
    myFrame.setSize(300,200);
    ```

  - Anzeigen eines Frames

    ```
    myFrame.setVisible(true);
    ```

  - Entfernen eines Frames

    ```
    myFrame.setVisible(false);
    ```

- Hinzufügen von Komponenten mit der Methode add()

  ```
  myFrame.add("Center", new Button(´´I like Java´´));
  ```

Beispiel

```
import java.awt.*;
public class FrameApplikationDemo{
        static Frame myFrame;
        static public void main(String argv[]){
                // Erzeugen eines Frames
                myFrame = new Frame(´´ A simple Frame ´´);
                myFrame.setSize( 300, 200 );
                // Anzeigen des Frames
                myFrame.setVisible(true);
                myFrame.add( "North", new Button(´´ I like Java ´´));
        }
}
```

Button

- Button sind Schaltflächen, die durch Anklicken die bezeichnete Aktion ausführen
- Konstruktoren
  - Button() erzeugt eine leere Schaltfläche ohne Beschriftung
  - Button(String) erzeugt eine Schaltfläche mit der Beschriftung

```
Button b  = new Button(´´Cancel´´);
```

- Methoden
  - String getLabel() - gibt den Wert für die Beschriftung des Buttons
  - setLabel(String) - setzt neue Beschriftung des Buttons
  - setEnabled(boolean) - Selektierbarkeit des Buttons

Listenfelder

- Listenfelder sind Listen mit mehreren Einträgen, aus denen eine oder mehrere ausgewählt werden können

```
List myList;
Frame myFrame
...
// Initialisierung des Listenfeldes
myList = new List( 3, false );
myList.add(´´Java´´); myList.add(´´Coffee´´);
myList.add(´´Expresso´´); myList.add(´´Cappuccino´´);
// Hinzufügen des Listenfeldes
myFrame.add("Center", myList);
```

Methoden der Klasse List

- getItem(int) - gibt die Zeichenkette des Eintrags an der angegebenen Position aus
- getItemCount() - gibt die Anzahl der Einträge in der Liste aus
- getSelectedIndex() - gibt die Indexposition des ausgewählten Eintrags
- getSelectedIndexes() - gibt eine Reihe von Indexpositionen aus
- getSelectedItem() - gibt den momentan ausgewählten Eintrag als Zeichenkette aus
- getSelectedItems() - gibt eine Reihe von Zeichenketten aus, die ausgewählte Einträge enthalten
- select(int) - wählt den Eintrag an der bezeichneten Position aus
- select(String) - wählt den Eintrag mit dieser Zeichenkette aus

Textfelder

- ein Textfeld ist ein umrandeter Bereich, in dem Daten eingeben werden können
- Konstruktoren
  - TextField() erzeugt ein leeres Textfeld, das 0 Zeichen breit ist
  - TextField(int) erzeugt ein leeres Textfeld mit einer bestimmten Anzahl an Zeichen
  - TextField(String), das die in der Zeichenkette enthaltene Anzahl an Zeichen breit ist und mit dieser initialisiert wird
  - TextField(String, int) erzeugt ein Textfeld mit der angegebenen Breite in Zeichen und initialisiert es mit der Zeichenkette

```
TextField hello = new TextField(´´Hello´´, 20);
```

Methoden der Klasse TextField

- getText() - gibt den Text des Feldes als Zeichenkette aus
- setText(String) - setzt die angegebene Zeichenkette in das Feld
- getColumns() - gibt die Breite des Textfeldes aus
- select(int, int) - wählt den Text zwischen zwei Positionen aus
- selectAll() - wählt den gesamten Text im Feld aus
- isEditable() - Abfrage, ob der Text editierbar ist
- setEditable(boolean) - setzt die Editierbarkeit des Textes
- getEchoChar() - gibt Ersatzzeichen für das eingegebene Zeichen aus
- setEchoChar(char) - setzt Ersatzzeichen für das eingegebene Zeichen
- echoCharIsSet() - Abfrage, ob das Feld ein Ersatzzeichen hat

Textbereiche

- Textbereiche können eine beliebige Breite und Höhe haben
- Standardmäßig haben sie auch Bildlaufleisten, so daß größere Textmengen untergebracht werden können
- Konstruktoren:
  - TextArea() erzeugt einen leeren Textbereich mit einer Länge von 0 Zeilen und einer Breite von 0 Spalten
  - TextArea(int, int) erzeugt einen Textbereich mit einer bestimmten Anzahl an Zeilen und Spalten (Zeichen)
  - TextArea(String) erzeugt einen Textbereich, der die Zeichenkette anzeigt und 0 Zeilen mal 0 Spalten groß ist
  - TextArea(String, int, int) erzeugt einen Textbereich mit einer bestimmten Anzahl an Zeilen und Spalten, der den bezeichneten Text anzeigt

Die Containerklasse Panel

- Panel können zur Zusammenfassung logisch zusammenhängender Komponenten verwendet werden

```
// Erzeugen des Panels
buttonPanel = new Panel();
// Hinzufügen von Buttons zum Panel
buttonPanel.add(new Button(´´clear´´));
buttonPanel.add(new Button(´´copy´´));
// Erzeugen des TextFeld
myTextField = new TextField("Hello",20);
// Hinzufügen der Komponenten zum Frame
myFrame.add(´´South´´, buttonPanel);
myFrame. add(´´Center´´, myTextField);
```

Die Containerklasse Frame

- mit einem Frame kann für eine Applikation eine Benutzeroberfläche erzeugt werden
- Frame´s können eine Menüleiste haben, die am oberen Rand plaziert wird

```
setMenuBar(MenuBar);
```

- zur Erstellung einer Menüleiste können die Klassen MenuBar, Menu und MenuItem verwendet werden

Layout-Manager

- Layout-Manager definieren Methoden zur Platzierung von Komponenten innerhalb eines Containers
- Sinn und Zweck der Layout-Manager
  - keine absolute Positionierung
  - automatische Anpassung der Größe
  - kein zu stark variierendes Erscheinungsbild
- standardmäßig eingestellte Layout-Manager
  - Panel => FlowLayout
  - Frame => BorderLayout

Anwendung von Layout-Managern

- Erzeugung eines Exemplars des gewünschten Layout-Managers und Registrierung des Layout-Managers beim Container

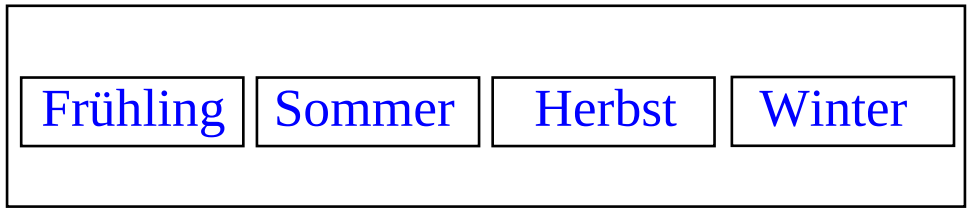
```
Frame frame = new Frame();
frame.setLayout(new FlowLayout());
```

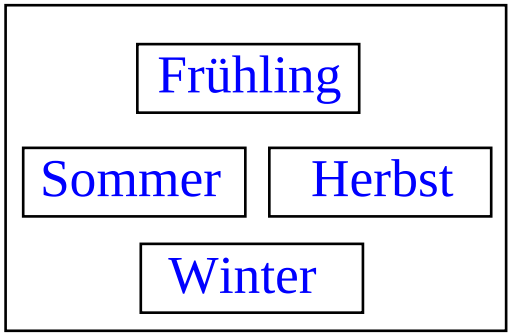
- Hinzufügen der Komponenten

```
frame.add(new Button(´´Cancel´´));
```

FlowLayout

- Komponenten werden von links nach rechts angeordnet
- passt eine Komponente nicht mehr in eine Zeile, wird sie automatisch auf die nächste Zeile umgebrochen
- ändert sich die Breite des Containers, passt der FlowLayout-Manager die Darstellung entsprechend an

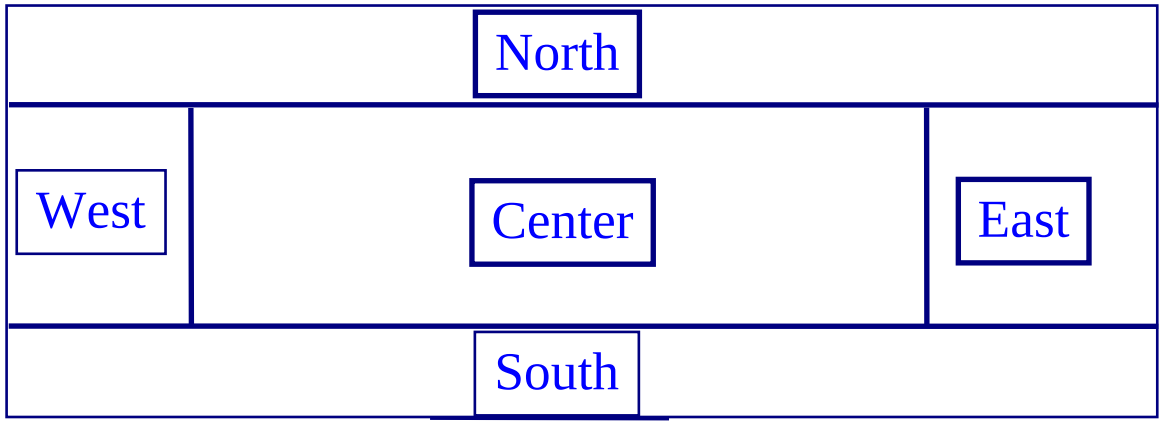




- Komponenten werden standardmäßig zentriert ausgerichtet

  FlowLayout.CENTER, FlowLayout.RIGHT bzw. FlowLayout.LEFT

BorderLayout



- bei der Anordnung der Komponenten wird die geographische Richtung angegeben
- die Komponenten am Rand bekommen nur den benötigen Platz
- die Komponente in der Mitte bekommt den restlichen Platz

```
BorderLayout bl = new BorderLayout();
setLayout(b1);
add(´´North´´, new Label(´´Titel´´));
```