Informatik B - Objektorientierte Programmierung in Java

Vorlesung 11:
GUI 1: AWT (1)

© SS 2005 Prof. Dr. F.M. Thiesing, FH Dortmund

Inhalt

- GUI-Programmierung mit Java
- AWT-Komponenten
- AWT-Steuerelemente
- AWT-Layoutmanager

GUI-Programmierung mit Java

- Alle unsere bisherigen Anwendungsbeispiele haben auf eine grafische Benutzungsoberfläche verzichtet.
- Die Programmierung von GUIs (engl. graphical user interface) gehört aber zu den wesentlichen Funktionalitätsbereichen, die die Java2-Plattform zur Verfügung stellt:
  - Aufbau von GUIs mit dem AWT
  - Aufbau von GUIs mit den Swing-Bibliotheken

GUI-Programmierung mit Java

- Zur Erstellung grafischer Benutzungsoberflächen besitzt Java (seit dem JDK 1.0) das AWT (abstract windowing toolkit), das im Paket `java.awt` leistungsfähige Klassen enthält.
- Dazu gehören:
  - Fenstertypen,
  - Steuerelemente (controls)
  - Klassen, die das Layout von Steuerelementen regeln (layout manager)
  - Klassen zur Menüsteuerung und zur Manipulation von Farben und Schriften
  - Grafikkontext

GUI-Programmierung mit Java

- Im JDK 1.1 wurde die Bearbeitung von Ereignissen auf das sog. Delegation-Event-Model umgestellt.
- Ab dem JDK 1.2 stehen in Java 2 mit den sog. Swing-Klassen leistungsfähige GUI-Bausteine zur Verfügung.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Komponenten

- Alle visuell darstellbaren Elemente in Java sind aus Basisklassen abgeleitet, die die benötigte Funktionalität der verschiedenen Steuerelemente bereitstellen.
  - Die Basisklasse des AWT ist `Component`.
  - Jedes Element eines GUI ist eine Komponente und daher von `Component` abgeleitet. (Ausnahme: Klassen für den Aufbau von Menüs sind aus `MenuComponent` abgeleitet.)

© Prof. Dr. Thiesing, FH Dortmund

AWT-Komponenten

- Die wichtigste Unterklasse von `Component` ist die Klasse `Container`. Sie ist der Ausgangspunkt aller Komponenten, die die Eigenschaft haben, selbst Komponenten enthalten zu können.
- Durch das Hinzufügen von Komponenten zu einem `Container` (mit Hilfe der Operation `add()`) entsteht eine Komponentenhierarchie, bei der jede Komponente ihren unmittelbaren `Container` als „Elternkomponente“ (`parent`) zugeordnet bekommt.
- Die Klasse `Component` ist eine abstrakte Klasse.

© Prof. Dr. Thiesing, FH Dortmund

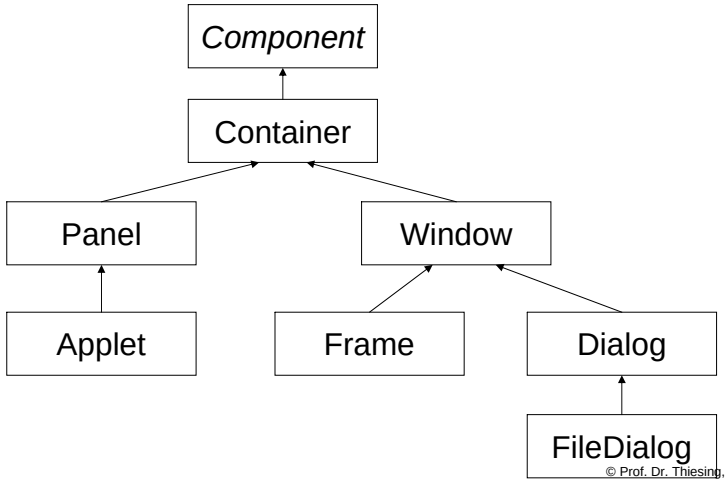
AWT-Komponenten

- Unterhalb von `Component` und `Container` finden sich die konkreten Gestaltungselemente des AWT, die zum Aufbau einer Benutzungsoberfläche zu verwenden sind.
- Dazu gehören Fensterklassen (`Window`, `Frame`, `Dialog`, `FileDialog`) und Klassen für Steuerelemente.
- Die Fensterklassen sind Containerklassen, da sie dazu dienen, weitere Elemente aufzunehmen.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Komponenten

- Klassenhierarchie (Ausschnitt)



© Prof. Dr. Thiesing, FH Dortmund

AWT-Komponenten

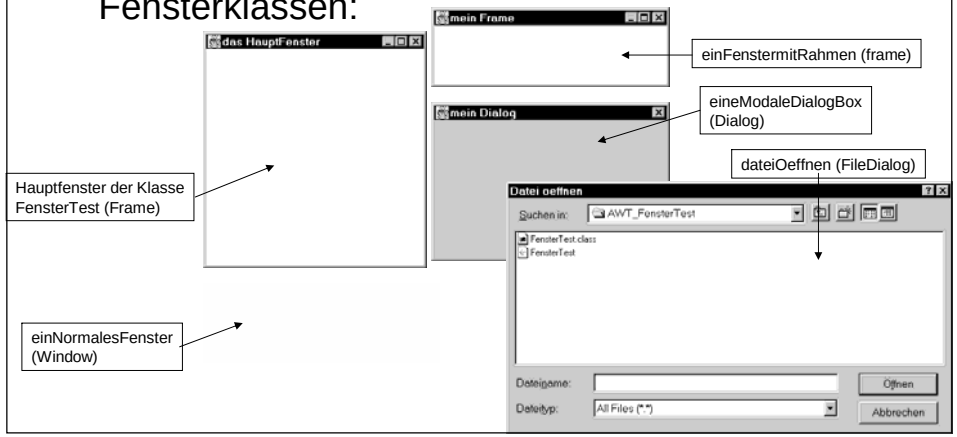
- Folgende Klassen dienen als Unterklassen von `Container` der Anordnung von Bildschirmelementen und der Fensterprogrammierung

Fenstertyp	Bedeutung
`Panel`	„Darstellungsfläche“: Containerklasse zur Organisation von Steuerelementen; kann selbst Teil eines anderen Containers sein.
`ScrollPane`	Containerklasse, die nur eine einzige Komponente enthält und die zu deren Betrachtung Rollbalken anbietet (scroll bars).
`Dialog`	Basisklasse für Dialogmasken. Der Dialog kann modal sein, d.h. das Programm blockieren, alle Benutzereingaben des Programms beanspruchen und andere Fenster erst dann wieder zum Zuge kommen lassen, wenn das modale Dialogfenster geschlossen wird.
`FileDialog`	Vordefinierte Klasse für „Datei öffnen“/“Datei speichern“-Dialoge
`Frame`	Fenster mit Rahmen, Titelleiste und Menü
`Window`	Einfaches Fenster (ohne Rahmen, Titelleiste und Menü)

© Prof. Dr. Thiesing, FH Dortmund

AWT-Komponenten

- Das folgende Beispiel (`FensterTest`) zeigt die Unterschiede zwischen den einzelnen Fensterklassen:



AWT-Steuerelemente

- Im AWT gibt es die folgenden Steuerelemente:
  - Button
    - Die Klasse `Button` stellt eine beschriftete Schaltfläche dar.
  - Label
    - Die Klasse `Label` erzeugt eine Textzeile, die auf dem Bildschirm angezeigt und vom Programm geändert werden kann.
  - TextField
    - Die Klasse `TextField` dient zur Eingabe und Anzeige einer Textzeile.
  - Canvas
    - Die Klasse `Canvas` ist eine Zeichenfläche zur Bildausgabe.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Steuerelemente

- TextArea
  - Die Klasse `TextArea` dient zur Eingabe und Anzeige mehrerer Textzeilen.
- Choice
  - Die Klasse `Choice` realisiert ein Feld, das durch Mausklick eine Liste aufblendet, woraus ein Eintrag ausgewählt werden kann.
- List
  - Die Klasse `List` ermöglicht die Auswahl von einem oder mehreren Einträgen aus einer Liste, die mit einem Sichtfenster bestimmter Größe und ggf. mit Rollbalken versehen ist.

© Prof. Dr. Thiesing, FH Dortmund

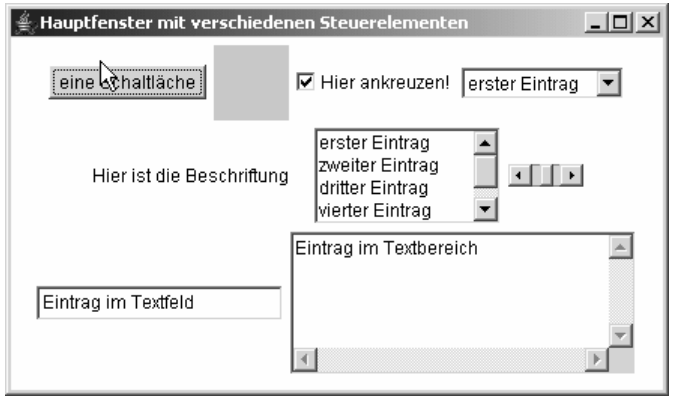
AWT-Steuerelemente

- Checkbox
  - Die Klasse `Checkbox` ist ein Markierungsfeld, das den Zustand „markiert“ oder „nicht markiert“ annehmen kann.
- Scrollbar
  - Die Klasse `Scrollbar` implementiert einen horizontalen oder vertikalen Schieberegler, mit dem ganzzahlige Werte aus einem vorgegebenen Wertebereich eingestellt werden können.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Steuerelemente

- Das folgende Programm (`ControlTest`) zeigt exemplarisch die Verwendung der einzelnen Steuerelemente:



© Prof. Dr. Thiesing, FH Dortmund

AWT-Steuerelemente

- Als Beispiel für die verschiedenen Steuerelemente soll die Klasse `Button` näher betrachtet werden.
- Sie steht stellvertretend für alle Steuerelemente, über die Aktionen ausgelöst werden können, und hat folgende Position in der Klassenhierarchie:
  - `java.lang.Object`
    - `java.awt.Component`
      - `java.awt.Button`
- Da sie nicht von `Container` abgeleitet ist, handelt es sich um eine einfache Komponente, die keine anderen Komponenten aufnehmen kann.

© Prof. Dr. Thiesing, FH Dortmund

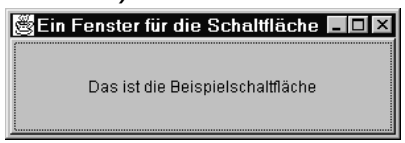
AWT-Steuerelemente

- `Button` verfügt über zwei Konstruktoren:
  - `Button()`
    - erzeugt einen Button ohne Beschriftung
  - `Button(String label)`
    - erzeugt einen Button mit Beschriftung
- Operationen
  - `void setLabel(String label)`
    - setzen der Beschriftung
  - `String getLabel()`

© Prof. Dr. Thiesing, FH Dortmund

AWT-Steuerelemente

- Beispiel (noch ohne Funktion des Buttons)




```
import java.awt.Frame;
import java.awt.Button;

class ButtonFenster
{
  public static void main(String[] args)
  {
   Frame buttonFenster =
         new Frame("Ein Fenster für die Schaltfläche");
   buttonFenster.setSize(300, 100);

   Button eineSchaltflaeche =
         new Button("Das ist die Beispielschaltfläche");
   buttonFenster.add(eineSchaltflaeche);
   buttonFenster.setVisible(true);
  }
}
```

© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- Überblick
  - Die Anordnung von Komponenten in einem Container erfolgt mit Hilfe von sogenannten Layoutmanagern.
  - Es gibt verschiedene Layoutmanager-Klassen, denen jeweils ein anderes Konzept zu Grunde liegt.
  - Allen gemeinsam ist, dass eine Platzierung durch Angabe der genauen Pixelwerte nicht erforderlich ist.
  - Ein Vorteil ist die automatische Anpassung der Größe der Komponenten bei Verkleinerung bzw. Vergrößerung des sie enthaltenden Containers.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- Für einen Container bestimmt man durch die Operation `void setLayout(LayoutManager mgr)`, welches Layout zu verwenden ist.
- Das AWT stellt folgende Layoutmanager-Klassen zur Verfügung:
  - `FlowLayout`,
  - `BorderLayout`,
  - `GridLayout`,
  - `CardLayout`,
  - `GridBagLayout`.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- Null-Layout
  - Ein sogenanntes Null-Layout wird durch Aufruf der Operation `setLayout(null)` im Container erzeugt.
  - Es wird kein Layoutmanager verwendet.
  - Alle Komponenten werden dann mit Hilfe der Component-Operationen `setLocation` und `setSize` oder einfacher mit `setBounds` pixelgenau positioniert.

■ Layoutmanager-Klassen

- Flow-Layout
  - `FlowLayout` ist der Standard-Layoutmanager für Objekte der Klasse `Panel`. Er ordnet die Komponenten zeilenweise von oben links nach unten rechts an.
  - Passt eine Komponente nicht mehr in die Zeile (z.B. nach Verkleinerung des Containers), so wird sie automatisch in der nächsten Zeile angeordnet.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- Border-Layout
  - `BorderLayout` ist der Standard-Layoutmanager für Objekte der Klasse `Window`.
  - Er ordnet maximal fünf Komponenten an den vier Seiten und im Zentrum des Containers an.
  - Die Platzierung einer Komponente mit `add(comp)` erfolgt grundsätzlich im Zentrum.
  - Mit `add(comp, pos)` wird die Komponente im Berich `pos` platziert, wobei für `pos` eine der String-Konstanten `NORTH, SOUTH, EAST, WEST, CENTER` der Klasse `BorderLayout` stehen darf.

© Prof. Dr. Thiesing, FH Dortmund

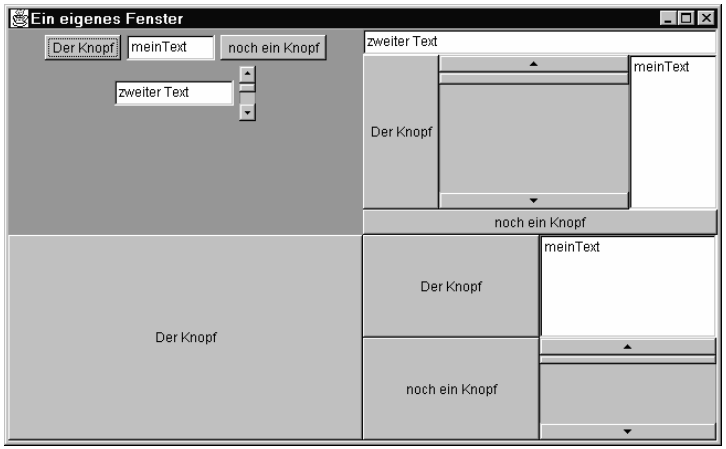
AWT-Layoutmanager

- Grid-Layout
  - `GridLayout` ordnet die Komponenten in einem Raster aus Zeilen gleicher Höhe und Spalten gleicher Breite an.
- Card-Layout
  - `CardLayout` stellt ein Layout aus mehreren sogenannten Karten zur Verfügung, die wie in einem Stapel übereinander gelegt und abwechselnd sichtbar gemacht werden können.
- Geschachtelte Layouts
  - Ein Container, der ein bestimmtes Layout aufweist, kann selbst wieder Container mit anderen Layouttypen enthalten, wie das folgende Beispiel zeigt.
  - Der Aufruf von **`pack()`** passt die Größe des Fensters an die zur Darstellung der Dialogelemente erforderlichen Platz an.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- Beispiel: **`AlleLayouts.java`**



© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- GridBag-Layout
  - Beim `GridBagLayout` handelt es sich um den aufwendigsten Layoutmanager, da nicht nur ein Raster Verwendung findet, sondern auch die enthaltenen Steuerelemente nicht auf eine gemeinsame Größe einer Rasterzelle zugeschnitten werden, d.h. sie können sich über mehr als eine Zelle in Höhe und Breite erstrecken.
  - Damit dies in sinnvoller Weise geschehen kann, müssen zusätzliche Informationen angegeben werden.
    - Für jede in ein `GridBagLayout` eingefügte Komponente kann man Layoutparameter mit Hilfe eines Objekts vom Typ `GridBagConstraints` angeben.
    - Zu dessen Eigenschaften gehören: Position im Raster, relative Gewichtung der Komponente, Orientierung der Verankerung, Füllmodus etc.

© Prof. Dr. Thiesing, FH Dortmund

AWT-Layoutmanager

- Beispiel: **`EingabeFormular.java`**
  - Eingabemaske für Adresseingaben
    - Jede Zeile soll aus einer Beschriftung und einem Eingabefeld bestehen.
    - Über die Constraints des `GridBagLayout` sollen die Beschriftungen rechtsbündig, die Eingabefelder linksbündig angeordnet werden.
    - Dazu werden für beide Typen von Steuerelementen `GridBagConstrainsts` definiert.