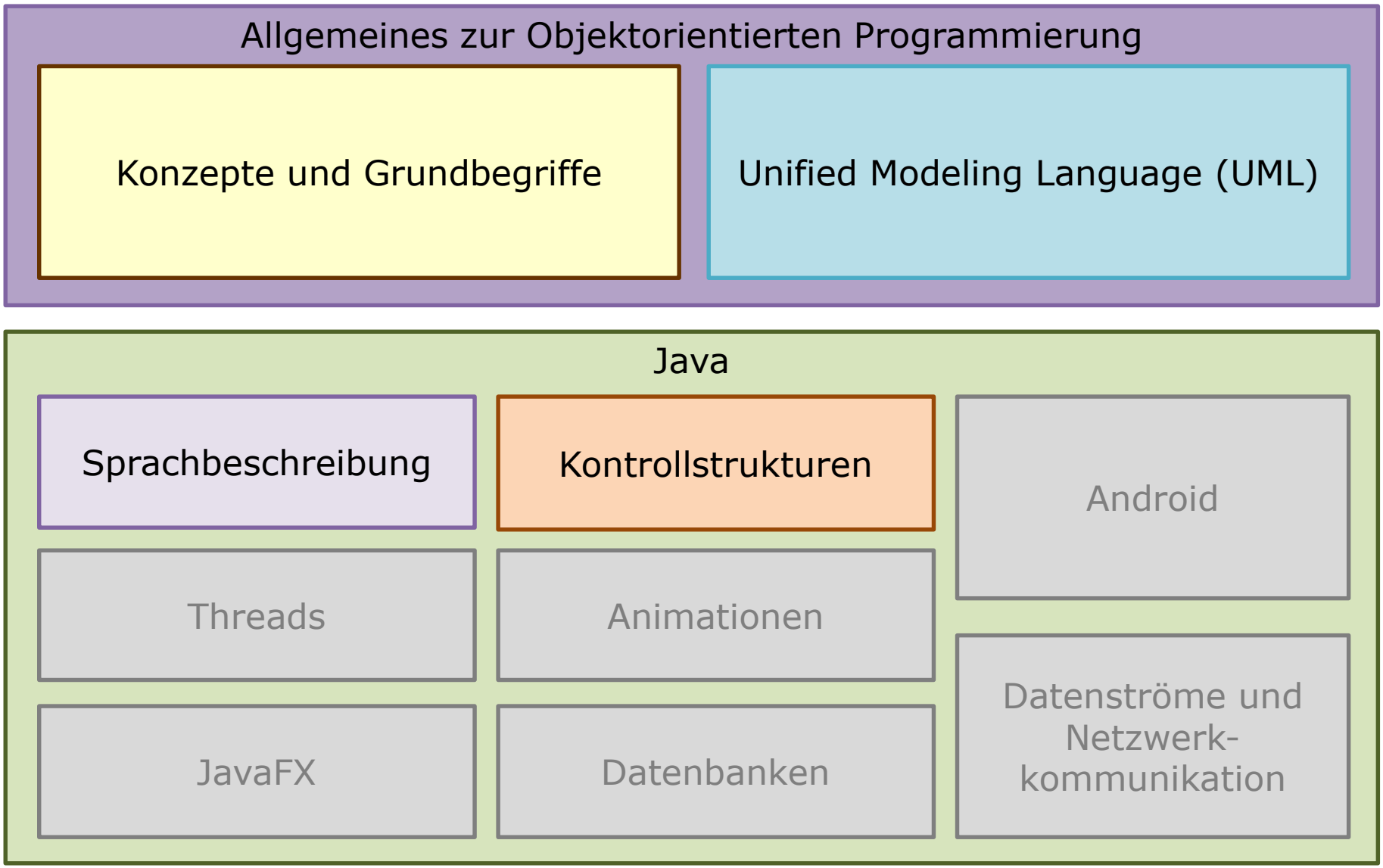
BERUFSAKADEMIE SACHSEN
STAATLICHE STUDIENAKADEMIE
DRESDEN
UNIVERSITY OF COOPERATIVE EDUCATION

Objektorientierte Programmierung

Java Foundation Classes (JFC)

Prof. Dr.-Ing. Tenshi Hara
tenshi.hara@ba-sachsen.de

Aufbau der Lehrveranstaltung



Bestandteile der JFC

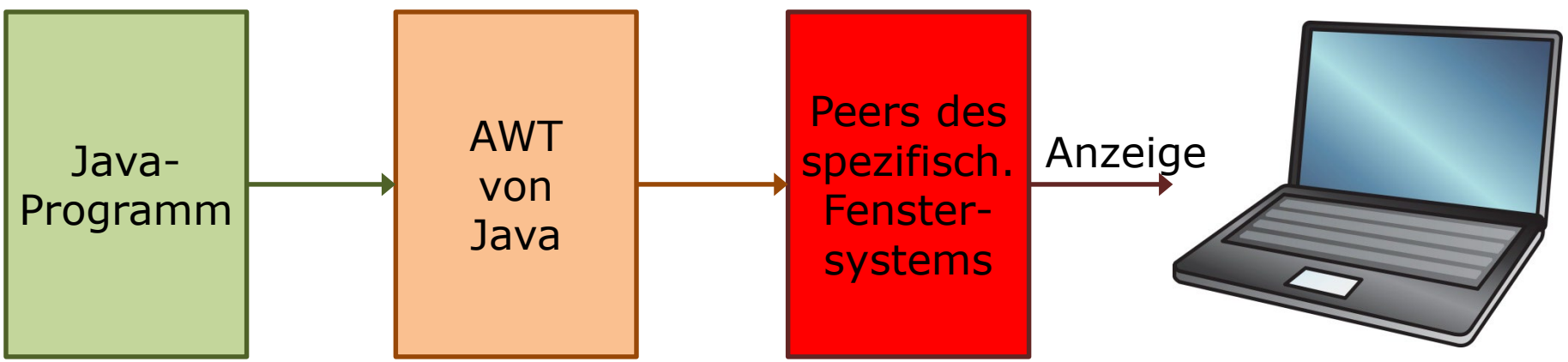
- Entwicklung grafischer Benutzerschnittstellen
- JFC bestehen aus drei User Interface Toolkits
  - Abstract Window Toolkit (AWT)
  - Swing-Komponenten und
  - Java-2D-API
- darin u.a. enthaltene Packages
  - Accessibility, Drag&Drop, Input Methods, Image I/O, Print Service und Sound
  - einige APIs verfügen über mehrere Packages
  - Java-2D-API enthält Klassen in `java.awt` und in `java.awt.image`

Pakete des AWT

- java.awt — Grundfunktionen
- java.awt.accessibility — Unterstützende Verfahren
- java.awt.color — Farben und Farbdefinitionen
- java.awt.datatransfer — Zwischenablage
- java.awt.dnd — Drag & Drop
- java.awt.event — Ereignisklassen und Listener
- java.awt.font — Font-Package von 2D-API
- java.awt.geom — Zeichen-Package von 2D-API
- java.awt.im — Eingabemethoden
- java.awt.image — Bildbearbeitung
- java.awt.peer — Peer-Schnittstellen
- java.awt.print — Druck-Unterstützung von 2D-API
- java.awt.swing — Swing-Komponenten
- java.awt.test — einzelnes Applet, das AWT-Funktionen testet

Plattformunabhängigkeit durch Peers

- AWT-Komponenten delegieren Funktionen an die Peers des entsprechenden Systems
- Peers sind spezifische Komponenten der jeweiligen grafischen Oberfläche



Vorteile und Nachteile von Peers

Vorteile

- schnelle Entwicklung anhand des AWT
- Abstraktion der Oberflächeneigenschaften

Nachteile:

- Komponenten haben kein einheitliches Aussehen unter verschiedenen Plattformen
- AWT-Heavyweight-Komponenten besitzen auch eventuell ungünstige Eigenschaften der Peers (unterschiedliche Zeilen-Begrenzer, Sichtbarkeit)
- eingeschränkter Umfang an grafischen Elementen

Hauptklassen des AWT

`Component`

- abstrakte Grundklasse: Menüs, Schaltflächen, Beschriftungen, Listen, …

`Container`

- abstrakte Grundklasse, welche die Klasse `Component` erweitert
- abgeleitete Klassen: u.a. `Panel`, `Applet`, `Window`, `Dialog` und `Frame`

`LayoutManager`

- Positionierung und Größe von Komponenten innerhalb eines Behälters

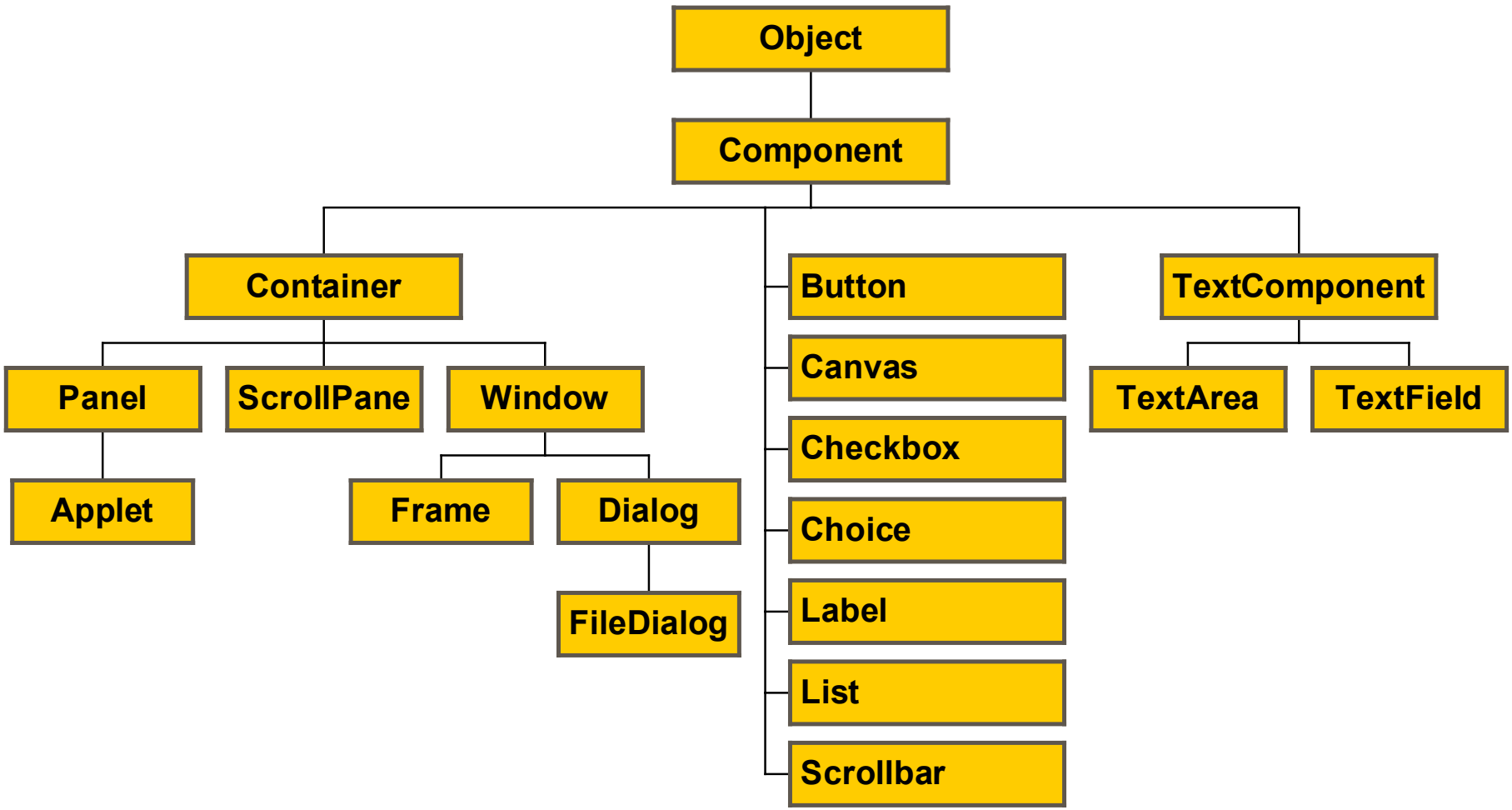
`Graphics`

- abstrakte Klasse für grafische Operationen

Komponenten des AWT

Name	Superklasse	Beschreibung
Applet	Panel	Basisklasse für Applets
Button	Component	Textschaltfläche
Canvas	Component	Leinwand für Grafiken
Checkbox	Component	markierbare boolsche Komponente
Choice	Component	Pop-down-Menü mit Texteinträgen
Dialog	Window	modal oder nichtmodal
FileDialog	Dialog	Dateiauswahl
Frame	Window	Fenster mit Titelleiste und Menüs
Label	Component	Anzeige einer Zeichenkette
List	Component	Bildlaufliste mit Texteinträgen
Panel	Container	Basisbehälter für Komponenten
Scrollbar	Component	veränderl. Komponente zum Elementbildlauf
ScrollPane	Container	Behälter mit Rollbalken
TextArea	TextComponent	mehrzeiliges, scrollbares Textfeld
TextComponent	Component	Superklasse von TextArea und TextField
TextField	TextComponent	einzeilige Komponente zur Texteingabe
Window	Container	randloses Fenster ohne Titel

Hierarchie der AWT-Komponenten



Grundfunktionen von Komponenten

- ein Graphics-Objekt
- Position und Größe
- spezifisches Peer
- übergeordneter Behälter
- Schriften und Schriftgrößen
- Vorder- und Hintergrundfarben
- lokale Bezeichner
- minimale, maximale und bevorzugte Größen

Ereignisse und Ereignisbehandlung

Ereignisbehandlung

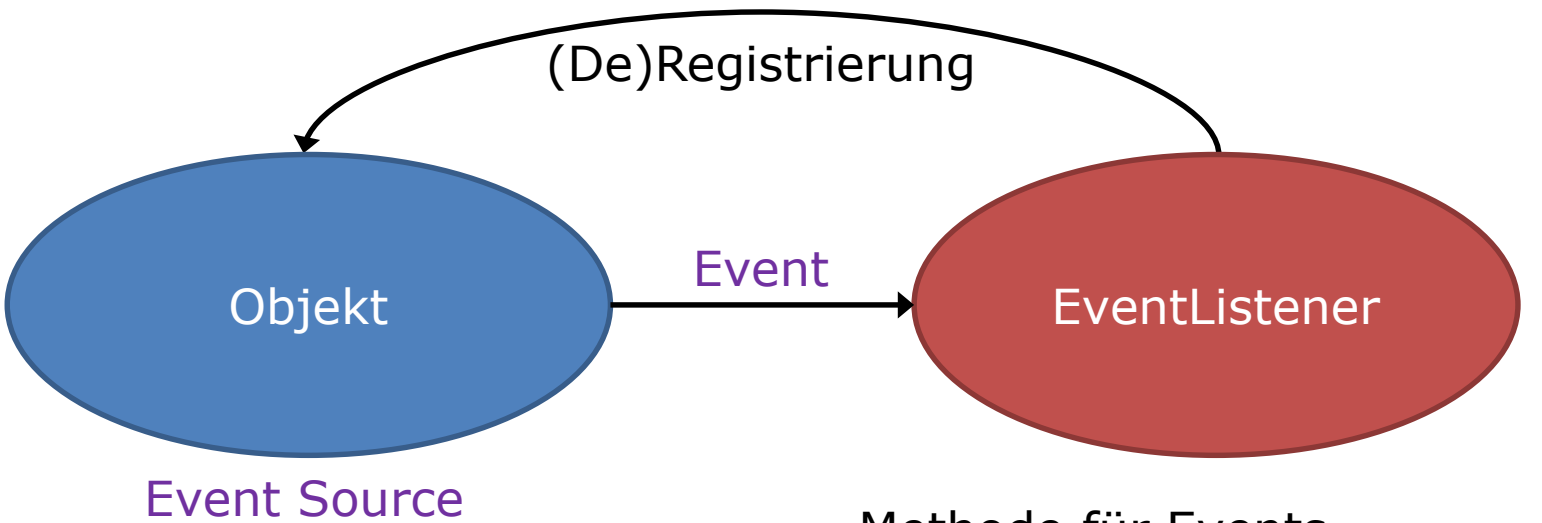
- Ereignisse sind Vorgänge bei Veränderungen von Objekten
  (Klick auf einen Knopf, Mausklick, Mausbewegung, Tastendruck, …)
- Objekte sind Ereignisquellen (Event Source)
  → sie können Meldungen verschicken (Events)
- in Java 1.0 wurden Ereignismeldungen an alle Elemente, in denen die Ereignisquelle eingebettet ist, geschickt
  - Behandlung in `handleEvent`
  - ineffizient und unflexibel; sollte nicht mehr verwendet werden

Ereignisbehandlung ab JDK 1.1

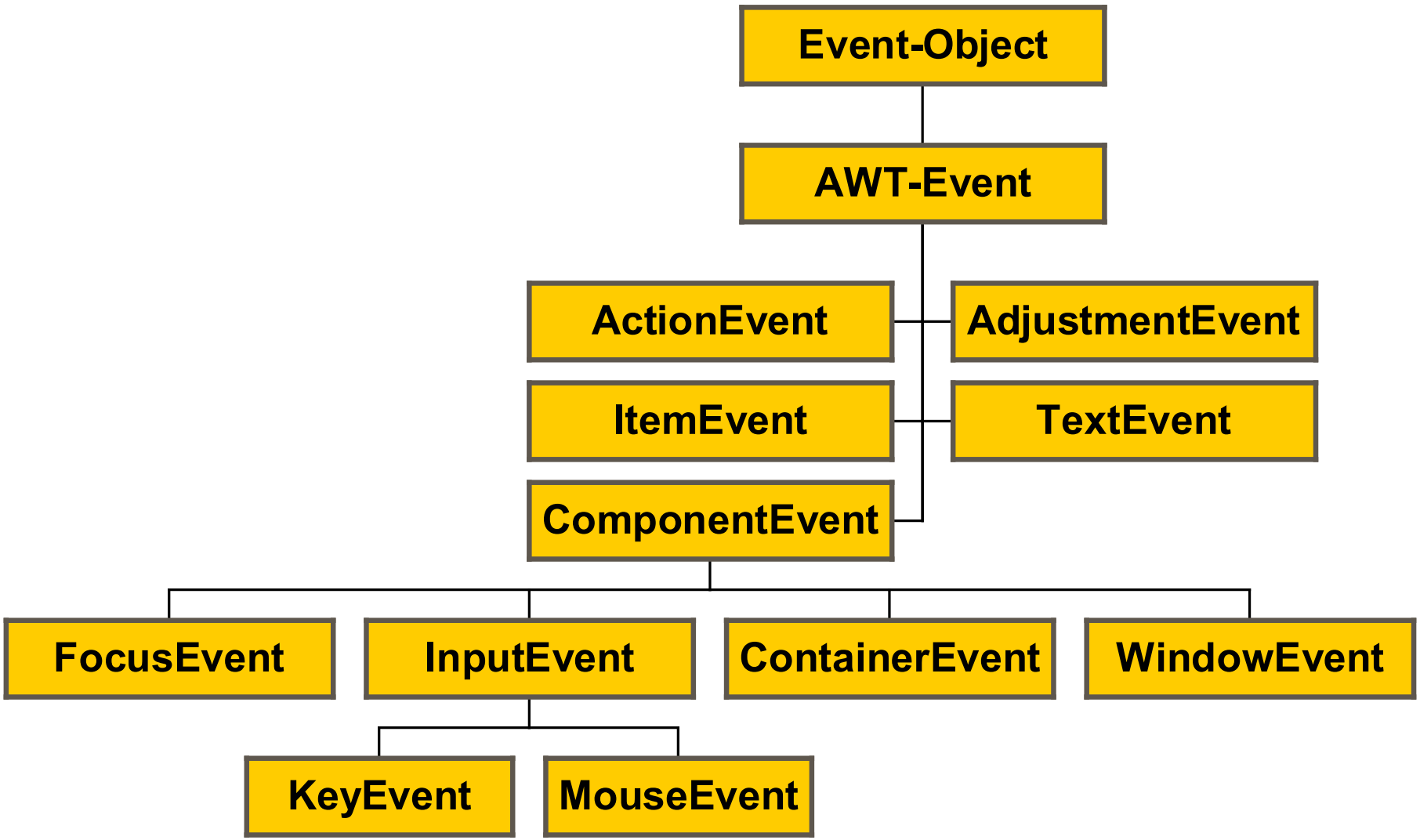
- ab Java 1.1 müssen sich alle Elemente bei den Ereignisquellen als Listener registrieren, wenn sie die Ereignisse verarbeiten wollen
- Meldungen werden nur an registrierte Listener geschickt
- Ereignisquellen verwalten eine Liste von Listenern
- diese Form des Eventhandling wird als Delegation bezeichnet
- folgende Schritte sind durchzuführen
  - Anlegen der Listener durch entsprechende Klassen bzw. Schnittstellen
  - Registrieren der Listener (`xxx.addXxxListener`)
  - Implementieren der entsprechenden Methoden (`actionPerformed`, `mousePressed`, …)

Events und Listener

- LowLevel-Events (bspw. Mausaktivitäten)
- HighLevel-Events (bspw. Menüeinträge, Listboxen)
- Interesse für Event (Aufruf einer Registrierungsmethode)
- `addXxxListener` und `removeXxxListener`



Hierarchie der Ereignisklassen



Komponenten und Listener – Zusammenfassung

Komponente / Listener	Button	Canvas, Label	Checkbox	Checkbox-MenuItem	Choice	Container, Panel	Dialog, Frame	List	MenuItem	Scrollbar	ScrollPane	TechtArea	TextField	Window
Action	✓							✓	✓				✓	
Adjustment										✓				
Component	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Container						✓	✓				✓			✓
Focus	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Item			✓	✓	✓									
Key	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Mouse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Text												✓	✓	
Window							✓							✓

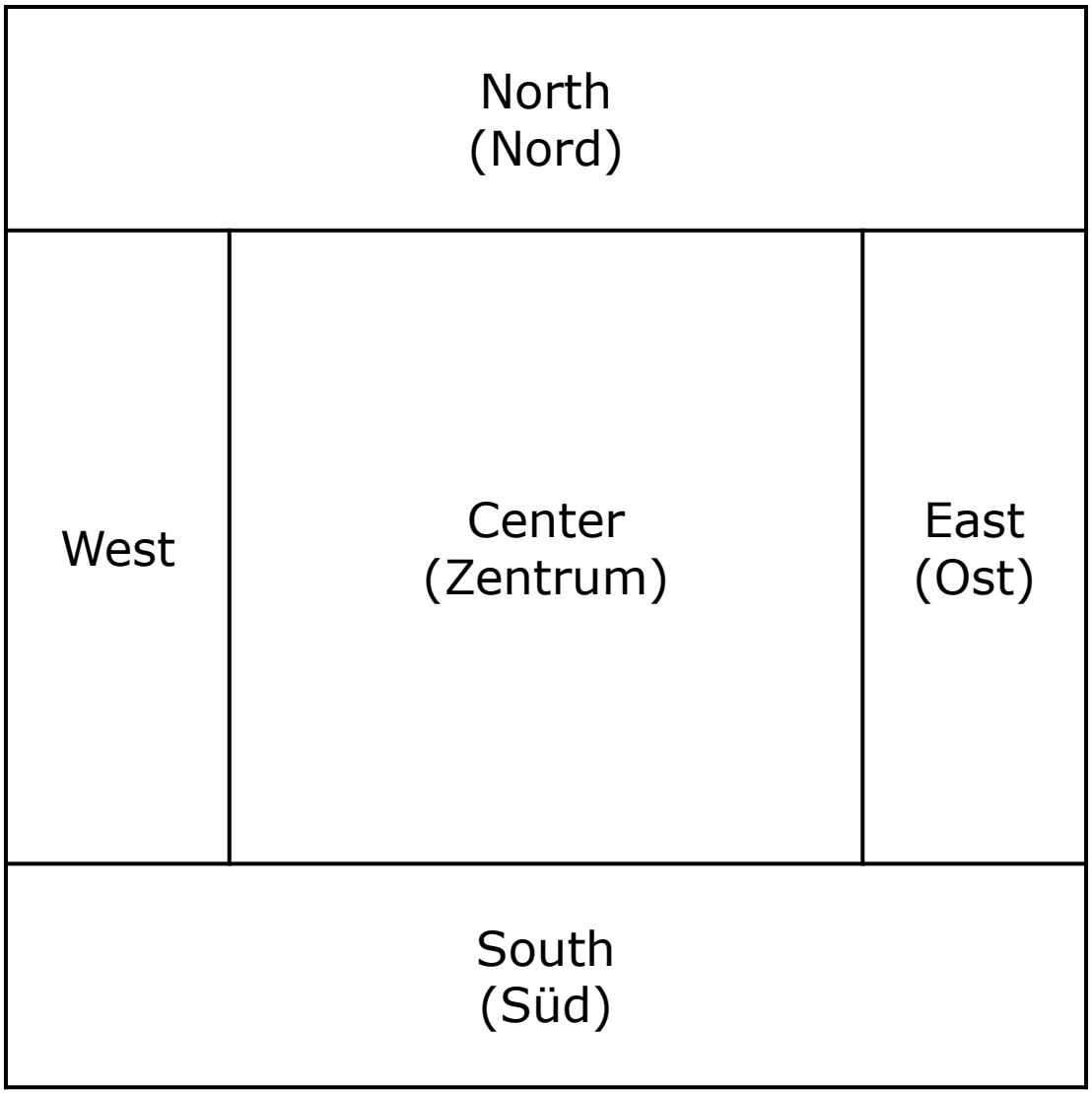
Positionierung von Elementen

Layout-Manager

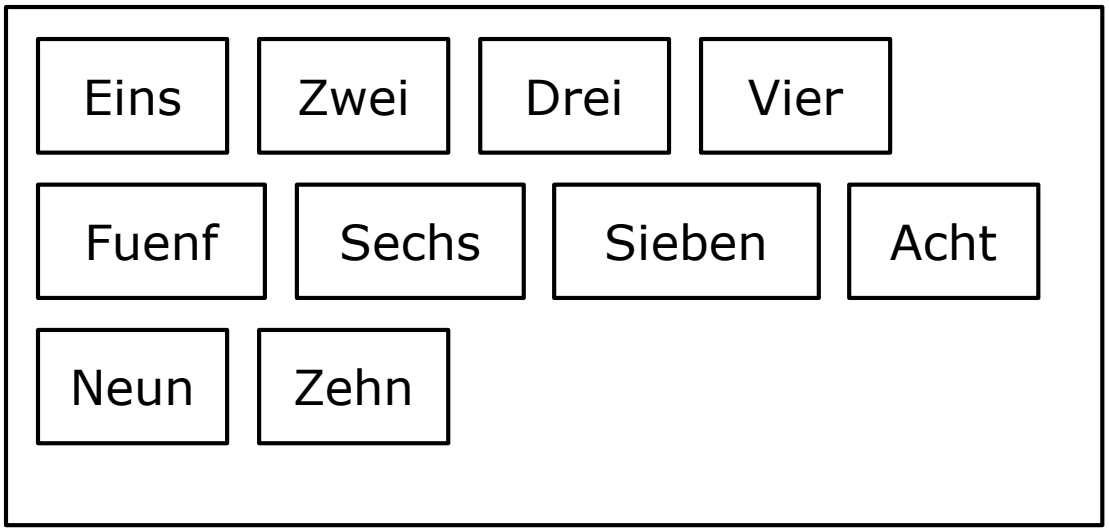
- automatische Anordnung von Komponenten entsprechend des gewählten Layout-Managers
- Anordnung auch bei veränderter Größe der Darstellungsfläche
- einfaches Hinzufügen neuer Komponenten
- Möglichkeit, den Layout-Manager auszuschalten und die Komponenten manuell zu platzieren

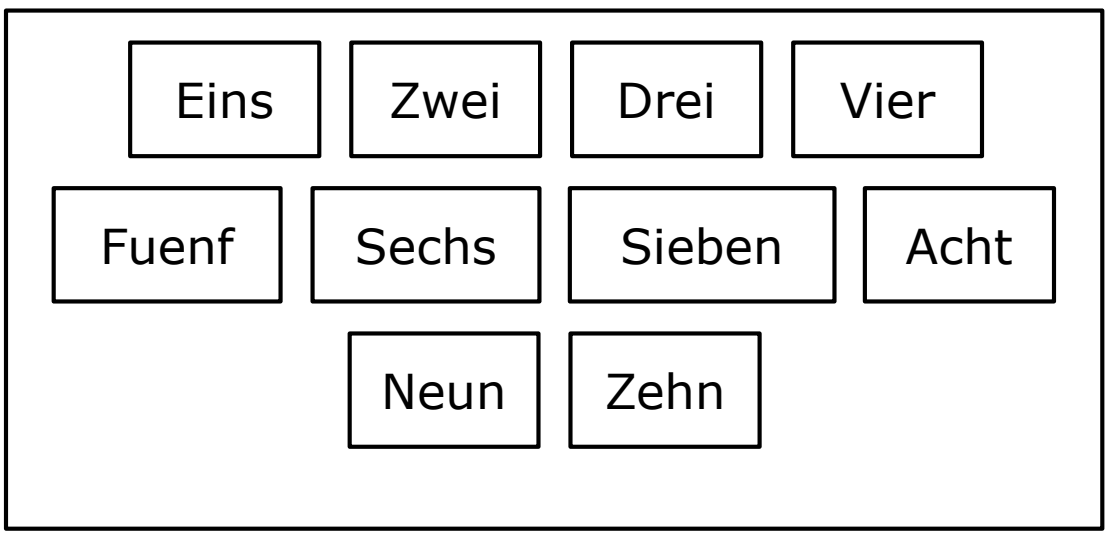
```
setLayout(null);
// …
xxx.setLocation(PixelX, PixelY);
// (xxx ist der Name der entsprechenden Komponente)
xxx.setSize(BreitePixel, HoehePixel);
// (xxx ist der Name der entsprechenden Komponente)
```

BORDER-LAYOUT



FLOW-LAYOUT

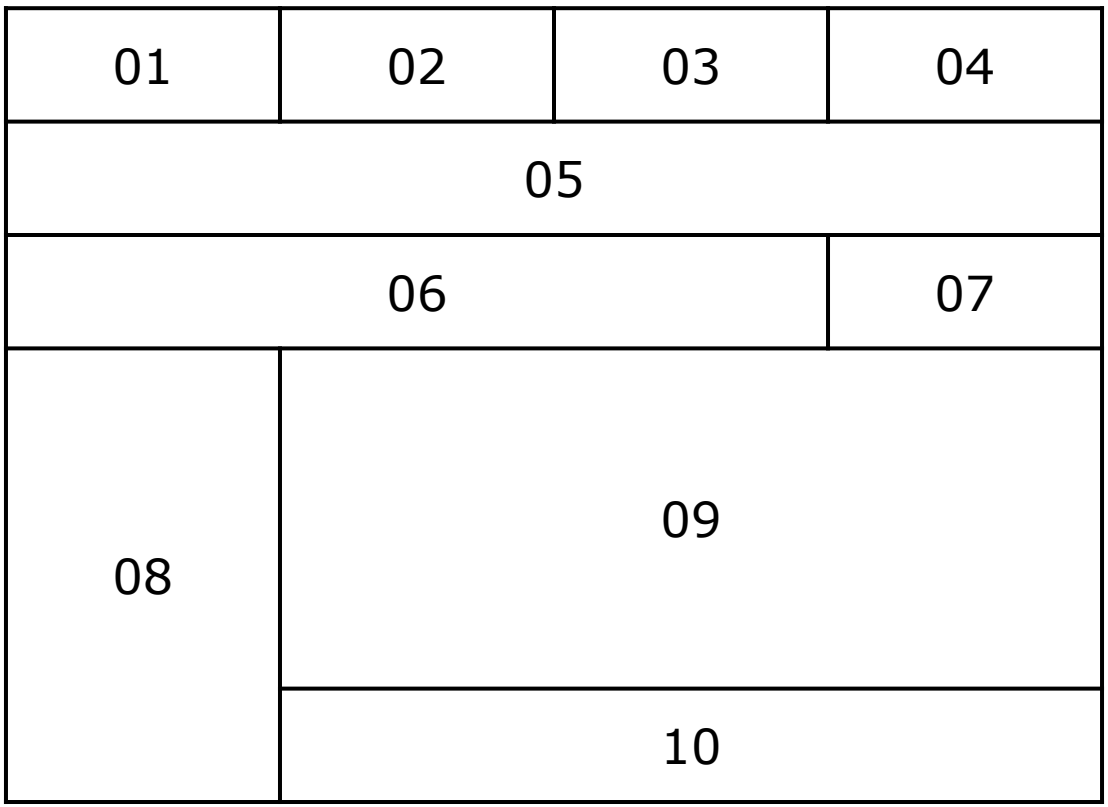

Eins
Zwei
Drei
Vier
Fuenf
Sechs
Sieben
Acht
Neun
Zehn


Eins
Zwei
Drei
Vier
Fuenf
Sechs
Sieben
Acht
Neun
Zehn

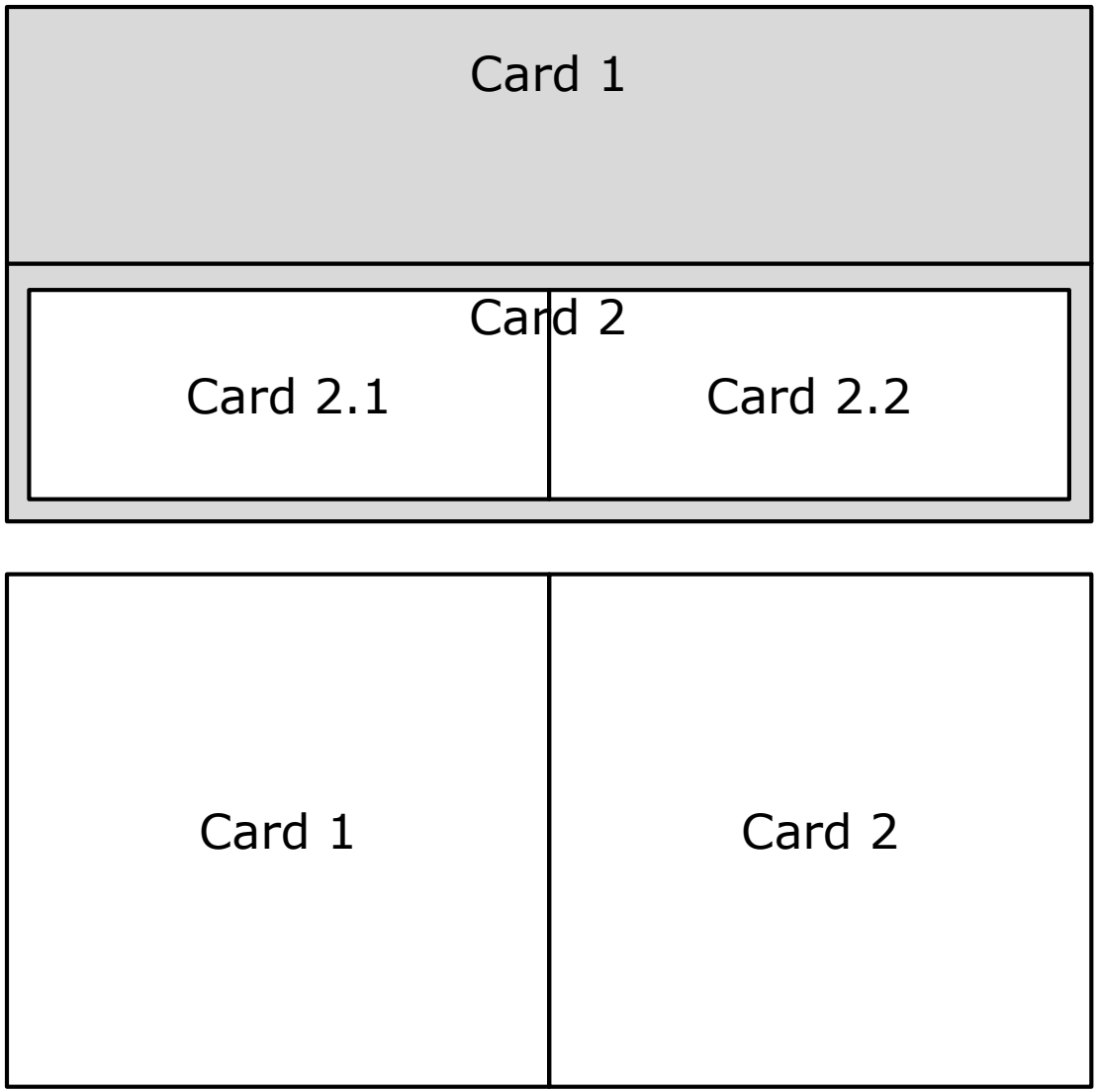
GRID-LAYOUT

01	02	03	04
05	06	07	08
09	10	11	12
13	14	15	16

GridBag-Layout



CARD-LAYOUT



Nachteile des AWT

- nur einfache grafische Benutzeroberflächen möglich
- größter gemeinsamer Teiler der Betriebssysteme
- fehlende von Grundfunktionen
  (Zwischenablage, Drucken, Popup-Menüs, Bildlaufleisten, …)
- Peer-basierte Architektur, dadurch
  - Hülle um komplexe, native Peers
  - jeder Peer in einem eigenen nativen Fenster
  - Peer-Fehler und Kompatibilitätsprobleme
    (zwischen verschiedenen Betriebssystemen)
- nicht solide objektorientiert
- nicht auf dieser Basis erweiterbar
- eigene Werkzeugsammlungen von Drittanbietern

Heavyweight und Lightweight

- Heavyweight-Komponenten sind mit einer Peer-Komponente verbunden und werden in einem eigenen undurchsichtigen Fenster dargestellt
- Lightweight-Komponenten haben kein natives Peer und werden im Fenster eines Heavyweight-Behälters dargestellt
  - sie können durchsichtigen Hintergrund haben
    → dadurch optisch nicht notwendig rechteckig
    → wohl aber mit rechteckiger Begrenzung
  - Swing-Behälter sind Heavyweight-Komponenten (Frames, Windows, Applets, Dialoge)
- für alle AWT-Komponenten gibt es entsprechende Swing-Komponenten

Zusätzliche Swing-Komponenten

- JDesktopPane, InternalPane — für MDI-Interface
- JEditorPane, JTextPane — editierbar
- JOptionPane — Popup-Fenster
- JPasswordField — Passwortfeld
- JProgressBar — Fortschrittsbalken
- JRadioButtonMenuItem — in Menü integrierbar
- JSeparator — Menü-Trennbalken
- JSlider — erweiterter Scrollbar
- JSplitPane — Container
- JTable — zweidimensionale Tabelle
- JToggleButton — bleibt gedrückt
- JToolbar — verschiebbar für Fenster
- JToolTip — Popup-Text
- JTree — Anzeige von Hierarchien

Pluggable Look & Feel

Metal java-eigenes Look & Feel

- Standard-Layout
- auf allen Plattformen verfügbar

Motif Unix-Oberflächen

- Motif ist Aufsatz auf X-Windows
- X-Windows ist Grafikerweiterung von Unix-Systemen

Windows Microsoft Windows

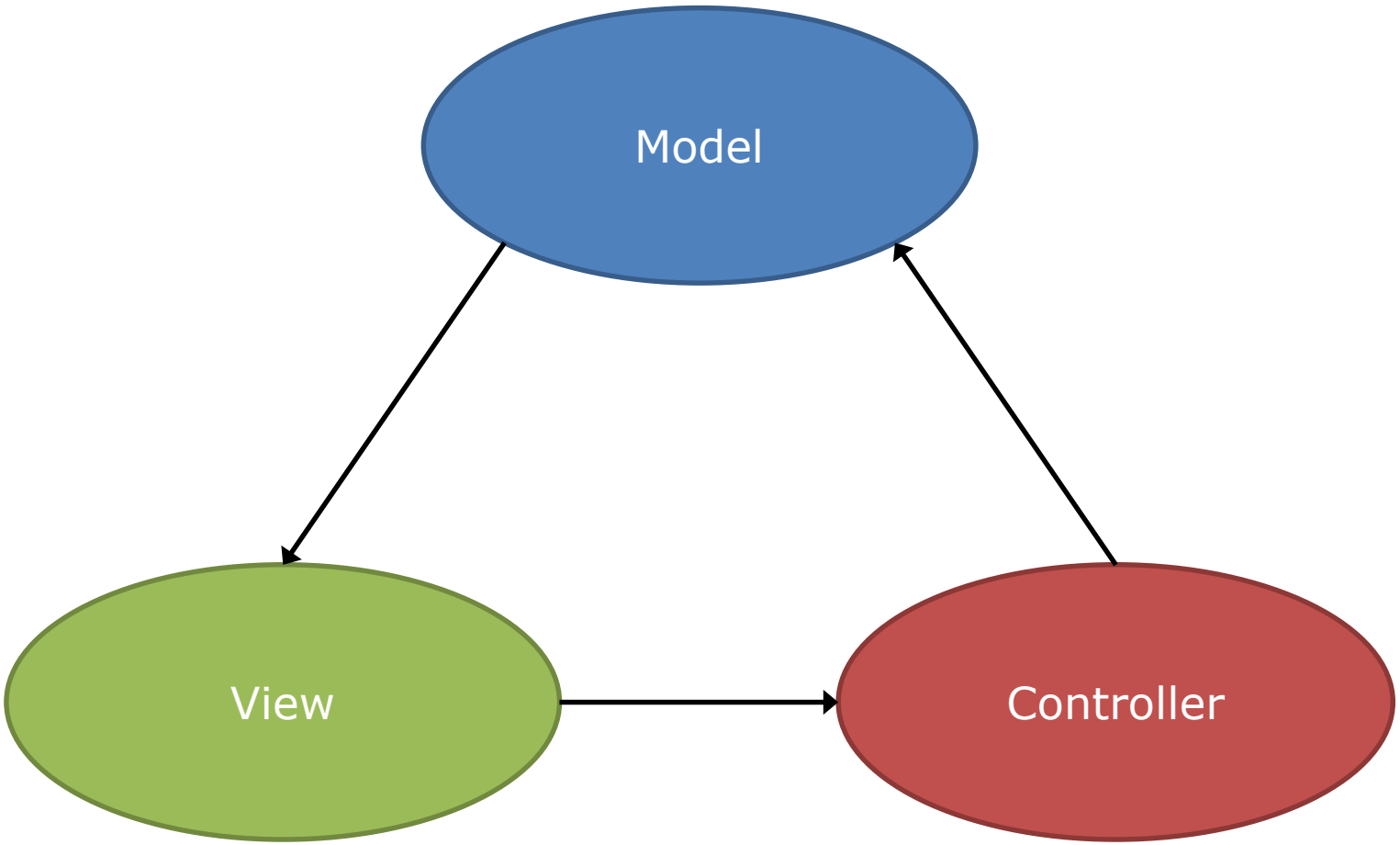
→ nur auf Windows-Plattformen verfügbar

Mac Apple Macintosh

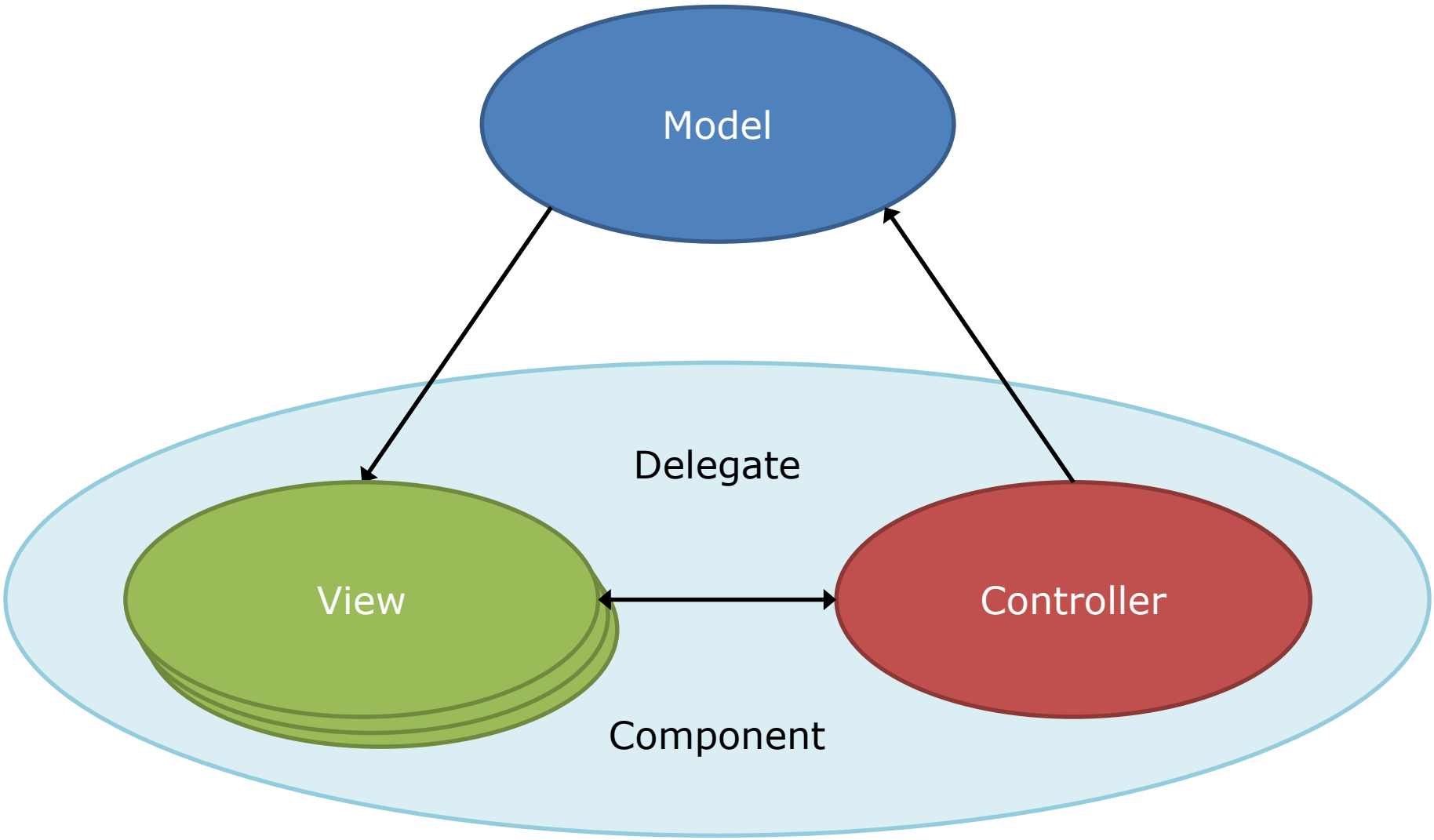
→ nur auf Macintosh verfügbar

Swing Entwurfsmuster

Wiederholung: klassisches MVC-Konzept



MVC-Konzept von Swing



Java-2D-API

- starke Erweiterung der Möglichkeiten zur Bildbearbeitung und der Zeichenfunktionen
- Attribute des Grafikkontextes
  - `Color` Farben
  - `Font` Schriftarten
  - `Stroke` Linienbreite
  - `Transform` Drehen, Verschieben
  - `Composite` Darstellung auf Ausgabegerät
  - `Clip` Clipping

ACCESSABILITY-API

- Anpassungsfähigkeit der Benutzerschnittstelle
- Einbindung von E/A-Geräten für Behinderte
  - Schriften zur besseren Lesbarkeit
  - Schaltflächen zur einfachen Aktivierung vergrößert darstellen
  - Braillezeile
  - Sprach-Ein- und -Ausgabe

LITERATURANGABEN

Java 2 SDK v 1.2.2, Grundlagen Programmierung, HERDT-Verlag für Bildungsmedien GmbH, Nackenheim

Guido Krüger: Handbuch der Java-Programmierung, Addison-Wesley Verlag, ISBN 3-8273-2201-4

Brit Schröter: KompaktReferenz Java2, DATA BECKER GmbH & Co. KG, ISBN 3-8158-1477-4

Internet HTML-Seiten, https://www.java.com/de/