

# Softwaretechnik

WiSe 09/10

Grafische Benutzerschnittstellen / Übungsblatt 10 (GUI für Wetterstation)

## Vorlesung 4.4

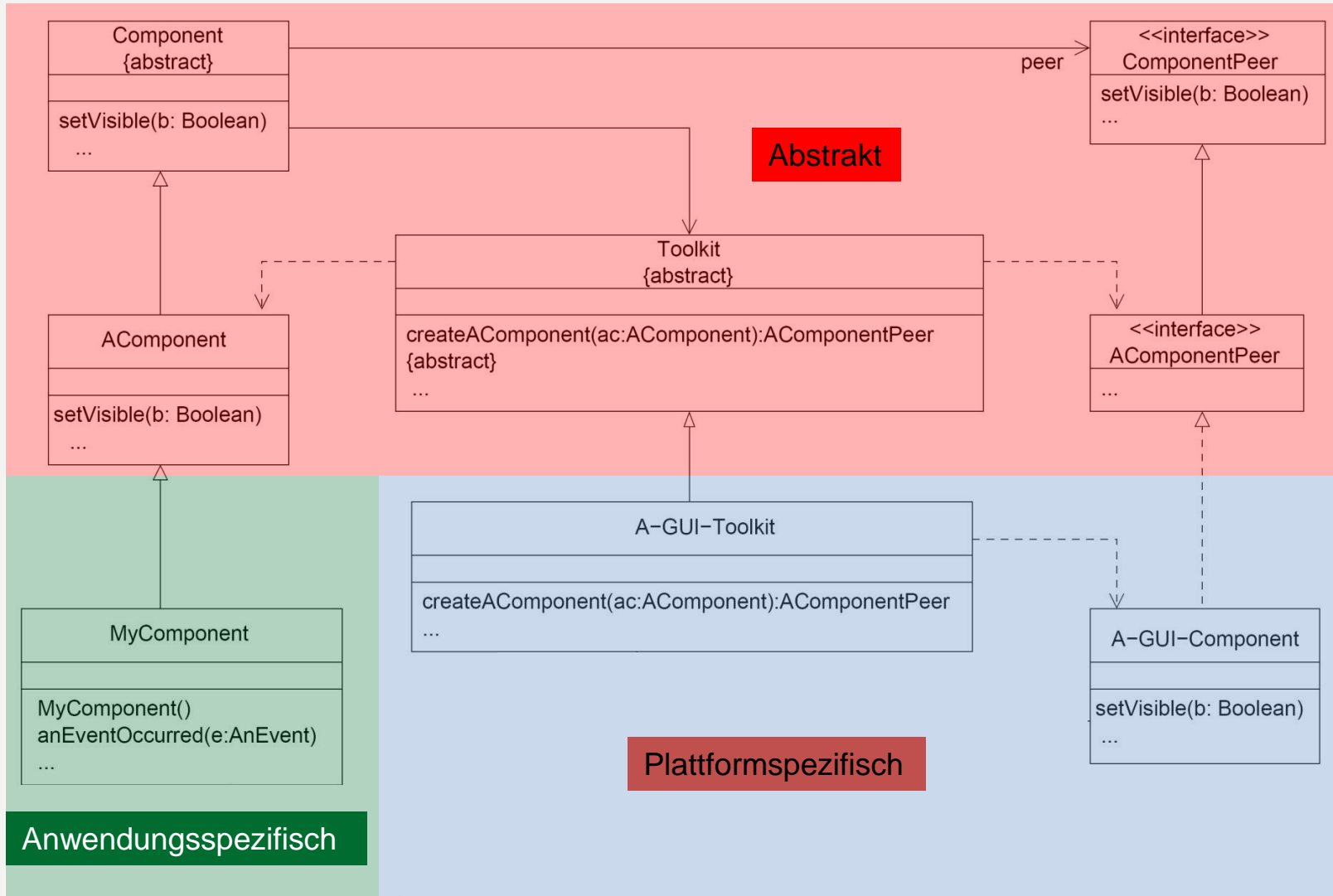
# GRAFISCHE BENUTZERSCHNITTSTELLEN

- Grafische Benutzerschnittstellen (GUIs, Graphical User Interfaces) bieten dem Benutzer Zugriff auf ein Anwendungssystem
- GUIs sind ereignisgesteuert: Der Benutzer löst Ereignisse (**Events**) aus (mit Maus, Tastatur, ...), die vom GUI-System empfangen und interpretiert werden
- Zur Programmierung von Benutzerschnittstellen verwendet man i.A. **GUI-Toolkits**
  - Ein GUI-Toolkit stellt vorgefertigte Interaktionselemente (**Widgets**) zur Verfügung (z.B. Window, Button, Checkbox, ...)
  - Individuelle GUI-Elemente können durch Spezialisierung gegebener Klassen definiert werden (Wiederverwendung)
- Beispiele für Java: AWT/Swing, SWT

- AWT (Abstract Window Toolkit) und Swing sind Teil der Java-Klassenbibliothek (entwickelt von Sun Microsystems)
- Bieten eine API zur Programmierung grafischer Benutzerschnittstellen (GUIs) für Java Programme
- Grundidee: plattformunabhängige Konstruktion von GUIs
- Beziehung AWT/Swing:
  - **AWT** wurde mit Java 1.0 eingeführt, Grundtechnologie
  - **Swing** wurde mit Java 1.2 eingeführt, baut auf AWT auf und ergänzt die dort angebotenen Widgets
  - (Aktuell: Java 1.6)

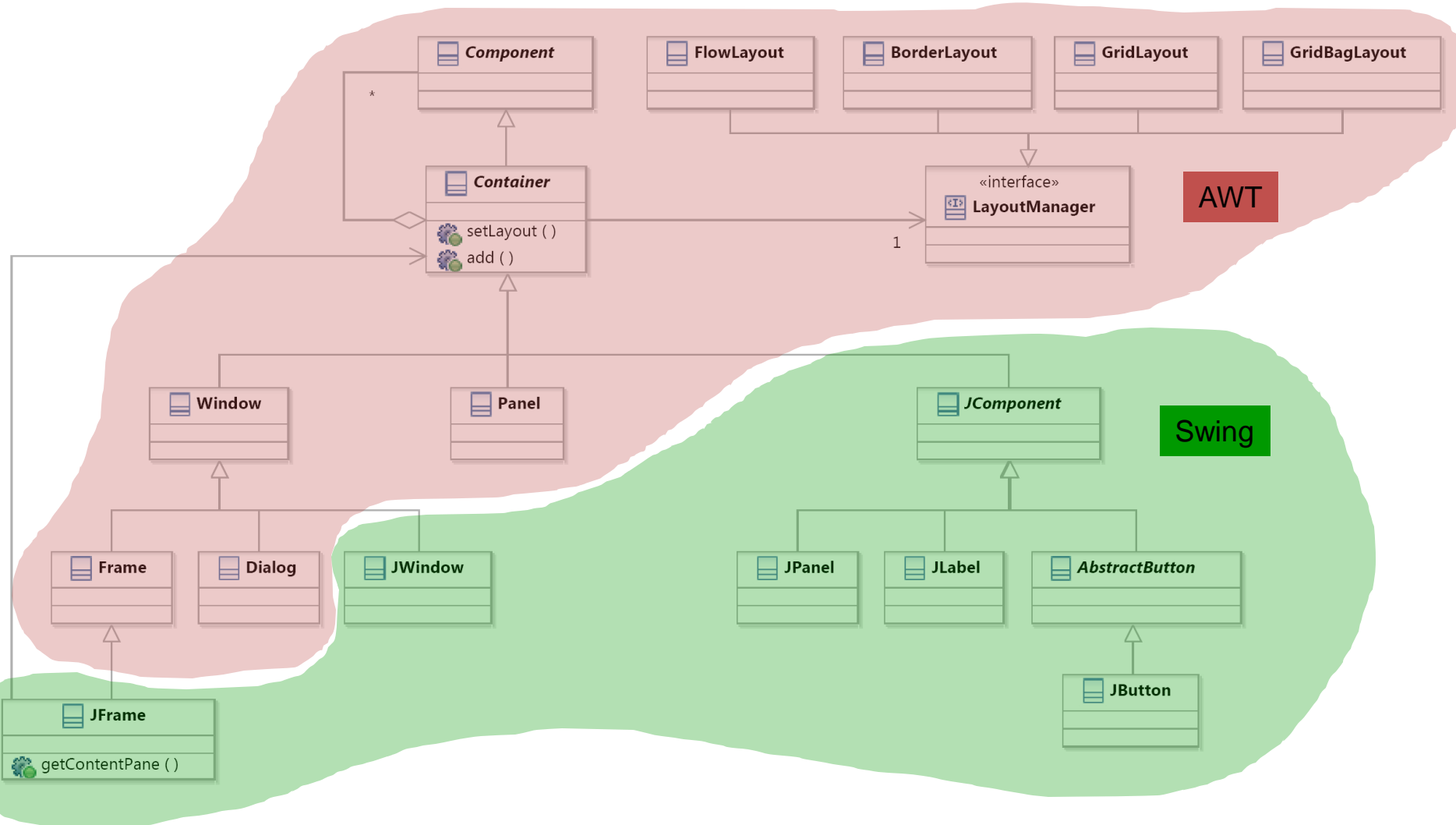


- Java, als plattformunabhängige Sprache, erfordert plattformunabhängige GUI-Spezifikation
- AWT/Swing definiert daher ein **abstraktes Toolkit**
  - ...enthält alle Komponenten und Factories
  - ...allerdings nur abstrakt (als Interfaces oder abstrakte Klassen)
- Für jede unterstützte Plattform gibt es konkrete Toolkits
  - Z.B. Windows, Linux, OSX
  - ...implementieren konkrete Factories und Komponenten





- AWT-Komponenten werden von einem Toolkit in entsprechende GUI-Komponenten einer speziellen Plattform (Native Components) übersetzt.
- Die plattformspezifischen GUI-Komponenten müssen ein entsprechendes **Peer-Interface** (des AWT) implementieren. Das Peer-Interface beschreibt die Anforderungen an die GUI-Komponente.
- Soll eine spezielle GUI-Plattform AWT unterstützen, dann muss ein entsprechendes GUI-Toolkit implementiert werden. Die abstrakte Klasse Toolkit (des AWT) beschreibt die Anforderungen an das GUI-Toolkit.
- Der Anwendungsentwickler muss die zur Anwendung gehörigen (plattformunabhängigen) GUI-Komponenten entwerfen (meist Swing-Komponenten).

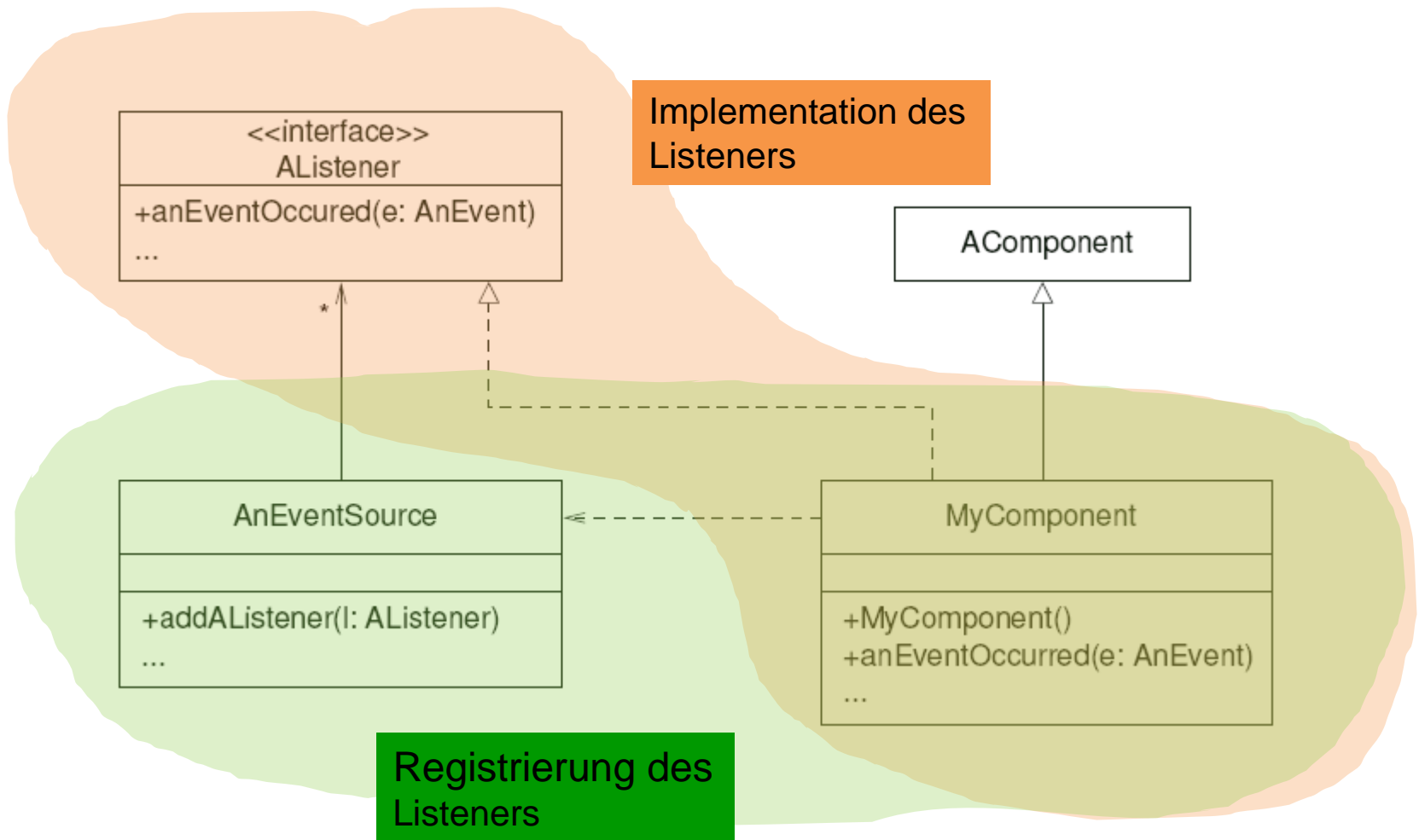




- AWT-Komponenten werden grundsätzlich in native Komponenten umgewandelt
- Bei Swing-Komponenten (mit **J** beginnende Klassen und einige weitere) wird unterschieden:
  - **Heavyweight-Komponenten** (JFrame, JDialog, JWindow, JApplet) werden ebenfalls in native Komponenten übersetzt
  - **Lightweight-Komponenten** (alle Spezialisierungen von Component) werden „von Hand“ gezeichnet
- (Echte) **Container** besitzen:
  - Eine **Content Pane**, in welche Child Components eingefügt werden
  - Einen **Layout Manager**, welcher die Anordnung der Child Components bestimmt



- Event Handler beschreiben Reaktionen auf Events:
- Tastendruck,
- Mausklicks,
- Änderungen in Textfeldern, ....
- Verwendung:
- Ein Event Handler (**Listener**) wird an einer **Event Source** (z.B. einem Button) registriert
- Die Methoden des Event Handlers werden dann bei Ereignissen aufgerufen (z.B. `actionPerformed()`).



- In AWT/Swing werden verschiedene Ereignisklassen unterschieden: KeyEvent, MouseEvent, ActionEvent, WindowEvent...
- Ist eine Komponente an Ereignissen eines bestimmten Typs interessiert, dann muss sie:
  1. sich bei der Komponente, in der ein solches Ereignis auftreten kann (AnEventSource) als Listener registrieren (addAListener).
  2. die beim Eintritt eines solchen Ereignisses aufgerufene Operation (anEventOccured) der passenden Listener-Schnittstelle (AListener) implementieren.
- Listener-Schnittstellen sind z.B. KeyListener, MouseListener, ActionListener, WindowListener.
- Operationen von Listener-Schnittstellen sind z.B. actionPerformed (von ActionListener), windowClosing (von WindowListener).

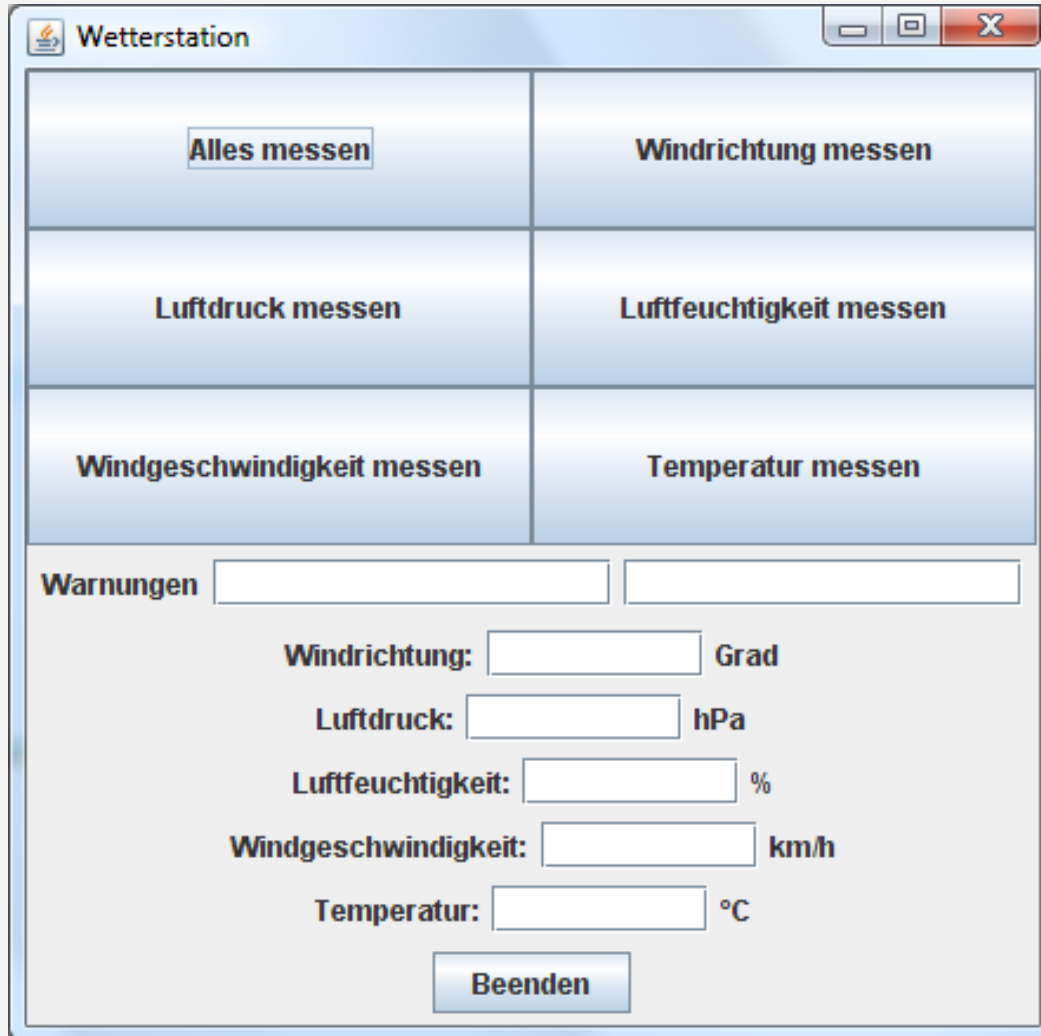
- Zur Programmierung von Benutzerschnittstellen verwendet man GUI-Toolkits.
- Anwendungsspezifische GUI-Elemente können durch Spezialisierung gegebener GUI-Klassen definiert werden.
- AWT/Swing bieten eine Klassenbibliothek zur plattformunabhängigen Programmierung von GUIs für Java Programme.
- Wesentliche Aufgaben bei der Realisierung einer GUI sind
  - die (statische) Konstruktion der GUI-Komponenten
  - die Programmierung der Ereignisbehandlung durch Implementierung entsprechender Listener-Interfaces

## Übung 10

# PROTOTYP-GUI WETTERSTATION

Es soll ein Prototyp für eine grafische Benutzeroberfläche (GUI) des Wetterstationssystems entwickelt werden. Gehen Sie dazu folgendermaßen vor:

1. Skizze eines Layouts für die Gestaltung der Oberfläche
2. Modellierung der GUI mit den benötigten AWT/Swing-Komponenten
3. Implementierung des Prototypen in Java unter Verwendung von AWT/Swing



**Wetterstation**

<b>Alles messen</b>	<b>Windrichtung messen</b>
<b>Luftdruck messen</b>	<b>Luftfeuchtigkeit messen</b>
<b>Windgeschwindigkeit messen</b>	<b>Temperatur messen</b>

**Warnungen**

**Windrichtung:**  Grad

**Luftdruck:**  hPa

**Luftfeuchtigkeit:**  %

**Windgeschwindigkeit:**  km/h

**Temperatur:**  °C

**Beenden**





### 1) Basis:

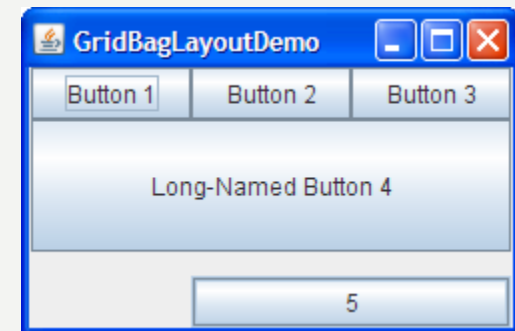
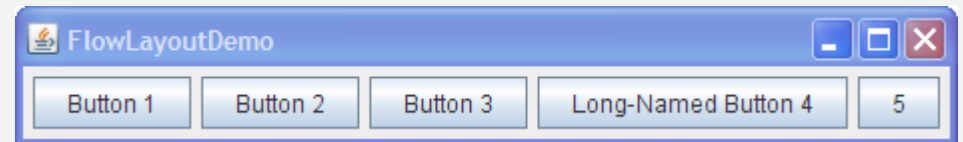
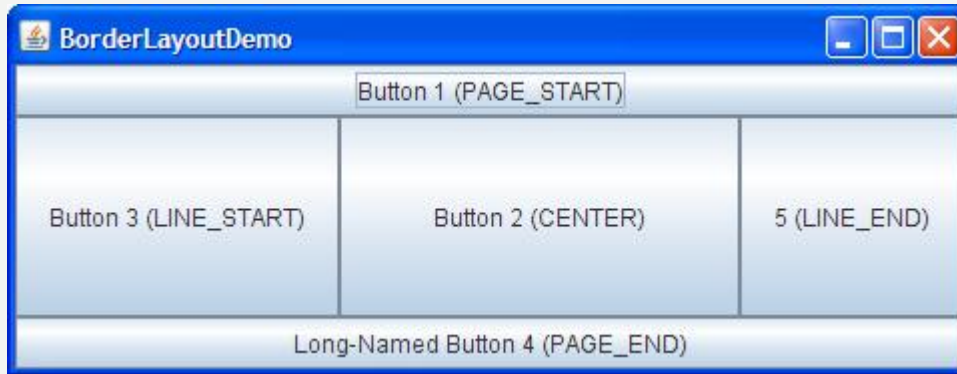
- i.d.R. JFrame, JDialog, oder JApplet
- RootPane benötigt **LayoutManager**

### 2) Aufbau der Grundstruktur:

- i.d.R. JPanel
- Jeweils eigene **LayoutManager**

### 3) Eigentliche Komponenten

- JButton, JTextField, ...



- Siehe [Java-Website](#)
- I/O-Relevante GUI-Elemente werden Felder
  - Buttons, Textfelder, ...
- Andere GUI-Elemente werden on-the-fly erzeugt
  - Panels, Layouts, Labels, ...

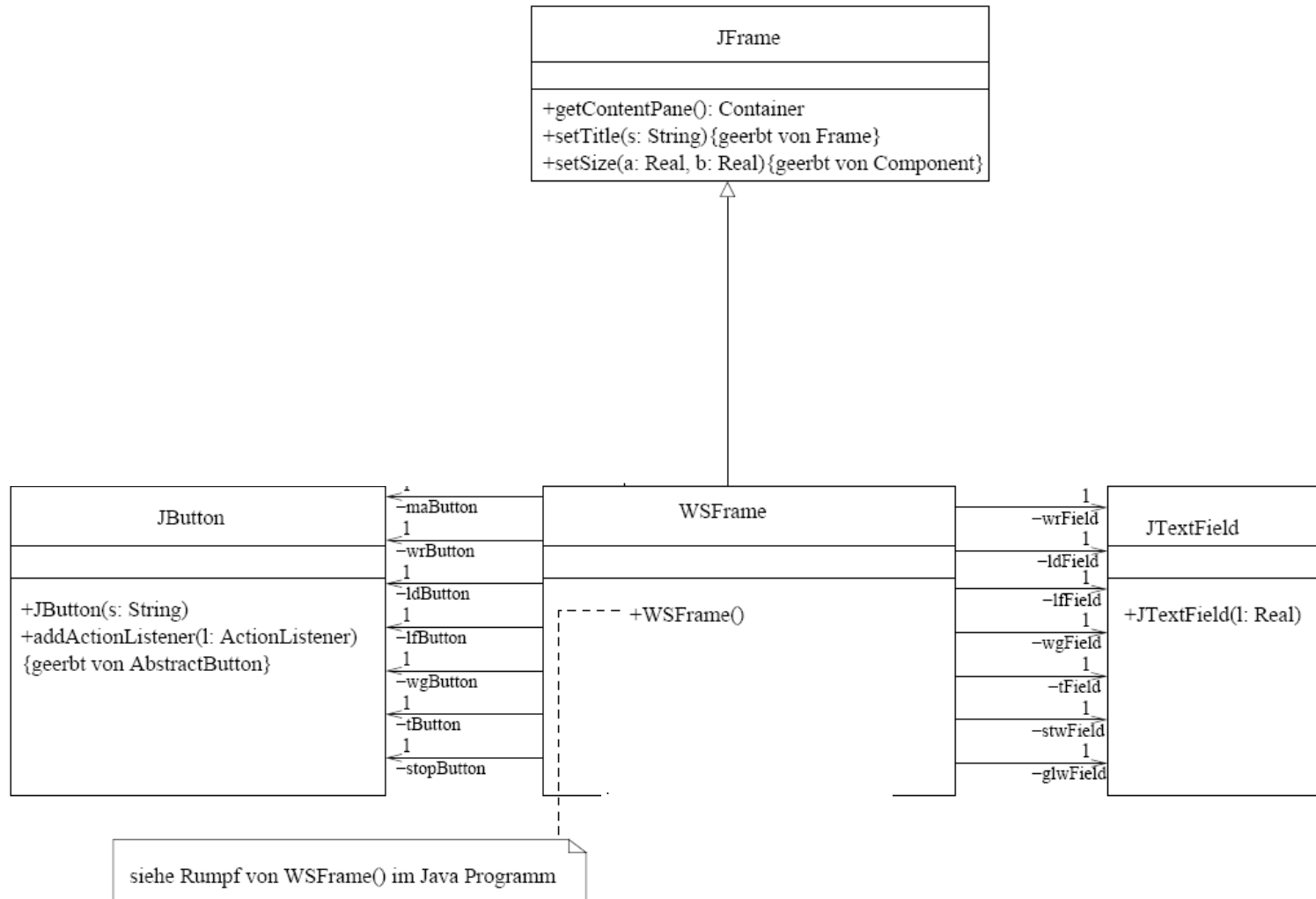
**Wetterstation**

<b>Alles messen</b>	<b>Windrichtung messen</b>
<b>Luftdruck messen</b>	<b>Luftfeuchtigkeit messen</b>
<b>Windgeschwindigkeit messen</b>	<b>Temperatur messen</b>

**Warnungen**

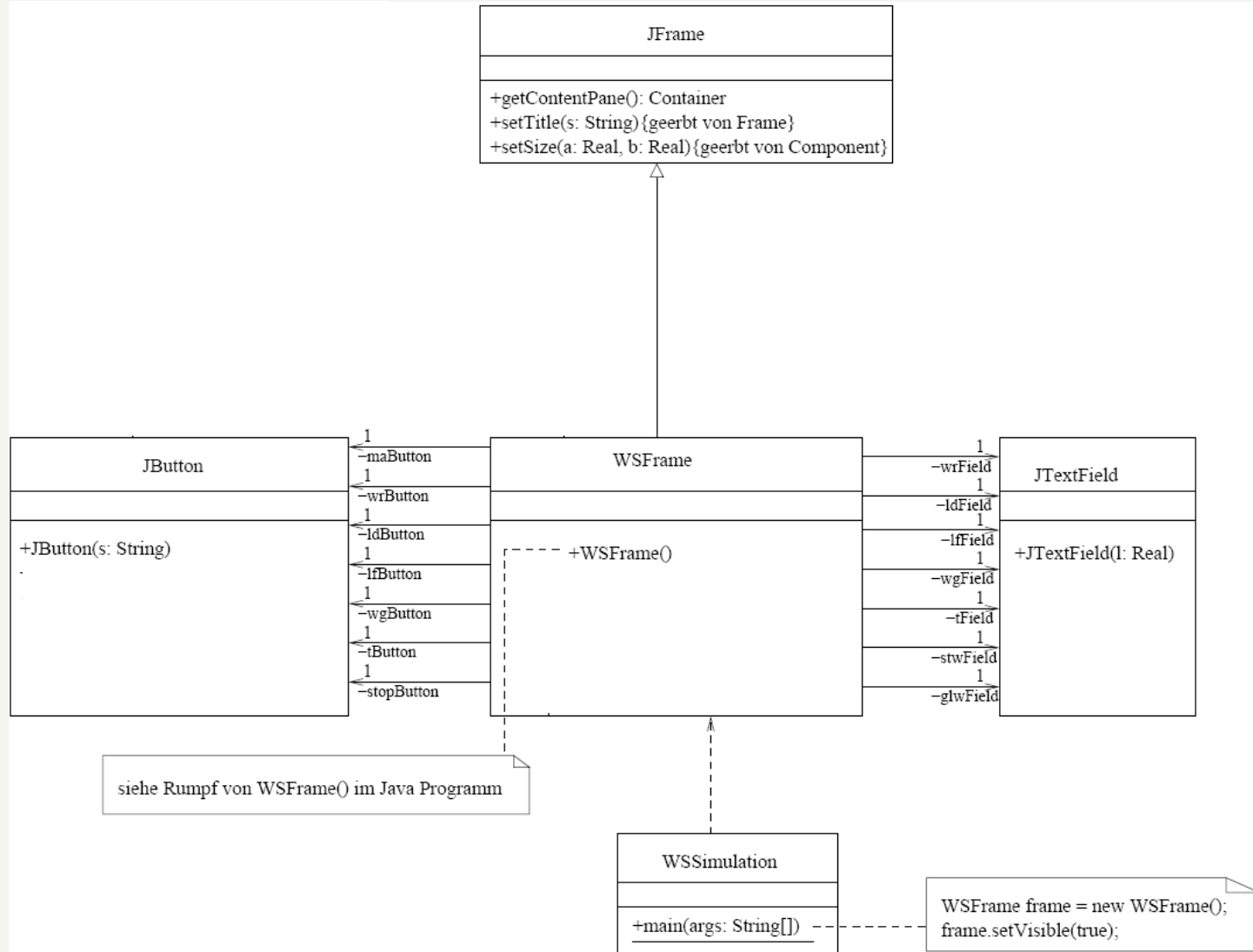
<b>Windrichtung:</b>	<input type="text"/>	<b>Grad</b>
<b>Luftdruck:</b>	<input type="text"/>	<b>hPa</b>
<b>Luftfeuchtigkeit:</b>	<input type="text"/>	<b>%</b>
<b>Windgeschwindigkeit:</b>	<input type="text"/>	<b>km/h</b>
<b>Temperatur:</b>	<input type="text"/>	<b>°C</b>

**Beenden**



Hands-On

# IMPLEMENTIERUNG





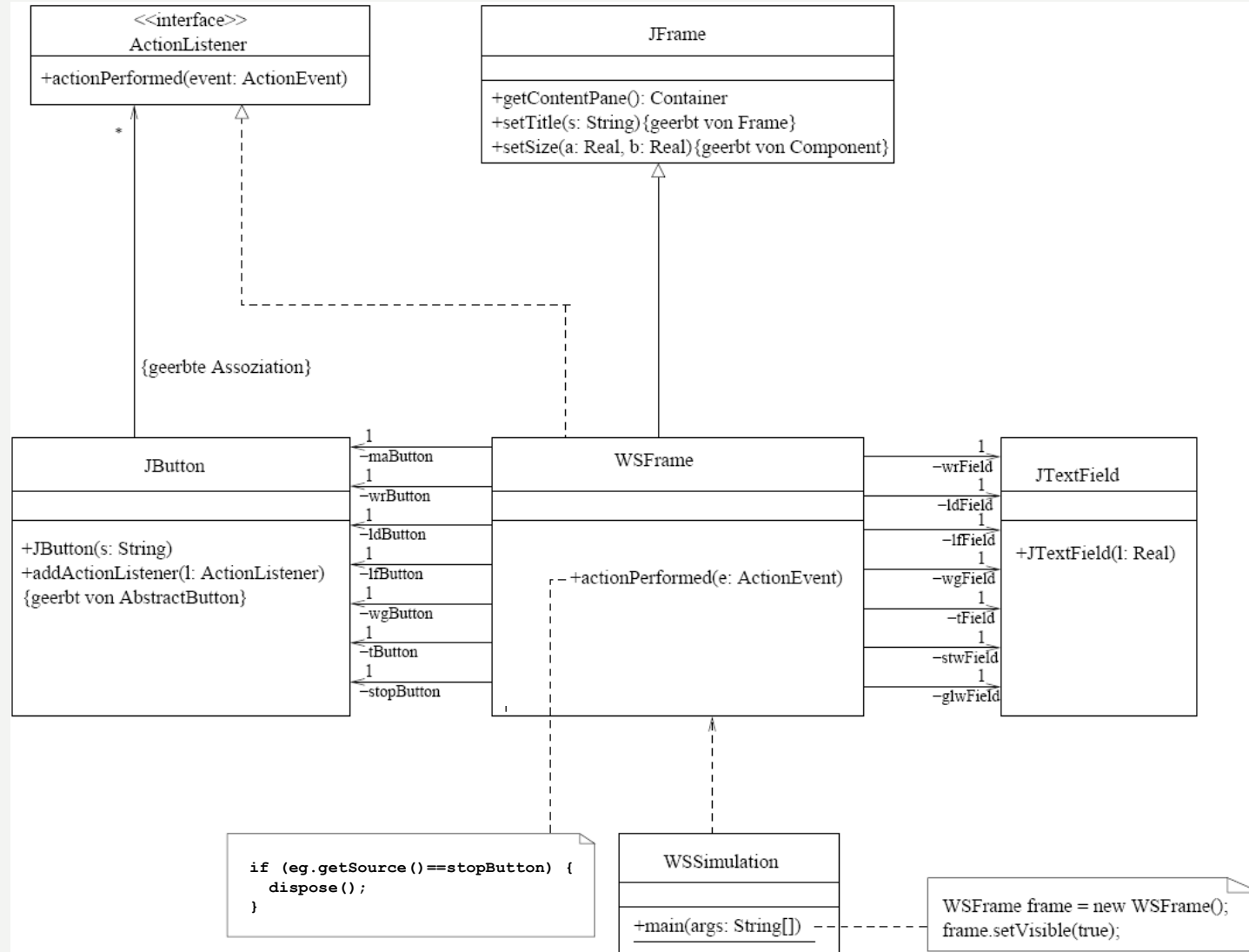
- **Vorgehensweise:**

1. Event Handler schreiben

- Neue (ggf. anonyme) oder bestehende Klasse entsprechendes Interface implementieren lassen
- Methoden in dieser Klasse ausimplementieren

2. Event Handler am Quellobjekt registrieren





Hands-On

# IMPLEMENTIERUNG



- Wir haben:
- Einen neuen JFrame entwickelt
- Eine Klasse zum Starten angelegt
- Im JFrame:
- Constructor handelt Erzeugung der GUI
- Strukturierung durch JPanels mit entsprechenden LayoutManagern
- Felder für relevante UI-Elemente; lokale Variablen für irrelevante
- Standard-Event Handling

GUI / Übung 10

**ENDE**