

# Übersicht

- **Einführung: Vom Maschinencode zum Objekt**
  - *Maschinenorientiert*
  - *Problemorientiert*
  - *Objektorientiert*
- **Die zentralen Metaphern: Gebäude und Genealogie**
  - *Klassenhierarchien stellen Verwandtschaft dar*
  - *Objektkomposition setzt Produkte aus Bauteilen zusammen*
  - *Viele Metaphern aus der Architektur*
- **Richtig entwerfen: Rhetorik für Programmierer**
  - *Ein Fenster ist ein Rechteck oder ein Fenster hat ein Rechteck?*
  - *Fliegengewichte*

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides

# Entwurfsmuster

Elemente wiederverwendbarer objektorientierter Software

Deutsche Übersetzung von Dirk Riehle

Addison–Wesley

Bonn 1996

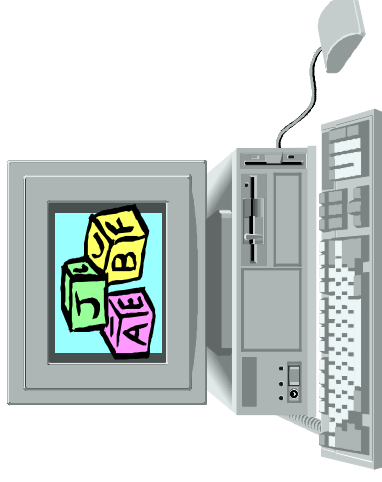
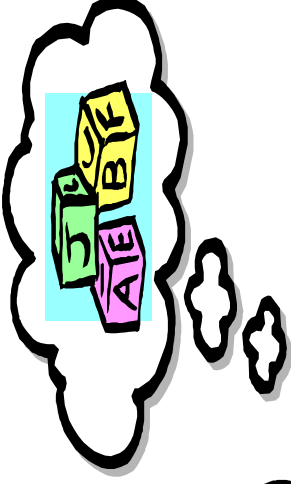
# Programmiersprachen

```
main(char** argv)
{
  cout << „Hallo Welt“ << '\n'
}
```



Compiler

```
0000:0000 00 0A 55 67 AD
0000:0010 01 0D FF FF 00
0000:0020 00 00 01 03 00
```



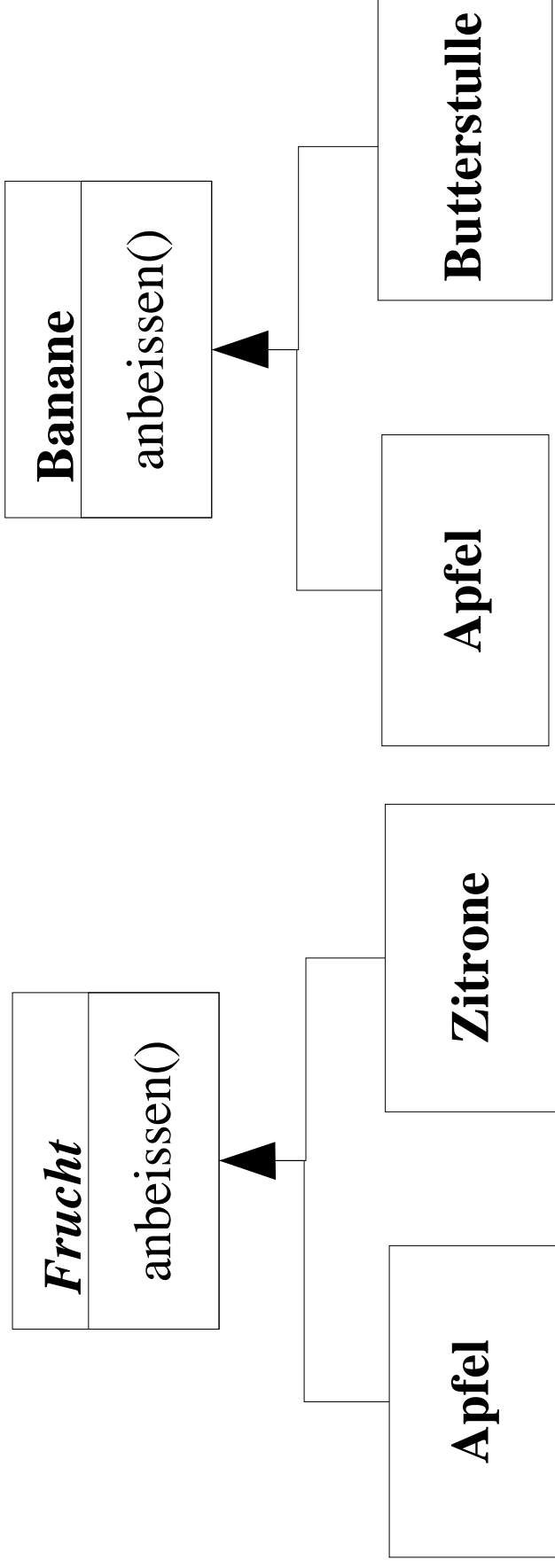
# Objekte und Klassen

- Objekte kapseln Eigenschaften und Methoden
- Objekte sind Exemplare (Instanzen) von Klassen
- Klassen stehen in einer hierarchischen Beziehung zueinander

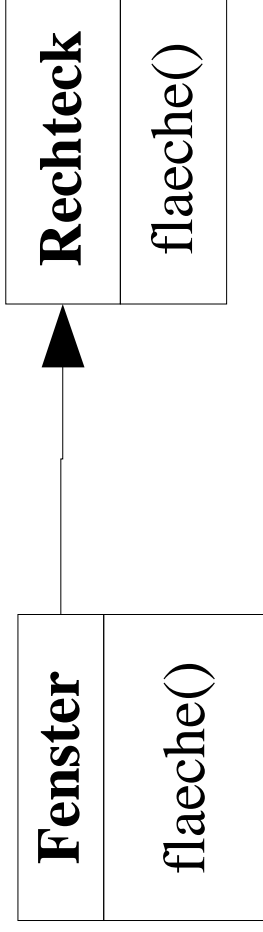
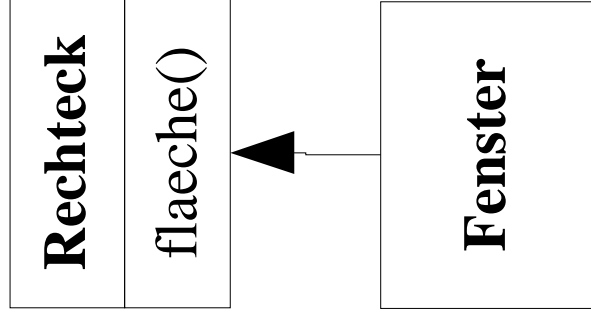
# Zwei Strukturen

- Klassenhierarchie
  - Statische Struktur
  - Zur Compilierzeit festgelegt
  - Vererbung
  - Aus Programmcode leicht ersichtlich
- Objektstruktur
  - Dynamische Struktur
  - Zur Laufzeit festgelegt
  - Komposition
  - Aus Programmcode schwer ersichtlich

# Gute und schlechte Klassenhierarchie



# Vererbung und Komposition

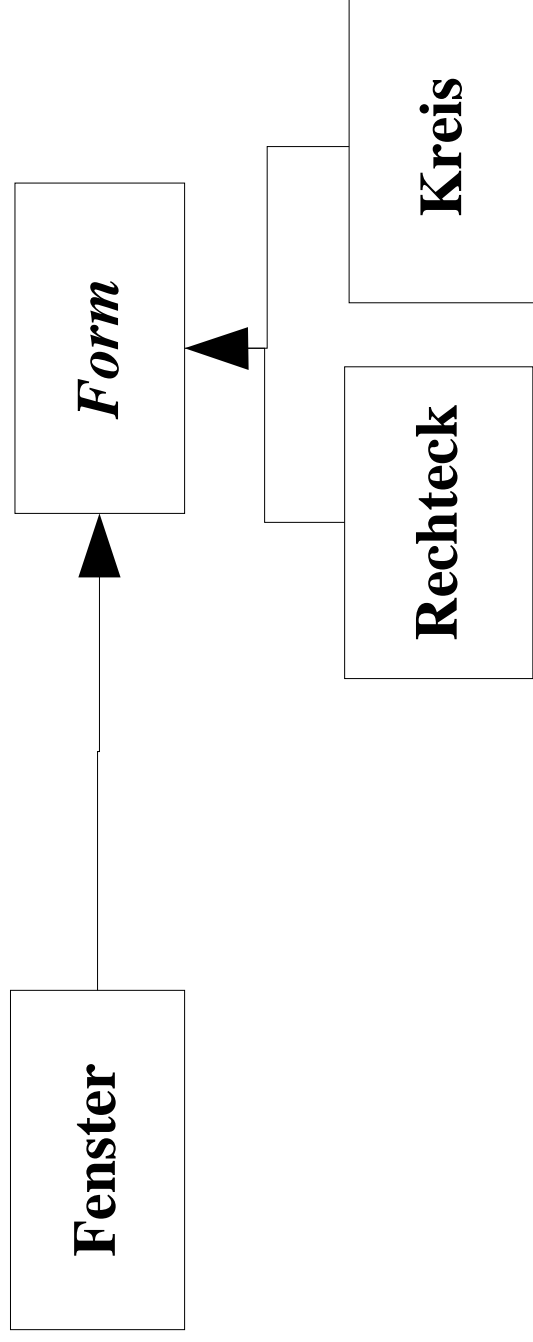


# Glaubensgrundsätze

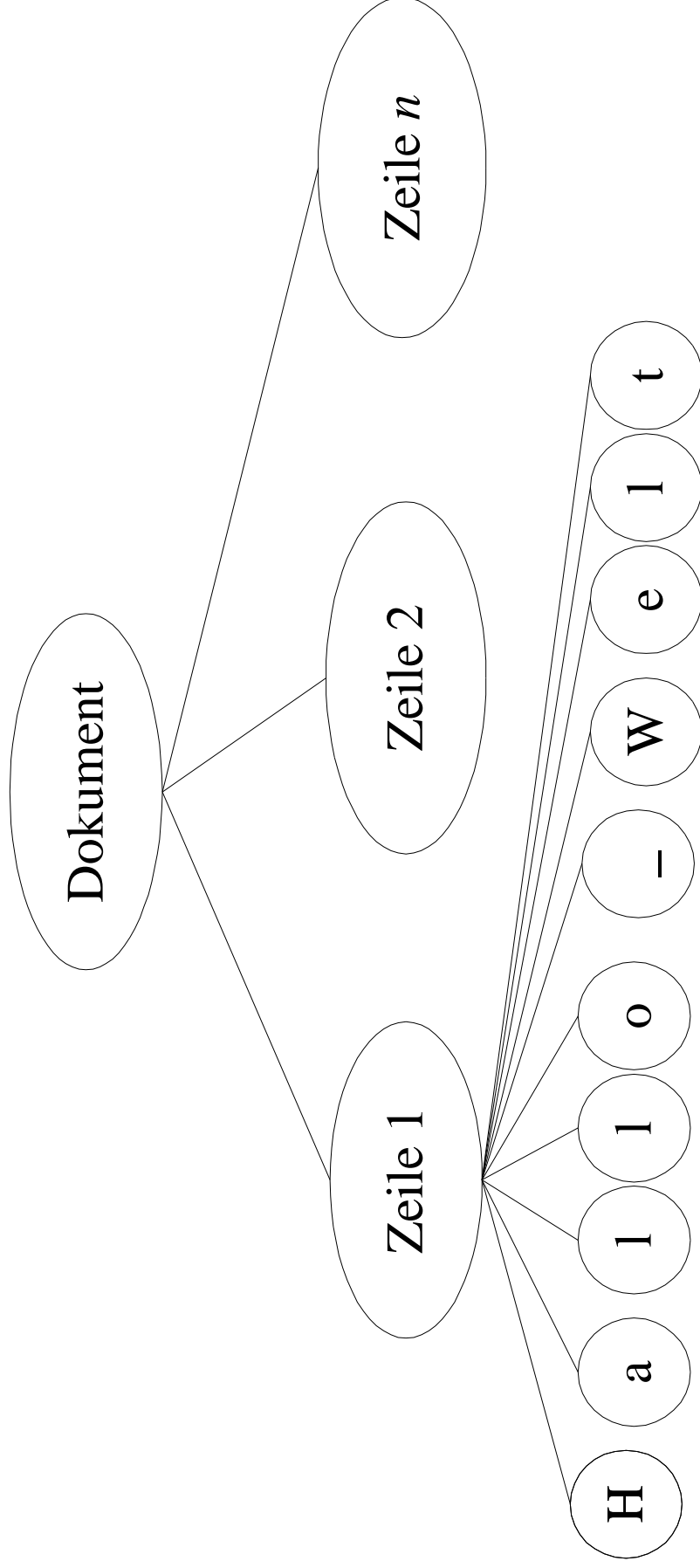
- Programmieren auf einen Schnittstelle hin, nicht auf eine Implementierung.
- Ziehen Objektkomposition der Klassenvererbung vor.



# freie Fensterformen



# Objektorientiertes Dokumentmodell



# Fliegengewichte

