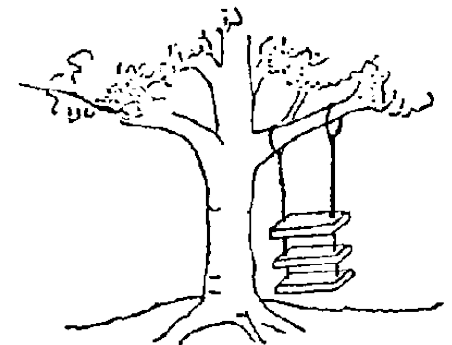


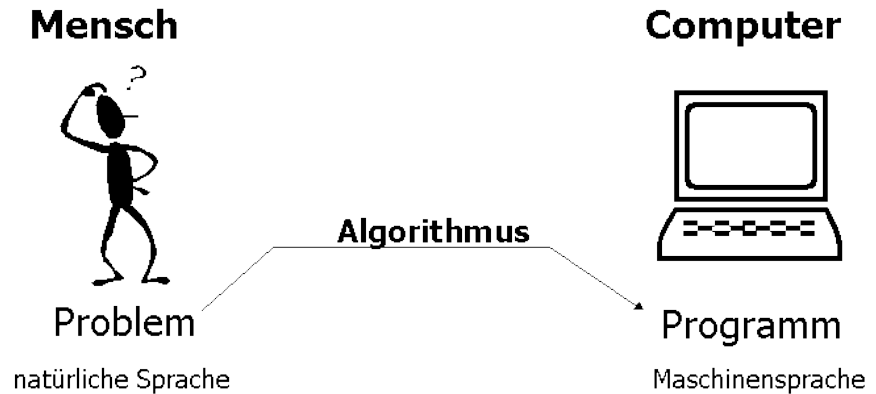
Grafische Darstellung von Quellcode

Software Entwicklung



Übersicht

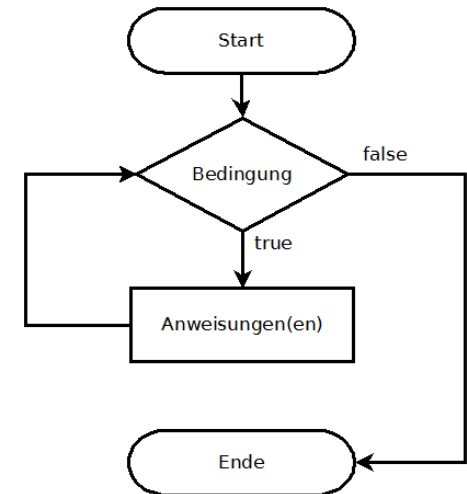
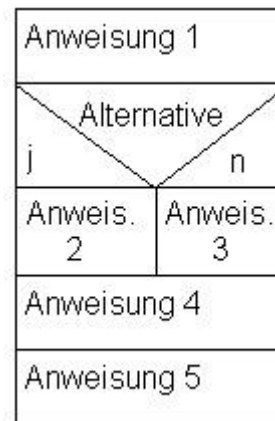
- Algorithms
- Diagramme
- Pseudocode



- UML (Unified Modelling Language)
- Use Case Diagramme
- UML Klassendiagramme im Detail

Algorithmen

- Algorithmen sind logische Abläufe, diese können programmiert werden oder grafisch dargestellt werden.
 - Struktogramme
 - Ablaufdiagramme
 - Pseudocode
- Programmiersprache C#



Darstellung von Algorithmen

als Pseudocode oder grafisch mit
Struktogramm & Ablaufdiagramm

Pseudocode

Beispiel in Pseudocode:

Algorithm arrayMax(A, n) :

Input: Ein Array A , der n Integerwerte enthält

Output: Das maximale Element in A

currentMax = $A[0]$

for $i = 1$ **to** $n - 1$ **do**

if currentMax < $A[i]$ **then**

 currentMax = $A[i]$

return currentMax

Struktogramm

- Eingabe des Namens
- Auswertung ob m/w
- Demnach „Guten Tag Frau X“ oder „Guten Tag Herr X“ ausgeben

Begrüßung

Der Name ist einzugeben

Das Geschlecht ist einzugeben (m/w)

Ist die Person weiblich?

ja

nein

Begrüßung: "Guten Tag Frau X"

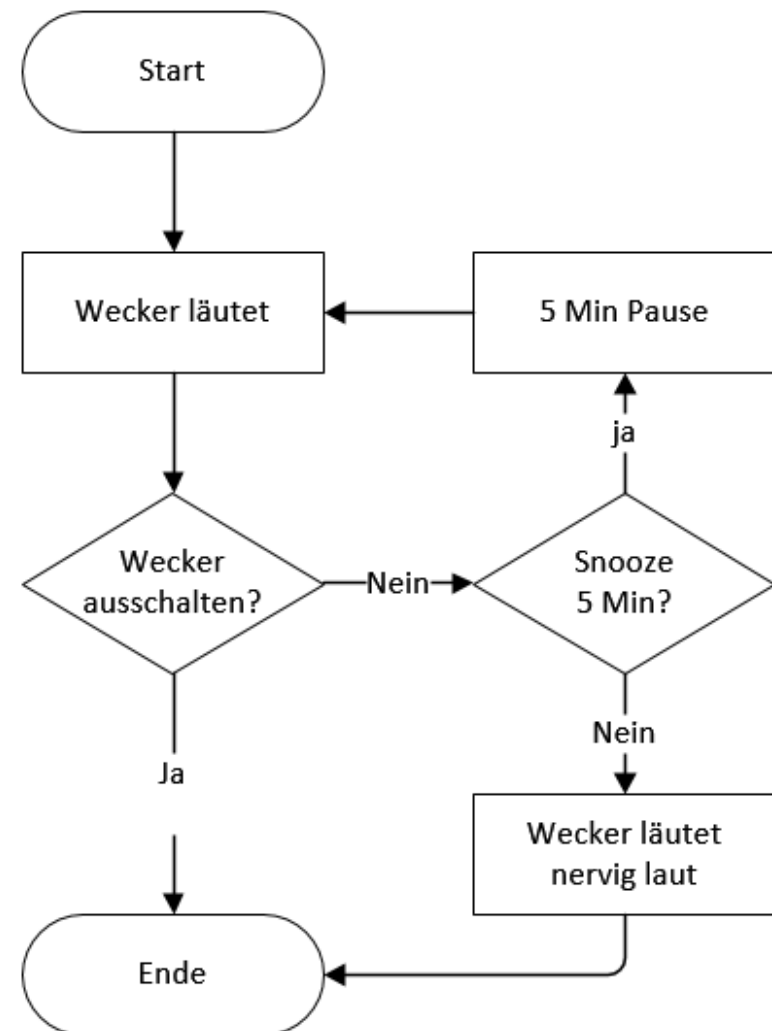
X = Name der Frau

Begrüßung: "Guten Tag Herr X"

X = Name des Herrn

Ablaufdiagramm: Morgens aufstehen

- Wecker läutet
- Ausschalten & aufstehen
- Snooze Butten: ja/nein

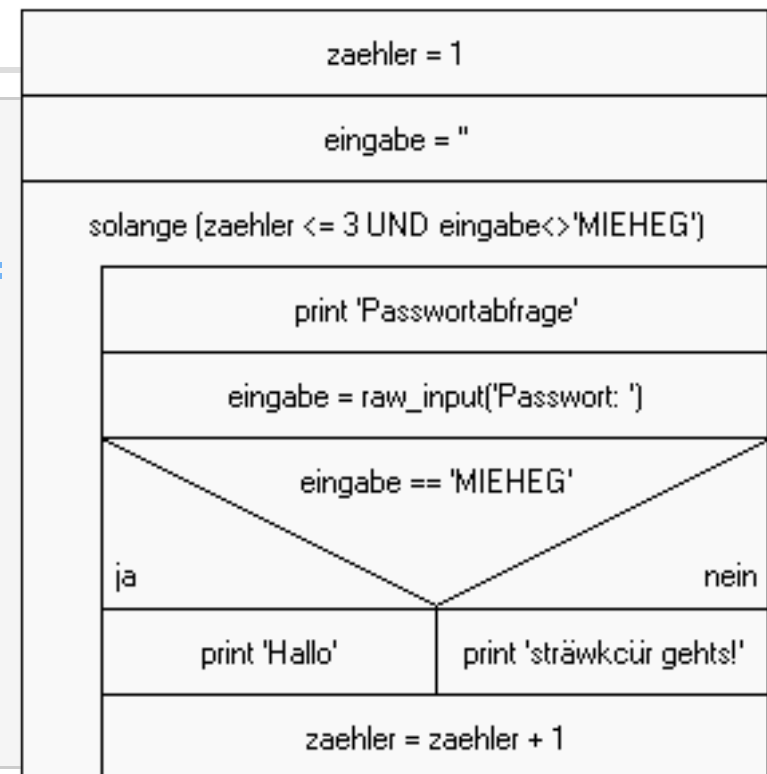


Aufgabe: Passwortabfrage

- Erstelle ein Ablaufdiagramm für folgende Passwortabfrage:

Python-Programm

```
zaehler = 1
eingabe = ''
while (zaehler <= 3 and eingabe <> 'MIEHEG'):
    # Einrückungen beachten!
    print 'Passwortabfrage'
    eingabe = raw_input('Passwort: ')
    if (eingabe == 'MIEHEG'):
        print 'Hallo'
    else:
        print 'sträwkcür gehts!'
    zaehler = zaehler + 1
```



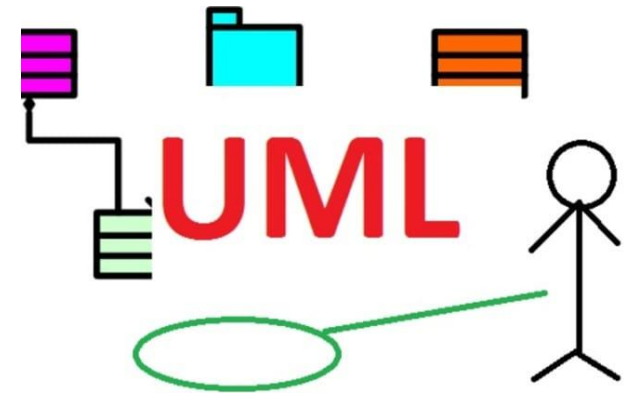


UML

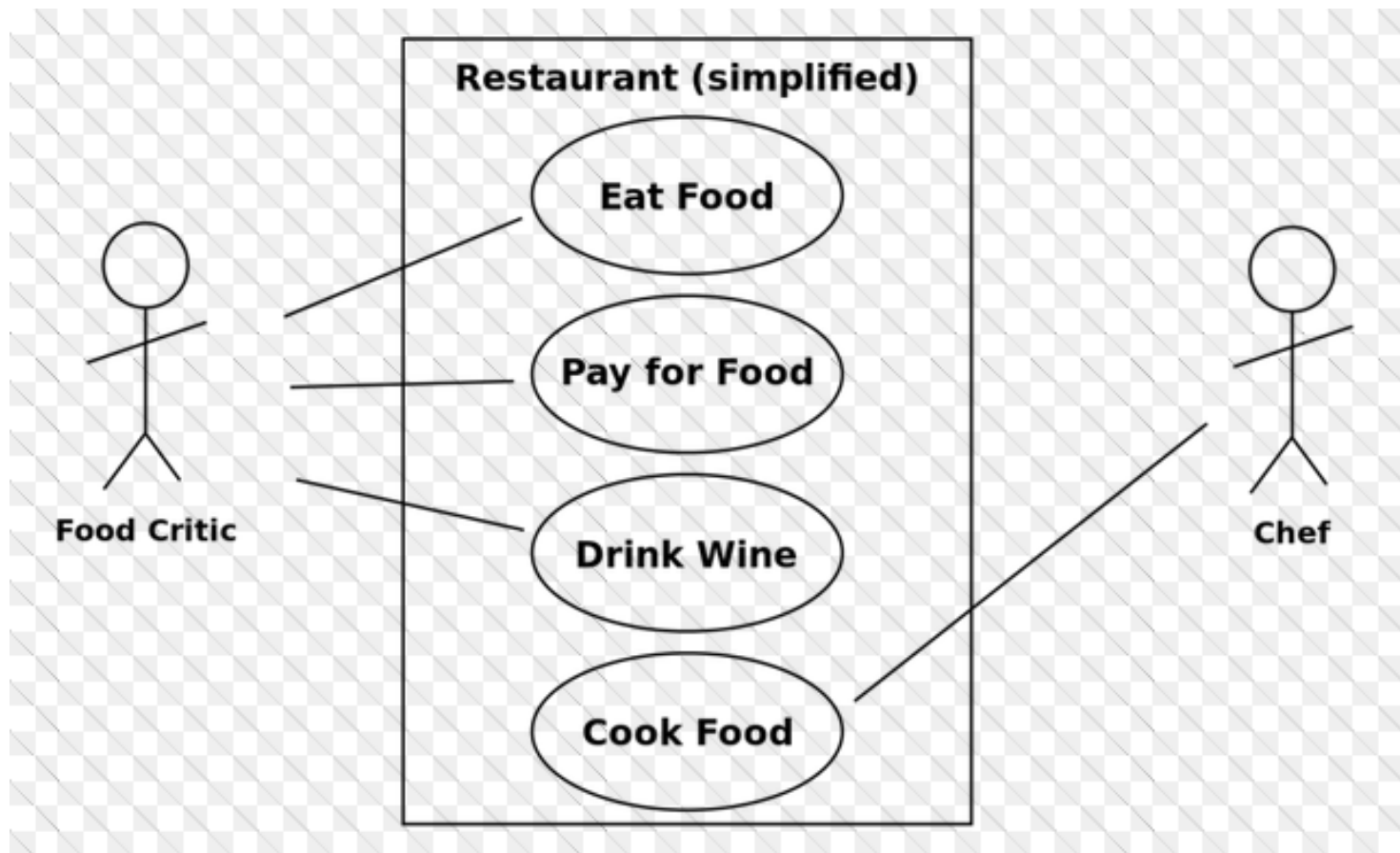
Unified Modelling Language

Arten von UML Diagrammen:

- UML Use Case Diagram
- UML Sequence Diagram
- UML Component Diagram
- UML Class Diagram
- UML Activity Diagram
- UML Collaboration Diagram
- UML Deployment Diagram
- UML Statechart Diagram
- UML Package Diagram

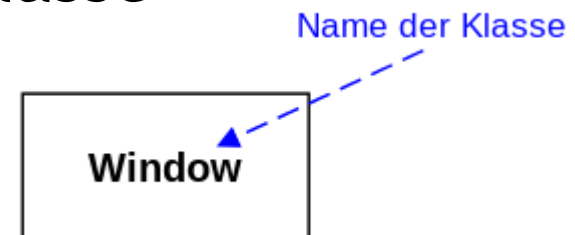


UML Usecase

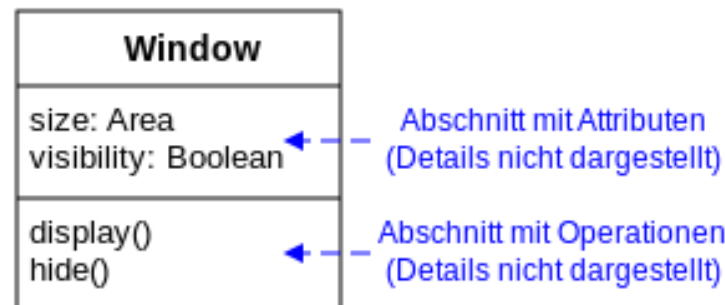


UML Klassendiagramm

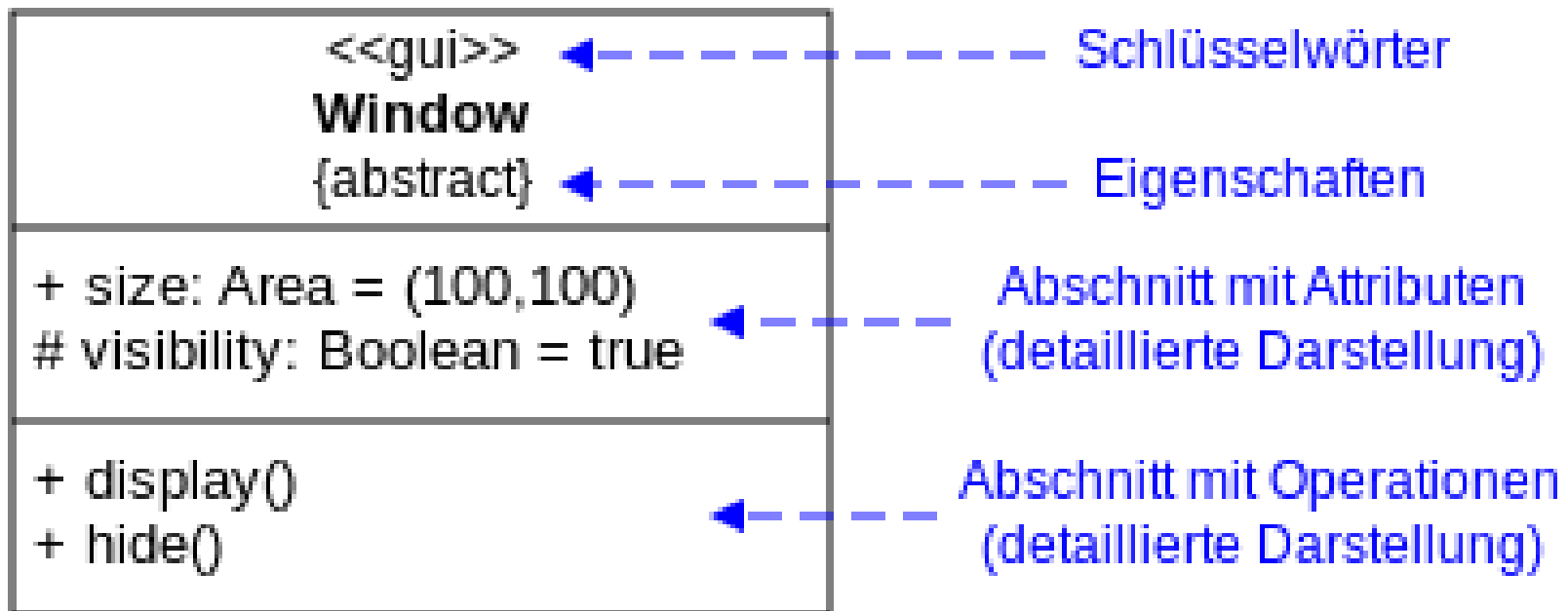
- Einfache Darstellung einer Klasse



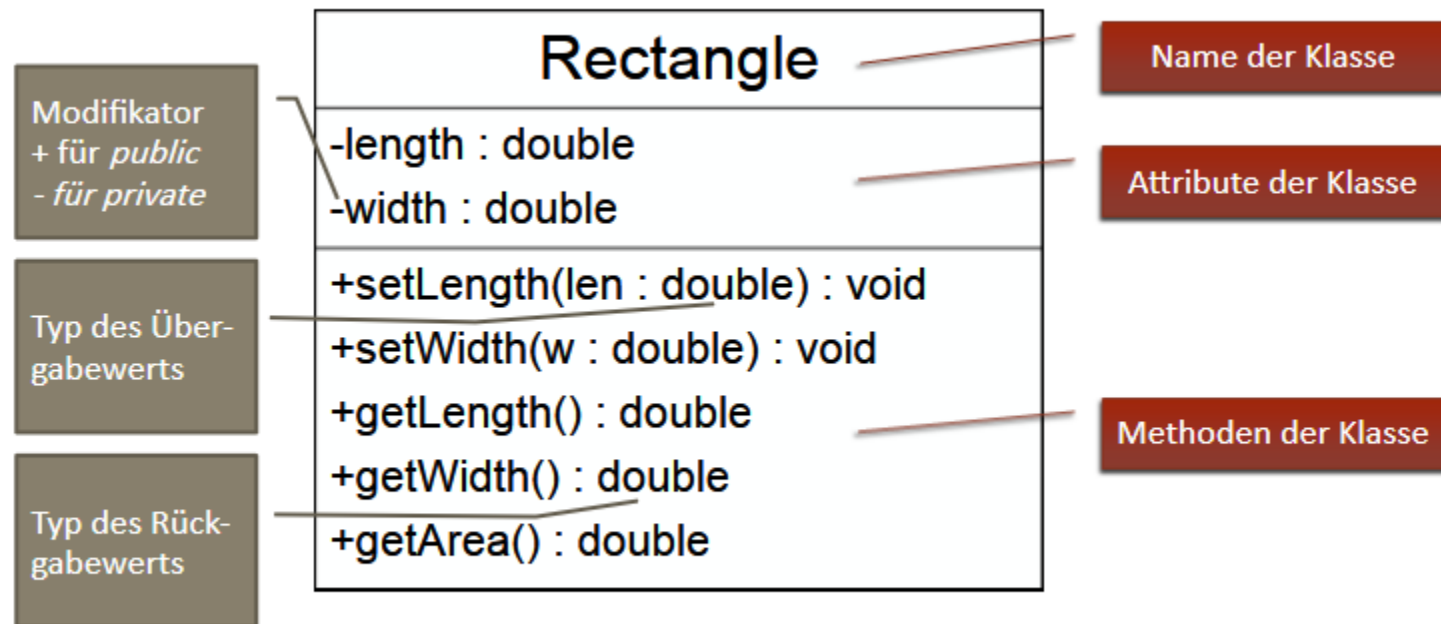
- Zusätzliche Darstellung mit Attributen und Methoden:



Detaillierte Darstellung von Window

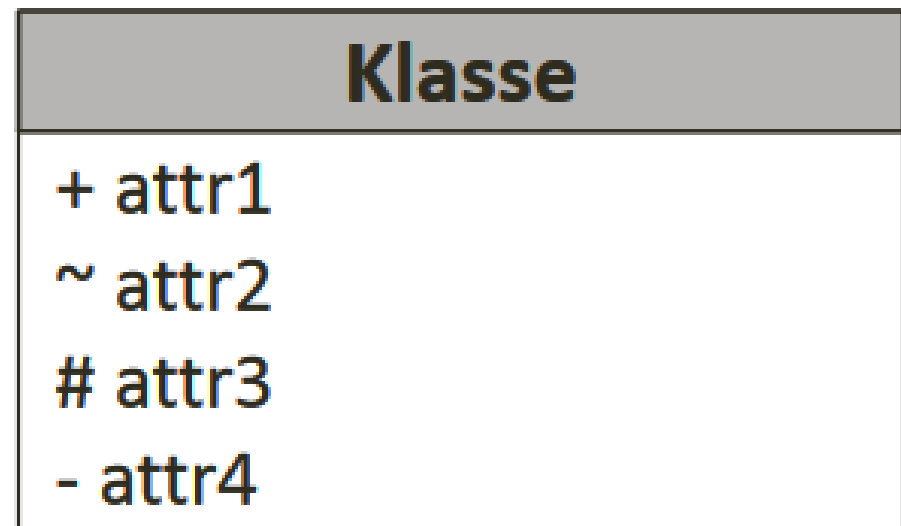


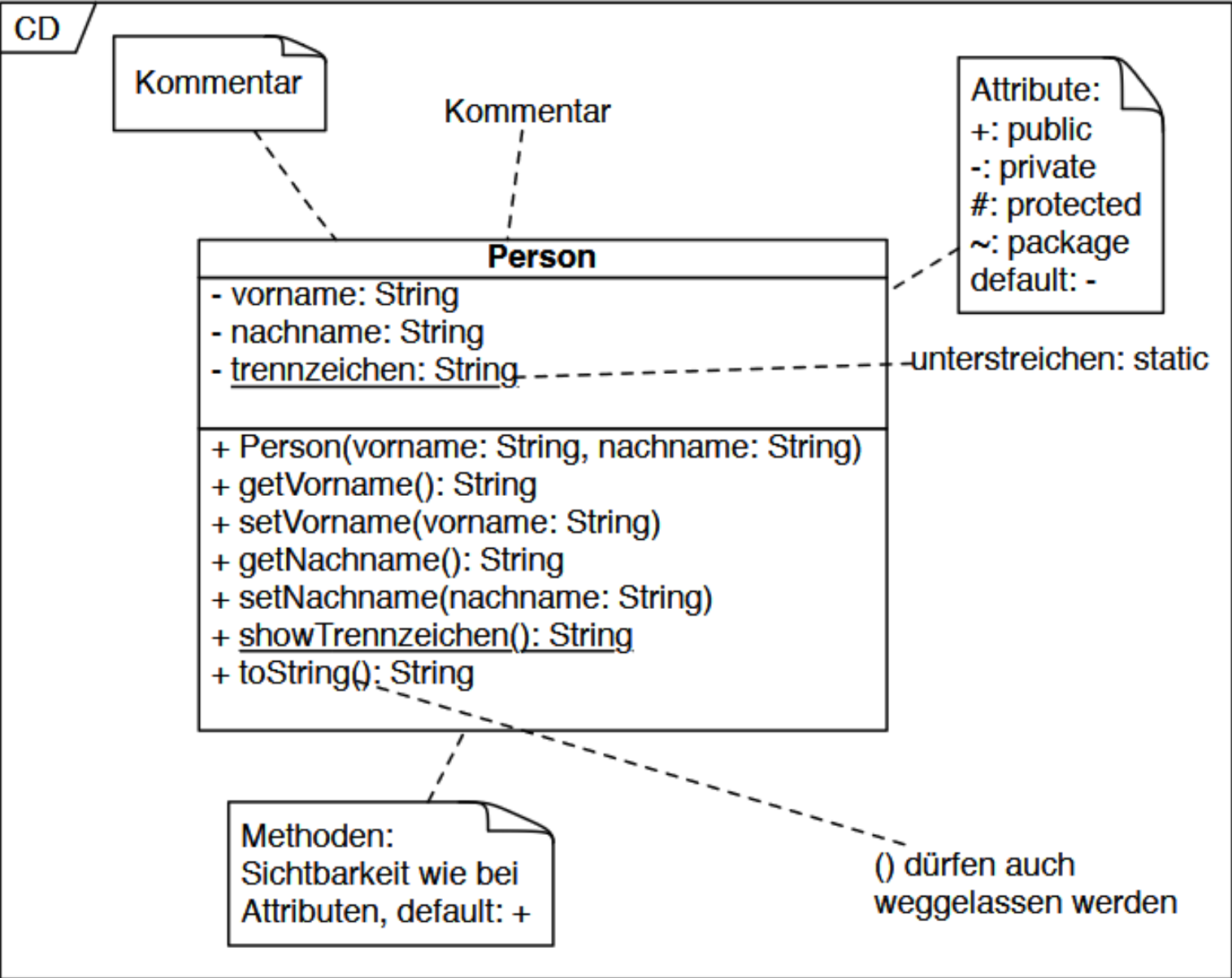
Detaillierte Darstellung einer Klasse



Sichtbarkeit im Klassendiagramm

- + public
- ~ package
- # protected
- - private

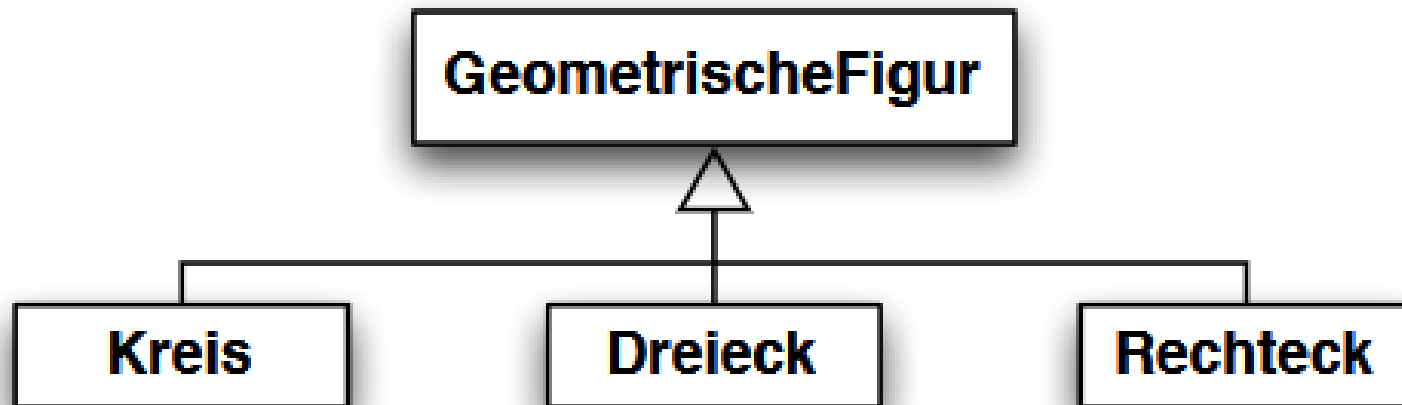




Person

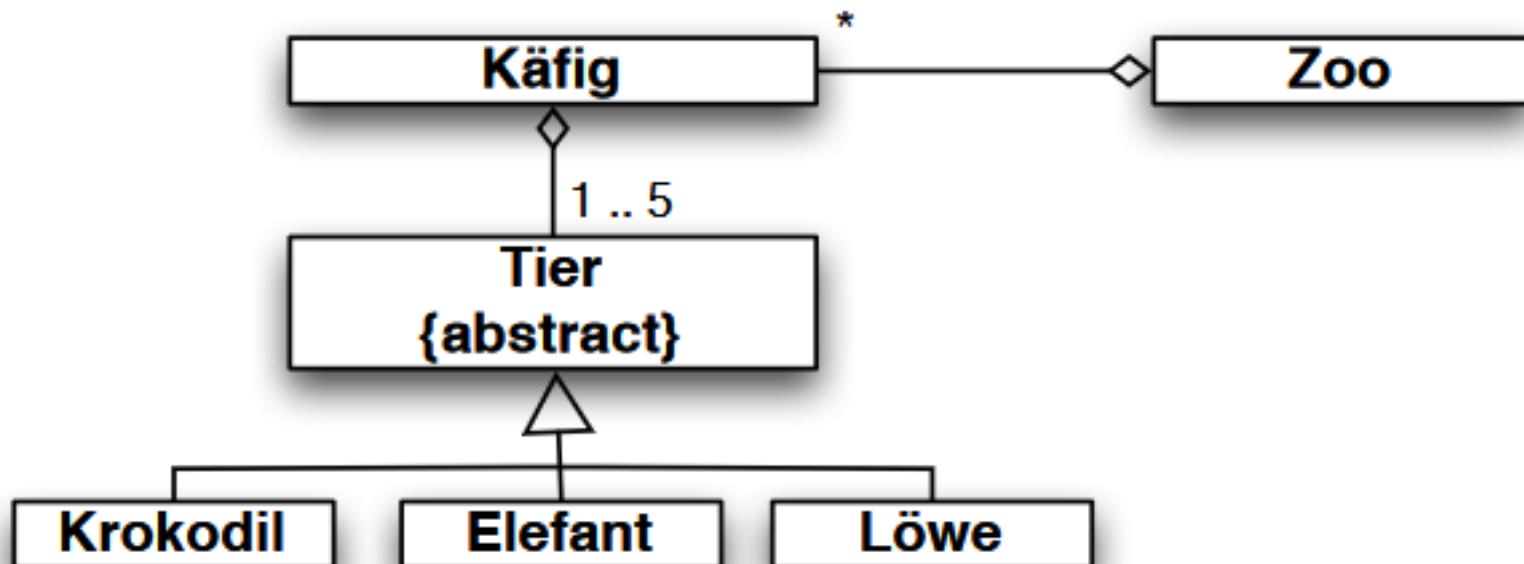
Vererbung

- Vererbungshierarchie zwischen der
 - Klasse Geometrische Figur und den
 - Subklassen Kreis, Dreieck und Rechteck



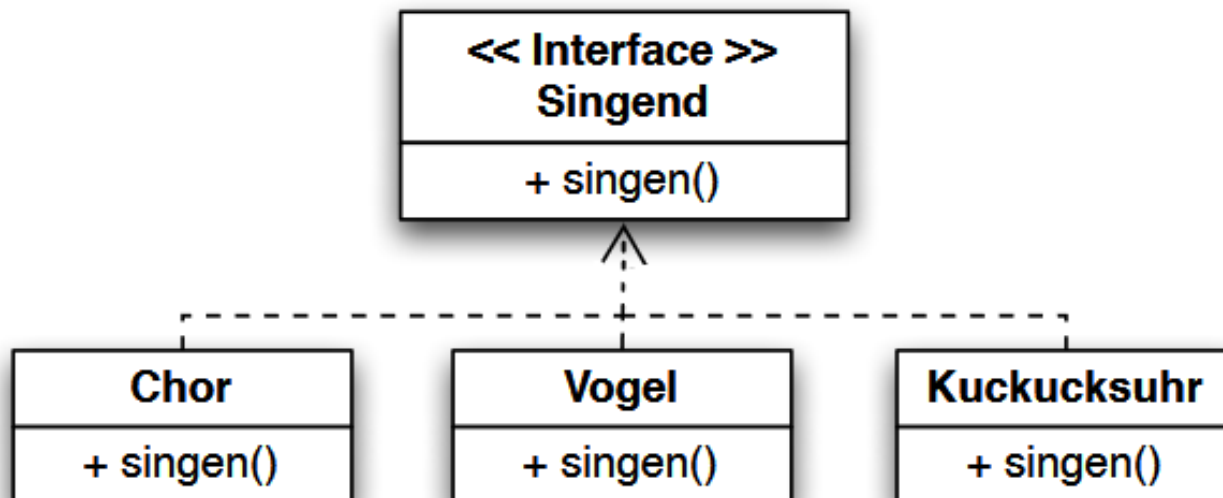
Klassendiagramm einer Zoo-Implementierung

- Ein Zoo besteht aus mehreren Käfigen.
- Käfige enthalten Tiere.
- Tiere (abstrakte Klasse) werden in Unterklassen genauer spezifiziert



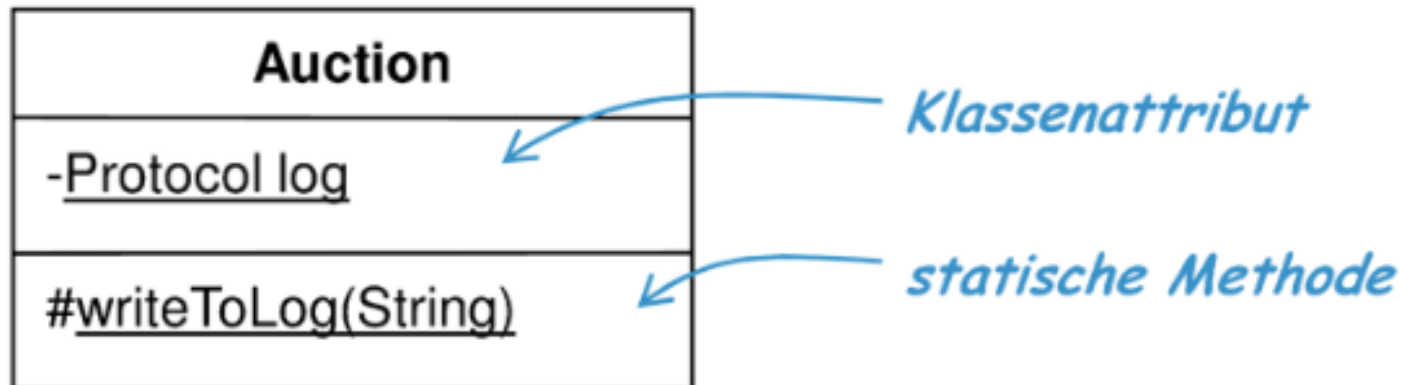
Interface implementieren

- drei Klassen, die als Methode „Singen“ verfügbar haben.
- Interface Singend enthält die Methode „Singen“, welche von Chor, Vogel & Kuckucksuhr implementiert werden

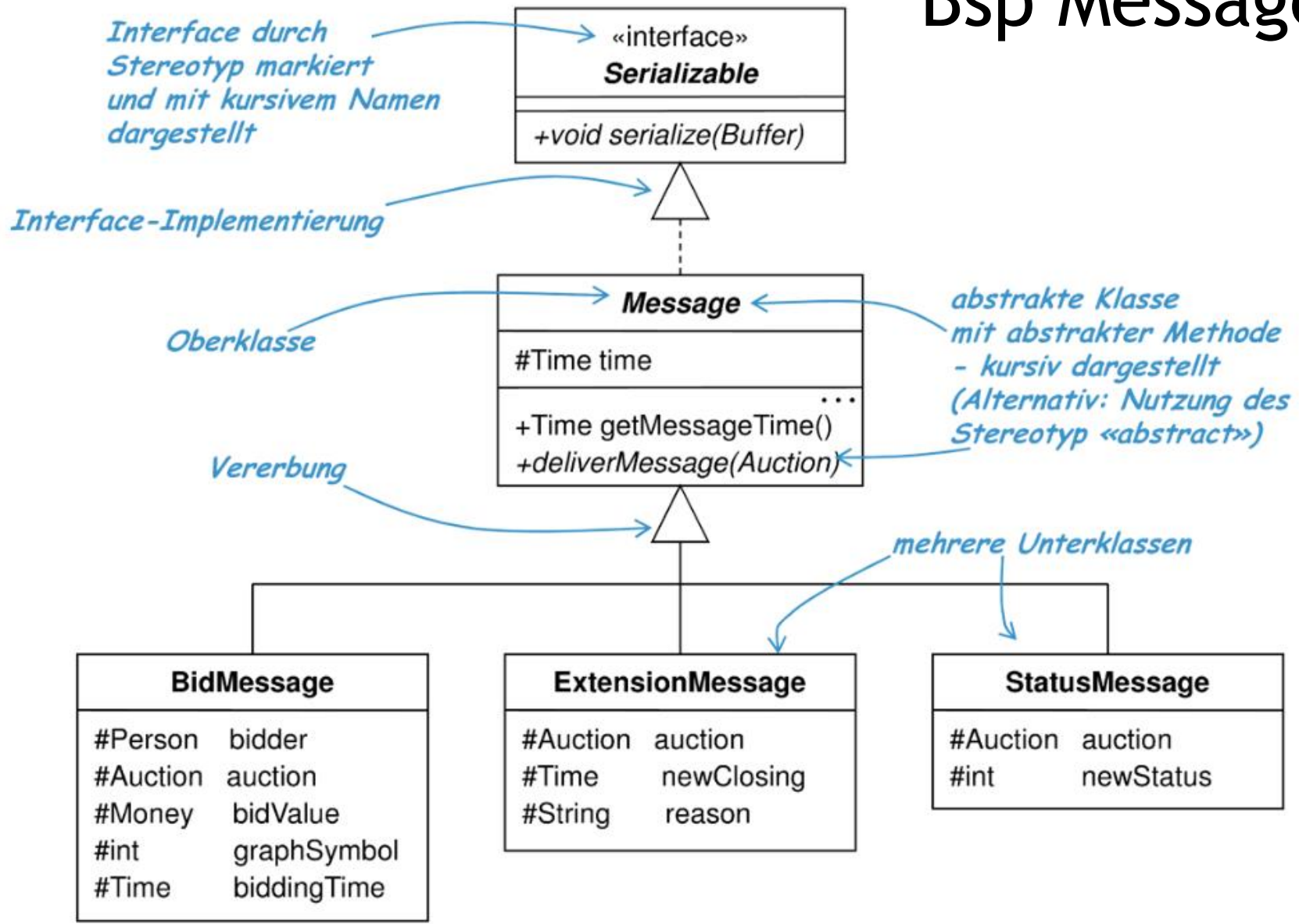


Klassen- vs Objektattribute

- Statische Attribute werden als Klassenattribute
- Statische Methoden als Klassenmethoden bezeichnet
- „Static Elemente“
werden in UML unterstrichen dargestellt



Bsp Message

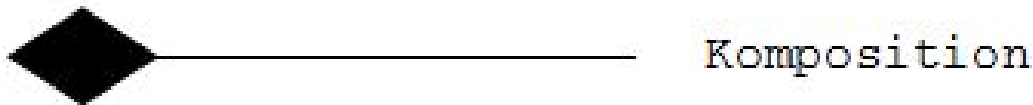


Beziehungen:

- Ist eine Beziehung -> Vererbung



- Hat eine Beziehung -> Aggregation oder Komposition

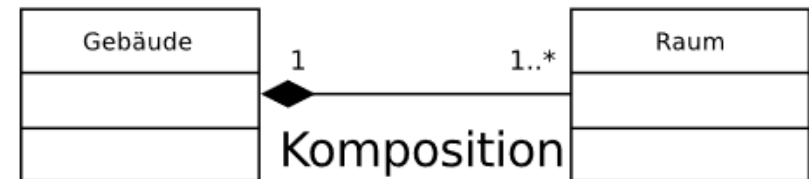


- Stellt Funktionalität bereit -> Implementiert







Aggregation oder Komposition

- Teil-Ganzes-Beziehung
- Aggregation
 - Vorlesung beinhaltet 3 oder mehr Studenten
- Komposition
 - Sonderfall der Aggregation:
Gebäude besteht aus Räumen
wobei, ein Raum nicht
ohne Gebäude sein kann



Aggregation oder Komposition

UML symbols

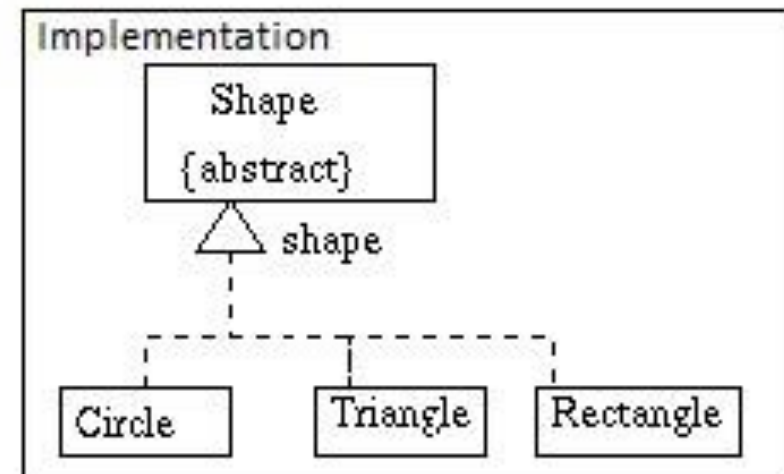
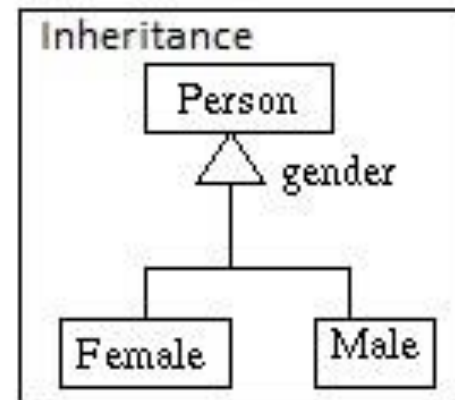
Association	Symbol
Composition	
Aggregation	
Inheritance	
Implementation	



Composition: every car has an engine.

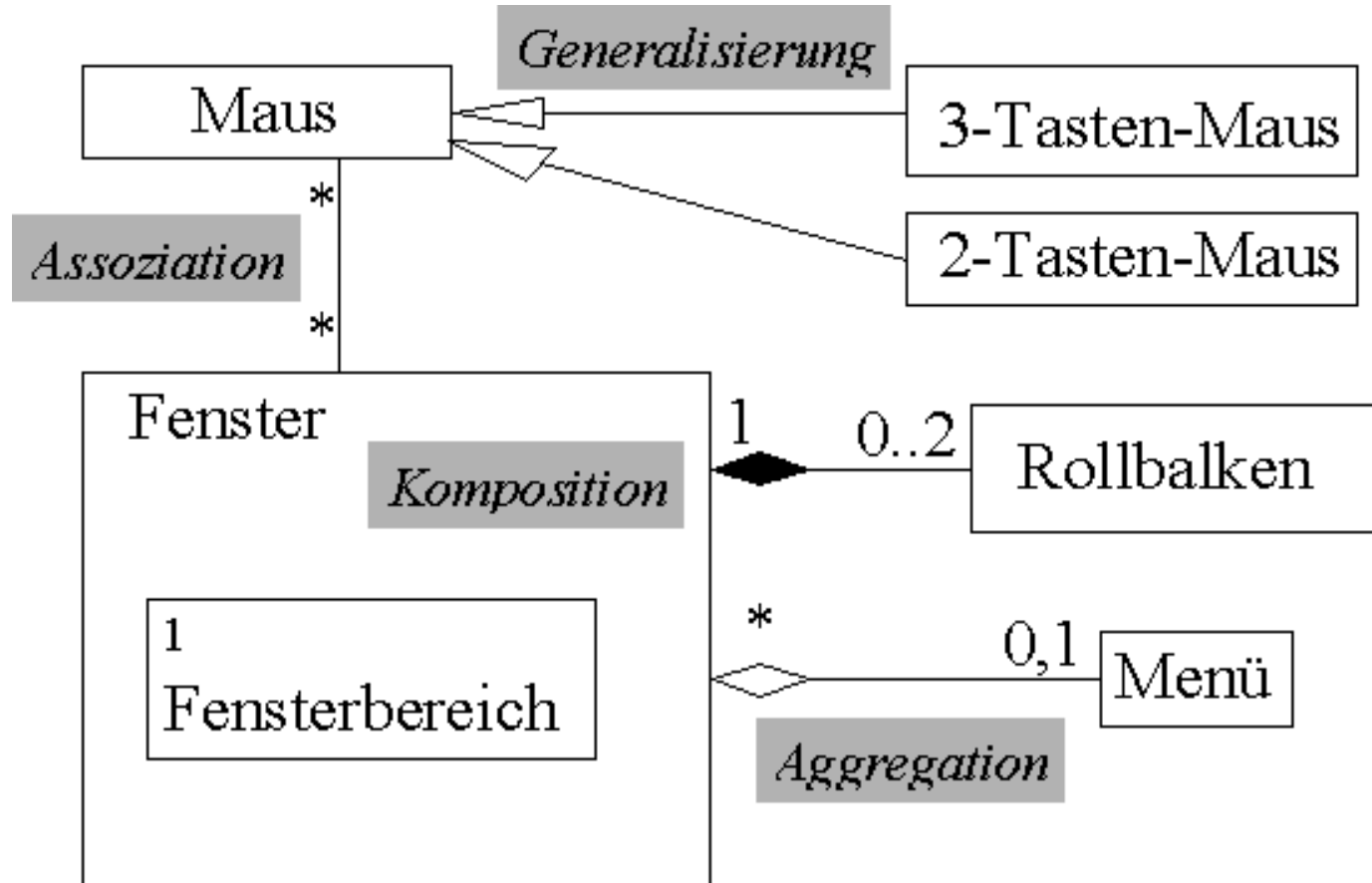


Aggregation: cars may have passengers, they come a

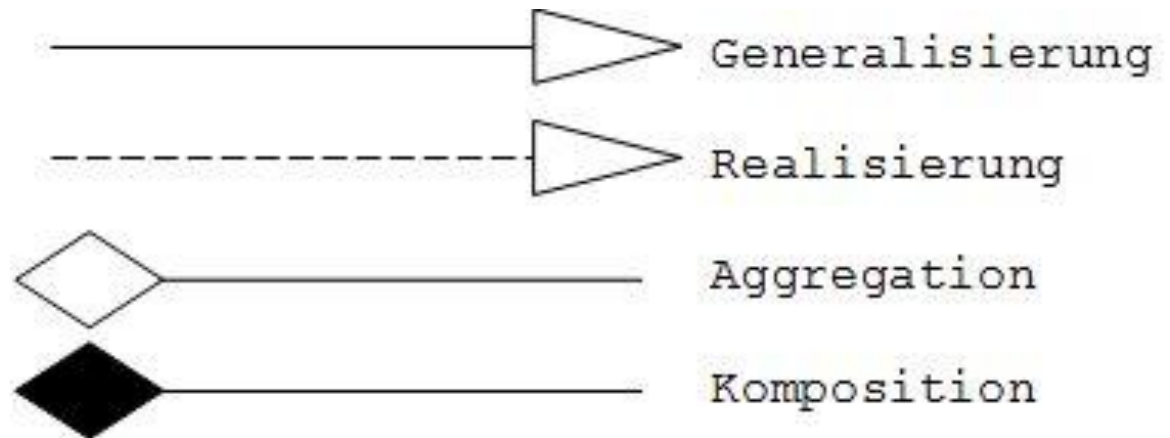
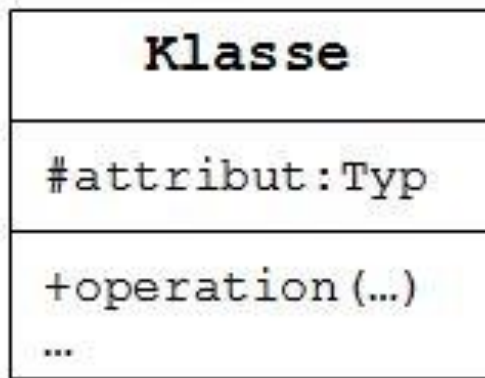


Assoziation

- Klassen die sich gegenseitig Nachrichten schicken können:



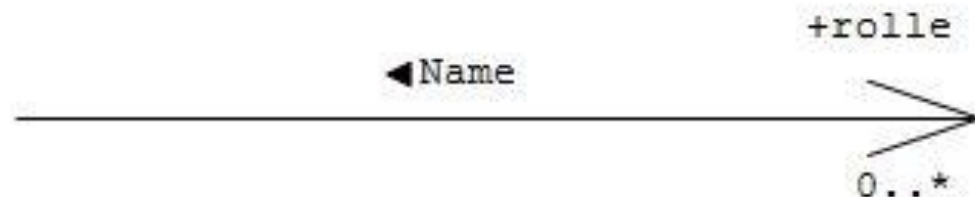
Beziehungen (Pfeile) in UML



Sichtbarkeit:

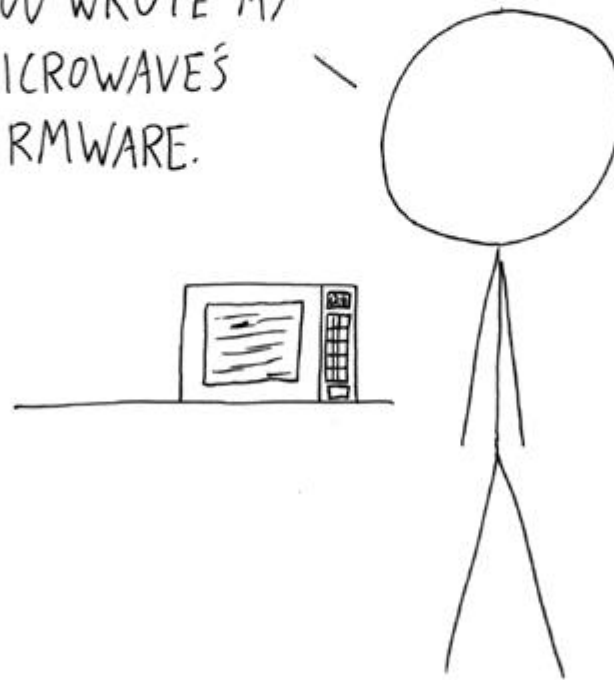
+ public
 # protected
 ~ package
 - privat

gerichtete Assoziation mit Beschriftung



A SOFTWARE ARCHITECT'S DREAM USER

THANK GOD YOU USED UML WHEN
YOU WROTE MY
MICROWAVE'S
FIRMWARE.



UML