

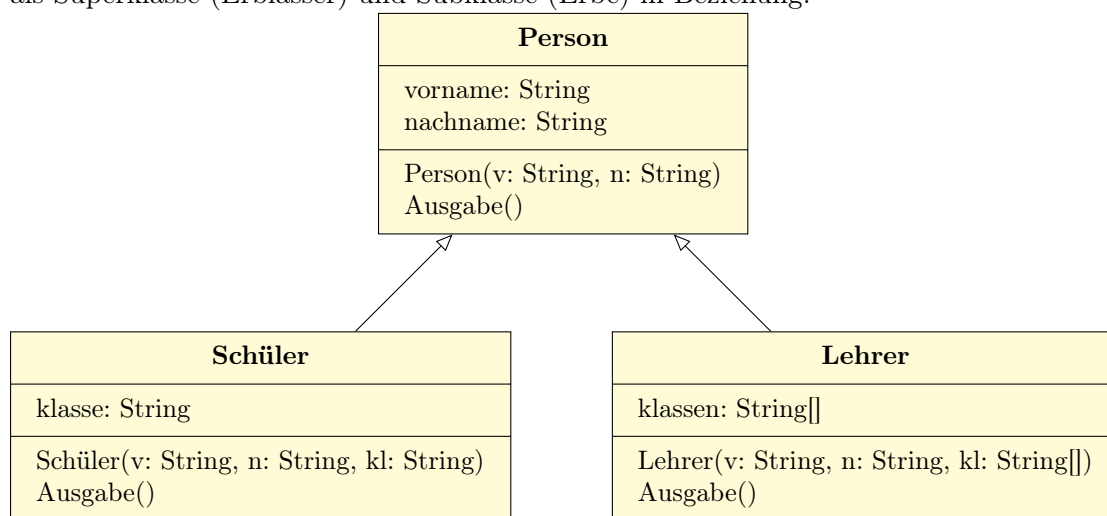
# UML, Vererbung und Aggregation

## 1. UML

Die *Unified Modeling Language* (vereinheitlichte Modellierungssprache), kurz *UML*, ist eine Möglichkeit, Strukturen und Zusammenhänge eines Programmes grafisch darzustellen.

## 2. Vererbung

Die sogenannte Vererbung ermöglicht es Informationen (Variablen) und Verhalten (Methoden / Operationen) weiterzugeben. Dies ist eine wesentliche Möglichkeit um Redundanz zu vermeiden. Die Erben fügen dann weitere Informationen und/oder Verhalten hinzu. Zwei Klassen stehen dabei zueinander als Superklasse (Erblasser) und Subklasse (Erbe) in Beziehung.<sup>1</sup>



Im Beispiel **spezialisieren** die Klassen **Schüler** und **Lehrer** die **Basisklasse** **Person** und fügen dieser weitere Attribute hinzu. Man spricht dabei auch von einer „ist-eine-Art-von“-Beziehung. Die Vererbung wird im UML-Diagramm mit einem leeren Pfeil dargestellt.

### 2.1 Umsetzung in Java

In Java werden Unterklassen mit dem `extends`-Befehl angelegt:

```
1 public class Schüler extends Person {
2     [...]
3 }
```

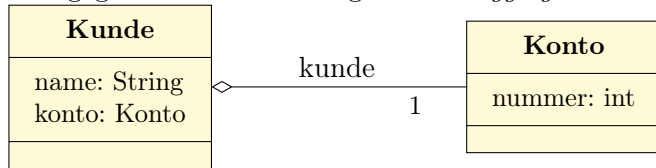
### 2.2 `protected` vs. `private`

Soll von einer Unterklasse auch auf Attribute der Hauptklasse zugegriffen werden, so dürfen diese nicht als `private` markiert sein. Stattdessen kann man hierzu das Schlüsselwort `protected` verwenden. Hierbei erhalten neben der eigentlichen Klasse (wie bei `private`) auch abgeleitete Unterklassen Zugriff.

<sup>1</sup>Quelle: [https://de.wikibooks.org/wiki/Java\\_Standard:\\_Vererbung](https://de.wikibooks.org/wiki/Java_Standard:_Vererbung)

### 3. Aggregation

Demgegenüber steht die sogenannte *Aggregation* oder auch „hat“-Beziehung.



Diese wird im UML-Diagramm mit einer leeren Raute dargestellt. Häufig wird auch die sogenannte *Multiplizität* als Zahl an die Verbindung geschrieben. Im Beispiel bedeutet diese, dass ein Kunde *genau ein* Konto hat.

Andere Multiplizitäten sind:

- **1**: genau eins
- **1..n**: mindestens eins, maximal n
- **1..\***: mindestens eins, ohne Maximale Anzahl
- **0..\***: muss keins enthalten, ohne maximale Anzahl (oft auch nur als **\*** geschrieben)

### 4. Aufgabe

Bringe die folgenden Klassen und deren Beziehungen in ein UML-Diagramm:

- Pflanze
- Ast
- Buche
- Baum
- Blatt
- Eiche
- Stamm
- Strauch
- Lavendel

### 5. Aufgabe

Zeichne ein UML-Diagramm mit folgenden Begriffen:

- Kamel
- Lama
- Alter
- Dromedar
- Höcker
- Größe
- Trampeltier
- Gewicht
- Farbe

### 6. Aufgabe: Komposition

Informiere dich über den Unterschied zwischen *Aggregation* und *Komposition* und nenne ein Beispiel dafür.

Wie wird eine Komposition im UML-Diagramm dargestellt?