

# Grundkonzepte der objektorientierten Programmierung

### 1. Java Grundlagen: Entwicklungszyklus, Entwicklungsumgebung

2. Datentypen, Kodierung, Binärzahlen, Variablen, Arrays

3. Ausdrücke, Operatoren, Schleifen und Verzweigungen

4. Blöcke, Sichtbarkeit und Methoden (Teil 1)

5. Grundkonzepte der Objektorientierung

6. Objektorientierung: Sichtbarkeit, Vererbung, Methoden (Teil 2), Konstruktor

7. Packages, lokale Klassen, abstrakte Klassen und Methoden, Interfaces, enum

8. Arbeiten mit Objekten: Identität, Listen, Komparatoren, Kopien, Wrapper, Iterator

9. Fehlerbehandlung: Exceptions und Logging

10. Utilities: Math, Date, Calendar, System, Random

11. Rekursion, Sortieralgorithmen und Collections

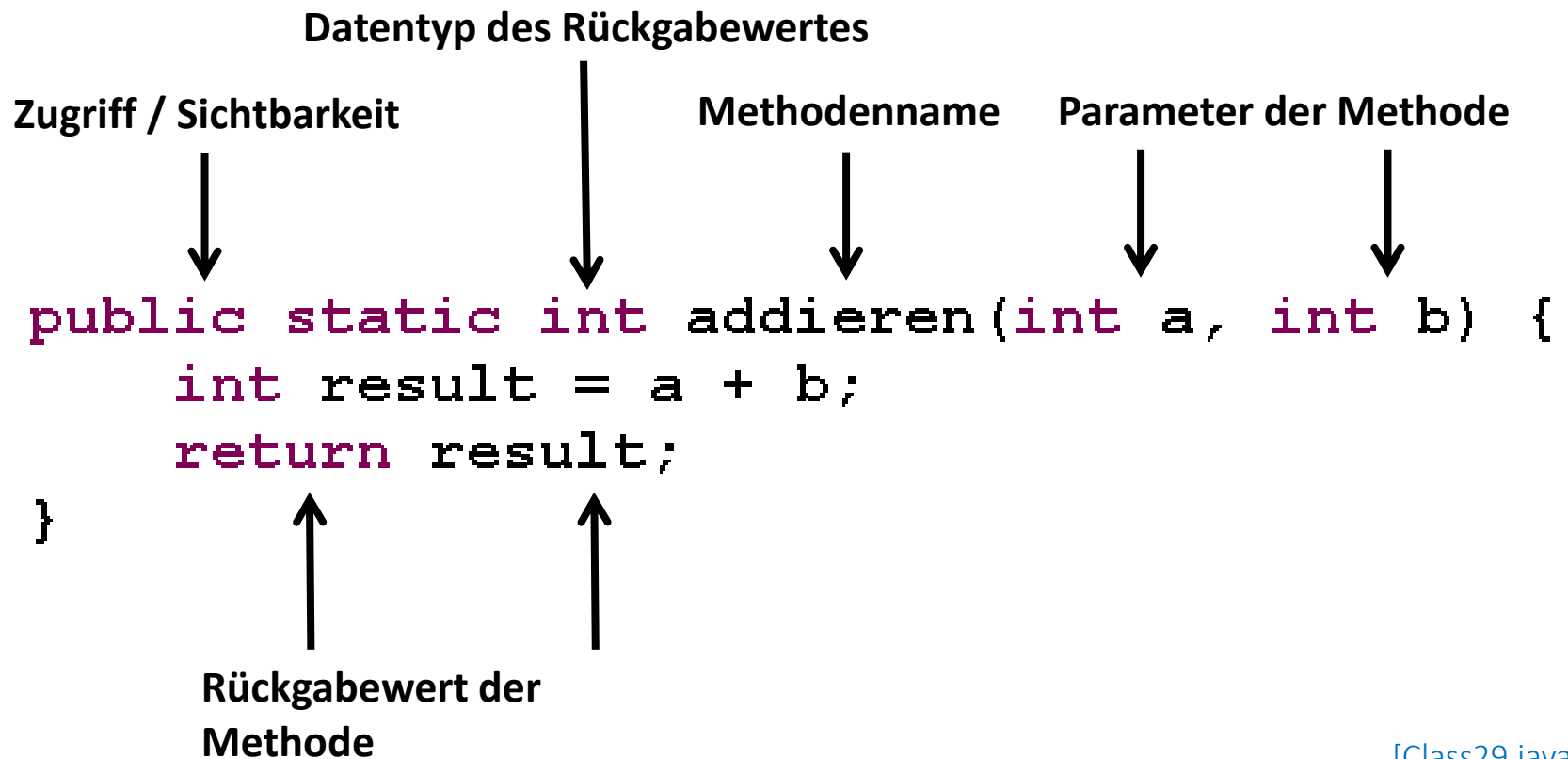
12. Nebenläufigkeit: Arbeiten mit Threads

13. Benutzeroberflächen mit Swing

14. Streams: Auf Dateien und auf das Netzwerk zugreifen

- Verzweigungen (if, switch, ...)
- Schleifen (while, for, ...)
- Blöcke
- Variablen-Sichtbarkeit
- Funktionen / Methoden

## Aufbau einer Methode

[\[Class29.java\]](#)

# Objektorientierte Programmierung (Grundkonzepte)

- Erste Programmiersprache: Simula67
- Bereits in den 60er Jahren entstanden
- Seit den 90er Jahren umfassend genutzt
- *„Die objektorientierte Programmierung war eine der Silver Bullets, die die Software-Industrie aus der Krise führte...“*
- Robustere, fehlerärmere, besser wartbare Programme

## Objekt / Instanz:

- Ein „tatsächlich existierendes Ding“ aus der Anwendungswelt des Programms
- **Beispiele:**
  - Der Kursteilnehmer Max Mustermann
  - Der Auftrag von Kunden Meier
  - Der Ort Regensburg
  - Die Uhrzeit 12:15:03 Uhr
  - Die E-Mail von Kunden Meier
  - ...

### Klasse:

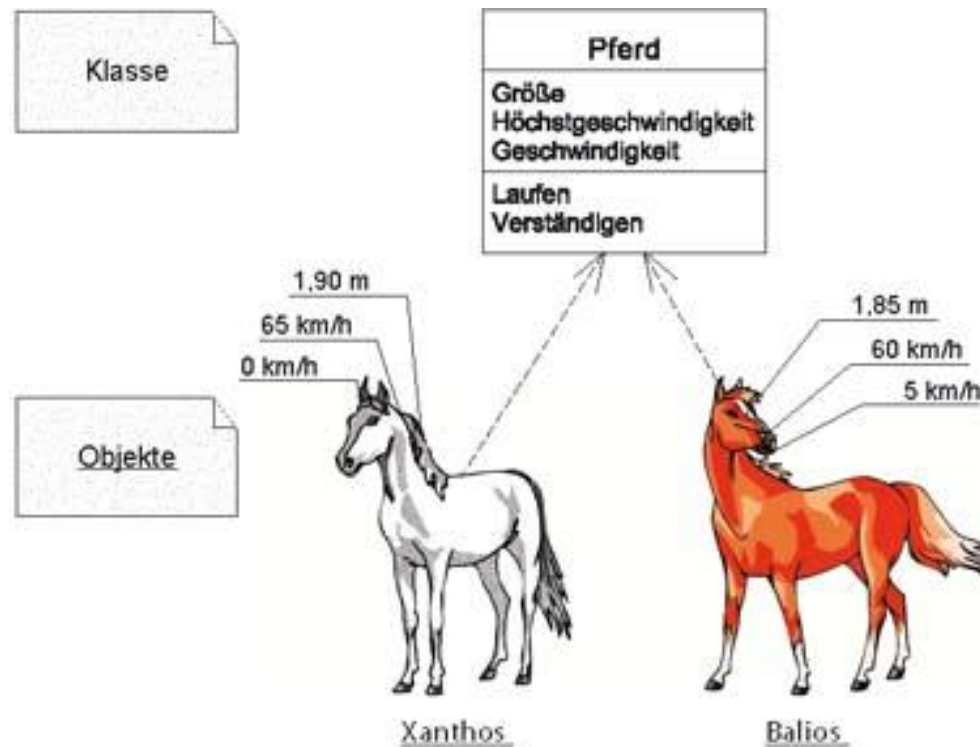
- Die Beschreibung eines oder mehrerer ähnlicher Objekte
- Beispiele:
  - Der Kursteilnehmer Max Mustermann - **Kursteilnehmer**
  - Der Auftrag von Kunden Meier - **Kunde**
  - Der Ort Regensburg - **Ort**
  - Die Uhrzeit 12:15:03 Uhr - **Uhrzeit**
  - Die E-Mail von Kunden Meier - **E-Mail**
  - ...



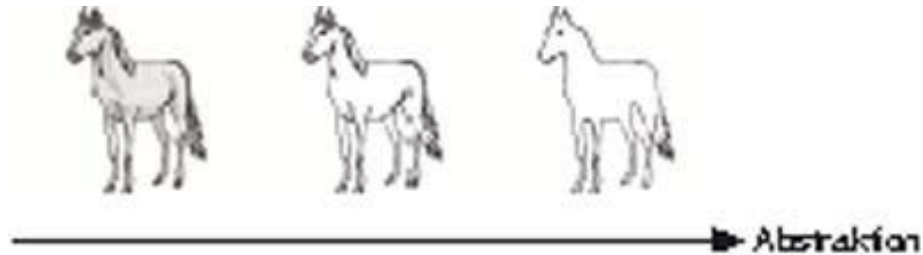
Die **Klassen-Beschreibung** umfasst zumindest drei Bestandteile:

1. Wie ist das Objekt (dieser Klasse) zu nutzen?
2. Welche Eigenschaften hat das Objekt und wie verhält es sich?
3. Wie wird das Objekt erzeugt?

- Beispiel: Klasse Pferd
- Objekte: Xanthos und Balios



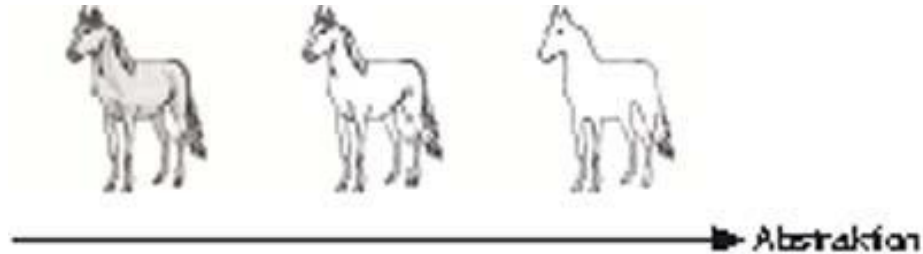
## Abstraktion



## Trennung zwischen Konzept und Umsetzung

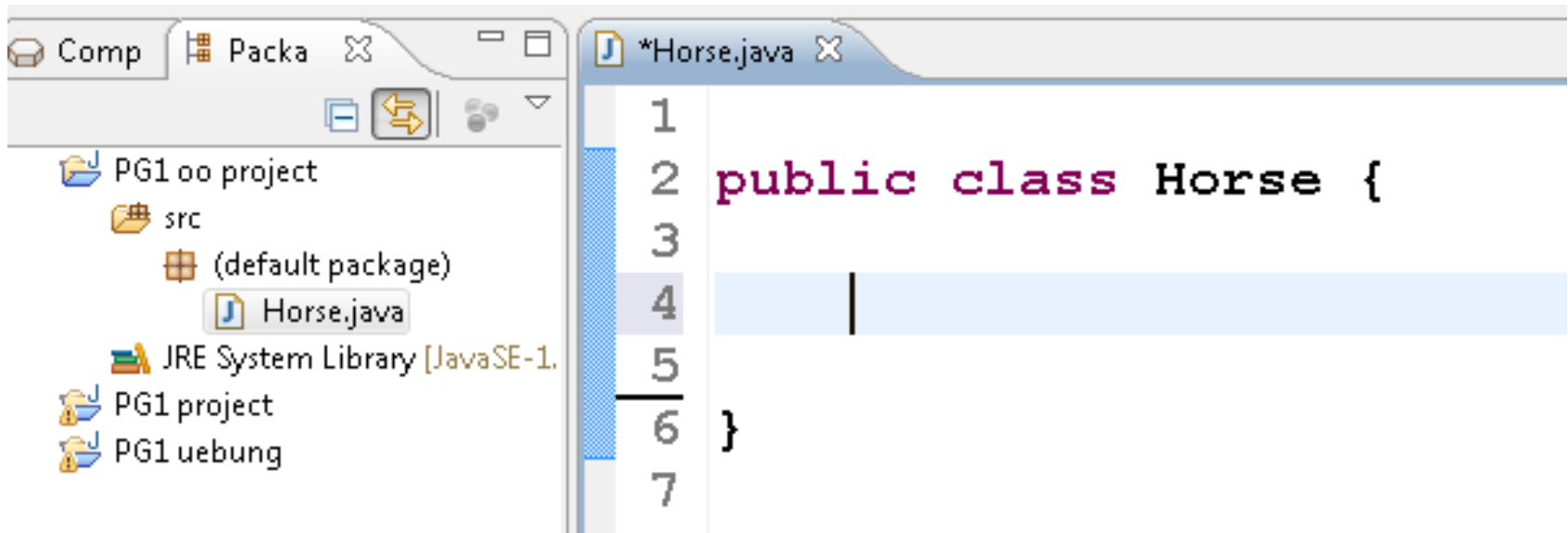
- Bauplan und Bauteil
- Rezept und Speise

## Abstraktion

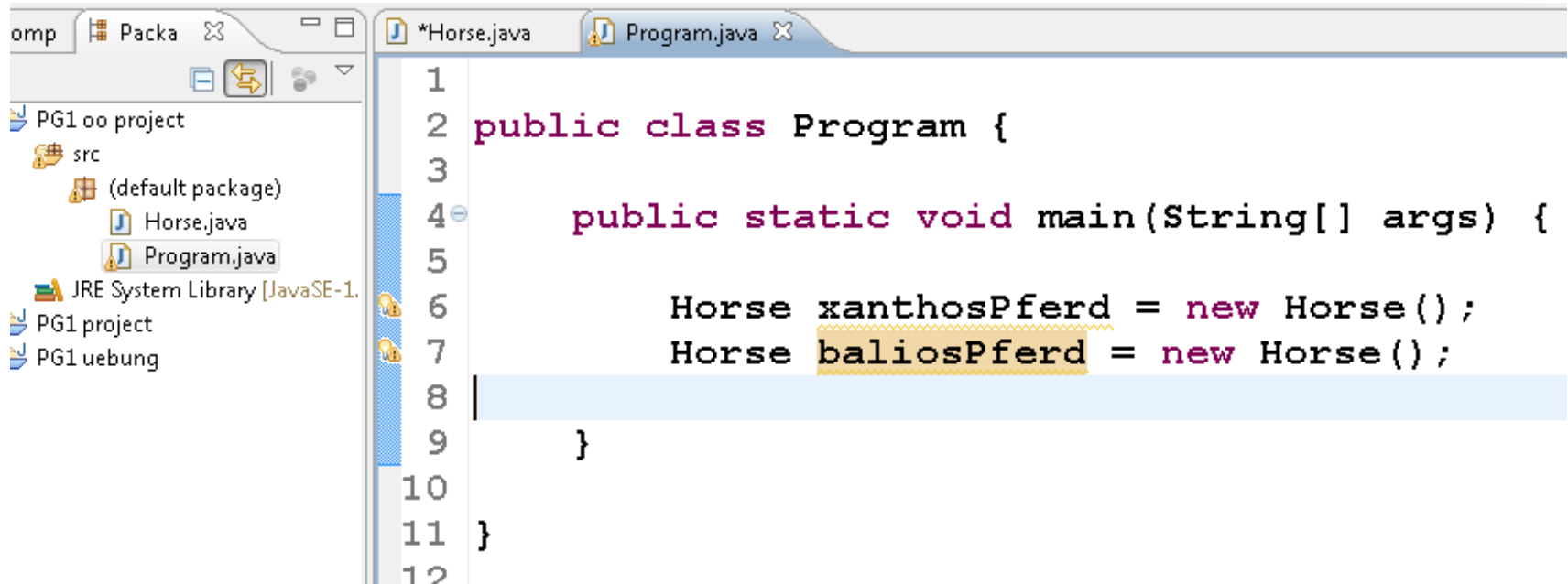


- *„Die Fähigkeit zur Abstraktion ist eine der wichtigsten Voraussetzungen zur Beherrschung komplexer Apparate und Techniken und kann in seiner Bedeutung nicht hoch genug eingeschätzt werden“*

## Klasse definieren

[\[Horse.java\]](#)

## Objekt erzeugen



The screenshot shows an IDE with a project named 'PG1 oo project'. The 'src' folder contains two files: 'Horse.java' and 'Program.java'. The 'Program.java' file is open, showing the following code:

```
1  
2 public class Program {  
3  
4     public static void main(String[] args) {  
5  
6         Horse xanthosPferd = new Horse();  
7         Horse baliosPferd = new Horse();  
8  
9     }  
10  
11 }  
12
```

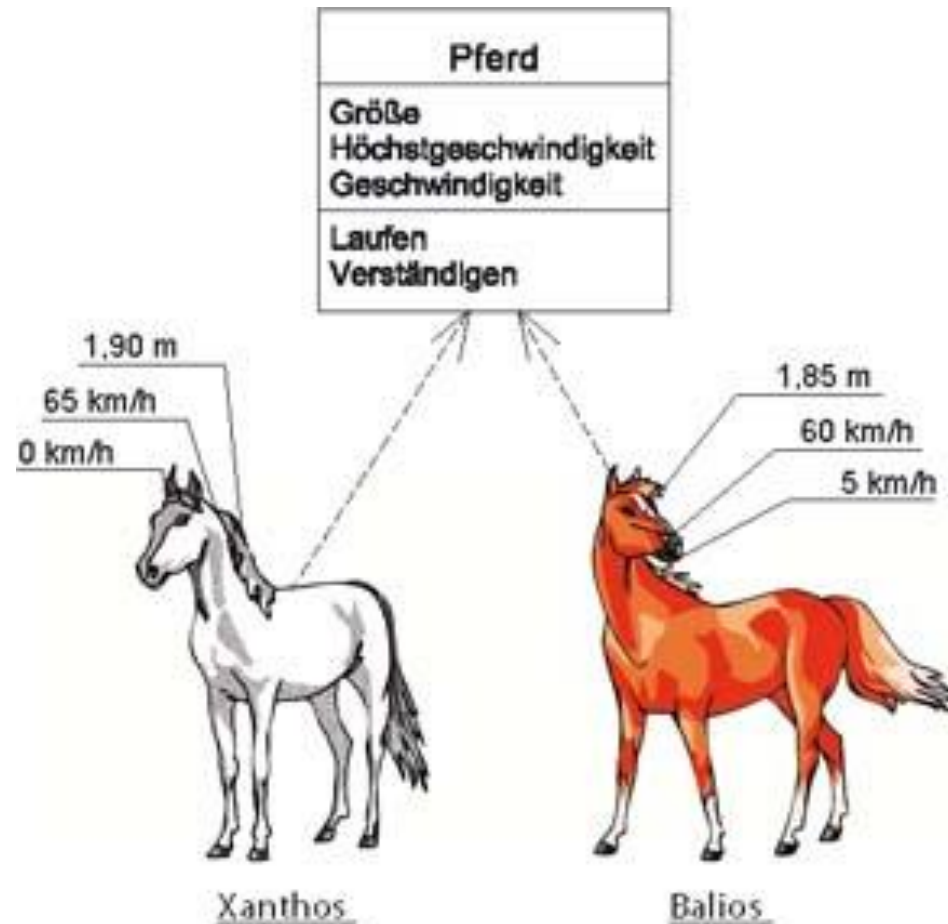
[\[Program.java\]](#)

- **Kapselung:** Zustand und Verhalten eines Objekts werden zusammengefasst und verborgen
- Der Zustand wird durch Instanzvariablen repräsentiert (auch Attribute, Member-Variablen oder Instanz-Merkmale genannt)
- Das Verhalten wird durch die Methoden (ehem. Funktionen) definiert

## Beispiel: Klasse Pferd

Instanzvariablen:

- Größe
- Höchstgeschwindigkeit
- Geschwindigkeit

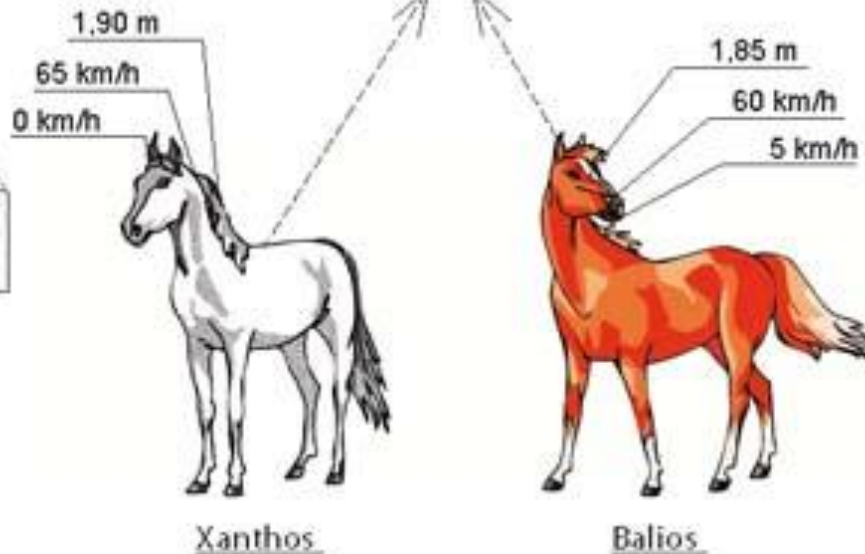
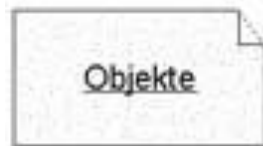






```

2 public class Horse {
3
4     private int     size;
5     private float   maximumSpeed;
6     private float   speed;
7
8 }
    
```

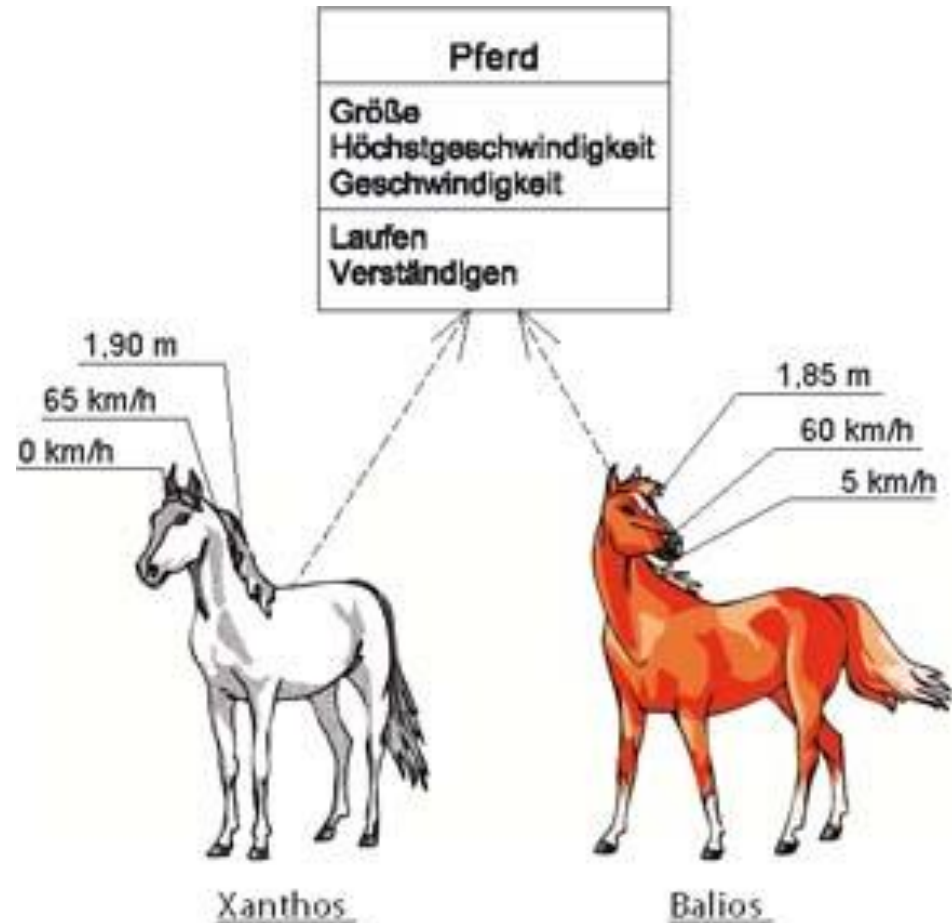


[Horse.java]

## Beispiel: Klasse Pferd

Methoden:

- Laufen
- Verständigen



```
Programmi.java  
public class Horse {  
  
    public int size;  
    public float maximumSpeed;  
    public float speed;  
    public String name;  
  
    public void run() {  
        speed = 20.0F;  
    }  
  
    public String speak() {  
        String result = "Mein Name ist " + name + ".";  
        result += "Ich laufe " + speed + " km/h.";  
        return result;  
    }  
}
```

[\[Horse.java\]](#)

```
2 public class Program {  
3  
4     public static void main(String[] args) {  
5  
6         Horse xanthosPferd = new Horse();  
7         Horse baliosPferd = new Horse();  
8  
9         xanthosPferd.name = "Xanthos";  
10        baliosPferd.name = "Balios";  
11  
12        System.out.println(xanthosPferd.speak());  
13        System.out.println(baliosPferd.speak());  
14    }  
15  
16 }
```

[\[Program.java\]](#)

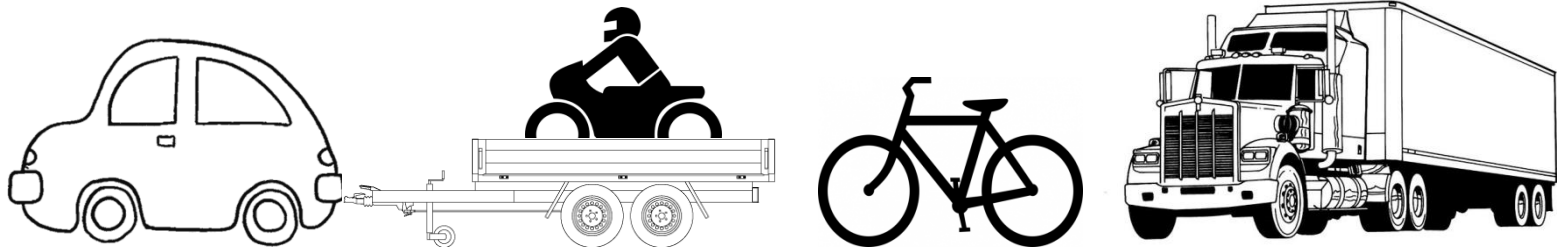
**Wiederverwendung:** Funktionalität einmal programmieren und in jeder Instanz nutzen



```
3 public static void main(String[] args) {  
4  
5     Horse[] horses = new Horse[10];  
6  
7     // 10 Pferde anlegen und benennen  
8     for (int i = 0; i < horses.length; i++) {  
9         Horse currentHorse = new Horse();  
10        currentHorse.name = "Pferd Nummer " + i;  
11        horses[i] = currentHorse;  
12    }  
13  
14    // Pferde laufen lassen  
15    for (int i = 0; i < horses.length; i++) {  
16        horses[i].run();  
17        System.out.println(horses[i].speak());  
18    }  
19 }  
20 }
```

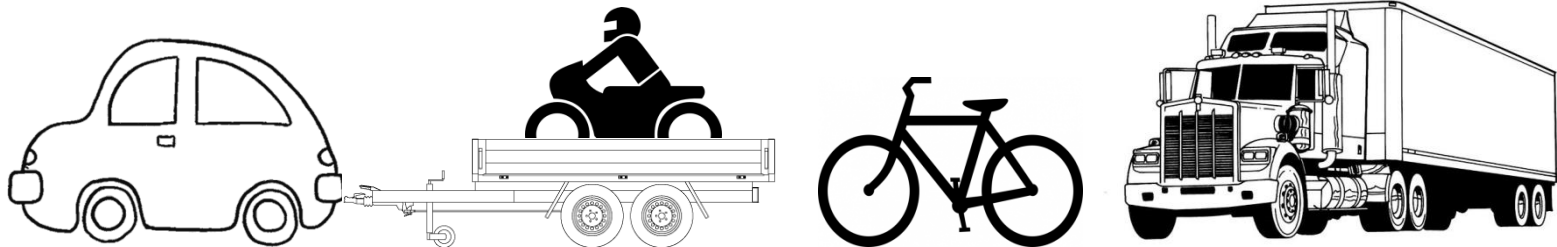
[\[RunningHorses.java\]](#)

- **Beziehungen zwischen Objekten und Klassen**
- Fahrzeug-Beispiel: Ein Auto ähnelt einem Lastwagen. Dieser kann einen Anhänger haben, auf dem ein Motorrad steht [vgl. Krüger, Stark, Handbuch der Java-Programmierung S. 158]. usw.



## Es gibt drei Beziehungsarten:

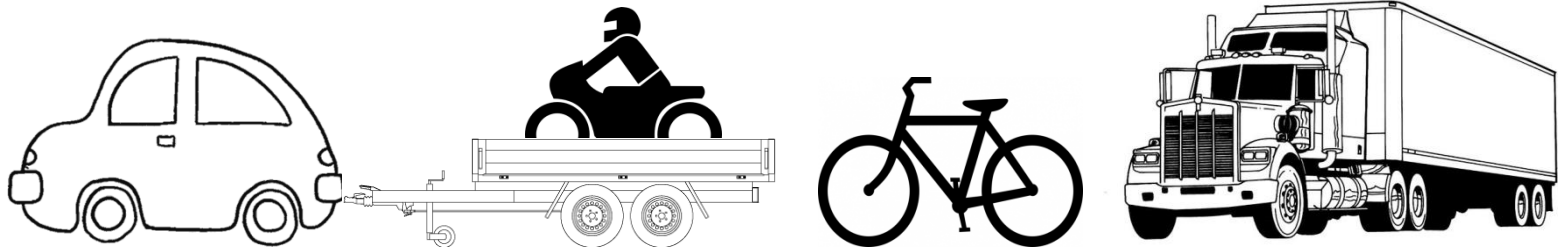
1. „is-a“-Beziehung (Generalisierung, Spezialisierung)
2. „part-of“-Beziehung (Aggregation, Komposition)
3. Verwendungs- und Aufrufbeziehung





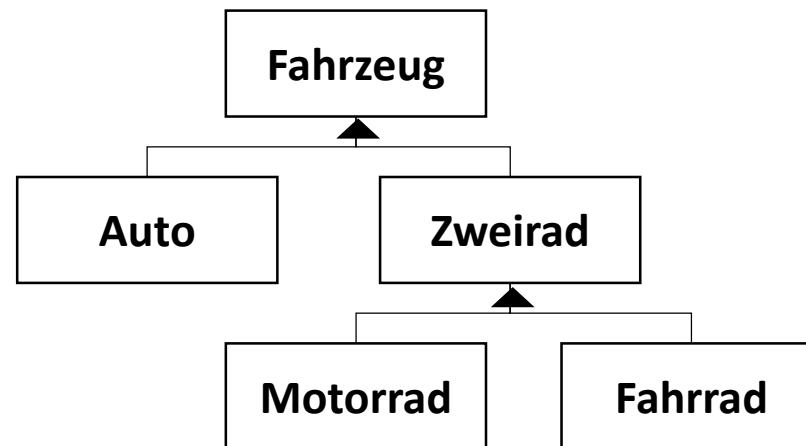
## Generalisierung und Spezialisierung:

- Ein Auto ist ein Fahrzeug
- Ein Zweirad ist ein Fahrzeug
- Ein Fahrrad und ein Motorrad sind Zweiräder



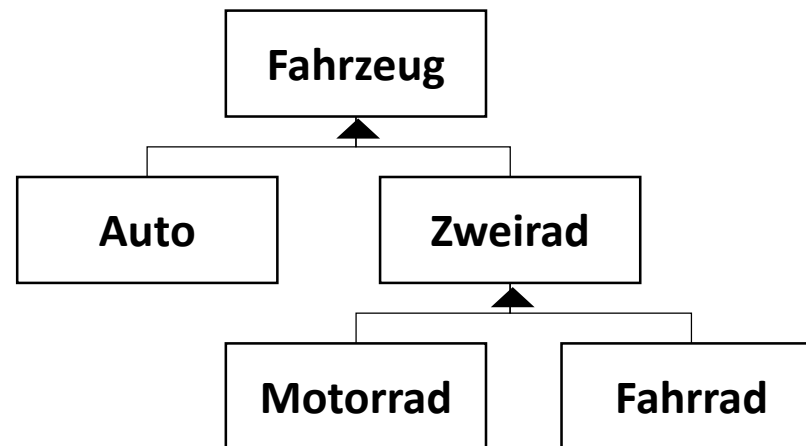
## Generalisierung und Spezialisierung:

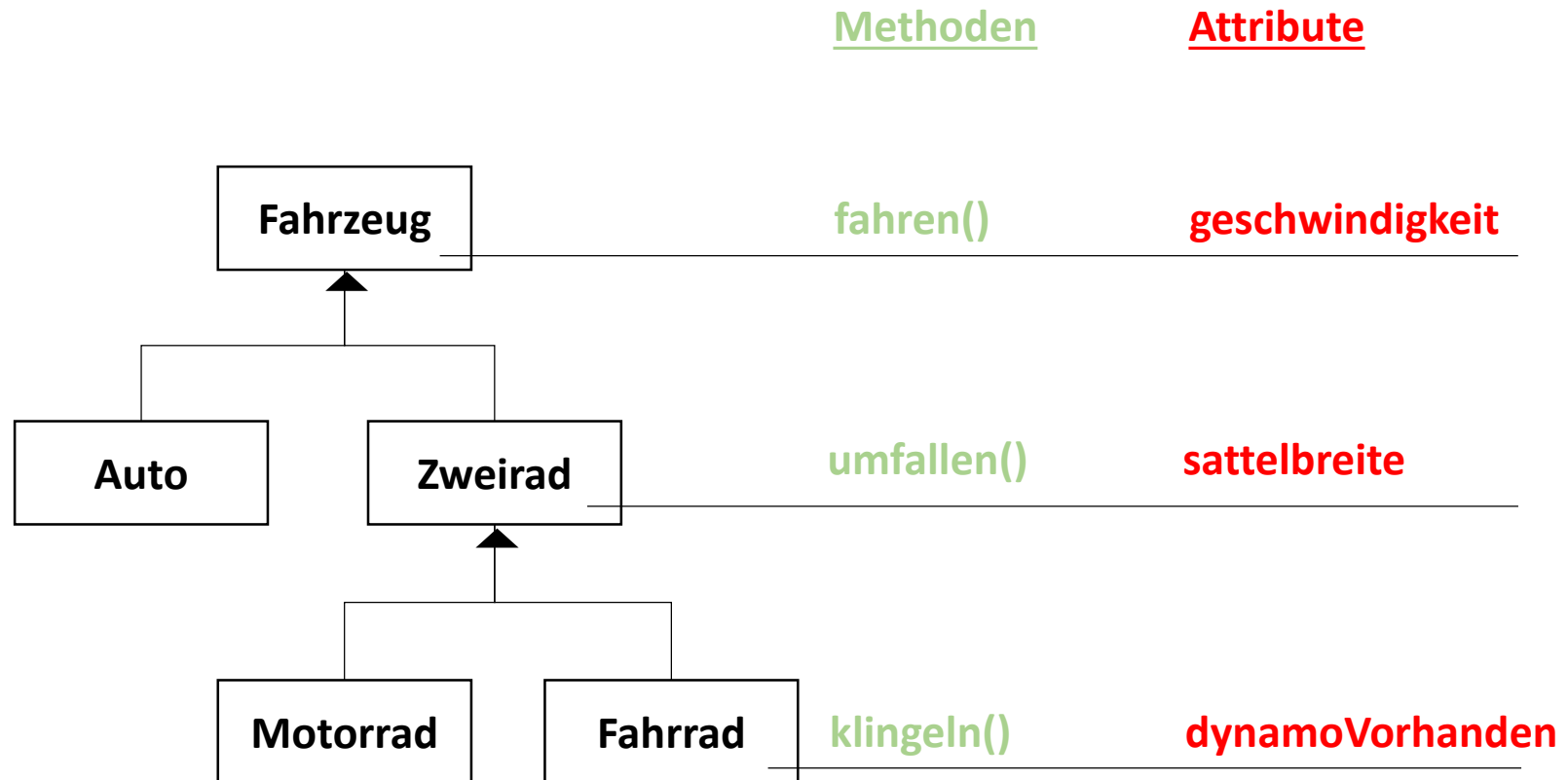
- Ein Auto ist ein Fahrzeug
- Ein Zweirad ist ein Fahrzeug
- Ein Fahrrad und ein Motorrad sind Zweiräder



## Generalisierung und Spezialisierung:

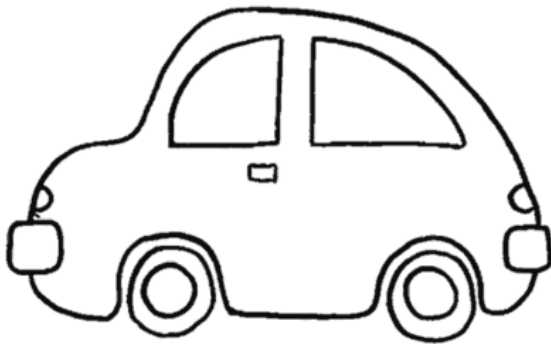
- Die „is-a“ Beziehung zwischen zwei Klassen A und B sagt aus, dass B ein A ist, d. h. dass B alle Eigenschaften von A besitzt.





## Zusammensetzung bzw. Komposition:

- Ein Auto hat Reifen
- Ein Auto hat Türen
- ...



## Zusammensetzung bzw. Komposition:

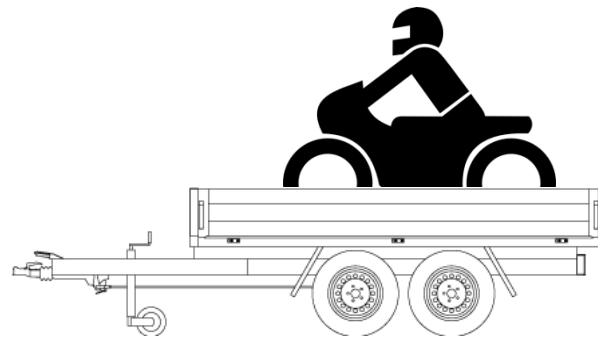
- Die „part-of“ Beziehung beschreibt, woraus ein Objekt besteht



```
1  public class Car {  
2  
3      Wheel leftFrontWheel;  
4      Wheel leftBackWheel;  
5      Wheel rightFrontWheel;  
6      Wheel rightBackWheel;  
7      Door  rightDoor;  
8      Door  leftDoor;  
9  
10 }
```

## Aggregation:

- Die „part-of“ Beziehung kann auch das einfache Aufnehmen eines anderen Objekts beschreiben.
- Der Anhänger trägt das Motorrad. Er setzt sich aber nicht aus diesem zusammen.



## Eine Methode nutzt eine Methode eines anderen Objekts

```
1 public class Program {  
2  
3     public static void main(String[] args) {  
4  
5         Horse xanthosPferd = new Horse();  
6         Horse baliosPferd = new Horse();  
7  
8         xanthosPferd.name = "Xanthos";  
9         baliosPferd.name = "Balios";  
10  
11         System.out.println(xanthosPferd.speak());  
12         System.out.println(baliosPferd.speak());  
13     }  
14  
15 }
```

[Program.java]



## Beispiele

## Ein Webshop verwaltet Artikel, Kunden und Aufträge

Ein Artikel ist gekennzeichnet durch

- Nummer
- Name
- Preis
- Gewicht



Ein **Kunde** ist bezeichnet durch:

- Nummer
- Name
- Anschrift
- Kreditlimit

Ein **Auftrag** besteht aus:

- Kunde
- Artikel
- Versandpreis



- Ein Kunde kann Artikel bestellen (Auftrag); Beispiel:
  - 6x 12212 Buch    20,00 Euro    120,00 Euro
  - 4x 32212 Block    3,00 Euro    12,00 Euro
- Ein Auftrag hat eine Gesamtsumme, diese umfasst die Summe der Artikelpreise und die Versandkosten
- Der Auftrag ist nur gültig, wenn die Gesamtsumme kleiner oder gleich dem Kreditlimit des Kunden ist
- Die Versandkosten sind abhängig vom Gewicht des Auftrags. Bis zu 10kg kosten 5 Euro, mehr als 10kg kosten 10 Euro

- Der Webshop ermöglicht es, einen Auftrag für einen Kunden zu erfassen
- Es wird geprüft, ob der Auftrag gültig ist (Kreditlimit)
- Der Gesamtpreis inkl. Versandkosten wird berechnet und der Auftrag wird mit allen Positionen, Kunden- und Gewichtsinformationen angezeigt

## Beispiel 2: Getränkeautomat

- Ein Getränkeautomat hat 3 Fächer für gekühlte Getränke
- Jedes Fach kann 10 Getränke aufnehmen
- Jedes Getränk hat einen Namen und einen Preis
- Der Automat gibt auf Knopfdruck ein Getränk aus (Vereinfachung: ohne zu bezahlen)



## Beispiel 2: Getränkeautomat

- Jedes Fach des Automaten kann aufgefüllt werden. Dabei dürfen aber nur so viele Flaschen nachgefüllt werden, wie in das Fach passen.
- Außerdem darf pro Fach nur ein Getränketyp vorhanden sein (bspw. Cola und Wasser in einem Fach nicht mischen)
- Der Automat kann jederzeit seinen Füllstand anzeigen

