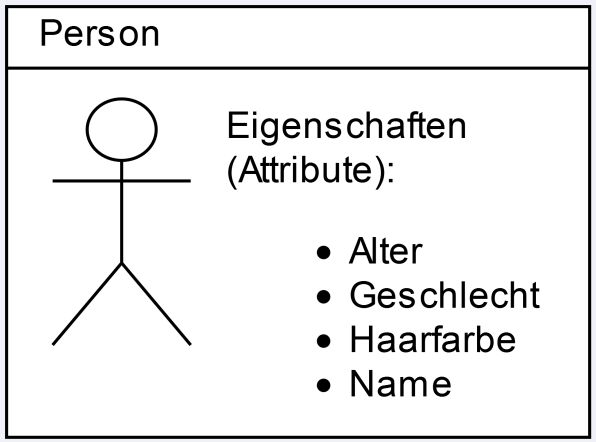
Java-Crashkurs

3. Objektorientierung | Benno Hölz, Matthias Ruf | 07.09.2023

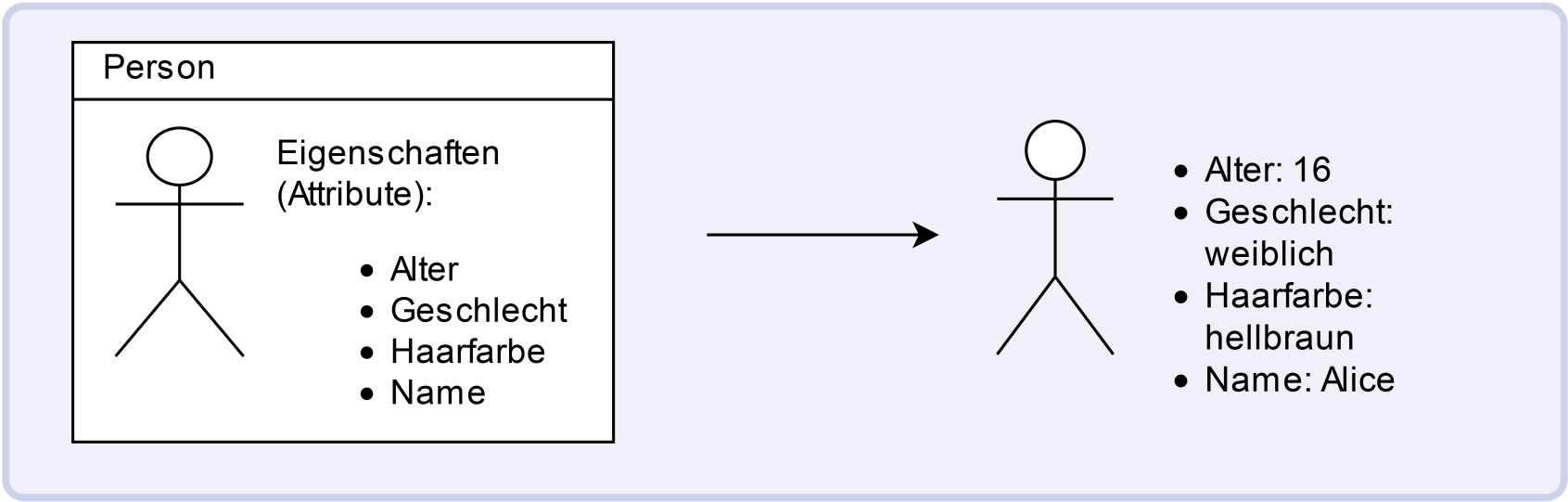
sfz®
schülerforschungszentrum
südwürttemberg

1. Was ist Objektorientierung?

Was ist Objektorientierung?



Was ist Objektorientierung?



2. Klassen

Klassen

Klasse

- Zusammenfassung von Eigenschaften ähnlicher Objekte
- Jede Klasse in einer eigenen Datei
- Datei muss den gleichen Namen besitzen wie Klasse

Beispiel: Auto.java

```
1 class Auto {
2 // [Attribute]
3 // [Methoden]
4 }
```

3. Attribute

Attribute

Attribute

- Eigenschaften eines Objektes
  → z.B. für ein Objekt vom Typ *Auto*: Hersteller, Farbe, Geschwindigkeit

```
1 class Auto {
2   public String hersteller;
3   public String farbe;
4   public int geschwindigkeit;
5
6   // [Methoden]
7 }
```

Attribute

Attribute

- Eigenschaften eines Objektes
  → z.B. für ein Objekt vom Typ *Auto*: Hersteller, Farbe, Geschwindigkeit
- Deklaration außerhalb von Methoden direkt in der Klasse

```
1 class Auto {
2   public String hersteller;
3   public String farbe;
4   public int geschwindigkeit;
5
6   // [Methoden]
7 }
```

Attribute – Sichtbarkeit

Sichtbarkeit von Attributen

- `private`: Nur innerhalb der Klasse sichtbar
- `public`: Auch von außerhalb sichtbar

Beispiel

```
1 class Auto{
2         public String hersteller;
3         private int kilometer;
4 }
```

Attribute – static

Statische Attribute

`static`: Zugriff auch ohne Objekterzeugung möglich

Beispiel

```
1 class Auto{
2         public String hersteller;
3         private int kilometer;
4         private static int anzahlAutosGesamt = 0;
5 }
```

4. Objekterzeugung

Objekterzeugung

- Erzeugung eines neuen Objektes mit *new*

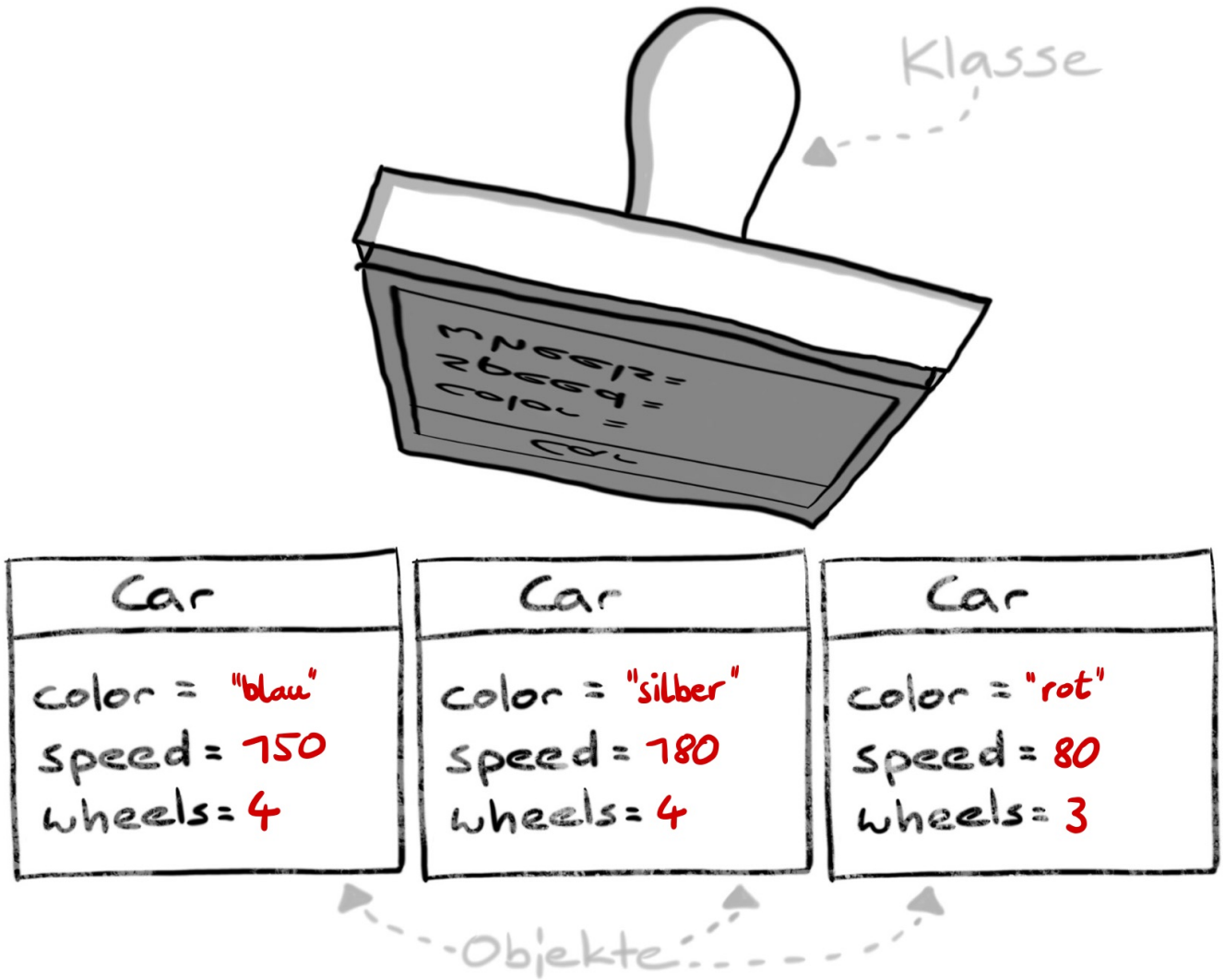
```
1 Auto auto = new Auto();
```

Objekterzeugung

- Zugriff auf Attribute (*public*):

```
1 // Wert setzen:
2 auto.geschwindigkeit = 240;
3
4 // Wert abrufen:
5 System.out.println(auto.geschwindigkeit); // 240
```

Objekterzeugung



5. Konstruktoren

Konstruktoren

Konstruktor

- Methode mit dem Namen der Klasse
- Automatische Ausführung bei Objekterzeugung
- meist: Initialisierung der Parameter eines Objektes

Konstruktoren

```
1  class Auto {
2    // Attribute:
3    String hersteller;
4    String farbe;
5    int geschwindigkeit;
6
7    // Konstruktor:
8    public Auto(String hersteller, String farbe, int
         geschwindigkeit) {
9      this.hersteller = hersteller;
10     this.farbe = farbe;
11     this.geschwindigkeit = geschwindigkeit;
12   }
13 }
```

this: Selbstverweis auf das Attribut der aktuellen Klasse

Konstruktoren

- mehrere Konstruktoren möglich (unterschiedliche Parameterlisten)
- kein Konstruktor definiert/Aufruf ohne Parameter
  $\Rightarrow$ Aufruf des **leeren Konstruktors**

Konstruktoren

- mehrere Konstruktoren möglich (unterschiedliche Parameterlisten)
- kein Konstruktor definiert/Aufruf ohne Parameter
  $\Rightarrow$ Aufruf des **leeren Konstruktors**
- Aufruf:

```
1 Auto audi = new Auto("Audi", "schwarz", 240);
2 Auto auto = new Auto(); // leerer Konstruktor
```

Zusammenfassung

- **Objekte, Klassen:** Zusammenfassung von Eigenschaften (**Attribute**)
- Objekterzeugung mit `new`
- **Konstruktor** (Ausführung bei Objekterzeugung)
- Sichtbarkeit: `public`, `private`
- `static` → Aufruf ohne Objekt