

Lernzettel

Konstruktoren, Instanz- und statische Methoden
sowie Objektinitialisierung

Universität:	Technische Universität Berlin
Kurs/Modul:	Einführung in die Informatik
Erstellungsdatum:	September 19, 2025



Zielorientierte Lerninhalte, kostenlos!
Entdecke zugeschnittene Materialien für deine Kurse:

<https://study.AllWeCanLearn.com>

Einführung in die Informatik

Lernzettel: Konstruktoren, Instanz- und statische Methoden sowie Objektinitialisierung

(1) Konstruktoren – Grundlagen.

Konstruktoren dienen der Initialisierung von Objekten. In Java hat ein Konstruktor denselben Namen wie die Klasse und keinen Rückgabewert.

Default-Konstruktor: Wird automatisch erzeugt, falls kein eigener Konstruktor definiert ist. Wird kein eigener Konstruktor bereitgestellt, existiert er weiterhin.

Überladen von Konstruktoren (Konstruktorsignaturen): Eine Klasse kann mehrere Konstruktoren mit unterschiedlichen Parameterlisten besitzen.

```
public class Fahrzeug {
    private String modell;
    private int baujahr;

    // Default-Konstruktor
    public Fahrzeug() {
        this("Unbekannt", 0);
    }

    // Parametrisierter Konstruktor
    public Fahrzeug(String modell, int baujahr) {
        this.modell = modell;
        this.baujahr = baujahr;
    }
}
```

this()-Aufruf innerhalb eines Konstruktors ruft einen anderen Konstruktor derselben Klasse auf.

super()-Aufruf in einer Unterklasse ruft den Konstruktor der Oberklasse auf.

```
class Ober {
    private String name;
    public Ober(String name) { this.name = name; }
}

class Unter extends Ober {
    private int tiefe;
    public Unter(String name, int tiefe) {
        super(name);
        this.tiefe = tiefe;
    }
}
```

```
}  
}
```

(2) Instanz- und statische Methoden.

Instanzmethoden gehören zu Objekten (Instanzen) einer Klasse. Sie benötigen ein Objekt, um aufgerufen zu werden.

Statische Methoden (mit dem Schlüsselwort `static`) gehören zur Klasse selbst und können ohne Objekt aufgerufen werden.

```
public class Rechner {  
    // Instanzmethode  
    public int addiere(int a, int b) {  
        return a + b;  
    }  
  
    // Statische Methode  
    public static int max(int a, int b) {  
        return (a > b) ? a : b;  
    }  
}
```

```
Rechner r = new Rechner();  
int s = r.addiere(3, 4);      // Instanz-Aufruf  
int m = Rechner.max(5, 7);   // statischer Aufruf
```

Statische Felder und Initialisierer:

```
public class Zähler {  
    private static int anzahl = 0;  
    private int id;  
  
    public Zähler() {  
        id = ++anzahl;  
    }  
  
    public static int getAnzahl() { return anzahl; }  
}
```

Init-Block (Instanzinitialisierer):

```
public class InitBeispiel {  
    private int x;  
    { x = 5; } // Instanzinitialisierungsblock  
}
```

(3) Objektinitialisierung – Reihenfolge und Ablauf.

Beim Erzeugen eines Objekts in Java erfolgt grob Folgendes: - Speicher wird allokiert; Felder erhalten Default-Werte. - Instanzinitialisierer (Feld-Initialisierer) und Instanz-Init-Blöcke werden in Reihenfolge ausgeführt. - Der Konstruktor der Klasse wird ausgeführt.

Hinweis: Static-Felder und statische Initialisierer werden ausgeführt, sobald die Klasse geladen wird, nicht pro Objekt.

(4) Kleines Praxis-Beispiel: Konstruktoren, Instanz- und statische Methoden

```
public class Auto {
    private String modell;
    private int baujahr;
    private static int anzahlAutos = 0;

    // Default-Konstruktor
    public Auto() {
        this("Unbekannt", 0);
    }

    // Parametrisierter Konstruktor
    public Auto(String modell, int baujahr) {
        this.modell = modell;
        this.baujahr = baujahr;
        anzahlAutos++;
    }

    public static int getAnzahlAutos() { return anzahlAutos; }

    public String getModell() { return modell; }
    public int getBaujahr() { return baujahr; }
}

Auto a = new Auto("Golf", 2019);
Auto b = new Auto();
System.out.println(Auto.getAnzahlAutos()); // Ausgabe: 2
```