

Mehr zu Referenzen

- Implementiere eine Klasse **Test** mit folgenden Eigenschaften:
 - Jedes Objekt der Klasse **Test** soll wissen, das wieviele Objekt dieser Klasse es ist und diese Zahl bei der Erzeugung ausgeben
 - Die Zahl soll **nicht** als Paramter an den Konstruktor übergeben werden
- Beispielhafte Verwendung:

```
Test t1 = new Test(); // gibt "1" aus
Test t2 = new Test(); // gibt "2" aus
// usw...
```

Aufgabe 1 - Überlegungen (1)

- Wo geben wir die aktuelle Nummer aus und erhöhen die Gesamtzahl der bisher erzeugten Objekte?
- Wo speichern wir die Zahl der bisher bereits erzeugten Objekte?
 - Ein Attribut in der Klasse **Test** scheidet aus. Wieso?

- Wir brauchen eine weitere Klasse **Zähler**, die einen Zählerstand speichert und eine Methode zum hochzählen anbietet
- Von der Klasse **Zähler** wird ein einziges Objekt erzeugt
- Alle Objekte der Klasse **Test** müssen dieses Zähler-Objekt kennen und nutzen

- Jedes Objekt der Klasse **Test** erhielt eine Referenz auf dasselbe Objekt der Klasse **Zaehler**
 - Im Konstruktor von **Test** konnte dann der Zählerstand erhöht und ausgegeben werden
- Ein Attribut der Form **int zaehler** in der Klasse **Test** kam nicht in Frage
 - Attribute sind **objektspezifisch**, nicht klassenspezifisch
- Die gefundene Lösung funktioniert, ist aber verhältnismäßig komplex
 - Wir mussten eine weitere Klasse definieren, die im Wesentlichen nur einen Wert vom Typ **int** speichert
- Geht es auch einfacher?
 - Würde ich fragen, wenn es nicht ginge? ;-)

Klassenattribute und -methoden

- Eine Klasse in Java kann über sogenannte **Klassenattribute** verfügen
- Ein solches Klassenattribut teilen sich alle Objekte der Klasse
 - Verändert ein Objekt den Wert des Klassenattributs, so “sehen” alle anderen Objekte auch diese Veränderung
- Das Schlüsselwort **static** zeigt an, dass ein Attribut ein Klassenattribut ist

```
class Test {  
    private static int zaehler; // Klassenattribut  
    private int irgendwas; // "gewöhnliches" Attribut  
  
    // ...  
  
}
```


- Schreibe eine weitere Klasse **Test2**, die das Problem aus Aufgabe 1 (Zählen und eigene Nummer ausgeben) durch die Verwendung eines Klassenattributs löst
 - Die Klasse **Zaehler** soll nicht mehr benutzt werden!
- Verändere das Programm so, dass sich jedes Objekt der Klasse **Test2** seine eigene Nummer “merkt”, sodass man sie auch zu einem späteren Zeitpunkt abrufen kann

- Definieren wir ein Klassenattribut wie im Beispiel oben, so wird es von Java auf 0 gesetzt
 - Was, wenn wir einen anderen Anfangswert wollen?
- Setzen im Konstruktor führt nicht zum gewünschten Ergebnis
 - Warum nicht?

Der Konstruktor wird jedes Mal aufgerufen, wenn ein Objekt der Klasse erzeugt wird, er würde also jedes Mal das Klassenattribut wieder auf den festgelegten Anfangswert zurücksetzen
- Die Zuweisung muss direkt bei der Deklaration erfolgen:

```
public static int irgendwas = 25;
```

Klassenmethoden (1)

- Das Schlüsselwort **static** kann auch vor Methoden platziert werden, z. B.

```
class Test {  
    public static void machWas() {  
        // ...  
    }  
}
```

- Was bringt das?

- Eine **Klassenmethode** ist eine Methode, die aufgerufen werden kann, ohne dass ein Objekt der Klasse existieren muss
 - Im Beispiel von der letzten Folie: `Test.machWas()`;
- Voraussetzungen:
 - Eine Klassenmethode darf nur lokale (also innerhalb der Methode definierte) Variablen sowie Klassenattribute benutzen

- Erweitere die Klasse **Test2** aus Aufgabe 2 um eine Klassenmethode, die den gespeicherten Zählerstand zurückgibt
- Versuche zusätzlich, eine Klassenmethode zu implementieren, die die individuelle Nummer zurückgibt