STATIC

static

Unabhängigkeit von Instanz

- keine Instanz nötig für Nutzung
- Wirkung möglich auf
  - Attribute
  - Methoden
  - Klasse

STATIC BEI VARIABLEN

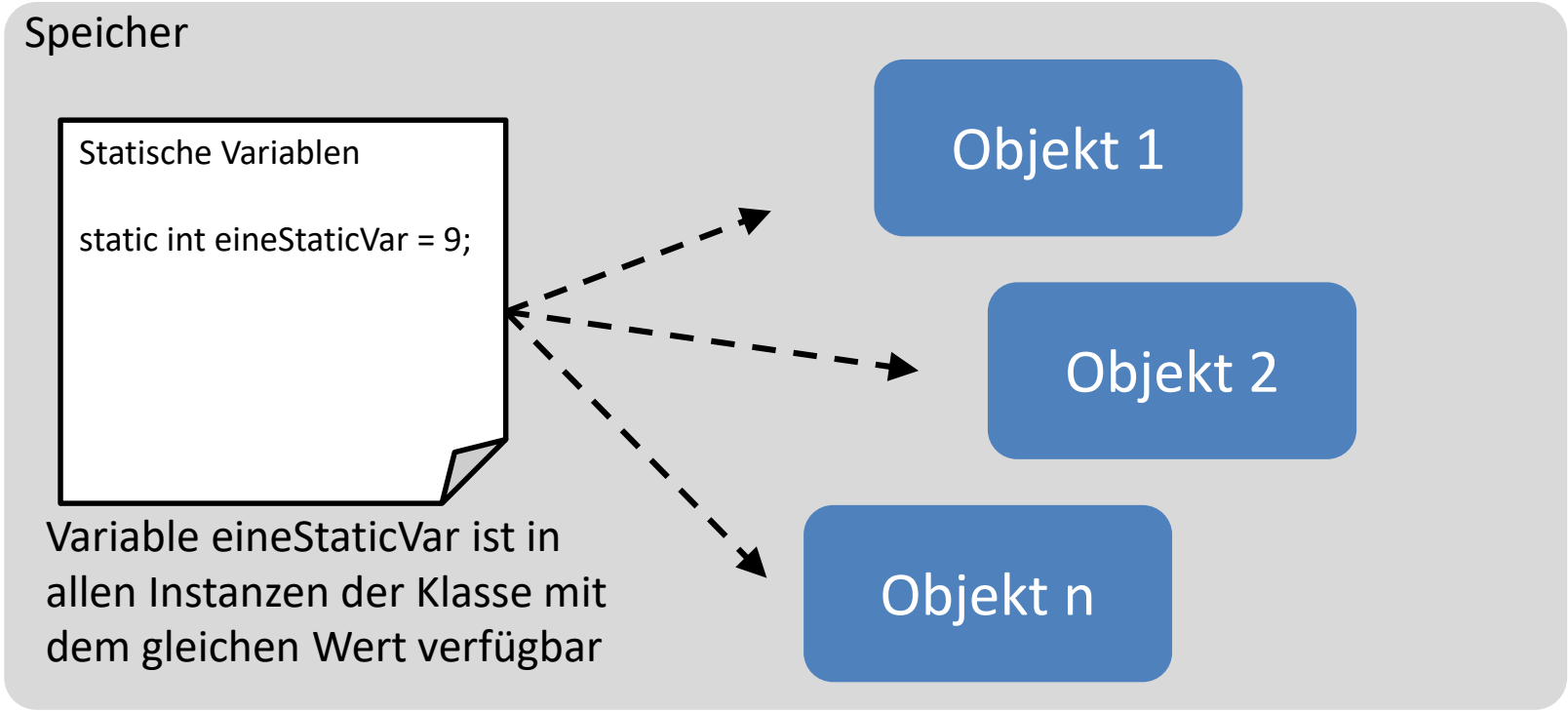
static bei Variablen

Wirkung von static auf Variable

- Variable ist nur ein mal für alle Instanzen vorhanden
- gehört nicht zu instanziiertem Objekt
- unabhängig von Instanz nutzbar

static verwenden wenn

- Wert der Variable unabhängig von Instanzen ist
- Wert der Variable für alle Objekte bekannt sein soll



static bei Variablen

Ein Wert pro Klasse

- Variable static existiert nur ein Mal für alle Objekte
- hochzählen bei neuem Instanziieren

```
public class ClassWithStaticVariable {

     private String eineStringVariable;
     private String zweiteStringVariable;

     public static int anzahl=0;

     public ClassWithStaticVariable(String var1, String var2) {
          eineStringVariable=var1;
          zweiteStringVariable=var2;

          // hochzaehlen der Anzahl der bisher instanziierten Objekte
          anzahl++;
     }
}
```

- Constructor wird zwei Mal aufgerufen -> zwei Mal hochzählen

```
ClassWithStaticVariable obj1 = new ClassWithStaticVariable("abc", "123");
ClassWithStaticVariable obj2 = new ClassWithStaticVariable("def", "456");

System.out.println("Anzahl (obj1): "+obj1.anzahl);
System.out.println("Anzahl (obj2): "+obj2.anzahl);
```

- Variable anzahl hat in obj1 und obj2 jeweils den gleichen Wert

demo005.statickeyword

STATIC BEI METHODEN

static bei Methoden

Wirkung von static auf Methode

demo005.statickeyword

- Methode kann unabhängig von Objekt aufgerufen werden
- Methode bezieht sich nicht auf ein Objekt
- Ausführung ist gleich für alle Objekte

```
public static void anzahlAufWertSetzen(int wert) {
    ClassWithStaticVariable.anzahl = wert;
}
```

Verwendungsbeispiele static

- Collections (versch. Methoden, um Algorithmen auf Collections auszuführen)
- Math (math. Funktionen, abhängig nur von übergebenen Werten, nicht von Objekten)
- Hilfsklassen mit allgemeinen Algorithmen
- Methode für Zugriff auf static Variablen

```
System.out.println("Anzahl (obj1): "+obj1.anzahl);
System.out.println("Anzahl (obj2): "+obj2.anzahl);

obj1.anzahlAufWertSetzen(10);

System.out.println("Anzahl nach setzen (obj1): "+obj1.anzahl);
System.out.println("Anzahl nach setzen (obj2): "+obj2.anzahl);
```

Wert der static Variable wird gesetzt

Wert ist gleich für beide Objekte

STATIC BEI KLASSEN

static bei Klassen

Möglichkeiten static bei Klassen

- „Normale“ Klasse kann nicht static sein
- innere Klassen können static sein
- Innere Klasse kann ohne Instanz einer äußeren Klasse genutzt werden

Anwendung

- Singleton Pattern

```
public class OuterClass {
     public static class NestedClass {
          public void fooMethod() {
               System.out.println("Ausgabe der fooMethod in NestedClass");
          }
     }
     ...
}
```

```
public static void main(String args[]) {
     OuterClass.NestedClass nestedClass = new OuterClass.NestedClass();
          nestedClass.fooMethod();
     }
}
```

Zusammenfassung

static führt zu Unabhängigkeit von Instanz

- keine Instanz nötig für Nutzung
- Wirkung möglich auf
  - Attribute
  - Methoden
  - Klasse