

Einführung: Methoden

- bisher haben wir sämtlichen Code in die main-Methode geschrieben
- sobald Programme länger und komplexer werden, wird der Code innerhalb der main-Methode immer unübersichtlicher
- Methoden ermöglichen nun, dass Programm besser zu strukturieren, indem wir einzelne Funktionen des Programms aus der main-Methode auslagern
- Methoden zerlegen großes komplexes Problem in viele kleine Teilprobleme

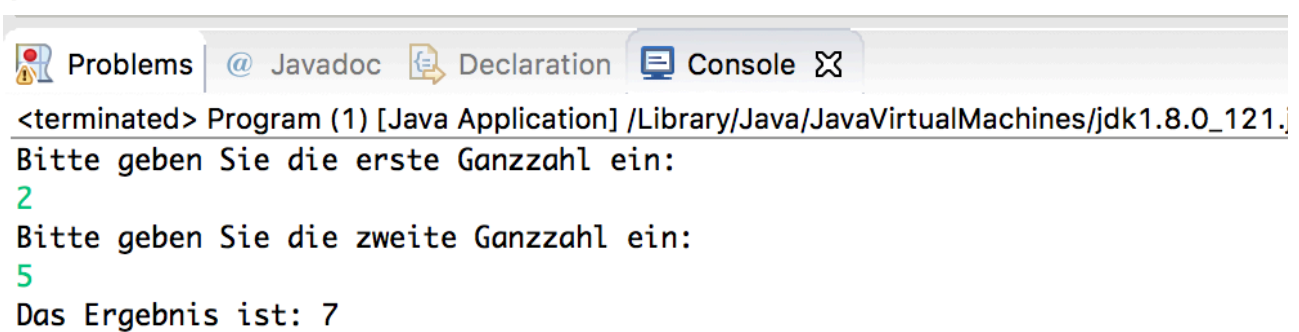
Vorteile von Methoden:

- bessere Strukturierung des Programms
- kleine Unterprogramme, welche bereits geschriebenen Code mit nur einem Befehl erneut ausführen können

Der Aufbau einer Methode

- um den Aufbau von Methoden verstehen zu können, sehen wir uns zunächst mal ein Beispielprogramm an
- in diesem Beispielprogramm schreiben wir den kompletten Code zunächst in die main-Methode

```
public static void main(String[] args) {  
  
    int number1;  
    int number2;  
    int result;  
  
    Scanner sc = new Scanner(System.in);  
  
    System.out.println("Bitte geben Sie die erste Ganzzahl ein: ");  
    number1 = sc.nextInt();  
  
    System.out.println("Bitte geben Sie die zweite Ganzzahl ein: ");  
    number2 = sc.nextInt();  
  
    result = number1 + number2;  
  
    System.out.println("Das Ergebnis ist: " + result);  
  
}
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121...  
Bitte geben Sie die erste Ganzzahl ein:  
2  
Bitte geben Sie die zweite Ganzzahl ein:  
5  
Das Ergebnis ist: 7
```

- dieses Programm möchten wir nun etwas umbauen
- und zwar soll der gesamte Code aus der main-Methode nun in eine extra Methode ausgelagert werden
- das ganze funktioniert folgendermaßen:

```

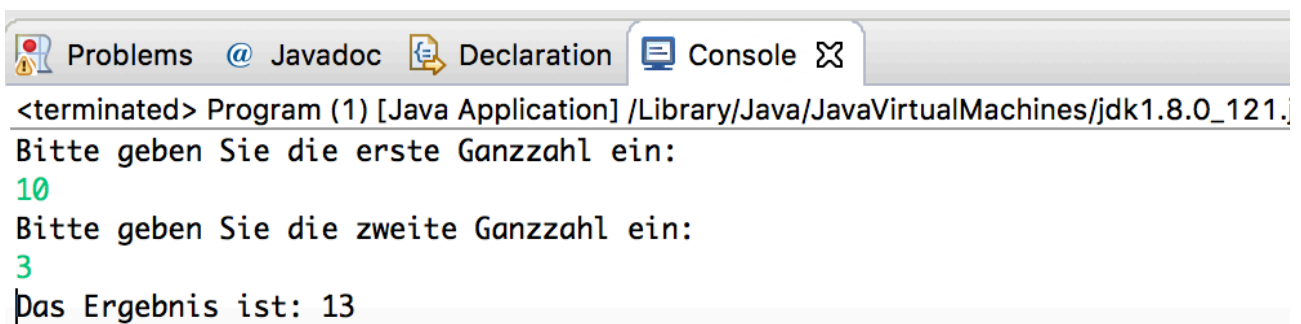
public static void main(String[] args) {
    addierer();
}

public static void addierer() {
    int number1;
    int number2;
    int result;
    Scanner sc = new Scanner(System.in);

    System.out.println("Bitte geben Sie die erste Ganzzahl ein: ");
    number1 = sc.nextInt();
    System.out.println("Bitte geben Sie die zweite Ganzzahl ein: ");
    number2 = sc.nextInt();

    result = number1 + number2;
    System.out.println("Das Ergebnis ist: " + result);
}

```



Problems Javadoc Declaration Console

 <terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121...

 Bitte geben Sie die erste Ganzzahl ein:

 10

 Bitte geben Sie die zweite Ganzzahl ein:

 3

 Das Ergebnis ist: 13

-nun haben wir uns neben der main-Methode eine weitere Methode namens addierer erstellt und den gesamten Code der main-Methode in diese neue addierer-Methode ausgelagert
 -in der main-Methode muss diese addierer-Methode dann nur noch aufgerufen werden
 => Methodenaufruf erfolgt über Bezeichner + Übergabeparameter (in diesem Fall gibt es keine Übergabeparameter, weshalb das runde Klammerpaar leer ist)

Allgemeiner Aufbau einer Methode:

- Methode besteht aus Zugriffsmodifizierer (in diesem Fall public static)
- einem Datentyp (in diesem Fall void)
- dem Bezeichner (in diesem Fall addierer)
- den Übergabeparametern (in diesem Fall gibt es keine Übergabeparameter, deshalb ist das runde Klammerpaar nach addierer leer)
- und dem Methodenblock (darin steht der auszuführende Code)

- Schlüsselwort void (übersetzt: leer) wird immer verwendet, wenn die Methode keinen Rückgabewert zurück gibt
- in der addierer-Methode ist das der Fall, denn dort taucht nirgendwo eine sogenannte return Anweisung auf (mit einer return Anweisung kann ein Wert zurückgegeben werden)

Variante 2: Ohne Rückgabewert, mit Übergabeparameter

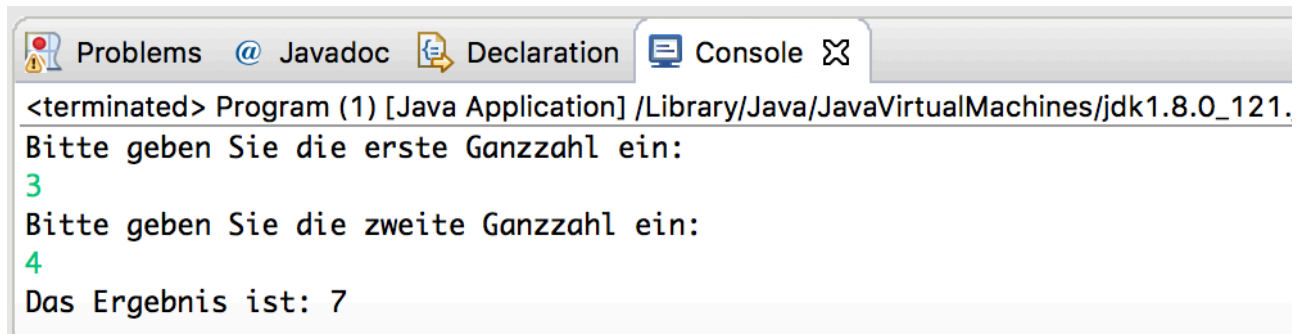
-das vorherige Beispielprogramm wird nun so umgebaut, dass wir nach wie vor keinen Rückgabewert haben aber diesmal mit sogenannten Übergabeparametern arbeiten

```
public static void main(String[] args) {
    int number1 = 0;
    int number2 = 0;
    int result = 0;
    addierer(number1, number2, result);
}

public static void addierer(int number1, int number2, int result) {
    Scanner sc = new Scanner(System.in);

    System.out.println("Bitte geben Sie die erste Ganzzahl ein: ");
    number1 = sc.nextInt();
    System.out.println("Bitte geben Sie die zweite Ganzzahl ein: ");
    number2 = sc.nextInt();

    result = number1 + number2;
    System.out.println("Das Ergebnis ist: " + result);
}
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.
Bitte geben Sie die erste Ganzzahl ein:
3
Bitte geben Sie die zweite Ganzzahl ein:
4
Das Ergebnis ist: 7
```

-die Variablen werden nun in der main-Methode deklariert und initialisiert und dann an die addierer-Methode übergeben

=> dadurch können diese Variablen jetzt auch in der addierer-Methode verwendet werden (Stichwort: Sichtbarkeit von Variablen)

-Reihenfolge der Übergabeparameter ist wichtig und muss beachtet werden

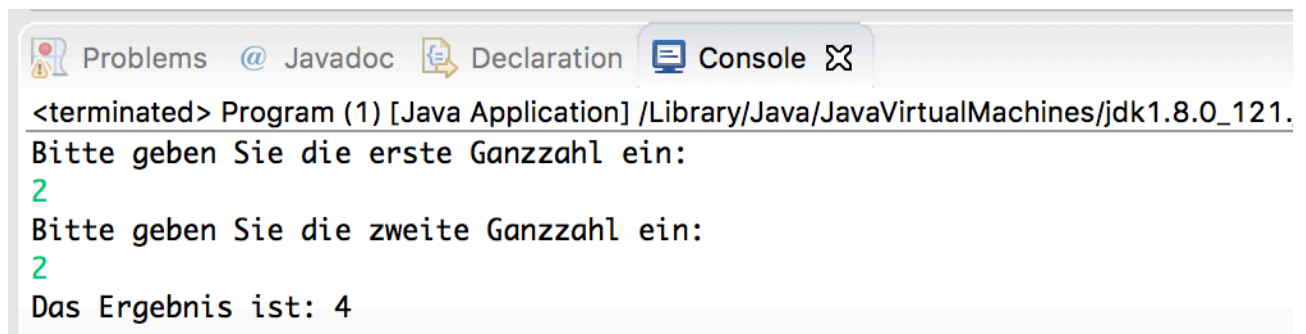
Hinweis: Bezeichner der Übergabeparameter können im Methodenkopf einfach umbenannt werden. Man ist nicht an die Bezeichner der übergebenen Variablen gebunden.

```
public static void main(String[] args) {
    int number1 = 0;
    int number2 = 0;
    int result = 0;
    addierer(number1, number2, result);
}

public static void addierer(int a, int b, int c) {
    Scanner sc = new Scanner(System.in);

    System.out.println("Bitte geben Sie die erste Ganzzahl ein: ");
    a = sc.nextInt();
    System.out.println("Bitte geben Sie die zweite Ganzzahl ein: ");
    b = sc.nextInt();

    c = a + b;
    System.out.println("Das Ergebnis ist: " + c);
}
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.
Bitte geben Sie die erste Ganzzahl ein:
2
Bitte geben Sie die zweite Ganzzahl ein:
2
Das Ergebnis ist: 4
```

Variante 3: Mit Rückgabewert, mit Übergabeparameter

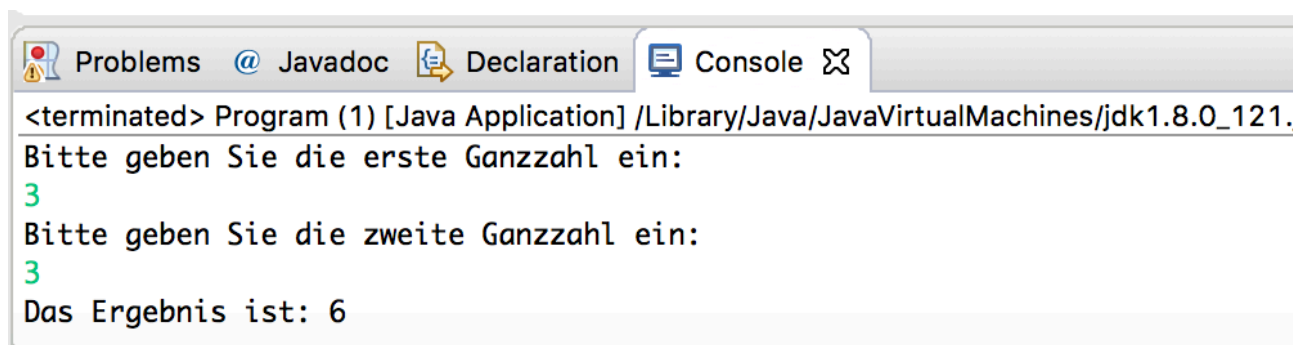
-das vorherige Beispielprogramm wird nun so umgebaut, dass wir nun einen Rückgabewert und Übergabeparameter haben

```
public static void main(String[] args) {
    int number1 = 0;
    int number2 = 0;
    int result = 0;
    result = addierer(number1, number2);
    System.out.println("Das Ergebnis ist: " + result);
}

public static int addierer(int a, int b) {
    Scanner sc = new Scanner(System.in);

    System.out.println("Bitte geben Sie die erste Ganzzahl ein: ");
    a = sc.nextInt();
    System.out.println("Bitte geben Sie die zweite Ganzzahl ein: ");
    b = sc.nextInt();

    return (a + b);
}
```



```
<terminated> Program (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_121.
Bitte geben Sie die erste Ganzzahl ein:
3
Bitte geben Sie die zweite Ganzzahl ein:
3
Das Ergebnis ist: 6
```

- Datentyp der Methode ist immer für den Datentyp des Rückgabewertes verantwortlich
- Rückgabewert kommt dort an, wo im Programm der Methodenaufruf erfolgt ist
- Rückgabewert wird in diesem Beispiel in der Variable result aufgefangen, damit der zurückgegebene Wert noch im weiteren Verlauf des Programms verwendet werden kann