

Eine Funktionalität unter einem Namen umsetzen



Methodenname: verschickeBrief

Parameter:

• Adresse, Briefmarke, Inhalt

Methodenkörper:

• Die Post verschickt den Brief

Rückgabetyp:

- void (senden und vergessen)
- alternativ einen Beleg





Methodenname: pizzaSalamiMachen

Parameter:

• Teig, Käse, Tomaten, Salami, Zeit

Methodenkörper:

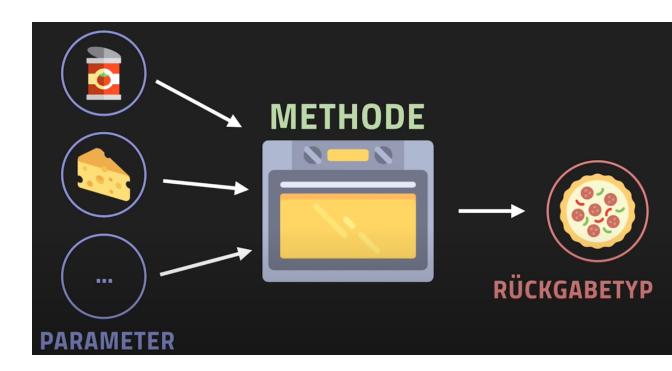
• Ofen backt die Pizza

Rückgabetyp:

• Pizza Salami 😲



Quelle: Kompezentnachweis Thema 3 einer Lehrnenden, https://www.youtube.com/watch?v=oSDtCcDXcTM



Verantwortung einer Sache

- Jemand (Methode), macht eine Sache gut und übernimmt die Arbeit für alle
- Der Methodennamen ist klar, das jeder weiss was geschieht

Beispiel Post

- Die Post ist gut in Briefe verschicken
- Sie Verschickt die Briefe für alle

Beispiel Ofen

- Der Ofen kann gut backen
- Backt die Pizza

Nutzen

- Eine Programmlogik wird nur einmal programmiert
- Eine Programmlogik kann an verschiedenen Orten verwendet werden
- Ändert sich die Logik wird diese überall geändert
- Der Methodennamen beschreibt direkt den Quellcode
- Kleine Methoden machen den Quellcode übersichtlicher

Code-Beispiel / Struktur

```
import mytools.StdInput;
public class Beispiel {
    public static void main(String[] args) {
        int userInput = readNumberInRange(40, 60); // Verwendung
        System.out.println("Sie haben " + userInput + " eingegeben!");
    private static int readNumberInRange(int min, int max) // Kopf / Signatur
        // Methodenkörper / Methoden-Body
        int userInput;
        do {
            System.out.print("Geben Sie eine Zahl zwischen " + min + " und " + max + " ein: ");
            userInput = StdInput.readInt();
        } while (min > userInput || userInput > max);
        return userInput;
```

Verwendung (Gelb)

Die Methode wird in einer Anweisung aufgerufen

Methoden Kopf / Signatur (Blau)

Definiert **wie** die Methode aufgerufen werden muss

Methodenkörper (Grün)

Definiert was die Methode genau macht

Code-Beispiel / Flow

```
1 import mytools.StdInput;
                                                                                          √ der "Flow" beim
    public class Beispiel {
                            der Zurückgegebene Wert vom Methodenaufruf
                                                                                          Rechts nach Link
                              wird in der Variable `userInput` gespeichert.
        public static void main(String[] args) {
             int userInput 4 eadNumberInRange(40, 60) 1 Methodenaufruf
             System.out.println("Sie haben " + userInput + " eingegeben!");
                                      'userInput' wird auf die Konsole ausgegeben
        private static int readNumberInRange(int min, int max) // Kopf / Signatur
10
11
12
             // Methodenkörper / Methoden-Body
                                                                 Übergebene Parameter
                                                                  min = 40, max = 60
13
             int userInput;  <--</pre>
14
            | do {
                 System.out.print("Geben Sie eine Zahl zwischen " + min + " und " + max + " ein: ");
15
                 userInput = StdInput.readInt();
16
          while (min > userInput || userInput > max); Solange bis der eingegebene
Wert zwischen 40 und 60 ist
17
18
          3 return userInput;
         } der `userInput` wird zurückgegeben "return"
19
20
21 }
```

- 1. Methodenaufruf, springt in die Methode
- 2. Schlaufe wird mehrfach wiederholt
- 3. return Wert wird ausgegeben
- 4. Der return Wert wird der Variable zugewiesen
- 5. Die Variable wird ausgegeben