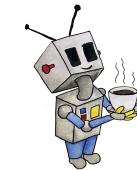


Methoden mit Rückgabewerten

openHPI-Java-Team

Hasso-Plattner-Institut



```
1 class Robot{  
2     void speak(){  
3         System.out.println("Hallo");  
4     }  
5 }
```

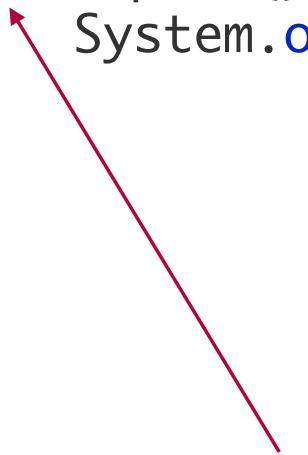
Rückgabetypen

- Datentyp oder **void**
- Ein Rückgabetyp muss immer bei der Methodendeklaration angegeben werden

Beispiel: keine Rückgabe



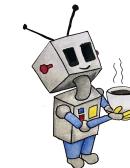
```
1 class Robot{  
2     void speak(){  
3         System.out.println("Hallo");  
4     }  
5 }
```



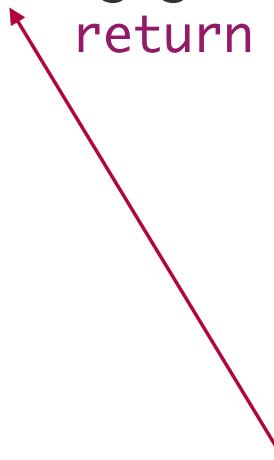
Wiederholung: void

- Bedeutet, dass die Methode nichts zurück gibt

Beispiel: Rückgabe von Strings



```
1 class Robot{  
2     String getName(){  
3         return "Robin";  
4     }  
5 }
```



Rückgabetyp String

- Bedeutet, dass die Methode einen String zurückgibt
- **return** gibt diesen String zurück (in diesem Fall "Robin")

return

```
<Rückgabetyp> <Methodename>(){  
    //mehr Quellcode  
    return <Rückgabewert>;  
}
```



- Rückgabewert muss dem Rückgabetyp entsprechen
- **return** gibt den Rückgabewert an die aufrufende Methode zurück

Verwendung von Rückgabewerten



```
1 class Robot{
2     String getName(){
3         return "Robin";
4     }
5 }
6 class Story{
7     static void main(String[] args){
8         Robot robin = new Robot();
9         String robotName = robin.getName();
10        //mehr Quellcode
11    }
12 }
```

- Nutzen Zuweisung um den Rückgabewert zu nutzen
- Können anschließend `robotName` wie gewohnt weiter verwenden

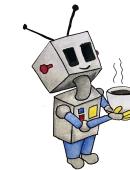
Verwendung von Rückgabewerten



```
1 class Robot{  
2     String getName(){  
3         return "Robin";  
4     }  
5 }  
6 class Story{  
7     static void main(String[] args){  
8         Robot robin = new Robot();  
9         String robotName = robin.getName();  
10        System.out.println(robotName);  
11    }  
12 }
```

- Benutzen Zuweisung um die Rückgabe der Methode zu speichern
- Können anschließend die Variable **robotName** wie gewohnt weiter verwenden

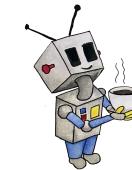
Verwendung von Rückgabewerten



```
1 class Robot{  
2     String getName(){  
3         return "Robin";  
4     }  
5 }  
6 class Story{  
7     static void main(String[] args){  
8         Robot robin = new Robot();  
9         String robotName = robin.getName();  
10        System.out.println(robotName);  
11    }  
12 }
```

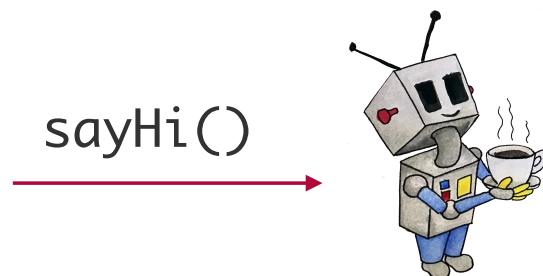
Ausgabe:
Robin

Ein etwas komplexeres Beispiel (1/3)

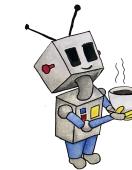


```
1 class Story{  
2     public static void main(String[] args){  
3         Robot robo1 = new Robot();  
4         robo1.sayHi();  
5     }  
6 }
```

- Aufruf der Methode `sayHi()` auf `robo1`



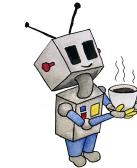
Ein etwas komplexeres Beispiel (2/3)



```
1 class Robot{  
2     String name = "Robin"; ←  
3     void sayHi(){ ←  
4         String me = getName(); ←  
5         System.out.println("Hi, ich bin " + me);  
6     }  
7     String getName(){ ←  
8         return name; ←  
9     }  
10 }
```

1. Aufruf von sayHi() von außen
2. Aufruf von getName()
 - a) Zugriff auf das Attribut **name** der Klasse Robot
 - b) Der Rückgabewert ("Robin") vom Typ String wird mit **return** zurückgegeben
3. Dieser wird der Variable **me** zugewiesen

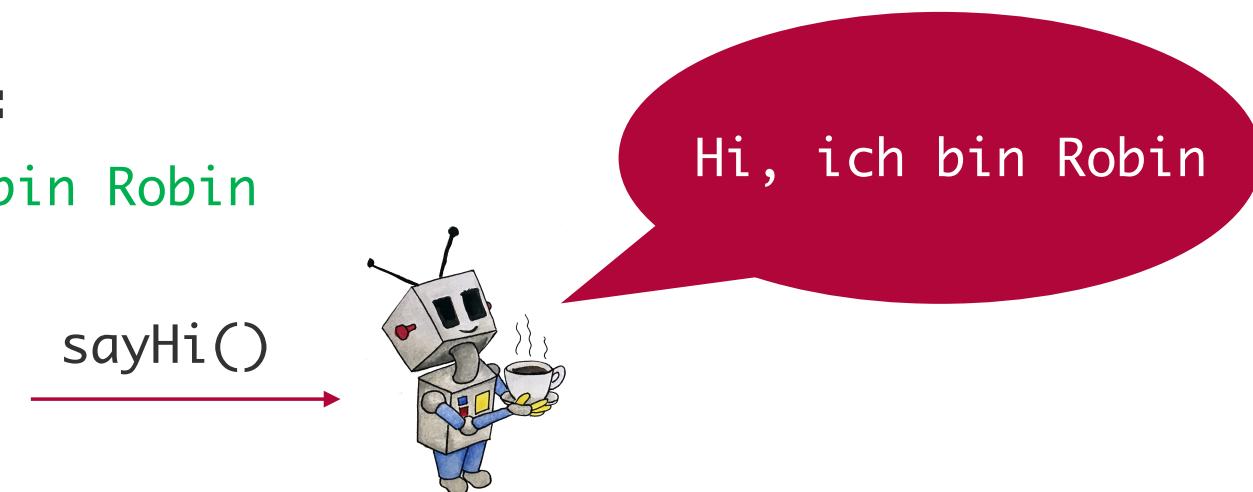
Ein etwas komplexeres Beispiel (3/3)



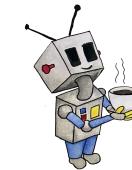
```
1 class Story{  
2     public static void main(String[] args){  
3         Robot robo1 = new Robot();  
4         robo1.sayHi();  
5     }  
6 }
```

Ausgabe:

Hi, ich bin Robin

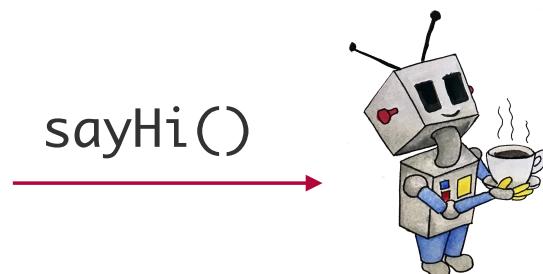


Ein noch komplexeres Beispiel (1/3)

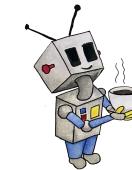


```
1 class Story{  
2     public static void main(String[] args){  
3         Robot robo1 = new Robot();  
4         System.out.println(robo1.sayHi());  
5     }  
6 }
```

- Aufruf von `sayHi()` auf `robo1`



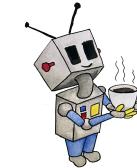
Ein noch komplexeres Beispiel (2/3)



```
1 class Robot{  
2     String name = "Robin";  
3     String sayHi(){ ←  
4         return "Hi, ich bin " + getName();  
5     }  
6     String getName(){ ←  
7         return name;  
8     }  
9 }
```

1. Aufruf von `getName()`
2. `getName()` gibt den Wert des Attributs `name` zurück
 - a) Der Rückgabewert ("Robin") vom Typ `String` wird mit `return` zurückgegeben
3. `sayHi()` gibt den konatenierten Satz zurück

Ein noch komplexeres Beispiel (3/3)



```
1 class Story{  
2     public static void main(String[] args){  
3         Robot robo1 = new Robot();  
4         System.out.println(robo1.sayHi());  
5     }  
6 }
```

Ausgabe:

Hi, ich bin Robin

