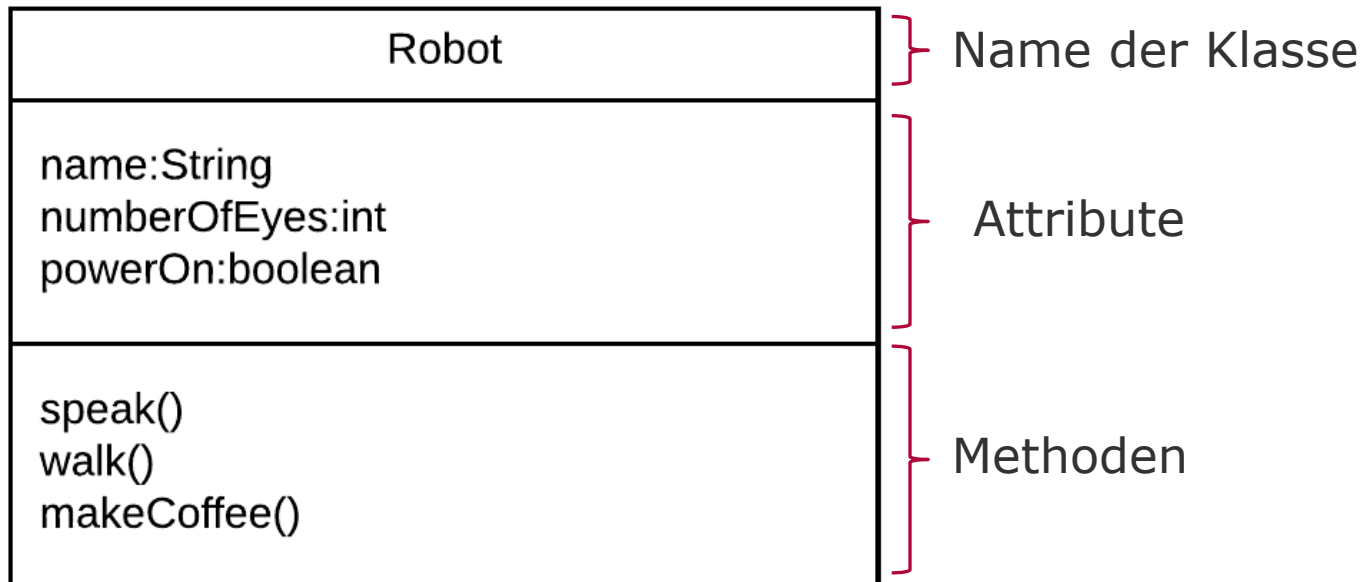


Methoden

openHPI-Java-Team
Hasso-Plattner-Institut



- Spezifizieren Operationen für alle Objekte einer Klasse
- Methodendefinitionen erfolgen **immer** innerhalb von Klassen
- Methoden definieren Verhalten





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

Methoden

- Syntax: <Rückgabetyp> <methodName>(){}
■ In den Codeblock zwischen { und } schreiben wir Anweisungen
 - Diese werden ausgeführt, wenn die Methode aufgerufen wird



Beispiel: leere Rückgabe

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

void

- Bedeutet, dass die Methode nichts zurück gibt
- "leere Rückgabe"



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

1. Programm startet in main()
2. Instanzieren Objekt `robin`
3. Aufruf von `speak()` auf dem Objekt `robin` (der Klasse `Robot`)



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

Allgemeine Form: `objectName.methodName();`

Ausgabe:

Hallo



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

- Der Methodenaufruf von `speak()` erfolgt aus der Klasse `Story` auf dem Objekt der Klasse `Robot`
 - also außerhalb der Klasse in der `speak()` definiert wurde



```
1 class Robot {  
2     void sayHelloWorld() { ←  
3         sayHello(); ←  
4         System.out.println("Welt");  
5     }  
6     void sayHello() { ←  
7         System.out.println("Hallo");  
8     }  
9 }
```

1. Aufruf von `sayHelloWorld()` von außen
2. `sayHelloWorld()` ruft `sayHello()` auf
3. Nach Ausführen von `sayHello()` wird der Rest von `sayHelloWorld()` ausgeführt



```
1 class Robot {  
2     void sayHelloWorld() {  
3         sayHello();  
4         System.out.println("Welt");  
5     }  
6     void sayHello() {  
7         System.out.println("Hallo");  
8     }  
9 }
```

Allgemein: `methodName()`;
ruft Methoden innerhalb derselben Klasse auf

Ausgabe:

Hallo

Welt