# Klassen in Groovy

Der Aufbau einer Klasse in Groovy wird hier am einfachen Beispiel der Klasse Mensch gezeigt. Eine Klasse wird in Groovy durch das Schlüsselwort **`class`** eingeleitet. Alles was zu der Klasse gehört, befindet sich dann zwischen zwei geschweiften Klammern.

```
1 class Mensch {
2 }
3
4 martin = new Mensch()
```

Dieses ist die einfachste Form der Klasse und es lässt sich von ihr auch direkt ein Objekt erzeugen. Innerhalb dieser Klasse werden als erstes die Attribute angeben. Groovy bietet hier den Vorteil an, dass zu jedem Attribut automatisch eine `set` und `get` Methode erzeugt wird, mit der man dem Attribut einen Wert zuweisen bzw. den Wert abfragen kann. Ein Attribut wird durch das Schlüsselwort **`def`** eingeleitet, worauf der Bezeichner des Attributs folgt.

```
1 class Mensch {
2   def name
3   def alter
4 }
5
6 martin = new Mensch()
7 martin.setName("Martin")
8 println(marting.getName())
```

Die Methoden werden unterhalb der Attribute in den Rumpf der Klasse geschrieben. Auch diese beginnen mit dem Schlüsselwort **`def`**, gefolgt vom Bezeichner der Methode. Was innerhalb der Methode gemacht werden soll, folgt in geschweiften Klammern. Will man in einer Methode auf Attribute oder andere Methoden der Klasse zugreifen, so wird ein **`this.`** davor gesetzt, um zu zeigen, dass es eigene Methoden bzw. Attribute sind.

```
1 class Mensch {
2   //Platzhalter für Attribute
3
4   def erhoeheAlter() {
5     this.alter = this.alter + 1
6   }
7 }
```

Wie auch im Objekt- und Klassendiagramm, sind in Groovy die Methoden anhand ihre runden Klammern nach dem Bezeichner zu erkennen. Sollen Werte als Parameter an die Methode übergeben werden, so müssen diese zwischen den runden Klammern angegeben sein. Mehrere Parameter werden dabei durch Kommata getrennt.

```
1 def sageEtwas(s) {
2   println("Ich␣sage␣" + s)
3 }
```

Wenn eine Methode eine Rückgabe liefern soll, so wird diese an der passenden Stelle mit dem Schlüsselwort **return** angegeben. Alles was im Programmcode in Zeilen nach dem **return** steht, wird nicht mehr ausgeführt, da die Rückgabe automatisch die Methode beendet.

```
1 def frageName() {
2   return "Mein␣Name␣ist␣" + this.name
3 }
```

Eine besondere Methode einer Klasse ist der Konstruktor. Er ist daran zu erkennen, dass er den gleichen Bezeichner hat, wie die Klasse selber. Aufgerufen wird der Konstruktor immer dann, wenn von der Klasse eine Objekt erzeugt wird. Mit ihm werden z. B. die Startwerte des Objekts gesetzt. Hat der Konstruktor Parameter, so müssen diese entsprechend auch bei der Erzeugung des Objekts mit angegeben werden. Beim Aufschreiben der Klasse wird der Konstruktor nach den Attribute und vor allen anderen Methoden geschrieben.

```
1 class Mensch {
2   //Attribute
3
4   def Mensch(name, alter) {
5     this.name = name
6     this.alter = alter
7   }
8
9   //Methoden
10 }
11
12 martin = new Mensch("Martin", 15)
```

Das vollständige Beispiel, mit Attributen, dem Konstruktor und verschiedenen Methoden könnte dann wie folgt aussehen:

```
1 class Mensch {
2   def name
3   def alter
4
5   def Mensch(name, alter) {
6     this.name = name
7     this.alter = alter
8   }
9
10  def erhoeheAlter() {
11    this.alter = this.alter + 1
12  }
13
14  def sageEtwas(s) {
15    println("Ich␣sage␣" + s)
16  }
17
18  def frageName() {
19    return "Mein␣Name␣ist␣" + this
          .name
20  }
21 }
22
23 martin = new Mensch("Martin", 15)
24 martin.sageEtwas("Hallo")
25 println(martin.frageName())
26 martin.erhoeheAlter()
27 println("Martin␣ist␣" + martin.
      getAlter() + "␣Jahre␣alt")
```