

### Downloads zum Buch

Unter der Webadresse *buch.daniel-braun.com* findest du:

- Links zu allen Downloads, die du benötigst
- Alle Plugins, die du im Rahmen des Buches programmieren wirst, falls du den Code nicht aus dem Buch abtippen möchtest

Mein besonderer Dank gilt Karl-Heinz Barzen, der den Entstehungsprozess dieses Buches unermüdlich mit zahlreichen hilfreichen Kommentaren und Anmerkungen begleitet und damit einen wichtigen Beitrag dazu geleistet hat, dass dieses Buch möglichst verständlich und einsteigerfreundlich wird.

Nun wünsche ich dir aber vor allem viel Spaß beim Lesen, Programmieren und Entdecken!

Daniel Braun



## Java

Ob bewusst oder unbewusst, eine der wichtigsten Entscheidungen, die man auf dem Weg zum Programmierer zu treffen hat, hast du bereits getroffen: welche Programmiersprache du lernen möchtest. Mit diesem Buch hast du dich nämlich für die Programmiersprache Java entschieden. Bevor wir uns aber mit den Besonderheiten von Java beschäftigen und damit, warum es eine gute Entscheidung ist, Java zu lernen, soll es zunächst um die Frage gehen, was Programmiersprachen eigentlich sind und warum sie benötigt werden.

### 1.1 Programmiersprachen

Beim Programmieren geht es im Wesentlichen darum, dass der Programmierer dem Computer eine bestimmte Aufgabe gibt, die dieser erledigen soll. Damit er das kann, braucht der Computer eine genaue Handlungsvorschrift, die auch *Algorithmus* genannt wird. Auch im Alltag begegnen uns oft Handlungsvorschriften, zum Beispiel in Form eines Rezepts:

1. 250 Gramm Mehl in eine Schüssel geben
2. 500 Milliliter Milch dazugeben
3. 2 Eier hinzugeben
4. Mit einer Prise Salz würzen
5. Umrühren

Fertig ist der Crêpes-Teig! Damit eine Handlungsvorschrift korrekt ausgeführt werden kann, müssen sich beide Seiten auf eine gemeinsame Sprache einigen. Wenn dir jemand ein Rezept auf Chinesisch gibt, kannst du vermutlich nicht viel damit anfangen.

Computer »sprechen« in Einsen und Nullen, also in einer Sprache, mit der Menschen nicht besonders gut umgehen können. Unsere Sprache wiederum, egal ob es sich um Deutsch, Englisch oder Chinesisch handelt, ist für den Computer viel zu ungenau. Nehmen wir zum Beispiel den Satz: »Da vorne ist eine Bank.« Obwohl es sich dabei um einen vollkommen korrekten deutschen Satz handelt, ist doch nicht eindeutig klar, was mit dem Satz eigentlich gemeint ist. Steht da vorne eine Parkbank, auf die man sich setzen kann, oder ist dort die Filiale einer Bank, auf der man Geld einzahlen und abheben kann?

Es wäre ein recht kostspieliger Fehler, wenn dein Computer beim Online-Shopping aus Versehen die Deutsche Bank statt einer Bank für den Garten kauft.

Algorithmen müssen deshalb nicht nur Handlungsvorschriften sein, sie müssen *eindeutige Handlungsvorschriften* sein. Auch mit Begriffen wie »eine Prise« kann ein Computer wenig anfangen. Aus diesem Grund nutzen wir Programmiersprachen, denn sie ermöglichen es uns, eindeutige Handlungsvorschriften festzulegen. Und obwohl sie auf den ersten Blick recht kompliziert scheinen, können wir sie doch leichter lernen als eine Sprache aus Nullen und Einsen.

Damit der Computer die Programmiersprache auch versteht, muss sie aber zunächst übersetzt werden, in die sogenannte *Maschinensprache*. Diese Übersetzung findet durch ein Programm statt, das *Compiler* genannt wird. Das Ergebnis sind dann sogenannte *Binärdateien*, die vom Computer ausgeführt werden können. Diese Binärdateien bestehen, wie in Abbildung 1.1 gezeigt, nur aus Nullen und Einsen.

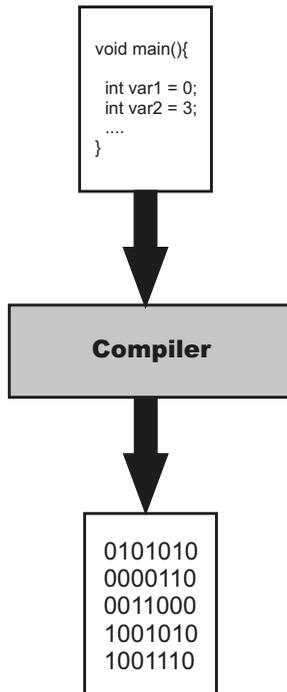


Abbildung 1.1: Funktionsweise eines Compilers

Der einfache Satz »Das ist ein Test.« wird so zum Beispiel zu einer 136 Zeichen langen Kette aus Nullen und Einsen, die du in Listing 1.1 sehen kannst.

```
01000100 01100001 01110011 00100000 01101001 01110011 01110100 00100000
01100101 01101001 01101110 00100000 01010100 01100101 01110011 01110100
00101110
```

**Listing 1.1:** Binärcodierung von »Das ist ein Test.«

### Merke

- Ein *Algorithmus* ist eine eindeutige Handlungsvorschrift.
- Der *Compiler* übersetzt Programmiersprache in Maschinensprache.
- Eine *Binärdatei* besteht aus Nullen und Einsen.

## 1.2 Besonderheiten von Java

Verschiedene Programmiersprachen haben verschiedene Vor- und Nachteile. Einige sind besonders leicht zu erlernen, wie zum Beispiel Python, andere, wie zum Beispiel C, sind besonders für zeitkritische Anwendungen geeignet, also Anwendungen, bei denen es auf schnelle Reaktionszeiten ankommt, und wieder andere sind besonders universell einsetzbar, wie zum Beispiel Java. Die eine »richtige« oder »beste« Programmiersprache gibt es daher nicht – je nach Anwendungsfall kann der Einsatz einer anderen Programmiersprache sinnvoll sein.

Der Hauptgrund, warum wir Java zum Programmieren unserer Plugins verwenden, ist, dass sowohl Minecraft selbst als auch der Minecraft-Server in Java programmiert sind. Außerdem können Java-Programme, im Gegensatz zu vielen in anderen Programmiersprachen geschriebenen Programmen, problemlos auf allen gängigen Betriebssystemen ausgeführt werden, also insbesondere auf Windows, GNU/Linux und macOS.

Damit das möglich ist, funktioniert der Java-Compiler anders als andere Compiler. Er wandelt die Programmiersprache nicht sofort in Maschinencode um, sondern zunächst in den sogenannten *Java-Bytecode*. Dieser ist ein Zwischenschritt zwischen der für Menschen gut lesbaren Programmiersprache und dem für den Computer gut lesbaren Maschinencode. Erst die sogenannte *Java Virtual Machine (JVM)* wandelt das Programm in Maschinencode um.

Der Vorteil: Statt jedes Programm in Maschinencode für jedes Betriebssystem, also zum Beispiel Windows, macOS und GNU/Linux übersetzen zu müssen, muss nur ein Programm, nämlich die Java Virtual Machine, für jedes Betriebssystem übersetzt werden – und das bedeutet deutlich weniger Aufwand.