

Grundkurs Programmieren in Java

» Hier geht's
direkt
zum Buch

DIE LESEPROBE

Einleitung

Was eint Shakespeare, Goethe, Tolstoi, Fontane und viele mehr? Sie alle haben Weltliteratur hervorgebracht und dabei auch über tragische Liebesbeziehungen geschrieben. Romeo und Julia, Die Leiden des jungen Werthers, Anna Karenina oder Effi Briest sind wohl vielen ein Begriff. Und wer selbst schon einmal unglücklich verliebt war, kann die beschriebenen Geschichten umso besser nachvollziehen. Doch was hat das alles mit diesem Buch zu tun und wie kann es Ihnen in einer solchen Situation helfen? Die traurige Wahrheit ist: Die Gemeinsamkeiten enden beim Wort Buch, und auf schwierige Lebensfragen haben wir leider auch keine Antwort. Auch hilft Ihnen dieses Buch in der Regel nicht, Ihren Schwarm zu beeindrucken oder unliebsamen Wettbewerb loszuwerden. Aber im Ernst: Wozu ist das Buch dann zu gebrauchen? Die folgenden Seiten verraten es Ihnen.

Java – mehr als nur kalter Kaffee?

Seit dem Einzug von Internet und World Wide Web in das öffentliche Leben surfen, mailen und chatten Milliarden von Menschen täglich in der virtuellen Welt. Und seitdem wir mit dem Smartphone auch unterwegs dauerhaft online sein können, ist ein Leben ohne das Internet in vielen Bereichen kaum vorstellbar geworden. Wer nicht mitmacht, gilt als altmodisch oder gar abgehängt. Interessanterweise setzte diese Entwicklung aber nicht erst im letzten Jahrzehnt ein, wie man zunächst denken könnte, sondern schon sehr viel früher, als Anfang der 1990er-Jahre das World Wide Web das Licht der Welt erblickte und selbst so ehrwürdige Organisationen wie Pizza Hut oder der Vatikan auf einmal im Netz der Netze vertreten waren. Oder wussten Sie, dass man bereits seit 1994 Pizza online bestellen kann und 1995 die Webseite des Vatikans online ging?¹

Im selben Jahr erschien auch Java, das dem damaligen Zeitgeist entsprechend als *Programmiersprache des Internets* vermarktet wurde. Schnell entwickelte sich Java deshalb zu einer der populärsten Programmiersprachen überhaupt, die Jahr für

¹ Zur Sicherheit wollen wir hier allerdings darauf hinweisen, dass das Internet und das World Wide Web zwei unterschiedliche Dinge sind. Das Internet, dessen Ursprünge sich bis ins Jahr 1969 zurückverfolgen lassen, bildet das weltumspannende Netz, über welches so unterschiedliche Dienste wie E-Mail, Video-Streaming oder eben das World Wide Web – also das Betrachten von Webseiten im Browser – bezogen werden können.

Jahr weit oben im TIOBE-Index liegt. Und das, obwohl das Sprachkonzept eigentlich nur wenig Neues bietet. Viel mehr orientierte man sich an der mehr als ein Jahrzehnt älteren Programmiersprache C++, vereinfachte diese so sehr, dass sie auch von weniger geübten Programmierer/-innen genutzt werden konnte, und legte darüber hinaus eine Klassenbibliothek bei, die keine Wünsche offen lässt. Über die Jahre haben sich Programmiersprache und Klassenbibliothek natürlich gewaltig weiterentwickelt und zu einer vollwertigen Konkurrenz zu den anderen gängigen Sprachen gemausert. Datenbank- oder Netzwerkzugriffe, anspruchsvolle Grafikanwendungen, Spieleprogrammierung – alles ist möglich. Und gerade im heute so aktuellen Bereich „Verteilte Anwendungsentwicklung“ bietet Java ein breites Spektrum an Möglichkeiten. Mit wenigen Programmzeilen gelingt es, Anwendungen zu schreiben, die das Internet bzw. das World Wide Web nutzen. Doch auch jede andere Art von Anwendung ist möglich, inklusive ausgefeilter grafischer Benutzungsoberflächen – und das sogar plattformunabhängig für alle gängigen Betriebssysteme.

Dies erklärt das große Interesse, das der Sprache Java seither entgegengebracht wird. Bedenkt man die Anzahl von Buchveröffentlichungen, Zeitschriftenbeiträgen, Webseiten, Foren, Blogs und Social Media Posts zum Thema, so wird der erfolgreiche Weg, den die Sprache Java hinter sich hat, offensichtlich. Aber auch im kommerziellen Bereich ist Java nicht mehr wegzudenken, denn die Produktpalette der meisten großen Softwarehäuser weist mittlerweile eine Java-Schiene auf. Und wer heute auch nur mit einem Smartphone telefoniert, kommt häufig (bewusst oder unbewusst) mit Java in Berührung. Für Sie als Leserin oder Leser dieses Buchs bedeutet das jedenfalls, dass es sicherlich kein Fehler ist, Erfahrung in der Programmierung mit Java zu haben.

Java für Anfänger – das Konzept dieses Buches

Wie schreibt man ein Buch über das Programmieren für absolute Neulinge, wenn man selbst seit vielen Jahren programmiert? Folgende Antwort haben wir darauf gefunden: Das Buch soll die grundlegenden Konzepte der Programmierung sowie die Programmiersprache Java möglichst vollständig und korrekt beschreiben, diese aber leicht verständlich vermitteln. Maßstab für die Qualität des Buches ist deshalb, dass es sich optimal zum Selbststudium sowie als Begleitmaterial für Vorlesungen im Bachelor-Studium einsetzen lässt, ohne Vorkenntnisse in den Bereichen Programmieren, Programmiersprachen oder Informatik vorauszusetzen. Vor allem unsere langjährige Erfahrung in der studentischen Programmierausbildung in einführenden und weiterführenden Kursen des Instituts AIFB² am KIT³ sowie des Zentrums für Wirtschaftsinformatik an der DHBW⁴ Karlsruhe sollen Ihnen hier zu Gute kommen.

² Institut für Angewandte Informatik und Formale Beschreibungsverfahren

³ Karlsruher Institut für Technologie

⁴ Duale Hochschule Baden-Württemberg

Beispielsweise gibt es gewisse Erfahrungswerte darüber, welche Themen gerade Neulingen besondere Probleme bereiten. Daher rührt der Entschluss, nicht sofort in fortgeschrittene Themen wie die für Java so charakteristische objektorientierte Programmierung einzusteigen, sondern erst einen grundlegenden Wortschatz zu erarbeiten und ein Verständnis dafür zu entwickeln, wie sich Programmieren in Java „anfühlt“. Vergleichen Sie das mit dem Erlernen einer gesprochenen Sprache wie zum Beispiel Spanisch. Auch hier muss man anhand vieler praktischer Beispiele zunächst die Sprache kennenlernen und durch regelmäßige Übung festigen. Dadurch entsteht eine gewisse Routine und Intuition, die es dann auch ermöglicht, tiefer in die Regeln guten Schreibstils oder die Besonderheiten der spanischen Lyrik einzusteigen.

Im ersten Teil des Buches geht es daher überwiegend darum, zunächst einmal die grundlegenden Konzepte der Programmierung zu verinnerlichen und ganz allgemein „algorithmisches Denken“ zu lernen. Die darauffolgenden Teile II bis VI bauen dann darauf auf und führen Sie immer mehr in fortgeschrittene Themen wie Objektorientierung, funktionale Programmierung oder die Entwicklung grafischer Oberflächen ein. Alle Kapitel sind hierfür mit Übungsaufgaben ausgestattet, die Sie zum besseren Verständnis bearbeiten sollten. Denn: *Man lernt eine Sprache nur, indem man sie spricht!*

Selbstredend können und wollen wir nicht auf jedes noch so kleine Detail eingehen, auch wenn Sie sicher feststellen werden, dass wir die Themen alles andere als oberflächlich behandeln. Wir möchten Sie daher bereits an dieser Stelle dazu ermutigen, regelmäßig einen Blick in die sogenannte API-Spezifikation⁵ der Klassenbibliothek [42] zu werfen und sich auch mit anderen Informationsquellen im Internet vertraut zu machen – nicht zuletzt, weil wir im „Programmieralltag“ von einem routinierten Umgang mit diesen Werkzeugen nur profitieren können. Eine kleine Starthilfe hierzu bietet das Kapitel in Anhang B.

Zusatzmaterial und Kontakt zu den Autoren

Alle Leserinnen und Leser sind herzlich eingeladen, die Autoren über Fehler und Unklarheiten zu informieren. Wenn eine Passage unverständlich war, sollte sie zur Zufriedenheit künftiger Leserinnen und Leser anders formuliert werden. Wenn Sie in dieser Hinsicht also Fehlermeldungen, Anregungen oder Fragen haben, können Sie über unsere Website

<http://www.grundkurs-java.de/>

Kontakt mit den Autoren aufnehmen. Dort finden Sie auch alle Beispielprogramme aus dem Buch, Lösungshinweise zu den Übungsaufgaben und ergänzende Materialien zum Download sowie Literaturhinweise, eine Liste eventueller Feh-

⁵ API steht für Application Programming Interface, die Programmierschnittstelle für eine Klasse, ein Paket oder eine ganze Klassenbibliothek.

ler im Buch und deren Korrekturen. Dozentinnen und Dozenten, die das Material dieses Buchs oder sogar Teile unserer Vorlesungsfolien für eigene Vorlesungen nutzen möchten, sollten sich mit uns in Verbindung setzen.

Im Literaturverzeichnis haben wir sowohl Bücher als auch Internet-Links angegeben, die aus unserer Sicht als weiterführende Literatur geeignet sind und neben Java im Speziellen auch einige weitere Themenbereiche wie zum Beispiel Informatik, Algorithmen, Nachschlagewerke, Softwaretechnik, Objektorientierung und Modellierung einbeziehen.

Verwendete Schreibweisen

Wir verwenden *Kursivschrift* zur Betonung bestimmter Wörter und **Fettschrift** zur Kennzeichnung von Begriffen, die im entsprechenden Abschnitt erstmals auftauchen und definiert bzw. erklärt werden. Im laufenden Text wird *Maschinenschrift* für Bezeichner verwendet, die in Java vordefiniert sind oder in Programmbeispielen eingeführt und benutzt werden, während reservierte Wörter (Schlüsselwörter, Wortsymbole), die in Java eine vordefinierte, unveränderbar festgelegte Bedeutung haben, in **fetter Maschinenschrift** gesetzt sind. Beide Schriften kommen auch in den vom Text abgesetzten Listings und Bildschirmausgaben von Programmen zum Einsatz. Java-Programme sind teilweise ohne und teilweise mit führenden Zeilennummern abgedruckt. Solche Zeilennummern sind dabei lediglich als Orientierungshilfe gedacht und natürlich *kein* Bestandteil des Java-Programms.

Literaturverweise auf Bücher und Web-Links werden stets in der Form [nr] mit der Nummer nr des entsprechenden Eintrags im Literaturverzeichnis angegeben. Im Stichwortverzeichnis verweisen fettgedruckte Seitenzahlen auf die Stellen im Buch, an denen die jeweiligen Begriffe eingeführt bzw. definiert werden.

Über die Autoren

- *Prof. Dr. Dietmar Ratz* ist Studiengangsleiter Wirtschaftsinformatik an der Dualen Hochschule Baden-Württemberg (DHBW) Karlsruhe und lehrt auch am Karlsruher Institut für Technologie (KIT).
- *Dipl.-Wirtsch.-Inf. (DH) Dennis Schulmeister-Zimolong* arbeitet als akademischer Mitarbeiter an der Dualen Hochschule Baden-Württemberg (DHBW) Karlsruhe sowie als Produktmanager bei der SOA People AG, Karlsruhe.
- *Prof. Dr. Detlef Seese* ist ehemaliger Professor für Angewandte Informatik am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) des Karlsruher Instituts für Technologie (KIT).
- *Dipl.-Wi.-Ing. Jan Wiesenberger* ist geschäftsführender Vorstand des FZI Forschungszentrum Informatik in Karlsruhe und Hauptgesellschafter des IT-Dienstleisters m+ps.

Teil I

Einstieg in das Programmieren in Java

Sicher können Sie es kaum erwarten, Ihr erstes Programm einzugeben und zu sehen, wie der Computer dieses tatsächlich ausführt. Denn von diesem Moment an öffnet sich eine völlig neue Welt für Sie – die Welt des Programmierens. In den folgenden Kapiteln lernen Sie daher alle Grundlagen, die Sie dafür benötigen.

Kapitel 1

Einige Grundbegriffe aus der Welt des Programmierens

Der Fahrkartenautomat in der Straßenbahn, der Kaffeeautomat in der Mensa, der digitale Sprachassistent zu Hause oder der Laptop im Hörsaal bzw. bei der Arbeit. Digitale Geräte sind aus unserem Alltag nicht mehr wegzudenken, denn überall werden inzwischen Daten verarbeitet oder Geräte gesteuert. Schätzte man Anfang der 1950er Jahre noch, dass ein Dutzend „Elektronengehirne“ den Bedarf der ganzen Erde decken könne, findet man heute weit mehr Computer in jedem privaten Haushalt.¹ Grundkenntnisse der Softwaretechnik und Informatik sind in der heutigen Zeit daher unerlässlich, wenn wir diese Geräte nicht als schiere Zauberei oder einfach nur wahnsinnig komplex wahrnehmen wollen. Wer werden in diesem Kapitel deshalb erst ein paar grundlegende Begriffe der Informatik klären und Sie in die Funktionsweise und Terminologie der Informatik entführen. Wenn Sie wollen, können Sie diese dann im Buch [] noch weiter vertiefen.

1.1 Computer, Software, Informatik und das Internet

Als **Computer** (deutsch: Rechner) bezeichnet man ein technisches Gerät, das schnell und meist zuverlässig nicht nur rechnen, sondern allgemein Daten bzw. Informationen automatisch verarbeiten und speichern (aufbewahren) kann. Im Unterschied zu einem normalen **Automaten** wie z. B. einem Getränkeautomaten, der nur festgelegte Aktionen ausführt, können wir einem Computer die Vorschrift, nach der er arbeiten soll, jeweils neu vorgeben. Beispiele für solche Arbeitsvorschriften oder Handlungsanleitungen wären die Regeln für Kreditberechnungen unserer Bank oder die Anleitung zur Steuerung von Signalanlagen für unsere Modelleisenbahn. In der Fachsprache heißt eine solche Handlungsanlei-

¹ Bedenken Sie hierbei, dass auch so unscheinbare Geräte wie Waschmaschine, Mikrowellenherd, Fernbedienung oder Funkmaus ein eingebettetes Computersystem beinhalten.

tung **Algorithmus**. Um dem Computer einen Algorithmus in einer präzisen Form mitzuteilen, muss man diesen als ein **Programm** (eine spezielle Handlungsanweisung, die für den Computer verständlich ist) formulieren. Der Computer, zusammen mit seinen Programmen, wird auch als **Computersystem** bezeichnet.

Generell gesehen setzt sich ein Computersystem zusammen aus den materiellen Teilen, der sogenannten **Hardware**, und den immateriellen Teilen, der sogenannten **Software**. Unter dem Begriff Software versteht man nicht nur Programme, sondern auch zugehörige Daten und Dokumentationen. Man unterscheidet dabei zwischen **Systemsoftware** und **Anwendungssoftware**. Zur erstgenannten Gruppe zählt man das Betriebssystem, Compiler, Datenbanken, Kommunikationsprogramme und spezielle Dienstprogramme. Beispiele für Anwendungssoftware, also Software, die Aufgaben der Anwender/-innen löst, sind Textverarbeitungs-, Tabellenkalkulations- und Zeichenprogramme.

Ein Computer setzt sich zusammen aus der **Zentraleinheit** und den **Peripheriegeräten**. Die Zentraleinheit besteht aus dem **Prozessor** (er führt die Programme aus), einem **ROM** (Read-Only Memory, deutsch: Nurlesespeicher), in dem elementare Systemprogramme, wie der nach dem Einschalten ausgeführte Code, abgelegt sind, und dem **Arbeitsspeicher** (in ihm werden Programme und Daten, die zur momentanen Programmausführung benötigt werden, kurzfristig gespeichert). Der Arbeitsspeicher wird häufig auch als **RAM** (Random Access Memory, deutsch: Direktzugriffsspeicher) bezeichnet. Unter dem Begriff Peripheriegeräte fasst man **Eingabegeräte** wie Tastatur und Maus, **Ausgabegeräte** wie Bildschirm und Drucker sowie **externen Speicher** wie Festplatten, DVD-ROM-Speicher oder Memorysticks zusammen.

Im Vergleich zum externen Speicher bietet der Arbeitsspeicher eines Rechners einen wesentlich schnelleren (lesenden und schreibenden) Zugriff, wobei sein Inhalt nach dem Ausschalten des Rechners wieder verloren geht. Die kleinste Einheit des Arbeitsspeichers wird **Speicherzelle** genannt. Sie besitzt einen Namen (Adresse) und kann einen Wert speichern. Externe Speichermedien hingegen bieten meist eine wesentlich höhere Speicherkapazität und dienen der langfristigen Aufbewahrung von Programmen und Informationen (Daten). Diese werden in sogenannten **Dateien** (englisch: **files**) abgelegt. Solche Dateien können wir uns beispielsweise wie ein Sammelalbum vorstellen, in das wir unsere Daten (die Bilder) ablegen (einkleben). Um eine Datei anzulegen, müssen wir ihr einen Namen geben. Dieser wird zusammen mit Angaben zur Größe der Datei in das vom Betriebssystem auf dem Speichermedium angelegte **Dateisystem** geschrieben. Dabei können wir auch mehrere Dateien zu sogenannten **Ordnern** bzw. **Verzeichnissen** (englisch: **folder** oder **directory**) zusammenfassen (ähnlich wie wir unsere Sammelalben in Ordnern abheften), die selbst wieder Namen bekommen und hierarchisch in weiteren Ordnern zusammengefasst werden können. Dateinamen werden häufig unter Verwendung eines Punktes (.) in zwei Teile gegliedert, den eigentlichen Namen und die sogenannte **Dateinamenerweiterung**, die meist den Typ der Datei angibt (z. B. `txt` für Text-Dateien, `java` für Java-Dateien oder `docx` für Dateien eines Textverarbeitungssystems).

Heutzutage sind so gut wie alle Computersysteme über Datenleitungen oder per Funk miteinander **vernetzt**, d. h. sie können untereinander Informationen austauschen. Man spricht dann von einem **Netz** oder Netzwerk (englisch: **net** oder **web**) und speziell von einem **Intranet**, wenn die Vernetzung innerhalb einer Organisation oder Firma erfolgt, oder vom **Internet**, dem weltweiten Computer-Netzwerk. Die Computer- und Softwaretechnik nahm seit ihren Kindertagen eine rasante Entwicklung von den Rechnern mit Transistortechnik und integrierten Schaltungen in Großrechnern und Minicomputern über die ersten Mikroprozessoren und Personal Computer Ende der 1970er Jahre, den PCs, Workstations und Supercomputer der 1980er und 1990er, bis hin zu den heute allgegenwärtigen Laptops und mobilen Endgeräten wie Tablets, Smartphones und Wearables. Recht bald entwickelte sich eine (seit 1960) eigenständige Wissenschaftsdisziplin, die **Informatik**. Sie beschäftigt sich mit der theoretischen Analyse und Konzeption, aber auch mit der konkreten Realisierung von Computersystemen in den Bereichen Hardware, Software, Organisationsstrukturen und Anwendung.

1.2 Was heißt Programmieren?

Wie bereits erwähnt, ist ein Programm nichts anderes als ein in einer **Programmiersprache** formulierter Algorithmus. Diese Sprache erlaubt es den Anwenderinnen und Anwendern (Programmiererinnen und Programmierern), mit dem Computer zu „sprechen“ und ihm Anweisungen zu geben. Dieses „Sprechen“ kann nun auf unterschiedliche Arten erfolgen. Man kann zum Beispiel eine Sprache verwenden, die der Computer (genau genommen der Prozessor) direkt „versteht“. Man nennt sie **Maschinensprache**. Da die Notation mit Nullen und Einsen nur schwer lesbar ist, verwendet man zur Formulierung von maschinennahen Programmen meist eine **Assemblersprache**, in der jedem binären Maschinenco-

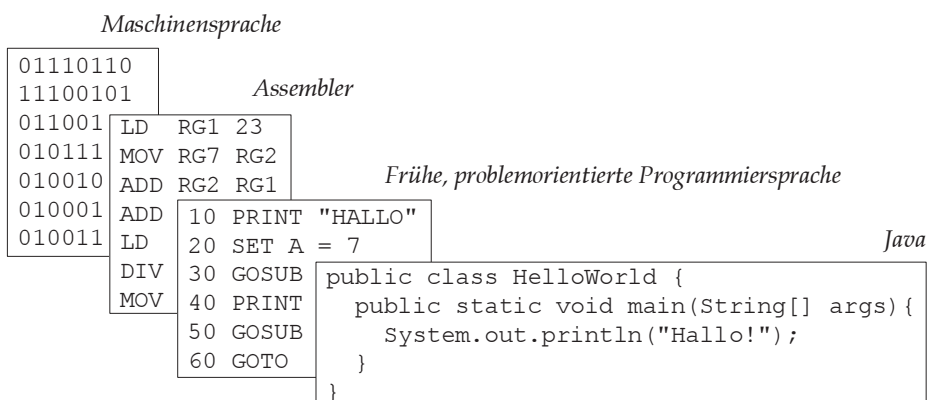


Abbildung 1.1: Programmiersprachen im Vergleich

de ein entsprechender, aus Buchstaben und Ziffern bestehender Assemblercode entspricht. Programme in solchen Sprachen kann der Prozessor direkt ausführen, doch ist man dabei sehr vom Prozessortyp abhängig. Alternativ kann auch eine benutzernahe bzw. problemnahe Programmiersprache zum Einsatz kommen, die man als **höhere Programmiersprache** oder auch **problemorientierte Programmiersprache** bezeichnet. In diesem Fall benötigt man allerdings einen **Übersetzer** (eine spezielle Software), der die Sätze der höheren Programmiersprache in die Maschinensprache oder eine spezielle „Zwischensprache“ überführt. Ein Übersetzer, der problemorientierte Programme in maschinennahe Programme transformiert, wird **Compiler** genannt. Werden Programme nicht vollständig übersetzt und später ausgeführt, sondern Anweisung für Anweisung übersetzt und unmittelbar ausgeführt, nutzt man einen **Interpreter**.

Traditionelle höhere Programmiersprachen benötigen für jeden Prozessortyp einen Compiler, der das jeweilige sogenannte **Quellprogramm** (geschrieben in der Programmiersprache) in ein sogenanntes **Zielprogramm** (ausführbar auf dem Prozessor) übersetzt (compiliert). Unser Programm muss daher für jeden Rechner- bzw. Prozessortyp übersetzt werden. Bei der Entwicklung der Sprache Java hingegen wurde das Ziel angestrebt, plattformunabhängig zu sein, indem man nur einen Compiler für alle Plattformen benötigt. Dieser Compiler übersetzt das Quellprogramm (auch Quellcode oder Quelltext genannt) in den sogenannten **Java-Bytecode**, der unabhängig von einem bestimmten Prozessor ist, jedoch nicht unmittelbar ausgeführt werden kann. Erst der **Java-Interpreter** analysiert den erzeugten Bytecode schrittweise und führt ihn aus. Der Bytecode ist also portabel (auf unterschiedliche Plattformen übertragbar) und sozusagen für eine Art virtu-

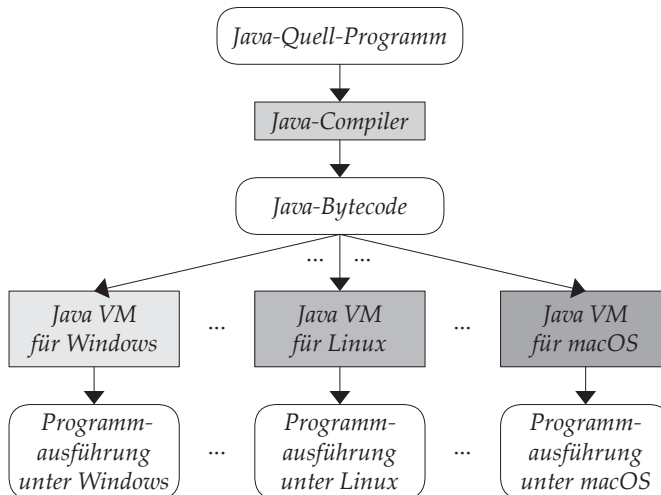


Abbildung 1.2: Vom Java-Quellprogramm zum ausführbaren Programm