1 Einleitung

Bevor es mit den Programmierthemen losgeht, möchte ich Ihnen dieses Buch vorstellen. Ich beginne damit, warum dieses Buch entstanden ist und wie es Ihnen hoffentlich helfen kann, ein noch besserer Java-Entwickler zu werden. Danach folgt eine Gliederung des Inhalts, damit Sie sich gut im Buch zurechtfinden.

1.1 Über dieses Buch

1.1.1 Motivation

Mein Ziel war es, ein Buch zu schreiben, wie ich es mir selbst immer als Hilfe gewünscht habe. Die hier vorgestellten Hinweise und Techniken sollen Sie auf Ihrem Weg vom engagierten Hobbyprogrammierer oder Berufseinsteiger zum erfahrenen Softwareentwickler begleiten. Dieser Weg ist ohne Anleitung gewöhnlich steinig und mit einige Mühen, Irrwegen und Problemen verbunden. Einige dieser leidvollen Erfahrungen möchte ich Ihnen ersparen. Aber auch erfahreneren Softwareentwicklern soll dieses Buch die Möglichkeit geben, über die im täglichen Einsatz lieb gewonnenen Gewohnheiten nachzudenken und die eine oder andere davon zu ändern, um die Produktivität weiter zu steigern. Mein Wunsch ist, dass sich nach Lektüre des Buchs für Sie die Ingenieurdisziplin der Softwareentwicklung mit der Kunst des Programmierens verbindet und Dinge auf einmal einfach so auf Anhieb funktionieren. Das ist etwas ganz anderes, als vor jedem Gang in die Testabteilung Magenschmerzen zu bekommen.

Sowohl der Berufseinstieg als auch die tägliche Arbeit können manchmal frustrierend sein. Meiner Meinung nach soll Softwareentwicklung aber Spaß und Freude bereiten, denn nur so können wir exzellente Resultate erzielen. In der Praxis besteht jedoch die Gefahr, in die Fettnäpfchen zu treten, die im Sourcecode hinterlassen wurden. Dies geschieht meistens dadurch, dass die existierende Lösung softwaretechnisch umständlich oder schlecht implementiert ist und/oder nicht bis zu Ende durchdacht wurde. Der Sourcecode ist dann häufig schwierig wart- und erweiterbar. Manchmal bereitet bereits das Auffinden der Stelle, an der man Modifikationen durchführen sollte, Probleme.

Dieses Buch soll aufzeigen, wie man die zuvor beschriebene »Altlasten«-Falle vermeidet oder aus ihr herauskommt und endlich Sourcecode schreiben kann und darf, der leicht zu lesen ist und in dem es Spaß macht, Erweiterungen zu realisieren. Grundlage dafür ist, dass wir uns einen Grundstock an Verhaltensweisen und an Wissen aneignen.

1.1.2 Was leistet dieses Buch und was nicht?

Wieso noch ein Buch über Java-Programmierung? Tatsächlich kann man sich diese Frage stellen, wo es doch unzählige Bücher zu diesem Thema gibt. Viele davon sind einführende Bücher, die häufig nur kurz die APIs anhand simpler Beispiele vorstellen. Die andere große Masse der Java-Literatur beschäftigt sich mit speziellen Themen, die für den »erfahrenen Einsteiger« bereits zu komplex geschrieben und in denen zu wenig erklärt ist. Genau hier setzt dieses Buch an und wagt den Spagat, den Leser nach der Lektüre einführender Bücher abzuholen und so weit zu begleiten, dass er mit einem guten Verständnis die Spezialliteratur lesen und gewinnbringend einsetzen kann.

Ziel dieses Buchs ist es, dem Leser fundierte Kenntnisse in Java und einigen praxisrelevanten Themenbereichen, unter anderem dem Collections-Framework und im Bereich Multithreading, zu vermitteln. Es werden vertiefende Blicke auf die zugrunde liegenden Details geworfen, um nach Lektüre des Buchs professionelle Programme schreiben zu können. Wie bereits angedeutet, bietet dieses Buch keinen Einstieg in die Sprache selbst, sondern es wird einiges an Wissen vorausgesetzt. In den einleitenden Grundlagenkapiteln zu einer professionellen Arbeitsumgebung, über objektorientiertes Design und Java sowie über die funktionale Programmierung mit Lambdas wird allerdings die Basis für das Verständnis der Folgekapitel geschaffen.

In diesem Buch versuche ich, einen lockeren Schreibstil zu verwenden und nur an den Stellen formal zu werden, wo dies wirklich wichtig ist, etwa bei der Einhaltung von Methodenkontrakten. Da der Fokus dieses Buchs auf dem praktischen Nutzen und dem guten Verständnis von Konzepten liegt, werden neben APIs auch häufig vereinfachte Beispiele aus der realen Welt vorgestellt. Die meisten der abgebildeten Listings stehen als kompilierbare und lauffähige Programme auf der Webseite zum Buch zum Download bereit. Im Buch selbst werden aus Platzgründen und zugunsten einer besseren Übersichtlichkeit in der Regel nur die wichtigen Passagen abgedruckt.

1.1.3 Wie und was soll mithilfe des Buchs gelernt werden?

Dieses Buch zeigt und erklärt einige in der Praxis bewährte Ansätze, Vorgehens- und Verhaltensweisen, ohne dabei alle Themengebiete bis in kleinste Detail auszuleuchten. Wichtiges Hintergrundwissen wird jedoch bei Bedarf vermittelt. Es wird der pragmatische Weg gegangen und bevorzugt die in der täglichen Praxis relevanten Themen vorgestellt. Sollte ein Thema bei Ihnen besonderes Interesse wecken und Sie weitere Informationen wünschen, so finden sich in den meisten Kapiteln Hinweise auf weiterführende Literatur. Dies ist im Prinzip auch schon der erste Tipp: ***Lesen Sie viele Bücher und schaffen Sie sich damit eine breite Wissensbasis.*** Ich zitiere hier aus Jon Bentleys Buch »Perlen der Programmierkunst« [3]: ***»Im Stadium des Entwurfsprozesses ist es unschätzbar, die einschlägige Literatur zu kennen.«***

Diesem Hinweis kann ich mich nur anschließen und möchte Ihnen hier speziell einige – meiner Meinung nach – ganz besondere Bücher ans Herz legen und empfehle ausdrücklich, diese Bücher begleitend oder ergänzend zu diesem Buch zu lesen:

- »**SCJP – Sun Certified Programmer & Developer for Java 2**« [72] – Die Vorbereitung zur SCJP-Zertifizierung wird mit all seinen Fallstricken und kniffligen Details auf unterhaltsame Weise von Kathy Sierra und Bert Bates aufbereitet.
- »**The Java Programming Language**« [2] – Ein unglaublich gutes Buch von Ken Arnold, James Gosling und David Holmes über die Sprache Java, das detailreich, präzise und dabei angenehm verständlich zu lesen ist.
- »**Effective Java**« [5] und [6] – Dieses Buch von Joshua Bloch habe ich auf der Java One 2001 in San Francisco gekauft und es hat mein Denken und Programmieren in Java stark beeinflusst. Mittlerweile existiert eine zweite Auflage, die auf JDK 6 aktualisiert wurde.
- »**Entwurfsmuster**« [24] – Das Standardwerk der sogenannten »Gang of Four« (Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides) habe ich 1998 kennengelernt und mit großem Interesse gelesen und gewinnbringend eingesetzt. Die vorgestellten Ideen sind beim Entwurf guter Software enorm hilfreich.
- »**Head First Design Patterns**« [22] – Dieses Buch einer anderen »Gang of Four« (Eric Freeman, Elizabeth Freeman, Kathy Sierra und Bert Bates) lässt Entwurfsmuster als ein unterhaltsames Thema erscheinen und erleichtert den Einstieg.
- »**Refactoring**« [21] – Einige Tricks und Kniffe zur Verbesserung von Sourcecode lernt man von Kollegen oder durch Erfahrung. Martin Fowler fasst dieses Wissen in dem genannten Buch zusammen und stellt ein systematisches Vorgehen zur Sourcecode-Transformation vor.
- »**Refactoring to Patterns**« [47] – Dieses Buch von Joshua Kerievsky verknüpft die Ideen von Refactorings mit denen zu Entwurfsmustern.
- »**Code Craft: The Practice of Writing Excellent Code**« [27] – Ein sehr lesenswertes Buch von Pete Goodlife, das diverse gute Hinweise gibt, wie man exzellenten Sourcecode schreiben kann, der zudem (nahezu) fehlerfrei, gut testbar sowie einfach zu warten ist.

Lesen hilft uns bereits, aber nur durch Übung und Einsatz in der Praxis können wir unsere Fähigkeiten verbessern. Weil ein Buch jedoch nicht interaktiv ist, werde ich bevorzugt eine schrittweise Vorstellung der jeweiligen Themen vornehmen, wobei zum Teil auch bewusst zunächst ein Irrweg gezeigt wird. Anhand der vorgestellten Korrekturen erkennt man dann die Vorteile viel deutlicher, als wenn nur eine reine Präsentation der Lösung erfolgen würde. Mit dieser Darstellungsweise hoffe ich, dass Sie sich ein paar gute Gewohnheiten antrainieren. Das fängt mit scheinbar einfachen Dingen wie der Vergabe von sinnvollen Namen für Variablen, Methoden und Klassen an und endet in der Verwendung von problemangepassten Entwurfsmustern. Anfangs erfordert dies erfahrungsgemäß ein wenig Fleiß, Einarbeitung, Disziplin und eventuell sogar etwas Überwindung. Daher werde ich bei der Vorstellung einer Technik jeweils sowohl auf die Vorteile als auch die Nachteile (wenn vorhanden) eingehen.

1.1.4 Wer sollte dieses Buch lesen?

Dieses Buch konzentriert sich auf Java als Programmiersprache – allerdings benötigen Sie bereits einige Erfahrung mit Java, um die Beispiele sowie die beschriebenen Tücken nachvollziehen zu können und möglichst viel von den Tipps und Tricks in diesem Buch zu profitieren. Wenn Sie dieses Buch in den Händen halten, gehe ich also davon aus, dass Sie sich schon (etwas) mit Java auseinandergesetzt haben.

Das Buch richtet sich im Speziellen an zwei Zielgruppen: Zum einen sind dies engagierte Hobbyprogrammierer, Informatikstudenten oder Berufseinsteiger, die von Anfang an lernen wollen, wie man professionell Software schreibt. Zum anderen sind dies erfahrenere Softwareentwickler, die ihr Wissen in einigen fortgeschritteneren Themen komplettieren wollen und vermehrt Priorität auf sauberes Design legen oder Coding Conventions, Codereviews und Unit-Testen bei der Arbeit etablieren wollen.

Abhängig vom Kenntnisstand zu Beginn der Lektüre starten Entwickler mit Erfahrung bei Teil II oder Teil III des Buchs und können die dort vorgestellten Techniken sofort gewinnbringend in der Praxis einsetzen. Lesern mit noch relativ wenig Erfahrung empfehle ich, den ersten Teil konzentriert und vollständig durchzuarbeiten, um sich eine gute Basis zu verschaffen. Dadurch wird das Verständnis der später vorgestellten Themen erleichtert, denn die nachfolgenden Teile des Buchs setzen die Kenntnis dieser Basis voraus und das Niveau nimmt ständig zu. Ziel ist es, nach Lektüre des Buchs den Einstieg in die professionelle Softwareentwicklung mit Java erreicht zu haben und viele dazu erforderliche Techniken sicher zu beherrschen. Das Buch ist daher mit diversen Praxistipps gespickt, mit denen Sie auf interessante Hintergrundinformationen oder auf mögliche Probleme hingewiesen werden und die wie folgt in den Text integriert sind:

Tipp: Praxistipp

In derart formatierten Kästen finden sich im späteren Verlauf des Buchs immer wieder einige wissenswerte Tipps und ergänzende Hinweise zum eigentlichen Text.

1.2 Aufbau des Buchs

Der Aufbau des Buchs gliedert sich in mehrere Teile. Folgende Aufzählung konkretisiert die dort vorgestellten Themen:

- **Teil I »Java-Grundlagen, Analyse und Design«** – Dieser Teil legt die Grundlagen für einen guten Softwareentwurf, in dem sowohl auf objektorientiertes Design als auch auf eine produktive Arbeitsumgebung mit den richtigen Hilfsmitteln eingegangen wird. Teil I beginnt in Kapitel 2 mit der Vorstellung einer sinnvoll ausgestatteten Arbeitsumgebung, die beim Entwickeln von professionellen Programmen hilft. Anschließend wird in Kapitel 3 das Thema objektorientiertes Design beschrieben. Damit sind die Grundlagen für einen professionellen, objektorientierten Softwareentwurf gelegt und die Vorbereitungen zum Implementieren getroffen.

Kapitel 4 stellt dann grundlegende Java-Sprachelemente vor, die zum Verständnis der Beispiele in den folgenden Kapiteln notwendig sind. Abgerundet wird der erste Teil mit Kapitel 5 durch eine Vorstellung der mit Java 8 eingeführten Lambda-Ausdrücke, die die funktionale Programmierung mit Java ermöglichen.

- **Teil II »Bausteine stabiler Java-Applikationen«** – Der zweite Teil beschäftigt sich mit Komponenten und Bausteinen stabiler Java-Applikationen. Es werden wichtige Kenntnisse fundamentaler Java-APIs vermittelt, aber auch fortgeschrittene Java-Techniken behandelt. Zunächst stellt Kapitel 6 das Thema Collections vor, um eine effiziente Wahl von Datenstrukturen zur Speicherung und Verwaltung von Daten zu ermöglichen. Im Anschluss gehe ich in Kapitel 7 auf Bulk Operation on Collections und das Stream-API als wesentliche Erweiterungen aus Java 8 ein. In Kapitel 8 beschäftigen wir uns mit der Erstellung wiederverwendbarer Softwarebausteine. Es ist wichtig, das Rad nicht immer neu zu erfinden, sondern auf einer stabilen Basis aufzubauen. Ein weiterer wichtiger Baustein beim professionellen Programmieren ist Multithreading. Kapitel 9 gibt eine fundierte Einführung und stellt auch fortgeschrittenere Themen, wie das Java-Memory-Modell und die Parallelverarbeitung mit Thread-Pools, vor. Weitere fortgeschrittenere Themen, etwa Reflection, Annotations und Garbage Collection, werden in Kapitel 10 behandelt. Schließlich schauen wir uns in Kapitel 11 die Verbesserungen bei der Datumsverarbeitung mit dem in Java 8 ergänzten Date and Time API an. Damit besitzen Sie dann schon eine gute Wissensbasis, aber was wären die meisten Programme ohne eine grafische Benutzeroberfläche? Weil das so wichtig ist, geht Kapitel 12 auf dieses Thema detailliert ein. Das Thema Internationalisierung und damit die Besonderheiten, die bei der Unterstützung verschiedener Länder und Sprachen zu beachten sind, werden in Kapitel 13 thematisiert.
- **Teil III »Neuerungen in Java 9«** – In diesem Teil gebe ich in zwei Kapiteln einen Überblick zu Java 9. Kapitel 14 startet mit Erweiterungen in der Syntax. Zudem werden diverse Ergänzungen in verschiedenen APIs vorgestellt. Die umfangreichste und bedeutendste Neuerung ist wohl die Modularisierung: Sowohl das JDK als auch eigene Programme lassen sich damit strukturieren und in Module untergliedern. Kapitel 15 behandelt dieses Themengebiet.

Nach der Lektüre dieser drei Teile sind Sie programmiertechnisch fit und bereit für das Schreiben eigener Anwendungen mit komplexeren Aufgabenstellungen. Auf dem Weg zu guter Software werden Sie aber vermutlich über das eine oder andere Problem stolpern. Wie Sie mögliche Fallstricke erkennen und beheben, ist Thema der folgenden Teile.

- **Teil IV »Fallstricke und Lösungen im Praxisalltag«** – Der vierte Teil beschreibt anhand von Beispielen mögliche Probleme aus dem Praxisalltag und stellt passende Lösungen vor. Auf diese Weise wird ein tieferes Verständnis für einen guten Softwareentwurf erlangt. Kapitel 16 betrachtet zunächst ausführlich mögliche Programmierprobleme, sogenannte »Bad Smells«. Diese werden analysiert und Lösungs-

möglichkeiten dazu aufgezeigt. Diverse Umbaumaßnahmen werden in Kapitel 17 als Refactorings vorgestellt. Kapitel 18 rundet diesen Teil mit der Präsentation einiger für den Softwareentwurf wichtiger Lösungsideen, sogenannter Entwurfsmuster, ab. Diese sorgen zum einen dafür, ein Problem auf eine dokumentierte Art zu lösen, und zum anderen, Missverständnisse zu vermeiden, da die Entwickler eine eigene, gemeinsame Designsprache sprechen.

- **Teil V »Qualitätssicherungsmaßnahmen«** – Qualitätssicherung ist für gute Software elementar wichtig und wird in Teil V vorgestellt. In den Bad Smells gewonnene Erkenntnisse werden in Kapitel 19 zu einem Regelwerk beim Programmieren, sogenannten Coding Conventions, zusammengefasst. Um neue Funktionalität und Programmänderungen abzusichern, schreiben wir Unit Tests. Nur durch eine breite Basis an Testfällen haben wir die Sicherheit, Änderungen ohne Nebenwirkungen auszuführen. Kapitel 20 geht detailliert darauf ein. Eine Qualitätskontrolle über Codereviews wird in Kapitel 21 thematisiert. Zu einer guten Qualität gehören aber auch nicht funktionale Anforderungen. Diese betreffen unter anderem die Performance eines Programms. In Kapitel 22 stelle ich daher einige Techniken zur Performance-Steigerung vor.
- **Anhang** – Zum besseren Verständnis der Abläufe beim Ausführen eines Java-Programms stellt Anhang A einige Grundlagen zur Java Virtual Machine vor.

1.3 Konventionen und ausführbare Programme

Verwendete Zeichensätze

Im gesamten Text gelten folgende Konventionen bezüglich der Schriftart: Der normale Text erscheint in der vorliegenden Schriftart. Dabei werden wichtige Textpassagen *kursiv* oder ***kursiv und fett*** markiert. Englische Fachbegriffe werden eingedeutscht groß geschrieben. Zusammensetzungen aus englischen und deutschen (oder eingedeutschten) Begriffen werden mit Bindestrich verbunden, z. B. Plugin-Manager. Namen von Refactorings, Bad Smells, Idiomen sowie Entwurfsmustern u. Ä. werden bei ihrer Verwendung in KAPITÄLCHEN dargestellt. Sourcecode-Listings sind in der Schrift `courier` gesetzt, um zu verdeutlichen, dass dieser Text einen Ausschnitt aus einem realen Java-Programm wiedergibt. Auch im normalen Text werden Klassen, Methoden, Konstanten und Übergabeparameter in dieser Schriftart dargestellt.