Einleitung

Software ist wichtig und wird immer wichtiger. Vielleicht haben Sie angesichts dieser Tatsache auch, so wie ich, das flaue Gefühl, dass die Qualität der Programme und die Kosten ihrer Entwicklung viel zu oft nicht den Erfordernissen entsprechen.

Es ist offensichtlich, dass die Erstellung von Software in vielerlei Richtungen entscheidend verbessert werden muss. Wenn das nicht gelingt, drohen Ausfälle mit gewaltigem Schadenspotenzial.

Berichte über – oft nach einem Update – nicht oder nur noch eingeschränkt funktionierende Bank- oder Buchungssysteme vermitteln einen kleinen Vorgeschmack darauf, was in Zukunft möglicherweise alles passieren kann.

Je mehr Software im Einsatz ist, desto bedeutender wird deren konsequente Verbesserung und Weiterentwicklung. Bereits heute entsteht ein großer Teil der Kosten nicht durch Neuentwicklungen, sondern durch die notwendigen Anpassungen.

Clean Code ist ein wichtiges Konzept, um Programme so zu schreiben, dass sie möglichst wenig Fehler enthalten und über lange Zeit stabil weiterentwickelt werden können. Jeder engagierte Entwickler sollte deshalb zumindest die wichtigsten Grundsätze kennen.

Über dieses Buch

Dieses Buch soll Ihnen die Gedankenwelt des Clean-Code-Konzepts nahebringen.

Das ist nicht mit der Darstellung einer Reihe von Regeln getan. Infolgedessen kann ich Ihnen auch nicht versprechen, dass Sie nach dem Studium dieses Buchs ein perfekter oder auch nur guter Clean-Code-Programmierer seien werden.

Ich kann Ihnen jedoch versprechen, dass Sie nach dem Lesen eine andere Sicht auf die Softwareentwicklung haben werden. Sie werden zumindest einiges besser verstehen und verständlicheren Code schreiben können. Und das ganz unabhängig davon, ob Sie das komplette Buch intensiv studiert oder nur Teile davon überflogen haben.

Softwareentwicklung kann gut mit der Expedition zu einem Ziel in schlecht erschlossenem Gelände verglichen werden. Je nachdem wie gut die Gegebenheiten bereits bekannt sind, lässt sich der erforderliche Aufwand mehr oder weniger genau vorhersagen. Je weniger Sie wissen, desto sorgfältiger muss die Unternehmung vorbereitet werden.

Mit diesem Buch will ich Ihnen helfen, sich zielstrebig auf Softwareprojekte einzustellen. Deshalb diskutiere ich nützliche handwerkliche Fertigkeiten und Verfahren ausführlich.

Daneben geht es mir aber auch um die mentale Vorbereitung. Anforderungen können sich stark unterscheiden und Schwierigkeiten beispielsweise durch den Umfang an Funktionen,

Leistungsanforderungen oder ein unzureichendes Verständnis der Anwendungsdomäne verursacht werden.

Jeder dieser Punkte erfordert ein anderes Herangehen. Diese Differenzierung wird viel zu häufig unterlassen.

Schließlich noch ein dritter Punkt: Sie lernen (hoffentlich) jeden Tag etwas dazu. Ich zeige Ihnen Wege, wie Sie das Gelernte nutzen können, um darauf aufbauend jeden Tag besser für Ihr Projekt zu arbeiten.

Mit dieser umfassenden Sicht hoffe ich, Ihnen auch dann zu helfen, wenn Sie die vorgestellten Techniken nicht oder nur stark eingeschränkt einsetzen können.

Und nicht zuletzt wünsche ich mir, dass dieses Buch auch dem einen oder anderen, der nicht selbst Code schreibt, aber eng mit IT-Entwicklungen verbunden ist, hilft, Softwareentwickler besser zu verstehen und ihnen den Freiraum zu gewähren, den sie für gute Ergebnisse brauchen.

Konventionen in diesem Buch

Wenn es konkret wird, verwende ich Java. Diese Programmiersprache ist sehr vielen Entwicklern zumindest bekannt und wird seit Jahren in zahlreichen Projekten eingesetzt. Allein aufgrund des vorhandenen Codebestands wird Java noch lange wichtig sein.

Allerdings ist Clean Code nicht an eine bestimmte Sprache gebunden. Fast alle Beispiele lassen sich ohne großen Aufwand in andere Sprachen übertragen. Und wahrscheinlich lernen Sie bei Ihren entsprechenden Versuchen sogar mehr, als wenn ich dieses Buch durch die Wiederholung von Beispielen in verschiedenen Programmiersprachen aufgebläht hätte.

Damit Sie sich gut zurechtfinden, erkläre ich Ihnen kurz die verwendeten Schriftarten und Hervorhebungen.

- ✔ In einem Buch über Code finden Sie erwartungsgemäß Quellcode-Listings. Zur Darstellung wird wie allgemein üblich eine Festbreiten-Schriftart verwendet:

  ```
  public class DasIstEinBeispiel extends Nothing {}
  ```

- ✔ Wenn im Text auf konkrete in Java definierte Namen Bezug genommen wird, werden diese ebenfalls in der Schriftart für Code dargestellt, zum Beispiel, wenn es um die Basisklasse `java.lang.Object` geht.

- ✔ Neue Begriffe werden in der Regel *kursiv* gesetzt.

  Manchmal wird die Kursivschreibung auch zur Hervorhebung benutzt.

Was Sie nicht lesen müssen

Selbstverständlich bin ich der Meinung, dass es sich lohnt, das gesamte Buch zu lesen. Aber natürlich müssen Sie das nicht.

Die Texte in Kästen sind Hintergrundinformationen, die Sie überspringen können.

Was für Sie wichtig ist und was nicht, hängt stark davon, warum Sie sich für dieses Thema interessieren und was Sie dazu bereits wissen.

Wenn Ihnen das Thema Clean Code noch relativ neu ist und Sie überhaupt erst einmal verstehen wollen, worum es dabei geht, sollten Sie mit den Kapiteln 1, 2 und 4 starten und dann mit einem Ihnen interessant erscheinenden Thema aus Teil III fortfahren.

Wenn Sie Clean Code bereits kennen, können Sie eigentlich sofort mit Teil III beginnen. Trotzdem empfehle ich Ihnen, unabhängig davon auch die Kapitel 6 und 7 zu lesen.

Für alle, die keine oder wenig Programmiererfahrung haben, sind die Kapitel 1, 2, 6, 7 und 21 geeignet, um das Verständnis für den Softwareentwicklungsprozess zu erhöhen.

Der Aufbau des Buchs ist so gewählt, dass es von Anfang bis Ende einer Linie folgt. Trotzdem ist es so geschrieben, dass Sie grundsätzlich jedes Kapitel für sich allein lesen können.

Wenn einzelne Gesichtspunkte in einem anderen Kapitel detailliert behandelt werden, finden Sie entsprechende Verweise – denen Sie folgen können oder nicht.

Wenn Sie etwas schon wissen oder es Ihnen unwichtig erscheint, überspringen Sie es – oder lesen Sie es später. Wie ich Ihnen überhaupt empfehlen möchte, es nicht beim einmaligen Durchlesen zu lassen. Ich bin ziemlich sicher, dass Sie im Lichte Ihrer gewonnenen Erfahrungen bei jedem erneuten Lesen einige Dinge besser oder erst richtig verstehen werden, die Ihnen zuvor entgangen sind.

Törichte Annahmen über die Leser

In diesem Buch dreht sich alles um Softwareentwicklung und Code. Daher sollte mindestens einer der folgenden Punkte auf Sie zutreffen.

- ✔ **Sie können programmieren.**

  Für das vollständige Verständnis ist eine gewisse Vertrautheit mit einer im Idealfall objektorientierten Programmiersprache nötig.

- ✔ **Sie wirken an IT-Projekten als Entwickler mit.**

  Praktisch erleben können Sie den Nutzen von Clean Code vor allem bei größeren und länger laufenden Softwareprojekten.

- ✔ **Sie wollen besseren Code produzieren.**

  Diese Motivation wird Ihnen ganz bestimmt helfen, schneller und gründlicher zu lernen.

- ✔ **Sie verfügen über Ausdauer und Stehvermögen.**

  Um sauberen Code zu schreiben, werden Sie möglicherweise lieb gewonnene Wege verlassen müssen. Eine solche Umstellung Ihrer Arbeitsweise ist nicht von heute auf morgen bewerkstelligt, sondern braucht Zeit und Training.

- ✔ **Sie sind eng mit der Entwicklung von Software verbunden.**

  Wenn Sie als Projektleiter, Product Owner oder in einer ähnlichen Rolle eng mit Entwicklern zusammenarbeiten, gehören Sie nicht unmittelbar zur angepeilten Zielgruppe. Sie können jedoch viel über deren Methoden und Probleme lernen und dadurch das gegenseitige Verständnis erheblich fördern. Abschnitte, die sich unmittelbar auf den Code beziehen, können Sie natürlich überspringen.

- ✔ **Sie sind bereits ein perfekter Clean-Coder.**

  Dann brauchen Sie dieses Buch eigentlich nicht. Lesen Sie es bitte trotzdem kritisch und machen Sie mich auf Fehler und Vergessenes aufmerksam. Oder – noch besser – verfassen Sie gleich »Cleaner Clean Code für Dummies«.

Wie dieses Buch aufgebaut ist

Dieses Buch besteht aus fünf Teilen.

Teil I: Das Clean-Code-Prinzip

In diesem Teil erfahren Sie die Grundgedanken des Clean-Code-Prinzips. Warum ist Code als solcher so wichtig? Was ist überhaupt guter und sauberer Code? Daneben geht es um Fragen des Abwägens zwischen konkurrierenden Zielstellungen.

Teil II: An Herausforderungen wachsen

In Teil II lernen Sie wesentliche Herausforderungen kennen, die Sie als Softwareentwickler bewältigen müssen. Der Schwerpunkt liegt dabei auf jenen Problemen, die vor dem Beginn des Codeschreibens gelöst sein sollten.

Diese Differenzierung der Problembereiche hilft Ihnen, keine unrealistischen Erwartungen bezüglich des Effekts von sauberer Programmierung zu hegen.

Teil III: Sauberen Code schreiben

Teil III zeigt Ihnen die wichtigsten Regeln zum Schreiben von sauberem Code. Dabei erfahren Sie jeweils auch, unter welchen Bedingungen eine Regel verwendet werden sollte und wann es besser ist, sie nicht anzuwenden.

Sie lernen zudem, wie Sie mögliche Fehler klassifizieren und dementsprechend sauber behandeln können.

Schließlich geht es auch noch um neuere Sprachbestandteile, und Sie sehen, wie diese ohne Beeinträchtigung der Codequalität sinnvoll eingegliedert werden können.

Teil IV: Wege zum Ziel

In Teil IV steht die praktische Umsetzung des Clean-Code-Konzepts im Mittelpunkt. Sie erhalten Tipps, wie es Ihnen gelingen kann, das Schreiben von sauberem Code in den Softwareentwicklungsprozess zu integrieren.

Teil V: Der Top-Ten-Teil

Im letzten Teil des Buchs habe ich noch ein paar nützliche Hinweise für Sie versammelt:

- ✔ Ratschläge, die Sie auf dem Weg zum Clean-Code-Entwickler vor Fehlern bewahren können und Ihnen hoffentlich helfen, das Ziel zu erreichen.
- ✔ Eine Liste von Webseiten, die Ihnen detailliertere Informationen zu den aufgeworfenen Themen und praktischen Fragen geben.

Symbole, die in diesem Buch verwendet werden

Die Glühbirne markiert einen Tipp. Falls Sie nur Wissen erwerben wollen, können Sie solche Hinweise überspringen. Wenn es Ihnen aber auch um die praktische Umsetzung geht, sollten Sie diese Tipps nicht nur einmal sorgfältig lesen.

An dieser Stelle finden Sie Hinweise, die Ihnen helfen sollen, bei der praktischen Anwendung des Dargestellten Fehler zu vermeiden.

Daher können Sie diese Texte überspringen, wenn Sie nur das Thema verstehen wollen.

Hier schränke ich die Bedeutung einzelner Wörter durch eine genauere Abgrenzung ein, um im folgenden Text Missverständnisse zu vermeiden. Solche Definitionen sollten Sie besser nicht ignorieren.

Mit dem Wegweiser kennzeichne ich zusätzliche Erläuterungen beispielweise durch Analogien aus anderen Bereichen oder kurze Beispiele. Für das Verständnis sind diese Ausführungen zwar nicht zwingend erforderlich, aber hoffentlich hilfreich.

Wenn Sie wenig Zeit haben oder das Kapitel bereits zum zweiten Mal lesen, können Sie diese Texte überspringen.

Umfangreichere Beispiele, die unabhängig vom umgebenden Text verständlich sind, werden auf diese Weise gekennzeichnet.

Sie können diese Teile beim ersten Lesen gern überspringen und eventuell später genauer studieren.

Wie es weitergeht

Ganz gleich, aus welchen Gründen Sie sich für Clean Code interessieren. Fangen Sie an zu lesen – das muss nicht auf Seite 1 sein – und versuchen Sie Ihre Erkenntnisse sinnvoll anzuwenden.

Bevor Sie nun loslegen, möchte Ihnen aber noch einen Rat geben. Den können Sie jetzt gern überspringen, aber lesen Sie ihn bitte auf jeden Fall noch, bevor Sie eventuell aufgeben:

Es gibt keinen Express auf dem Weg zum guten Clean-Code-Entwickler. Viel eher ist ein Fußmarsch vonnöten. Wie lang und wie beschwerlich dieser Marsch ist, hängt vom Ausgangspunkt und den spezifischen Schwierigkeiten ab, die Sie überwinden müssen.

Und sollte Ihnen unterwegs einmal alles zu lange dauern und der Mut sinken, dann kann Ihnen diese jüdische Weisheit vielleicht neue Energie und Zuversicht geben: »Wer langsam und besonnen geht, doch oft zuerst am Ziele steht.«

In diesem Sinne – auf denn!