Vorwort zur zweiten Ausgabe

In den 1990er-Jahren arbeiteten wir mit Unternehmen, deren Projekte Probleme bereiteten. Wir ertappten uns dabei, wie wir jedem das Gleiche sagten: Vielleicht sollten Sie das testen, bevor Sie es ausliefern? Warum geht der Build für diesen Code nur auf dem Rechner von Maria? Warum hat niemand die Anwender gefragt?

Um bei neuen Kunden Zeit zu sparen, begannen wir mit unseren Notizen. Und aus diesen Notizen wurde *Der Pragmatische Programmierer*. Zu unserer Überraschung schien das Buch den Nerv der Zeit zu treffen und es ist in den letzten 20 Jahren nach wie vor beliebt.

Aber 20 Jahre sind in Bezug auf Software viele Lebensalter. Nehmen Sie einen Entwickler von 1999 und setzen Sie ihn heute in ein Team und er würde sich sehr abkämpfen, in dieser fremden neuen Welt klarzukommen. Aber die Welt der 1990er-Jahre ist dem heutigen Entwickler ebenso fremd. Die Verweise des Buchs auf Dinge wie CORBA, CASE-Tools und indexbasierte Schleifen waren bestenfalls kurios und wahrscheinlich eher verwirrend.

Gleichzeitig haben 20 Jahre keinerlei Auswirkungen auf den gesunden Menschenverstand gehabt. Die Technologie mag sich geändert haben, aber die Menschen eben nicht. Praktiken und Ansätze, die damals eine gute Idee waren, sind auch heute noch gut. Diese Aspekte des Buchs sind gut gealtert.

Als es also an der Zeit war, zum 20. Jahrestag diese Ausgabe zu erstellen, mussten wir eine Entscheidung treffen. Wir könnten die Technologien, auf die wir verweisen, durchgehen, aktualisieren und dann Feierabend machen. Oder wir könnten die Annahmen, die den von uns empfohlenen Praktiken zugrunde liegen, im Lichte weiterer zwei Jahrzehnte Erfahrung noch einmal überprüfen.

Am Ende haben wir beides getan.

Daher ist dieses Buch so etwas wie ein Schiff des Theseus.[1] Etwa ein Drittel der Themen in diesem Buch sind brandneu. Von den übrigen wurden die meisten teilweise oder vollständig umgeschrieben. Unsere Absicht war es, die Dinge klarer, relevanter und hoffentlich etwas zeitloser zu machen.

Wir mussten einige schwierige Entscheidungen treffen. Wir haben den „Ressourcen"-Anhang weggelassen, sowohl weil es unmöglich wäre, auf dem neuesten Stand zu bleiben, als auch weil es einfacher ist, das Gewünschte zu finden. Angesichts des derzeitigen Überflusses an parallel arbeitender Hardware und des Mangels an guten Möglichkeiten, damit umzugehen, haben wir Themen im Zusammenhang mit Concurrency neu organisiert und verfasst. Ergänzt wird das durch Inhalte, die die sich verändernden Einstellungen und Umgebungen reflektieren: von der agilen Bewegung, die wir mit ins Leben gerufen haben, über eine steigende

[1] Wenn im Laufe der Jahre jede Komponente eines Schiffs ersetzt wird, wenn sie ausfällt, ist das daraus resultierende Schiff dann immer noch das gleiche Schiff?

Akzeptanz funktionaler Programmiersprachen bis hin zum wachsenden Bedürfnis nach Berücksichtigung von Privatsphäre und Sicherheit.

Interessanterweise gab es zwischen uns jedoch wesentlich weniger Diskussionen über den Inhalt dieser Ausgabe als beim Verfassen der ersten. Wir waren beide der Meinung, dass die wichtigen Sachen leichter zu erkennen sind.

Das Buch, das Sie in Händen halten, ist jedenfalls das Ergebnis. Bitte genießen Sie es. Vielleicht eignen Sie sich ein paar neue Praktiken an. Vielleicht beschließen Sie, dass einige der von uns vorgeschlagenen Dinge falsch sind. Lassen Sie sich auf Ihr Handwerk ein. Sagen Sie uns Ihre Meinung.

Denken Sie aber vor allem daran, dass es Spaß machen soll.

Aufbau des Buchs

Dieses Buch ist mehr als eine Sammlung von Tipps. Jedes Thema ist in sich abgeschlossen und behandelt einen bestimmten Aspekt. Zahlreiche Querverweise sollen helfen, jedes Thema in Zusammenhänge einzuordnen. Sie können das Buch mit seinen *Topics* in beliebiger Reihenfolge lesen.

Gelegentlich werden Sie auf einen Kasten mit der Beschriftung *Tipp n* stoßen (so wie *Tipp 1*, Kümmern Sie sich um Ihr Können). Tipps betonen einerseits entscheidende Stellen im Text, andererseits sind sie auch für sich alleine gültig – wir selbst wenden sie täglich an. Eine Zusammenfassung aller Tipps finden Sie auf dem Beihefter im Buch.

An geeigneten Stellen haben wir Übungen und weiterführende Aufgaben eingefügt. Während die Übungen relativ einfache Lösungen haben, sind die Aufgaben ziemlich offen gestellt. Um Ihnen eine Vorstellung von unserer Denkweise zu geben, haben wir unsere Lösungen zu den Übungen im Anhang zusammengetragen. Nur wenige Übungen haben eine einzige richtige Lösung. Die Aufgaben sind eher als Grundlage für Diskussionen oder Ausarbeitungen in Programmierkursen für Fortgeschrittene gedacht.

Am Ende finden Sie auch eine kurze Bibliografie, in der die Bücher und Artikel aufgeführt sind, auf die wir ausdrücklich verweisen.

Schall und Rauch

> *„Wenn ich ein Wort gebrauche“, sagte Humpty Dumpty in leicht verächtlichem Ton, „so bedeutet es das, was ich will, dass es bedeutet – nicht mehr und nicht weniger.“*
>
> Lewis Carroll, „Alice hinter den Spiegeln“

Quer über das Buch verteilt finden sich Fachjargonschnipsel – entweder vollkommen vernünftige Wörter, die für eine technische Bedeutung missbraucht wurden, oder schreckliche Wortschöpfungen, denen von Informatikern ohne Gefühl für Sprache eine Bedeutung zugewiesen wurde. Wir versuchen, solche Fachbegriffe zu definieren oder zumindest ihre Bedeutung zu beschreiben, wenn wir sie zum ersten Mal verwenden. Dennoch sind uns sicher einige durch die Lappen gegangen und andere wie Objekt und Relationale Datenbank sind so gebräuchlich, dass eine Definition nur langweilen würde. Wenn Sie auf ein unbekanntes Wort stoßen, überspringen Sie es bitte nicht. Nehmen Sie sich die Zeit, es im Internet oder einem Informatiklehrbuch nachzuschlagen, und schicken Sie uns eine E-Mail, damit wir in der nächsten Auflage eine Definition angeben können.

Wir haben uns entschieden, Rache an den Informatikern zu nehmen. Manchmal gibt es durchaus gute Fachwörter für Konzepte, doch wir haben beschlossen, diese Wörter zu ignorieren. Warum? Weil der vorhandene Fachjargon in der Regel auf einen bestimmten Problembereich oder auf eine bestimmte Entwicklungsphase beschränkt ist. Eine der Grundphilosophien dieses Buchs ist jedoch, dass die meisten von uns vorgeschlagenen Techniken universell sind: Modularität zum Beispiel betrifft Quelltext, Entwurf und Teamorganisation. Wenn wir gebräuchliche Fachwörter in einem breiteren Kontext verwenden wollten, erschien es verwirrend, und wir konnten den ganzen Ballast der ursprünglichen Bedeutung nicht loswerden. In diesen Fällen haben wir zum Verfall der Sprache beigetragen und unsere eigenen Wörter erfunden.

Quelltexte und andere Quellen

Die meisten Beispiele in diesem Buch sind kompilierbaren Quelltexten entnommen, die Sie von der Plus.Hanser-Webseite herunterladen können:

Die Quelltexte zu diesem Buch

Geben Sie auf *plus.hanser-fachbuch.de* einfach diesen Code ein:

Dann steht Ihnen der Download zur Verfügung.

Hinweis: In diesem Buch stehen die Namen der entsprechenden Dateien jeweils vor den Codes, die diesen Quelltexten entnommen wurden.

Außerdem haben wir eine eigene Website zum Pragmatischen Programmierer:

Die Website zum Pragmatischen Programmierer

Auf unserer Website

https://pragprog.com/titles/tpp20

finden Sie auch Links zu nützlichen Quellen zusammen mit Aktualisierungen des Buches und Neuigkeiten zu weiteren Aktivitäten der Pragmatischen Programmierer.

Danksagungen für die zweite Ausgabe

Wir haben in den letzten 20 Jahren buchstäblich Tausende von interessanten Gesprächen über die Programmierung geführt, Menschen auf Konferenzen, bei Kursen und manchmal sogar im Flugzeug getroffen. Jedes einzelne hat unser Verständnis des Entwicklungsprozesses erweitert und zu den Aktualisierungen in dieser Ausgabe beigetragen. Wir danken Ihnen allen (und sagen Sie uns immer wieder, wenn wir uns irren).

Dank geht auch an die Teilnehmer am Beta-Prozess des Buchs. Ihre Fragen und Kommentare haben uns geholfen, die Dinge besser zu erklären.

Bevor wir zur Beta-Version übergingen, baten wir einige Leute um Kommentare zu diesem Buch. Dank an VM (Vicky) Brasseur, Jeff Langr und Kim Shrier für eure detaillierten Kommentare und an José Valim und Nick Cuthbert für eure technischen Überprüfungen.

Wir bedanken uns bei Ron Jeffries, dass wir das Sudoku-Beispiel verwenden durften.

Vielen Dank an die Leute von Pearson, die bereit waren, dass wir dieses Buch auf unsere Weise verwirklichen durften.

Ein besonderer Dank gilt der unentbehrlichen Janet Furlow, die alles meistert, was sie angeht, und uns bei der Stange hält.

Und schließlich Applaus für all die Pragmatischen Programmierer da draußen, die in den letzten zwanzig Jahren die Programmierung für alle besser gemacht haben! Wir freuen uns auf zwanzig weitere Jahre.