Working Effectively With Legacy Code Robert C Martin Series

Alistair Cockburn beschreibt ausführlich, was Uses Cases sind, welche Bestandteile hineingehören und wie man sie strukturieren sollte. Besonders nützlich sind seine Erörterungen, wie man mit großen Mengen von Use Cases umgeht. Im zweiten Teil seines Buchs geht Cockburn auf verschiedene praktische Probleme ein. Es geht um Fragen wie "Woran erkennen wir, dass wir fertig sind?" oder "Wie fügen sich Use Cases in den Gesamtprozess ein?". Im dritten Teil werden die wichtigsten Themen noch einmal als knappe Referenz zusammengefasst.

Get more out of your legacy systems: more performance, functionality, reliability, and manageability Is your code easy to change? Can you get nearly instantaneous feedback when you do change it? Do you understand it? If the answer to any of these questions is no, you have legacy code, and it is draining time and money away from your development efforts. In this book, Michael Feathers offers start-to-finish strategies for working more effectively with large, untested legacy code bases. This book draws on material Michael created for his renowned Object Mentor seminars: techniques Michael has used in mentoring to help hundreds of developers, technical managers, and testers bring their legacy systems under control. The topics covered include Understanding the mechanics of software change: adding features, fixing bugs, improving design, optimizing performance Getting legacy code into a test harness Writing tests that protect you against introducing new problems Techniques that can be used with any language or platform—with examples in Java, C++, C, and C# Accurately identifying where code changes need to be made Coping with legacy systems that aren't object-oriented Handling applications that don't seem to have any structure This book also includes a catalog of twenty-four dependency-breaking techniques that help you work with program elements in isolation and make safer changes.

Das Standardwerk zum «Trilemma der Globalisierung» - jetzt mit einem Nachwort von Professor Gabriel Felbermayr. «Dani Rodrik führt uns pointiert vor Augen, dass wir uns in einem Dreieck der Unvereinbarkeiten bewegen: Zwischen den Zielen mehr Demokratie, mehr nationale Selbstbestimmung und mehr wirtschaftliche Globalisierung gibt es keine Schnittmenge. Die Demokratie wollen und dürfen wir nicht aufgeben, die Globalisierung werden wir nicht verhindern. Die Auseinandersetzung mit Rodriks These zwingt deshalb dazu, über nationale Grenzen hinauszudenken und den Multilateralismus zu stärken.» Wolfgang Schäuble

Working Effectively with Legacy CodeWORK EFFECT LEG CODE _p1Prentice Hall Professional

Lesbare, wartbare und zuverlässige Tests entwickelnStubs, Mock-Objekte und automatisierte FrameworksEinsatz von .NET-Tools inkl. NUnit, Rhino Mocks und Typemock Isolator Unit Testing, richtig durchgeführt, kann den Unterschied ausmachen zwischen einem fehlgeschlagenen Projekt und einem erfolgreichen, zwischen einer wartbaren Code-Basis und einer, die niemand freiwillig anpackt, zwischen dem Nach-Hause-Kommen um 2 Uhr nachts oder zum Abendessen, selbst noch kurz vor dem Release-Termin. Roy Osherove führt Sie Schritt für Schritt von einfachen Tests zu Tests, die wartbar, lesbar und zuverlässig sind. Er geht danach auf die Grundlagen des Interaction Testings ein und stellt schließlich bewährte Vorgehensweisen für das Schreiben, das Verwalten und das Warten der Unit Tests in echten Projekten vor. Darüber hinaus werden auch fortgeschrittene Themen behandelt wie Mocks, Stubs und Frameworks wie etwa Typemock Isolator und Rhino Mocks. Sie werden eine Menge zu fortgeschrittenen Testmustern und zur Testorganisation, zum Arbeiten mit Legacy Code und auch zu untestbarem Code erfahren. Und Sie lernen Werkzeuge kennen, die Sie beim Testen von Datenbanken und anderen Technologien brauchen werden. Alle Beispiele sind mit Visual Studio in C# geschrieben, so dass die Beispiele insbesondere für .NET-Entwickler nützlich sind. Aber auch für Programmierer anderer Sprachen wird das Buch von großem Nutzen sein, da die Prinzipien des Unit Testings für andere Sprachen dieselben sind. Roys Blog finden Sie auf ISerializable.com. Aus dem Inhalt: Verwenden eines Test-Frameworks (NUnit)Grundlegende TestattributeStubs zum Auflösen von AbhängigkeitenInteraction Testing mit Mock-ObjektenTesthierarchie und OrganisationDie Säulen guter TestsIntegration von Unit Testing in das UnternehmenUmgang mit Legacy Code

Get more out of your legacy systems: more performance, functionality, reliability, and manageability Is your code easy to change? Can you get nearly instantaneous feedback when you do change it? Do you understand it? If the answer to any of these questions is no, you have legacy code, and it is draining time and money away from your development efforts. In this book, Michael Feathers offers start-to-finish strategies for working more effectively with large, untested legacy code bases. This book draws on material Michael created for his renowned Object Mentor seminars: techniques Michael has used in mentoring to help hundreds of developers, technical managers, and testers bring their legacy systems under control. The topics covered include Understanding the mechanics of software change: adding features, fixing bugs, improving design, optimizing performance Getting legacy code into a test harness Writing tests that protect you against introducing new problems Techniques that can be used with any language or platform--with examples in Java, C++, C, and C# Accurately identifying where code changes need to be made Coping with legacy systems that aren't object-oriented Handling applications that don't seem to have any structure This book also includes a catalog of twenty-four dependency-breaking techniques that help you work with program elements in isolation and make safer changes.

Mit diesen sieben Sprachen erkunden Sie die wichtigsten Programmiermodelle unserer Zeit. Lernen Sie die dynamische Typisierung kennen, die Ruby, Python und Perl so flexibel und verlockend macht. Lernen Sie das Prototyp-System verstehen, das das Herzstück von JavaScript bildet. Erfahren Sie, wie das Pattern Matching in Prolog die Entwicklung von Scala und Erlang beeinflusst hat. Entdecken Sie, wie sich die rein funktionale Programmierung in Haskell von der Lisp-Sprachfamilie, inklusive Clojure, unterscheidet. Erkunden Sie die parallelen Techniken, die das Rückgrat der nächsten Generation von Internet-Anwendungen bilden werden. Finden Sie heraus, wie man Erlangs "Lass es abstürzen"-Philosophie zum Aufbau fehlertoleranter Systeme nutzt. Lernen Sie das Aktor-Modell kennen, das das parallele Design bei Io und Scala bestimmt. Entdecken Sie, wie Clojure die Versionierung nutzt, um einige der schwierigsten Probleme der Nebenläufigkeit zu lösen. Hier finden Sie alles in einem Buch. Nutzen Sie die Konzepte einer Sprache, um kreative Lösungen in einer anderen Programmiersprache zu finden – oder entdecken Sie einfach eine Sprache, die Sie bisher nicht kannten. Man kann nie wissen – vielleicht wird sie sogar eines ihrer neuen Lieblingswerkzeuge.

Können Sie Ihren Code leicht ändern? Können Sie fast unmittelbar Feedback bekommen, wenn Sie ihn ändern? Verstehen Sie ihn? Wenn Sie eine dieser Fragen mit nein beantworten, arbeiten Sie mit Legacy Code, der Geld und wertvolle Entwicklungszeit kostet. Michael Feathers erläutert in diesem Buch Strategien für den gesamten Entwicklungsprozess, um effizient mit großen, ungetesteten Code-Basen zu arbeiten. Dabei greift er auf erprobtes Material zurück, das er für seine angesehenen Object-Mentor-Seminare entwickelt hat. Damit hat er bereits

zahlreichen Entwicklern, technischen Managern und Testern geholfen, ihre Legacy-Systeme unter Kontrolle zu bringen. Darüber hinaus finden Sie auch einen Katalog mit 24 Techniken zur Aufhebung von Dependencies, die Ihnen zeigen, wie Sie isoliert mit Programmelementen arbeiten und Code sicherer ändern können.

JavaScript ist eine mächtige, objektorientierte Skriptsprache, deren Code in HTML-Seiten eingebettet und vom Browser interpretiert und ausgeführt wird. Richtig eingesetzt, eignet sie sich aber auch für die Programmierung komplexer Anwendungen und hat im Zusammenhang mit HTML5 noch einmal an Bedeutung gewonnen. Diese Kurzreferenz ist ein Auszug aus der überarbeiteten und ergänzten Neuauflage von JavaScript – Das umfassende Referenzwerk, 6. Auflage, der JavaScript-Bibel schlechthin. JavaScript kurz & gut befasst sich in den ersten neun Kapiteln mit der neuesten Version des Sprachkerns (ECMAScript 5) und behandelt die Syntax der Sprache, Typen, Werte, Variablen, Operatoren und Anweisungen sowie Objekte, Arrays, Funktionen und Klassen. All dies ist nicht nur für die Verwendung von JavaScript in Webbrowsern, sondern auch beim Einsatz von Node auf der Serverseite relevant. In den folgenden fünf Kapiteln geht es um die Host-Umgebung des Webbrowsers. Es wird erklärt, wie Sie clientseitiges JavaScript für die Erstellung dynamischer Webseiten und -applikationen verwenden und mit JavaScript auf die HTML5-APIs zugreifen. Diese Kapitel liefern Informationen zu den wichtigsten Elementen von clientseitigem JavaScript: Fenster, Dokumente, Elemente, Stile, Events, Netzwerke und Speicherung.

Mehr denn je ist das effektive Management der IT entscheidend für die Wettbewerbsfähigkeit von Organisationen. Viele Manager in softwarebasierten Unternehmen ringen damit, eine Balance zwischen Agilität, Zuverlässigkeit und Sicherheit ihrer Systeme herzustellen. Auf der anderen Seite schaffen es High-Performer wie Google, Amazon, Facebook oder Netflix, routinemäßig und zuverlässig hundertoder gar tausendmal pro Tag Code auszuliefern. Diese Unternehmen verbindet eins: Sie arbeiten nach DevOps-Prinzipien. Die Autoren dieses Handbuchs folgen den Spuren des Romans Projekt Phoenix und zeigen, wie die DevOps-Philosophie praktisch implementiert wird und Unternehmen dadurch umgestaltet werden können. Sie beschreiben konkrete Tools und Techniken, die Ihnen helfen, Software schneller und sicherer zu produzieren. Zudem stellen sie Ihnen Maßnahmen vor, die die Zusammenarbeit aller Abteilungen optimieren, die Arbeitskultur verbessern und die Profitabilität Ihres Unternehmens steigern können. Themen des Buchs sind: Die Drei Wege: Die obersten Prinzipien, von denen alle DevOps-Maßnahmen abgeleitet werden. Einen Ausgangspunkt finden: Eine Strategie für die DevOps-Transformation entwickeln, Wertketten und Veränderungsmuster kennenlernen, Teams schützen und fördern. Flow beschleunigen: Den schnellen Fluss der Arbeit von Dev hin zu Ops ermöglichen durch eine optimale Deployment-Pipeline, automatisierte Tests, Continuous Integration und Continuous Delivery. Feedback verstärken: Feedback-Schleifen verkürzen und vertiefen, Telemetriedaten erzeugen und Informationen unternehmensweit sichtbar machen. Kontinuierliches Lernen ermöglichen: Eine Just Culture aufbauen und ausreichend Zeit reservieren, um das firmenweite Lernen zu fördern.

Vom Absolutrang bis zum Zweifach-Varianzanalysemodell – alles, was Sie über weiterführende Statistik wissen sollten Es gibt Qualen, große Qualen und Statistik, so sehen es viele Studenten. Mit diesem Buch lernen Sie weiterführende Statistik so leicht wie möglich. Deborah Rumsey zeigt Ihnen, wie Sie Varianzanalysen und Chi-Quadrat-Tests berechnen, wie Sie mit Regressionen arbeiten, ein Modell erstellen, Korrelationen bilden, nichtparametrische Prozeduren durchführen und vieles mehr. Aber auch die Grundlagen der Statistik bleiben nicht außen vor und deshalb erklärt Ihnen die Autorin, was Sie zu Mittelwerten, Vertrauensintervallen und Co wissen sollten. So lernen Sie die Methoden, die Sie brauchen, und erhalten das Handwerkszeug, um erfolgreich Ihre Statistikprüfungen zu bestehen. Sie erfahren: • Wie Sie mit multiplen Regressionen umgehen • Was es mit dem Vorzeichentest und dem Vorzeichenrangtest auf sich hat • Wie Sie sich innerhalb der statistischen Techniken zurechtfinden • Was das richtige Regressionsmodell für Ihre Analyse ist • Wie Regression und ANOVA zusammenhängen

In Visionare der Programmierung - Die Sprachen und ihre Schopfer werden exklusive Interviews mit den Entwicklern von historischen wie auch von hoch aktuellen Programmiersprachen veroffentlicht. In dieser einzigartigen Zusammenstellung erfahren Sie uber die Hintergrunde, die zu den spezifischen Design-Entscheidungen in den Programmiersprachen gefuhrt haben und uber die ursprungliche Ziele, die die Entwickler im Kopf hatten, als sie eine neue Programmiersprache entwarfen. Ebenso konnen Sie lesen, wieso Abweichungen zum ursprunglichen Design entstanden und welchen Einflu die jeweilige Sprache auf die heutige Softwareentwicklung noch besitzt. Adin D. Falkoff: APL Thomas E. Kurtz: BASIC Charles H. Moore: FORTH Robin Milner: ML Donald D. Chamberlin: SQL Alfred Aho, Peter Weinberger und Brian Kernighan: AWK Charles Geschke und John Warnock: PostScript Bjarne Stroustrup: C++ Bertrand Meyer: Eiffel Brad Cox und Tom Love: Objective-C Larry Wall: Perl Simon Peyton Jones, Paul Hudak, Philip Wadler und John Hughes: Haskell Guido van Rossum: Python Luiz Henrique de Figueiredo und Roberto Ierusalimschy: Lua James Gosling: Java Grady Booch, Ivar Jacobson und James Rumbaugh: UML Anders Hejlsberg: Delphi-Entwickler und fuhrender Entwickler von C#

Verhaltensregeln für professionelle Programmierer Erfolgreiche Programmierer haben eines gemeinsam: Die Praxis der Software-Entwicklung ist ihnen eine Herzensangelegenheit. Auch wenn sie unter einem nicht nachlassenden Druck arbeiten, setzen sie sich engagiert ein. Software-Entwicklung ist für sie eine Handwerkskunst. In Clean Coder stellt der legendäre Software-Experte Robert C. Martin die Disziplinen, Techniken, Tools und Methoden vor, die Programmierer zu Profis machen. Dieses Buch steckt voller praktischer Ratschläge und behandelt alle wichtigen Themen vom professionellen Verhalten und Zeitmanagement über die Aufwandsschätzung bis zum Refactoring und Testen. Hier geht es um mehr als nur um Technik: Es geht um die innere Haltung. Martin zeigt, wie Sie sich als Software-Entwickler professionell verhalten, gut und sauber arbeiten und verlässlich kommunizieren und planen. Er beschreibt, wie Sie sich schwierigen Entscheidungen stellen und zeigt, dass das eigene Wissen zu verantwortungsvollem Handeln verpflichtet. In diesem Buch lernen Sie: Was es bedeutet, sich als echter Profi zu verhalten Wie Sie mit Konflikten, knappen Zeitplänen und unvernünftigen Managern umgehen Wie Sie beim Programmieren im Fluss bleiben und Schreibblockaden überwinden Wie Sie mit unerbittlichem Druck umgehen und Burnout vermeiden Wie Sie Ihr Zeitmanagement optimieren Wie Sie für Umgebungen sorgen, in denen Programmierer und Teams wachsen und sich wohlfühlen Wann Sie Nein sagen sollten – und wie Sie das anstellen Wann Sie Ja sagen sollten – und was ein Ja wirklich bedeutet Großartige Software ist etwas Bewundernswertes: Sie ist leistungsfähig, elegant, funktional und erfreut bei der Arbeit sowohl den Entwickler als auch den Anwender. Hervorragende Software wird nicht von Maschinen geschrieben, sondern von Profis, die sich dieser Handwerkskunst unerschütterlich verschrieben haben. Clean Coder hilft Ihnen, zu diesem Kreis zu gehören. Über den Autor: Robert C. Uncle Bob Martin ist seit 1970 Programmierer und bei Konferenzen in aller Welt ein begehrter Redner. Zu seinen Büchern gehören Clean Code – Refactoring, Patterns, Testen und Techniken für sauberen Code und Agile Software Development: Principles, Patterns, and Practices. Als überaus produktiver Autor hat Uncle Bob Hunderte von Artikeln, Abhandlungen und Blogbeiträgen verfasst. Er war Chefredakteur bei The C++ Report und der erste Vorsitzende der Agile Alliance. Martin gründete und leitet die Firma Object Mentor, Inc., die sich darauf spezialisiert hat, Unternehmen bei der Vollendung ihrer Projekte behilflich zu sein.

Dieses Werk ist Teil der Buchreihe TREDITION CLASSICS. Der Verlag tredition aus Hamburg veroffentlicht in der Buchreihe TREDITION CLASSICS Werke aus mehr als zwei Jahrtausenden. Diese waren zu einem Grossteil vergriffen oder nur noch antiquarisch erhaltlich. Mit der Buchreihe TREDITION CLASSICS verfolgt tredition das Ziel, tausende Klassiker der Weltliteratur verschiedener Sprachen wieder als gedruckte Bucher zu verlegen - und das weltweit! Die Buchreihe dient zur Bewahrung der Literatur und Forderung der Kultur. Sie tragt so dazu bei, dass viele tausend Werke nicht in Vergessenheit geraten

h2> Kommentare, Formatierung, Strukturierung Fehler-Handling und Unit-Tests Zahlreiche Fallstudien, Best Practices, Heuristiken und Code Smells Clean Code - Refactoring, Patterns, Testen und Techniken für sauberen Code Aus dem Inhalt: Lernen Sie, guten Code von schlechtem zu unterscheiden Sauberen Code schreiben und schlechten Code in guten umwandeln Aussagekräftige Namen sowie gute Funktionen, Objekte und Klassen erstellen Code so formatieren, strukturieren und kommentieren, dass er bestmöglich lesbar ist Ein vollständiges Fehler-Handling implementieren, ohne die Logik des Codes zu verschleiern Unit-Tests schreiben und Ihren Code testgesteuert entwickeln Selbst schlechter Code kann funktionieren. Aber wenn der Code nicht sauber ist, kann er ein Entwicklungsunternehmen in die Knie zwingen. Jedes Jahr gehen unzählige Stunden und beträchtliche Ressourcen verloren, weil Code schlecht geschrieben ist. Aber das muss nicht sein. Mit Clean Code präsentiert Ihnen der bekannte Software-Experte Robert C. Martin ein revolutionäres Paradigma, mit dem er Ihnen aufzeigt, wie Sie guten Code schreiben und schlechten Code überarbeiten. Zusammen mit seinen Kollegen von Object Mentor destilliert er die besten Praktiken der agilen Entwicklung von sauberem Code zu einem einzigartigen Buch. So können Sie sich die Erfahrungswerte der Meister der Software-Entwicklung aneignen, die aus Ihnen einen besseren Programmierer machen werden – anhand konkreter Fallstudien, die im Buch detailliert durchgearbeitet werden. Sie werden in diesem Buch sehr viel Code lesen. Und Sie werden aufgefordert, darüber nachzudenken, was an diesem Code richtig und falsch ist. Noch wichtiger: Sie werden herausgefordert, Ihre professionellen Werte und Ihre Einstellung zu Ihrem Beruf zu überprüfen. Clean Code besteht aus drei Teilen:Der erste Teil beschreibt die Prinzipien, Patterns und Techniken, die zum Schreiben von sauberem Code benötigt werden. Der zweite Teil besteht aus mehreren, zunehmend komplexeren Fallstudien. An jeder Fallstudie wird aufgezeigt, wie Code gesäubert wird – wie eine mit Problemen behaftete Code-Basis in eine solide und effiziente Form umgewandelt wird. Der dritte Teil enthält den Ertrag und den Lohn der praktischen Arbeit: ein umfangreiches Kapitel mit Best Practices, Heuristiken und Code Smells, die bei der Erstellung der Fallstudien zusammengetragen wurden. Das Ergebnis ist eine Wissensbasis, die beschreibt, wie wir denken, wenn wir Code schreiben, lesen und säubern. Dieses Buch ist ein Muss für alle Entwickler, Software-Ingenieure, Projektmanager, Team-Leiter oder Systemanalytiker, die daran interessiert sind, besseren Code zu produzieren. Über den Autor: Robert C. »Uncle Bob« Martin entwickelt seit 1970 professionell Software. Seit 1990 arbeitet er international als Software-Berater. Er ist Gründer und Vorsitzender von Object Mentor, Inc., einem Team erfahrener Berater, die Kunden auf der ganzen Welt bei der Programmierung in und mit C++, Java, C#, Ruby, OO, Design Patterns, UML sowie Agilen Methoden und eXtreme Programming helfen.

JUnit ist die Standardbibliothek zum Schreiben automatisierter Tests in Java. Dieses Buch enthält Grundlagen- und Expertenwissen für das effiziente Entwickeln automatisierter Tests in Java mit JUnit. Es vermittelt einen kompakten Überblick über alle Features von JUnit 3.8.1 bis JUnit 4.11 und zeigt anhand von Beispielen aus Tests bekannter Open-Source-Projekte, wann Sie diese sinnvoll einsetzen können. Darüber hinaus behandelt der Autor Open-Source-Bibliotheken wie Mockito und FEST, die das Schreiben von JUnit-Tests erleichtern, und verdeutlicht Programmierregeln für das Schreiben richtig "guter" Tests. Im Einzelnen werden behandelt: - Testgetriebene Entwicklung - Assertion-Bibliotheken - Unit-Tests mit Mock-Objekten - Programmieren gut verständlicher Tests - Programmieren schneller Tests - Tests abseits vom Happy Path - Nichtfunktionale Tests Abgerundet wird das Buch durch Hinweise und Tipps, wie Sie JUnit effektiv zusammen mit den bekannten Java-IDEs Eclipse und IntelliJ IDEA sowie mit den Build-Tools Ant und Maven einsetzen können.

Diese deutsche Übersetzung des Bestsellers von Aaron Hillegass ist das Standardwerk zur Mac-Programmierung. Hillegass behandelt alle Grundlagen, die Sie zur Programmierung für den Mac mit Cocoa brauchen, um featurereiche Anwendungen für OS X zu entwickeln. Das Buch ist eine wertvolle Ressource für jeden Mac-Programmierer!

Der neue Endzeit-Thriller vom internationalen Bestseller-Autor M. R. Carey: die Vorgeschichte zum erfolgreich verfilmten Zombie-Horror-Roman »The Girl with All The Gifts« (»Die Berufene«). Ein parasitärer Pilz hat unzählige Menschen in »Hungrige« verwandelt, ausgemergelte Zombies, die nach lebendigem Fleisch gieren. In einem zum Labor umgebauten Panzer ist eine kleine Gruppe von Militärs und Wissenschaftlern auf der Suche nach Antworten, unter ihnen der 15-jährige Hochbegabte Steven. Bei einem seiner heimlichen Streifzüge entdeckt er eine Gruppe von Kindern unter den Hungrigen, die sich eigentümlich kontrolliert verhalten. Steven erkennt in den Kindern die einmalige Chance auf ein Heilmittel – doch wie weit wird er gehen, um die Menschheit – oder die, die sich dafür halten – zu retten? »Der würdige Nachfolger für einen der besten Zombie-Romane ever« Tor.com »Ein gruseliger, mitreißender Pageturner darüber, was es heißt, ein Mensch zu sein« Kirkus Review »M.R. Carey ist eine Klasse für sich.« Daily Mail

• Umfassend überarbeitete und aktualisierte Neuauflage des Standardwerks in vollständig neuer Übersetzung • Verbesserungsmöglichkeiten von bestehender Software anhand von Code-Smells erkennen und Code effizient überarbeiten • Umfassender Katalog von Refactoring-Methoden mit Code-Beispielen in JavaScript Seit mehr als zwanzig Jahren greifen erfahrene Programmierer rund um den Globus auf dieses Buch zurück, um bestehenden Code zu verbessern und leichter lesbar zu machen sowie Software besser warten und erweitern zu können. In diesem umfassenden Standardwerk zeigt Ihnen Martin Fowler, was die Vorteile von Refactoring sind, wie Sie verbesserungsbedürftigen Code erkennen und wie Sie ein Refactoring – unabhängig von der verwendeten Programmiersprache – erfolgreich durchführen. In einem umfangreichen Katalog gibt Fowler Ihnen verschiedene Refactoring-Methoden mit ausführlicher Erläuterung, Motivation, Vorgehensweise und einfachen Beispielen in JavaScript an die Hand. Darüber hinaus behandelt er insbesondere folgende Schwerpunkte: • Allgemeine Prinzipien und Durchführung des Refactorings • Refactoring anwenden, um die Lesbarkeit, Wartbarkeit und Erweiterbarkeit von Programmen zu verbessern • Code-Smells erkennen, die auf Verbesserungsmöglichkeiten durch Refactoring hinweisen • Entwicklung zuverlässiger Tests für das Refactoring • Erkennen

von Fallstricken und notwendigen Kompromissen bei der Durchführung eines Refactorings Diese vollständig neu übersetzte Ausgabe wurde von Grund auf überarbeitet, um den maßgeblichen Veränderungen der modernen Programmierung Rechnung zu tragen. Sie enthält einen aktualisierten Katalog von Refactoring-Methoden sowie neue Beispiele für einen funktionalen Programmieransatz.
Bill ist IT-Manager bei Parts Unlimited. An einem Dienstagmorgen erhält er auf der Fahrt zur Arbeit einen Anruf seines CEO. Die neue IT-Initiative der Firma mit dem Codenamen Projekt Phoenix ist entscheidend für die Zukunft von Parts Unlimited, aber das Projekt hat Budget und Zeitplan massiv überzogen. Der CEO will, dass Bill direkt an ihn berichtet und das ganze Chaos in neunzig Tagen aufräumt, denn sonst wird Bills gesamte Abteilung outgesourct. Mit der Hilfe eines Vorstandsmitglieds und dessen mysteriöser Philosophie der Drei Wege wird Bill klar, dass IT-Arbeit mehr mit dem Fertigungsbereich in einer Fabrik zu tun hat als er sich je vorstellen konnte. Die Zeit drängt: Bill muss dafür sorgen, dass der Arbeitsfluss auch zwischen den Abteilungen deutlich besser läuft und das Business-Funktionalität zuverlässig bereitgestellt wird. Drei Koryphäen der DevOps-Bewegung liefern hier die rasante und unterhaltsame Story, in der sich jeder, der im IT-Bereich arbeitet, wiederfinden wird. Sie erfahren nicht nur, wie Sie Ihre eigene IT-Organisation verbessern können - nach der Lektüre dieses Buchs werden Sie IT auch nie wieder so sehen wie zuvor.