sebastian BERGMANN
stefan PRIEBSCH

SOFTWAREQUALITÄT IN PHP PROJEKTEN

Digg // eZ Components // studiVZ // swoodoo //
symfony // TYPO3 // Zend Framework

HANSER

Bergmann, Priebsch

Softwarequalität in PHP-Projekten

Sebastian Bergmann
Stefan Priebsch

Softwarequalität in PHP-Projekten

HANSER

Die Autoren:

Sebastian Bergmann & Stefan Priebsch, thePHP.cc, Wolfratshausen

Bibliografische Information Der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Gesamtlektorat: Fernando Schneider
Sprachlektorat: Sandra Gottmann, Münster-Nienberge
Herstellung: Thomas Gerhardy
Coverconcept: Marc Müller-Bremer, www.rebranding.de, München
Coverrealisierung: Stephan Rönigk
Datenbelichtung, Druck und Bindung: Kösel, Krugzell
Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702
Printed in Germany

ISBN 978-3-446-41923-0

www.hanser.de/computer

Inhaltsverzeichnis

Geleitwort

Die Entwicklung qualitativ hochwertiger Software sowie das Sicherstellen der Software-Qualität sind keine neuartigen Konzepte, und kaum jemand wird widersprechen, dass diese Konzepte für die Softwareentwicklung von enormer Bedeutung sind. Ich hatte das Privileg, über viele Jahre wirklich missionskritische Software zu entwickeln. Ich meine, die Art von Software, von der Menschenleben abhängen.

Während dieser Zeit habe ich eine Menge darüber gelernt, wie man einen Prozess für die Qualitätssicherung zu Beginn eines Projektes einführt und bis zum unternehmenskritischen Einsatz in Produktion vorantreibt. Das Gestalten eines Entwicklungsprozesses, der zu qualitativ hochwertiger Software führt, ist nicht trivial und erfordert nicht nur Unterstützung durch das Management, sondern auch dessen eigenes Engagement. Dies hat Einfluss auf die Aufbauorganisation des Unternehmens sowie seine Mitarbeiter, Systeme und Prozesse.

Meiner Meinung nach stellen die Probleme, die sich aus der großen Reichweite sowie aus der hohen Entwicklungsgeschwindigkeit des Internets ergeben, die Probleme in den Schatten, denen ich mich bei der Entwicklung der oben erwähnten missionskritischen Software stellen musste. Während viele dieser neuen Softwaresysteme „nur" unternehmenskritisch sind, sind sie in Wahrheit nicht weniger wichtig und müssen zusätzliche Schwierigkeiten wie beispielsweise Internationalisierung, Schutz vor bekannten und neuen Angriffen im Web sowie die Arbeit in verteilten Teams und mit immer kürzeren Releasezyklen bewältigen.

Im E-Commerce-Bereich schlägt sich ein Ausfall der Anwendung direkt in einem Ausfall von Einnahmen nieder; daher ist dort der Bedarf der Software-Qualität noch größer. Besonderes Augenmerk liegt hierbei auf Compliance sowie der Möglichkeit, eventuelle Fehler schnell beheben und diese Fehlerbehebung sofort testen und umgehend bereitstellen zu können. Die Anwendung muss nicht nur „online", sondern tatsächlich in der Lage sein, Transaktionen in Echtzeit zu verarbeiten.

Die wachsende Bedeutung der *User Experience* führt außerdem dazu, dass die durch die Benutzer der Anwendung wahrgenommene Qualität unternehmenskritisch wird: die Software muss nicht nur korrekt funktionieren, sondern auch den Erwartungen der Benutzer genügen. Ist dies nicht der Fall, müssen die ent-

sprechenden Änderungen in kürzester Zeit so umsetzbar sein, dass die Software-Qualität nicht beeinträchtigt wird. Sowohl die Prozesse für die Entwicklung und Qualitätssicherung der Software als auch die eingesetzten Systeme müssen diese schnellen Entwicklungen unterstützen.

Diese Herausforderungen haben zu signifikanten Änderungen im Bereich der Qualitätssicherung geführt, besonders im Vergleich dazu, wie früher auftragsentscheidende Software entwickelt wurde. Die Softwareentwicklung hat in den letzten Jahren große Fortschritte gemacht. Best Practices wurden etabliert und das Bewusstsein für Software-Qualität wurde gestärkt. Zu den wichtigsten Fortschritten gehört hierbei die Erkenntnis, dass die Entwickler in die Qualitätssicherung einbezogen werden müssen und die entsprechende Verantwortung nicht alleine bei der QA-Abteilung liegen darf. Die Methodik der kontinuierlichen Integration entschärft eines der größten Probleme – und einen der größten Flaschenhälse – bei der Entwicklung von qualitativ hochwertiger Software: die Integrationsphase. Der strategische Fokus auf automatisiertes Testen ermöglicht es, Fehlerbehebungen schneller ausliefern zu können. Hierdurch können Dienstgütevereinbarungen (englisch: *Service Level Agreements*) nicht nur eingehalten, sondern meist auch übererfüllt werden, was zu einer höheren Kundenzufriedenheit führt.

Dieses Buch behandelt die verschiedenen Disziplinen der Qualitätssicherung und ihre Auswirkungen auf Menschen, Systeme, Prozesse sowie Werkzeuge. Hierbei liegt der Fokus auf der praktischen Anwendung in PHP-basierten Projekten. Die in diesem Buch zusammengetragenen Fallstudien sind von unschätzbarem Wert. Sie erlauben das Lernen von anderen Entwicklerteams, beispielsweise wie diese Best Practices und Werkzeuge „in der richtigen Welt“einsetzen. Die Autoren verfügen über eine beispiellose Mischung aus theoretischem Hintergrundwissen und praktischer Erfahrung aus dem Alltag großer Projekte, die mit PHP realisiert wurden. Darüber hinaus leisten sie durch die Entwicklung von Werkzeugen für die Qualitätssicherung einen entscheidenden Beitrag für das PHP-Ökosystem. Ich kann mir keine besseren Autoren für ein solches Buch vorstellen.

Ich bin mir sicher, dass Ihnen dieses Buch dabei helfen wird, die Qualität Ihrer Projekte zu steigern, sodass sowohl Ihr Entwicklerteam als auch das Management auf die entwickelte Software stolz sein können.

Andi Gutmans, CEO Zend Technologies Ltd.

Vorwort

„Experience: that most brutal of teachers.
But you learn, my God do you learn.“ — C.S. Lewis

Über dieses Buch

Dem TIOBE Programming Community Index zufolge ist PHP die populärste Programmiersprache nach C/C++ und Java [TIOBE Mai 2010]. Gartner geht davon aus, dass viele neue, geschäftskritische Projekte von Unternehmen in einer dynamischen Sprache realisiert werden, und sieht PHP als derzeit stärksten Vertreter dieser Art von Programmiersprache [Gartner 2008]. PHP war von Anfang an für die Webprogrammierung konzipiert und dürfte um die Jahrtausendwende einer der wesentlichen Motoren des Dotcom-Booms gewesen sein.

Mittlerweile ist PHP zu einer Mehrzweckprogrammiersprache gereift und unterstützt sowohl die prozedurale als auch die objektorientierte Programmierung. Waren in der Vergangenheit Themen wie Performanz, Skalierbarkeit und Sicherheit Dauerbrenner in der PHP-Community, ist in den letzten Jahren auch Architektur und Qualität vermehrt Aufmerksamkeit zuteil geworden. In der Beratungspraxis zeigt sich, dass immer mehr Unternehmen ihre PHP-basierte Software modernisieren und ihren Entwicklungsprozess meist nach agilen Werten neu gestalten möchten. Eine Modernisierung der Codebasis ist meist entweder durch eine Migration von PHP 4 auf PHP 5 oder, um die Entwicklung zu standardisieren, durch die Einführung eines Frameworks motiviert.

Vor diesem Hintergrund verwundert es nicht, dass es heute eine Fülle an PHP-Frameworks gibt, die beim Lösen wiederkehrender Anwendungsfälle und bei der Standardisierung der Anwendungsentwicklung helfen wollen. Dynamische und statische Testverfahren sowie automatisierte Builds und kontinuierliche Integration sind für viele PHP-Entwickler längst keine Fremdwörter mehr. Aus PHP-Programmierung ist, gerade in unternehmenskritischen Projekten, Software-Engineering mit PHP geworden.

Ist dies ein PHP-Buch?

Anhand von Beispielen aus der PHP-Welt vermittelt dieses Buch die Planung, Durchführung und Automation von Tests für die unterschiedlichen Softwareschichten, die Messung von Software-Qualität mithilfe von Software-Metriken sowie den Einsatz geeigneter Methoden wie beispielsweise kontinuierlicher Integration. Wir gehen davon aus, dass unsere Leser entweder erfahrene PHP-Entwickler sind, die sich für die Qualitätssicherung in PHP-Projekten interessieren, oder Entwickler, die mit einer anderen Programmiersprache so weit vertraut sind, dass sie den Beispielen folgen können.

Dieses Buch kann und will keine Einführung in die (objektorientierte) Programmierung mit PHP 5 sein. Und obwohl sich viele Unternehmen im Rahmen der Migration von PHP 4 nach PHP 5 zum ersten Mal intensiver mit Qualitätssicherung in einem PHP-Projekt auseinandersetzen, so kann auch die Migration von PHP-Umgebungen und -Anwendungen in diesem Buch nicht behandelt werden. Für diese beiden Themen sei auf [Bergmann 2005] und [Priebsch 2008] verwiesen.

Neben den Entwicklern müssen sich auch Projektleiter und Qualitätsbeauftragte mit dem Thema Software-Qualität befassen. Wir hoffen, dass dieses Buch das gegenseitige Verständnis zwischen den verschiedenen an Software-Projekten beteiligten Gruppen fördert und allen Lesern eine Motivation bietet, die interne Qualität (siehe Abschnitt 1.1.2) ihres Codes zu verbessern.

Aufbau des Buches

Der Idee folgend, dass man am besten durch Erfahrung – auch und gerade aus den Erfahrungen anderer – lernt, stellt dieses Buch Fallstudien zusammen, die einen Blick hinter die Kulissen bekannter Firmen und Projekte erlauben und wertvolle Praxiserfahrungen vermitteln.

Der erste Abschnitt, *Grundlagen*, erklärt, was wir unter der Qualität von Software verstehen und wie man die unterschiedlichen Schichten einer Software testen kann.

Der Abschitt *Best Practices* zeigt erprobte Vorgehensweisen, unter anderem in Bezug auf das Schreiben von Unit-Tests, und wie diese von den Entwicklern von Digg Inc. und des TYPO3-Projektes umgesetzt werden. Das Kapitel *Bad Practices* greift dieselbe Thematik aus der anderen Richtung auf und zeigt die Fallstricke, auf die man beim Schreiben von Unit-Tests achten sollte.

Im Abschnitt *Server und Services* wird das Testen von serviceorientierten APIs und Server-Komponenten behandelt.

Der Abschnitt *Architektur* zeigt am Beispiel von Symfony, wie sowohl ein Framework selbst als auch die auf dessen Basis entwickelten Anwendungen getestet werden können. Am Beispiel der Graph-Komponente aus den eZ Components wird erklärt, wie eine gute Architektur aus lose gekoppelten Objekten selbst das Testen von binären Ausgabedaten wie Grafiken möglich macht. Das *Testen von*

Datenbank-Interaktionen ist ein Thema, das mehrere Schichten der Architektur einer Anwendung betrifft, und daher ein eigenes Kapitel wert.

Im Abschnitt *QA im Großen* berichten die Entwickler von studiVZ und Swoodoo von ihren Erfahrungen mit der Qualitätssicherung in großen Projekten und Teams. Das Kapitel *Kontinuierliche Integration* schlägt einen Bogen von den dynamischen zu den statischen Testverfahren und zeigt, wie die vielen verschiedenen Werkzeuge der Qualitätssicherung effektiv zusammen eingesetzt werden können.

Der letzte Abschnitt, *Nichtfunktionale Aspekte*, rundet das Buch mit einer Betrachtung der Qualitätsaspekte *Gebrauchstauglichkeit*, *Performanz* und *Sicherheit* ab.

Vorstellung der Autoren

Sebastian Bergmann, thePHP.cc

Diplom-Informatiker Sebastian Bergmann ist ein Pionier der Qualitätssicherung in PHP-Projekten. Sein Test-Framework PHPUnit ist ein De-facto-Standard. Er ist aktiv an der Entwicklung von PHP beteiligt und Schöpfer verschiedener Entwicklungswerkzeuge. Sebastian Bergmann ist ein international nachgefragter Experte. Als Autor gibt er seine langjährige Erfahrung in Büchern und Fachartikeln weiter und hält Vorträge auf Fachkonferenzen rund um die Welt.

Stefan Priebsch, thePHP.cc

Diplom-Informatiker Stefan Priebsch ist Spezialist für die Entwicklung PHP-basierter Software. Seine Erfahrung bringt er in Entwicklungswerkzeuge ein. Als Experte für objektorientierte Programmierung, Software-Architektur und Frameworks spricht er regelmäßig auf internationalen IT-Konferenzen. Stefan Priebsch ist Autor zahlreicher Bücher und Fachartikel über sämtliche Aspekte des Software-Lebenszyklus.

Robert Lemke, TYPO3 Association und Karsten Dambekalns, TYPO3 Association

Robert Lemke ist Gründungsmitglied der TYPO3 Association und leitet die Entwicklung des Rewrites von TYPO3 sowie des Frameworks FLOW3. Er hat eine besondere Vorliebe für agile Entwicklungsmethoden und es sich zum Ziel gesetzt, neue Ansätze wie Domain-Driven Design oder aspektorientiertes Programmieren in der PHP-Welt zu etablieren. Robert lebt in Lübeck, zusammen mit seiner Frau Heike, seiner Tochter Smilla und Vibiemme, ihrer Espressomaschine.

Karsten Dambekalns programmiert seit 1999 in PHP und entdeckte 2002 die immensen Möglichkeiten von TYPO3. Er ist heute einer der Kernentwickler von TYPO3 und FLOW3 sowie Mitglied im Steering Committee der TYPO3 Association.

Nach der Gründung einer eigenen Firma im Jahre 2000 steht Karsten Dambekalns seit 2008 wieder als Freelancer voll im Dienst der TYPO3-Entwicklung. Außerdem ist er Autor, spricht auf Konferenzen und verbringt den Großteil seiner Freizeit mit seiner Frau und seinen drei Kindern.

In ihrem Kapitel *TYPO3: die agile Zukunft eines schwergewichtigen Projekts* stellen Robert Lemke und Karsten Dambekalns Grundsätze und Techniken vor, mit denen das TYPO3-Projekt die Software-Qualität nachhaltig verbessern konnte.

Benjamin Eberlei, direkt:effekt GmbH

Benjamin Eberlei ist Softwareentwickler bei der direkt:effekt GmbH. In seiner Freizeit pflegt und entwickelt er Komponenten für das Zend-Framework sowie einige kleinere Open-Source-Projekte.

In seinem Kapitel *Bad Practices in Unit-Tests* zeigt Benjamin Eberlei, welche Fehler man beim Schreiben von Tests vermeiden sollte, um den größtmöglichen Nutzen aus dem Testen von Software ziehen zu können.

Matthew Weier O'Phinney, Zend Technologies Ltd.

Matthew Weier O'Phinney arbeitet als Project Lead für das Zend Framework. Zu seinen Aufgaben zählen das Release-Management ebenso wie die Implementierung und Verbreitung von Best Practices sowie die Kommunikation mit der Community. Matthew ist ein aktiver Befürworter von Open-Source-Software und PHP. Seit 2000 programmiert er in PHP und anderen Sprachen. Für Magazine wie PHP Architect und in seinem eigenen Blog[1] sowie in der Zend DevZone[2] schreibt er über aktuelle Themen.

In seinem Kapitel *Testen von serviceorientierten APIs* geht Matthew Weier O'Phinney auf die besonderen Herausforderungen beim Testen von Webdiensten ein und präsentiert Ansätze und Lösungen, die sich im Zend Framework-Projekt bewährt haben.

Tobias Schlitt, Qafoo GmbH

Tobias Schlitt ist ausgebildeter Fachinformatiker und Diplom-Informatiker. Er beschäftigt sich seit 1999 mit der Entwicklung von Webanwendungen auf Basis von PHP und war, nach mehrjähriger Aktivität im PEAR-Projekt, maßgeblich an Architektur und Entwicklung der eZ Components beteiligt. Daneben beteiligt er sich dauerhaft an verschiedenen Open-Source-Projekten rund um PHP. Als anerkannter Experte ist er beratend in den Bereichen Software-Architektur und Qualitätssicherung tätig. Tobias Schlitt gründete Mitte 2010 zusammen mit Manuel Pichler und Kore Nordmann, beide ebenfalls als Autoren in diesem Buch vertre-

[1] `http://weierophinney.net/matthew/`
[2] `http://devzone.zend.com/`

ten, die Qafoo GmbH, welche Experten-Consulting, Training und Support rund um die Entwicklung von hoch-qualitativem PHP-Code und Qualitätssicherung in Software-Projekten anbietet.

In seinem Kapitel *Wie man einen WebDAV-Server testet* zeigt Tobias Schlitt, dass man beim Testen manchmal unkonventionelle Wege gehen muss, um seine Ziele erreichen zu können.

Fabien Potencier, Sensio Labs

Fabien Potencier[1] entdeckte das Web 1994, zu einer Zeit, als das Verbinden mit dem Internet noch von kreischenden Tönen eines Modems begleitet wurde. Als passionierter Entwickler begann er sofort mit der Entwicklung von Webseiten mit Perl. Mit der Veröffentlichung von PHP 5 legte er seinen Fokus auf PHP und startete 2004 das Symfony Framework[2], um in seiner Firma die Mächtigkeit von PHP für Kunden voll nutzen zu können. Fabien ist ein "Serienunternehmer", der 1998 neben anderen Firmen auch Sensio, einen auf Webtechnologien und Internet-Marketing spezialisierten Dienstleister, gegründet hat. Ferner ist er der Schöpfer mehrerer Open-Source-Projekte, Autor, Blogger und Referent auf internationalen Konferenzen sowie stolzer Vater von zwei wundervollen Kindern.

In seinem Kapitel *Testen von Symfony und Symfony-Projekten* berichtet Fabien Potencier von seinen Erfahrungen aus dem Symfony-Projekt und zeigt unter anderem, wie das Testen von Symfony zu besseren Programmierschnittstellen geführt hat.

Kore Nordmann, Qafoo GmbH

Kore Nordmann entwickelt, leitet und plant seit mehreren Jahren verschiedene PHP-basierte Open-Source-Projekte. Im Sommer 2010 hat er zusammen mit Manuel Pichler und Tobias Schlitt die Qafoo GmbH gegründet und steht darüber Unternehmen zur Verfügung, um die Architektur und Qualität von Software-Projekten zu verbessern. Seine Expertise teilt er regelmäßig auf verschiedenen Konferenzen sowie in Artikeln und Büchern mit. Neben der Entwicklung der eZ Components leitet er die Entwicklung von Arbit[3], einer neuen Software für die Verwaltung von Software-Projekten, inklusive der Integration von Werkzeugen für die Qualitätssicherung.

In seinem Kapitel *Testen von Grafikausgaben* beschreibt Kore Nordmann, wie es mit einer guten Architektur und dem Einsatz von Mock-Objekten möglich ist, selbst eine Komponente, die binäre Grafikausgaben erzeugt, umfassend zu testen.

[1] `http://fabien.potencier.org/`
[2] `http://www.symfony-project.org/`
[3] `http://arbitracker.org/`

Michael Lively Jr, SellingSource LLC.

Michael Lively arbeitet seit 2001 mit PHP und bringt sich seit 2005 in der PHP Testing Community ein. Er ist der Schöpfer der Erweiterung für Datenbanktests in PHPUnit, zu dem er auch weitere Patches beigetragen hat. Michael Lively arbeitet als Lead Developer und Application Architect für die SellingSource LLC mit Sitz in Las Vegas. Zu seinem Arbeitsbereich gehört die Entwicklung einer mit PHP realisierten Enterprise-Plattform für Kreditmanagement, die von Hunderten Maklern für Millionen von Kunden verwendet wird.

In seinem Kapitel *Testen von Datenbank-Interaktionen* dokumentiert Michael Lively Jr die Funktionalität der Datenbankerweiterung von PHPUnit und zeigt, wie dieses mächtige Werkzeug effektiv eingesetzt werden kann.

Christiane Philipps, DailyDeal GmbH, und Max Horváth, Vodafone GmbH

Max Horváth ist Lead Software Engineer bei Vodafone Internet Services und beschäftigt sich seit zehn Jahren mit Webentwicklung. In dieser Zeit hat er mit Unternehmern, Entwicklern und Designern zusammengearbeitet, um Webprojekte für große und kleine Unternehmen umzusetzen. Während der Arbeit an der Fallstudie für dieses Buch war er Team Lead Mobile Development bei VZnet Netzwerke.

Christiane Philipps ist CTO bei der DailyDeal GmbH. Die Fachinformatikerin arbeitet seit 2000 als Consultant und in Festanstellung für und mit Webunternehmen. Ihr Herz schlägt besonders für Agile Testing und Agile Leadership, Themen, über die sie auch in ihrem Blog[1] regelmäßig schreibt. Von Frühjahr 2008 bis Herbst 2009 arbeitete sie bei VZnet Netzwerke, davon ein Jahr lang als Leiterin des Bereichs Quality Assurance & Deployment.

In ihrem Kapitel *Qualitätssicherung bei studiVZ* berichten Christiane Philipps und Max Horváth, wie sie PHPUnit und Selenium RC erfolgreich in einem der größten sozialen Netzwerke Europas eingeführt haben.

Manuel Pichler, Qafoo GmbH, und Sebastian Nohn, Ligatus GmbH

Manuel Pichler kam erstmals 1999 mit der Programmiersprache PHP in Kontakt und ist ihr bis heute treu geblieben. Während seines Studiums sammelte er erste Erfahrungen im Bereich Qualitätssicherung. Im Anschluss an das Studium arbeitete er als Software-Architekt und entwickelte im Rahmen dieser Tätigkeit verschiedene Erweiterungen für CruiseControl, die dann 2007 die Grundlage für phpUnderControl bildeten. Neben diesem Projekt ist er auch der Schöpfer von

[1] `http://agile-qa.de/`

PHP_Depend[1] und PHPMD[2], Werkzeugen zur statischen Analyse von PHP-Code. Im Sommer 2010 hat er zusammen mit Kore Nordmann und Tobias Schlitt die Qafoo GmbH gegründet und bietet darüber Support und Schulungen rund um das Themengebiet Qualitätssicherung an.

Sebastian Nohn beschäftigt sich seit 1996 mit dynamischen Websites und seit 2002 mit Qualitätssicherung im kommerziellen und Open-Source-Bereich. Er war einer der ersten, der CruiseControl für die Nutzung in PHP-Projekten adaptierte, und gab den Anstoß für die Entwicklung von phpUnderControl. Zurzeit ist er bei der Ligatus GmbH, einem der führenden Performance-Marketing-Anbieter beschäftigt, wo er den Bereich Qualitätssicherung aufbaute und die Verantwortung als Teamleiter für die Bereiche Qualitätssicherung und Infrastruktur trägt. Sebastian Nohn ist Wirtschaftsinformatiker und schreibt in unregelmäßigen Abständen in seinem Weblog[3] über IT-Themen.

In ihrem Kapitel *Kontinuierliche Integration* berichten Manuel Pichler und Sebastian Nohn, wie kontinuierliche Integration, nachträglich eingeführte Unit-Tests, Software-Metriken und weitere statische Testverfahren dazu beigetragen haben, die Qualität einer Legacy-Applikation deutlich zu erhöhen.

Lars Jankowfsky, swoodoo AG

Lars Jankowfsky ist CTO der Swoodoo AG und verantwortlich für den PHP-basierten Flug- und Hotel-Preisvergleich. Seit mehr als 15 Jahren entwickelt er Webanwendungen und nutzt PHP seit den frühen Anfängen. Neben der Entwicklung von Software ist die Leitung von eXtreme Programming-Teams eine seiner Leidenschaften.

In seinem Kapitel *swoodoo – eine wahrhaft agile Geschichte* zeigt Lars Jankowfsky, wie agile Methoden und eine serviceorientierte Architektur die sanfte und kontinuierliche Evolution einer Anwendung ermöglicht haben.

Jens Grochtdreis

Jens Grochtdreis[4] ist freier Webentwickler und Berater. Er ist dabei auf moderne Frontend-Entwicklung und Barrierefreiheit spezialisiert. Vor seiner Selbstständigkeit arbeitete er zehn Jahre in Agenturen, unter anderem an Projekten für eine sehr große deutsche Bank und einen großen Telekommunikationsanbieter. Jens gründete 2005 die Webkrauts[5], um für ein besseres Medium zu streiten. Wenn er nicht gerade bloggt, twittert, surft oder codet, dann entspannt er sich bei Comics, Krimis, hört Blues oder kocht.

---

[1] `http://pdepend.org/`
[2] `http://phpmd.org/`
[3] `http://nohn.net/blog/`
[4] `http://grochtdreis.de`
[5] `http://webkrauts.de/`

In seinem Kapitel *Gebrauchstauglichkeit* zeigt Jens Grochtdreis, wie einfach nutzbare und verständliche Webseiten entwickelt werden können und wie die Gebrauchstauglichkeit getestet werden kann.

Brian Shire

Brian Shire entdeckte das Programmieren im Alter von acht Jahren auf einem Apple IIe. Wenn er keine Spiele spielte, lernte er die Programmiersprache BASIC. Während der Arbeit an der Fallstudie für dieses Buch arbeitete er bei Facebook Inc., wo er für die Skalierung der PHP-Infrastruktur verantwortlich war. In seinen vier Jahre bei Facebook wuchs die Plattform von 5 Millionen Nutzern auf 175 Millionen Nutzer. In dieser Zeit wurde Brian zu einem Kernentwickler von APC, einem Bytecode- und Daten-Cache für PHP. Er trug außerdem zur Entwicklung des PHP-Interpreters sowie verschiedenen Erweiterungen in PECL bei. Brian teilt seine Erfahrung und sein Wissen als Referent auf internationalen Konferenzen und in seinem Blog[1]. Zurzeit lebt er in San Francisco.

In seinem Kapitel *Performanz* motiviert Brian Shire das Testen der Performanz von Webanwendungen und führt in die wichtigsten Werkzeuge und Methoden dafür ein.

Arne Blankerts, thePHP.cc

Arne Blankerts hat langjährige Erfahrung als IT-Entwicklungsleiter. Seine Software fCMS nutzt innovativ XML-Technologien und ist Basis unternehmenskritischer Anwendungen internationaler Konzerne. Er ist aktiv an der Dokumentation von PHP beteiligt. Arne Blankerts ist Experte für IT-Sicherheit und schreibt darüber eine Kolumne in einem Fachmagazin. Er ist gefragter Referent auf internationalen IT-Konferenzen, Buchautor und veröffentlicht Fachartikel.

In seinem Kapitel *Sicherheit* zeigt Arne Blankerts, wie einfach das Schreiben grundsätzlich sicherer Anwendungen ist, wenn man die gängigen Angriffsvektoren kennt und einige wichtige Regeln beachtet.

[1] `http://tekrat.com`

Stichwortverzeichnis