HANSER

Ralf Wirdemann, Thomas Baustert

Rapid Web Development mit Ruby on Rails

ISBN-10: 3-446-41498-3
ISBN-13: 978-3-446-41498-3

Vorwort

Weitere Informationen oder Bestellungen unter
http://www.hanser.de/978-3-446-41498-3
sowie im Buchhandel.

Vorwort

Als wir im Herbst 2004 über ein Blog-Posting von David Heinemeier Hansson auf Ruby on Rails gestoßen sind, konnten wir nicht absehen, welche Bedeutung Rails für die Web-Entwicklung im Allgemeinen und für unsere eigene Softwareentwicklung im Speziellen haben würde.

Nach vielen Jahren Java- und insbesondere J2EE-Entwicklung waren wir zunächst einmal überrascht, wie einfach Softwareentwicklung sein kann. Eigentlich waren wir so überrascht, dass wir anfangs nicht glauben konnten, dass sich Rails wirklich für die Entwicklung größerer Web-Anwendungen eignen würde.

Die Monate November und Dezember des Jahres 2004 verbrachten wir mit der Entwicklung kleinerer Rails-Applikationen, um zunächst die Technologie kennen zu lernen. Diese Erfahrung hat unsere anfängliche Skepsis deutlich gemindert, zumal wir mehr und mehr feststellen konnten, dass Softwareentwicklung mit Rails nicht nur einfach ist, sondern darüber hinaus auch zu sauber entworfenen Systemen führt, die sich durch ihre Wartbarkeit und damit Langlebigkeit auszeichnen.

Seit Januar 2005 entwickeln wir Rails-Applikationen im kommerziellen Umfeld. Was wir seitdem täglich neu erfahren, ist eine bisher nicht gekannte Produktivität und eine neue „Leichtigkeit der Softwareentwicklung“, die wir Ihnen mit diesem Buch nahebringen wollen.

Hamburg, im November 2005 *Ralf Wirdemann und Thomas Baustert*

Vorwort zur 2. Auflage

Fast zeitgleich mit Abschluss der Arbeiten an der 2. Auflage unseres Buches hat Rails am 25. Juli 2006 seinen 2. Geburtstag gefeiert. Nach nunmehr zwei Jahren Rails ist eines sicher: Wenn die Welt ein neues Web-Framework brauchte, dann dieses.

Konferenzen, Zeitschriftenartikel, Bücher und eine ständig zunehmende Anzahl an Rails-Projekten zeigen dies. Dabei sind es nicht nur kleine Internetagenturen, die ihre Entwicklung auf Rails umstellen, sondern auch große Firmen, die vorhandene JEE-Lösungen portieren oder neue Projekte auf Basis von Rails starten.

Rails- und Ruby-Konferenzen sind innerhalb kürzester Zeit ausverkauft. Während die seit Jahren in den USA stattfindende Ruby-Konferenz noch bis zum Jahr 2004 eher ein Nischendasein fristete, war die im Oktober 2006 stattfindende Ruby-Konferenz in Denver innerhalb von vier Stunden nach Öffnung der Registrierungs-Website ausverkauft.

Die Popularität von Rails ist längst aus den USA zu uns nach Europa und in andere Teile der Welt übergeschwappt. Die deutsche Rails Commmunity wächst ständig. Lokale Usergruppen organisieren regelmäßige Treffen. Die erste deutsche Rails-Konferenz steht in den Startlöchern. Die Anzahl der Rails-Projekte in Deutschland nimmt kontinuierlich zu.

Rails hat der Sprache Ruby zu neuem Ruhm verholfen. Dies zeigen nicht zuletzt die von O'Reilly veröffentlichten Verkaufszahlen: Im Jahr 2005 wurden 1552% mehr Ruby-Bücher verkauft, als im Vergleichszeitraum des Vorjahres. Viele Entwickler haben die Eleganz und Ausdrucksstärke von Ruby kennen und schätzen gelernt. Wir sind vielen skeptischen Entwicklern begegnet, wenn es um den Umstieg von Java auf Ruby ging. Wir sind bisher jedoch keinem Entwickler begegnet, der nach erfolgtem Umstieg zurück in die Java-Welt wollte.

Das Rails-Framework wurde in den letzten zwei Jahren kontinuierlich weiterentwickelt und verbessert. Rails bleibt dabei trotzdem schlank und einfach. Neue Features werden nur dann ins Framework aufgenommen, wenn sie sich in der Praxis bewährt haben und allgemeinen Nutzen versprechen. Ein Beispiel hierfür sind die seit Rails 1.1 verfügbaren Integrationstests, die von 37signals im Rahmen ihrer Campfire-Software entwickelt wurden und erst nach ihrem erfolgreichen Einsatz Einzug in das Rails-Framework hielten. Ein anderes Beispiel für eine einfache, dafür aber umso wirkungsvollere Verbesserung sind Active Record-Migrations, die

die bis Rails 1.0 verwendeten SQL-Skripte ersetzen und die inkrementelle Pflege von Datenbankschemata ermöglichen.

Trotz anhaltender Euphorie gibt es weiterhin viel zu tun. Z.B. zeichnet sich auch nach zwei Jahren kein eindeutiger Favorit am Internationalisierungshimmel ab. Diese Erkenntnis war für uns Anlass genug, das Internationalisierungskapitel der ersten Auflage vollständig neu zu schreiben und die aktuell verfügbaren und praxiserprobten Lösungen vorzustellen.

Aber auch für uns persönlich hat sich in den letzten zwei Jahren vieles geändert. Geblieben ist die Begeisterung für Rails als Web-Framework und Ruby als Programmiersprache. Hinzugekommen sind eine Menge neuer Erfahrungen, viele Leute, die wir im letzten Jahr kennen gelernt haben und die inzwischen selbst von PHP oder Java auf Rails umgestiegen sind.

Unser Buch versteht sich weiterhin als Ein- und Aufsteigerbuch für die Ruby on Rails-Entwicklung. Wir haben versucht, die 2. Auflage unseres Buches entscheidend zu verbessern. Neben vielen Korrekturen und Anmerkungen unserer Leser enthält die Neuauflage alle wesentlichen Änderungen von Rails 1.1[1]:

- Active Record-Migrations
- neue Active Record-Assoziationen
- RJS-Templates
- Formulare mit form_for()
- Unterstützung unterschiedlicher Clients mit response_to()
- Integrationstests
- Nutzung von Apache und Mongrel

Darüber hinaus enthält die zweite Auflage ein ausführliches Kapitel zum Thema Deployment mit Capistrano, dem Standardwerkzeug für die automatisierte Auslieferung und Verteilung von Rails-Applikationen.

Wir wünschen Ihnen viel Spaß und Freude beim Durcharbeiten dieses Buches und vor allem Produktivität und Erfolg für Ihr nächstes Ruby on Rails-Projekts.

Hamburg, im September 2006 *Ralf Wirdemann und Thomas Baustert*

[1] bzw. von Rails 1.0, sofern sie es nicht in die 1. Auflage geschafft haben.

Vorwort zur 3. Auflage

Mit Erscheinen der dritten Auflage unseres Buches ist Ruby on Rails mehr als 3 Jahre alt und hat die Versionsnummer 2 erreicht. Rails 2 markiert einen weiteren Meilenstein in der Entwicklung des Frameworks. Neben vielen kleineren Neuerungen dürfte das Thema REST die wohl einschneidendste Neuerungen dieses Major-Releases sein.

REST ist zwar schon seit Rails 1.2 fester Bestandteil des Frameworks, seit Rails 2 aber zum bevorzugtem Paradigma für die Entwicklung von Web-Applikationen mit Rails geworden. Entwickler müssen umdenken und dabei eine neue Sichtweise aufs Web entwickeln: Web-Applikationen sind nunmehr keine Ansammlung dynamischer Webseiten mehr, sondern vielmehr eine Menge miteinander verbundener Ressourcen, deren HTML-Repräsentation im Browser nur eine von vielen möglichen Repräsentationsvarianten ist.

Eine logische Konsequenz aus der Verwendung von Ressourcen ist Erweiterung von Rails um Multiview-Fähigkeit: Controller erkennen das vom Client gewünschte Ressourcen-Format und reagieren darauf durch Auslieferung eines bestimmten Templates, z.B. *index.html.erb* für normale Browser-Requests, oder *index.iphone.erb* für Requests von mobilen Internet-Geräten.

Weitere Änderungen gab es in den Frameworks Action Pack und Active Record. Zum Beispiel können Ressourcen zukünftig direkt an Helper in Controllern und Views übergeben, was den Source-Code noch einmal schlanker und lesbarer macht. Einige Beispiele: *redirect_to(@person)*, *link_to(@person.name, @person)* oder *form_for(@person)*. Im Bereich Active Record werden Migrationen einfacher und Fixtures übersichtlicher.

Rails hat aber nicht nur zugenommen, sondern auch abgespeckt: Das Subframework Action Web Service gibt es nicht mehr. Durch die konsequente Umstellung auf REST hat ein neues Web-Service Paradigma Einzug in die Rails-Welt gehalten. RESTful entwickelte Anwendungen benötigen kein spezielles Web-Service Interface mehr, da die Anwendung von Haus aus eine REST-Schnittstelle zur Verfügung stellt, das nicht nur von Browsern, sondern von jedem REST sprechenden Client genutzt werden kann. Die Anwendung wird so zur API.

Sie sehen schon, Rails 2 ist vollgepackt mit vielen großen, aber auch kleinen wichtigen Änderungen und Verbesserungen. Wir haben unser Buch vollständig überar-

beitet und um die Rails 2 spezifischen Änderungen erweitert und die auf älteren Rails-Versionen basierenden Beispiele angepasst.

Wir bedanken uns bei unseren Kollegen und Reviewern Sascha Teske und Michael Voigt für ihre Korrekturarbeit und das Testen unserer Beispiele.

Hamburg, im Mai 2008 *Ralf Wirdemann und Thomas Baustert*