HANSER

Vorwort

Peter Armstrong

Flexible Rails

Flex 3 auf Rails 2

Übersetzt aus dem Englischen von Dorothea Heymann-Reder

ISBN: 978-3-446-41573-7

Weitere Informationen oder Bestellungen unter

http://www.hanser.de/978-3-446-41573-7

sowie im Buchhandel.

Vorwort

Anfang 2006 erzählte ich in einer Kneipe meinen Freunden von meiner Begeisterung für das neue Framework „Ruby On Rails“. Mike Jones, einem Flash-Entwickler, erklärte ich, dass sich Ruby On Rails doch prima in sein neues Lieblingsspielzeug Adobe Flex 2 (Beta) integrieren ließe. Beide Technologien waren aus dem Wunsch entstanden, coole Sachen einfacher zu erstellen. Sie waren wie füreinander geschaffen, und mir war klar, dass schon bald jemand kommen und sie integrieren würde.

Im April und Mai desselben Jahres schrieb ich in meinem Blog liverail.net ein zweiteiliges Tutorial und hielt eine Präsentation vor der London Flash Platform User Group, wobei ich in Flex eine RIA CRUD-Schnittstelle mit Ruby On Rails als Backend entwickelte.

Das war nur der Anfang einer richtigen Bewegung. Immer zahlreicher wurden all jene, die beide Technologien miteinander kombinierten. Sie fanden in der Entwickler-Community immer mehr Anhänger – angefangen von Flash/Flex-Entwicklern, die ihre ersten Ausflüge in die Backend-Programmierung unternahmen, bis hin zu erfahrenen Ruby-Programmierern, die nicht im Traume daran gedacht hätten, etwas mit Flash zu entwickeln, weil sie sich durch „die Timeline“ abschrecken ließen. Seit meinen ersten Blogbeiträgen sind mehrere Entwickler begeistert auf Flex und Rails eingeschwenkt und haben Integrationssoftware geschrieben, neue Firmen gegründet und Blogs veröffentlicht – allen voran Peter, der engagiert, passioniert und verrückt genug war, zu erkennen, dass es an der Zeit war, ein ganzes Buch über dieses Thema zu schreiben.

Peter ist bei seinem Konzept geblieben, er sah es wachsen und gedeihen: Jeden Monat kommen neue Projekte mit Flex und Rails heraus. Peter hat kontinuierlich Flexible Rails entwickelt, ist mit Flex 3 und Rails 2 immer up to date geblieben, arbeitet mit dem Flex-MVC-Framework Cairngorm und ist mit RubyAMF auf der Höhe der Zeit. Sein Buch enthält gleich mehrere Tutorials, die uns durch den kompletten Lebenszyklus seines RIA-Phantom-Startups „Pomodo“ geleiten, von der Datenbank bis zum Desktop mit Adobe AIR.

Wenn Sie ein neues RIA-Startup anpeilen, einen internen Data-Warehouse-Client planen oder einfach etwas anderes tun wollen, ist dieses Buch das Richtige für Sie.

– Stuart Eccles

Technischer Direktor und Mitgründer von Made by Many Ltd., GB.

Über dieses Buch

Viele technische Bücher sind wie Disneyland: sie sehen vielversprechend aus, sind aber teuer, dauern ewig, und am Ende ist man enttäuscht.

Dieses Buch ist anders.

In *Flexible Rails* erstellen wir eine wirkliche Anwendung – na ja, eine Anwendung, die einer wirklichen so nahe kommt, wie es in einem Buch irgend möglich ist. Im Laufe der Zeit erläutern wir die im Code eingeführten Konzepte und den Code selbst. Da der gesamte Code der MIT-Lizenz unterliegt, können Sie daraus nehmen, was Sie wollen, und es als Grundlage des Web 2.0-Startups Ihrer Träume verwenden, ohne einen Cent an mich (oder Manning) zahlen zu müssen. (Wenn Sie es mit dem Startup zum Millionär bringen, nehme ich natürlich gerne ein kleines Geschenk an!)

Roadmap

Wie die meisten iterativ entwickelten Anwendungen besteht unser Buch aus vier Teilen:

1. Erste Schritte
2. Anwendungserstellung
3. Refactoring
4. Abschlussarbeiten

In Teil 1, „Erste Schritte“, erledigen wir die nötigen Setup-Arbeiten, um im Rest des Buchs zum spaßigen Teil überzugehen. Wir installieren alles, schreiben eine Flex/Rails-Version von „Hello World“, legen in Rails Benutzer an, melden sie an und klinken schließlich die Flex-Oberfläche ein. Dieser Teil enthält drei Iterationen:

- *Iteration 1:* „Warum sind wir hier? Und wohin gehen wir?“ – Diese Iteration erklärt den Zweck des Buchs, die Geschichte von Flex und Rails, wie beide zusammenpassen, und gibt einen Überblick.
- *Iteration 2:* „Hello World“– Diese Iteration enthält drei separate Anweisungsmengen (Windows oder Mac OS X+Flex Builder 3, Windows+Flex SDK und Mac OS X+Flex SDK), um alles Erforderliche zu installieren und „Hello World“ zum Laufen zu bringen.

- *Iteration 3:* „Erste Schritte“– In dieser Iteration richten wir MySQL ein und fügen die Funktionalität zum Anlegen von Konten und Anmelden von Benutzern unserer Rails-Anwendung hinzu, wobei wir das restful_authentication-Plugin verwenden. Danach integrieren wir die Flex-Oberfläche, um die Rails-Funktionalität für Konten und Benutzeranmeldung nutzen zu können.

In Teil 2, „Anwendungserstellung“, tauchen wir tief in die Verwendung von Flex mit Rails ein. Am Ende dieses Teils werden Sie die Grundlagen der Integration beider Technologien meistern. Dieser Teil enthält vier Iterationen:

- *Iteration 4:* „Die Flex-Benutzeroberfläche“ – In dieser Iteration erstellen wir eine Stub-UI für den Hauptteil der Flex-Anwendung.
- *Iteration 5:* „Rails mit REST“– Als Nächstes fügen wir Rails-Modelle und -Controllers für Aufgaben, Projekte und Standorte hinzu sowie die nötigen Migrationen zur Erstellung Ihrer Datenbanktabellen. Auch REST wird in dieser Iteration eingeführt. Abschließend werden einige wichtige Sicherheitsbelange angesprochen, die von Anfang an zu berücksichtigt sind.
- *Iteration 6:* „Flex auf Rails“ – In dieser Iteration integrieren wir die in Iteration 4 erstellten Flex-Benutzeroberflächen sowie die in Iteration 5 hinzugekommenen Rails-Controller.
- *Iteration 7:* „Validierung“ – Wir fügen dem Prozess der Benutzerkontenerstellung volle Validierungsunterstützung auf der Rails-Seite und der Flex-Seite hinzu.

An diesem Punkt sind wir bereit, höhere Ziele in Angriff zu nehmen, was wir in Teil 3 „Refactoring“ auch tun werden. Dieser Teil enthält zwei Iterationen:

- *Iteration 8:* „Refactoring auf Cairngorm“ – Wir stellen den Code aus Teil 2 auf Cairngorm um, ein Anwendungs-Framework für Flex.
- *Iteration 9:* „Zustandsverwaltung auf dem Client“ – Der Code wird erneut umgeschrieben, dieses Mal, um ein richtiges Objektmodell hinzuzufügen, statt bloß XML auf dem Client zu verwenden.

Am Ende dieses Teils verstehen Sie mehr vom Entwurf in Flex und von den Möglichkeiten des Datenaustauschs zwischen Flex und Rails. Wenn beim Refactoring das Objektmodell von seiner Transportmethode (derzeit XML) abgekoppelt wird, können wir auch eine andere Transportmethode in Betracht ziehen.

Im letzten Teil, „Schlussarbeiten“, schließen wir die Anwendung ab, stellen sie auf RubyAMF ab und erweitern sie um die Adobe Integrated Runtime (AIR).

Dieser Teil enthält drei Iterationen:

- *Iteration 10:* „Fertigstellung der Anwendung“ – In dieser Iteration erstellen wir die restlichen Features in Pomodo.
- *Iteration 11:* „Refactoring auf RubyAMF“ – Wir versetzen Pomodo durch ein erneutes Refactoring in die Lage, RubyAMF statt XML zur Datenübermittlung zwischen Flex und Rails zu verwenden. Da AMF ein Binärprotokoll und XML Text ist (und zwar ein äußerst langatmiger), kann dies massive Performance-Verbesserungen bringen.

- *Iteration 12:* „Rails on AIR (Adobe Integrated Runtime)“ – In dieser letzten Iteration des Buchs konvertieren wir den Code, damit er auf AIR läuft, und umschreiben das in dieser Iteration erstellte Notely-Feature so, dass es die AIR-spezifischen Features nutzen kann. Das ist nicht als komplette AIR-Einführung gedacht, sondern soll Ihnen lediglich einen Eindruck von einer der spannenden Möglichkeiten vermitteln, Ihre Flex+Rails-Anwendungen über das traditionelle Modell von Webanwendungen hinaus zu entwickeln.

Insgesamt verfolgt dieses Buch den Ansatz, Sie tief in die Welt von Flex und Rails eintauchen zu lassen. Anstatt Ihnen langweilige Theorie und an den Haaren herbeigezogene Beispiele zu servieren, werden wir gemeinsam eine echte Anwendung entwickeln und in diesem Prozess alles Nötige lernen. Auch behaupte ich nicht, dass dieses Buch einzigartig sei: Ich zitiere viele exzellente Quellen, darunter nicht nur Bücher, sondern auch viele Blogbeiträge. Ein Markenzeichen der Rails-Community sind ihre überaus produktiven Blogger; weil die meisten Mitglieder der Rails-Community aus diesen Blogs lernen, ist es nur fair, im Buch eine kurze Erklärung zu liefern und zu den Einzelheiten auf die Blogs zu verweisen, anstatt Blogbeiträge mit eigenen Worten hier wiederzugeben.

Was dieses Buch nicht ist

Dieses Buch ist als informatives, interessantes, nützliches und gelegentlich auch unterhaltsames Tutorial für Software-Entwickler gedacht, egal, wie viel Erfahrung sie mit Flex, Ruby oder Rails haben mögen. Ich versuche nicht, ein umfassendes Tutorial über Ruby, Rails, Flex oder ActionScript 3 abzuliefern, da jedes einzelne dieser Themen ein eigenes Buch füllen würde. Zum Glück gibt es solche Werke bereits:

- Ruby – *Programming Ruby*, 2. Aufl.; The Ruby Way, 2. Aufl.
- Rails – Agile Web Development with Rails, 2. Aufl.
- Ruby and Rails – *Ruby for Rails*, 1. Aufl.
- Flex 3 – *Flex 3 Developer's Guide* (kostenlose, 1 435 Seiten lange PDF von Adobe)
- ActionScript 3 – *Programming ActionScript 3.0* (kostenlose, 576 Seiten lange PDF von Adobe)

Dieses Buch will *nicht* den vorgenannten Werken Konkurrenz machen, sondern setzt voraus, dass Sie diese (oder das entsprechende Wissen) entweder besitzen oder bereit sind, sie zu kaufen. (Die PDFs über Flex und ActionScript 3 sind sogar gratis.) Wenn Sie tatsächlich ernsthaft mit Rails arbeiten möchten, sollten Sie die zweite Auflage von *Programming Ruby* erwerben (die erste gibt es zwar umsonst, ist aber überholt) sowie die zweite Auflage von *Agile Web Development with Rails (AWDwR)* und/oder die erste Auflage von *Ruby for Rails.*

Dieses Buch soll genügend Informationen und externe Verweise geben, damit auch Entwickler ohne Erfahrung mit Flex, Ruby oder Rails ihm folgen können und – sofern erforderlich – Hilfe finden, aber nicht so umfassend, dass Leser, die bereits die Grundlagen von Rails oder Flex beherrschen, sich langweilen. Ich gehe davon aus, dass die meisten Leser

einem der beiden Lager angehören (Rails-Entwickler, die Flex als Alternative zu AJAX lernen möchten, oder Flex-Entwickler, die eine andere Technologie als Java für die Serverseite suchen). Wenn Sie weder Flex- noch Rails-erfahren sind, aber bereits Web- oder Desktop-UI-Software entwickelt haben, müssten Sie ebenfalls in der Lage sein, den Ausführungen zu folgen. Viele Leser sind so „eingestiegen".

Ein Wort zu den Iterationen

Jeglicher Code im Buch kann

- von http://www. flexiblerails.com/code-samples
- oder von der Verlags-Website unter http:// www.manning.com/armstrong
- oder von http://www.manning.com/FlexibleRails

heruntergeladen werden. Der Download besteht aus einer großen Zip-Datei mit je einem Ordner für jede Iteration im Buch, ausgenommen Iteration 1, die keinen Code enthält. Auf diese Weise können Sie jede Iteration starten und nachvollziehen, indem Sie das Verzeichnis der vorherigen Iteration benutzen. Wenn Ihnen Tipparbeit zuwider ist, können Sie auch während der Lektüre jede fertige Iteration laden.

Da ich im besten Programmierer-Sinne faul bin, sind nicht alle Codebeispiele separate Projekte, sondern Kopien desselben Projekts in verschiedenen Phasen. Ich empfehle Ihnen, einen Ordner namens „current" zu erstellen und in Flex Builder zu laden. Auf diese Weise können Sie am Ende jeder Iteration Ihren aktuellen Ordner löschen und diese Iteration an die Stelle der aktuellen ablegen. Wenn Sie Flex Builder dann neu starten, merkt er lediglich, dass sich eine Reihe von Dateien geändert hat, während das Projekt immer noch dasselbe ist. Das gilt aber nur, wenn Sie nicht das Flex Framework SDK verwenden.

Es sollte klar sein, dass dieses Vorgehen nichts mit echter Programmierarbeit zu tun hat. Für die wirkliche Entwicklung verwenden Sie Subversion (oder Git) und lassen es das aktuelle public\bin-Verzeichnis ignorieren, das Sie für die Flex-Ausgabe anlegen. In Anhang A können Sie genauer nachlesen, wie Subversion mit Flex und Rails verwendet wird. Wenn Sie mit Git arbeiten, benötigen Sie keine Anleitung.

Welches Flex?

Die kurze Antwort lautet: Flex 3.

Die längere: Ein Großteil des wurde Buchs ursprünglich mit Flex 2 erstellt und das Ganze vor der jüngsten Überarbeitung auf Flex 3 Beta 2 aktualisiert. Also ist der gesamte Code in diesem Buch mit Flex 3 Beta 2 erstellt worden. Als das Buch im Satz war, wurde der Flex-Code mit Flex 3 Beta 3 getestet, was weitgehend ohne Auswirkungen blieb. Nur in Iteration 12 („Rails on AIR") benannte Flex 3 Beta 3 die `Shell. shell` in `NativeApplication.nativeApplication` um, woraufhin ich den Code in Iteration 12 entsprechend aktualisierte. Jeglicher Flex-Code, außer dem in Iteration 12, der AIR verwendet, funktioniert in Flex 2 und Flex 3.

Welches Rails?

Die kurze Antwort: Rails 2.

Die etwas längere: Die *jüngste Überarbeitung* dieses Buchs wurde mit dem ersten Release Candidate von Rails 2 vorgenommen, dessen Gem-Version die 1.99.0 ist. Rails 2 Final mit der Gem-Version 2.0.1 kam heraus, als das Buch bereits im Satz war. Also ging ich noch während der Herstellung hin und aktualisierte mein Rails auf 2.0.1, um das ganze Buch auf Rails 2.0.1 umzustellen. Sie können also beruhigt sein: Auch wenn im Text des Buchs Rails 1.99.0 erwähnt wird, wurde der Code sowohl mit Rails 1.99.0 als auch mit Rails 2.0.1 getestet. Der Code, der unter http:// www.flexiblerails.com/code-samples und auf der Website des Verlags zum Download bereitsteht, verwendet Rails 1.99.0. (Da ich auch Rails 2.0.1 einbezog, hätte ich diesen Code ebenso gut freigeben können, aber das Format hätte nicht gepasst. Da durch Kopieren und Einfügen in einer PDF die Formatierung verloren geht, habe ich das unterlassen.)

Die *wirklich lange Antwort* ist, dass ich dieses Buch mehr als zweimal überarbeitete: Die frühen Iterationen schrieb ich bereits, als Rails noch in der Version 1.1 vorlag. Diese Iterationen waren ursprünglich kürzer, dafür gab es mehr als 20 von ihnen. (Für das Layout dieses Buchs wäre das ein unbeherrschbarer Albtraum, da alle Bugfixes übertragen werden müssen.) So machte ich mich im Mai 2007 daran, das Buch komplett umzuschreiben, wobei die Zahl der Iterationen drastisch reduziert und der gesamte Code auf Rails 1.2 aktualisiert wurde. Im Sommer 2007 schloss ich mit Manning den Verlagsvertrag zur Veröffentlichung des Buchs, doch kurz darauf erschien bereits der Preview Release von Rails 2, der dafür sorgte, dass das Buch bereits vor Erscheinen veraltet sein würde. Also wurde im Oktober 2007 *das Buch abermals komplett umgeschrieben* und auf den Preview Release von Rails 2 (Gem-Version 1.2.3.7707) umgestellt. Als dann im November und Dezember 2007 die Endüberarbeitung anstand, aktualisierte ich das Buch von der ersten bis zur letzten Seite auf den ersten Release Candidate von Rails (Gem-Version 1.99.0), um schließlich, als das Buch bereits gesetzt wurde, die Funktionsfähigkeit des Codes mit Rails 2 Final (Gem-Version 2.0.1) zu testen, *wieder* von der ersten bis zur letzten Seite.

Schreiben bedeutet Neuschreiben, vor allem, wenn sich das Thema so schnell bewegt wie die Kombination von Flex und Rails.

Die Codebeispiele

Zur besseren Lesbarkeit zeige ich den Quellcode einer Datei mit ***neuen oder modifizierten Codezeilen in fettkursiv*** und ~~Codezeilen, die gelöscht werden sollen, in durchgestrichener Codeschrift~~. Unveränderte Teile einer Datei lasse ich oft aus und setze ein Auslassungszeichen (...) an ihre Stelle. Wird ein großer Codeabschnitt gelöscht, verwende ich das Auslassungszeichen auch innerhalb dieses Abschnitts (da es Papierverschwendung ist, lange durchgestrichene Codepassagen abzudrucken). Wenn Sie Code aus den Beispielen in Ihre Programme kopieren, achten Sie darauf, die durchgestrichenen Zeilen wegzulassen oder zu löschen. Beach-

ten Sie, dass dieses Buch 64-spaltigen Code zeigt, was zu einigen rein formatbedingten Modifikationen am generierten Rails-Code führt, damit dieser gut in die 64 Spalten passt. Diese Änderungen werden nicht gekennzeichnet oder erläutert, weil das nervig wäre. Manchmal ist es schlicht nicht möglich, Code hübsch in die 64 Spalten einzupassen (Rails-Code bevorzugt oft lange Zeilen, und die Inline-Ereignisbehandlung im MXML-Code tendiert ebenfalls dazu). In diesen Fällen wird der Code automatisch umbrochen und mit einem Fortsetzungssymbol angezeigt.

Eine Zip-Datei mit dem vollständigen Code können Sie

- sowohl von http://www. flexiblerails.com/code-samples
- als auch von der Verlags-Website herunterladen.

Author Online

Der Kauf von *Flexible Rails* umfasst freien Zugriff auf ein privates Webforum des Manning-Verlags, in dem Sie das Buch kommentieren, technische Fragen stellen und Hilfe von den Autoren und anderen Benutzern erhalten können. Um ins Forum zu gelangen und um es zu abonnieren, gehen Sie mit dem Webbrowser zu http://www.manning.com/ Flexible-Rails oder http://www.manning.com/armstrong. Auf dieser Seite erfahren Sie, wie Sie nach Registrierung Zugang zum Forum bekommen, welche Hilfe geboten wird und welche Verhaltensregeln zu befolgen sind.

Manning möchte den Lesern ein Portal bieten, auf dem ein sinnvoller Dialog zwischen einzelnen Lesern und dem Autor möglich ist. Wir übernehmen keine Verpflichtung für ein bestimmtes Maß an Mitwirkung seitens der Autoren, deren Beiträge zum Forum freiwillig (und unentgeltlich) sind. Am besten stellen Sie dem Autor interessante Fragen, die seine Neugier wecken.

Das Author Online-Forum und die Archive vorangegangener Diskussionen sind auf der Verlags-Website zugänglich, solange das Buch noch gedruckt wird.

Über den Autor

Peter Armstrong ist seit Juli 2004 Flex-Entwickler (also seit Flex 1.0) und interessiert sich seit Mitte des Jahres 2005 für Ruby on Rails (also noch vor Rails 1.0). Bevor er auf Flex umstellte, war er fünf Jahre lang Java-Swing-Entwickler, mit einem kurzen Zwischenspiel als PHP-Entwickler im Jahre 2000, während der Neue Markt boomte. Peter kam von Swing und fand Flex anfangs attraktiv, weil es ihm so vertraut schien und eher an Swing als an Webentwicklung erinnerte. Nach fünf Jahren Java war mit Ruby und Ruby on Rails ein frischer Wind zu spüren.

Peter ist der Organisator der Vancouver Ruby/Rails Meetup Group (http:// ruby.meetup.com/ 112/). Vorträge über die Verwendung von Flex mit Rails hielt er auf der Vancouver Flash/Flex Meetup Group, auf der RailsConf 2007 BOF, beim Vancouver RIA Developer Camp, bei Rails to Italy 2007 und bei VanDev.

Die Website des Autors zu diesem Buch ist http://www.flexiblerails.com und das zugehörige Blog ist http://www.flexiblerails.com/blog. Peters persönliches Blog finden Sie unter http://www.peterarmstrong.com. Peters Consulting-Firma, spezialisiert auf Flex und Rails-Entwicklung, Schulung und Workshops finden Sie unter http://www.ruboss.com.

Peter lebt mit seiner Frau Caroline und seinem Sohn Evan in der Gegend von Vancouver, British Columbia. Wenn er nicht gerade programmiert, schreibt, liest oder Vater und Ehemann ist, fährt er gerne Snowboard und spielt Computerspiele. Gäbe es nicht Desktop Tower Defense, Slashdot und reddit, wäre dieses Buch locker einen Monat früher fertig geworden, wenn nicht eher!