

HANSER

# **Rapid Web Development mit Ruby on Rails**

Ralf Wirdemann, Thomas Baustert

ISBN 3-446-40932-7

Inhaltsverzeichnis

Weitere Informationen oder Bestellungen unter  
<http://www.hanser.de/3-446-40932-7> sowie im Buchhandel

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Für wen dieses Buch bestimmt ist	2
1.2	Organisation des Buches	3
1.3	Web-Site zum Buch	5
<b>2</b>	<b>Überblick und Installation</b>	<b>7</b>
2.1	Was ist Ruby on Rails?	7
2.2	Bestandteile von Rails	10
2.2.1	Komponenten und Zusammenspiel	10
2.2.2	Action Pack	11
2.2.3	Active Record	12
2.2.4	Action Mailer	12
2.2.5	Ajax	12
2.2.6	Web Services	12
2.2.7	Unit Tests	13
2.3	Installation	13
2.3.1	Windows	13
2.3.2	Linux	14
2.3.3	Mac OS X	15
2.3.4	Glückwunsch! Willkommen an Bord!	15
<b>3</b>	<b>Hands-on Rails</b>	<b>17</b>
3.1	Entwicklungsphilosophie	18
3.2	Domain-Modell	18
3.3	OnTrack Product Backlog	19
3.4	Aufsetzen der Infrastruktur	20
3.5	Projekte erfassen, bearbeiten und löschen	21
3.5.1	Modell erzeugen	21
3.5.2	Datenbankmigration	22

---

3.5.3	Controller erzeugen	27
3.6	Iterationen hinzufügen	29
3.7	Zwischenstand	36
3.8	Iterationen anzeigen	36
3.9	Iterationen bearbeiten und löschen	38
3.10	Tasks hinzufügen	40
3.11	Tasks anzeigen, bearbeiten und löschen	44
3.12	Struktur in die Seiten bringen	46
3.13	Validierung	47
3.14	Benutzerverwaltung	51
3.15	Login	52
3.16	Tasks zuweisen	55
3.17	Endstand und Ausblick	57
<b>4</b>	<b>Active Record</b>	<b>59</b>
4.1	Active Record-Klassen	59
4.1.1	Mehr über Modellattribute	61
4.1.2	Mehr über Primärschlüssel	62
4.1.3	Mehr über Tabellennamen	63
4.2	Active Record direkt verwenden	64
4.3	Objekte erzeugen, laden, aktualisieren und löschen	65
4.3.1	Erzeugung	66
4.3.2	Objekte laden	67
4.3.3	Objekte aktualisieren	67
4.3.4	Objekte löschen	69
4.4	Mehr über Finder	70
4.4.1	Suchbedingungen: conditions	70
4.4.2	Ordnung schaffen: order	71
4.4.3	Limitieren: limit	71
4.4.4	Seitenweise: limit und offset	71
4.4.5	Weitere Parameter: joins und include	72
4.5	Dynamische Finder	73
4.6	Kann ich weiterhin SQL verwenden?	73
4.7	Metadaten – Daten über Daten	75
4.8	Assoziationen	76
4.8.1	Grundsätzliches	76
4.8.2	1:1-Beziehungen: has_one – belongs_to	78

---

4.8.3	1:N-Beziehungen: <code>has_many – belongs_to</code> . . . . .	84
4.8.4	N:M-Beziehungen: <code>has_and_belongs_to_many</code> . . . . .	90
4.8.5	Polymorphe Assoziationen: <code>has_many – belongs_to</code> . . . . .	94
4.8.6	<code>has_many :through</code> . . . . .	97
4.9	Aggregation . . . . .	99
4.10	Vererbung . . . . .	100
4.11	Transaktionen . . . . .	103
4.12	Von Bäumen und Listen . . . . .	106
4.12.1	<code>acts_as_list()</code> . . . . .	106
4.12.2	<code>acts_as_tree()</code> . . . . .	108
4.13	Validierung . . . . .	110
4.13.1	Validierungs-Klassenmethoden . . . . .	111
4.14	Callbacks . . . . .	115
4.14.1	Überschreiben von Callback-Methoden . . . . .	116
4.14.2	Callback-Makros . . . . .	116
4.14.3	Observer . . . . .	118
4.15	Konkurrierende Zugriffe und Locking . . . . .	119
4.15.1	Optimistisches Locking . . . . .	119
<b>5</b>	<b>Action Controller</b> . . . . .	<b>121</b>
5.1	Controller-Grundlagen . . . . .	121
5.1.1	Actions . . . . .	122
5.1.2	Responses . . . . .	123
5.2	Datenaustausch . . . . .	126
5.2.1	Vom Controller zum View . . . . .	126
5.2.2	Vom View zum Controller . . . . .	127
5.2.3	Aus der Action in den View und zurück . . . . .	129
5.3	Redirects . . . . .	129
5.3.1	Weitere Parameter von <code>redirect_to()</code> . . . . .	130
5.4	Unterschiedliche Response-Formate: <code>respond_to()</code> . . . . .	131
5.5	Sessions . . . . .	133
5.5.1	Session-Daten löschen . . . . .	134
5.5.2	Session und Modelle . . . . .	134
5.5.3	Session-Optionen . . . . .	135
5.6	Der Flash-Speicher . . . . .	136
5.6.1	Weitere Flash-Methoden . . . . .	137
5.7	Filter . . . . .	137

---

5.7.1	Around-Filter . . . . .	139
5.7.2	Bedingungen . . . . .	139
5.7.3	Filterklassen und Inline-Filter . . . . .	140
5.7.4	Filtervererbung . . . . .	141
5.7.5	Filterketten . . . . .	141
5.8	Routing und URL-Generierung . . . . .	142
5.8.1	Routing . . . . .	142
5.8.2	URL-Generierung . . . . .	145
5.9	Layout . . . . .	146
5.9.1	Automatische Layoutzuweisung . . . . .	147
5.9.2	Explizite Layoutzuweisung . . . . .	148
5.9.3	Dynamische Bestimmung des Layouts . . . . .	149
5.9.4	Action-spezifische Layouts . . . . .	149
5.10	Datei-Downloads . . . . .	150
5.10.1	Die Methode send_data() . . . . .	150
5.10.2	Die Methode send_file() . . . . .	150
<b>6</b>	<b>Action View . . . . .</b>	<b>153</b>
6.1	HTML-Templates . . . . .	153
6.2	Helper-Module . . . . .	155
6.3	Action View Helper . . . . .	156
6.3.1	Formulare . . . . .	157
6.3.2	Formular-Helper mit Bezug zu Modellen . . . . .	158
6.3.3	Formular-Helper ohne Bezug zu Modellen . . . . .	160
6.3.4	HTML-Tags . . . . .	161
6.3.5	Texte und Zahlen . . . . .	162
6.3.6	Datum und Zeit . . . . .	163
6.3.7	Auswahlboxen . . . . .	163
6.3.8	Verweise und URLs . . . . .	165
6.3.9	Ressourcen einbinden . . . . .	167
6.3.10	JavaScript . . . . .	168
6.3.11	Code speichern und wiederverwenden . . . . .	168
6.3.12	Debugging . . . . .	169
6.4	Layouts . . . . .	169
6.5	Partial Views – ein View aus Teilen . . . . .	170
6.6	Components – Funktionalität wieder verwenden . . . . .	172
6.7	Pagination – seitenweise blättern . . . . .	174

---

6.8	Anzeige Fehlermeldungen . . . . .	176
6.9	XML-Templates . . . . .	177
<b>7</b>	<b>Internationalisierung . . . . .</b>	<b>179</b>
7.1	Internationalisierung oder Lokalisierung? . . . . .	179
7.2	Lokalisierung und Codes . . . . .	180
7.2.1	Language Tag . . . . .	180
7.2.2	Locale . . . . .	180
7.3	Checkliste . . . . .	181
7.4	Internationalisierung vorbereiten . . . . .	181
7.5	Ruby Gettext-Package . . . . .	184
7.5.1	Installation . . . . .	184
7.5.2	Texte übersetzen . . . . .	184
7.5.3	Die Schritte im Überblick . . . . .	189
7.5.4	Anwendung und Gettext zusammenbringen . . . . .	189
7.5.5	Lokalisierte Templates . . . . .	191
7.5.6	Controller . . . . .	191
7.5.7	Modelle . . . . .	191
7.5.8	Dynamische Texte und Pluralisierung . . . . .	193
7.5.9	Datum, Zahlen und Währungen . . . . .	195
7.5.10	Wochen- und Monatsnamen . . . . .	196
7.5.11	Sortierung . . . . .	198
7.5.12	Zusammenfassung . . . . .	198
7.6	Globalize . . . . .	199
7.6.1	Vorbereitung und Installation . . . . .	199
7.6.2	Übersetzungen einpflegen . . . . .	201
7.6.3	Lokalisierte Templates . . . . .	203
7.6.4	Controller . . . . .	204
7.6.5	Modelle . . . . .	204
7.6.6	Dynamische Texte und Pluralisierung . . . . .	206
7.6.7	Datum, Zahlen und Währungen . . . . .	206
7.6.8	Wochen- und Monatsnamen . . . . .	208
7.6.9	Sortierung . . . . .	208
7.6.10	Zusammenfassung . . . . .	208
7.7	Gettext oder Globalize . . . . .	208
7.8	Weitere Bibliotheken . . . . .	209
7.8.1	GLoc . . . . .	209

---

7.8.2	ICU4R	210
<b>8</b>	<b>Action Mailer</b>	<b>211</b>
8.1	E-Mail-Versand	211
8.1.1	E-Mail erstellen	211
8.1.2	E-Mail-Objekt erzeugen	214
8.1.3	E-Mail versenden	215
8.1.4	Testen	216
8.1.5	Multipart E-Mails	218
8.2	E-Mail-Empfang	220
8.2.1	E-Mail empfangen	221
8.2.2	Empfang testen	222
<b>9</b>	<b>Web Services</b>	<b>225</b>
9.1	Web Service-Grundlagen	225
9.2	Web Service Interfaces	226
9.2.1	Die Methode <code>api_method()</code>	226
9.2.2	Parameter-Typen	227
9.2.3	Parameter-Formate	228
9.3	Web Service-Implementierung	228
9.3.1	Direct Dispatching	229
9.3.2	Layered Dispatching	230
9.3.3	Delegated Dispatching	231
9.4	Generatoren	232
9.5	Testen von Web Services	232
9.6	Filter	233
9.7	Web Service Clients	234
9.7.1	Umstellung auf <code>ActionWebService::Struct</code>	234
9.7.2	Ein SOAP-Client für den Ontrack-Service	235
9.8	WSDL-Generierung	236
<b>10</b>	<b>Web 2.0 – Ajax und Co.</b>	<b>237</b>
10.1	Ajax	237
10.1.1	Ajax und Rails	239
10.1.2	Hello, Ajax World	240
10.1.3	Ajax-Formulare	241
10.1.4	Feldbeobachter	244
10.1.5	Callback-Methoden	247

10.1.6	Drag and Drop	248
10.1.7	Autocompletion	254
10.1.8	Automatisches Aktualisieren	254
10.2	Effekte – Visuelles Feedback für den Anwender	255
10.2.1	Elemente ein- und ausblenden	255
10.2.2	Dem Anwender zeigen, dass der Server arbeitet	256
10.2.3	Weitere Effekte	257
10.3	RJS Templates	257
10.3.1	Hinzufügen von Tasks visuell optimieren	258
10.3.2	Weitere Techniken	261
10.3.3	Debugging und Testen	264
10.4	Zusammenfassung und Ausblick	265
<b>11</b>	<b>Web-Entwicklung mit Rails in der Praxis</b>	<b>267</b>
11.1	Sicherheit	267
11.1.1	SQL-Injection	267
11.1.2	Batch-Update von Modellen	268
11.1.3	Unsichere Dateidownloads	269
11.1.4	Direkteingabe einer URL mit ID	270
11.2	Debugging	270
11.3	Active Record-Tipps	271
11.3.1	Legacy-Datenbanken	271
11.3.2	Modelle ans Schema anpassen	271
11.3.3	Verwendung von Views	272
11.3.4	Andersnamige Primärschlüssel	272
11.3.5	Zusammengesetzte Schlüssel	272
11.3.6	Case-sensitive Strings	273
11.4	Vermeiden von doppelten Logins	273
11.5	GET oder POST?	274
11.6	Performance und Skalierbarkeit	275
11.6.1	Caching	275
11.6.2	Skalierung von Rails-Anwendungen	280
11.7	Datei-Up- und -Downloads	282
11.7.1	Ein Modell für Dateianhänge	283
11.7.2	Ein View zum Upload von Dateien	284
11.7.3	Eine Controller-Action zum Speichern von Dateien	284
11.7.4	Ein View zum Download von Dateianhängen	286



---

11.8 Fehlermeldungen sortieren . . . . .	286
<b>12 Die Anwendung in Produktion bringen . . . . .</b>	<b>291</b>
12.1 Umgebungen in Rails . . . . .	291
12.1.1 Umgebung definieren . . . . .	291
12.1.2 Entwicklung . . . . .	292
12.1.3 Test . . . . .	292
12.1.4 Produktion . . . . .	293
12.2 Datenbank und Webserver . . . . .	294
12.2.1 WEBrick . . . . .	294
12.2.2 Apache und CGI . . . . .	294
12.2.3 Apache und FastCGI . . . . .	295
12.2.4 Apache und Mongrel . . . . .	296
12.2.5 LightTPD und FastCGI . . . . .	297
12.2.6 SCGI – Alternative zu FastCGI . . . . .	298
12.3 Anwendung ausliefern . . . . .	298
12.4 Anwendung warten . . . . .	298
12.4.1 Wartung . . . . .	299
12.4.2 Monitoring . . . . .	300
<b>13 Deployment mit Capistrano (und Subversion) . . . . .</b>	<b>303</b>
13.1 Quickstart: Capistrano in 10 Minuten . . . . .	304
13.1.1 Voraussetzungen . . . . .	304
13.1.2 Installation von Capistrano . . . . .	306
13.1.3 Anwendung Capistrano-ready machen . . . . .	306
13.1.4 Konfiguration . . . . .	307
13.1.5 Setup des entfernten Verzeichnisses . . . . .	309
13.1.6 Erstes Deployment . . . . .	309
13.1.7 Fallstricke . . . . .	311
13.1.8 Nachfolgende Deployments . . . . .	312
13.2 Datenbanksetup und Migration . . . . .	313
13.3 Rollback eines Release . . . . .	314
13.3.1 Rollback mit Datenbankmigration . . . . .	314
13.4 Tasks . . . . .	314
13.4.1 Ausführen von Tasks . . . . .	315
13.4.2 Mehr über run und sudo . . . . .	316
13.4.3 Weitere Task-Helper: delete, put und render . . . . .	317
13.4.4 Transaktionen und Rollbacks . . . . .	317

---

13.4.5	Überschreiben von Standardtasks . . . . .	318
13.4.6	Tasks erweitern: Before- und After-Tasks . . . . .	318
13.5	Variablen . . . . .	319
13.6	FastCGI-Utilities . . . . .	320
13.6.1	Spawner . . . . .	320
13.6.2	Reaper . . . . .	321
13.7	Gemeinsame Dateien – das Shared-Verzeichnis . . . . .	321
13.7.1	Eine „persistente“ Datenbank-Konfiguration . . . . .	321
13.8	Capistrano-Referenz . . . . .	323
13.9	Capistrano-Konfiguration: deploy.rb . . . . .	324
<b>14</b>	<b>Testgetriebene Entwicklung mit Ruby und Test::Unit . . . . .</b>	<b>325</b>
14.1	Unit Tests – eine Definition . . . . .	325
14.2	Ein Beispiel . . . . .	326
14.3	Warum testen? . . . . .	327
14.4	Test::Unit . . . . .	328
14.4.1	Strukturierung von Unit Tests . . . . .	329
14.4.2	Wohin mit den Tests? . . . . .	331
14.4.3	Ausführen der Tests . . . . .	331
14.4.4	Unabhängigkeit von Tests . . . . .	332
14.5	Testgetriebene Softwareentwicklung . . . . .	333
14.5.1	TODO-Listen . . . . .	334
14.5.2	Beispiel . . . . .	335
14.6	Retrospektive . . . . .	336
<b>15</b>	<b>Testgetriebene Entwicklung mit Ruby on Rails . . . . .</b>	<b>337</b>
15.1	Generierte Testklassen . . . . .	337
15.2	Testdatenbank . . . . .	338
15.3	Testausführung über Rake . . . . .	339
15.4	Eine Programmierepisode . . . . .	339
15.4.1	Entwicklung einer Modellklasse . . . . .	340
15.4.2	Entwicklung des Controlllers . . . . .	344
15.4.3	Anpassung des Views . . . . .	345
15.4.4	Geänderte Anforderungen . . . . .	347
15.4.5	Retrospektive . . . . .	353
15.5	Unit Tests – Testen von Modellklassen . . . . .	354
15.5.1	Struktur und Elemente von Modelltests . . . . .	354
15.5.2	Testmethoden . . . . .	355

---

15.5.3	Testdaten – Fixtures . . . . .	356
15.5.4	Transaktionale Fixtures . . . . .	358
15.5.5	Testrezepte für Modelle . . . . .	358
15.6	Funktionale Tests – Testen von Controllern und Views . . . . .	363
15.6.1	Struktur und Elemente von Controller-Tests . . . . .	364
15.6.2	Testmethoden . . . . .	364
15.6.3	Kontrollfluss-Assertions . . . . .	365
15.6.4	Routing-Assertions . . . . .	367
15.6.5	Datencontainer-Assertions . . . . .	368
15.6.6	Template-Assertions . . . . .	369
15.6.7	Testrezepte für Controller und Views . . . . .	371
15.7	Integrationstests . . . . .	374
15.7.1	Test-DSLs . . . . .	376
15.7.2	Sessions . . . . .	377
15.8	Testen von E-Mails und Web Services . . . . .	378
15.9	Mock-Objekte . . . . .	378
15.10	Zusammenfassung . . . . .	379
15.11	Assertions – Übersicht . . . . .	380
	<b>Anhang . . . . .</b>	<b>381</b>
	<b>Literaturverzeichnis . . . . .</b>	<b>391</b>
	<b>Stichwortverzeichnis . . . . .</b>	<b>393</b>