

Inhaltsverzeichnis

1	Einleitung	1
1.1	Zielsetzung	1
1.2	Konventionen	2
1.3	Was ist neu in Rails 3.1?	3
1.4	Aufbau des Buchs	7
1.5	Wer dieses Buch lesen sollte	8
1.6	Wie man dieses Buch lesen sollte	8
1.7	Webseite zum Buch	8
1.8	Danksagungen	9
2	Grundlagen	11
2.1	Paradigmen und Vorgehensweisen	11
2.1.1	Don't repeat yourself	12
2.1.2	Convention over Configuration	13
2.2	Architektur einer Rails-Anwendung	14
2.2.1	Model-View-Controller	14
2.2.2	Zusammenspiel einzelner Schichten	15
2.2.3	REST	17
2.2.4	Umgebungen	19
2.3	Wichtige Kernmodule	20
2.3.1	ActiveRecord	21
2.3.2	ActiveModel	22
2.3.3	ActiveSupport	23
2.3.4	Railties	23
2.3.5	ActionPack	23
2.3.6	ActionMailer	26
2.4	Ruby Version Manager	27
2.4.1	Installation	28
2.4.2	Ruby-Implementierungen installieren	28
2.4.3	Verwalten der Gems	29
2.4.4	RVM aktuell halten	30
2.5	Gem-Verwaltung mit Bundler	30
2.6	Rake	32

2.7	Asset-Pipeline	34
2.7.1	Zugriff auf Asset-Dateien	35
2.7.2	Zusammenführen von Assets	35
2.7.3	Vorverarbeitung	36
2.7.4	Komprimierung	37
2.8	Rails auf der Kommandozeile	37
3	Rails in der Praxis	41
3.1	Einleitung	41
3.1.1	Vorstellung der Beispielanwendung	41
3.1.2	Das Datenmodell	42
3.2	Anwendung anlegen	44
3.3	Anwendungsverzeichnis	45
3.4	Initiale Konfigurationen	48
3.4.1	Startseite anlegen	49
3.4.2	Standardlayout einer Rails-Anwendung	50
3.4.3	Verwalten der Stylesheets und JavaScript-Dateien	51
3.4.4	Anpassen des Layouts	53
3.5	Benutzerauthentifizierung	55
3.5.1	Installation von devise	55
3.5.2	Basiskonfiguration	55
3.5.3	User-Model erstellen	57
3.5.4	Views und Routen	58
3.5.5	Benachrichtigungen	59
3.5.6	Helfer für die Arbeit mit Benutzern	60
3.5.7	Profilseite eines Benutzers	60
3.5.8	Passwortbestätigung entfernen	62
3.5.9	Authentifizierung per Login-Attribut	62
3.6	Links verwalten	65
3.6.1	Model anlegen	66
3.6.2	Beziehungen zwischen Models	67
3.6.3	Anlegen von Controllern und Views	68
3.6.4	Neue Links und Routen anlegen	70
3.6.5	Validierung der Models	72
3.6.6	Gesamtübersicht aller Links	74
3.6.7	Übersichtlichkeit dank Paginierung	77
3.7	Links mit Tags kategorisieren	79
3.7.1	Einführen neuer Models	79
3.7.2	Beziehungen zwischen Tags und Links	79
3.7.3	Verwendung eines virtuellen Attributs	80
3.7.4	Autocomplete mit Ajax	83
3.7.5	Unterschiedliche Ausgabeformate in einem Controller	86
3.8	Fazit	87

4	Alternative Komponenten	89
4.1	Suchen mit Apache Solr	89
4.1.1	Solr in das Beispielprojekt einbinden	89
4.1.2	Suchen im Controller	92
4.1.3	Das Suchformular	93
4.1.4	Indexierung von Daten	94
4.1.5	Highlighting der Suchergebnisse	95
4.1.6	Suche verfeinern mit Facetten	97
4.1.7	Fazit	99
4.2	Rails 3 und NoSQL	100
4.2.1	Wahl der NoSQL-Datenbank	102
4.2.2	MongoDB – Aufbau und Prinzipien	104
4.2.3	Der Objekt-Dokument-Mapper Mongoid	105
4.2.4	Ein bestehendes Model mit Mongoid verwalten	106
4.2.5	Benutzerverwaltung auf MongoDB umstellen	109
4.2.6	Abfragen	110
4.2.7	Observer	111
4.2.8	Mongoid und Sunspot	112
4.2.9	Besonderheiten von Mongoid	112
4.2.10	Andere NoSQL-Datenbanken	114
4.2.11	Fazit	117
5	Tests und Qualitätssicherung	119
5.1	Behavior Driven Development	120
5.1.1	Die Prinzipien	121
5.1.2	Ein üblicher Arbeitsablauf	121
5.1.3	Verbreitete Werkzeuge	123
5.2	Komponententests mit RSpec	123
5.2.1	Grundlagen	123
5.2.2	Aufbau der Tests	124
5.2.3	Ausführen auf der Kommandozeile	128
5.2.4	Anforderungen spezifizieren durch Matcher	131
5.2.5	Mocks und Stubs	135
5.2.6	Testen der Model-Schicht	137
5.2.7	Testen der Controller-Schicht	146
5.2.8	Testen der Helper-Schicht	151
5.2.9	Testen des Routings	155
5.3	Factory Girl als Fixture-Alternative	157
5.3.1	Installation im Beispielprojekt	158
5.3.2	Wie wird es verwendet?	159
5.3.3	Sequenzen – fortlaufende Factories	160
5.3.4	Beziehungen zwischen Factories	161
5.3.5	Fazit	162

5.4	Integrationstests mit Cucumber	162
5.4.1	Cucumber mit Rails verwenden	163
5.4.2	Cucumber-Features erstellen	165
5.4.3	Unser erstes Feature	165
5.4.4	Formulare testen	167
5.4.5	Wiederverwendbare Steps benutzen	167
5.4.6	Ein Bookmark speichern	170
5.4.7	Mehrere Datensätze erstellen	170
5.4.8	Tests durch Tags gruppieren	171
5.4.9	JavaScript testen mit Selenium	173
5.4.10	Fazit	174
5.5	Metriken – Projekte überwachen	175
5.5.1	Gängige Metriken für Rails-Anwendungen	175
5.5.2	Metriken erstellen mit metric_fu	177
5.5.3	Richtig einschätzen und verwenden	178
6	Rails erweitern	181
6.1	Eigene Models mit ActiveRecord erstellen	181
6.1.1	Validierung	182
6.1.2	Serialisierung	186
6.1.3	Verwendung innerhalb der Anwendung	188
6.1.4	In Formularen verwenden	191
6.2	Railties	193
6.2.1	Verwenden der Railties-Schnittstelle	193
6.2.2	Auslagern in ein eigenes RubyGem	195
6.3	Generatoren selbst entwickeln	196
6.3.1	Das Grundgerüst erstellen	197
6.3.2	Templates	197
6.3.3	Hooks zu anderen Generatoren	200
6.3.4	Verwendung auf der Kommandozeile	201
6.3.5	In ein Gem auslagern	201
6.4	Rails-Engines	204
6.4.1	Engine anlegen	205
6.4.2	Aufbau	205
6.4.3	Anwendungsschichten einer Engine	206
6.4.4	Einbinden in die Rails-Anwendung	208
6.4.5	Fazit	209
7	Best Practices	211
7.1	Während der Entwicklung	211
7.1.1	Fat Models – Skinny Controllers	211
7.1.2	Filter in Controllern	213
7.1.3	Häufig verwendete Abfragen hinterlegen	216

7.1.4	Model-Zustände mit Observern verfolgen	218
7.1.5	Datenbank-Seeding	221
7.1.6	Isoliertes Installieren von Gems	222
7.1.7	Das Potenzial von RVM in Projekten ausreizen	222
7.2	Vor Angriffen schützen	223
7.2.1	SQL-Injections bei Ruby on Rails	224
7.2.2	Cross-Site-Scripting verhindern	225
7.2.3	Rechteüberprüfung beim Verwalten von Datensätzen	226
7.2.4	Mass-Assignment von Model-Attributen	228
7.3	Capistrano	231
7.3.1	Konfiguration für die Beispielanwendung	233
7.3.2	Wichtige Tasks	235
7.3.3	Entwicklung eines eigenen Capistrano-Tasks	235
7.3.4	Capistrano-Tasks durch Hooks erweitern	236
7.3.5	Deployments durchführen	237
7.3.6	Staging von Deployments	237
7.3.7	Fazit	240
	Nützliche Weblinks	241
	Glossar	243
	Stichwortverzeichnis	245