



craig WALLS

# SPRING

## IM EINSATZ

3. Auflage



HANSER

Walls  
Spring im Einsatz



### **Bleiben Sie auf dem Laufenden!**

Unser **Computerbuch-Newsletter** informiert Sie monatlich über neue Bücher und Termine. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter:

**[www.hanser-fachbuch.de/newsletter](http://www.hanser-fachbuch.de/newsletter)**





Craig Walls

# Spring im Einsatz

3., überarbeitete Auflage

HANSER

Übersetzung: Frank Langenau, Chemnitz

Titel der Originalausgabe: „Spring in Action“, 5th Edition, © 2019 by Manning Publications Co.

Authorized translation of the English edition. This translation is published and sold by permission of Manning Publications, the owner of all rights to publish and sell the same.

Alle in diesem Buch enthaltenen Informationen, Verfahren und Darstellungen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autor, Übersetzer und Verlag übernehmen infolgedessen keine juristische Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieser Informationen – oder Teilen davon – entsteht.

Ebenso übernehmen Autor, Übersetzer und Verlag keine Gewähr dafür, dass beschriebene Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Buch berechtigt deshalb auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

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.

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdruckes und der Vervielfältigung des Buches, oder Teilen daraus, vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren) – auch nicht für Zwecke der Unterrichtsgestaltung – reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Copyright für die deutsche Ausgabe:

© 2020 Carl Hanser Verlag, [www.hanser-fachbuch.de](http://www.hanser-fachbuch.de)

Lektorat: Brigitte Bauer-Schiewek

Copy editing: Petra Kienle, Fürstenfeldbruck

Layout: Manuela Treindl, Fürth

Umschlagdesign: Marc Müller-Bremer, München, [www.rebranding.de](http://www.rebranding.de)

Umschlagrealisation: Max Kostopoulos

Titelmotiv: © Stephan Rönigk

Datenbelichtung, Druck und Bindung: Kösel, Krugzell

Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702

Printed in Germany

Print-ISBN: 978-3-446-45512-2

E-Book-ISBN: 978-3-446-45731-7

E-Pub-ISBN: 978-3-446-46323-3

# Inhalt

<b>Vorwort</b> .....	<b>XIII</b>
<b>Danksagungen</b> .....	<b>XV</b>
<b>Über dieses Buch</b> .....	<b>XVII</b>
<b>1 Erste Schritte mit Spring</b> .....	<b>3</b>
1.1 Was ist Spring? .....	4
1.2 Eine Spring-Anwendung initialisieren .....	6
1.2.1 Ein Spring-Projekt mit der Spring Tool Suite initialisieren .....	7
1.2.2 Die Spring-Projektstruktur untersuchen .....	11
1.3 Eine Spring-Anwendung schreiben .....	17
1.3.1 Web-Requests verarbeiten .....	17
1.3.2 Die View definieren .....	19
1.3.3 Den Controller testen .....	20
1.3.4 Die Anwendung erstellen und ausführen .....	21
1.3.5 Spring Boot DevTools kennenlernen .....	23
1.3.6 Rückblick .....	25
1.4 Die Spring-Landschaft im Überblick .....	26
1.4.1 Der Core des Spring Frameworks .....	26
1.4.2 Spring Boot .....	27
1.4.3 Spring Data .....	27
1.4.4 Spring Security .....	28
1.4.5 Spring Integration und Spring Batch .....	28
1.4.6 Spring Cloud .....	28
1.5 Zusammenfassung .....	29
<b>2 Webanwendungen entwickeln</b> .....	<b>31</b>
2.1 Informationen anzeigen .....	31
2.1.1 Die Domäne einrichten .....	33
2.1.2 Eine Controller-Klasse erstellen .....	34
2.1.3 Die View entwerfen .....	38
2.2 Formularübermittlungen verarbeiten .....	43
2.3 Formulareingaben validieren .....	49
2.3.1 Validierungsregeln deklarieren .....	49
2.3.2 Validierung bei Formularbindung durchführen .....	52

2.3.3	Validierungsfehler anzeigen	53
2.4	Mit View-Controllern arbeiten	55
2.5	Eine View-Template-Bibliothek auswählen	57
2.5.1	Vorlagen zwischenspeichern	59
2.6	Zusammenfassung	60
<b>3</b>	<b>Mit Daten arbeiten</b>	<b>61</b>
3.1	Daten mit JDBC lesen und schreiben	61
3.1.1	Die Domäne für Persistenz anpassen	64
3.1.2	Mit JdbcTemplate arbeiten	65
3.1.3	Ein Schema definieren und Daten im Voraus laden	69
3.1.4	Daten einfügen	72
3.2	Daten mit Spring Data JPA persistent speichern	81
3.2.1	Spring Data JPA zum Projekt hinzufügen	81
3.2.2	Die Domäne als Entitäten annotieren	82
3.2.3	JPA-Repositories deklarieren	86
3.2.4	JPA-Repositories anpassen	87
3.3	Zusammenfassung	90
<b>4</b>	<b>Zugriffskontrolle mit Spring Security</b>	<b>91</b>
4.1	Spring Security aktivieren	91
4.2	Spring Security konfigurieren	94
4.2.1	Speicherinterner Benutzerspeicher	96
4.2.2	JDBC-basierter Benutzerspeicher	97
4.2.3	LDAP-gestützter Benutzerspeicher	100
4.2.4	Benutzerauthentifizierung anpassen	104
4.3	Webanfragen sichern	112
4.3.1	Anfragen sichern	112
4.3.2	Eine eigene Anmeldeseite erstellen	115
4.3.3	Abmelden	118
4.3.4	CSRF-Angriffe verhindern	118
4.4	Den Benutzer ermitteln	120
4.5	Zusammenfassung	122
<b>5</b>	<b>Mit Konfigurationseigenschaften arbeiten</b>	<b>123</b>
5.1	Automatische Konfiguration optimieren	124
5.1.1	Die Umgebungsabstraktion von Spring verstehen	124
5.1.2	Eine Datenquelle konfigurieren	126
5.1.3	Den eingebetteten Server konfigurieren	128
5.1.4	Protokollieren konfigurieren	129
5.1.5	Spezielle Eigenschaftswerte verwenden	130
5.2	Eigene Konfigurationseigenschaften erzeugen	131
5.2.1	Holder für Konfigurationseigenschaften definieren	134
5.2.2	Metadaten von Konfigurationseigenschaften deklarieren	136
5.3	Konfigurieren mit Profilen	139
5.3.1	Profilspezifische Eigenschaften definieren	140

5.3.2	Profile aktivieren .....	141
5.3.3	Beans mit Profilen bedingt erstellen .....	142
5.4	Zusammenfassung .....	144
<b>6</b>	<b>REST-Dienste erstellen.....</b>	<b>147</b>
6.1	RESTful Controller programmieren .....	148
6.1.1	Daten vom Server abrufen.....	150
6.1.2	Daten an den Server senden .....	155
6.1.3	Daten auf dem Server aktualisieren.....	156
6.1.4	Daten vom Server löschen.....	159
6.2	Hypermedia aktivieren.....	160
6.2.1	Hyperlinks hinzufügen .....	162
6.2.2	Ressourcenassembler erstellen.....	165
6.2.3	Eingebettete Beziehungen benennen .....	169
6.3	Datengestützte Dienste aktivieren .....	171
6.3.1	Ressourcenpfade und Beziehungsnamen anpassen .....	173
6.3.2	Paging und Sortieren .....	175
6.3.3	Benutzerdefinierte Endpunkte hinzufügen .....	177
6.3.4	Benutzerdefinierte Hyperlinks zu Spring-Data-Endpunkten hinzufügen ..	179
6.4	Zusammenfassung .....	180
<b>7</b>	<b>REST-Dienste konsumieren.....</b>	<b>181</b>
7.1	REST-Endpunkte mit RestTemplate konsumieren.....	182
7.1.1	Ressourcen mit GET abrufen.....	184
7.1.2	Ressourcen mit PUT senden .....	185
7.1.3	Ressourcen mit DELETE löschen .....	186
7.1.4	Ressourcendaten per POST senden .....	186
7.2	Mit Traverson in REST APIs navigieren.....	187
7.3	Zusammenfassung .....	189
<b>8</b>	<b>Nachrichten asynchron senden .....</b>	<b>191</b>
8.1	Nachrichten mit JMS senden .....	192
8.1.1	JMS einrichten .....	192
8.1.2	Nachrichten mit JmsTemplate senden .....	194
8.1.3	JMS-Nachrichten empfangen.....	202
8.2	Mit RabbitMQ und AMQP arbeiten.....	206
8.2.1	RabbitMQ zu Spring hinzufügen .....	207
8.2.2	Nachrichten mit RabbitTemplate senden.....	208
8.2.3	Nachrichten von RabbitMQ empfangen .....	212
8.3	Messaging mit Kafka .....	217
8.3.1	Spring für Kafka-Messaging einrichten.....	218
8.3.2	Nachrichten mit KafkaTemplate senden .....	219
8.3.3	Kafka-Listener schreiben.....	221
8.4	Zusammenfassung .....	222



<b>9</b>	<b>Spring integrieren</b>	<b>223</b>
9.1	Einen einfachen Integrationsfluss deklarieren	224
9.1.1	Integrationsflüsse mit XML definieren	225
9.1.2	Integrationsflüsse in Java konfigurieren	227
9.1.3	Die DSL-Konfiguration von Spring Integration verwenden	229
9.2	Die Landschaft von Spring Integration im Überblick	231
9.2.1	Nachrichtenkanäle	232
9.2.2	Filter	233
9.2.3	Transformer	234
9.2.4	Router	236
9.2.5	Splitter	237
9.2.6	Dienstaktivatoren	240
9.2.7	Gateways	242
9.2.8	Kanaladapter	243
9.2.9	Endpunktmodule	245
9.3	Einen E-Mail-Integrationsfluss erstellen	246
9.4	Zusammenfassung	252
<b>10</b>	<b>Einführung in Reactor</b>	<b>255</b>
10.1	Reaktive Programmierung verstehen	256
10.1.1	Reactive Streams definieren	257
10.2	Erste Schritte mit Reactor	260
10.2.1	Reaktive Datenflüsse grafisch darstellen	261
10.2.2	Reactor-Abhängigkeiten hinzufügen	262
10.3	Allgemeine reaktive Operationen anwenden	263
10.3.1	Reaktive Typen erstellen	264
10.3.2	Reaktive Typen kombinieren	268
10.3.3	Reaktive Streams transformieren und filtern	272
10.3.4	Logische Operationen auf reaktiven Typen ausführen	281
10.4	Zusammenfassung	283
<b>11</b>	<b>Reaktive APIs entwickeln</b>	<b>285</b>
11.1	Mit Spring WebFlux arbeiten	285
11.1.1	Einführung in Spring WebFlux	287
11.1.2	Reaktive Controller schreiben	288
11.2	Funktionale Anfrage-Handler definieren	293
11.3	Reaktive Controller testen	296
11.3.1	GET-Anfragen testen	296
11.3.2	POST-Anfragen testen	299
11.3.3	Mit einem Live-Server testen	300
11.4	REST APIs reaktiv konsumieren	301
11.4.1	Ressourcen mit GET-Anfragen abrufen	302
11.4.2	Ressourcen senden	304
11.4.3	Ressourcen löschen	305
11.4.4	Fehler behandeln	306
11.4.5	Anfragen vermitteln	308

11.5	Reaktive Web-APIs sichern .....	309
11.5.1	Reaktive Websicherheit konfigurieren .....	310
11.5.2	Einen reaktiven Dienst für Benutzerdetails konfigurieren .....	311
11.6	Zusammenfassung .....	313
<b>12</b>	<b>Daten reaktiv persistent speichern .....</b>	<b>315</b>
12.1	Die reaktive Geschichte von Spring Data .....	316
12.1.1	Reaktives Spring Data auf den Punkt gebracht .....	316
12.1.2	Zwischen reaktiven und nichtreaktiven Typen konvertieren .....	317
12.1.3	Reaktive Repositories entwickeln .....	319
12.2	Mit reaktiven Cassandra-Repositories arbeiten .....	319
12.2.1	Spring Data Cassandra aktivieren .....	320
12.2.2	Cassandra-Datenmodellierung verstehen .....	323
12.2.3	Domänentypen für Cassandra-Persistenz abbilden .....	323
12.2.4	Reaktive Cassandra-Repositories programmieren .....	329
12.3	Reaktive MongoDB-Repositories programmieren .....	332
12.3.1	Spring Data MongoDB aktivieren .....	332
12.3.2	Domänentypen auf Dokumente abbilden .....	334
12.3.3	Reaktive MongoDB-Repository-Schnittstellen schreiben .....	338
12.4	Zusammenfassung .....	341
<b>13</b>	<b>Service-Discovery .....</b>	<b>345</b>
13.1	Denken in Microservices .....	346
13.2	Eine Dienstregistrierung einrichten .....	348
13.2.1	Eureka konfigurieren .....	352
13.2.2	Eureka skalieren .....	355
13.3	Dienste registrieren und entdecken .....	357
13.3.1	Eureka-Clienteigenschaften konfigurieren .....	358
13.3.2	Dienste konsumieren .....	359
13.4	Zusammenfassung .....	365
<b>14</b>	<b>Konfiguration verwalten .....</b>	<b>367</b>
14.1	Konfiguration teilen .....	368
14.2	Config Server ausführen .....	369
14.2.1	Config Server aktivieren .....	370
14.2.2	Das Konfigurations-Repository füllen .....	373
14.3	Gemeinsame Konfigurationen konsumieren .....	376
14.4	Anwendungs- und profilspezifische Eigenschaften bereitstellen .....	378
14.4.1	Anwendungsspezifische Eigenschaften bereitstellen .....	378
14.4.2	Eigenschaften von Profilen bereitstellen .....	379
14.5	Konfigurationseigenschaften geheim halten .....	381
14.5.1	Eigenschaften in Git verschlüsseln .....	382
14.5.2	Geheimnisse in Vault speichern .....	385
14.6	Konfigurationseigenschaften im laufenden Betrieb aktualisieren .....	390
14.6.1	Konfigurationseigenschaften manuell aktualisieren .....	391
14.6.2	Konfigurationseigenschaften automatisch aktualisieren .....	393

14.7 Zusammenfassung .....	401
<b>15 Fehler und Latenzzeiten behandeln .....</b>	<b>403</b>
15.1 Trennschalter im Überblick .....	403
15.2 Trennschalter deklarieren .....	405
15.2.1 Latenz reduzieren .....	408
15.2.2 Schwellenwerte für Trennschalter verwalten .....	409
15.3 Fehler überwachen .....	411
15.3.1 Das Hystrix-Dashboard – eine Einführung .....	412
15.3.2 Hystrix-Threadpools .....	415
15.4 Hystrix-Streams aggregieren .....	416
15.5 Zusammenfassung .....	418
<b>16 Mit Spring Boot Actuator arbeiten .....</b>	<b>421</b>
16.1 Actuator im Überblick .....	421
16.1.1 Den Basispfad von Actuator konfigurieren .....	423
16.1.2 Actuator-Endpunkte aktivieren und deaktivieren .....	424
16.2 Actuator-Endpunkte konsumieren .....	425
16.2.1 Wichtige Anwendungsinformationen abrufen .....	426
16.2.2 Konfigurationsdetails ansehen .....	429
16.2.3 Anwendungsaktivität anzeigen .....	437
16.2.4 Laufzeit-Metriken erfassen .....	440
16.3 Actuator anpassen .....	443
16.3.1 Informationen zum Endpunkt/info beisteuern .....	443
16.3.2 Benutzerdefinierte Zustandsindikatoren definieren .....	448
16.3.3 Benutzerdefinierte Metriken registrieren .....	449
16.3.4 Benutzerdefinierte Endpunkte erstellen .....	451
16.4 Actuator sichern .....	454
16.5 Zusammenfassung .....	456
<b>17 Spring verwalten .....</b>	<b>457</b>
17.1 Spring Boot Admin verwenden .....	457
17.1.1 Einen Admin-Server erstellen .....	458
17.1.2 Admin-Clients registrieren .....	460
17.2 Admin-Server im Detail .....	464
17.2.1 Integritätsdaten und allgemeine Anwendungsinformationen anzeigen .....	465
17.2.2 Schlüsselmetriken überwachen .....	467
17.2.3 Umgebungseigenschaften untersuchen .....	468
17.2.4 Protokollierungsstufen anzeigen und festlegen .....	469
17.2.5 Threads überwachen .....	470
17.2.6 HTTP-Anfragen verfolgen .....	471
17.3 Den Admin-Server sichern .....	473
17.3.1 Anmelden beim Admin-Server aktivieren .....	473
17.3.2 Beim Actuator authentifizieren .....	474
17.4 Zusammenfassung .....	475

<b>18 Spring mit JMX überwachen</b> .....	<b>477</b>
18.1 Mit Actuator-MBeans arbeiten .....	477
18.2 Eigene MBeans erstellen .....	480
18.3 Benachrichtigungen senden .....	482
18.4 Zusammenfassung .....	483
<b>19 Spring bereitstellen</b> .....	<b>485</b>
19.1 Bereitstellungsoptionen abwägen .....	486
19.2 WAR-Dateien erstellen und bereitstellen .....	487
19.3 JAR-Dateien zu Cloud Foundry verschieben .....	489
19.4 Spring Boot in einem Docker-Container ausführen .....	492
19.5 Der Weg ist das Ziel .....	496
19.6 Zusammenfassung .....	497
<b>A Bootstrapping von Spring-Anwendungen</b> .....	<b>499</b>
A.1 Ein Projekt mit Spring Tool Suite initialisieren .....	499
A.2 Ein Projekt mit IntelliJ IDEA initialisieren .....	503
A.3 Ein Projekt mit NetBeans initialisieren .....	507
A.4 Ein Projekt unter start.spring.io initialisieren .....	511
A.5 Ein Projekt von der Befehlszeile initialisieren .....	515
A.5.1 curl und die Initializr API .....	515
A.5.2 Befehlszeilenoberfläche von Spring Boot .....	517
A.6 Spring-Anwendungen mit einem Meta-Framework erstellen .....	519
A.7 Projekte erstellen und ausführen .....	519
<b>Stichwortverzeichnis</b> .....	<b>521</b>



# Vorwort

Nachdem ich fast 15 Jahre mit Spring gearbeitet und mehrere Ausgaben dieses Buches geschrieben habe (ganz zu schweigen von „*Spring Boot in Action*“), sollte man meinen, dass sich kaum noch etwas Aufregendes und Neues über Spring sagen lässt, wenn es um das Vorwort für dieses Buch geht. Aber wie so oft sieht es in der Realität ganz anders aus!

Jedes einzelne Release von Spring, Spring Boot und allen anderen Projekten im Spring-Ökosystem schafft neue erstaunliche Möglichkeiten, die den Spaß an der Entwicklung von Anwendungen wieder aufleben lassen. Mit dem Release 5.0 von Spring und dem Release 2.0 von Spring Boot gibt es so viel mehr Spring zu genießen, dass es ein Kinderspiel war, eine weitere Ausgabe von *Spring im Einsatz* zu schreiben.

Das Großartige an Spring 5 ist die Unterstützung für reaktive Programmierung, unter anderem für Spring WebFlux, ein brandneues reaktives Web-Framework, dessen Programmiermodell sich an Spring MVC orientiert. Damit können Entwickler Webanwendungen schaffen, die sich besser skalieren lassen und weniger Threads effektiver nutzen. In Richtung des Backends einer Spring-Anwendung erlaubt es die neueste Edition von Spring Data, reaktive, nicht blockierende Daten-Repositories aufzubauen. Und alles dies baut auf Project Reactor auf, einer Java-Bibliothek für das Arbeiten mit reaktiven Typen.

Zusätzlich zu den neuen reaktiven Programmierfeatures von Spring 5 bietet Spring Boot 2 nun sogar mehr Unterstützung als je zuvor für die Autokonfiguration sowie einen vollständig neu konzipierten Actuator, mit dem sich eine laufende Anwendung inspizieren und manipulieren lässt.

Da Entwickler zudem ihre monolithischen Anwendungen in diskrete Microservices aufteilen wollen, bietet Spring Cloud Einrichtungen, die es erleichtern, Microservices zu konfigurieren, zu erkennen und sie widerstandsfähiger gegen Ausfälle zu machen.

Ich freue mich, sagen zu können, dass diese fünfte Ausgabe von *Spring im Einsatz* – hier vorliegend als Übersetzung in der dritten Auflage – alle diese und noch mehr Themen abdeckt! Wenn Sie ein erfahrener Veteran von Spring sind, wird *Spring im Einsatz* Ihr Leitfaden für alles Neue sein, das Spring zu bieten hat. Wenn Sie andererseits neu in Spring einsteigen, dann gibt es keinen besseren Zeitpunkt als jetzt, um richtig loszulegen. Die ersten Kapitel bringen Sie im Handumdrehen an den Start!

Die 15 Jahre Arbeit mit Spring sind eine spannende Zeit gewesen. Und da nun diese Edition von *Spring im Einsatz* vor Ihnen liegt, bin ich versessen darauf, diese Begeisterung mit Ihnen zu teilen!



# Danksagungen

Zu den erstaunlichsten Dingen bei Spring und Spring Boot ist zu nennen, dass sie automatisch alle grundlegenden Installationen für eine Anwendung bereitstellen, wodurch Sie sich als Entwickler vorrangig auf die Logik konzentrieren können, die Ihre Anwendung im Speziellen ausmacht. Leider gibt es keine solchen magischen Hilfsmittel, um ein Buch zu schreiben. Oder etwa doch?

Bei Manning haben mehrere Leute ihre magischen Kräfte entfaltet, um sicherzustellen, dass dieses Buch das Beste wird, was möglich ist. Vielen Dank insbesondere an Jenny Stout, meine Entwicklungsredakteurin, und das Produktteam, darunter Projektleiterin Janet Vail, die Copyeditoren Andy Carroll und Frances Buran sowie das Korrektorat mit Katie Tennant und Melody Dolab. Dank auch dem Fachlektor Joshua White, der gründlich und hilfreich war.

In dieser Zeit haben wir auch Feedback von mehreren Gutachtern erhalten, die dafür gesorgt haben, den Kurs zu halten und die richtigen Themen abzudecken. Dafür danke ich Andrea Barisone, Arnaldo Ayala, Bill Fly, Colin Joyce, Daniel Vaughan, David Witherspoon, Eddu Melendez, Iain Campbell, Jetro Coenradie, John Gunvaldson, Markus Matzker, Nick Rakochy, Nusry Firdousi, Piotr Kafel, Raphael Villela, Riccardo Noviello, Sergio Fernandez Gonzalez, Sergiy Pylypets, Thiago Presa, Thorsten Weber, Waldemar Modzelewski, Yagiz Erkan und Željko Trogrlić.

Wie immer gäbe es absolut keinen Grund, dieses Buch zu schreiben, wenn da nicht die erstaunliche Arbeit des Spring-Entwicklerteams wäre. Ich kann nur darüber staunen, was es geschaffen hat und wie wir den Entwicklungsstil von Software immer wieder verändern.

Ein großer Dank geht an meine Mitstreiter auf der No Fluff/Just Stuff-Tour. Ich lerne weiterhin so viel von jedem von euch! Besonders danken möchte ich Brian Sletten, Nate Schutta und Ken Kousen für die Gespräche und E-Mails über Spring, die zur Gestaltung dieses Buches beigetragen haben.

Nochmals vielen Dank an die Phönizier. Ihr wisst, was ihr getan habt.

An meine wundervolle Frau Raymie, die Liebe meines Lebens, meinen süßesten Traum und meine Inspiration gerichtet: Danke für dein Engagement und dafür, dass du dich mit einem weiteren Buchprojekt abgefunden hast. Und an meine süßen und wundervollen Mädchen, Maisy und Madi: Ich bin so stolz auf euch und auf die erstaunlichen jungen Damen, die ihr einmal sein werdet. Ich liebe euch alle mehr, als ihr es euch vorstellen könnt oder ich es auszudrücken vermag.





# Über dieses Buch

*Spring im Einsatz* soll Sie in die Lage versetzen, erstaunliche Anwendungen mit dem Spring Framework, Spring Boot und einer breiten Palette von Ergänzungstools des Spring-Ökosystems zu erstellen. Zunächst erfahren Sie, wie Sie webbasierte, datenbankgestützte Java-Anwendungen mit Spring und Spring Boot entwickeln. Anschließend geht es über die Grundlagen hinaus und es wird gezeigt, wie Sie die Integration mit anderen Anwendungen realisieren, mit reaktiven Typen programmieren und dann eine Anwendung in diskrete Microservices aufteilen. Schließlich wird erörtert, wie Sie eine Anwendung für die Bereitstellung fit machen. Obwohl alle Projekte im Spring-Ökosystem eine ausgezeichnete Dokumentation bieten, gibt Ihnen dieses Buch etwas, was Sie in den Referenzdokumentationen nicht finden: einen praktischen, projektgetriebenen Leitfaden, um die Elemente von Spring im Rahmen einer realen Anwendung zusammenzubringen.

## Wer dieses Buch lesen sollte

Die vorliegende Ausgabe von *Spring im Einsatz*, richtet sich an Java-Entwickler, die erste Schritte mit Spring Boot und dem Spring Framework unternehmen möchten, sowie an erfahrene Spring-Entwickler, die über die Grundlagen hinausgehen und die neuesten Features von Spring kennenlernen wollen.

## Wie dieses Buch organisiert ist: eine Roadmap

Das Buch umfasst 19 Kapitel, die in fünf Teile gegliedert sind. **Teil I** befasst sich mit den grundlegenden Themen für das Erstellen von Anwendungen:

- *Kapitel 1* führt Spring und Spring Boot ein und zeigt, wie Sie ein Spring-Projekt initialisieren. In diesem Kapitel unternehmen Sie die ersten Schritte und erstellen eine Spring-Anwendung, die Sie im weiteren Verlauf des Buches erweitern und vervollständigen werden.
- *Kapitel 2* erläutert, wie Sie die Web-Ebene einer Anwendung mit Spring MVC aufbauen. In diesem Kapitel erstellen Sie Controller, die Webanfragen behandeln, und Views, die Informationen im Webbrowser darstellen.
- *Kapitel 3* beschäftigt sich mit dem Backend einer Spring-Anwendung, wo die Daten in einer relationalen Datenbank persistent gespeichert werden.
- In *Kapitel 4* nutzen Sie Spring Security, um Benutzer zu authentifizieren und nicht autorisierten Zugriff auf eine Anwendung zu verhindern.
- *Kapitel 5* macht deutlich, wie Sie eine Spring-Anwendung mit Spring-Boot-Konfigurationseigenschaften konfigurieren. Außerdem lernen Sie, wie Sie eine Konfiguration mithilfe von Profilen selektiv anwenden.

Die Themen in **Teil II** helfen Ihnen, wenn Sie Ihre Spring-Anwendung mit anderen Anwendungen integrieren:

- *Kapitel 6* erweitert die in Kapitel 2 begonnene Diskussion zu Spring MVC und zeigt, wie sich REST APIs in Spring schreiben lassen.
- *Kapitel 7* tauscht die Rollen gegenüber Kapitel 6 und zeigt, wie eine Spring-Anwendung eine REST API konsumieren kann.
- *Kapitel 8* befasst sich mit asynchroner Kommunikation, damit eine Spring-Anwendung per Java Message Service, RabbitMQ oder Kafka Nachrichten sowohl senden als auch empfangen kann.
- In *Kapitel 9* geht es um deklarative Anwendungsintegration mit dem Spring-Integrations-Projekt.

**Teil III** ist der neuen Unterstützung für reaktive Programmierung in Spring gewidmet:

- *Kapitel 10* stellt Project Reactor vor, die reaktive Programmierbibliothek, die die reaktiven Features von Spring 5 untermauert.
- *Kapitel 11* beschäftigt sich noch einmal mit der REST-API-Entwicklung und führt Spring WebFlux ein, ein neues Web-Framework, das sich stark an Spring MVC orientiert, dabei aber ein neues reaktives Modell für die Web-Entwicklung bietet.
- In *Kapitel 12* werfen wir einen Blick darauf, wie sich reaktive Datenpersistenz mit Spring Data programmieren lässt, um Daten in und aus den Datenbanken Cassandra und Mongo zu schreiben und zu lesen.

**Teil IV** zerlegt das monolithische Anwendungsmodell und führt Sie in die Entwicklung mit Spring Cloud und Microservices ein:

- *Kapitel 13* befasst sich mit dem Erkennen von Diensten, wobei Sie Spring mit der Eureka-Registrierung von Netflix verwenden, um Spring-basierte Microservices sowohl zu registrieren als auch zu erkennen.
- In *Kapitel 14* zentralisieren Sie die Anwendungskonfiguration auf einem Konfigurations-server, der die Konfiguration für mehrere Microservices gemeinsam nutzt.
- *Kapitel 15* führt das Trennschalter-Muster (Circuit Breaker) mit Hystrix ein, das es ermöglicht, Microservices für den Fehlerfall robuster zu machen.

In **Teil V** bereiten Sie eine Anwendung für die Produktion vor und lernen, wie Sie sie bereitstellen:

- *Kapitel 16* stellt den Spring Boot Actuator vor, eine Erweiterung zu Spring Boot, mit der sich die Interna einer laufenden Spring-Anwendung als REST-Endpunkte zugänglich machen lassen.
- In *Kapitel 17* sehen Sie, wie Sie mit Spring Boot Admin eine benutzerfreundliche browser-basierte administrative Anwendung auf Actuator aufsetzen können.
- *Kapitel 18* erläutert, wie sich Spring-Beans als JMX MBeans zugänglich machen und konsumieren lassen.
- Abschließend zeigt *Kapitel 19*, wie Sie Ihre Spring-Anwendung in den verschiedensten Produktionsumgebungen bereitstellen.

Im Allgemeinen sollten Entwickler, die in Spring einsteigen, mit Kapitel 1 beginnen und alle Kapitel nacheinander durcharbeiten. Erfahrene Spring-Entwickler ziehen es vielleicht vor, gleich mit dem Thema zu beginnen, das sie vorrangig interessiert. Es sei aber darauf

hingewiesen, dass jedes Kapitel auf dem vorhergehenden aufbaut, sodass möglicherweise der Kontext unklar ist, wenn Sie gleich in der Mitte des Buches einsteigen.

## Über den Code

Dieses Buch enthält viele Beispiele im Quellcode sowohl in nummerierten Listings als auch in Form von Codefragmenten, die in den laufenden Text eingefügt sind. In beiden Fällen ist der Quellcode in *Einer-Schreibmaschinenschrift-wie-dieser* formatiert, um ihn vom normalen Text zu trennen. Manchmal ist der Code auch **fett** gedruckt, um ihn von Code abzuheben, der sich gegenüber vorherigen Schritten im Kapitel geändert hat, beispielsweise wenn ein neues Feature zu einer schon vorhandenen Codezeile hinzukommt.

Der Quellcode zu den Beispielen in diesem Buch steht auf der Website des Verlages der Originalausgabe unter [www.manning.com/books/spring-in-action-fifth-edition](http://www.manning.com/books/spring-in-action-fifth-edition) sowie im GitHub-Konto des Autors unter [github.com/habuma/spring-in-action-5-samples](https://github.com/habuma/spring-in-action-5-samples) zum Download bereit.

## Buchforum

Beim Kauf dieser Ausgabe von *Spring im Einsatz* erhalten Sie kostenfreien Zugriff auf ein privates Web-Forum unter Leitung von Manning Publications, in dem Sie Kommentare zum Buch abgeben, technische Fragen stellen und Hilfe vom Autor und von anderen Benutzern erhalten können. Um auf das Forum zuzugreifen, besuchen Sie <https://forums.manning.com/forums/spring-in-action-fifth-edition>. Mehr über die Foren von Manning und die Verhaltensregeln erfahren Sie auch unter <https://forums.manning.com/forums/about>.

Manning engagiert sich für die Leser, um ihnen einen Treffpunkt zu bieten, an dem ein sinnvoller Dialog zwischen Lesern untereinander und zwischen dem Leser und dem Autor stattfinden kann. Es handelt sich dabei nicht um eine Verpflichtung dem Autor gegenüber, in einem bestimmten Umfang mitzuwirken, wobei der Beitrag zum Forum freiwillig (und unbezahlt) bleibt. Stellen Sie doch dem Autor herausfordernde Fragen, damit sein Interesse nicht verloren geht! Das Forum und die Archive früherer Diskussionen sind über Website des Verlages zugänglich, solange das Buch lieferbar ist.

## Andere Online-Quellen

Brauchen Sie zusätzliche Hilfe?

- Die Spring-Website bietet unter <https://spring.io/guides> mehrere nützliche Leitfäden (von denen der Autor selbst einige geschrieben hat).
- Die Tags für Spring auf StackOverflow (<https://stackoverflow.com/questions/tagged/spring>) und Spring Boot auf StackOverflow sind empfehlenswerte Orte rund um das Thema Spring, um Fragen zu stellen und anderen zu helfen. Gleichzeitig ist es ein guter Weg, um mehr über Spring zu lernen, wenn man anderen bei der Beantwortung ihrer Spring-Fragen hilft!

## Über den Autor

CRAIG WALLS ist leitender Ingenieur bei Pivotal. Er ist ein eifriger Förderer des Spring Frameworks, ist häufig in lokalen Benutzergruppen und Konferenzen anzutreffen und schreibt über Spring. Wenn er nicht gerade mit Code um sich wirft, plant Craig seine nächste Reise nach Disney World oder Disneyland und verbringt so viel Zeit wie möglich mit seiner Frau, seinen beiden Töchtern, zwei Vögeln und drei Hunden.





# Teil I: Die Grundzüge von Spring

In Teil I dieses Buches schreiben Sie zum Einstieg eine Spring-Anwendung und lernen dabei die Grundlagen von Spring kennen.

- *Kapitel 1* gibt Ihnen einen kurzen Überblick über die wesentlichen Elemente von Spring und Spring Boot und zeigt, wie Sie ein Spring-Projekt initialisieren, während Sie Taco Cloud, Ihre erste Spring-Anwendung, erstellen.
- In *Kapitel 2* dringen Sie tiefer in Spring MCV ein und lernen, wie Sie Modelldaten im Browser darstellen und wie Sie Formulareingaben verarbeiten und auf Gültigkeit überprüfen. Außerdem bekommen Sie einige Tipps zur Auswahl einer View-Vorlagenbibliothek.
- In *Kapitel 3* starten Sie die Taco-Cloud-Anwendung mit Datenpersistenz aus. Dabei lernen Sie die JDBC-Vorlage von Spring kennen und erfahren, wie Sie Daten einfügen und wie Sie JPA-Repositories mit Spring Data deklarieren.
- *Kapitel 4* befasst sich mit der Sicherheit für Ihre Spring-Anwendung. Dabei geht es um die automatische Konfiguration von Spring Security, das Definieren benutzerdefinierter Benutzerspeicher, die Anpassung der Anmeldeseite und die Absicherung gegen CSRF-(Cross-Site-Request-Forgery-)Angriffe.
- Zum Abschluss von Teil I geht *Kapitel 5* auf Konfigurationseigenschaften ein. Hier lernen Sie, wie Sie automatisch konfigurierte Beans optimieren, Konfigurationseigenschaften auf Anwendungskomponenten anwenden und mit Spring-Profilen arbeiten.



# 1

## Erste Schritte mit Spring



### Die Themen dieses Kapitels:

- Die Grundzüge von Spring und Spring Boot
- Ein Spring-Projekt initialisieren
- Die Spring-Landschaft im Überblick

Obwohl der griechische Philosoph Heraklit nicht gerade als Softwareentwickler bekannt war, schien er das Thema gut im Griff zu haben. Er wird mit den Worten zitiert: »Die einzige Konstante ... ist die Veränderung.« In dieser Aussage steckt eine grundlegende Wahrheit der Softwareentwicklung.

Heute entwickeln wir Anwendungen in einer anderen Art und Weise als vor einem Jahr, vor fünf Jahren, vor zehn Jahren und zweifellos vor 15 Jahren, als Rod Johnson in seinem Buch *Expert One-on-One J2EE Design and Development* (Wrox, 2002, <http://mng.bz/oVjy>) eine anfängliche Form des Spring Frameworks eingeführt hat.

Damals hat man vor allem browserbasierte Webanwendungen entwickelt, die von relationalen Datenbanken unterstützt wurden. Auch wenn diese Art der Entwicklung immer noch eine Rolle spielt und Spring für derartige Anwendungen gut gerüstet ist, sind wir heute auch interessiert an der Entwicklung von Anwendungen, die aus Microservices bestehen und darauf ausgelegt sind, Daten dauerhaft in der Cloud in verschiedensten Datenbanken zu speichern. Zudem zielt ein neu erwachtes Interesse an reaktiver Programmierung darauf ab, mit nicht blockierenden Operationen größere Skalierbarkeit und verbesserte Performance zu erreichen.

Mit dem Voranschreiten der Softwareentwicklung hat sich auch das Spring Framework geändert, um modernen Entwicklungsansprüchen gerecht zu werden. Dazu gehören auch Microservices und reaktive Programmierung. Mit der Einführung von Spring Boot tritt Spring auch an, um sein eigenes Entwicklungsmodell zu vereinfachen.

Egal, ob Sie eine einfache datenbankgestützte Webanwendung entwickeln oder eine moderne Anwendung um Microservices herum aufbauen – das Spring-Framework hilft Ihnen, Ihre Ziele zu erreichen. Dieses Kapitel ist der erste Schritt auf einer Tour durch die moderne Anwendungsentwicklung mit Spring.



## ■ 1.1 Was ist Spring?

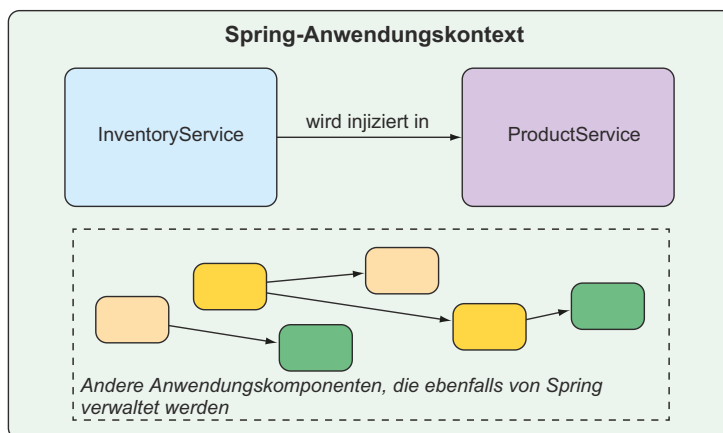
Mir ist klar, dass Sie wahrscheinlich darauf brennen, eine Spring-Anwendung zu schreiben, und ich versichere Ihnen, dass Sie noch in diesem Kapitel eine einfache Anwendung entwickeln werden. Doch zuerst möchte ich mit einigen grundlegenden Spring-Konzepten den Weg bereiten, damit Sie besser verstehen, was Spring am Laufen hält.

Jede nichttriviale Anwendung besteht aus vielen Komponenten, die jeweils für ihren eigenen Teil der gesamten Anwendungsfunktionalität verantwortlich sind. Dabei stimmen sie sich mit den anderen Anwendungselementen ab, um ihre Aufgaben zu erledigen. Wenn die Anwendung läuft, müssen diese Komponenten irgendwie erzeugt und einander bekannt gemacht werden.

In seinem Kern (Core) bietet Spring einen *Container*, oftmals auch als *Spring-Anwendungskontext* bezeichnet, der Anwendungskomponenten erzeugt und verwaltet. Diese Komponenten oder *Beans* werden im Spring-Anwendungskontext miteinander zu einer vollständigen Anwendung verknüpft, ähnlich wie aus Bausteinen, Mörtel, Holz, Nägeln, Rohrleitungen und Kabeln ein Haus entsteht.

Das Verknüpfen der Beans untereinander beruht auf einem Muster, das als *Dependency Injection* (DI) bekannt ist. Anstatt Komponenten den Lebenszyklus anderer Beans, von denen sie abhängig sind, erzeugen und verwalten zu lassen, stützt sich eine DI-Anwendung auf eine separate Entität (den Container), um alle Komponenten zu erzeugen und zu verwalten und diese in die Beans, die auf sie angewiesen sind, zu injizieren. In der Regel geschieht dies über Konstruktorargumente oder die Zugriffsmethoden von Eigenschaften.

Nehmen wir zum Beispiel an, dass es unter den vielen Komponenten einer Anwendung zwei gibt, um die es Ihnen geht: einen Inventurdienst (mit dem sich Lagerbestände abrufen lassen) und einen Produktdienst (der die grundlegenden Produktdaten bereitstellt). Der Produktdienst hängt vom Inventurdienst ab, um vollständige Informationen über Produkte liefern zu können. Bild 1.1 veranschaulicht die Beziehungen zwischen diesen Beans und dem Spring-Anwendungskontext.



**Bild 1.1** Anwendungskomponenten werden durch den Spring-Anwendungskontext verwaltet und ineinander injiziert.

Aufbauend auf dem Core-Container von Spring und einem kompletten Portfolio verwandter Bibliotheken bekommen Sie ein Web-Framework, eine Vielzahl von Optionen für dauerhafte Datenspeicherung, ein Sicherheits-Framework, Integration mit anderen Systemen, Laufzeitüberwachung, Microservice-Unterstützung, ein reaktives Programmiermodell und viele andere Features, die für die moderne Anwendungsentwicklung notwendig sind.

Aus historischer Sicht würde man den Anwendungskontext von Spring dazu bringen, Beans miteinander zu verbinden, indem man mit einer oder mehreren XML-Dateien die Komponenten und ihre Beziehungen zu anderen Komponenten beschreibt. So deklariert das folgende XML zwei Beans, eine `InventoryService`-Bean und eine `ProductService`-Bean, und verknüpft die `InventoryService`-Bean über ein Konstruktorargument mit der `ProductService`-Bean:

```
<bean id="inventoryService"
      class="com.example.InventoryService" />

<bean id="productService"
      class="com.example.ProductService" />
  <constructor-arg ref="inventoryService" />
</bean>
```

In neueren Versionen von Spring ist aber eine Java-basierte Konfiguration gebräuchlicher. Die folgende Java-basierte Konfigurationsklasse ist der XML-Konfiguration äquivalent:

```
@Configuration
public class ServiceConfiguration {
    @Bean
    public InventoryService inventoryService() {
        return new InventoryService();
    }

    @Bean
    public ProductService productService() {
        return new ProductService(inventoryService());
    }
}
```

Die Annotation `@Configuration` teilt Spring mit, dass es sich um eine Konfigurationsklasse handelt, die dem Spring-Anwendungskontext Beans bereitstellt. Die Methoden der Konfigurationsklasse sind mit `@Bean` annotiert. Dies zeigt an, dass die von ihnen zurückgegebenen Objekte als Beans in den Anwendungskontext eingefügt werden sollen (wobei ihre jeweiligen Bean-IDs standardmäßig die gleichen sind wie die Namen der Methoden, die sie definieren).

Die Java-basierte Konfiguration bietet gegenüber der XML-basierten Konfiguration mehrere Vorteile, darunter größere Typsicherheit und verbesserte Möglichkeiten zur Refaktorisierung. Selbst dann ist eine explizite Konfiguration mit Java oder XML nur dann notwendig, wenn Spring nicht in der Lage ist, die Komponenten automatisch zu konfigurieren.

Die automatische Konfiguration geht auf die Spring-Techniken *Autowiring* und *Komponentensuche* (Component Scanning) zurück. Mit einer Komponentensuche kann Spring automatisch Komponenten im Klassenpfad einer Anwendung erkennen und sie als Beans

im Spring-Anwendungskontext erzeugen. Beim Autowiring injiziert Spring automatisch die Komponenten zusammen mit den anderen Beans, von denen sie abhängen.

In jüngster Zeit geht die automatische Konfiguration mit Einführung von Spring Boot weit über Komponentensuche und Autowiring hinaus. Spring Boot ist eine Erweiterung des Spring Frameworks, die mehrere Produktivitätsverbesserungen bietet. Die bekannteste dieser Verbesserungen ist die *Autokonfiguration*. Spring Boot kann dabei anhand von Einträgen im Klassenpfad, in den Umgebungsvariablen und anderen Faktoren vernünftige Schätzungen abgeben, welche Komponenten konfiguriert und miteinander verbunden werden müssen.

Die Autokonfiguration würde ich Ihnen gern anhand von Beispielcode demonstrieren, doch leider kann ich es nicht. Denn die Autokonfiguration ist wie der Wind. Man kann zwar ihre Wirkungen sehen, doch es gibt keinen Code, den ich Ihnen zeigen und sagen könnte: »Sehen Sie! Hier ist ein Beispiel für die Autokonfiguration!« Es passiert etwas, Komponenten werden aktiviert und Funktionalität wird bereitgestellt, ohne dass Sie Code schreiben müssen. Gerade dieser fehlende Code ist wichtig für die Autokonfiguration und das macht ihre Vorzüge aus.

Die Spring-Boot-Autokonfiguration hat die Anzahl der expliziten Konfigurationen (ob mit XML oder Java), die für das Erstellen einer Anwendung erforderlich sind, drastisch reduziert. Und wenn Sie mit dem Beispiel in diesem Kapitel fertig sind, haben Sie tatsächlich eine funktionsfähige Spring-Anwendung vor sich, die nur eine einzige Zeile Spring-Konfigurationscode enthält!

Mit Spring Boot geht die Spring-Entwicklung so viel leichter von der Hand, dass es schwer vorstellbar ist, Spring-Anwendungen ohne Spring Boot zu entwickeln. Aus diesem Grund behandelt dieses Buch Spring und Spring Boot so, als wären sie ein und dasselbe. Wir werden so weit wie möglich auf Spring Boot setzen und eine explizite Konfiguration nur verwenden, wenn es wirklich notwendig ist. Und da die Spring-XML-Konfiguration die Old-School-Methode für das Arbeiten mit Spring ist, konzentrieren wir uns hauptsächlich auf die Java-basierte Konfiguration von Spring.

Doch genug der Vorrede. Im Titel dieses Buches ist die Phrase *im Einsatz* enthalten. Fangen Sie also an und schreiben Sie Ihre erste Anwendung mit Spring.

## ■ 1.2 Eine Spring-Anwendung initialisieren

Anhand dieses Buches erstellen Sie die Online-Anwendung Taco Cloud. Kunden können damit die wundervollsten Speisen bestellen, die je von Menschen kreiert wurden – Tacos. Selbstverständlich verwenden Sie Spring, Spring Boot und eine Vielzahl verwandter Bibliotheken und Frameworks, um dieses Ziel zu erreichen.

Für die Initialisierung einer Spring-Anwendung stehen Ihnen mehrere Optionen zur Verfügung. Zwar könnte ich Ihnen die Schritte erläutern, wie Sie eine Projektverzeichnisstruktur manuell einrichten und eine Build-Spezifikation definieren, doch wäre das Zeitverschwendung – die Zeit lässt sich besser für den Anwendungscode nutzen. Deshalb werden Sie sich auf den Spring Initializr stützen, um die Anwendung hochzufahren.

Der Spring Initializr ist sowohl eine browserbasierte Webanwendung als auch eine REST-API. Damit erzeugen Sie eine Spring-Projektgerüststruktur, die Sie mit jeder gewünschten Funktionalität ausfüllen können. Es gibt verschiedene Wege, um auf Spring Initializr zuzugreifen:

- von der Webanwendung unter <http://start.spring.io>,
- von der Befehlszeile mit dem Befehl `curl`,
- von der Befehlszeile mit der Befehlszeilenoberfläche von Spring Boot,
- beim Erstellen eines neuen Projekts mit der Spring Tool Suite,
- beim Erstellen eines neuen Projekts mit IntelliJ IDEA,
- beim Erstellen eines neuen Projekts mit NetBeans.

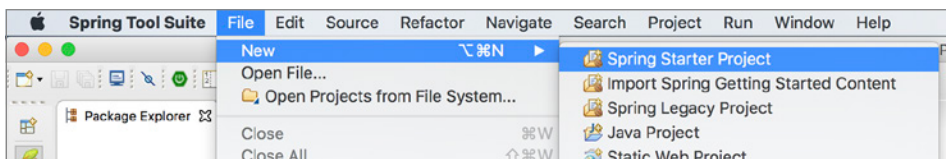
Anstatt mehrere Seiten dieses Kapitels zu opfern, um jede dieser Optionen zu besprechen, habe ich diese Details im Anhang zusammengestellt. In diesem Kapitel und im gesamten Buch zeige ich Ihnen, wie Sie ein neues Projekt mit meiner Lieblingsoption erzeugen: mit der Unterstützung von Spring Initializr in der Spring Tool Suite.

Wie der Name schon andeutet, ist die Spring Tool Suite eine hervorragende Spring-Entwicklungsumgebung. Sie bietet aber auch ein praktisches Spring-Boot-Dashboard-Feature, das in keiner der anderen IDE-Optionen verfügbar ist (zumindest bei Redaktionsschluss dieses Buches).

Wenn Sie nicht mit der Spring Tool Suite arbeiten, ist das auch in Ordnung, wir können trotzdem Freunde bleiben. Springen Sie zum Anhang und ersetzen Sie jeweils die Initializr-Option durch die für Sie am besten geeigneten Anweisungen in den folgenden Abschnitten. Beachten Sie aber, dass ich mich in diesem Buch gelegentlich auf spezielle Features der Spring Tool Suite beziehe, beispielsweise das Spring Boot Dashboard. Wenn Sie die Spring Tool Suite nicht verwenden, müssen Sie diese Anweisungen entsprechend Ihrer IDE anpassen.

### 1.2.1 Ein Spring-Projekt mit der Spring Tool Suite initialisieren

Um mit einem neuen Spring-Projekt zu beginnen, wählen Sie im Menü FILE den Befehl NEW und dann SPRING STARTER PROJECT. Bild 1.2 zeigt die Menüstruktur.



**Bild 1.2** Ein neues Projekt mit dem Initializr in der Spring Tool Suite starten

Nachdem Sie SPRING STARTER PROJECT ausgewählt haben, erscheint ein Assistentendialogfeld für ein neues Projekt (Bild 1.3). Die erste Seite des Assistenten fragt einige allgemeine Projektinformationen ab, unter anderem den Projektnamen, eine Beschreibung und andere wichtige Informationen. Wenn Sie mit dem Inhalt einer *pom.xml*-Datei von Maven vertraut sind, werden Sie die meisten Felder als Elemente erkennen, die in einer Maven-Build-Spezifikation landen. Füllen Sie das Dialogfeld für die Taco-Cloud-Anwendung wie in Bild 1.3 gezeigt aus und klicken Sie dann auf NEXT.

**New Spring Starter Project**

Name:

Use default location

Location:

Type:  Packaging:

Java Version:  Language:

Group:

Artifact:

Version:

Description:

Package:

Working sets

Add project to working sets

Working sets:

**Bild 1.3** Allgemeine Projektinformationen für die Taco-Cloud-Anwendung festlegen

Auf der nächsten Seite des Assistenten wählen Sie die Abhängigkeiten aus, die Sie Ihrem Projekt hinzufügen möchten (siehe Bild 1.4). Im oberen Teil des Dialogfelds können Sie festlegen, auf welcher Version von Spring Ihr Projekt aufbauen soll. Standardeinstellung ist die aktuelle Version, die jeweils verfügbar ist. Im Allgemeinen empfiehlt es sich, die Vorgabe beizubehalten, außer wenn Sie eine andere Zielversion verwenden müssen.

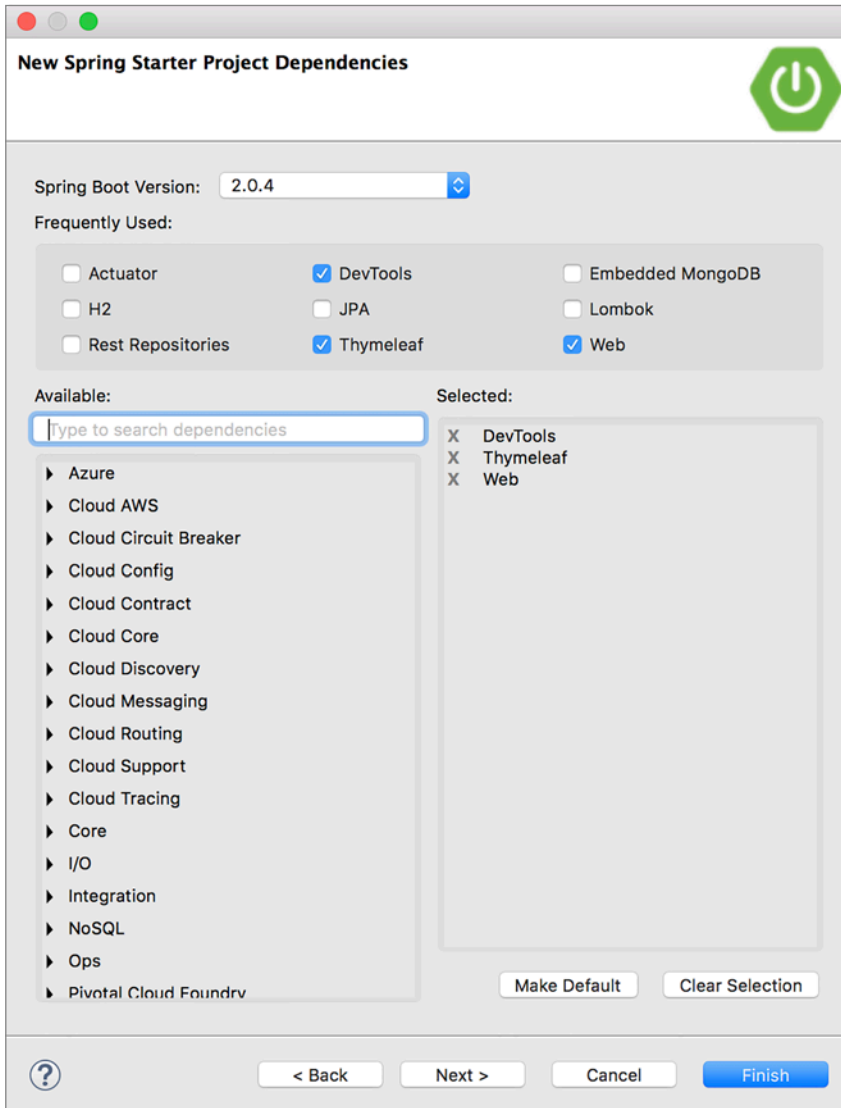
Hinsichtlich der Abhängigkeiten selbst können Sie entweder die verschiedenen Abschnitte erweitern und die gewünschten Abhängigkeiten manuell herausuchen oder Sie gehen über das Suchfeld im oberen Teil der Liste AVAILABLE. Für die Taco-Cloud-Anwendung beginnen Sie mit den in Bild 1.4 gezeigten Abhängigkeiten.

An dieser Stelle können Sie auf FINISH klicken, um das Projekt zu generieren und es Ihrem Arbeitsbereich hinzuzufügen. Doch wenn Sie unternehmungslustig sind, klicken Sie noch einmal auf NEXT, um die letzte Seite des Assistenten, die in Bild 1.5 dargestellt ist, für ein neues Projekt zu öffnen.

Standardmäßig ruft der Assistent für ein neues Projekt den Spring Initializr unter <http://start.spring.io> auf, um das Projekt zu generieren. Da Sie diese Standardeinstellung normalerweise nicht überschreiben müssen, können Sie auf der zweiten Seite des Assistenten auf FINISH klicken. Falls Sie aber aus irgendeinem Grund Ihren eigenen Klon von Initializr

hosten (vielleicht als lokale Kopie auf Ihrem Computer oder einen zugeschnittenen Klon, der innerhalb der Firewall Ihrer Firma läuft), können Sie im Feld BASE URL die Adresse Ihrer Instanz von Initializr eintragen, bevor Sie auf FINISH klicken.

Nachdem Sie auf FINISH geklickt haben, wird das Projekt vom Initializr heruntergeladen und in Ihren Arbeitsbereich gebracht. Warten Sie etwas, bis das Laden und Erstellen abgeschlossen ist. Dann können Sie beginnen, die Funktionalität der Anwendung zu entwickeln. Werfen Sie vorher aber einen Blick auf das, was der Initializr Ihnen geliefert hat.



**Bild 1.4** Starter-Abhängigkeiten auswählen