

<b>Vorwort</b> .....	<b>11</b>
<b>1 Spring Boot in a Nutshell</b> .....	<b>15</b>
Die drei Grundeigenschaften von Spring Boot .....	15
Starter für vereinfachtes Abhängigkeitsmanagement .....	15
Ausführbare JARs für ein vereinfachtes Deployment .....	17
Autokonfiguration .....	18
Zusammenfassung .....	20
<b>2 Ihre Werkzeuge auswählen und beginnen</b> .....	<b>21</b>
Maven oder Gradle? .....	21
Apache Maven .....	21
Gradle .....	23
Die Wahl zwischen Maven und Gradle .....	24
Java oder Kotlin? .....	25
Java .....	25
Kotlin .....	25
Zwischen Java und Kotlin wählen .....	27
Eine Version von Spring Boot wählen .....	27
Der Spring Initializr .....	27
Rauf auf die Kommandozeile .....	32
In Integrated Development Environments (IDEs) bleiben .....	34
Unterwegs in main() .....	36
Zusammenfassung .....	37
<b>3 Ihr erstes Spring-Boot-REST-API erstellen</b> .....	<b>39</b>
Das Wie und Warum von APIs .....	39
Was ist REST, und warum ist das wichtig? .....	40

Ihr API im HTTP-Verbstil . . . . .	41
Zurück zum Initializr . . . . .	41
Erzeugen einer einfachen Domain . . . . .	43
GET-ting . . . . .	45
@RestController in a Nutshell . . . . .	45
POST-ing . . . . .	49
PUT-ting . . . . .	49
DELETE-ing . . . . .	50
Und mehr . . . . .	50
Vertrauen ist gut, Kontrolle ist besser . . . . .	52
Zusammenfassung . . . . .	55
<b>4 Datenbankzugriff für Ihre Spring-Boot-Anwendung . . . . .</b>	<b>57</b>
Die Autokonfiguration für den Datenbankzugriff vorbereiten . . . . .	57
Was wollen wir erreichen? . . . . .	58
Hinzufügen einer Datenbankabhängigkeit . . . . .	58
Code hinzufügen . . . . .	59
Speichern und Abrufen von Daten . . . . .	66
Ein bisschen Nachpolieren . . . . .	69
Zusammenfassung . . . . .	71
<b>5 Konfigurieren und Untersuchen Ihrer Spring-Boot-Anwendung . . . . .</b>	<b>73</b>
Anwendungskonfiguration . . . . .	74
@Value . . . . .	75
@ConfigurationProperties . . . . .	79
Potenzielle Drittanbieter-Option . . . . .	84
Autoconfiguration Report . . . . .	87
Der Actuator . . . . .	88
Den Actuator dazu bringen, mehr zu verraten . . . . .	94
Umweltbewusster werden mit dem Actuator . . . . .	95
Mehr protokollieren mit dem Actuator . . . . .	97
Zusammenfassung . . . . .	98
<b>6 Tief in das Thema Daten einsteigen . . . . .</b>	<b>101</b>
Entitäten definieren . . . . .	102
Template-Unterstützung . . . . .	103
Repository-Unterstützung . . . . .	103
@Before . . . . .	104
Einen Template-basierten Dienst mit Redis erstellen . . . . .	104
Das Projekt initialisieren . . . . .	104
Den Redis-Dienst entwickeln . . . . .	105

Von Template auf Repository umsteigen . . . . .	114
Mit dem Java Persistence API (JPA) einen Repository-basierten Dienst erstellen . . . . .	117
Das Projekt initialisieren . . . . .	118
Den JPA-(MySQL-)Dienst entwickeln . . . . .	118
Daten laden . . . . .	123
Mithilfe einer dokumentenorientierten NoSQL-Datenbank einen Repository-basierten Dienst erstellen . . . . .	127
Das Projekt initialisieren . . . . .	127
Den MongoDB-Dienst entwickeln . . . . .	129
Einen Repository-basierten Dienst mithilfe einer NoSQL-Graphdatenbank erstellen . . . . .	135
Das Projekt initialisieren . . . . .	135
Den Neo4j-Dienst entwickeln . . . . .	136
Zusammenfassung . . . . .	145
<b>7 Anwendungen mittels Spring MVC erstellen . . . . .</b>	<b>147</b>
Spring MVC: Was bedeutet das? . . . . .	147
Endanwender-Interaktionen mittels Template-Engines . . . . .	148
Das Projekt initialisieren . . . . .	149
Die Aircraft-Positions-Anwendung entwickeln . . . . .	149
Nachrichten übergeben . . . . .	156
Den PlaneFinder ausbauen . . . . .	157
Die Anwendung Aircraft Positions erweitern . . . . .	161
Konversationen mit WebSocket erzeugen . . . . .	165
Was ist WebSocket? . . . . .	165
Die Aircraft-Positions-Anwendung umbauen . . . . .	166
Zusammenfassung . . . . .	172
<b>8 Reaktive Programmierung mit Project Reactor und Spring WebFlux . . . . .</b>	<b>175</b>
Einführung in die reaktive Programmierung . . . . .	175
Project Reactor . . . . .	179
Tomcat versus Netty . . . . .	181
Reaktiver Datenzugriff . . . . .	181
R2DBC mit H2 . . . . .	182
Reaktives Thymeleaf . . . . .	193
RSocket für eine vollständig reaktive Interprozess-Kommunikation . . . . .	194
Was ist RSocket? . . . . .	194
RSocket einsetzen . . . . .	195
Zusammenfassung . . . . .	200

<b>9</b>	<b>Spring-Boot-Anwendungen zur erhöhten Produktionsbereitschaft testen . . .</b>	<b>201</b>
	Unit-Tests . . . . .	201
	@SpringBootTest vorgestellt . . . . .	202
	Wichtige Unit-Tests für die Aircraft-Positions-Anwendung . . . . .	203
	Refaktorisieren für ein besseres Testen. . . . .	208
	Slices testen . . . . .	215
	Zusammenfassung . . . . .	220
<b>10</b>	<b>Ihre Spring-Boot-Anwendung sichern. . . . .</b>	<b>221</b>
	Authentifizierung und Autorisierung . . . . .	221
	Authentifizierung . . . . .	222
	Autorisierung . . . . .	223
	Spring Security in aller Kürze . . . . .	223
	Die HTTP Firewall . . . . .	224
	Sicherheitsfilterketten . . . . .	224
	Request- und Response-Header . . . . .	225
	Implementieren einer formularbasierten Authentifizierung und Autorisierung mit Spring Security. . . . .	225
	Spring-Security-Abhängigkeiten hinzufügen . . . . .	225
	Authentifizierung hinzufügen . . . . .	232
	Autorisierung . . . . .	237
	Implementieren von OpenID Connect und OAuth2 zur Authentifizierung und Autorisierung . . . . .	246
	Die Client-Anwendung Aircraft Positions . . . . .	248
	PlaneFinder-Ressourcenserver . . . . .	254
	Zusammenfassung . . . . .	262
<b>11</b>	<b>Das Deployment Ihrer Spring-Boot-Anwendung . . . . .</b>	<b>263</b>
	Wieder zurück zum ausführbaren Spring-Boot-JAR. . . . .	264
	Ein »vollständig ausführbares« Spring-Boot-JAR bauen . . . . .	265
	Was bedeutet das? . . . . .	270
	JARs auspacken . . . . .	270
	Deployment von Spring-Boot-Anwendungen in Containern . . . . .	276
	Ein Container-Image aus der IDE heraus erzeugen . . . . .	278
	Ein Container-Image von der Kommandozeile aus erzeugen . . . . .	280
	Verifizieren, dass das Image existiert . . . . .	281
	Die Container-Anwendung ausführen. . . . .	282
	Hilfsmittel zum Untersuchen der Container-Images einer Spring-Boot-Anwendung . . . . .	283
	pack . . . . .	283
	dive. . . . .	285
	Zusammenfassung . . . . .	285

<b>12 In die reaktive Programmierung intensiv eintauchen</b> .....	<b>287</b>
Wann sollte man reaktiv arbeiten? .....	288
Reaktive Anwendungen testen .....	288
Zuerst aber: refaktorisieren .....	289
Und nun das Testen .....	296
Diagnostizieren und Debuggen reaktiver Anwendungen .....	304
Hooks.onOperatorDebug() .....	305
Checkpoints .....	314
ReactorDebugAgent.init() .....	317
Zusammenfassung .....	319
<b>Index</b> .....	<b>321</b>