



→ 2., aktualisierte und erweiterte Auflage



Ihns · Heldt · Koschek · Ehm · Sahling · Schlömmer

EJB 3.1 professionell

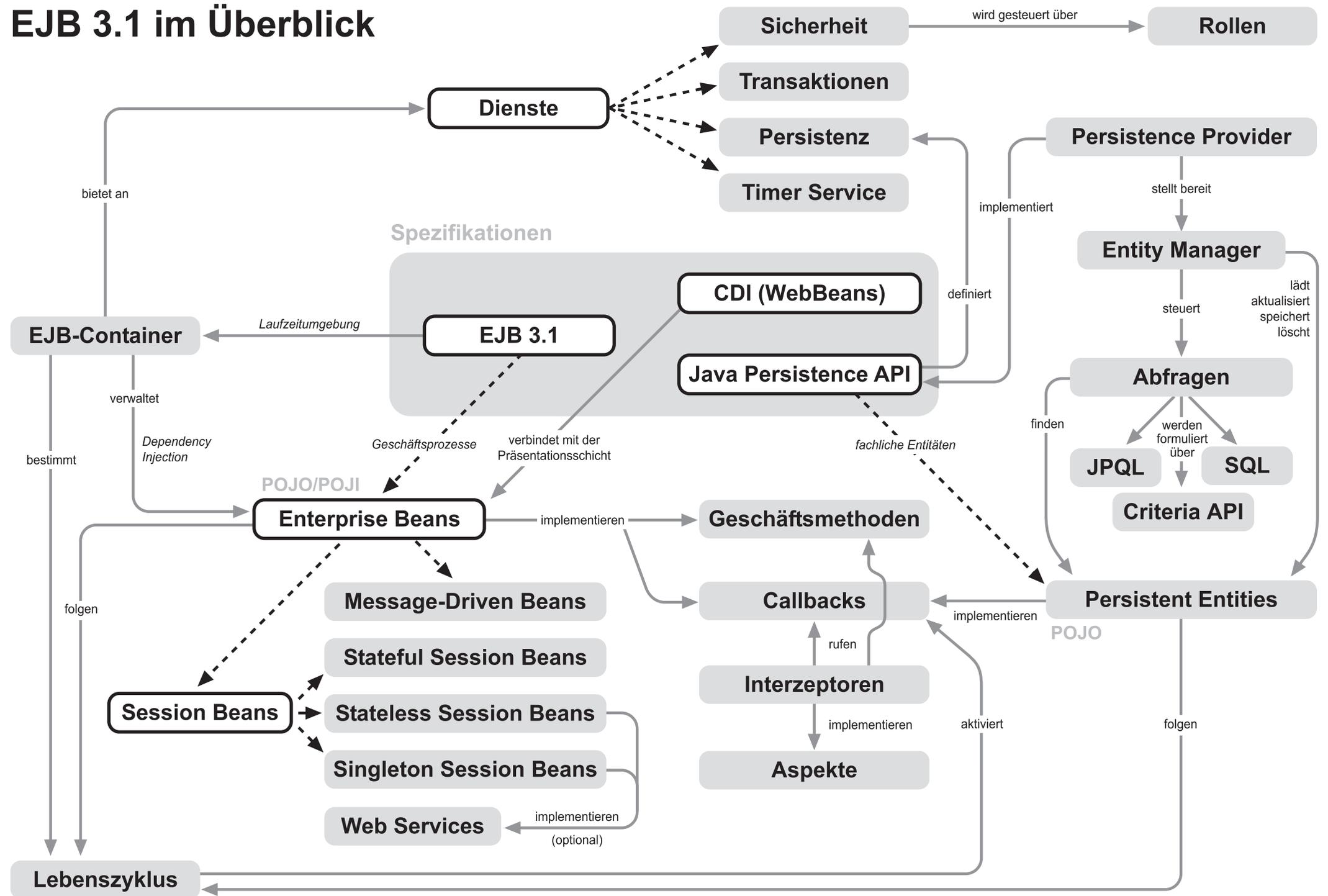
Grundlagen- und Expertenwissen
zu Enterprise JavaBeans 3.1

inkl.
JPA 2.0

 X EDITION

dpunkt.verlag

EJB 3.1 im Überblick



EJB 3.1 professionell

iX-Edition

In der iX-Edition erscheinen Titel, die vom dpunkt.verlag gemeinsam mit der Redaktion der Computerzeitschrift iX ausgewählt und konzipiert werden. Inhaltlicher Schwerpunkt dieser Reihe sind Standardwerke zu professioneller Datenverarbeitung und Internet.

**Oliver Ihns · Stefan M. Heldt · Holger Koschek · Joachim Ehm ·
Carsten Sahling · Roman Schlömmer**

EJB 3.1 professionell

**Grundlagen- und Expertenwissen zu
Enterprise JavaBeans 3.1 – inkl. JPA 2.0**

Unter Mitarbeit von Carl Anders Düvel, Norman Erck und
Daniel Steinhöfer

2., aktualisierte und erweiterte Auflage



dpunkt.verlag

Lektorat: René Schönfeldt
Copy-Editing: Ursula Zimpfer, Herrenberg
Herstellung: Birgit Bäuerlein
Umschlaggestaltung: Helmut Kraus, www.exclam.de
Druck und Bindung: M.P. Media-Print Informationstechnologie GmbH, 33100 Paderborn

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.

ISBN:
Buch 978-3-89864-612-3
PDF 978-3-89864-851-6
ePub 978-3-86491-025-8

2., aktualisierte und erweiterte Auflage 2011
Copyright © 2011 dpunkt.verlag GmbH
Ringstraße 19 B
69115 Heidelberg

Die vorliegende Publikation ist urheberrechtlich geschützt. Alle Rechte vorbehalten. Die Verwendung der Texte und Abbildungen, auch auszugsweise, ist ohne die schriftliche Zustimmung des Verlags urheberrechtswidrig und daher strafbar. Dies gilt insbesondere für die Vervielfältigung, Übersetzung oder die Verwendung in elektronischen Systemen.

Es wird darauf hingewiesen, dass die im Buch verwendeten Soft- und Hardware-Bezeichnungen sowie Markennamen und Produktbezeichnungen der jeweiligen Firmen im Allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Alle Angaben und Programme in diesem Buch wurden mit größter Sorgfalt kontrolliert. Weder Autor noch Verlag können jedoch für Schäden haftbar gemacht werden, die in Zusammenhang mit der Verwendung dieses Buches stehen.

Inhaltsübersicht

	Vorwort	1
1	Einleitung – Jetzt wird’s leicht!	9
Teil I	Grundlagen	31
2	Leichtgewichtige, POJO-basierte Enterprise-Applikationen	33
3	EJB-Komponentenarchitektur	55
4	EJB Lite – das abgespeckte EJB	87
Teil II	Technologie	93
5	Session Beans	95
6	Session Beans als Webservice	153
7	Message-Driven Beans	179
8	Entity Beans	213
9	Persistenzabbildung	215
10	JPA-Abfragen	327
11	Der Lebensraum der Enterprise Beans	361
12	Callback-Mechanismen	407
13	Interzeptoren	431
14	Timer Service	461
15	Sicherer Zugriff auf EJB-Komponenten	481
Teil III	Ergänzende Themen	493
16	CDI (Web Beans)	495
17	Testen von EJB-Komponenten	513
18	Migration von EJB 2.x nach 3.x	537
Anhang		577
	Literatur – offline und online	579
	Stichwortverzeichnis	583

Inhaltsverzeichnis

Vorwort	1
Was ist neu in der zweiten Auflage?	1
Für wen ist dieses Buch?	3
Von wem ist dieses Buch?	3
Danksagungen	7
Widmung	7
Ihre Kommentare und Anmerkungen	8
1 Einleitung – Jetzt wird's leicht!	9
1.1 Die Evolution der Enterprise JavaBeans	9
1.1.1 Der Fluch der Komplexität	10
1.1.2 Kritik an den früheren EJB-Versionen	10
1.1.3 Ein neues Denken schafft eine neue Architektur	11
1.1.4 Der konsequente nächste Schritt	12
1.2 Aufbau des Buches	13
1.2.1 Die Themenbereiche	13
1.2.2 Die Kapitel: Struktur und Inhalte	14
1.3 Konventionen	20
1.3.1 Notation	20
1.3.2 Literaturverweise bzw. Referenzen	20
1.3.3 Quellcode	21
1.3.4 Piktogramme	21
1.3.5 Verwendung von Anglizismen	22
1.3.6 Gleichberechtigung in der Sprache	22
1.4 Die Beispielapplikation »Ticket2Rock«	23
1.4.1 Kurzbeschreibung	23
1.4.2 Anwendungsfälle	24
1.4.3 Fachliche Entitäten	26
1.5 Verwendete Technologien und Produkte	28
1.6 Die Website zum Buch	29

Teil I	Grundlagen	31
2	Leichtgewichtige, POJO-basierte Enterprise-Applikationen	33
2.1	Kurz gefasst	33
2.2	Der Blick zurück	33
2.3	Einleitung	34
2.4	Hauptziele für EJB 3.x	37
2.4.1	Motive	37
2.4.2	»Einfach machen!«	38
2.4.3	Vereinfachung der Mikroarchitektur von EJB-Komponenten	38
2.4.4	Vereinfachung des Entwicklungsprozesses	39
2.4.5	Vereinfachung der Nutzung von EJB-Komponenten	41
2.4.6	Neuentwicklung einer leistungsfähigen Persistenzlösung	42
2.5	POJOs und POJIs	42
2.5.1	POJO	42
2.5.2	POJI	43
2.5.3	Unterschiede zwischen EJB 3.x und EJB 2.x	43
2.6	Inversion of Control und Dependency Injection	44
2.6.1	Ziele im Kontext von EJB 3.x	46
2.6.2	Auswirkungen auf die EJB-3.x-Architektur	46
2.6.3	»Hollywood, wir kommen!« – ein Beispiel	47
2.7	Annotationen und Deployment-Deskriptoren	48
2.7.1	Einführung	48
2.7.2	Deployment-Deskriptoren – so schlecht wie ihr Ruf?	49
2.7.3	Annotationen	49
2.7.4	Ein Plädoyer für Deployment-Deskriptoren	52
2.7.5	Der Deployment-Deskriptor hat das letzte Wort	53
2.8	Configuration by Exception	54
3	EJB-Komponentenarchitektur	55
3.1	Kurz gefasst	55
3.2	Grundlegende Konzepte	56
3.2.1	Komponentenarchitektur	56
3.2.2	Java EE	57
3.2.3	Java SE	59
3.2.4	Der Java-EE-Applikationsserver	59
3.2.5	Der EJB-Container	61

3.3	Übersicht der EJB-Typen	67
3.3.1	Session Bean	68
3.3.2	Message-Driven Bean	70
3.3.3	Persistent Entity	71
3.4	Aufrufmodelle	72
3.4.1	Aufrufmodell: »entfernt«	73
3.4.2	Aufrufmodell »lokal«	76
3.4.3	Aufrufmodell »nachrichtenbasiert«	81
3.4.4	Aufrufmodelle und EJB-Typen im Überblick	86
4	EJB Lite – das abgeseckte EJB	87
4.1	Kurz gefasst	87
4.2	Der Blick zurück	87
4.3	Leicht zu sein bedarf es wenig	88
4.4	Was mach ich nun mit dem EJB-Lite-Profil?	91
Teil II	Technologie	93
5	Session Beans	95
5.1	Kurz gefasst	95
5.2	Der Blick zurück	95
5.3	Einführung	96
5.4	Stateless Session Beans	97
5.4.1	Transaktionen	98
5.4.2	Instanzen-Pooling	99
5.4.3	Webservices	99
5.5	Stateful Session Beans	99
5.5.1	Aktivierung und Passivierung	101
5.5.2	Transaktionen	102
5.6	Singleton Session Beans	103
5.6.1	Transaktionen	104
5.6.2	Nebenläufigkeit	104
5.7	Stateless, Stateful und Singleton Session Beans im Vergleich	105
5.8	Mikroarchitektur einer Session Bean	107
5.8.1	Namenskonventionen	108
5.8.2	Zusammenspiel der Elemente	108
5.8.3	Erzeugen von Session Beans	110
5.8.4	Löschen von Session Beans	111

- 5.9 Lebenszyklus von Stateless Session Beans 112
 - 5.9.1 Zustand »does not exist« 112
 - 5.9.2 Übergang von »does not exist« zu »method-ready pool« 112
 - 5.9.3 Zustand »method-ready pool« 113
 - 5.9.4 Übergang von »method-ready pool« zu »does not exist« 113
- 5.10 Lebenszyklus von Stateful Session Beans 114
 - 5.10.1 Zustand »does not exist« 115
 - 5.10.2 Übergang von »does not exist« zu »method-ready« 115
 - 5.10.3 Zustände »method-ready« und »method-ready in TX« 115
 - 5.10.4 Zustand »passive« 117
 - 5.10.5 Übergang in den Zustand »does not exist« 117
- 5.11 Lebenszyklus von Singleton Session Beans 118
 - 5.11.1 Zustand »does not exist« 118
 - 5.11.2 Übergang von »does not exist« zu »method-ready« 119
 - 5.11.3 Zustand »method-ready« 120
 - 5.11.4 Übergang von »method-ready« zu »does not exist« 120
- 5.12 Business Interface 120
- 5.13 No-Interface Client View 123
- 5.14 Bean-Klasse 125
 - 5.14.1 Deklarative Transaktionalität 130
 - 5.14.2 Transaktionen in Handarbeit 136
 - 5.14.3 Transaktionen im Ausnahmezustand 138
 - 5.14.4 EJB-Kontext 140
 - 5.14.5 Checkliste 142
- 5.15 Nebenläufigkeit bei Singleton Session Beans 143
 - 5.15.1 Nebenläufigkeit – Container-Managed 144
 - 5.15.2 Nebenläufigkeit – Bean-Managed 146
- 5.16 Asynchrone Methodenaufrufe 147
- 5.17 Timer Service 151
- 6 Session Beans als Webservice 153**
 - 6.1 Kurz gefasst 153
 - 6.2 Der Blick zurück 153
 - 6.3 Was ist ein Webservice? 154

6.4	Stateless und Singleton Session Beans als Webservice	156
6.4.1	@WebService	159
6.4.2	@SOAPBinding	163
6.4.3	@WebMethod	169
6.4.4	@Oneway	170
6.4.5	@WebParam	171
6.4.6	@WebResult	174
6.4.7	@HandlerChain	175
6.5	Vorgehensweise bei der Erstellung von Webservices	176
6.6	Die Abbildung von Rückgabetypen und Parametern beeinflussen	176
6.7	Ein Webservice-Client	177
7	Message-Driven Beans	179
7.1	Kurz gefasst	179
7.2	Der Blick zurück	180
7.3	Nachrichtenbasierte Kommunikation	180
7.3.1	Charakteristika und Vorteile	183
7.3.2	Kommunikationsmodelle	183
7.4	Java Message Service (JMS)	185
7.4.1	Service Provider Interface	186
7.4.2	JMS API	186
7.5	Charakteristika von Message-Driven Beans	189
7.5.1	JMS Message-Driven Beans	192
7.5.2	Connector-based Message-Driven Beans	192
7.6	Lebenszyklus von Message-Driven Beans	193
7.7	Transaktionalität	194
7.8	Bean-Klasse	195
7.8.1	@MessageDriven	196
7.8.2	@ActivationConfigProperty	197
7.8.3	Message Listener Interface	200
7.8.4	Beantworten von Nachrichten	201
7.8.5	Checkliste	203
7.9	Deployment-Deskriptor	203
7.10	Timer Service	205
7.11	Ein JMS-Client	205
7.12	Message Linking	207

8	Entity Beans	213
8.1	Kurz gefasst	213
8.2	Der Blick zurück	213
8.3	Aus Entity Beans werden Persistent Entities	214
9	Persistenzabbildung	215
9.1	Kurz gefasst	215
9.2	Der Blick zurück	216
9.3	Persistenz? Abbildung?	216
9.4	Persistent Entities	219
9.4.1	Lightweight	219
9.4.2	Persistent	220
9.4.3	Domain Object	220
9.4.4	Lebenszyklus	221
9.5	Persist my POJO!	221
9.5.1	Annotation oder Deployment-Deskriptor?	221
9.5.2	Beispiel	222
9.6	Grundkonzepte	225
9.6.1	Persistence Provider	225
9.6.2	Entity-Manager	226
9.6.3	Persistenzeinheit	229
9.6.4	Persistenzkontext	230
9.7	Deployment-Deskriptoren	232
9.7.1	persistence.xml	232
9.7.2	orm.xml	234
9.8	Arbeiten mit dem Entity-Manager	235
9.8.1	Dauerhaftes Speichern in der Datenbank (persist)	237
9.8.2	Aktualisieren des persistenten Objektzustands (merge)	237
9.8.3	Löschen einer Persistent Entity (remove)	238
9.8.4	Finden einer Persistent Entity in der Datenbank (find, getReference)	238
9.8.5	Sofortiges Ausführen der Datenbankoperation (flush)	239
9.8.6	Blockieren einer Persistent Entity (lock)	240
9.8.7	Aktualisieren des Zustands der Objektinstanz (refresh)	241
9.8.8	Leben im Persistenzkontext (clear, contains)	241
9.8.9	Losgelöst (detach)	241
9.8.10	Abfragen (create...Query)	242

9.8.11	Transaktionen (joinTransaction)	242
9.8.12	Zugriff auf den Persistence Provider (getDelegate)	242
9.8.13	Beenden des Entity-Managers (close)	243
9.8.14	Suche nach Entitäten (getCriteriaBuilder)	243
9.8.15	Das Metamodell der Persistenz (getMetaModel)	243
9.9	Abbildung von Datentypen	243
9.9.1	Zugriff auf persistente Felder	244
9.9.2	Einfache Datentypen (@Basic)	246
9.9.3	Eingebettete Objekte (@Embeddable)	247
9.9.4	Große Objekte (@Lob)	247
9.9.5	Datum und Zeit (@Temporal)	248
9.9.6	Aufzählungen (@Enumerated)	248
9.10	Abbildung in Datenbanktabellen	250
9.10.1	@Table	250
9.10.2	@Column	251
9.11	Primärschlüssel	252
9.11.1	Einfache Primärschlüssel (@Id)	253
9.11.2	Zusammengesetzte Primärschlüssel (@IdClass, @EmbeddedId)	253
9.11.3	Generierung von Primärschlüsseln	260
9.12	Abbildung von Objektbeziehungen	266
9.12.1	Die glorreichen Sieben	266
9.12.2	Unidirektionale Eins-zu-Eins-Beziehung	268
9.12.3	Bidirektionale Eins-zu-Eins-Beziehung	272
9.12.4	Unidirektionale Eins-zu-Viele-Beziehung	274
9.12.5	Bidirektionale Eins-zu-Viele-Beziehung	278
9.12.6	Unidirektionale Viele-zu-Eins-Beziehung	281
9.12.7	Bidirektionale Viele-zu-Eins-Beziehung	282
9.12.8	Unidirektionale Viele-zu-Viele-Beziehung	282
9.12.9	Bidirektionale Viele-zu-Viele-Beziehung	285
9.12.10	Kaskadieren von Persistenzoperationen	288
9.13	Eingebettete Objekte	291
9.14	Abbildung auf mehrere Datenbanktabellen	295
9.14.1	»Single-Table-Mapping«	296
9.14.2	Multi-Table-Mapping	297
9.15	Vererbung und Polymorphie	301
9.15.1	Erben und Vererben	301
9.15.2	single table per class hierarchy strategy (SINGLE_TABLE)	303

9.15.3	single table per concrete entity class strategy (TABLE_PER_CLASS)	309
9.15.4	joined subclass strategy (JOINED)	312
9.16	Fetching-Strategien	317
9.16.1	Eager Load	318
9.16.2	Lazy Load	318
9.16.3	Deklaration der Fetching-Strategie	319
9.16.4	Lazy Load und Detached Objects	322
9.16.5	Caching von Entitäten	323
10	JPA-Abfragen	327
10.1	Kurz gefasst	327
10.2	Der Blick zurück	327
10.3	Abfragen mit der Query API	328
10.3.1	Queries	329
10.3.2	Named Queries	335
10.3.3	Native Queries	336
10.3.4	Criteria Queries	341
10.4	Java Persistence Query Language (JPQL)	342
10.4.1	SELECT	342
10.4.2	FROM	348
10.4.3	WHERE	350
10.4.4	ORDER BY	354
10.4.5	GROUP BY	355
10.4.6	HAVING	356
10.4.7	Schreibende Massenoperationen	356
10.5	Criteria API	357
10.5.1	Aufbau von Abfragen	357
10.5.2	CriteriaQuery	358
10.5.3	Operatoren, Funktionen und Ausdrücke	359
11	Der Lebensraum der Enterprise Beans	361
11.1	Kurz gefasst	361
11.2	Der Blick zurück	362
11.3	Der Enterprise Naming Context	362
11.4	Globale JNDI-Namen	363
11.4.1	Namensschema	363
11.4.2	Beispiel	364

11.5	Konfigurationsalternativen	365
11.5.1	Annotationsen	365
11.5.2	Deployment-Deskriptoren	366
11.5.3	Kombination von Annotationen und Deployment-Deskriptoren	366
11.6	Arbeiten mit dem ENC	366
11.6.1	Bestückung mittels Deployment-Deskriptoren ..	368
11.6.2	Bestückung mittels Annotationen	368
11.6.3	Zugriff via JNDI-Lookup	369
11.6.4	Zugriff via EJBContext	369
11.6.5	Dependency Injection mit Deployment- Deskriptoren	370
11.6.6	Dependency Injection mit Annotationen	371
11.6.7	Sichtbarkeit von Einträgen im ENC	374
11.7	Auswirkungen auf den Softwaretest	375
11.8	Ressourcentypen	375
11.8.1	Enterprise Beans (@EJB)	376
11.8.2	Extern verwaltete Ressourcen (@Resource)	382
11.8.3	Resource Environment Entries (@Resource) ...	386
11.8.4	Umgebungsvariablen (@Resource)	389
11.8.5	Persistenzkontext (@PersistenceContext)	392
11.8.6	Persistenzeinheiten (@PersistenceUnit)	397
11.8.7	Message Destinations	401
11.8.8	Webservices (@WebServiceRef)	402
12	Callback-Mechanismen	407
12.1	Kurz gefasst	407
12.2	Der Blick zurück	407
12.3	Inversion of Control	408
12.4	Deklaration einer Callback-Methode	409
12.4.1	Callback-Annotationen	409
12.4.2	Deklaration im Deployment-Deskriptor	410
12.4.3	Für jede Bean die passenden Callbacks	412
12.4.4	Regeln für Callback-Methoden	412
12.5	Aufrufreihenfolge für Callback-Methoden	415
12.6	Callbacks für Stateless Session Beans	415
12.6.1	@PostConstruct	416
12.6.2	@PreDestroy	418

12.7	Callbacks für Stateful Session Beans	419
12.7.1	@PostConstruct und @PreDestroy	420
12.7.2	@PrePassivate	420
12.7.3	@PostActivate	422
12.8	Callbacks für Singleton Session Beans	422
12.8.1	@PostConstruct und @PreDestroy	423
12.8.2	@PostActivate und @PrePassivate werden ignoriert	423
12.9	Callbacks für Message-Driven Beans	424
12.9.1	@PostConstruct und @PreDestroy	424
12.9.2	@PostActivate und @PrePassivate werden ignoriert	425
12.10	Callbacks für Persistent Entities	425
12.10.1	Aufrufreihenfolge im Objektverbund	426
12.10.2	@PrePersist und @PostPersist	428
12.10.3	@PreUpdate und @PostUpdate	428
12.10.4	@PreRemove und @PostRemove	429
12.10.5	@PostLoad	429
13	Interzeptoren	431
13.1	Kurz gefasst	431
13.2	Der Blick zurück	432
13.3	Was ist aspektorientierte Programmierung?	432
13.4	Klassifikation	434
13.4.1	Interzeptoren für Geschäftsmethoden	435
13.4.2	Interzeptoren für Timeout-Methoden von Timer-Objekten	435
13.4.3	Interzeptoren für Lebenszykluseignisse	436
13.4.4	Default-Interzeptoren	436
13.4.5	Entity Listener	437
13.4.6	Default Entity Listener	437
13.5	Interzeptoren für Geschäftsmethoden	438
13.5.1	Definition	438
13.5.2	Verwendung	440
13.5.3	InvocationContext	443
13.5.4	Aufrufreihenfolge	444
13.5.5	Ausnahmebehandlung	445
13.5.6	@AroundInvoke-Methode in der Bean-Klasse	445

13.6	Interzeptoren für Timeout-Methoden von Timer-Objekten	446
13.6.1	Definition	446
13.6.2	Verwendung	447
13.6.3	Aufrufreihenfolge	447
13.6.4	Ausnahmebehandlung	449
13.7	Interzeptoren für Lebenszyklusereignisse	449
13.7.1	Definition	449
13.7.2	Verwendung	450
13.7.3	Aufrufreihenfolge	451
13.7.4	Ausnahmebehandlung	452
13.8	Default-Interzeptoren	452
13.9	Entity Listener	453
13.9.1	Definition	453
13.9.2	Verwendung	455
13.9.3	Aufrufreihenfolge	456
13.10	Default Entity Listener	457
13.11	Sind Interzeptoren und Entity Listener aspektorientiert?	458
14	Timer Service	461
14.1	Kurz gefasst	461
14.2	Der Blick zurück	462
14.3	Programmgesteuerte Timer	463
14.3.1	Die Timeout-Methode	465
14.3.2	Das Interface javax.ejb.TimerService	466
14.4	Timer-Objekte	470
14.5	Automatische Timer	471
14.6	Ausdrucksmöglichkeiten für Zeitpläne	474
14.7	Timer und EJB-Typen	475
14.7.1	Stateless Session Bean Timer	476
14.7.2	Singleton Session Bean Timer	478
14.7.3	Message-Driven Bean Timer	478
14.7.4	Timer und Entitäten	478
14.8	Timer und Transaktionen	479
14.9	Timer und Interzeptoren	480
14.10	Checkliste	480

15	Sicherer Zugriff auf EJB-Komponenten	481
15.1	Kurz gefasst	481
15.2	Der Blick zurück	481
15.3	Überblick	482
15.4	Authentifizierung	483
15.5	Sicherheitsrollen	484
15.6	Verwendung von Rollen	484
15.6.1	@RolesAllowed	484
15.6.2	@PermitAll	486
15.6.3	@DenyAll	487
15.7	Ausführen in einem anderen Kontext (@RunAs)	488
15.8	Identität von Message-Driven Beans und Timer Services	489
15.9	Programmgesteuerter Zugriff auf den Security-Kontext	490
15.10	Regeln für Security-Annotationen	491
Teil III Ergänzende Themen		493
16	CDI (Web Beans)	495
16.1	Kurz gefasst	495
16.2	Der Blick zurück	496
16.3	Einführung	496
16.3.1	Grundkonzepte	497
16.3.2	Let's rock – ein praktisches Beispiel	498
16.4	Das Bean-Verständnis von CDI	501
16.4.1	Managed Beans	501
16.4.2	Session Beans	501
16.5	Die Werkzeugkiste	502
16.5.1	Bean-Typen	503
16.5.2	Qualifier	504
16.5.3	Alternativen	505
16.5.4	Scopes	506
16.5.5	Expression Language Name	510
16.5.6	Interzeptoren	510
16.5.7	Dekoratoren	511

17	Testen von EJB-Komponenten	513
17.1	Kurz gefasst	513
17.2	Dies ist kein Buch über Softwaretests!	513
17.3	Der Blick zurück	515
17.4	Warum testen?	515
17.5	Wann testen?	516
17.6	Wie und wo testen?	517
17.6.1	Akzeptanztests und fachliche Tests	518
17.6.2	Integrative Tests	519
17.6.3	Last- und Performanztests	519
17.7	Testen von Enterprise Beans	520
17.7.1	Testen von Stateless Session Beans	522
17.7.2	Testen von Stateful Session Beans	523
17.7.3	Testen von Message-Driven Beans	524
17.7.4	Testen von Persistent Entities	525
17.7.5	Testen von Transaktionen	530
17.7.6	Testen von Interzeptoren	531
17.7.7	Testen von Lebenszyklusmethoden	533
17.7.8	Testen von Timer Services	536
18	Migration von EJB 2.x nach 3.x	537
18.1	Kurz gefasst	537
18.2	Der Blick zurück	537
18.3	Sanfte Migration	538
18.3.1	Gleichzeitiger Betrieb von 2.x- und 3.x-Komponenten	538
18.3.2	Kommunikation zwischen 2.x- und 3.x-Komponenten	540
18.3.3	Migration von Session Beans	544
18.3.4	Migration von Message-Driven Beans	551
18.3.5	Migration von Entity Beans	554
18.3.6	Migration von Data Transfer Objects	559
18.3.7	Migration von Clients	560
18.4	Der Einfluss von EJB 3.x auf J2EE-Entwurfsmuster	561
18.4.1	Business Delegate	562
18.4.2	Session Facade	563
18.4.3	Message Facade/Service Activator	564
18.4.4	EJB Command	565
18.4.5	EJB Home Factory/Service Locator	565
18.4.6	Business Interface	566
18.4.7	Data Transfer Object (DTO)/Value Object	567

18.4.8	DTO Factory	568
18.4.9	Data Transfer Hash Map	568
18.4.10	Value List Handler	569
18.4.11	Generic Attribute Access	570
18.4.12	Data Transfer Row Set	571
18.4.13	Composite Entity	571
18.4.14	Dual Persistent Entity Bean	572
18.4.15	Data Access Command Bean/ Data Access Object (DAO)	572
18.4.16	JDBC for Reading/Fast Lane Reader	573
18.4.17	Version Number	574
18.4.18	Muster zur Generierung von Primärschlüsseln	575
18.4.19	Fazit	575
Anhang		577
Literatur – offline und online		579
Stichwortverzeichnis		583

Vorwort

Dieses Buch ist ein praxisorientiertes Grundlagenwerk über die Version 3.1 der Komponententechnologie Enterprise JavaBeans. Es beschreibt und erläutert die Grundlagen, die Konzepte und die praktische Anwendung von EJB 3.1 sowie die Migration von bestehenden Applikationen, die auf Vorgängerversionen der EJB- bzw. Java-EE-Technologie basieren.

*Praxisorientierte
Sichtweise*

Der Schwerpunkt dieses Buches liegt auf der praxisnahen Vermittlung von grundlegendem und tiefer gehendem Wissen über die Enterprise-JavaBeans-3-Technologie. Wir komplettieren die Einführung und Vertiefung durch praxisrelevante Themen, die nicht direkt im EJB-Standard behandelt werden. Dazu gehören das Testen von (verteilten) Komponenten, die Persistenzabbildung von EJB-Komponenten (Java Persistence API, JPA) und deren Verwendung in der Benutzeroberfläche (Contexts and Dependency Injection for the Java EE Platform, CDI) sowie die Beschreibung von Migrationspfaden für bestehende J2EE-Systeme nach EJB 3.

Schwerpunkt

Bei der Konzeption und Umsetzung dieses Buches haben wir sehr großen Wert auf eine praxisorientierte Sichtweise gelegt. Die Technologie selbst, aber auch die weiterführenden Themen werden anschaulich anhand eines durchgängigen Beispiels beschrieben. Best Practices aus dem Erfahrungsschatz der Autoren, gesammelt in verschiedenen Softwareentwicklungsprojekten, runden das Bild ab.

Durchgängiges Beispiel

Was ist neu in der zweiten Auflage?

Der Unterschied zwischen der Version 3.0, die wir in der ersten Ausgabe dieses Buches vorgestellt haben, und der hier beschriebenen Version 3.1 ist klein, wenn man allein die Versionsnummer als Maß wählt. Inhaltlich hat sich in der EJB-Spezifikation und in deren Umfeld jedoch viel getan:

EJB 3.1 Lite

EJB 3.x ist im Vergleich zu den Vorgängerversionen ein leichtgewichtiges Framework. Wer es noch leichtgewichtiger braucht, der kann die abgespeckte Variante namens EJB 3.1 Lite verwenden, die nur die allernötigsten EJB-Funktionen beinhaltet.

Session Beans

- Aufrufe können asynchron erfolgen,
- alle `public`-Methoden werden automatisch ohne explizites Interface zur Verfügung gestellt (No-Interface View),
- die Singleton Session Bean feiert ihre Premiere.

Java Persistence API (JPA)

Die Java Persistence API (JPA) ist zwar in einer eigenen Spezifikation beschrieben, darf aber in einem EJB-Buch nicht fehlen. Was wäre eine Business-Applikation ohne Persistenz? Die Version 2.0 der JPA bietet eine ganze Reihe von Neuerungen, darunter auch die folgenden:

- Abfragen können nicht nur mit der Java Persistence Query Language (JPQL), sondern jetzt auch mithilfe der Criteria API formuliert werden,
- optimistisches und pessimistisches Locking ist nun möglich und gezielt steuerbar,
- der Entity Manager hat deutliche Verbesserungen erfahren,
- zur Optimierung von Datenzugriffen steht ein Level-2-Cache zur Verfügung und
- mittels Orphan Removal werden verwaiste Entitäten eines Objekt-netzes automatisch gelöscht.

Der Lebensraum der Enterprise Beans

Die in der Spezifikation definierten globalen JNDI-Namen müssen von allen EJB-3.1-Implementierungen unterstützt werden; damit sind JNDI-Namen endlich portabel.

Timer Service

- Timer können auf Basis kalenderbasierter Zeitpläne definiert und automatisch angelegt werden (Annotation `@Schedule`),
- wobei die Zeitangaben sich auf eine bestimmte Zeitzone beziehen können.

CDI

Auch bei der Contexts and Dependency Injection (CDI) handelt es sich streng genommen um eine separate Spezifikation. Ihr zentrales Thema ist die Standardisierung und Vereinfachung des Zusammenspiels zwischen der Präsentationsschicht JavaServer Faces (JSF) und der mit EJB-Technologie implementierten Geschäftslogik. Mit dem Überblick über CDI runden wir das Themenspektrum dieses Buches ab; schließlich haben viele Applikationen ein Web-Frontend und müssen sich deshalb mit der Frage auseinandersetzen, wie die Enterprise-Beans möglichst transformationsarm an der Benutzeroberfläche dargestellt und manipuliert werden können.

Testen von EJB-Komponenten

- Als Mock-Framework für die Tests verwenden wir jetzt Mockito,
- ausgeführt werden die Tests im neuen Embeddable Container, der sich gut in eine Continuous-Integration-Umgebung einfügt.

Für wen ist dieses Buch?

Dieses Buch richtet sich an Softwareentwickler, Softwarearchitekten und (technische) Projektleiter, die sich mit dem Design und der Entwicklung von Enterprise-Applikationen auf der Basis von Enterprise JavaBeans 3 beschäftigen.

*Softwareentwickler,
-architekten und
(technische) Projektleiter*

Wenn Sie praktische Hinweise für die Entwicklung leichtgewichtiger, mächtiger, komponentenbasierter Enterprise-Java-Systeme suchen, die auf Konzepten wie POJO (Plain Old Java Objects) und Inversion of Control fußen, dann ist dieses Buch etwas für Sie.

Sie sollten fundierte Kenntnisse der objektorientierten Softwareentwicklung, der Unified Modeling Language (UML) sowie der Programmiersprache Java mitbringen. Auch mit den grundlegenden Konzepten von Java 6 sollten Sie vertraut sein. Insbesondere Metadaten (Annotationen) und Generics werden in der EJB-3.1-Spezifikation und in unserer Beispielapplikation intensiv verwendet.

Voraussetzungen

Von wem ist dieses Buch?

Auch wenn auf dem Buchcover nur sechs Autoren genannt sind, so haben tatsächlich neun Personen ihren Beitrag dazu geleistet.



*V.l.n.r.: Oliver Ihns,
Jo Ehm, Stefan M. Heldt,
Holger Koschek,
Carsten Sahling,
Roman Schlömmner,
Norman Erck,
Daniel Steinhöfer,
Carl Anders Düvel*

Oliver Ihns

beschäftigt sich seit mehr als 20 Jahren mit Architekturen und Technologien verteilter, objekt- und serviceorientierter Systeme als Berater, Coach, Referent auf Fachkonferenzen und Autor von Büchern und Fachartikeln. Er ist Mitglied in verschiedenen Expert Groups (u. a. EJB 3.1, JPA 1.0 und WebBeans/CDI) im Java Community Process (JCP) und definiert und entwickelt auf diese Weise federführend Kerntechnologien der Java-Welt mit. Sein Tätigkeitsfeld erstreckt sich von den technologischen Grundlagen von verteilten Systemen über die Architekturberatung und das Architekturmanagement (EAM, SOA, Bebauungspläne) bis hin zu den strategischen Aspekten von Unternehmensarchitekturen und der Auditierung von Systemen. Oliver Ihns wurde 2005 von Sun Microsystems in den kleinen Kreis der Java Champions berufen. Er ist Mitglied des Vorstands der Holisticon AG, einer international tätigen Management- und IT-Beratung.

Stefan M. Heldt

arbeitet als Berater, Coach und Trainer bei der Holisticon AG. Als Softwarearchitekt unterstützt er Unternehmen im Bereich Architekturen und Technologien für verteilte unternehmenskritische Anwendungen sowie bei der Modellierung und Implementierung von Geschäftsprozessen. Neben seiner beratenden Tätigkeit ist er als Autor von Fachartikeln und als Referent auf Konferenzen präsent. Als Mitglied

der Expert Group für EJB 3.2 im Java Community Process (JCP) wird er diese Technologie zukünftig an der Quelle aktiv mitgestalten.

Holger Koschek

arbeitet als Berater und Coach bei der Holisticon AG. Seit 1997 ist er in kleinen bis sehr großen, mitunter auch verteilten internationalen IT-Projekten unterwegs. Aufbauend auf seinen Erfahrungen mit der Architektur und Entwicklung verteilter objektorientierter Anwendungssysteme unterstützt er Unternehmen bei der Einführung agiler Vorgehensweisen sowie bei der Modernisierung ihrer Softwarearchitekturen. Sein Wissen gibt er gerne in Form von Büchern, Fachartikeln und Konferenzbeiträgen weiter. Holger Koschek ist Autor der »Geschichten vom Scrum« (dpunkt.verlag 2009) und Koautor von »Unternehmensportale« (mit Martina Großmann, Springer-Verlag 2005).

Jo Ehm

ist Senior Consultant bei der Holisticon AG. Er ist GPM/IPMA-zertifizierter Projektmanager und unterstützt neben seiner Arbeit als Projektmanager Kundenprojekte auch als Softwarearchitekt, Coach und Lead Software Engineer. Sein Fokus liegt auf den Technologien und Architekturen für serviceorientierte und webbasierte Enterprise-Applikationen sowie auf der Modellierung und Implementierung von Geschäftsprozessen. Daneben hat er einen Schwerpunkt auf (agile) Projektmanagementmethoden und Testmethoden gelegt.

Carsten Sahling

ist Senior Consultant bei der Holisticon AG. Er ist Softwarearchitekt und Experte für Multiprojektmanagement. Dank seiner GPM/IPMA-Projektmanager-Zertifizierung und langjähriger Erfahrung mit der Architektur und der Entwicklung objektorientierter und komponentenorientierter verteilter Systeme verbindet er erfolgreich Führungsqualität in großen Projekten mit softwaretechnischer Expertise. Seit einigen Jahren beschäftigt er sich intensiv mit agilen Methoden und hat als Certified ScrumMaster und Certified Scrum Professional in der Praxis bewiesen, dass agile Methoden auch bei komplexen Architekturen der bessere Weg sind.

Roman Schlömmer

ist Senior Consultant bei der Holisticon AG. Er ist Certified Scrum-Master und seit über 10 Jahren als Business Analyst, Softwarearchitekt, Trainer, Coach und leitender Entwickler in IT-Projekten für überwiegend große Unternehmen tätig. Sein Fokus erstreckt sich von der technologischen Basis objekt-, komponenten- und serviceorientierter Architekturen für verteilte Unternehmensanwendungen über agile Softwareentwicklungsprozesse bis hin zur (Enterprise-) Architekturberatung. In seiner beruflichen Praxis hat Roman Schlömmer in diesen Bereichen häufig mit verteilten Teams in einem globalen, multikulturellen Umfeld gearbeitet.

Carl Anders Düvel

ist Consultant bei der Holisticon AG. Sein Wissen erstreckt sich von den Technologien für objekt-, komponenten- und serviceorientierte Architekturen für verteilte Unternehmensanwendungen bis zur methodisch geführten Anforderungsanalyse. Dabei hilft ihm seine fundierte theoretische sowie praktische Erfahrung im Bereich der Qualitätssicherung von Softwareentwicklungsprojekten und der anwenderzentrierten Softwareentwicklung.

Norman Erck

arbeitet als Consultant bei der Holisticon AG. Nach ersten Praxiserfahrungen im Bereich des E-Commerce und der wissenschaftlichen Tätigkeit am Lehrstuhl für Datenbank- und Informationssysteme der BTU Cottbus sowie einem Exkurs in die Automobilindustrie unterstützt er Unternehmen bei der Etablierung moderner Webtechnologien. Dabei konzentriert er sich auf die Themengebiete Web, E-Business und Portale. Als Mitglied der Expert Group für Contexts and Dependency Injection (CDI) im Java Community Process (JCP) gestaltet Norman Erck eine wichtige Java-EE-Technologie aktiv mit.

Daniel Steinhöfer

ist Consultant bei der Holisticon AG und seit mehreren Jahren als Softwareingenieur und Berater in IT-Projekten im Einsatz. Der Entwurf und die Umsetzung von komponentenbasierten, verteilten Systemen und stabilen Softwarearchitekturen, insbesondere im JEE-Umfeld, bilden den Schwerpunkt seiner Arbeit. Testgetriebene Entwicklung und agile Vorgehensweisen hat der Certified ScrumMaster bei verschiedenen namhaften Kunden mit Erfolg einsetzen und etablieren können.

Danksagungen

Die Erstellung eines Buches kostet enorm viel Zeit, ist mühsam und zwischendurch immer mal wieder mit Rückschlägen verbunden (vor allem, wenn die Reviewer einem ganze Kapitel um die Ohren hauen). Umso wichtiger ist es, Menschen um sich zu haben, die einen dabei unterstützen.

Wir bedanken uns ganz herzlich bei allen Menschen, die uns – in welcher Form auch immer – bei der Erstellung dieses Buches geholfen haben. Unser Dank gilt

- dem Team vom dpunkt.verlag, allen voran René Schönfeldt, für das Vertrauen in dieses Buchprojekt und die gute Zusammenarbeit,
- Dierk Harbeck für seine Beiträge zur ersten Auflage, die in dieser zweiten Auflage weiterleben,
- Jochen Jörg für die Ticket2Rock-Idee und die tatkräftige Unterstützung bei der Realisierung,
- Simon Zambrowski für Ticket2Rock-Support und viele wertvolle Diskussionen, in denen er schonungslos die Sicht der Leser vertreten hat,
- Jo Ehm für die Abbildungen, die dieses gehaltvolle Thema verständlich und ansprechend in Szene setzen und unumstritten mehr sagen als die berühmten tausend Worte,
- allen aufmerksamen Lesern der ersten Auflage, deren Verbesserungsvorschläge und Fehlerkorrekturen in die zweite Auflage eingeflossen sind,
- unseren Familien für ihr Verständnis und ihre Unterstützung.

Widmung

Für Martina, Joshua, Julina, meinen Vater und Sven.

O.I.

Für Astrid, Mattis und Jasper.

S.M.H.

Für Andrea, Nele, Marit und Lotta.

H.K.

Für Britta

(für die Geduld und die gestohlene Zeit).

J.E.

*Für Anni, Annii und Svenni, meine drei Großen,
und det kleene Krümelchen.*

C.S.

Für Pubbi und Rona.

R.S.

Für Jaqueline, meine Eltern & Marie.

C.A.D.

Für meine Freunde

(die auch mal ohne mich das Nachtleben genießen konnten).

N.E.

*Für Mama, Papa, Roland, Diana –
mein Herz, meine Freunde und meine Koautoren.*

D.S.

Ihre Kommentare und Anmerkungen

Genau wie Softwareprogramme, so sind auch Bücher weder fehlerfrei, noch decken sie alle Ideen, Varianten oder Kniffe des jeweiligen Themas ab. Fehler kann man korrigieren. Was fehlt, kann man ergänzen oder nachtragen.

Sollten Sie Kommentare und/oder Anmerkungen zu diesem Buch haben, dann zögern Sie nicht, uns diese zukommen zu lassen. Über Rückmeldungen zum Buch oder zum Thema EJB 3 freuen wir uns sehr. Sie erreichen uns unter der E-Mail-Adresse post@ejb3buch.de oder persönlich unter <vorname>.<nachname>@holisticon.de.