

Inhaltsverzeichnis

Geleitwort	1
Vorwort	5
Für wen ist dieses Buch?	5
Von wem ist dieses Buch?	6
Danksagungen	8
Widmung	9
Ihre Kommentare und Anmerkungen	9
1 Einleitung – Jetzt wird's leicht!	11
1.1 Die Evolution der Enterprise JavaBeans	11
1.1.1 Der Fluch der Komplexität	12
1.1.2 Kritik an den früheren EJB-Versionen	12
1.1.3 Ein neues Denken schafft eine neue Architektur ...	13
1.2 Aufbau des Buches	14
1.2.1 Die Themenbereiche	15
1.2.2 Die Kapitel: Struktur und Inhalte	16
1.3 Konventionen	21
1.3.1 Notation	21
1.3.2 Literaturverweise bzw. Referenzen	22
1.3.3 Quellcode	22
1.3.4 Piktogramme	22
1.3.5 Verwendung von Anglizismen	23
1.3.6 Gleichberechtigung in der Sprache	23
1.4 Die Beispielapplikation »Ticket2Rock«	24
1.4.1 Kurzbeschreibung	24
1.4.2 Anwendungsfälle	25
1.4.3 Fachliche Entitäten	26

1.5	Verwendete Technologien und Produkte	29
1.6	Die Website zum Buch	30
2	Leichtgewichtige, POJO-basierte Enterprise-Applikationen	31
2.1	Kurz gefasst	31
2.2	Der Blick zurück	31
2.3	Einleitung	32
2.4	Hauptziele für EJB 3.0	35
2.4.1	Motive	35
2.4.2	»Einfach machen!«	36
2.4.3	Vereinfachung der Mikroarchitektur von EJB-Komponenten	36
2.4.4	Vereinfachung des Entwicklungsprozesses	37
2.4.5	Vereinfachung der Nutzung von EJB-Kompo- nenten	38
2.4.6	Neuentwicklung einer leistungsfähigen Persistenz- lösung	39
2.5	POJOs und POJIs	40
2.5.1	POJO	40
2.5.2	POJI	41
2.5.3	Unterschiede zwischen EJB 3.0 und EJB 2.x	41
2.6	Inversion of Control und Dependency Injection	42
2.6.1	Ziele im Kontext von EJB 3.0	43
2.6.2	Auswirkungen auf die EJB-3.0-Architektur	44
2.6.3	»Hollywood, wir kommen!« – ein Beispiel	44
2.7	Annotationen und Deployment-Deskriptoren	46
2.7.1	Einführung	46
2.7.2	Deployment-Deskriptoren – so schlecht wie ihr Ruf?	47
2.7.3	Annotationen	47
2.7.4	Ein Plädoyer für Deployment-Deskriptoren	51
2.7.5	Der Deployment-Deskriptor hat das letzte Wort ..	52
2.8	Configuration by Exception	52
3	EJB-Komponentenarchitektur	55
3.1	Kurz gefasst	55
3.2	Grundlegende Konzepte	55
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	69
3.3.3	Persistent Entity	71
3.4	Aufrufmodelle	71
3.4.1	Aufrufmodell: »synchron, entfernt«	72
3.4.2	Aufrufmodell »synchron, lokal«	76
3.4.3	Aufrufmodell »asynchron, nachrichtenbasiert« ...	79
3.4.4	Aufrufmodelle und EJB-Typen im Überblick	84
4	Session Beans	85
4.1	Kurz gefasst	85
4.2	Der Blick zurück	85
4.3	Einführung	86
4.4	Stateless Session Beans	87
4.4.1	Transaktionen	88
4.4.2	Instanz-Pooling	89
4.4.3	Web Services	89
4.5	Stateful Session Beans	89
4.5.1	Aktivierung und Passivierung	91
4.5.2	Transaktionen	92
4.6	Stateless und Stateful Session Beans im Vergleich	92
4.7	Mikroarchitektur einer Session Bean	93
4.7.1	Namenskonventionen	94
4.7.2	Zusammenspiel der Elemente	95
4.7.3	Erzeugen von Session Beans	96
4.7.4	Löschen von Session Beans	97
4.8	Lebenszyklus von Stateless Session Beans	98
4.8.1	Zustand »does not exist«	98
4.8.2	Übergang von »does not exist« zu »method-ready pool«	98
4.8.3	Zustand »method-ready pool«	99
4.8.4	Übergang von »method-ready pool« zu »does not exist«	99

4.9	Lebenszyklus von Stateful Session Beans	100
4.9.1	Zustand »does not exist«	101
4.9.2	Übergang von »does not exist« zu »method-ready«	101
4.9.3	Zustände »method-ready« und »method-ready in TX«	101
4.9.4	Zustand »passive«	102
4.9.5	Übergang in den Zustand »does not exist«	103
4.10	Business Interface	104
4.11	Bean-Klasse	107
4.11.1	Deklarative Transaktionalität	110
4.11.2	Transaktionen in Handarbeit	115
4.11.3	Transaktionen im Ausnahmezustand	118
4.11.4	EJB-Kontext	119
4.11.5	Checkliste	121
4.12	Timer Service	122
5	Session Beans als Web Service	123
5.1	Kurz gefasst	123
5.2	Der Blick zurück	123
5.3	Was ist ein Web Service?	124
5.4	Stateless Session Beans als Web Service	126
5.4.1	@WebService	128
5.4.2	@SOAPBinding	132
5.4.3	@WebMethod	139
5.4.4	@Oneway	140
5.4.5	@WebParam	140
5.4.6	@WebResult	143
5.4.7	@HandlerChain	145
5.5	Szenarien beim Einsatz von Web Services	145
5.6	Ein Web-Service-Client	146
6	Message-Driven Beans	147
6.1	Kurz gefasst	147
6.2	Der Blick zurück	147
6.3	Nachrichtenbasierte Kommunikation	148
6.3.1	Charakteristika und Vorteile	150
6.3.2	Kommunikationsmodelle	151

6.4	Java Message Service (JMS)	153
6.4.1	Service Provider Interface	153
6.4.2	JMS API	154
6.5	Charakteristika von Message-Driven Beans	156
6.5.1	JMS Message-Driven Beans	159
6.5.2	Connector-based Message-Driven Beans	159
6.6	Lebenszyklus von Message-Driven Beans	161
6.7	Transaktionalität	161
6.8	Bean-Klasse	162
6.8.1	@MessageDriven	163
6.8.2	@ActivationConfigProperty	165
6.8.3	Message-Listener-Interface	167
6.8.4	Beantworten von Nachrichten	169
6.8.5	Checkliste	170
6.9	Deployment-Deskriptor	170
6.10	Timer Service	172
6.11	Ein JMS-Client	173
6.12	Message Linking	174
7	Entity Beans	181
7.1	Kurz gefasst	181
7.2	Der Blick zurück	181
7.3	Aus Entity Beans werden Persistent Entities	182
8	Persistenzabbildung	183
8.1	Kurz gefasst	183
8.2	Der Blick zurück	183
8.3	Persistenz? Abbildung?	184
8.4	Persistent Entities	186
8.4.1	Lightweight	187
8.4.2	Persistent	188
8.4.3	Domain Object	188
8.4.4	Lebenszyklus	189
8.5	Persist my POJO!	189
8.5.1	Annotation oder Deployment-Deskriptor?	189
8.5.2	Beispiel	190

8.6	Grundkonzepte	193
8.6.1	Persistence Provider	193
8.6.2	Entity-Manager	194
8.6.3	Persistenzeinheit	197
8.6.4	Persistenzkontext	198
8.7	Deployment-Deskriptoren	200
8.7.1	persistence.xml	200
8.7.2	orm.xml	202
8.8	Arbeiten mit dem Entity-Manager	203
8.8.1	Dauerhaftes Speichern in der Datenbank (persist)	204
8.8.2	Aktualisieren des persistenten Objektzustands (merge)	204
8.8.3	Löschen einer Persistent Entity (remove)	205
8.8.4	Finden einer Persistent Entity in der Datenbank (find, getReference)	205
8.8.5	Sofortiges Ausführen der Datenbankoperation (flush)	206
8.8.6	Blockieren einer Persistent Entity (lock)	207
8.8.7	Aktualisieren des Zustands der Objektinstanz (refresh)	207
8.8.8	Leben im Persistenzkontext (clear, contains)	207
8.8.9	Abfragen (create...Query)	208
8.8.10	Transaktionen (joinTransaction)	208
8.8.11	Zugriff auf den Persistence Provider (getDelegate)	208
8.8.12	Beenden des Entity-Managers (close)	208
8.9	Abbildung von Datentypen	209
8.9.1	Zugriff auf persistente Felder	209
8.9.2	Einfache Datentypen (@Basic)	210
8.9.3	Eingebettete Objekte (@Embeddable)	212
8.9.4	Große Objekte (@Lob)	212
8.9.5	Datum und Zeit (@Temporal)	212
8.9.6	Aufzählungen (@Enumerated)	213
8.10	Abbildung in Datenbanktabellen	214
8.10.1	@Table	215
8.10.2	@Column	215
8.11	Primärschlüssel	217
8.11.1	Einfache Primärschlüssel (@Id)	218
8.11.2	Zusammengesetzte Primärschlüssel (@IdClass, @EmbeddedId)	218
8.11.3	Generierung von Primärschlüsseln	225

8.12	Abbildung von Objektbeziehungen	231
8.12.1	Die glorreichen Sieben	231
8.12.2	Unidirektionale Eins-zu-Eins-Beziehung	234
8.12.3	Bidirektionale Eins-zu-Eins-Beziehung	237
8.12.4	Unidirektionale Eins-zu-Viele-Beziehung	239
8.12.5	Bidirektionale Eins-zu-Viele-Beziehung	243
8.12.6	Unidirektionale Viele-zu-Eins-Beziehung	246
8.12.7	Bidirektionale Viele-zu-Eins-Beziehung	247
8.12.8	Unidirektionale Viele-zu-Viele-Beziehung	247
8.12.9	Bidirektionale Viele-zu-Viele-Beziehung	249
8.12.10	Kaskadieren von Persistenzoperationen	252
8.13	Eingebettete Objekte	255
8.14	Abbildung auf mehrere Datenbanktabellen	258
8.14.1	»Single-Table-Mapping«	258
8.14.2	Multi-Table-Mapping	259
8.15	Vererbung und Polymorphie	263
8.15.1	Erben und Vererben	264
8.15.2	single table per class hierarchy strategy (SINGLE_TABLE)	266
8.15.3	single table per concrete entity class strategy (TABLE_PER_CLASS)	272
8.15.4	joined subclass strategy (JOINED)	275
8.16	Fetching-Strategien	280
8.16.1	Eager Load	281
8.16.2	Lazy Load	281
8.16.3	Deklaration der Fetching-Strategie	282
8.16.4	Lazy Load und Detached Objects	285
9	Abfragen mit JPQL	287
9.1	Kurz gefasst	287
9.2	Der Blick zurück	287
9.3	Abfragen	288
9.3.1	Queries	288
9.3.2	Named Queries	294
9.3.3	Native Queries	295
9.4	Java Persistence Query Language (JPQL)	300
9.4.1	SELECT	300
9.4.2	FROM	305
9.4.3	WHERE	308

9.4.4	ORDER BY	311
9.4.5	GROUP BY	312
9.4.6	HAVING	312
9.4.7	Schreibende Massenoperationen	313
10	Der Lebensraum der Enterprise Beans	315
10.1	Kurz gefasst	315
10.2	Der Blick zurück	316
10.3	Der Enterprise Naming Context	316
10.4	Konfigurationsalternativen	317
10.4.1	Annotationen	317
10.4.2	Deployment-Deskriptoren	317
10.4.3	Kombination von Annotationen und Deployment-Deskriptoren	318
10.5	Arbeiten mit dem ENC	318
10.5.1	Bestückung mittels Deployment-Deskriptoren ...	319
10.5.2	Bestückung mittels Annotationen	320
10.5.3	Zugriff via JNDI-Lookup	320
10.5.4	Zugriff via EJBContext	321
10.5.5	Dependency Injection mit Deployment- Deskriptoren	322
10.5.6	Dependency Injection mit Annotationen	323
10.6	Auswirkungen auf den Softwaretest	326
10.7	Ressourcen-Typen	327
10.7.1	Enterprise Beans (@EJB)	327
10.7.2	Extern verwaltete Ressourcen (@Resource)	332
10.7.3	Resource Environment Entries (@Resource)	337
10.7.4	Umgebungsvariablen (@Resource)	340
10.7.5	Persistenzkontext (@PersistenceContext)	343
10.7.6	Persistenzeinheiten (@PersistenceUnit)	348
10.7.7	Message Destinations	352
10.7.8	Web Services (@WebServiceRef)	352
11	Callback-Mechanismen	357
11.1	Kurz gefasst	357
11.2	Der Blick zurück	357
11.3	Inversion of Control	358

11.4	Deklaration einer Callback-Methode	359
11.4.1	Callback-Annotationen	359
11.4.2	Deklaration im Deployment-Deskriptor	360
11.4.3	Für jede Bean die passenden Callbacks	362
11.4.4	Regeln für Callback-Methoden	362
11.5	Aufrufreihenfolge für Callback-Methoden	364
11.6	Callbacks für Stateless Session Beans	365
11.6.1	@PostConstruct	366
11.6.2	@PreDestroy	368
11.7	Callbacks für Stateful Session Beans	368
11.7.1	@PostConstruct und @PreDestroy	369
11.7.2	@PrePassivate	370
11.7.3	@PostActivate	371
11.8	Callbacks für Message-Driven Beans	371
11.8.1	@PostConstruct und @PreDestroy	372
11.8.2	@PostActivate und @PrePassivate werden ignoriert	372
11.9	Callbacks für Persistent Entities	372
11.9.1	Aufrufreihenfolge im Objektverbund	373
11.9.2	@PrePersist und @PostPersist	376
11.9.3	@PreUpdate und @PostUpdate	376
11.9.4	@PreRemove und @PostRemove	377
11.9.5	@PostLoad	377
12	Interzeptoren	379
12.1	Kurz gefasst	379
12.2	Der Blick zurück	380
12.3	Was ist aspektorientierte Programmierung?	380
12.4	Klassifikation	382
12.4.1	Interzeptoren für Geschäftsmethoden	383
12.4.2	Interzeptoren für Lebenszyklus-Ereignisse	383
12.4.3	Default-Interzeptoren	384
12.4.4	Entity Listener	384
12.4.5	Default Entity Listener	385
12.5	Interzeptoren für Geschäftsmethoden	385
12.5.1	Definition	385
12.5.2	Verwendung	387
12.5.3	InvocationContext	390

12.5.4	Aufrufreihenfolge	390
12.5.5	Ausnahmebehandlung	392
12.5.6	@AroundInvoke-Methode in der Bean-Klasse ...	392
12.6	Interzeptoren für Lebenszyklus-Ereignisse	393
12.6.1	Definition	393
12.6.2	Verwendung	394
12.6.3	Aufrufreihenfolge	394
12.6.4	Ausnahmebehandlung	395
12.7	Default-Interzeptoren	396
12.8	Entity Listener	397
12.8.1	Definition	397
12.8.2	Verwendung	399
12.8.3	Aufrufreihenfolge	400
12.9	Default Entity Listener	401
12.10	Sind Interzeptoren und Entity Listener aspektorientiert? ..	402
13	Timer Service	405
13.1	Kurz gefasst	405
13.2	Der Blick zurück	406
13.3	Timer Service API	407
13.3.1	Die Timeout-Methode	409
13.3.2	Das Interface javax.ejb.TimerService	410
13.3.3	Das Interface javax.ejb.Timer	412
13.4	Timer Service und Transaktionen	413
13.5	Timer Service und EJB-Typen	413
13.5.1	Stateless Session Bean Timer	414
13.5.2	Message-Driven Bean Timer	415
13.5.3	Timer und Entitäten	416
13.6	Checkliste	417
14	Sicherer Zugriff auf EJB-Komponenten	419
14.1	Kurz gefasst	419
14.2	Der Blick zurück	419
14.3	Überblick	420
14.4	Authentifizierung	421
14.5	Sicherheitsrollen	422

14.6	Verwendung von Rollen	422
14.6.1	@RolesAllowed	422
14.6.2	@PermitAll	424
14.6.3	@DenyAll	425
14.7	Ausführen in einem anderen Kontext (@RunAs)	426
14.8	Identität von Message-Driven Beans und Timer Services ..	427
14.9	Programmgesteuerter Zugriff auf den Security-Kontext ..	427
14.10	Regeln für Security-Annotationen	429
15	Testen von EJB-Komponenten	431
15.1	Kurz gefasst	431
15.2	Dies ist kein Buch über Softwaretests!	431
15.3	Der Blick zurück	432
15.4	Warum testen?	433
15.5	Wann testen?	433
15.6	Wie und wo testen?	435
15.6.1	Akzeptanztests und fachliche Tests	436
15.6.2	Integrative Tests	436
15.6.3	Last- und Performanztests	437
15.7	Testen von Enterprise Beans	438
15.7.1	Testen von Stateless Session Beans	440
15.7.2	Testen von Stateful Session Beans	441
15.7.3	Testen von Message-Driven Beans	442
15.7.4	Testen von Persistent Entities	443
15.7.5	Testen von Transaktionen	448
15.7.6	Testen von Interzeptoren	449
15.7.7	Testen von Lebenszyklusmethoden	452
15.7.8	Testen von Timer-Services	454
16	Migration von EJB 2.x nach 3.0	455
16.1	Kurz gefasst	455
16.2	Der Blick zurück	455
16.3	Sanfte Migration	456
16.3.1	Gleichzeitiger Betrieb von 2.x- und 3.0-Komponenten	456
16.3.2	Kommunikation zwischen 2.x- und 3.0-Komponenten	458
16.3.3	Migration von Session Beans	461

16.3.4	Migration von Message-Driven Beans	468
16.3.5	Migration von Entity Beans	471
16.3.6	Migration von Data Transfer Objects	476
16.3.7	Migration von Clients	477
16.4	Der Einfluss von EJB 3.0 auf J2EE-Entwurfsmuster	478
16.4.1	Business Delegate	479
16.4.2	Session Facade	480
16.4.3	Message Facade / Service Activator	481
16.4.4	EJB Command	482
16.4.5	EJB Home Factory / Service Locator	482
16.4.6	Business Interface	483
16.4.7	Data Transfer Object (DTO) / Value Object ...	484
16.4.8	DTO Factory	485
16.4.9	Data Transfer Hash Map	485
16.4.10	Value List Handler	486
16.4.11	Generic Attribute Access	487
16.4.12	Data Transfer Row Set	487
16.4.13	Composite Entity	488
16.4.14	Dual Persistent Entity Bean	489
16.4.15	Data Access Command Bean / Data Access Object (DAO)	489
16.4.16	JDBC for Reading / Fast Lane Reader	490
16.4.17	Version Number	490
16.4.18	Muster zur Generierung von Primärschlüsseln .	491
16.4.19	Fazit	492
17	Literatur – offline und online	495
18	Stichwortverzeichnis	499