

Robert F. Beeger · Arno Haase · Stefan Rook · Sebastian Sanitz

Hibernate

**Persistenz in Java-Systemen mit Hibernate
und dem Java Persistence API**

2., überarbeitete und erweiterte Auflage



dpunkt.verlag

Robert F. Beeger
robert.beeger@akquinet.de

Arno Haase
arno.haase@haase-consulting.com

Stefan Roock
stefan.roock@akquinet.de

Sebastian Sanitz
sebastian.sanitz@akquinet.de

Lektorat: René Schönfeldt
Copy-Editing: Annette Schwarz, Ditzingen
Herstellung: Birgit Bäuerlein
Umschlaggestaltung: Helmut Kraus, www.exclam.de
Druck und Bindung: Koninklijke Wöhrmann B.V., Zutphen, Niederlande

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über <http://dnb.ddb.de> abrufbar.

ISBN 978-3-89864-447-1

2., überarbeitete und erweiterte Auflage 2007
Copyright © 2007 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.

5 4 3 2 1 0

Inhalt

1	Einleitung	1
Teil I	Erste Schritte	9
2	OR-Mapping	11
3	Hibernate – ein einfaches Beispiel	35
Teil II	Hibernate-Technologie	57
4	Mapping-Dateien	59
5	Konfiguration	91
6	Sessions	117
7	Datenbankabfragen	151
Teil III	Architekturen	167
8	Architektur für Hibernate-basierte Rich-Client-Anwendungen	169
9	Architektur für Hibernate-basierte Webanwendungen	185
10	Architektur für Hibernate-basierte Client-Server-Anwendungen	211

Teil IV	Spezialthemen	235
11	Performancetuning	237
12	Fortgeschrittenes Mapping	261
13	XDoclet	293
14	Anpassungen	305
15	Die Java Persistence API aus EJB 3	313
16	Spring	341
17	IDE-Erweiterungen	349
	Ausblick	355
Anhang		357
A	Migration von JDO zu Hibernate und von Hibernate zu JDO	359
B	Testen mit Hibernate	369
	Index	379

Inhaltsverzeichnis

1	Einleitung	1
Teil I	Erste Schritte	9
2	OR-Mapping	11
2.1	RDBMS-Crashkurs	11
2.2	ER-Diagramme	14
2.3	SQL-Crashkurs	15
2.4	ACID und Transaktionen	17
2.5	Ansätze zum OR-Mapping	21
2.6	Probleme beim OR-Mapping	22
2.6.1	Konvertierung von Basisdatentypen	23
2.6.2	Identität	23
2.6.3	Mengenwertige Attribute und Kardinalitäten ...	24
2.6.4	Vererbung	25
2.6.5	Sonstiges	29
2.7	Transparente Persistenz	29
2.8	Standards und Werkzeuge	30
2.9	Referenzen	32
3	Hibernate – ein einfaches Beispiel	35
3.1	Installation von Hibernate und H2	35
3.2	Vorbereitung der Entwicklungsumgebung	37
3.2.1	Die Auswahl der richtigen Bibliotheken	37
3.2.2	Installation der Plugins für den Zugriff auf die Datenbank	39

3.3	Ein erstes Beispiel mit Hibernate	41
3.3.1	Konfiguration	42
3.3.2	Das Beispiel – eine Testklasse für die Grundoperationen von Hibernate	46
3.4	Zusammenfassung	55
3.5	Referenzen	55
Teil II	Hibernate-Technologie	57
4	Mapping-Dateien	59
4.1	Eine einfache Mapping-Datei	59
4.2	Primärschlüssel	62
4.3	Geschäftsobjekte und Wert-Klassen	63
4.4	Eine erste Collection von Wertobjekten	65
4.5	Listen, Bags, Sets, Maps und Arrays	69
4.5.1	Bag	69
4.5.2	Set	70
4.5.3	SortedSet	71
4.5.4	Map	72
4.5.5	SortedMap	74
4.5.6	Array	74
4.5.7	Übersicht	74
4.6	Beziehungen zwischen Geschäftsobjekten	76
4.6.1	Referenzen auf Geschäftsobjekte	76
4.6.2	Collections von Geschäftsobjekten	78
4.6.3	Bidirektionale Beziehungen	80
4.7	Vererbung	83
4.7.1	Polymorpher Zugriff	84
4.7.2	Tabelle je Klassenhierarchie	85
4.7.3	Tabelle je konkrete Klasse	86
4.7.4	Tabelle je Klasse	88
4.8	Zusammenfassung	89
4.9	Referenzen	90
5	Konfiguration	91
5.1	Initialisierung von Hibernate	91
5.1.1	Einfaches Beispiel	91
5.1.2	Configuration-Objekte	93

5.2	Woher kommen die Connections?	93
5.2.1	Datenbanktreiber	94
5.2.2	Connection-Pool	96
5.2.3	DataSource über JNDI	99
5.2.4	Die Anwendung liefert die Connection	100
5.3	Datenbankdialekte	101
5.4	Verschiedene Quellen für Konfiguration	103
5.4.1	hibernate.properties	103
5.4.2	Methoden zum Setzen von Property's	104
5.4.3	Methoden zum Registrieren von Mappings	105
5.4.4	hibernate.cfg.xml	107
5.5	Logging	109
5.6	Spezielle Themen	111
5.6.1	Transaktionen	111
5.6.2	Caches	112
5.6.3	Lebenszyklus von Objekten	113
5.6.4	Performancetuning	113
5.6.5	Query's	113
5.6.6	Erzeugen von DDL	114
5.6.7	Mapping	114
5.7	Referenzen	115
6	Sessions	117
6.1	Hibernate-Sessions	117
6.1.1	Geschäftsobjekte speichern, laden und löschen	118
6.1.2	Caching	120
6.1.3	Proxys	121
6.1.4	Lebensdauer von Sessions	122
6.2	Transaktionen	122
6.3	Das Hibernate-Zustandsmodell persistenter Objekte ...	125
6.4	Configuration, SessionFactory, Session, Transaction im Konzert	128
6.5	Persistenz über Erreichbarkeit	129
6.6	Sperren	130

6.7	Schnittstelle der Session	135
6.7.1	Allgemeine Verwaltung	135
6.7.2	Umgang mit Connections	136
6.7.3	Umgang mit einzelnen Objekten	136
6.7.4	Umgang mit Abfragen (Querys)	139
6.7.5	Umgang mit dem Cache	140
6.7.6	Hibernate-Zustandsmodell verfeinert	141
6.7.7	Das Statistics-API	142
6.8	Application Server	142
6.9	Caches	143
6.10	Hibernate-Exceptions	144
6.11	Referenzen	150
7	Datenbankabfragen	151
7.1	Die Hibernate Query Language	151
7.2	Typsichere Abfragen mit Criteria	161
7.3	Abfragen mit SQL	163
7.4	Filter	165
7.5	Referenzen	166
Teil III	Architekturen	167
8	Architektur für Hibernate-basierte Rich-Client-Anwendungen	169
8.1	Schichten in Anwendungssystemen	169
8.2	Ausflug: Datenzugriffsobjekte	171
8.3	Model-View-Controller	173
8.4	Umgang mit Sessions	175
8.5	Sperrstrategien	178
8.5.1	Pessimistische Sperrstrategie	178
8.5.2	Optimistische Sperrstrategie	180
8.6	Exceptionhandling	181
8.7	Referenzen	183

9	Architektur für Hibernate-basierte Webanwendungen	185
9.1	Gängige Architekturen	185
9.1.1	JSPs pur	185
9.1.2	Front-Controller	186
9.1.3	Seitenbasierte Architektur	187
9.2	Zugriff auf die persistenten Daten	188
9.2.1	Data Transfer Object	188
9.2.2	Active Record	189
9.2.3	Verwendung eines OR-Mapping-Rahmenwerkes	189
9.3	Kurze Einführung in JavaServer Faces	190
9.3.1	Aufbau einer JSF-Anwendung	190
9.3.2	JSPs	191
9.3.3	Seiten-Beans und Services	192
9.4	Einsatz von Hibernate in Webanwendungen	193
9.4.1	Die benötigten Bibliotheken	193
9.4.2	Die SessionFactory	194
9.4.3	Die Session	196
9.4.4	Die Services	201
9.4.5	Zugriff auf die persistenten Objekte bei der Generierung der Antwort	201
9.4.6	Zusammenstecken aller Seiten-Beans und Services	203
9.5	Ajax	204
9.6	Fortgeschrittenere Themen	205
9.6.1	ThreadLocal oder nicht ThreadLocal	205
9.6.2	CurrentSessionContext	207
9.6.3	Connection-Pooling	207
9.7	Referenzen	208
10	Architektur für Hibernate-basierte Client-Server-Anwendungen	211
10.1	Was sind Client-Server-Anwendungen?	211
10.2	Aufteilung in Client- und Serverteil	213
10.2.1	Lazy Loading mit Hibernate	213
10.2.2	Ansätze ohne Hibernate	214
10.2.3	Zusammenfassende Betrachtungen	215

10.3	Entwicklung einer Client-Server-Anwendung mit JBoss	216
10.3.1	Definition der DataSource	217
10.3.2	Konfiguration von Hibernate	219
10.3.3	Modellierung der Serverseite	220
10.3.4	Verwaltung der Sessions	222
10.3.5	Deklarative Transaktionsverwaltung	224
10.3.6	Lange Transaktionen	227
10.3.7	Entwicklung der Clientseite und Anpassung der Services	228
10.3.8	Anzeige von Massendaten	229
10.4	Deployment der Anwendung im JBoss	230
10.4.1	Ersetzen veralteter Bibliotheken	230
10.4.2	Hibernate-Archive	230
10.5	EJB 3	231
10.6	Webservices	231
10.7	Referenzen	234
Teil IV Spezialthemen		235
11	Performancetuning	237
11.1	Allgemeines zu Performance und Optimierung	237
11.2	Performancetuning und Hibernate	239
11.3	Performancetuning beim Lesen	239
11.3.1	Vorbemerkungen	239
11.3.2	Lazy Loading	240
11.3.3	Fetch-Strategien	243
11.4	Performance beim Schreiben	246
11.4.1	Batched Updates	246
11.4.2	Unveränderliche Objekte	247
11.4.3	UPDATE ohne vorheriges Laden	247
11.4.4	DELETE ohne vorheriges Laden	248
11.4.5	Direkte Verwendung der Connection	248
11.5	Collections	249
11.5.1	Auswahl des Collection-Typs	249
11.5.2	Queries statt großer Collections	251

11.6	Second-Level-Caches	252
11.6.1	Architektur von Second-Level-Caches	253
11.6.2	Verteilung und Transaktionsisolation	253
11.6.3	Auswahl der Cache-Implementierung	255
11.6.4	Festlegen des Caching-Verhaltens	256
11.6.5	Bewertung	259
11.7	Referenzen	259
12	Fortgeschrittenes Mapping	261
12.1	Allgemeines	261
12.1.1	Catalog und Schema	261
12.1.2	Packages in Mappings	262
12.1.3	Unqualifizierte Klassennamen in Querys	263
12.1.4	Column als Element	263
12.2	Hibernate-Typen	264
12.3	Export des SQL-Schemas	266
12.3.1	Das API für den Export	266
12.3.2	SQL-Typen	267
12.3.3	Constraints	267
12.3.4	Indizes	269
12.3.5	Kommentare	270
12.4	Custom Types	271
12.5	Primärschlüssel	274
12.5.1	Mitgelieferte Generatoren für Primärschlüssel	274
12.5.2	Selbst definierte Generatoren für Primärschlüssel	278
12.5.3	Primärschlüssel und der Lebenszyklus von Objekten	278
12.5.4	Zusammengesetzte Primärschlüssel	279
12.6	Zugriffsstrategien auf Java-Attribute	281
12.7	Custom SQL und Stored Procedures	282
12.7.1	Schreibzugriffe	282
12.7.2	Lesende Zugriffe	283
12.7.3	Fazit	285
12.8	Abgeleitete Attribute	286
12.9	<one-to-one>-Mapping	287
12.10	Das Element <join>	288

12.11	<any>-Mapping	289
12.12	Kaskadierung von Operationen	291
13	XDoclet	293
13.1	XDoclet	293
13.2	XDoclet ANT-Tasks	294
13.2.1	Beispiel-Quellcode	296
13.2.2	Weitere Parameter für Hibernate-Module	299
13.3	XDoclet-Tags	299
13.4	Ablage der generierten Dateien	300
13.5	Generierte Mapping-Dateien und Versionsverwaltung ..	301
13.6	Bewertung	302
13.7	Referenzen	304
14	Anpassungen	305
14.1	Interceptoren	305
14.2	Events	307
14.3	Naming Strategies	310
14.4	Referenzen	312
15	Die Java Persistence API aus EJB 3	313
15.1	Mapping mit Annotations	314
15.2	Konfiguration mit Annotationen	317
15.3	Abbildung von Wertobjekten	318
15.4	Referenzen zu Geschäftsobjekten	320
15.5	Lazy Loading	326
15.6	Kaskadierung	328
15.7	Vererbung	328
15.8	EntityManager	330
15.9	PersistenceContext	334
15.10	Java Persistence Query Language	334
15.11	Transaktionen	338
15.12	Callbacks	339
15.13	Referenzen	340

16	Spring	341
16.1	Springs Integration von Hibernate	342
16.2	Der Sinn der Spring-Kapsel um Hibernate	342
16.3	Kurze und lange Transaktionen	343
16.4	DAOs und Services	344
16.5	Beispiel für die Verwendung der Hibernate-Integration in Spring	344
16.5.1	Implementierung des DAOs	344
16.5.2	Konfiguration	346
16.5.3	Verwendung	348
16.6	Referenzen	348
17	IDE-Erweiterungen	349
17.1	Hibernate Tools	349
17.1.1	Installation	349
17.1.2	Einrichten des Projekts für Hibernate Tools ...	350
17.1.3	Editieren von Konfigurations- und Mapping-Dateien	351
17.1.4	HQL-Prototyping	351
17.2	Hibero	352
17.2.1	Installation	352
17.2.2	Einrichtung eines Moduls für Hibero	353
17.2.3	Editieren der Hibernate-Dateien	353
17.2.4	HQL-Prototyping	353
17.3	Referenzen	354
	Ausblick	355
Anhang		357
A	Migration von JDO zu Hibernate und von Hibernate zu JDO	359
A.1	Von JDO zu Hibernate	360
A.1.1	Migration ohne Legacy-Daten	360
A.1.2	Migration mit Legacy-Daten	363
A.1.3	Migration der Anwendung	364
A.2	Von Hibernate zu JDO	366
A.3	Referenzen	367

B	Testen mit Hibernate	369
B.1	Grundlagen	369
B.2	JUnit	370
B.3	Ansatzpunkte für Tests in Hibernate-basierten Systemen	373
B.3.1	Test der Mapping-Dateien	374
B.3.2	Test der Geschäftsobjekte	375
B.3.3	Test der Fachlogik	375
B.3.4	Test der Benutzungsoberfläche	375
B.4	Verwendung von Datenbanken für Tests	375
B.5	Erzeugung von Testdaten	377
B.6	Referenzen	378
	Index	379