

Covers Version 1.2

NHibernate IN ACTION

Pierre Henri Kuate
Tobin Harris
Christian Bauer
Gavin King

FOREWORD BY **Ayende Rahien**

 MANNING



contents

foreword xvii
preface xxi
acknowledgments xxiii
about this book xxv

PART 1 DISCOVERING ORM WITH NHIBERNATE 1

1 Object/relational persistence in .NET 3

1.1 What is persistence? 5

Relational databases 5 ▪ Understanding SQL 6 ▪ Using SQL in .NET applications 6 ▪ Persistence in object-oriented applications 6 ▪ Persistence and the layered architecture 7

1.2 Approaches to persistence in .NET 9

*Choice of persistence layer 9 ▪ Implementing the entities 11
Displaying entities in the user interface 13 ▪ Implementing
CRUD operations 14*

1.3 Why do we need NHibernate? 15

*The paradigm mismatch 15 ▪ Units of work and conversations 16
Complex queries and the ADO.NET Entity Framework 18*

- 1.4 Object/relational mapping 21
 - What is ORM? 21* ▪ *Why ORM? 21*
- 1.5 Summary 23

2 ***Hello NHibernate!*** 24

- 2.1 “Hello World” with NHibernate 25
 - Installing NHibernate 25* ▪ *Create a new Visual Studio project 25*
 - Creating the Employee class 26* ▪ *Setting up the database 27*
 - Creating an Employee and saving to the database 27* ▪ *Loading an Employee from the database 29* ▪ *Creating a mapping file 29*
 - Configuring your application 31* ▪ *Updating an Employee 32* ▪ *Running the program 33*
- 2.2 Understanding the architecture 33
 - The core interfaces 35* ▪ *Callback interfaces 36* ▪ *Types 37*
 - Extension interfaces 37*
- 2.3 Basic configuration 38
 - Creating a SessionFactory 38* ▪ *Configuring the ADO.NET database access 41*
- 2.4 Advanced configuration settings 44
 - Using the application configuration file 44* ▪ *Logging 47*
- 2.5 Summary 48

PART 2 **NHIBERNATE DEEP DIVE** 49

3 ***Writing and mapping classes*** 51

- 3.1 The CaveatEmptor application 52
 - Analyzing the business domain 52* ▪ *The CaveatEmptor domain model 53*
- 3.2 Implementing the domain model 55
 - Addressing leakage of concerns 55* ▪ *Transparent and automated persistence 55* ▪ *Writing POCOs 56*
 - Implementing POCO associations 58* ▪ *Adding logic to properties 61*
- 3.3 Defining the mapping metadata 63
 - Mapping using XML 63* ▪ *Attribute-oriented programming 65*

- 3.4 Basic property and class mappings 66
 - Property mapping overview* 66
 - *Using derived properties* 68
 - Property access strategies* 68
 - *Taking advantage of the reflection optimizer* 70
 - *Controlling insertion and updates* 71
 - *Using quoted SQL identifiers* 72
 - *Naming conventions* 72
 - *SQL schemas* 73
 - *Declaring class names* 74
 - *Manipulating metadata at runtime* 75
- 3.5 Understanding object identity 76
 - Identity versus equality* 76
 - *Database identity with NHibernate* 77
 - *Choosing primary keys* 79
- 3.6 Fine-grained object models 81
 - Entity and value types* 81
 - *Using components* 82
- 3.7 Introducing associations 86
 - Unidirectional associations* 86
 - *Multiplicity* 86
 - *The simplest possible association* 87
 - *Making the association bidirectional* 88
 - A parent/child relationship* 90
- 3.8 Mapping class inheritance 91
 - Table per concrete class* 92
 - *Table per class hierarchy* 93
 - *Table per subclass* 95
 - *Choosing a strategy* 98
- 3.9 Summary 99

4 Working with persistent objects 100

- 4.1 The persistence lifecycle 101
 - Transient objects* 102
 - *Persistent objects* 102
 - *Detached objects* 103
 - *The scope of object identity* 104
 - *Outside the identity scope* 105
 - *Implementing Equals() and GetHashCode()* 106
- 4.2 The persistence manager 110
 - Making an object persistent* 110
 - *Updating the persistent state of a detached instance* 111
 - *Retrieving a persistent object* 112
 - Updating a persistent object transparently* 113
 - *Making an object transient* 113
- 4.3 Using transitive persistence in NHibernate 114
 - Persistence by reachability* 115
 - *Cascading persistence with NHibernate* 116
 - *Managing auction categories* 117
 - Distinguishing between transient and detached instances* 120

- 4.4 Retrieving objects 121
 - Retrieving objects by identifier* 122
 - *Introducing Hibernate Query Language* 123
 - *Query by Criteria* 124
 - *Query by Example* 124
 - *Fetching strategies* 125
 - *Selecting a fetching strategy in mappings* 127
 - *Tuning object retrieval* 132
- 4.5 Summary 133

5 **Transactions, concurrency, and caching** 134

- 5.1 Understanding database transactions 135
 - ADO.NET and Enterprise Services/COM+ transactions* 136
 - The NHibernate ITransaction API* 137
 - *Flushing the session* 138
 - *Understanding connection-release modes* 139
 - Understanding isolation levels* 140
 - *Choosing an isolation level* 141
 - *Setting an isolation level* 143
 - *Using pessimistic locking* 143
- 5.2 Working with conversations 146
 - An example scenario* 146
 - *Using managed versioning* 147
 - Optimistic and pessimistic locking compared* 149
 - *Granularity of a session* 150
 - *Other ways to implement optimistic locking* 151
- 5.3 Caching theory and practice 152
 - Caching strategies and scopes* 153
 - *The NHibernate cache architecture* 155
 - *Caching in practice* 159
- 5.4 Summary 164

6 **Advanced mapping concepts** 166

- 6.1 Understanding the NHibernate type system 167
 - Associations and value types* 167
 - *Bridging from objects to database* 168
 - *Mapping types* 168
 - *Built-in mapping types* 169
 - *Using mapping types* 172
- 6.2 Mapping collections of value types 181
 - Storing value types in sets, bags, lists, and maps* 181
 - *Collections of components* 186
- 6.3 Mapping entity associations 189
 - One-to-one associations* 189
 - *Many-to-many associations* 193
- 6.4 Mapping polymorphic associations 200
 - Polymorphic many-to-one associations* 201
 - *Polymorphic collections* 203
 - *Polymorphic associations and table-per-concrete-class* 204
- 6.5 Summary 205

- 7 Retrieving objects efficiently 207**
- 7.1 Executing queries 208
 - The query interfaces 208* ▪ *Binding parameters 211* ▪ *Using named queries 214* ▪ *Using query substitutions 215*
 - 7.2 Basic queries for objects 215
 - The simplest query 215* ▪ *Using aliases 216* ▪ *Polymorphic queries 217* ▪ *Restriction 217* ▪ *Comparison operators 218* ▪ *String matching 220* ▪ *Logical operators 221* ▪ *Ordering query results 221*
 - 7.3 Joining associations 222
 - NHibernate join options 223* ▪ *Fetching associations 224* ▪ *Using aliases with joins 226* ▪ *Using implicit joins 228* ▪ *Theta-style joins 229* ▪ *Comparing identifiers 230*
 - 7.4 Writing report queries 231
 - Projection 232* ▪ *Using aggregation 234* ▪ *Grouping 234* ▪ *Restricting groups with having 236* ▪ *Improving performance with report queries 236* ▪ *Obtaining DataSets 237*
 - 7.5 Advanced query techniques 238
 - Dynamic queries 238* ▪ *Collection filters 240* ▪ *Subqueries 242*
 - 7.6 Native SQL 243
 - Using the ISQLQuery API 244* ▪ *Named SQL queries 246* ▪ *Customizing create, retrieve, update, and delete commands 248*
 - 7.7 Optimizing object retrieval 249
 - Solving the n+1 selects problem 249* ▪ *Using Enumerable() queries 252* ▪ *Caching queries 253* ▪ *Using profilers and NHibernate Query Analyzer 255*
 - 7.8 Summary 255

PART 3 NHIBERNATE IN THE REAL WORLD 257

8 Developing NHibernate applications 259

- 8.1 Inside the layers of an NHibernate application 260
 - Using patterns and methodologies 261* ▪ *Building and testing the layers 263* ▪ *The domain model 263* ▪ *The business layer 266* ▪ *The persistence layer 268* ▪ *The presentation layer 269*
- 8.2 Solving issues related to .NET features 270
 - Working with web applications 271* ▪ *.NET remoting 271*

- 8.3 Achieving goals and solving problems 272
 - Design goals applied to an NHibernate application* 272
 - Identifying and solving problems* 274 ▪ *Use the right tool for the right job* 276
- 8.4 Integrating services: the case of audit logging 277
 - Doing it the hard way* 278 ▪ *Doing it the NHibernate way* 278 ▪ *Other ways of integrating services* 283
- 8.5 Summary 284

9 Writing real-world domain models 286

- 9.1 Development processes and tools 287
 - Top down: generating the mapping and the database from entities* 288 ▪ *Middle out: generating entities from the mapping* 292 ▪ *Bottom up: generating the mapping and the entities from the database* 293 ▪ *Automatic database schema maintenance* 294
- 9.2 Legacy schemas 296
 - Mapping a table with a natural key* 297 ▪ *Mapping a table with a composite key* 298 ▪ *Using a custom type to map legacy columns* 302 ▪ *Working with triggers* 303
- 9.3 Understanding persistence ignorance 305
 - Abstracting persistence-related code* 305 ▪ *Applying the Observer pattern to an entity* 307
- 9.4 Implementing the business logic 309
 - Business logic in the business layer* 310 ▪ *Business logic in the domain model* 310 ▪ *Rules that aren't business rules* 312
- 9.5 Data-binding entities 312
 - Implementing manual data binding* 313 ▪ *Using data-bound controls* 314 ▪ *Data binding using NHibernate* 315 ▪ *Data binding using ObjectViews* 315
- 9.6 Filling a DataSet with entities' data 316
 - Converting an entity to a DataSet* 316 ▪ *Using NHibernate to assist with conversion* 317
- 9.7 Summary 317

10 Architectural patterns for persistence 319

- 10.1 Designing the persistence layer 320
 - Implementing a simple persistence layer* 321 ▪ *Implementing a generic persistence layer* 326

- 10.2 Implementing conversations 335
 - Approving a new auction* 336
 - *Loading objects on each request* 337
 - *Using detached persistent objects* 338
 - *Using the session-per-conversation pattern* 340
 - *Choosing an approach to conversations* 344
- 10.3 Using NHibernate in an Enterprise Services application 345
 - Rethinking DTOs* 345
 - *Enabling distributed transactions for NHibernateHelper* 346
- 10.4 Summary 348
- appendix A SQL fundamentals* 349
- appendix B Going forward* 352
- index* 355