

Hibernate (framework)


From Wikipedia, the free encyclopedia

Hibernate ORM (Hibernate in short) is an object-relational mapping tool for the Java programming language. It provides a framework for mapping an object-oriented domain model to a relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct, persistent database accesses with high-level object handling functions.

Hibernate is free software that is distributed under the GNU Lesser General Public License 2.1.

Hibernate's primary feature is mapping from Java classes to database tables, and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from the manual handling and object conversion of the result set.

Hibernate ORM



Developer(s)	Red Hat
Stable release	v5.2.5 / November 24, 2016
Repository	github.com/hibernate/hibernate-orm (https://github.com/hibernate/hibernate-orm)
Development status	Active
Written in	Java
Operating system	Cross-platform (JVM)
Platform	Java Virtual Machine
Type	Object-relational mapping
License	GNU Lesser General Public License
Website	hibernate.org (http://hibernate.org)

Contents

- 1 Mapping
- 2 Hibernate Query Language (HQL)
- 3 Persistence
- 4 Integration
- 5 Entities and components
- 6 History
- 7 Application programming interface
 - 7.1 org.hibernate.SessionFactory interface
 - 7.2 org.hibernate.Session interface
- 8 Software components
- 9 See also
- 10 References
- 11 Bibliography
- 12 External links
 - 12.1 Further reading

Mapping

The mapping of Java classes to database tables is implemented by the configuration of an XML file or by using Java Annotations. When using an XML file, Hibernate can generate skeleton source code for the persistence classes. This is auxiliary when annotations are used. Hibernate can use the XML file or the Java annotations to maintain the database schema.

There are provided facilities to arrange one-to-many and many-to-many relationships between classes. In addition to managing associations between objects, Hibernate can also manage reflexive associations wherein an object has a one-to-many relationship with other instances of the class type.

Hibernate supports the mapping of custom value types. This makes the following scenarios possible:

- Overriding the default SQL type when mapping a column to a property.

- Mapping Java Enums to columns as though they were regular properties.
- Mapping a single property to multiple columns.

Definition: Objects in an object-oriented application follow OOP principles, while objects in the back-end follow database normalization principles, resulting in different representation requirements. This problem is called "object-relational impedance mismatch". Mapping is a way of resolving the object-relational impedance mismatch problem.

Mapping informs the ORM tool of what Java class object to store in which database table.

Hibernate Query Language (HQL)

Hibernate provides an SQL inspired language called Hibernate Query Language (HQL) that allows SQL-like queries to be written against Hibernate's data objects. *Criteria Queries* are provided as an object-oriented alternative to HQL. Criteria Query is used to modify the objects and provide the restriction for the objects. HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates database independent queries so that there is no need to write database-specific queries. Without this capability, changing the database would require individual SQL queries to be changed as well, leading to maintenance issues.

Persistence

Hibernate provides transparent persistence for Plain Old Java Objects (POJOs). The only strict requirement for a persistent class is a no-argument constructor, not necessarily *public*. Proper behavior in some applications also requires special attention to the *equals()* and *hashCode()* methods.^[1]

Collections of data objects are typically stored in Java collection classes such as implementations of the Set and List interfaces. Java generics, introduced in Java 5, are supported. Hibernate can be configured to lazy load associated collections. Lazy loading is the default as of Hibernate 3.

Related objects can be configured to *cascade* operations from one to the other. For example, a parent Album object can be configured to cascade its save and/or delete operation to its child Track objects.

Integration

Hibernate can be used both in standalone Java applications and in Java EE applications using servlets, EJB session beans, and JBI service components. It can also be included as a feature in other programming languages. For example, Adobe integrated Hibernate into version 9 of ColdFusion (which runs on J2EE app servers) with an abstraction layer of new functions and syntax added into CFML.

Entities and components

In Hibernate jargon, an *entity* is a stand-alone object in Hibernate's persistent mechanism which can be manipulated independently of other objects. In contrast, a *component* is subordinate to an entity and can be manipulated only with respect to that entity. For example, an Album object may represent an entity; but the Tracks object associated with the Album objects would represent a *component* of the Album entity, if it is assumed that Tracks can only be saved or retrieved from the database through the Album object. Unlike J2EE, Hibernate can switch databases.

History

Hibernate was started in 2001 by Gavin King with colleagues from Cirrus Technologies as an alternative to using EJB2-style entity beans. The original goal was to offer better persistence capabilities than those offered by EJB2; by simplifying the complexities and supplementing certain missing features.

In early 2003, the Hibernate development team began Hibernate2 releases, which offered many significant improvements over the first release.

JBoss, Inc. (now part of Red Hat) later hired the lead Hibernate developers in order to further its development.

In 2005, Hibernate version 3.0 was released. Key features included a new Interceptor/Callback architecture, user defined filters, and JDK 5.0 Annotations (Java's metadata feature). As of 2010, Hibernate 3 (version 3.5.0 and up) was a certified implementation of the Java Persistence API 2.0 specification via a wrapper for the Core module which provides conformity with the JSR 317 (<http://jcp.org/en/jsr/detail?id=317>) standard.^[2]

In Dec 2011, Hibernate Core 4.0.0 Final was released. This includes new features such as multi-tenancy support, introduction of ServiceRegistry (a major change in how Hibernate builds and manages "services"), better session opening from SessionFactory, improved integration via *org.hibernate.integrator.spi.Integrator* and auto discovery, internationalization support, message codes in logging, and a more distinction between the API, SPI or implementation classes.^[3]

In Dec 2012, Hibernate ORM 4.1.9 Final was released.^[4]

In Mar 2013, Hibernate ORM 4.2 Final was released.^[5]

In Dec 2013, Hibernate ORM 4.3.0 Final was released.^[6] It features Java Persistence API 2.1.^[7]

In Sep 2015, Hibernate ORM 5.0.2 Final was released. It has improved bootstrapping, hibernate-java8, hibernate-spatial, Karaf support.

Application programming interface

The Hibernate API is provided in the Java package `org.hibernate`.^[8]

`org.hibernate.SessionFactory` (<http://docs.jboss.org/hibernate/stable/core/javadocs/org/hibernate/SessionFactory.html>) interface

`org.hibernate.Session` interface

The `org.hibernate.Session` interface^[9] represents a Hibernate session, i.e., the main point of the manipulation performed on the database entities. The latter activities include (among the other things) managing the persistence state (transient, persisted, detached) of the objects, fetching the persisted ones from the database and the management of the transaction demarcation.

A session is intended to last as long as the logical transaction on the database. Due to the latter feature, Session implementations are not expected to be thread safe nor to be used by multiple clients.

Software components

The Hibernate software includes the following components:^[10]

- Hibernate ORM (known as Hibernate Core before release 4.1^[11]) – the base software for an object-relational mapping solution for Java environments^[12]

- **Hibernate Annotations** (merged into **Hibernate Core/ORM** since version 3.6^[13]) – metadata that governs the transformation of data between the object-oriented model and the relational database model according to the JSR 317 Java Persistence API (JPA 2)^[14]
- **Hibernate EntityManager** – together with **Hibernate Annotations**, a wrapper that implements a JSR 317 Java Persistence API (JPA 2) persistence solution on top of **Hibernate Core**^[15]
- **Hibernate Envers** – auditing and versioning of persistent classes^[16]
- **Hibernate OGM (Object/Grid Mapper)** – an extension to store data in a NoSQL store^[17]
- **Hibernate Shards** – horizontal partitioning for multiple relational databases^[18]
 - While **Hibernate Shards** is not compatible with 4.x releases of **Hibernate Core**, some of the **Shards** capability was integrated into **Core** in the 4.0 release
- **Hibernate Search** – integrates the full text library functionality from **Apache Lucene** in the **Hibernate** and **JPA** model^[19]
- **Hibernate Tools** – a set of tools implemented as a suite of **Eclipse** plugins and **Ant** tasks included in **JBoss Developer Studio**^[20]
- **Hibernate Validator** – the reference implementation of JSR 303 Bean Validation^[21]
- **Hibernate Metamodel Generator** – an annotation processor that creates JSR 317 Java Persistence API (JPA 2) static metamodel classes using the JSR 269 Pluggable Annotation Processing API^[22]
- **NHibernate** – an object-relational mapping solution for the .NET Framework^[23]

See also

- List of JBoss software
- List of object-relational mapping software
- **NHibernate**
- **Serialization**
- **Service Data Object**
- **MySQL**
- **Apache Cassandra**
- **Spring Framework**

References

1. "Equals and hashCode". JBoss Community.
2. "Hibernate 3.5.0-Final release". In Relation To...
3. <http://www.hibernate.org/orm/downloads/>
4. <http://in.relation.to/23792.lace>
5. <http://grepcode.com/snapshot/repository.jboss.org/nexus/content/repositories/releases/org.hibernate/hibernate-core/4.2.0.Final>
6. <http://grepcode.com/snapshot/repository.jboss.org/nexus/content/repositories/releases/org.hibernate/hibernate-core/4.3.0.Final>
7. <http://hibernate.org/orm/downloads/>
8. <http://docs.jboss.org/hibernate/stable/core/javadocs/index.html?overview-summary.html>
9. <http://docs.jboss.org/hibernate/stable/core/javadocs/org/hibernate/Session.html>
10. "Hibernate: Relational Persistence for Java and .NET". JBoss Community.
11. "Hibernate ORM 4.1.0 Release". JBoss Community.
12. "HIBERNATE - Relational Persistence for Idiomatic Java". JBoss Community.
13. "No more hibernate-annotations module". JBoss Community.
14. "Hibernate Annotations". JBoss Community.
15. "Hibernate EntityManager". JBoss Community.
16. "Hibernate Envers – Easy Entity Auditing". JBoss Community.
17. "Hibernate OGM". JBoss Community.
18. "Hibernate Shards". JBoss Community.
19. "Hibernate Search". JBoss Community.
20. "Hibernate Tools for Eclipse and Ant". JBoss Community.
21. "Hibernate Validator". JBoss Community.
22. "Hibernate Metamodel Generator". JBoss Community.

23. "NHibernate". NHibernate Forge.

Bibliography

- Linwood, Jeff; Minter, Dave (May 28, 2010), *Beginning Hibernate* (Second ed.), Apress, p. 400, ISBN 1-4302-2850-4
- Bernard, Emmanuel; Griffin, John (December 30, 2008), *Hibernate Search in Action* (First ed.), Manning Publications, p. 488, ISBN 1-933988-64-9
- Elliott, James; O'Brien, Tim (April 22, 2008), *Harnessing Hibernate* (First ed.), O'Reilly Media, p. 380, ISBN 0-596-51772-6
- King, Gavin; Christian, Bauer (November 24, 2006), *Java Persistence with Hibernate* (Second ed.), Manning Publications, p. 880, ISBN 1-932394-88-5
- Linwood, Jeff; Minter, Dave (August 25, 2006), *Beginning Hibernate: From Novice to Professional* (Third ed.), Apress, p. 360, ISBN 1-59059-693-5
- Minter, Dave; Linwood, Jeff (June 27, 2005), *Pro Hibernate 3* (First ed.), Apress, p. 242, ISBN 1-59059-511-4
- Iverson, Will (December 2, 2004), *Hibernate: A J2EE Developer's Guide* (First ed.), Addison Wesley, p. 384, ISBN 0-321-26819-9
- Pugh, Eric; Gradecki, Joseph D. (October 8, 2004), *Professional Hibernate (Programmer to Programmer)* (First ed.), Wrox, p. 456, ISBN 0-7645-7677-1
- King, Gavin; Christian, Bauer (August 1, 2004), *Hibernate In Action* (Second ed.), Manning Publications, p. 400, ISBN 1-932394-15-X
- James, Elliott (May 10, 2004), *Hibernate: A Developer's Notebook* (First ed.), O'Reilly Media, p. 190, ISBN 0-596-00696-9

External links

- Official website (<http://www.hibernate.org>)
- Interview with Gavin King, founder of Hibernate (<http://www.javafree.org/artigo/871462/Interview-with-Gavin-King-founder-of-Hibernate.html>)
- Hibernate Reference Documentation (<http://docs.jboss.org/hibernate/stable/core/manual/en-US/html/>)
- scale4j is highly scalable domain oriented data-distributed platform for java built on top of hibernate (<http://code.google.com/p/scale4j/>)
- TorpedoQuery - Type safe Hibernate query builder (<http://torpedoquery.org>)
- Best Recommended books for Hibernate framework (<http://java360.co.in/best-recommended-books-for-hibernate-framework/>)
- How Hibernate interpret Order of JOIN on a Table? (<http://www.javaquery.com/2014/10/how-hibernate-interpret-order-of-join.html>)
- When Hibernate uses INNER JOIN? (<http://www.javaquery.com/2014/09/when-hibernate-uses-inner-join.html>)
- How To Call Stored Procedure In Hibernate (<http://www.javaquery.com/2014/01/how-to-call-stored-procedure-in.html>)
- How to create Hibernate query log? (<http://www.javaquery.com/2013/12/how-to-create-hibernate-query-log.html>)

Further reading

- Java Hibernate Tutorials (<http://memorynotfound.com/category/database/hibernate/>)
- Hibernate Tutorials (<http://howtodoinjava.com/hibernate-tutorials/>)
- Hibernate Tutorial (<http://www.mkymong.com/tutorials/hibernate-tutorials/>)
- Hibernate Annotations Tutorial (<http://www.javatips.net/blog/2011/12/hibernate-annotations-tutorial>)
- Java Persistence (JPA 2.0) Tutorial With Hibernate (<http://www.javatips.net/blog/2012/12/java-persistence-jpa-2-0-tutorial-with-hibernate>)
- Hibernate Tutorial by Java Code Geeks (<http://www.javacodegeeks.com/2015/03/hibernate-tutorial.html>)
- Hibernate Training (<http://wisensolutions.com/IT-Courses/JPA-Hibernate-Training>)
- Hibernate Tutorials (<http://www.devglan.com/tutorial/topics/hibernate>)

Retrieved from "[https://en.wikipedia.org/w/index.php?title=Hibernate_\(framework\)&oldid=769473266](https://en.wikipedia.org/w/index.php?title=Hibernate_(framework)&oldid=769473266)"

Categories: [Object-relational mapping](#) | [Java platform](#) | [Java enterprise platform](#) | [Red Hat software](#) | [Cross-platform software](#) | [Persistence frameworks](#)

- This page was last modified on 9 March 2017, at 19:41.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.