Java Persistence API

From Wikipedia, the free encyclopedia

The **Java Persistence API (JPA)** is a Java application programming interface specification that describes the management of relational data in applications using Java Platform, Standard Edition and Java Platform, Enterprise Edition.

Persistence in this context covers three areas:

- the API itself, defined in the `javax.persistence` package
- the Java Persistence Query Language (JPQL)
- object/relational metadata

The reference implementation for JPA is EclipseLink.

Contents

History

The final release date of the JPA 1.0 specification was 11 May 2006 as part of Java Community Process JSR 220. The JPA 2.0 specification was released 10 December 2009. The JPA 2.1 specification was released 22 April 2013.

Entities

A persistence entity is a lightweight Java class whose state is typically persisted to a table in a relational database. Instances of such an entity correspond to individual rows in the table. Entities typically have relationships with other entities, and these relationships are expressed through object/relational metadata. Object/relational metadata can be specified directly in the entity class file by using annotations, or in a separate XML descriptor file distributed with the application.

The Java Persistence Query Language

The Java Persistence Query Language (JPQL) makes queries against entities stored in a relational database. Queries resemble SQL queries in syntax, but operate against entity objects rather than directly with database tables.

Motivation

Prior to the introduction of EJB 3.0 specification, many enterprise Java developers used lightweight persistent objects, provided by either persistence frameworks (for example Hibernate) or data access objects instead of entity beans. This is because entity beans, in previous EJB specifications, called for too much complicated code and heavy resource footprint, and they could be used only in Java EE application servers because of interconnections and dependencies in the source code between beans and DAO objects or persistence framework. Thus, many of the features originally presented in third-party persistence frameworks were incorporated into the Java Persistence API, and, as of 2006, projects like Hibernate (version 3.2) and TopLink Essentials have become themselves implementations of the Java Persistence API specification.

Related technologies

Enterprise JavaBeans

The EJB 3.0 specification (itself part of the Java EE 5 platform) included a definition of the Java Persistence API. However, end-users do not need an EJB container or a Java EE application server in order to run applications that use this persistence API.[1] Future versions of the Java Persistence API will be defined in a separate JSR and specification rather than in the EJB JSR/specification.

The Java Persistence API replaces the persistence solution of EJB 2.0 CMP (Container Managed Persistence).

Java Data Objects API

The Java Persistence API was developed in part to unify the *Java Data Objects API*, and the *EJB 2.0 Container Managed Persistence (CMP) API*. As of 2009 most products supporting each of those APIs support the Java Persistence API.

The Java Persistence API specifies persistence only for relational database management systems. That is, JPA focuses on object-relational mapping (ORM) (note that there are JPA providers who support other database models besides relational database, but this is outside the scope of what JPA was designed for). Refer to JPA 2 spec section 1 introduction for clarification of the role of JPA, which states very clearly *"The technical objective of this work is to provide an object/relational mapping facility for the Java application developer using a Java domain model to manage a relational database."*

The Java Data Objects specification supports ORM, as well as persistence to other types of database models, for example flat file databases and NoSQL databases, including document databases, graph databases, as well as literally any other conceivable datastore.

Service Data Object API

The designers[2] of the Java Persistence API aimed to provide for relational persistence, with many of the key areas taken from object-relational mapping tools such as Hibernate and TopLink. Java Persistence API improved on and replaced EJB 2.0, evidenced by its inclusion in EJB 3.0. The Service Data Objects (SDO) API (JSR 235) has a very different objective to the Java Persistence API and is considered [3][4] complementary. The SDO API is designed for service-oriented architectures, multiple data formats rather than only relational data, and multiple programming languages. The Java Community Process manages the Java version of the SDO API; the C++ version of the SDO API is managed via OASIS.

Hibernate

Hibernate provides an open source object-relational mapping framework for Java. Versions 3.2 and later provide an implementation for the Java Persistence API.[5] Gavin King founded the Hibernate project.[6] He represented JBoss on JSR 220,[7] the JCP expert group charged with developing JPA. This led to ongoing controversy and speculation surrounding the relationship between JPA and Hibernate. Sun Microsystems has stated[8] that ideas came from several frameworks, including Hibernate and Java Data Objects

JPA 2.0

Development of a new version of JPA 2.0 was started in July 2007 in the Java Community Process as JSR 317. JPA 2.0 was approved as final on 10 December 2009. The focus of JPA 2.0 was to address features that were present in some of the popular ORM vendors, but could not gain consensus approval for JPA 1.0.

Main features included were:

- Expanded object-relational mapping functionality
  - support for collections of embedded objects, linked in the ORM with a many-to-one relationship
  - ordered lists
  - combinations of access types
- A criteria query API
- standardization of SQL Hints
- standardization of additional metadata to support DDL generation
- support for validation
- Shared object cache support.

Vendors supporting JPA 2.0:

- Batoo JPA
- DataNucleus (formerly JPOX)
- EclipseLink (formerly Oracle TopLink)
- IBM, for WebSphere Application Server[9]
- JBoss with Hibernate
- Kundera (https://github.com/impetus-opensource/Kundera)
- ObjectDB
- OpenJPA
- OrientDB from Orient Technologies
- Versant Corporation JPA (not relational, object database)[10]

JPA 2.1

Development of a new version of JPA 2.1 was started in July 2011 as JSR 338. JPA 2.1 was approved as final on 22 May 2013.

Main features included were:

- Converters - allowing custom code conversions between database and object types.
- Criteria Update/Delete - allows bulk updates and deletes through the Criteria API.
- Entity Graphs - allow partial or specified fetching or merging of objects.
- JPQL/Criteria enhancements - arithmetic sub-queries, generic database functions, join ON clause, TREAT option.
- Schema Generation
- Stored Procedures - allows queries to be defined for database stored procedures.

Vendors supporting JPA 2.1

- DataNucleus
- EclipseLink
- Hibernate

JPA Future Work

Future JPA specification information is available here:

- JPA Specification Mailing Lists (https://java.net/projects/jpa-spec/lists)
- JPA Specification JIRA (https://java.net/jira/browse/JPA_SPEC)

In November 2015, Linda DeMichiel announced on the javaee-spec users mailing that Lukas Jungmann took over as the specification lead. Linda's announcement also stated, "[w]e plan to do an MR for JPA 2.2 in the Java EE 8 time frame".[11][12][13]

Tools

- Eclipse JPA (Dali)
- NetBeans JPA Modeler

See also

- .NET Persistence API (NPA)
- JDBC
- MyBatis
- OpenXava
- pureQuery
- SAP NetWeaver Application Server
- XQJ

References

1. Hibernate EntityManager: Java SE environments (http://docs.jboss.org/hibernate/entitymanager/3.5/reference/en/html_single/#architecture-javase)
   Hibernate EntityManager: Obtaining an EntityManager in a Java SE environment (http://docs.jboss.org/hibernate/entitymanager/3.5/reference/en/html_single/#d0e980)
2. "JSR 220 Members".
3. Barreto, Charlton. "SDO and JPA". *Digital Walkabout*. Retrieved 5 May 2011.
4. Edwards, Mike. "SDO and Java Persistence Architecture (JPA)". *Open SOA*. osoa.org. Retrieved 5 May 2011.
5. "hibernate.org - Java Persistence with Hibernate". JBoss. Retrieved 2008-11-17. "Hibernate implements the Java Persistence object/relational javaAPI and persistence management interfaces"
6. *Java Persistence with Hibernate*. Manning Publications. ISBN 9781617290459. Retrieved 8 December 2013. "Gavin King is the founder of the Hibernate project"
7. "JBoss.com - Industry Leadership". JBoss. Retrieved 2008-11-17. "JSR 220, EJB 3.0 Spec Committee, Gavin King, Bill Burke, Marc Fleury"
8. "Java Persistence API FAQ". Sun Microsystems. Archived from the original on 2008-08-22. Retrieved 2010-07-01. "The Java Persistence API draws upon the best ideas from persistence technologies such as Hibernate, TopLink, and JDO"
9. "IBM WebSphere Application Server V7 Feature Pack for OSGi Applications and Java Persistence API". *Download web site*. IBM. 27 April 2010. Retrieved 8 December 2013.
10. "Versant JPA 2-Step Download". *Download web site*. Actian. Retrieved 8 December 2013.
11. "Java EE Platform Specification: users@javaee-spec.java.net: Archive — Project Kenai". *java.net*. Retrieved 2016-11-08.
12. Java (2015-10-27), *JavaOne LIVE - Tuesday, Mission*, retrieved 2016-11-08
13. Jungmann, Lukas (10 October 2015). "What's New in the Java Persistence API (JSR 338) [CON7631]". *JavaOne 2015 Session Catalog*. Retrieved 2016-11-08 – via rainfocus.com.

External links

General info

Wikibooks has a book on the topic of: ***Java Persistence***

- Documentation for the final version of the EJB3 spec (called JSR220) (http://jcp.org/aboutJava/communityprocess/final/jsr220/index.html)
- GlassFish's Persistence page (http://glassfish.dev.java.net/javaee5/persistence/)
- JCP Persistence page (http://jcp.org/aboutJava/communityprocess/final/jsr317/index.html)

Tutorials

- Java EE 6 Persistence API Javadoc (http://docs.oracle.com/javaee/6/api/javax/persistence/package-summary.html)
- Java EE 6 Persistence API tutorial (http://docs.oracle.com/javaee/6/tutorial/doc/bnbpy.html)
- Java EE 7 Persistence API Javadoc (http://docs.oracle.com/javaee/7/api/javax/persistence/package-summary.html)
- Java EE 7 Persistence API tutorial (http://docs.oracle.com/javaee/7/tutorial/partpersist.htm)
- JPA Tutorial by Prasad Kharkar (http://www.thejavageek.com/jpa-tutorials/)
- JPA Tutorial from Java Code Geeks (http://www.javacodegeeks.com/2015/02/jpa-tutorial.html)
- JPA Tutorial using OpenJPA as implementation (http://www.coderpanda.com/jpa-tutorial/)
- Persistence in the Java EE 5 tutorial (http://java.sun.com/javaee/5/docs/tutorial/doc/?wp406141&JavaEETutorialPartFour.html#wp996871)

Categories: Java APIs | Java enterprise platform | Java specification requests | Object-relational mapping | Persistence

---