**Introduction to Hibernate Framework**

## **Hibernate Framework**

Hibernate is a Java framework that simplifies the development of Java application to interact with the database. It is an open source, lightweight, ORM (Object Relational Mapping) tool. Hibernate implements the specifications of JPA (Java Persistence API) for data persistence.

Hibernate framework uses many objects such as session factory, session, transaction etc. alongwith existing Java API such as JDBC (Java Database Connectivity), JTA (Java Transaction API) and JNDI (Java Naming Directory Interface)

@Entity

Annotate all your entity beans with @Entity.

@Entity

public class Company implements Serializable {

...

}

### @Table

Specify the database table this Entity maps to using the name attribute of @Table annotation. In the example below, the data will be stored in 'company' table in the database.

@Entity

@Table(name = "company")

public class Company implements Serializable {

...

}

### @Column

Specify the column mapping using @Column annotation.

@Entity

@Table(name = "company")

public class Company implements Serializable {

@Column(name = "name")

private String name;

...

}

### @Id

Annotate the id column using @Id.

@Entity

@Table(name = "company")

public class Company implements Serializable {

@Id

@Column(name = "id")

private int id;

...

}

### @GeneratedValue

Let database generate (auto-increment) the id column.

@Entity

@Table(name = "company")

public class Company implements Serializable {

@Id

@Column(name = "id")

@GeneratedValue

private int id;

...

}

### @Version

Control versioning or concurrency using @Version annotation.

@Entity

@Table(name = "company")

public class Company implements Serializable {

@Version

@Column(name = "version")

private Date version;

...

}

### @OrderBy

Sort your data using @OrderBy annotation. In example below, it will sort all contacts in a company by their firstname in ascending order.

@OrderBy("firstName asc")

private Set contacts;

### @Transient

Annotate your transient properties with @Transient.

### @Lob

Annotate large objects with @Lob.

## **ORM Tool**

An ORM tool simplifies the data creation, data manipulation and data access. It is a programming technique that maps the object to the data stored in the database.



The ORM tool internally uses the JDBC API to interact with the database.

## **What is JPA?**

Java Persistence API (JPA) is a Java specification that provides certain functionality and standard to ORM tools. The **javax.persistence** package contains the JPA classes and interfaces.

## **Advantages of Hibernate Framework**

Following are the advantages of hibernate framework:

### **1) Open Source and Lightweight**

Hibernate framework is open source under the LGPL license and lightweight.

### **2) Fast Performance**

The performance of hibernate framework is fast because cache is internally used in hibernate framework. There are two types of cache in hibernate framework first level cache and second level cache. First level cache is enabled by default.

### **3) Database Independent Query**

HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates the database independent queries. So you don't need to write database specific queries. Before Hibernate, if database is changed for the project, we need to change the SQL query as well that leads to the maintenance problem.

### **4) Automatic Table Creation**

Hibernate framework provides the facility to create the tables of the database automatically. So there is no need to create tables in the database manually.

### **5) Simplifies Complex Join**

Fetching data from multiple tables is easy in hibernate framework.

### **6) Provides Query Statistics and Database Status**

Hibernate supports Query cache and provide statistics about query and database status.

# Hibernate Example using Annotation in Eclipse

The hibernate application can be created with annotation. There are many annotations that can be used to create hibernate application such as @Entity, @Id, @Table etc.

Hibernate Annotations are based on the JPA 2 specification and supports all the features.

All the JPA annotations are defined in the **javax.persistence** package. Hibernate EntityManager implements the interfaces and life cycle defined by the JPA specification.

The core advantage of using hibernate annotation is that you don't need to create mapping (hbm) file. Here, hibernate annotations are used to provide the meta data.

# Web Application with Hibernate (using XML)

Here, we are going to create a web application with hibernate. For creating the web application, we are using JSP for presentation logic, Bean class for representing data and DAO class for database codes.

As we create the simple application in hibernate, we don't need to perform any extra operations in hibernate for creating web application. In such case, we are getting the value from the user using the JSP file.

Hibernate Architecture

1. [Hibernate Architecture](https://www.javatpoint.com/hibernate-architecture)
2. [Elements of Hibernate Architecture](https://www.javatpoint.com/hibernate-architecture#elements)
   1. [SessionFactory](https://www.javatpoint.com/hibernate-architecture#e1)
   2. [Session](https://www.javatpoint.com/hibernate-architecture#e2)
   3. [Transaction](https://www.javatpoint.com/hibernate-architecture#e3)
   4. [ConnectionProvider](https://www.javatpoint.com/hibernate-architecture#e4)
   5. [TransactionFactory](https://www.javatpoint.com/hibernate-architecture#e5)

The Hibernate architecture includes many objects such as persistent object, session factory, transaction factory, connection factory, session, transaction etc.

The Hibernate architecture is categorized in four layers.

* Java application layer
* Hibernate framework layer
* Backhand api layer
* Database layer

Let's see the diagram of hibernate architecture:

This is the high level architecture of Hibernate with mapping file and configuration file