<https://github.com/sanjaydegaonkar/HibernateRepo>

JPA is an API, it can be used to perform persistence operations in Enterprise Application.

JPA is an abstraction provided by SUN microsystems along with EJBs3.0 version and it was implemented almost all the application servers. just like JDBC is abstraction, whose implemenations are provided by all database vendors.

JPA is a specification, it can be used to perform persistence operations in enterprise applications by following ORM rules and regulations.

Popular JPA implementations are

1.Hibernate--->JBOSS

2.EclipseLink ------>Eclipse Foundation------>GlassFish

3.Open JPA --------->Apache Software Foundations

If we want to use JPA in our enterprise applications then we have to use either of the above implementations.

Basic JPA Application

---------------------

1.Create a Java Project with JPA implementation library[Hibernate]

2.Create Entity class

3.Create Entity Mapping file.

4.Create JPA configuration file.(Persistence.xml)

5.Create Test Application

6.Run Test Application

1.Create a Java Project with JPA implementation library[Hibernate]

------------------------------------------------------------------

Create Java project with JPA library i.e. Hibernate JARs

2.Create Entity class

---------------------

The main purpose of Entity class is to store data which we want to store in DB or to retrivae data from DB.

Ex. Employee.java

public class Employee

{

private int eno;

private String ename

private float esal;

private String eaddr;

//getter and setter

}

3.Create Entity Mapping file.

-----------------------------

The main intention of Entity Mapping file is to provide mapping between Entity class and database table, ID property and PK column, normal properties and normal columns from Object oriented data model and Relational data model.

Ex.Employee.xml/orm.xml(in case of hibernate Employee.hbm.xml)

<hibernate-mapping>

<class name="com.capgemini.entity.Employee" table="emp1">

<id name="eno"/>

<property name="ename"/>

<property name="esal"/>

<property name="eaddr"/>

</class>

</hibernate-mapping>

Note : We can use Annotations in Entity class as an alternative to mapping file.

@Entity

@Table

@Id

@Column javax.persistence package

4.Create JPA configuration file.(Persistence.xml)

-------------------------------------------------

It is same as Hibernate configration file, it will provide all JPA configuration details like

1. JDBC Parameter configurations

driver class name

driver url

database user name

database password

2.Cache Mechanisms Configurations

3.Transaction Configurations

4.Dialect Configurations

.........

In JPA, configuration file name is "persistence.xml" (default name), it must be created under META-INF folder under SRC.

src/META-INF/persistence.xml

----------------------------

<persistence....XSD.....>

<persistence-unit name="emp"> //session factory from hibernate

<mapping-file>Employee.xml</mapping-file>

or

<class>com.capgemini.entity.Employee</class>//if annotation used

<properties>

<property name="javax.persistence.jdbc.driver" value="oracle.jdbc.OracleDriver"/>

<property name="javax.persistence.jdbc.url" value="jdbc:oracle:thin:@localhost:1521:xe"/>

<property name="javax.persistence.jdbc.user" value="system"/>

<property name="javax.persistence.jdbc.password" value="password" />

<property name="hibernate.dialect" value="org.hibernate.dialect.OracleDialect" />

<property name="hibernate.show\_sql" value="true" />

</properties>

</persistence-unit>

</persistence>

5.Create Test Application

-------------------------

The main intention of Test Application is to perform persistence operations.

Steps:

------

1.Create EntityManagerFactory object.

2.Create EntityManager object

3.Create EntityTranscation object as per the requirement.

4.Perform persistence operations.

5.Perform commit or rollback operations.

1.Create EntityManagerFactory object.

-------------------------------------

javax.persistence.EntityManagerFactory is a Factory of EntityManager objects.

To create EntityManagerFactory class object we have to use following methods from javax.persistence class.

public static EntityManagerFactory createEntityManagerFactory(String persistence\_unit\_name)

Ex.

EntityManagerFactory factory= Persistence.createEntityManagerFactory("emp");

2.Create EntityManager object

-----------------------------

It includes the methods to perform persistence operations.

To get EntityManager object we have to use following method from EntityManager class.

public EntityManager createEntityManager()

Ex.

EntityManager entityManager = factory.createEntityManager();

3.Create EntityTranscation object as per the requirement.

---------------------------------------------------------

If we perform select operation then EntityTransaction object is not required.

EntityTransaction object is required for only non select operations.

public EntityTransaction getTransaction()

Ex.

EntityTransaction entityTransaction = entityManager.getTransaction();

entityTransaction.begin();

4.Perform persistence operations.

---------------------------------

1.public void persist(Object obj)

To insert an object into DB

2.public Object find(Class cls,Serializable pk\_val)

For getting an object from Database

3.public void remove(Object obj)

For removing an Object from Database.

CRUD

----

C-create-insert-persist()

R-Retrieve - read-find(..)

U-

D-Remove-remove-remove(..)

To perform update operations, get Entity object from database and update data in Entity object by calling setXXXX() then commit the Transaction

Ex.

EntityTransaction tx = entityManager.getTransaction();

tx.begin();

Employee emp = (Employee)entityManager.find(Employee.class,111)

emp.setEname("Sachin");

emp.setEsal("75000");

emp.setEaddr("Pune");

tx.commit();

5.Perform commit or rollback operations.

----------------------------------------

public void commit()

public void rollback()

6.Run Test application

----------------------