**Java Persistent API**

1. It is a specification using which the several ORM tool like [Hibernate, iBatis, TopLink] has been built with their own features. In the Object Relational Mapping, we will be mapping class as a database table and variable as the columns of the table.
2. The entire ORM tool using JPA for their implementation.
3. Q: When we have JPA then why we need other ORM tool.

A: JPA is just a specification, it does not provide implementation, its implementation has been built by several ORM tool like hibernate, that is if we have to use JPA then we will use JPA features in our project using some ORM tool like Hibernate or spring data JPA. So if you are using Hibernate then there be only hibernating. But if we are using JPA then either we have to use Hibernate, or spring data JPA or iBatis.

The importance of JPA is if we have implanted our feature using hibernate and if in future if we want to use some other ORM tool then we can easily switch from one ORM tool to another ORM tool

For example in case of Spring Boot Project we use Spring Boot JPA which is built upon Hibernate and JPA.

So to use JAP in our project we have to use one of the implementation tool like Hibernate in the project. So let’s add one dependency in pom.xml

|  |
| --- |
| Pom.xml:  <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <version>5.4.32.Final</version>  </dependency>    <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>8.0.15</version>  </dependency> |
| Persistence.xml  <persistence version=*"2.2"*  xmlns=*"http://xmlns.jcp.org/xml/ns/persistence"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://xmlns.jcp.org/xml/ns/persistence*  [*http://xmlns.jcp.org/xml/ns/persistence/persistence\_2\_2.xsd*](http://xmlns.jcp.org/xml/ns/persistence/persistence_2_2.xsd)*"* >  <persistence-unit name=*"pu"*>  <!--persistence-unit name="jpa-tutorial"-->  <!-- <class>com.mcnz.jpa.examples.Player</class> -->  <!--  <jar-file>hibernate-and-jpa-entities.jar</jar-file>  <exclude-unlisted-classes>true</exclude-unlisted-classes>  -->  <properties>  <!--property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/-->  <property name=*"javax.persistence.jdbc.driver"* value=*"com.mysql.cj.jdbc.Driver"*/>  <property name=*"javax.persistence.jdbc.url"* value=*"jdbc:mysql://localhost:3306/office"*/>  <property name=*"hibernate.dialect"* value=*"org.hibernate.dialect.MySQL8Dialect"* />  <property name=*"javax.persistence.jdbc.user"* value=*"root"*/>  <property name=*"javax.persistence.jdbc.password"* value=*"Admin@123"*/>  <property name=*"hibernate.show\_sql"* value = *"true"* />  <property name=*"hibernate.format\_sql"* value = *"true"* />    <property name=*"javax.persistence.schema-generation.database.action"* value=*"drop-and-create"*/>  <!--  <property name="javax.persistence.query.timeout" value="99"/>  <property name="javax.persistence.lock.timeout" value="99"/>  <shared-cache-mode>ALL</shared-cache-mode>    <property name="javax.persistence.schema-generation.database.action" value="drop-and-create" />  <property name="javax.persistence.sql-load-script-source" value="table-records.sql" />  -->  </properties>    </persistence-unit>  </persistence> |
| **public** **class** App {  **public** **static** **void** main( String[] args )  {  EntityManagerFactory factory= Persistence.*createEntityManagerFactory*("pu");  // Created factory using persistant.xml  // pu:- is defined inside the persistance.xml inside persistanc unit tag    EntityManager manager = factory.createEntityManager();  // Created EntityManager will will manage all the entities present in the application.  Student find = manager.find(Student.**class**, 5);  System.***out***.println(find);  }  } |
| @Entity  **public** **class** Student {  @Id  **private** **int** rollno;  **private** String name;  **private** **int** marks;  } |

1. All the classes which represent a database table in JPA or Hibernate is called entities. Like we have ER diagram in database similarly all the tables are represented by entities and to manage those entity class we have EntitiyManager interface in JPA.
2. Since EntityManager is an interface so we cannot instantiate like [EntityManager = new EntityManager();]

