Upto hibernate 3.2.4 version, Mapping file is mandatory. From hibernate 3.2.5 version,Mapping file is optional and we can use annotations in place of mapping file.

Hibernate uses JPA provided annotation from **javax.persistence** package.

[1.@Entity](mailto:1.@Entity) :- to declare domain class as Entity class.

[2.@Table:-](mailto:2.@Table:-) To provide the mapped table details w.r.t domain class.

Name: It is member of @Table annotation. It takes a table name.

[3.@ID:-](mailto:3.@ID:-) to declare domain class property as ID property.

[4.@Column:-](mailto:4.@Column:-) to declare domain class property as Column. This property represents the column in DB table.

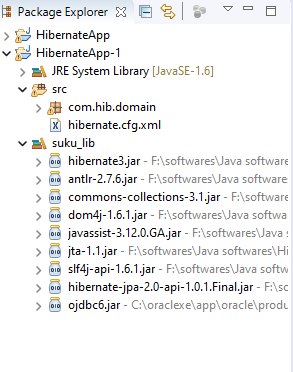
Name:- it member of @ID and @column annotation tag. It takes the column name of table.

Note:- These annotations can be used along with either properties or getXXX(-) method in Domain class.

If we use annotation tags instead of Mapping file in hibernate applications then we should specifiy annotated class/Entity class in hibernate configuration file by using the below tag.

<mapping class=”address of annotated class”/>

Example:-



Domain class: Student.java

**package** com.hib.domain;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

@Entity

@Table(name="student")

**public** **class** Student {

@Id

@Column(name="sid")

**private** **int** sid;

@Column(name="sname")

**private** String sname;

@Column(name="marks")

**private** **float** marks;

**public** **int** getSid() {

**return** sid;

}

**public** **void** setSid(**int** sid) {

**this**.sid = sid;

}

**public** String getSname() {

**return** sname;

}

**public** **void** setSname(String sname) {

**this**.sname = sname;

}

**public** **float** getMarks() {

**return** marks;

}

**public** **void** setMarks(**float** marks) {

**this**.marks = marks;

}

}

Hibernate.cfg.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>oracle.jdbc.OracleDriver</property>

<property name=*"hibernate.connection.url"*>jdbc:oracle:thin:@localhost:1521:xe</property>

<property name=*"hibernate.connection.username"*>SYSTEM</property>

<property name=*"hibernate.connection.password"*>tiger</property>

<property name=*"hibernate.dialect"*>org.hibernate.dialect.Oracle10gDialect</property>

**<mapping class=*"com.hib.domain.Student"*/>**

</session-factory>

</hibernate-configuration>

Test.java

**import** org.hibernate.cfg.Configuration;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

**try** {

**Configuration cfg=new ~~AnnotationConfiguration~~();**

cfg.configure("hibernate.cfg.xml");

SessionFactory sf= cfg.buildSessionFactory();

Session s=sf.openSession();

Student s1=**new** Student();

s1.setSid(7);

s1.setSname("sukumar");

s1.setMarks(79.7f);

Transaction t=s.beginTransaction();

**int** pk=(Integer)s.save(s1);

**if**(pk==7) {

System.***out***.println("Record Inserted");

t.commit();

}

**else**

{

System.***out***.println("Record not inserted");

}

}

**catch**(HibernateException e) {

System.***out***.println(e.getMessage());

}

}

}

Output:-

