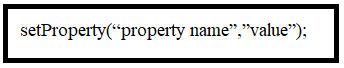
It is not mandatory to create hibernate configuration file in hibernate application. In hibernate applications, we provide DB configuration in two ways.

1. Programmetic Approach.
2. Declarative Approach.

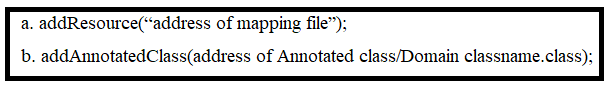
b.1. By using property file.

b.2. By using XML file.

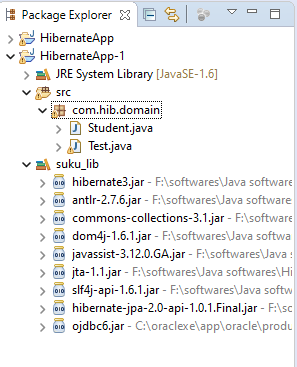
**1.Programmetic Approach:-** we will provide DB configuration (DB url, Driver name, username,password) to Configuration object of Test.java application using setProperty(-,-) method.



The mapping file is provided to Configuration object using either



Example:- In this example , Annotated class is created instead of “mapping file”.



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;

}

}

Client Application:Test.java

**package** com.hib.domain;

**import** org.hibernate.HibernateException;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.~~AnnotationConfiguration~~;

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

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

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

**try** {

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

**cfg.setProperty("hibernate.connection.driver\_class","oracle.jdbc.OracleDriver");**

**cfg.setProperty("hibernate.connection.url","jdbc:oracle:thin:@localhost:1521:xe");**

**cfg.setProperty("hibernate.connection.username","system");**

**cfg.setProperty("hibernate.connection.password","tiger");**

**cfg.setProperty("hibernate.connection.dialect","org.hibernate.dialect.Oracle10gDialect");**

**cfg.addAnnotatedClass(Student.class);**

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());

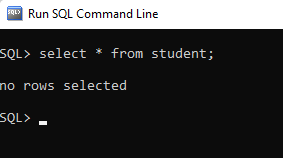
}

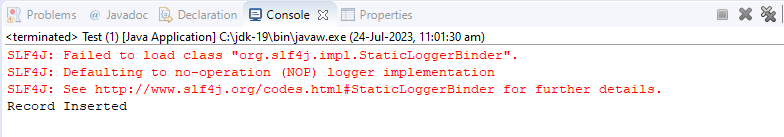
}

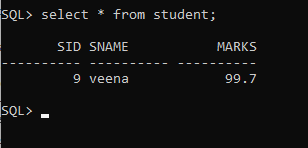
}

Output:-

---------







**2. Declarative Approach:**

**2.1. Properties File:-**  we create property file with any name. But recommended name is “hibernate” and recommended extension is “properties”. Place this file in src folder.

Ex:- hibernate.properties

This file have configuration details in the form of key /value pairs like

Hibernate.connection.url=jdbc:oracle:thin:@localhost:1521:xe

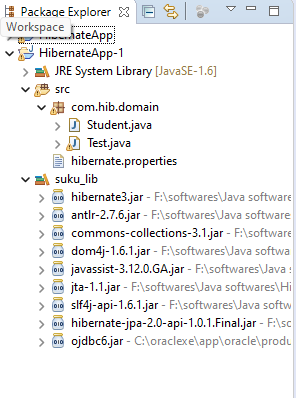
Hibernate.connection.username=system

---

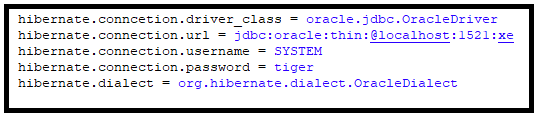
Etc.

In test application, create “configuration object”. While creating the configuration object , hibernate searches for “hibernate.properties” file by default in src folder. If hibernate find it, it read content and stores in configuration object otherwise it raises exception.

Example:-



**Property File: hibernate.properties**



**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;

}

}

**ClientApp:Test.java**

**package** com.hib.domain;

**import** org.hibernate.HibernateException;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.~~AnnotationConfiguration~~;

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

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

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

**try** {

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

cfg.addAnnotatedClass(Student.**class**);

SessionFactory sf= cfg.buildSessionFactory();

Session s=sf.openSession();

Transaction t=s.beginTransaction();

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

s1.setSid(9);

s1.setSname("veena");

s1.setMarks(99.7f);

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

**if**(pk==9) {

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

t.commit();

}

**else**

{

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

}

}

**catch**(HibernateException e) {

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

e.printStackTrace();

}

}

}

Output:-

