以员工与部门为例

1. 创建两个类
2. Employee.Java
3. **import** java.io.Serializable;
4. **public** **class** Employee **implements** Serializable { //一定要实现Serializable接口，这是属于多的一方
6. **private** **int** empId;
7. **private** String empName;
8. **private** **double** salary;
9. **private** Dept dept;
10. //这里自己添加get和set方法
11. //也可以添加toString方法
12. }

Dept.java类

**import** java.util.HashSet;

**import** java.util.Set;

**public** **class** Dept {

**private** **int** deptId;

**private** String deptName;

**private** Set<Employee> emps=**new** HashSet<Employee>();

1. //这里自己添加get和set方法
2. //也可以添加toString方法

}



<!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 >

<!-- 数据库配置 -->

<property name=*"hibernate.connection.driver\_class"*>com.mysql.jdbc.Driver</property>

<property name=*"hibernate.connection.url"*>jdbc:mysql://localhost:3306/empresa?serverTimezone=UTC</property>

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

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

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

<!-- 配置自动建表 -->

<property name=*"hibernate.hbm2ddl.auto"*>update</property>

<!-- 加载所有映射 -->

<mapping resource=*"my/kenny/entity/Employee.hbm.xml"*/>

<mapping resource=*"my/kenny/entity/Dept.hbm.xml"*/>

</session-factory>

</hibernate-configuration>



<?xml version=*"1.0"*?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

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

<hibernate-mapping package=*"my.kenny.entity"*>

<class name=*"Dept"* table=*"t\_dept"*>

<!-- 主键映射-->

<id name=*"deptId"*>

<generator class=*"native"*/>

</id>

<!-- 非主键映射-->

<property name=*"deptName"* length=*"20"*></property>

<!-- 一对多关联映射的配置 -->

<!—想一次性级联保存的话，最好加上cascade="all"否则报错 -->

<set name=*"emps"* table=*"t\_employee"* cascade="all"><!—其实这个不安全，正确的做法，cascade="none"，这个是默认值先保存Employee对象，再保存Dept对象 -->

<key column=*"dept\_Id"*></key>

<one-to-many class=*"Employee"*/>

</set>

</class>

</hibernate-mapping>



<?xml version=*"1.0"*?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

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

<hibernate-mapping package=*"my.kenny.entity"*>

<class name=*"Employee"* table=*"t\_employee"*>

<!-- 主键映射-->

<id name=*"empId"* >

<generator class=*"native"*/>

</id>

<!-- 非主键映射-->

<property name=*"empName"* length=*"20"* type=*"string"*></property>

<property name=*"salary"* type=*"double"*></property>

<!-- 多对一映射 -->

<many-to-one name=*"dept"* column=*"dept\_Id"* class=*"Dept"*></many-to-one>

</class>

</hibernate-mapping>

测试代码

**package** my.kenny.test;

**import** java.util.HashSet;

**import** java.util.Set;

**import** my.kenny.dao.EmployeeDao;

**import** my.kenny.entity.Dept;

**import** my.kenny.entity.Employee;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

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

**import** org.junit.Test;

**public** **class** TestApp {

@Test

**public** **void** saveDept(){

//创建部门对象

// Dept dept=new Dept();

// dept.setDeptName("人事部");

// //创建几个员工对象

// Employee e1=new Employee();

// e1.setEmpName("Jack");

// e1.setSalary(3000);

// //e1.setDept(dept);

// Employee e2=new Employee();

// e2.setEmpName("John");

// e2.setSalary(3500);

// //e2.setDept(dept);

//

// Set<Employee> emps=new HashSet<Employee>();

// emps.add(e1);

// emps.add(e2);

// dept.setEmps(emps);

//

// EmployeeDao eld=new EmployeeDao();

// eld.saveDept(dept);

Configuration conf=**new** Configuration();

conf.configure();

SessionFactory sf=conf.buildSessionFactory();

Session session =sf.openSession();

Dept dept=**new** Dept();

dept.setDeptName("研发部");

//创建几个员工对象

Employee e1=**new** Employee();

e1.setEmpName("randlo");

e1.setSalary(3000);

e1.setDept(dept);

Employee e2=**new** Employee();

e2.setEmpName("margret");

e2.setSalary(3500);

e2.setDept(dept);

Transaction ts=session.beginTransaction();

//Set<Employee> emps=new HashSet<Employee>();

dept.getEmps().add(e1);

dept.getEmps().add(e2);

// dept.setEmps(emps);

session.save(dept);

ts.commit();

session.close();

}

@Test

**public** **void** testSaveEmp(){

Employee e1=**new** Employee();

e1.setEmpName("Jack");

e1.setSalary(3000);

//e1.setDept(1);

EmployeeDao eld=**new** EmployeeDao();

eld.saveEmp(e1);

}

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

//new TestApp().saveDept();

}

}