#  1.1我们对获取session的工具类，升级，让它可以直接返回 全新的session和线程相关的session

**final** **public** **class** HibernateUtil {

**private** **static** SessionFactory *sessionFactory*=**null**;

//使用线程局部模式

**private** **static** ThreadLocal<Session> *threadLocal*=**new** ThreadLocal<Session>();

//不然随便实力话

**private** HibernateUtil(){};

**static** {

*sessionFactory*=**new** Configuration().configure("com/hsp/config/hsp.cfg.xml").buildSessionFactory();

}

//获取全新的全新的sesession

**public** **static** Session openSession(){

**return** *sessionFactory*.openSession();

}

//获取和线程关联的session

**public** **static** Session getCurrentSession(){

Session session=*threadLocal*.get();

//判断是否得到了session 如果没有得到就开始 让他得到

**if**(session==**null**){

session=*sessionFactory*.openSession();

//把session对象设置到 threadLocal,相当于该session已经和线程绑定

//这样就可以不要cfg 中的配置了

// <property name="current\_session\_context\_class">thread</property>

*threadLocal*.set(session);

}

**return** session;

}

}

# 2.进行测试

**private** **static** **void** testHibernateUtil(){

Session session = HibernateUtil.*getCurrentSession*();

Session session2 = HibernateUtil.*getCurrentSession*();

/\*System.out.println(employee); \*/

System.*out*.println(session.hashCode()+" "+session2.hashCode());

}

937 [main] INFO org.hibernate.impl.SessionFactoryObjectFactory - Not binding factory to JNDI, no JNDI name configured

306351104 306351104