实验四，　Spring2整合Hibernate3

```
SpringHibernateDemo
        |
        |--------tarena
                 |
                 |------spring-config.xml
```

```
        <!-- 加入Spring2Core -->
        <!-- 加入Spring2AOP -->
        <!-- 加入Spring2PersistenceCore -->
        <!-- 加入Hibernate3.1Core -->
        <!-- 加入MySQL -->
```

步骤一，创建POJO对象

```
public class User implements java.io.Serializable {

        private Integer userId;
        private String userName;
        private String userGender;

        public User() {
        }

        public User(String userName, String userGender) {
                this.userName = userName;
                this.userGender = userGender;
        }

        public Integer getUserId() {
                return this.userId;
        }
```

```
	public void setUserId(Integer userId) {
		this.userId = userId;
	}

	public String getUserName() {
		return this.userName;
	}

	public void setUserName(String userName) {
		this.userName = userName;
	}

	public String getUserGender() {
		return this.userGender;
	}

	public void setUserGender(String userGender) {
		this.userGender = userGender;
	}

}
```

步骤二，创建POJO映射文件

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="tarena.User" table="user" catalog="test">
        <id name="userId" type="integer">
            <column name="user_id" />
            <generator class="increment" />
        </id>
        <property name="userName" type="string">
            <column name="user_name" length="20" />
        </property>
        <property name="userGender" type="string">
```

```
      <column name="user_gender" length="1" />
    </property>
  </class>
</hibernate-mapping>
```

步骤三，创建接口

```
public interface IUserDao {
        public void find(int id);
        public void insert(User user);
        public void update(User user);
        public void delete(User user);
}
```

步骤四，创建接口的实现类

```
public class UserDaoImpl extends HibernateDaoSupport implements
IUserDao {

        @Transactional(propagation = Propagation.REQUIRED)
        public void delete(User user) {
                System.out.println("正在删除用户。。。");
        }

        @Transactional(readOnly = true)
        public void find(int id) {
                System.out.println("正在查询用户。。。");
                String hql = "FROM User u WHERE u.userId =
?";
                Object[] args = { id };
                List<User> users =
this.getHibernateTemplate().find(hql, args);
                System.out.print(users.size());
        }

        @Transactional(propagation = Propagation.REQUIRED)
        public void insert(User user) {
                System.out.println("正在增加用户。。。");
                this.getHibernateTemplate().save(user);
```

```
                    // 产生异常，事务回滚，不会入库
                    //Integer.parseInt("abc123");
          }

          @Transactional(propagation = Propagation.REQUIRED)
          public void update(User user) {
                    System.out.println("正在修改用户。。。");
          }

}
```

步骤五，

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:tx="http://www.springframework.org/schema/tx"

xsi:schemaLocation="http://www.springframework.org/schema/bea
ns

http://www.springframework.org/schema/beans/spring-beans-2.0.x
sd
          http://www.springframework.org/schema/tx

http://www.springframework.org/schema/tx/spring-tx-2.0.xsd">

          <!-- 修改XML头信息 -->

          <!-- 带连接池功能的
org.apache.commons.dbcp.BasicDataSource -->
          <!-- 数据源 -->
          <bean id="DriverManagerDataSource"

class="org.springframework.jdbc.datasource.DriverManagerDataSour
ce">
                    <property name="driverClassName">
```

```
<value>com.mysql.jdbc.Driver</value>
                </property>
                <property name="url">

<value>jdbc:mysql://localhost:3306/test</value>
                </property>
                <property name="username">
                        <value>root</value>
                </property>
                <property name="password">
                        <value>admin</value>
                </property>
        </bean>



        <!-- SessionFactory -->
        <bean id="LocalSessionFactoryBean"

class="org.springframework.orm.hibernate3.LocalSessionFactoryBea
n">
                <property name="dataSource">
                        <ref bean="DriverManagerDataSource"
/>
                </property>
                <property name="mappingResources">
                        <list>

<value>tarena/User.hbm.xml</value>
                        </list>
                </property>
        </bean>


        <!-- 持久层 -->
        <bean id="UserDaoImpl" class="tarena.UserDaoImpl">
                <property name="sessionFactory">
                        <ref bean="LocalSessionFactoryBean"
/>
                </property>
```

```
    </bean>



    <!-- Hibernate事务管理器 -->
    <bean id="HibernateTransactionManager"

class="org.springframework.orm.hibernate3.HibernateTransactionMa
nager">
        <property name="sessionFactory">
            <ref bean="LocalSessionFactoryBean"
/>
        </property>
    </bean>



    <!-- Spring2声明事务管理 -->
    <tx:annotation-driven

transaction-manager="HibernateTransactionManager" />

</beans>
```

步骤六，

```
public class Test {

    public static void main(String[] args) {

        ApplicationContext context = new
ClassPathXmlApplicationContext(
                "tarena/spring-config.xml");

        IUserDao userDao = (IUserDao)
context.getBean("UserDaoImpl");
```

```
                        // userDao.find(5);

                        User user = new User();
                        user.setUserName("zhaojun");
                        user.setUserGender("m");

                        userDao.insert(user);
            }
}
```

步骤七，
正在增加用户。。。