Hibernate框架第一天

今天任务

1. 使用Hibernate框架完成对客户的增删改查的操作

教学导航

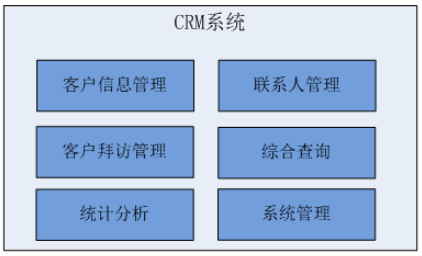
1. 能够说出Hibernate的执行流程
2. 能够独立使用Hibernate框架完成增删改查的操作

框架和CRM项目的整体介绍

1. 什么是CRM
    * CRM（Customer Relationship Management）客户关系管理，是利用相应的信息技术以及互联网技术来协调企业与顾客间在销售、营销和服务上的交互，向客户提供创新
    * 其最终目标是将面向客户的各项信息和活动集成起来，组建一个以客户为中心的企业，实现对面向客户的活动的全面管理

2. CRM的模块
    * CRM系统实现了对企业销售、营销、服务等各阶段的客户信息、客户活动进行统一管理。
    * CRM系统功能涵盖企业销售、营销、用户服务等各各业务流程，业务流程中与客户相关活动都会在CRM系统统一管理。
    * 下边列出一些基本的功能模块，包括：
        * 客户信息管理
        * 联系人管理
        * 商机管理
        * 统计分析等

CRM系统的模块



3. 模块的具体功能
    * 客户信息管理
        * 对客户信息统一维护，客户是指存量客户或拟营销的客户，通过员工录入形成公司的“客户库”是公司最重要的数据资源。

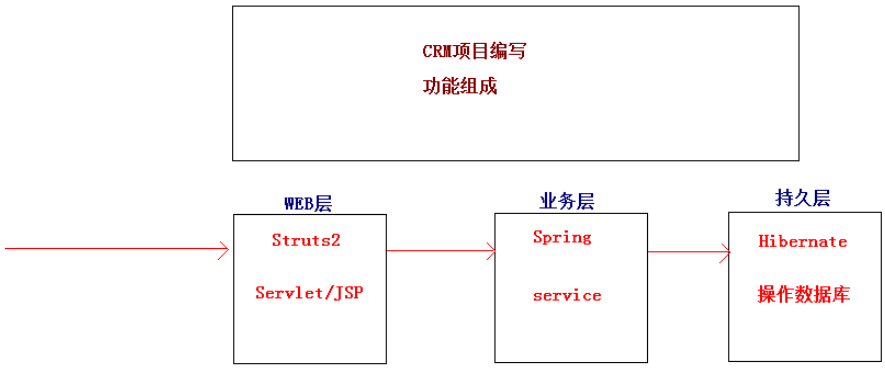
    * 联系人管理
        * 对客户的联系人信息统一管理，联系人是指客户企业的联系人，即企业的业务人员和客户的哪些人在打交道。

    * 客户拜访管理
        * 业务员要开发客户需要去拜访客户，客户拜访信息记录了业务员与客户沟通交流方面的不足、采取的策略不当、有待改进的地方或值得分享的沟通技巧等方面的信

* 综合查询
    * 客户相关信息查询，包括：客户信息查询、联系人信息查询、商机信息查询等

* 统计分析
    * 按分类统计客户信息，包括：客户信息来源统计、按行业统计客户、客户发展数量统计等

* 系统管理
    系统管理属于crm系统基础功能模块，包括：数据字典、账户管理、角色管理、权限管理、操作日志管理等



Hibernate框架的学习路线

1. 注意：Hibernate框架知识点非常多，比较杂乱，大家要做好笔记记录的工作
2. 学习的路线
    * 第一天：主要是学习框架的入门，自己搭建框架，完成增删改查的操作
    * 第二天：主要学习一级缓存、事务管理和基本的查询
    * 第三天：主要学习一对多和多对多的操作等
    * 第四天：基本查询和查询的优化

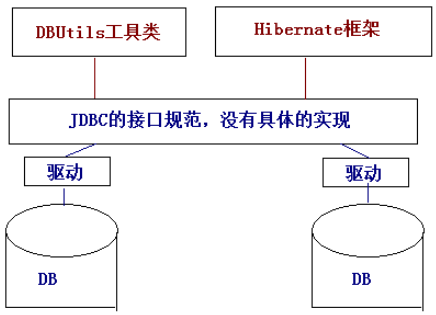
Hibernate未来的开发位置：持久层的框架

默认规范：

```
class User{
   int uid;
   String name;

}


t_user(
  uid
  name
);
```



好处：简化了开发，提供了新的功能（缓存功能）

什么是框架：框架的软件的半成品，帮助咱们完成部分的功能

案例一：完成客户的CRUD的操作

需求分析

1. CRM系统中客户信息管理模块功能包括
    * 新增客户信息
    * 客户信息查询
    * 修改客户信息
    * 删除客户信息

2. 要实现客户的新增功能

技术分析之Hibernate框架的概述

Hibernate框架的概述

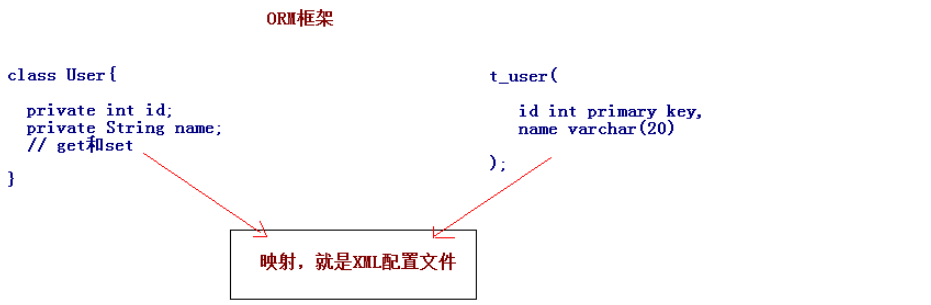
1. Hibernate框架的概述
    * Hibernate称为
    * Hibernate是一个开放源代码的对象关系映射（ORM）框架，它对JDBC进行了非常轻量级的对象封装，使得Java程序员可以随心所欲的使用对象编程思维来操纵数据库。
    * Hibernate可以应用在任何使用JDBC的场合，既可以在Java的客户端程序使用，也可以在Servlet/JSP的Web应用中使用。
    * Hibernate是轻量级JavaEE应用的持久层解决方案，是一个关系数据库ORM框架

2. 记住：Hibernate是一个持久层的ORM框架！！！

什么是ORM（对象关系映射）

1. ORM映射：Object Relational Mapping
    * O：面向对象领域的Object（JavaBean对象）
    * R：关系数据库领域的Relational（表的结构）
    * M：映射Mapping（XML的配置文件）

2. 简单一句话：Hibernate使程序员通过操作对象的方式来操作数据库表记录



操作数据库，操作User对象（Java程序员），通过映射的关联操作数据库的表结构（Hibernate框架完成的）

Hibernate优点

1. 优点
    * Hibernate对JDBC访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码
    * Hibernate是一个基于jdbc的主流持久化框架，是一个优秀的orm实现，它很大程度的简化了dao层编码工作
    * Hibernate的性能非常好，因为它是一个轻量级框架。映射的灵活性很出色。它支持很多关系型数据库，从一对一到多对多的各种复杂关系

技术分析之Hibernate框架的快速入门

第一步：下载Hibernate5的运行环境

1. 下载相应的jar包等
    * http://sourceforge.net/projects/hibernate/files/hibernate-orm/5.0.7.Final/hibernate-release-5.0.7.Final.zip/download

2. 解压后对目录结构有一定的了解

第二步：创建表结构

1. 建表语句如下
```
Create database hibernate_day01;
Use hibernate_day01;
CREATE TABLE `cst_customer` (
  `cust_id` bigint(32) NOT NULL AUTO_INCREMENT COMMENT '客户编号(主键)',
  `cust_name` varchar(32) NOT NULL COMMENT '客户名称(公司名称)',
  `cust_user_id` bigint(32) DEFAULT NULL COMMENT '负责人id',
  `cust_create_id` bigint(32) DEFAULT NULL COMMENT '创建人id',
  `cust_source` varchar(32) DEFAULT NULL COMMENT '客户信息来源',
  `cust_industry` varchar(32) DEFAULT NULL COMMENT '客户所属行业',
  `cust_level` varchar(32) DEFAULT NULL COMMENT '客户级别',
  `cust_linkman` varchar(64) DEFAULT NULL COMMENT '联系人',
  `cust_phone` varchar(64) DEFAULT NULL COMMENT '固定电话',
  `cust_mobile` varchar(16) DEFAULT NULL COMMENT '移动电话',
  PRIMARY KEY (`cust_id`)
) ENGINE=InnoDB AUTO_INCREMENT=94 DEFAULT CHARSET=utf8;
```

第三步：搭建Hibernate的开发环境

1. 创建WEB工程，引入Hibernate开发所需要的jar包
    * MySQL的驱动jar包
    * Hibernate开发需要的jar包（资料/hibernate-release-5.0.7.Final/lib/required/所有jar包）
    * 日志jar包（资料/jar包/log4j/所有jar包）

第四步：编写JavaBean实体类

1. Customer类的代码如下：
```
public class Customer {
    private Long cust_id;
    private String cust_name;
    private Long cust_user_id;
    private Long cust_create_id;
    private String cust_source;
    private String cust_industry;
    private String cust_level;
    private String cust_linkman;
    private String cust_phone;
    private String cust_mobile;
    // 省略get和set方法
}
```

第五步：创建类与表结构的映射

1. 在JavaBean所在的包下创建映射的配置文件
   * 默认的命名规则为：实体类名.hbm.xml
   * 在xml配置文件中引入约束（引入的是hibernate3.0的dtd约束，不要引入4的约束）
   ```
   <!DOCTYPE hibernate-mapping PUBLIC
       "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
       "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
   ```

2. 如果不能上网，编写配置文件是没有提示的，需要自己来配置
   * 先复制http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd --> window --> preferences --> 搜索xml --> 选择xml catalog --> 点击add --> 现在URI --

3. 编写映射的配置文件
   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <!DOCTYPE hibernate-mapping PUBLIC
       "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
       "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

   <hibernate-mapping>
       <class name="com.itheima.domain.Customer" table="cst_customer">
           <id name="cust_id" column="cust_id">
               <generator class="native"/>
           </id>
           <property name="cust_name" column="cust_name"/>
           <property name="cust_user_id" column="cust_user_id"/>
           <property name="cust_create_id" column="cust_create_id"/>
           <property name="cust_source" column="cust_source"/>
           <property name="cust_industry" column="cust_industry"/>
           <property name="cust_level" column="cust_level"/>
           <property name="cust_linkman" column="cust_linkman"/>
           <property name="cust_phone" column="cust_phone"/>
           <property name="cust_mobile" column="cust_mobile"/>
       </class>
   </hibernate-mapping>
   ```

第六步：编写Hibernate核心的配置文件

1. 在src目录下，创建名称为hibernate.cfg.xml的配置文件
2. 在XML中引入DTD约束
   ```
   <!DOCTYPE hibernate-configuration PUBLIC
       "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
       "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
   ```

3. 打开：资料/hibernate-release-5.0.7.Final/project/etc/hibernate.properties，可以查看具体的配置信息
   * 必须配置的4大参数
   ```
   #hibernate.connection.driver_class com.mysql.jdbc.Driver
   #hibernate.connection.url jdbc:mysql:///test
   #hibernate.connection.username gavin
   #hibernate.connection.password
   ```
   * 数据库的方言（必须配置的）
   ```
   #hibernate.dialect org.hibernate.dialect.MySQLDialect
   ```
   * 可选的配置
   ```
   #hibernate.show_sql true
   #hibernate.format_sql true
   #hibernate.hbm2ddl.auto update
   ```
   * 引入映射配置文件（一定要注意，要引入映射文件，框架需要加载映射文件）
   ```
   * <mapping resource="com/itheima/domain/Customer.hbm.xml"/>
   ```

4. 具体的配置如下
   ```
   <?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">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.url">jdbc:mysql:///hibernate_day01</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">root</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

        <mapping resource="com/itheima/domain/Customer.hbm.xml"/>
    </session-factory>
</hibernate-configuration>
```

第七步：编写Hibernate入门代码

1. 具体的代码如下

```
/**
 * 测试保存客户
 */
@Test
public void testSave(){
    // 先加载配置文件
    Configuration config = new Configuration();
    // 默认加载src目录下的配置文件
    config.configure();
    // 创建SessionFactory对象
    SessionFactory factory = config.buildSessionFactory();
    // 创建session对象
    Session session = factory.openSession();
    // 开启事务
    Transaction tr = session.beginTransaction();
    // 编写保存代码
    Customer c = new Customer();
    // c.setCust_id(cust_id);   已经自动递增
    c.setCust_name("测试名称");
    c.setCust_mobile("110");
    // 保存客户
    session.save(c);
    // 提交事务
    tr.commit();
    // 释放资源
    session.close();
    factory.close();
}
```

回忆：快速入门

1. 下载Hibernate框架的开发包
2. 编写数据库和表结构
3. 创建WEB的项目，导入了开发的jar包
   * MySQL驱动包、Hibernate开发的必须要有的jar包、日志的jar包
4. 编写JavaBean，以后不使用基本数据类型，使用包装类
5. 编写映射的配置文件（核心），先导入开发的约束，里面正常配置标签
6. 编写hibernate的核心的配置文件，里面的内容是固定的
7. 编写代码，使用的类和方法

技术分析之：Hibernate常用的配置文件

Hibernate配置文件之映射配置文件

1. 映射文件，即Stu.hbm.xml的配置文件
    * <class>标签　　-- 用来将类与数据库表建立映射关系
        * name　　-- 类的全路径
        * table　　-- 表名.(类名与表名一致,那么table属性也可以省略)
        * catalog　　-- 数据库的名称，基本上都会省略不写

    * <id>标签　　-- 用来将类中的属性与表中的主键建立映射，id标签就是用来配置主键的。
        * name　　-- 类中属性名
        * column　　-- 表中的字段名.(如果类中的属性名与表中的字段名一致,那么column可以省略.)
        * length　　-- 字段的程度，如果数据库已经创建好了，那么length可以不写。如果没有创建好，生成表结构时，length最好指定。

    * <property>　　-- 用来将类中的普通属性与表中的字段建立映射.
        * name　　-- 类中属性名
        * column　　-- 表中的字段名.(如果类中的属性名与表中的字段名一致,那么column可以省略.)
        * length　　-- 数据长度
        * type　　-- 数据类型（一般都不需要编写，如果写需要按着规则来编写）
            * Hibernate的数据类型　　type="string"
            * Java的数据类型　　type="java.lang.String"
            * 数据库字段的数据类型　　<column name="name" sql-type="varchar"/>

Hibernate配置文件之核心配置文件

1. 核心配置文件的两种方式
    * 第一种方式是属性文件的形式，即properties的配置文件
        * hibernate.properties
            * hibernate.connection.driver_class=com.mysql.jdbc.Driver
        * 缺点
            * 不能加载映射的配置文件，需要手动编写代码去加载

    * 第二种方式是XML文件的形式，开发基本都会选择这种方式
        * hibernate.cfg.xml
            * <property name="hibernate.connection.driver_class" >com.mysql.jdbc.Driver</property>
        * 优点
            * 格式比较清晰
            * 编写有提示
            * 可以在该配置文件中加载映射的配置文件（最主要的）

2. 关于hibernate.cfg.xml的配置文件方式
    * 必须有的配置
        * 数据库连接信息:
            hibernate.connection.driver_class　　-- 连接数据库驱动程序
            hibernate.connection.url　　-- 连接数据库URL
            hibernate.connection.username　　-- 数据库用户名
            hibernate.connection.password　　-- 数据库密码

        * 方言:
            hibernate.dialect　　-- 操作数据库方言

    * 可选的配置
        * hibernate.show_sql　　-- 显示SQL
        * hibernate.format_sql　　-- 格式化SQL
        * hibernate.hbm2ddl.auto　　-- 通过映射转成DDL语句
            * create　　-- 每次都会创建一个新的表.---测试的时候
            * create-drop　　-- 每次都会创建一个新的表,当执行结束之后,将创建的这个表删除.---测试的时候
            * update　　-- 如果有表,使用原来的表.没有表,创建一个新的表.同时更新表结构.
            * validate　　-- 如果有表,使用原来的表.同时校验映射文件与表中字段是否一致如果不一致就会报错.

    * 加载映射
        * 如果XML方式：<mapping resource="cn/itcast/hibernate/domain/User.hbm.xml" />

技术分析之Hibernate常用的接口和类

Configuration类和作用

1. Configuration类
    * Configuration对象用于配置并且启动Hibernate。
    * Hibernate应用通过该对象来获得对象-关系映射文件中的元数据，以及动态配置Hibernate的属性，然后创建SessionFactory对象。

    * 简单一句话：加载Hibernate的配置文件，可以获取SessionFactory对象。

2. Configuration类的其他应用（了解）
    * 加载配置文件的种类，Hibernate支持xml和properties类型的配置文件，在开发中基本都使用XML配置文件的方式。
        * 如果采用的是properties的配置文件，那么通过Configuration configuration = new Configuration();就可以假装配置文件
            * 但是需要自己手动加载映射文件
            * 例如：config.addResource("cn/itcast/domain/Student.hbm.xml");

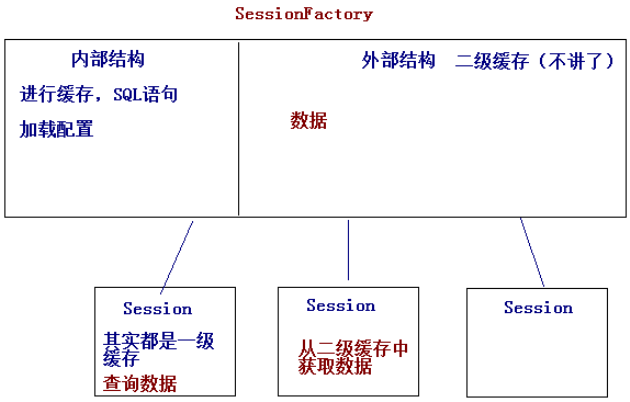
        * 如果采用的XML的配置文件，通过Configuration configuration = new Configuration().configure();加载配置文件

SessionFactory：重要

1. 是工厂类，是生成Session对象的工厂类
2. SessionFactory类的特点
    * 由Configuration通过加载配置文件创建该对象。
    * SessionFactory对象中保存了当前的数据库配置信息和所有映射关系以及预定义的SQL语句。同时，SessionFactory还负责维护Hibernate的二级缓存。
        * 预定义SQL语句
            * 使用Configuration类创建了SessionFactory对象是，已经在SessionFacotry对象中缓存了一些SQL语句
            * 常见的SQL语句是增删改查（通过主键来查询）
            * 这样做的目的是效率更高

    * 一个SessionFactory实例对应一个数据库，应用从该对象中获得Session实例。
    * SessionFactory是线程安全的，意味着它的一个实例可以被应用的多个线程共享。
    * SessionFactory是重量级的，意味着不能随意创建或销毁它的实例。如果只访问一个数据库，只需要创建一个SessionFactory实例，且在应用初始化的时候完成。
    * SessionFactory需要一个较大的缓存，用来存放预定义的SQL语句及实体的映射信息。另外可以配置一个缓存插件，这个插件被称之为Hibernate的二级缓存，被多线程

3. 总结
    * 一般应用使用一个SessionFactory,最好是应用启动时就完成初始化。



编写HibernateUtil的工具类

1. 具体代码如下

```
public class HibernateUtil {
```

```
    private static final Configuration cfg;
    private static final SessionFactory factory;
    static{
        // 给常量赋值
        // 加载配置文件
        cfg = new Configuration().configure();
        // 生成factory对象
        factory = cfg.buildSessionFactory();
    }
    // 获取Session对象
    public static Session openSession(){
        return factory.openSession();
    }
}
```

Session接口

1. 概述
    * Session是在Hibernate中使用最频繁的接口。也被称之为持久化管理器。它提供了和持久化有关的操作，比如添加、修改、删除、加载和查询实体对象
    * Session 是应用程序与数据库之间交互操作的一个单线程对象，是 Hibernate 运作的中心
    * Session是线程不安全的
    * 所有持久化对象必须在 session 的管理下才可以进行持久化操作
    * Session 对象有一个一级缓存，显式执行 flush 之前，所有的持久化操作的数据都缓存在 session 对象处
    * 持久化类与 Session 关联起来后就具有了持久化的能力

2. 特点
    * 不是线程安全的。应避免多个线程使用同一个Session实例
    * Session是轻量级的，它的创建和销毁不会消耗太多的资源。应为每次客户请求分配独立的Session实例
    * Session有一个缓存，被称之为Hibernate的一级缓存。每个Session实例都有自己的缓存

3. 常用的方法
    * save(obj)
    * delete(obj)
    * get(Class,id)
    * update(obj)
    * saveOrUpdate(obj)　　-- 保存或者修改（如果没有数据，保存数据。如果有，修改数据）
    * createQuery()　　-- HQL语句的查询的方式

Transaction接口

1. Transaction是事务的接口
2. 常用的方法
    * commit()　　-- 提交事务
    * rollback()　　-- 回滚事务

3. 特点
    * Hibernate框架默认情况下事务不自动提交.需要手动提交事务
    * 如果没有开启事务，那么每个Session的操作，都相当于一个独立的事务

开发步骤

1. 准备环境
    * 在资料/crm/ui/WebRoot下所有的文件，拷贝到工程中
    * 引入JSTL的标签库，JSP页面会报错

编写代码