

1. JavaEE三层架构

1. web层: Struts2, SpringMVC
 2. service层: Spring
 3. dao层: Hibernate, MyBatis
- 3-1. 对数据库进行CRUD操作

2. MVC思想

1. M: 模型 (javabean)
2. V: 视图 (jsp)
3. C: 控制器 (servlet)

3. Hibernate概述

1. 什么是框架

写程序，使用框架之后，框架帮我们实现一部分功能，使用框架的好处是可以少写一部分代码就可以实现系统功能。

2. 什么是Hibernate框架

1. hibernate框架是应用在JavaEE三层架构中的dao层的框架。

2. 在dao层里面做对数据库的crud操作，使用hibernate实现crud操作，hibernate底层代码就是jdbc，hibernate对jdbc进行了封装，使用hibernate好处，不需要写复杂的jdbc代码了，不需要写sql语句实现。

3. hibernate是开源的轻量级的框架。

4. hibernate版本

1. hibernate3.x
2. hibernate4.x
3. hibernate5.x (学习)

hibernate-release-5.0.7.Final.zip\hibernate-release-5.0.7.Final - 解包大小为 284.1 MB			
名称	压缩前	压缩后	类型
.. (上级目录)			文件夹
documentation			文件夹
lib			文件夹
project			文件夹

lib: hibernate相关jar包

documentation: hibernate相关文档

project: hibernate相关源代码

4. ORM思想

1. hibernate使用orm思想对数据库进行crud操作。
2. 在web阶段学习javabean，更正确的叫法是实体类。
3. orm: object relational mapping, 对象关系型数据库映射

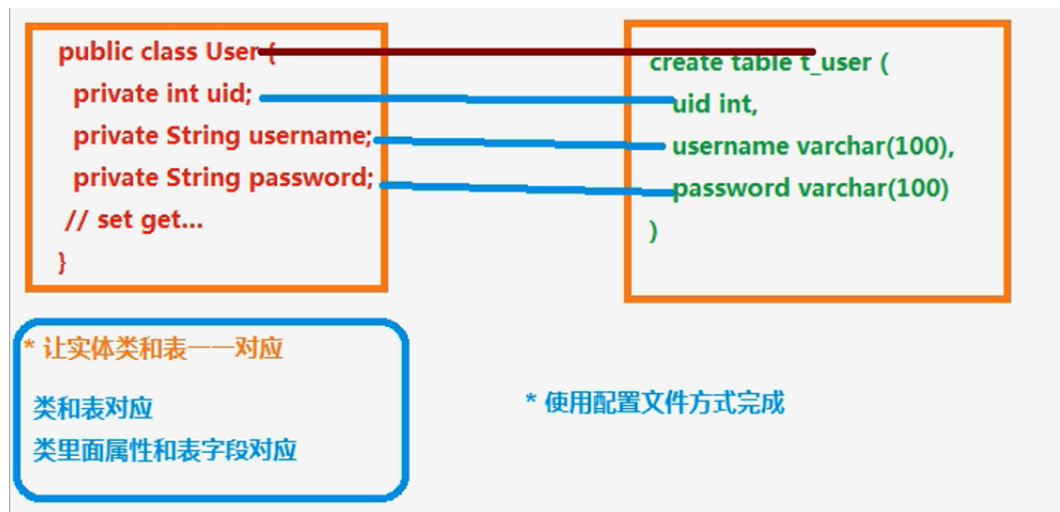
文字描述:

3-1. 让实体类和数据库表进行一一对应关系

让实体类首先和数据库表对应

让实体类属性和表里的字段对应

画图描述



3-2. 不需要直接操作数据库中的表，而是去操作表所对应的实体类。

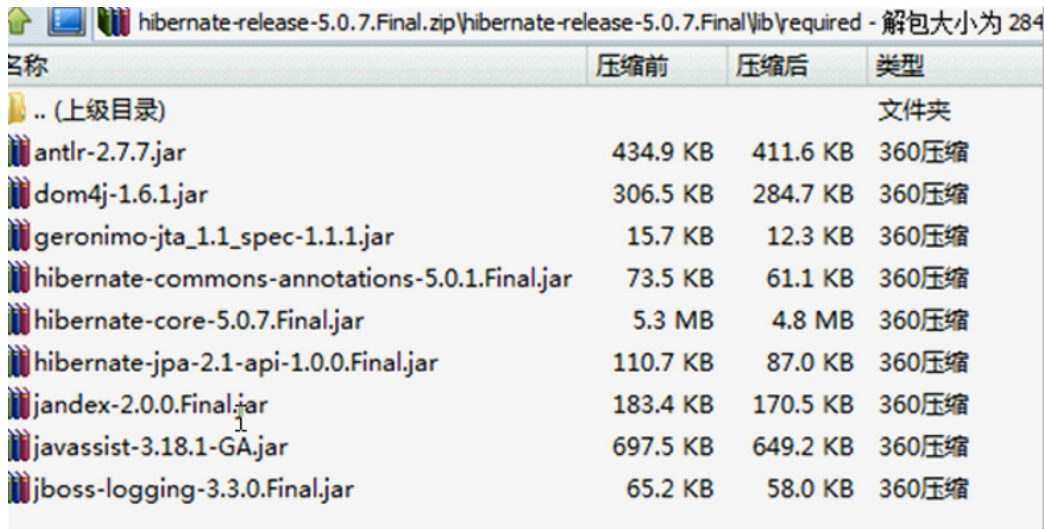
3-3. 调用hibernate封装的session对象对实体类进行操作。

```
//向数据库表中加入lucy实体类对象
```

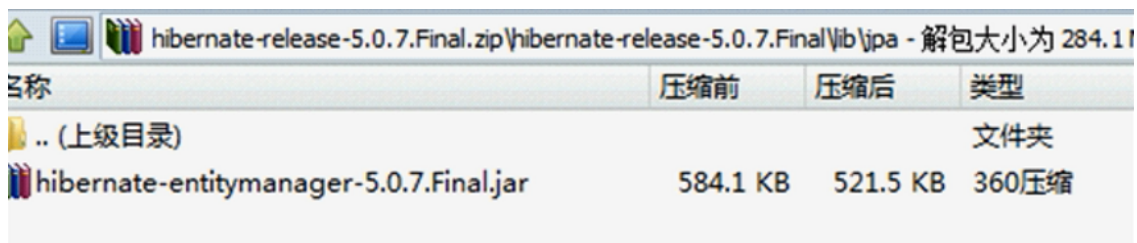
```
User user = new User();  
user.setUsername("lucy");  
session.save(user);
```

5. 搭建Hibernate环境

第一步：导入hibernate的jar包。

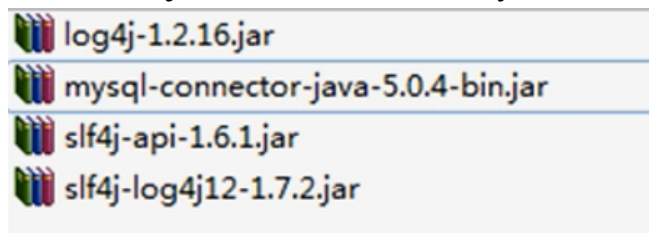


名称	压缩前	压缩后	类型
.. (上级目录)			文件夹
antlr-2.7.7.jar	434.9 KB	411.6 KB	360压缩
dom4j-1.6.1.jar	306.5 KB	284.7 KB	360压缩
geronimo-jta_1.1_spec-1.1.1.jar	15.7 KB	12.3 KB	360压缩
hibernate-commons-annotations-5.0.1.Final.jar	73.5 KB	61.1 KB	360压缩
hibernate-core-5.0.7.Final.jar	5.3 MB	4.8 MB	360压缩
hibernate-jpa-2.1-api-1.0.0.Final.jar	110.7 KB	87.0 KB	360压缩
jandex-2.0.0.Final.jar	183.4 KB	170.5 KB	360压缩
javassist-3.18.1-GA.jar	697.5 KB	649.2 KB	360压缩
jboss-logging-3.3.0.Final.jar	65.2 KB	58.0 KB	360压缩

















名称	压缩前	压缩后	类型
.. (上级目录)			文件夹
hibernate-entitymanager-5.0.7.Final.jar	584.1 KB	521.5 KB	360压缩

因为使用hibernate的时候，有日志信息输出，hibernate本身没有支持日志输出的jar包，导入其他日志的jar包。不要忘记还有MySQL驱动的jar包。



log4j-1.2.16.jar
mysql-connector-java-5.0.4-bin.jar
slf4j-api-1.6.1.jar
slf4j-log4j12-1.7.2.jar

所有需要的jar如下：

 antlr-2.7.7.jar	2016-06-08 17:53	360压缩	435 KB
 dom4j-1.6.1.jar	2016-06-08 17:53	360压缩	307 KB
 geronimo-jta_1.1_spec-1.1.1.jar	2016-06-08 17:53	360压缩	16 KB
 hibernate-commons-annotations-5.0....	2016-06-08 17:53	360压缩	74 KB
 hibernate-core-5.0.7.Final.jar	2016-06-08 17:53	360压缩	5,453 KB
 hibernate-entitymanager-5.0.7.Final.jar	2016-01-13 12:39	360压缩	585 KB
 hibernate-jpa-2.1-api-1.0.0.Final.jar	2016-06-08 17:53	360压缩	111 KB
 jandex-2.0.0.Final.jar	2016-06-08 17:53	360压缩	184 KB
 javassist-3.18.1-GA.jar	2016-06-08 17:53	360压缩	698 KB
 jboss-logging-3.3.0.Final.jar	2016-06-08 17:53	360压缩	66 KB
 log4j-1.2.16.jar	2016-06-08 17:55	360压缩	471 KB
 mysql-connector-java-5.0.4-bin.jar	2016-06-08 17:53	360压缩	485 KB
 slf4j-api-1.6.1.jar	2016-06-08 17:55	360压缩	25 KB
 slf4j-log4j12-1.7.2.jar	2016-06-08 17:55	360压缩	9 KB

第二步：创建实体类

```
public class User {
    private int uid;
    private String username;
    private String password;
    private String address;
    public int getUid() {
        return uid;
    }
    public void setUid(int uid) {
        this.uid = uid;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getAddress() {
```

```

        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
}

```

2-1. 使用hibernate时候，不需要自己手动创建表，hibernate帮我们把表创建。前提是我们要在配置文件中进行配置。

第三步：配置实体类和数据库表的一一对应关系（映射关系）

3-1. 使用配置文件实现映射关系，创建xml格式的配置文件，映射配置文件的名称和位置没有固定要求。建议：在实体类所在的包里面创建， User.hbm.xml。

3-2. 配置是xml格式，在配置文件中首先引入xml约束， dtd约束。

```

<?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">

```

3-3. 配置映射关系。

```

<hibernate-mapping>
    <!-- 1 配置类和表对应
        class标签
        name属性：实体类全路径
        table属性：数据库表名称
    -->
    <class name="cn.itcast.entity.User" table="t_user">
    <!-- 2 配置实体类id和表id对应
        hibernate要求实体类有一个属性唯一值
        hibernate要求表有字段作为唯一值
    -->
    <!-- id标签
        name属性：实体类里面id属性名称
        column属性：生成的表字段名称
    -->
        <id name="uid" column="uid">
    <!-- 设置数据库表id增长策略
        native:生成表id值就是主键自动增长
    -->

```

```

        <generator class="uuid"></generator>
    </id>
    <!-- 配置其他属性和表字段对应
        name属性：实体类属性名称
        column属性：生成表字段名称
    -->
    <property name="username" column="username"></property>
    <property name="password" column="password"></property>
    <property name="address" column="address"></property>
</class>
</hibernate-mapping>

```

第四步：创建hibernate核心配置文件

4-1. 核心配置文件格式xml，但是核心配置文件名称和位置固定的

- 位置：必须在src下面
- 名称：hibernate.cfg.xml

4-2. 引入dtd约束

```

<?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">

```

4-3. hibernate操作过程中，只会加载核心配置文件，其他配置文件不会加载

```

<hibernate-configuration>
    <session-factory>
        <!-- 第一部分： 配置数据库信息 必须的 -->
        <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">sorry</property>
        <!-- 第二部分： 配置hibernate信息 可选的-->
        <!-- 输出底层sql语句 -->
        <property name="hibernate.show_sql">true</property>
        <!-- 输出底层sql语句格式 -->
        <property name="hibernate.format_sql">true</property>
        <!-- hibernate帮创建表，需要配置之后
            update: 如果已经有表，更新，如果没有，创建

```

```

-->
<property name="hibernate.hbm2ddl.auto">update</property>
<!-- 配置数据库方言
    在mysql里面实现分页 关键字 limit, 只能使用mysql里面
    在oracle数据库, 实现分页rownum
    让hibernate框架识别不同数据库的自己特有的语句
-->
<property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
<!-- 第三部分: 把映射文件放到核心配置文件中 必须的-->
<mapping resource="cn/itcast/entity/User.hbm.xml"/>
</session-factory>
</hibernate-configuration>

```

6. 实现添加操作

第一步：加载hibernate核心配置文件

第二步：创建SessionFactory对象

第三步：使用SessionFactory创建session对象

第四步：开启事务

第五步：写具体逻辑crud操作

第六步：提交事务

第七步：关闭资源

看到效果：（1）是否生成表
（2）看表是否有记录

@Test

```

public void testAdd() {
    //第一步 加载hibernate核心配置文件, 到src下面找到名称是hibernate.cfg.xml
    Configuration cfg = new Configuration();
    cfg.configure();
    //第二步 创建SessionFactory对象, 读取hibernate核心配置文件内容, 创建
    sessionFactory, 在过程中, 根据映射关系, 在配置数据库里面把表创建
    SessionFactory sessionFactory = cfg.buildSessionFactory();
    //第三步 使用SessionFactory创建session对象, 类似于jdbc中的连接对象conn

```



```

Session session = sessionFactory.openSession();
//第四步 开启事务
Transaction tx = session.beginTransaction();
//第五步 写具体逻辑 crud操作, 添加功能
User user = new User();
user.setUsername("小王");
user.setPassword("250");
user.setAddress("日本");
session.save(user);
//第六步 提交事务
tx.commit();
//第七步 关闭资源
session.close();
sessionFactory.close();
}

```

Create Table

```

CREATE TABLE `t_user` (
  `uid` int(11) NOT NULL AUTO INCREMENT,
  `username` varchar(255) DEFAULT NULL,
  `password` varchar(255) DEFAULT NULL,
  `address` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`uid`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8

```

	uid	username	password	address
<input type="checkbox"/>	1	小王	250	日本
★	(NULL)	(NULL)	(NULL)	(NULL)

7. hibernate 配置文件详解

1. Hibernate映射配置文件

1. 映射配置文件名称和位置没有固定要求
2. 映射配置文件中, 标签name属性值写实体类相关内容
 - 2-1. class标签的name属性值实体类全路径
 - 2-2. id标签和property标签name属性值 实体类属性名称
3. id标签和property标签的column属性可以省略的

3-1. 不写值默认和name属性值一样的

4. property标签type属性，设置生成表字段的类型，自动对应类型（忽略）

2. Hibernate核心配置文件

2-1. 配置写位置要求

```
<hibernate-configuration>
    <session-factory>
        .....
    </session-factory>
</hibernate-configuration>
```

2-2. 配置三部分要求

- (1) 数据库部分必须的
- (2) hibernate部分可选的
- (3) 映射文件必须的

2-3. 核心配置文件名称和位置固定的

- (1) 位置：src目录下
- (2) 名称：hibernate.cfg.xml

8. Hibernate核心API

1. Configuration

1-1. 到src下面找到名称hibernate.cfg.xml配置文件，创建对象，把配置文件放到对象里面（加载核心配置文件）

```
Configuration cfg = new Configuration();
cfg.configure();
```

2. SessionFactory（重点）

2-1. 使用configuration对象创建sessionFactory对象

(1) 创建sessionfactory过程中做事情：根据核心配置文件中，有数据库配置，有映射文件部分，到数据库里面根据映射关系把表创建。

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

2-2. 创建sessionFactory过程中，这个过程特别耗资源的

- (1) 在hibernate操作中，建议一个项目一般创建一个sessionFactory对象

2-3. 具体实现

(1) 写工具类，写静态代码块实现

* 静态代码块在类加载时候执行，执行一次

```
static Configuration cfg = null;
static SessionFactory sessionFactory = null;
//静态代码块实现
static {
    //加载核心配置文件
    cfg = new Configuration();
    cfg.configure();
    sessionFactory = cfg.buildSessionFactory();
}

//提供方法返回sessionFactory
public static SessionFactory getSessionFactory() {
    return sessionFactory;
}
```

3. Session (重点)

3-1. session类似于jdbc中connection

3-2. 调用session里面不同的方法实现crud操作

- (1) 添加 save方法
- (2) 修改 update方法
- (3) 删除 delete方法
- (4) 根据id查询 get方法

3-3. session对象单线程对象

- (1) session对象不能共用，只能自己使用

4. Transaction

4-1. 事务对象

Transaction tx = session.beginTransaction;

4-2. 事务提交和回滚方法

```
tx.commit();
tx.rollback();
```

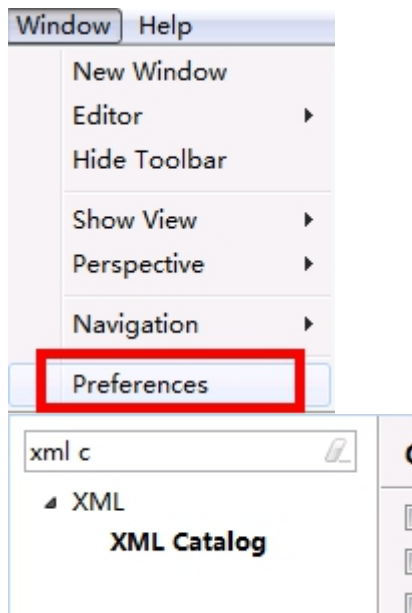
4-3. 事务概念

- (1) 事务四个特性：原子性、一致性、隔离性、持久性

9. 解决配置文件没有提示问题

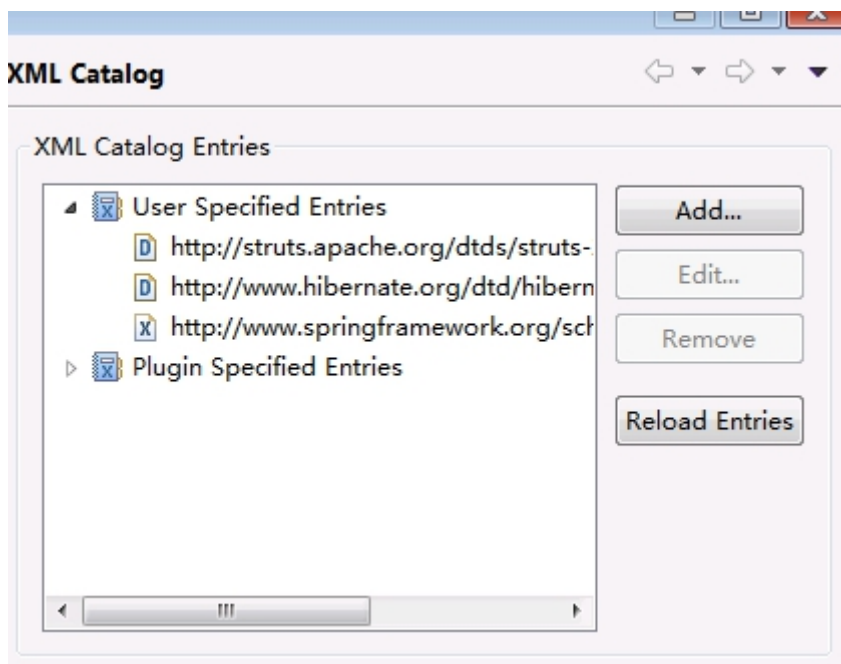
1. 可以上网

2. 把约束文件引入到eclipse中



(1) 在配置文件中复制一句话

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



Location: D:\资料\2-hibernate的dtd文件\hibernate-mapping-3.0\dtd

Workspace... File System...

Key type: URI

Key: http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd

☐ Alternative web address:

重启eclipse开发工具