目录

[概述 1](#_Toc56429124)

[常规使用 1](#_Toc56429125)

[注解 2](#_Toc56429126)

[带参数的CRUD 3](#_Toc56429127)

[插入insert 3](#_Toc56429128)

[查询 4](#_Toc56429129)

[一对多查询 4](#_Toc56429130)

[多对多 5](#_Toc56429131)

# 概述

mybatis 是一个优秀的基于 java 的持久层框架，它内部封装了 jdbc，使开发者只需要关注 sql 语句本身，而不需要花费精力去处理加载驱动、创建连接、创建 statement 等繁杂的过程。

mybatis 通过 xml 或注解的方式将要执行的各种 statement 配置起来，并通过 java 对象和 statement 中sql 的动态参数进行映射生成最终执行的 sql 语句，最后由 mybatis 框架执行 sql 并将结果映射为 java 对象并返回。

泛型一定要先申明再使用



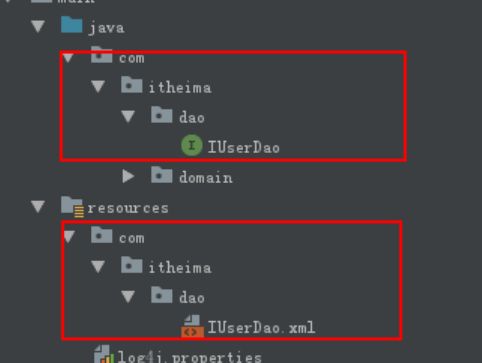
## 常规使用

通过快速入门示例，我们发现使用 mybatis 是非常容易的一件事情，因为只需要编写 Dao 接口并且按照 mybatis 要求编写两个配置文件，就可以实现功能。

配置文件一是**SqlMapConfig.xml**是关于数据名称账号密码的配置，链接方式等配置



配置文件二是 resources文件夹下面的同路径同名文件，select中配置的id是接口的方法名称，resultType是接口文件的路径

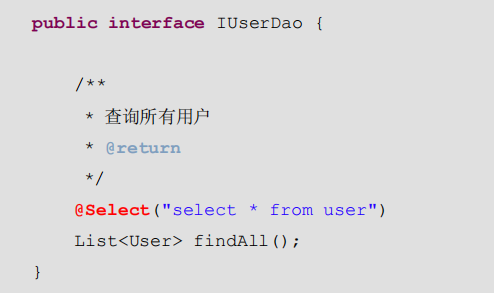


测试类的写法，配置文件的读取和实现类的读取



## 注解

使用注解的方法编写测试类，可以省略在resources下同路径编写同名类的步骤

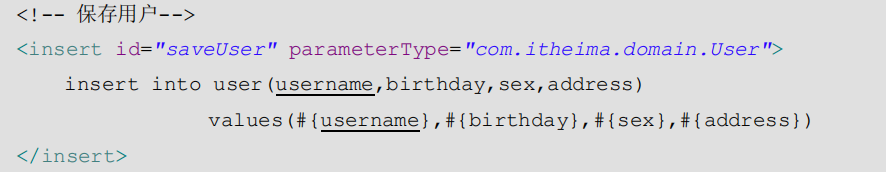




### 入门案例

## 带参数的CRUD

### 插入insert



parameterType 属性：

代表参数的类型，因为我们要传入的是一个类的对象，所以类型就写类的全名称。

sql 语句中使用#{}字符：

它代表占位符，相当于原来 jdbc 部分所学的?，都是用于执行语句时替换实际的数据。

具体的数据是由#{}里面的内容决定的。

#{}中内容的写法：

由于我们保存方法的参数是 一个 User 对象，此处要写 User 对象中的属性名称。

它用的是 ognl 表达式。

ognl 表达式：

它是 apache 提供的一种表达式语言，全称是：

Object Graphic Navigation Language 对象图导航语言

它是按照一定的语法格式来获取数据的。

语法格式就是使用 #{对象.对象}的方式

#{user.username}它会先 去找 user 对象，然后在 user 对象中找到 username 属性，并调用

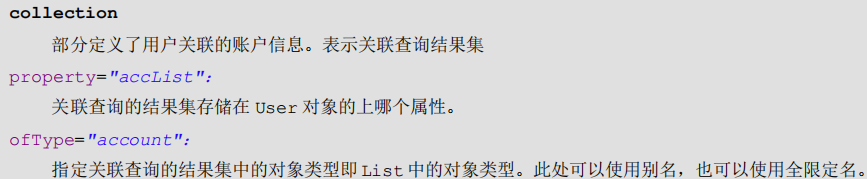
getUsername()方法把值取出来。但是我们在 parameterType 属性上指定了实体类名称，所以可以省略 user.

而直接写 username。

# 查询

## 一对多查询

在同名的映射文件.xml中配置collection属性



/\*\*

\*

\* <p>Title: MybastisCRUDTest</p>

\* <p>Description: 一对多的操作</p>

\* <p>Company: http://www.itheima.com/ </p>

\*/

**public class** UserTest {

**private** InputStream in ;

**private** SqlSessionFactory factory;

**private** SqlSession session;

**private** IUserDao userDao;

@Test

**public void** testFindAll() {

//6.执行操作

List<User> users = userDao.findAll();

**for**(User user : users) {

System.***out***.println("-------每个用户的内容---------");

System.***out***.println(user);

System.***out***.println(user.getAccounts());

}

}

@Before//在测试方法执行之前执行

**public void** init()**throws** Exception {

//1.读取配置文件

in = Resources.*getResourceAsStream*("SqlMapConfig.xml");

//2.创建构建者对象

SqlSessionFactoryBuilder builder = **new** SqlSessionFactoryBuilder();

//3.创建 SqlSession 工厂对象

factory = builder.build(in);

//4.创建 SqlSession 对象

session = factory.openSession();

//5.创建 Dao 的代理对象

userDao = session.getMapper(IUserDao.**class**);

}

@After//在测试方法执行完成之后执行

**public void** destroy() **throws** Exception{

session.commit();

//7.释放资源

session.close();

in.close();

}

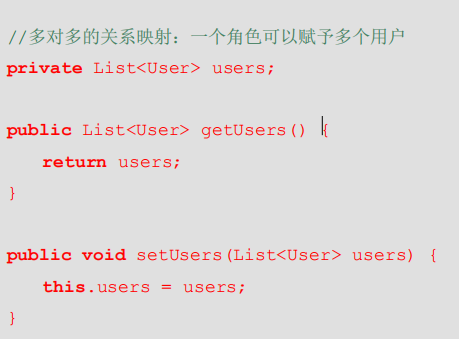
}

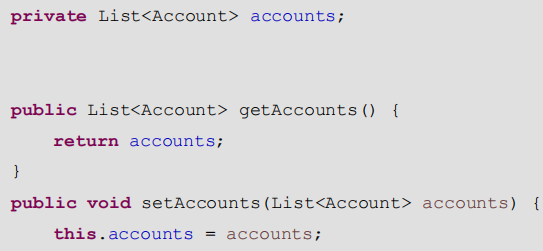
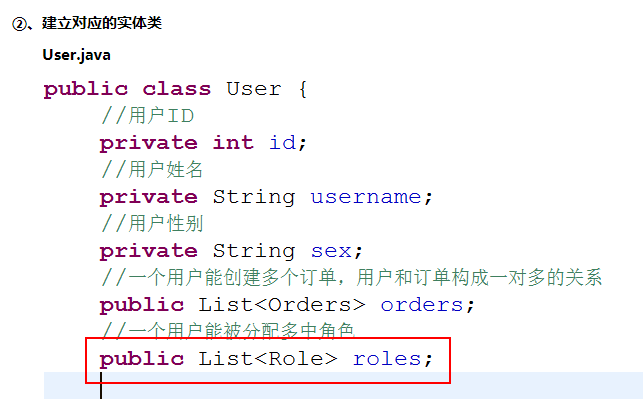
@after @before会自动在执行该类方法的时候执行，运行测试方法的时候，直接选中类中该方法的部分运行就可以了？

## 多对多

角色实体类

一对一、多对多、一对多最核心的部分在于 下图中的private List<User> users中的list<User>对应一个List集合的User对象 ，多对多的实现类里面有着两个LIST类型的属性用来存放两个多





同路径下的配置文件基本没区别，区别在实现类的写法上

