MyBatis 是一款优秀的**持久层**框架;

它支持自定义 SQL、存储过程以及高级映射(方便我们写sql)。

MyBatis 免除了几乎所有的 JDBC 代码以及设置参数和获取结果集的工作。

MyBatis 可以通过简单的 XML 或注解来配置和映射原始类型、接口和 Java POJO（Plain Old Java Objects，普通老式 Java 对象）为数据库中的记录。

环境准备：

第一步构建，

创建maven项目，pom.xml里添加dependencies,一个mysql driver，一个mybatis，一个JUnit ,.删除src，这样父模块的pom.xml就可以被子模块重复使用



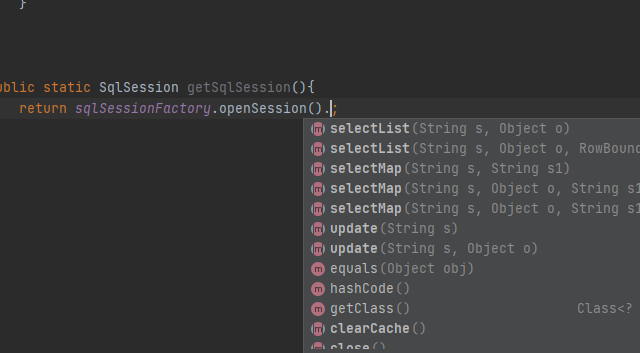
第二步util

原来JDBC里的util就是读取properties里的driver url啥的，现在一样，不过是到了resources里的mybatis-config.xml文件

SqlSessionFactoryBuilder.build(inputStream)->sqlSessionFactory->sqlSession

同时大幅简化，我们只需要在静态模块里面加载mybatis-config里的数据并创造一个**sqlSessionFactory,**然后再用sqlSessionFactory 创建一个**SqlSession**对象，//他就相当于原来的Connection， SqlSession包含了面向数据库执行sql语句的所有方法

//sqlSession是为了后面实例化Interface Mapper



sqlSession可以update Select啥的

public class MybatisUtils {  
  
 private static SqlSessionFactory sqlSessionFactory;  
 static {  
  
 try {  
 String resource="mybatis-config.xml";  
 InputStream inputStream = Resources.getResourceAsStream(resource);  
 sqlSessionFactory = new SqlSessionFactoryBuilder().build(inputStream);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
  
 }  
  
 public static SqlSession getSqlSession(){  
 return sqlSessionFactory.openSession();  
 }  
}

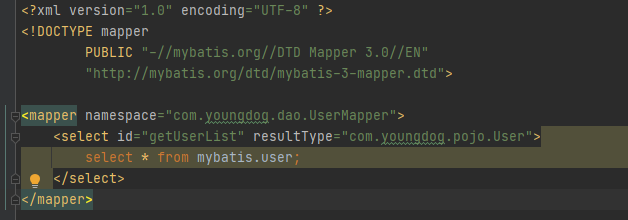
=========================

3.pojo实体类，他不负责业务,只是为了UserMapper.xml告诉你return啥，UserMapper告诉你啥方法

4.Mapper 接口以及对应配置文件，重头戏

一个interface和一个.xml配置文件，

配置文件代替了原来的java implement文件，并要在mybatis-config.xml里登记



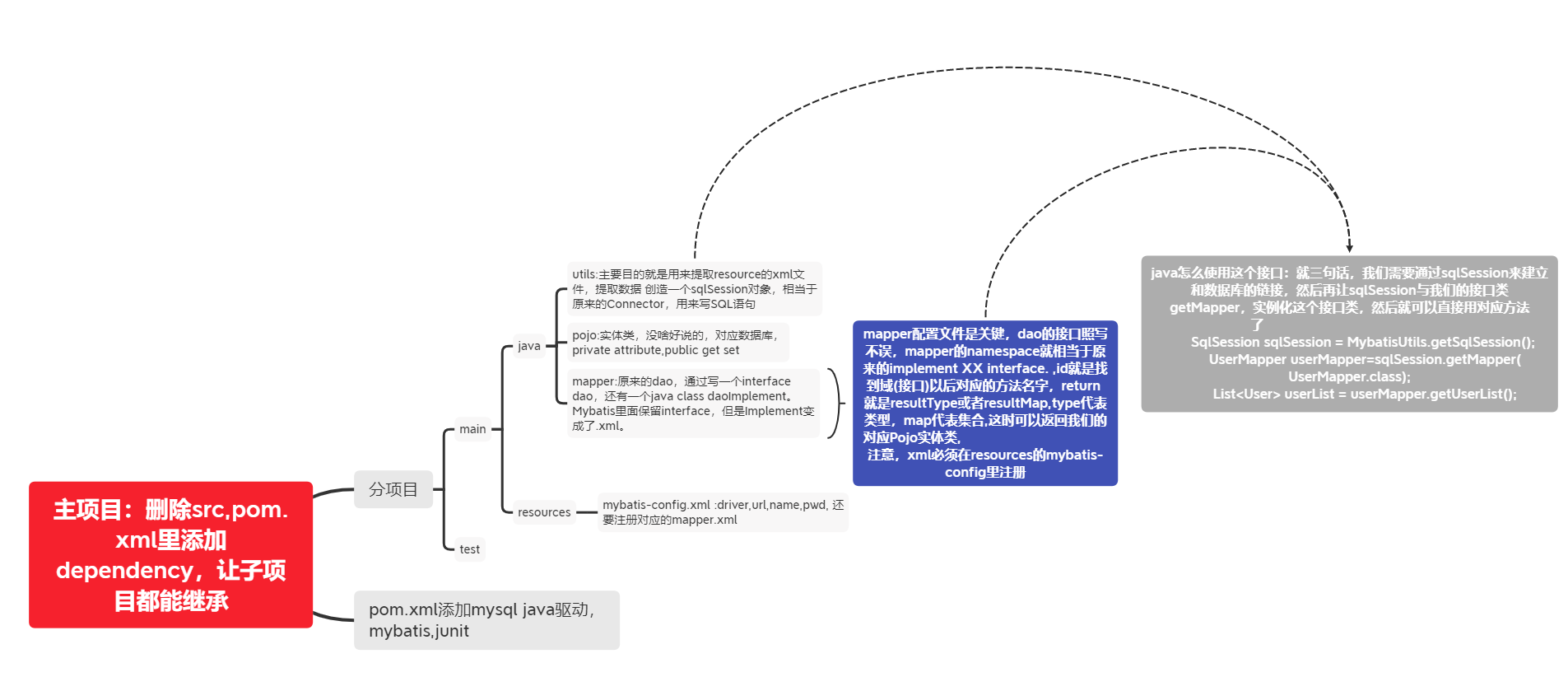
namespace是要对应的interface

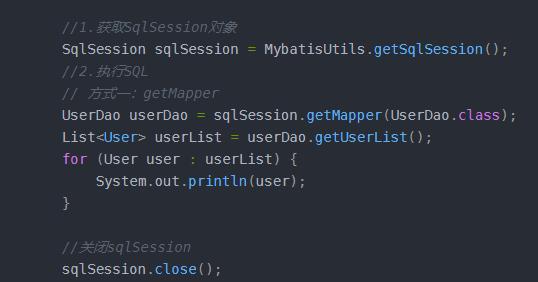
id是对应method，

实装了各种接口

5.运行，先得到sqlSession对象，并且通过他的getMapper来实例化对应的InterFace,

返回的接口对象就可以直接用了，然后就是直接面向对象编程了//类似动态代理，根据接口和对应的.xml配置文件，生成一个对应的impl





sqlSession最后一定要关闭，

**CRUD**

在架设好utils,pojo,resources里的mybatis-config.xml里以后，我们只需要在Mapper里添加接口，UserMapper.xml实装接口就行，使用就是测试类，只要改这三个

三步：

Mapper添加对应接口方法，Select多条需要return List,一条就return 单个，其他的return int

Mapper.xml实现·接口，先要用namespace找到对应interface,

**Select:**

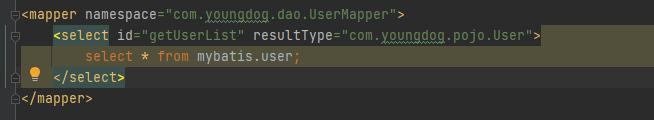
id：对应接口名字，

resultType：对应实体类

parameterType: 如果是限定查询可能需要参数，参数的类

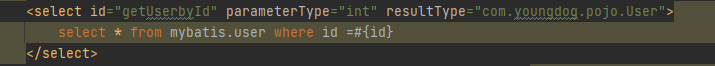
return多个就是List，return一个条目就是单个







他通过id select到，因此只return一个条目，所以是User



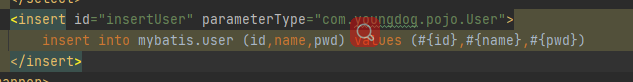
parameter Type就是跟着接口的parameter一路写下去

引用接口的parameter就是#{},

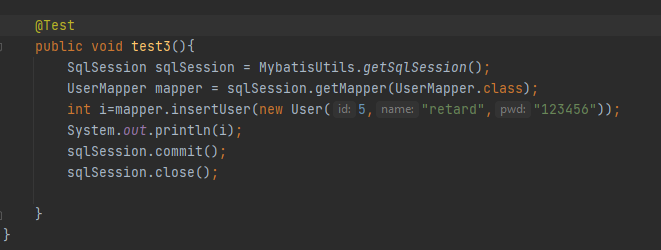
**Insert**

增删改 有一个特殊点，他们return的都是int,如果return 1，代表这个操作成功没有语法上的错误，如果return 0代表语法有问题

 return类型为int



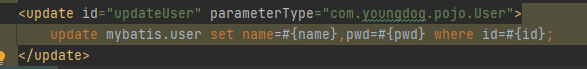
Mapper接口是public的，所以可以直接引用user的id，name,pwd//引用object的属性可以直接提取出来

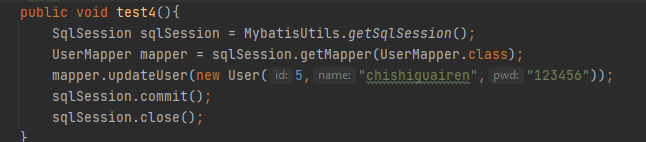


增删改还必须要commit

**Update**

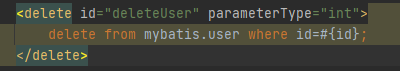
****

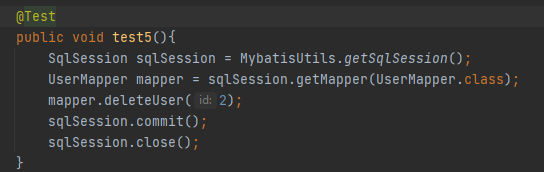
****

****

**Delete**

****

****

****

**Map**

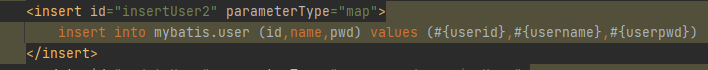
**两种用法**

1.有的时候pojo类/数据库特长特复杂，我们可以用Map作为参数，

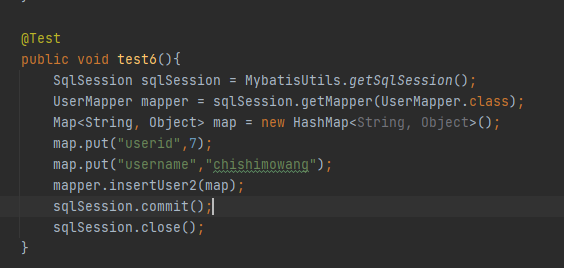


key是String,用来手写字段

Object是因为String,Integer都能作为Object

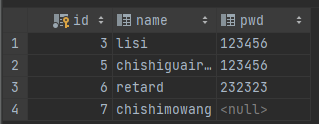


userid,username,userpwd就是 key



可以看出我们并没有put pwd，

也就是说他把pwd设置成了null,如果实体类/数据库字段参数过多，这样可以少填很多可以为Null或者默认的数据



2.比如说select,我们同时搜索两个int类型的参数，这时可以使用map,



例如这里getUserbyId, 如果id是可以重复的，我们需要另外一个int子id才能确认，这时可以用map

总结：

Map传递参数，直接在sql中取出key即可！ 【parameter=“map”】

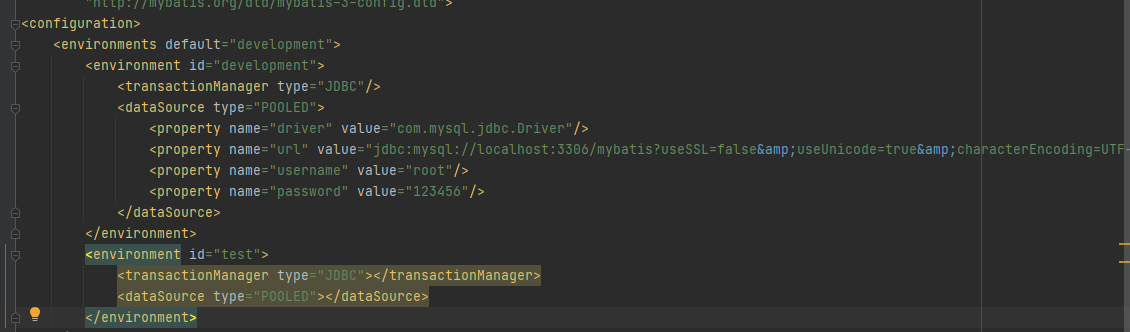
对象传递参数，直接在sql中取出对象的属性即可！ 【parameter=“Object”】 //不用管private

只有一个基本类型参数的情况下，可以直接在sql中取到

多个参数用Map , **或者注解！**

配置文件解析

1.environment



environment改个id就能有多个environment,默认事务管理器是JDBC,还有一个managed，不常用

、默认dataSource是POOLED, WEB响应更快

要学会的就是**：尽管可以配置多个环境，但每个 SqlSessionFactory 实例只能选择一种环境**

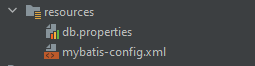
学会使用配置多套运行环境！

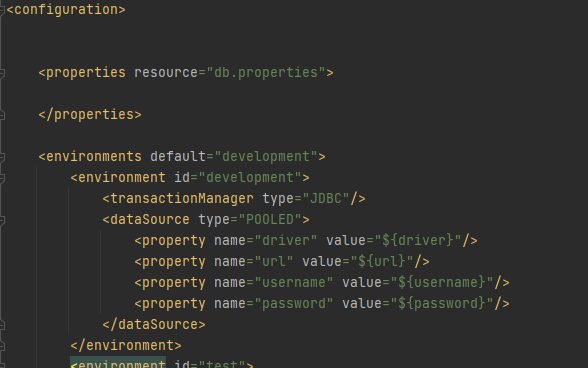
MyBatis默认的事务管理器就是JDBC ，连接池：POOLED

2.properties

这些属性可以在外部进行配置，并可以进行动态替换。你既可以在典型的 Java 属性文件中配置这些属性，也可以在 properties 元素的子元素中设置。【db.poperties】

换句话来说，现在可以用db.properties,然后environment里再去引用



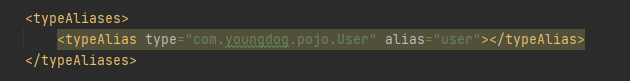


3.typeAliases

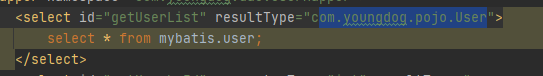
主要目的就是为类（尤其是pojo类）设置一个别名，这个别名仅用于.xml配置（例如Mapper的xml）

3.1

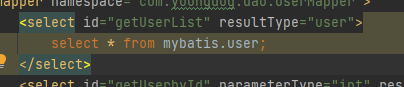
typeAlias



原来

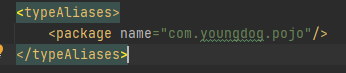


这样xml实装文件我们可以简写pojo类user



3.2

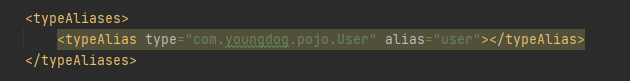
package



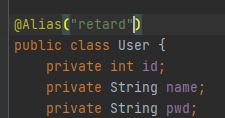
指定一个package，然后这个package目录下所有class将会有一个别名，这个别名默认为小写文件名字，例如



这时alias默认为user

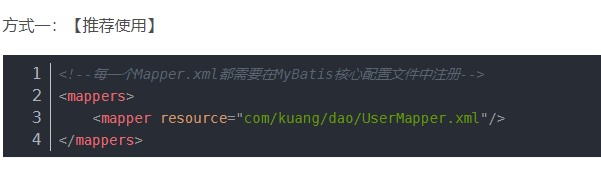


如果想改变默认，要使用Alias



4.映射器Mappers

注册绑定我们所写的mapper文件



方法二：



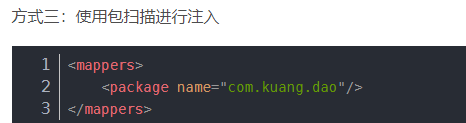
class的缺陷

1.接口和配置文件必须同名

比如接口叫UserMapper,那么配置文件就必须交UserMapper.xml

2.接口与配置文件必须在一个包下

方法三：



这个包下Mapper都被注册

还是老问题，接口和配置文件必须同名