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



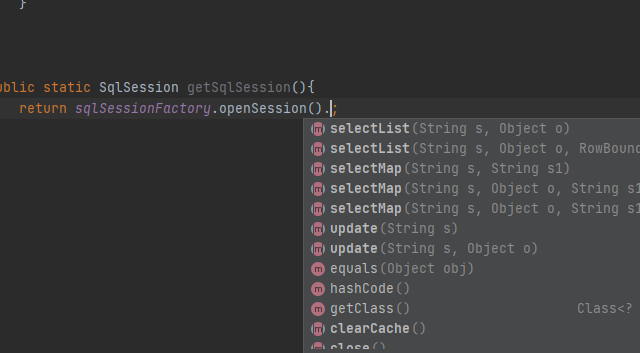
第二步util

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();  
 }  
}

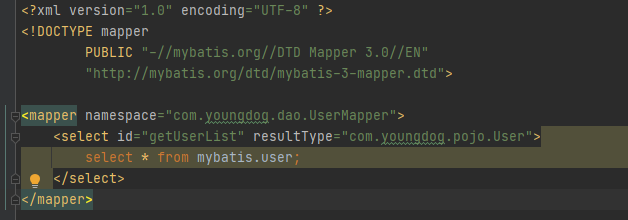
=========================

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



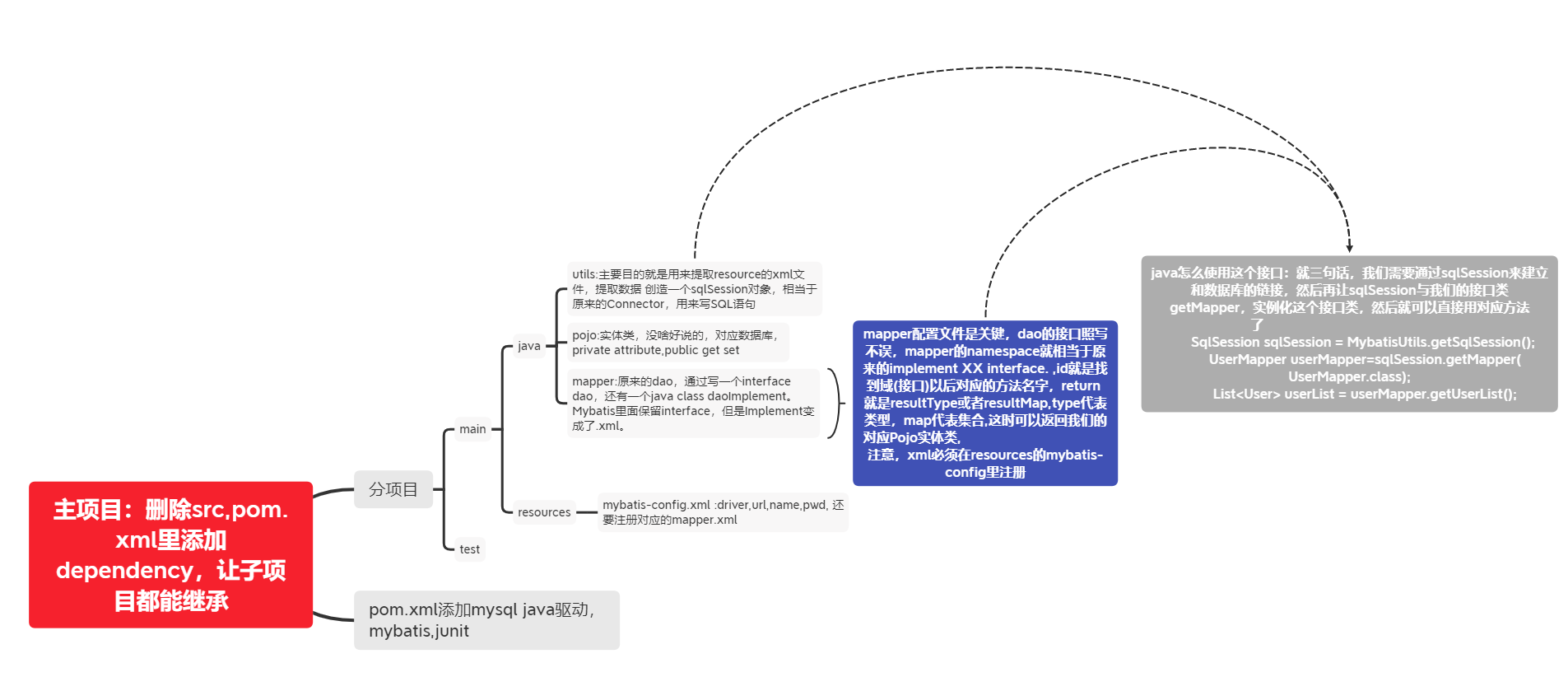
namespace是要对应的interface

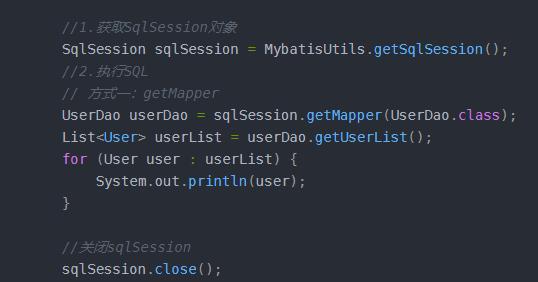
id是对应method，

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，