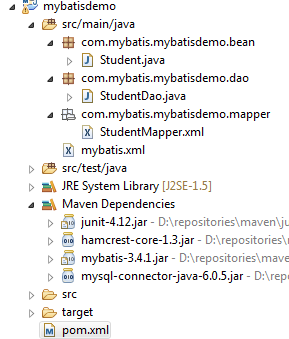
# Mybatis

## 第一个Mybatis小工程

1. **目录结构如下：**



1. **Maven依赖（本工程为Maven工程，如不是Maven工程则需手动导入包）**

**<!-- Maven中pom.xml的MyBatis依赖配置 -->**

<dependency>

    <groupId>org.mybatis</groupId>

            <artifactId>mybatis</artifactId>

            <version>4.3.1</version>

</dependency>

**<!-- Maven中pom.xml的Sql依赖配置 -->**

<dependency>

           <groupId>mysql</groupId>

           <artifactId>mysql-connector-java</artifactId>

           <version>6.0.5</version>

</dependency>

**Pom.xml文件如下：**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.mybatis</groupId>

<artifactId>mybatisdemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>mybatisdemo</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<!-- Maven中pom.xml的MyBatis依赖配置 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>3.4.1</version>

</dependency>

<!-- Maven中pom.xml的Sql依赖配置 -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>6.0.5</version>

</dependency>

</dependencies>

</project>

1. **创建数据库以及数据表并插入示例数据，这里假设创建的是mybatis库和students表，我使用的是MySql数据库。**

-- 创建mybatis数据库

create database anxin

-- 切换到mybatis数据库

use anxin

-- 创建students表

create table students(id int auto\_increment primary key, name varchar(50));

-- 插入示例数据

insert into students(name) values('kolbe');

1. **新建MyBatis的核心配置文件（包括数据源信息、事务管理器、映射配置等），建议将该文件放置类源文件根目录下，我直接把它取名为mybatis.xml，具体配置如下：**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

<environments default="development">

<environment id="development">

<!-- 事务管理类型，JDBC表示直接使用JDBC的提交和回滚设置，依赖于数据源得到的连接来管理事务 -->

<transactionManager type="JDBC" />

<!-- 数据库连接池POOLED表示使用数据库连接池 -->

<dataSource type="POOLED">

<property name="driver" value="com.mysql.cj.jdbc.Driver" />

<property name="url" value="jdbc:mysql://127.0.0.1:3306/anxin" />

<property name="username" value="root" />

<property name="password" value="281215" />

</dataSource>

</environment>

</environments>

<mappers>

<!-- SQL代码和映射信息配置文件 -->

<mapper resource="com/mybatis/mybatisdemo/mapper/StudentMapper.xml" />

</mappers>

</configuration>

1. **新建一个名为Student的Java Bean类.**

package com.mybatis.mybatisdemo.bean;

public class Student {

private int id;

private String name;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "Student [id="+id+",name="+name+"]";

}

}

1. **新建SQL语句映射文件StudentMapper.xml**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<!-- namespace必须是唯一的，建议用该文件所在的包路径全类名 -->

<mapper namespace="com.mybatis.mybatisdemo.mapper.StudentMapper">

<!-- select：表示这是一个查询操作 -->

<!-- id：用来配合namespace来定位这个操作，也就是在调用时，将采用ccom.mybatis.mybatisdemo.mapper.StudentMapper.getStudent -->

<!-- resultType：表示返回值类型，必须是全类名，MyBatis将自动为我们将查询结果封装成Student对象 -->

<!-- parameterType：表示传进来的参数的类型，因为传的是id主键，所以是int型 -->

<select id="getStudent" parameterType="int" resultType="com.mybatis.mybatisdemo.bean.Student">

<!-- #{id}：表示占位符，也就是调用时必须为其传一个id参数进来，注意是#不是$ -->

select \* from students where id = #{id}

</select>

</mapper>

1. 新建一个测试类StudentDao，来测试配置的效果

package com.mybatis.mybatisdemo.dao;

import java.io.InputStream;

import org.apache.ibatis.io.Resources;

import org.apache.ibatis.session.SqlSession;

import org.apache.ibatis.session.SqlSessionFactory;

import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import org.junit.Test;

import com.mybatis.mybatisdemo.bean.Student;

public class StudentDao {

@Test

public void testGetStudent()throws Exception{

//1.加载Mybatis配置文件：mybatis.xml(它也关联mappers结点下的映射文件)

InputStream inputStream = Resources.getResourceAsStream("mybatis.xml");

//2.SqlSessionFactoryBuidler实例通过输入流调用buid方法构建SqlSession工厂

SqlSessionFactory sqlSessionFactory = new SqlSessionFactoryBuilder().build(inputStream);

//3.通过工厂获取SqlSession实例，SqlSession完全包含了面向数据库执行SQL命令所需的所有方法

SqlSession session = sqlSessionFactory.openSession();

//4.准备基本信息

//4.1 statement：用来定位映射文件（StudentMapper.xml）中的语句（通过namespace id+select id）

String statement = "com.mybatis.mybatisdemo.mapper.StudentMapper.getStudent";

//4.2 parameter：传进去的参数，也就是需要获取students表中主键值为1的记录

int paramter = 3;

//5.SqlSession实例来直接执行已映射的SQL语句，selectOne表示获取的是一条记录

Student student = session.selectOne(statement,paramter);

System.out.println(student);

//6.关闭输入流和SqlSession实例

inputStream.close();

session.close();

}

}