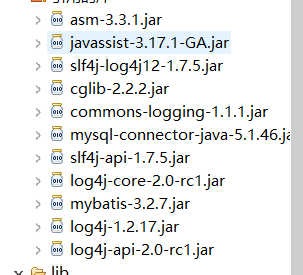
Mybatis操作数据库步骤

第一步：

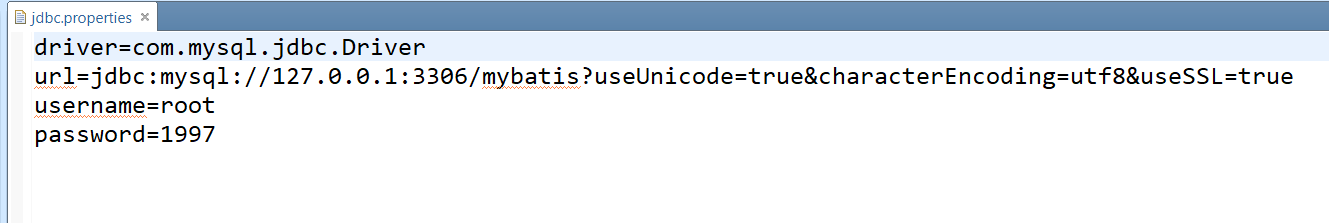
1.1加入mybatis核心包、依赖包、数据驱动包等11个包



1.2：创建源文件夹config并在下面创建mybatis-config.xml

第二步：配置mybatis-config.xml

1.1配置数据库详情文件



1.2配置mybatis-config.xml

<?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>

<!—引入数据库详情配置-->

<properties resource="jdbc.properties"></properties>

<typeAliases>

<!-- 自己取别名 -->

<!-- <typeAlias type="com.yb.entity.User" alias="User"/>-->

<!-- 自动取别名，别名为类名 -->

<package name="com.yb.entity"/>

</typeAliases>

<!-- 和spring整合后 environments配置将废除 -->

<environments default=*"development"*>

<environment id=*"development"*>

<!-- 使用jdbc事务管理 -->

<transactionManager type=*"JDBC"* />

<!-- 数据库连接池 -->

<dataSource type=*"POOLED"*>

<property name="driver" value="${driver}"/>

<property name="url" value="${url}"/>

<property name="username" value="${username}"/>

<property name="password" value="${password}"/> </dataSource>

</environment>

</environments>

<mappers>

<!—加载sql映射文件 -->

<mapper resource="sqlmap/Fwjd.xml"/>

<package name="com.yb.mapper"/>

</mappers>

</configuration>

第三步：配置sql映射文件

3.1创建对应表的实体类

3.2配置sql映射

<?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：命名空间，用于隔离sql，还有一个很重要的作用，后面会讲 -->

<mapper namespace=*"test"*>

<!-- id:statement的id 或者叫做sql的id-->

<!-- parameterType:声明输入参数的类型 -->

<!-- resultType:声明输出结果的类型，应该填写pojo的全路径 -->

<!-- #{}：输入参数的占位符，相当于jdbc的？ -->

<select id=*"queryUserById"* parameterType=*"int"*

resultType=*"cn.itcast.mybatis.pojo.User"*>

SELECT \* FROM `user` WHERE id = #{id}

</select>

</mapper>

第四步：测试程序

**public** **class** MybatisTest {

**private** SqlSessionFactory sqlSessionFactory = **null**;

@Before

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

// 1. 创建SqlSessionFactoryBuilder对象

SqlSessionFactoryBuilder sqlSessionFactoryBuilder = **new** SqlSessionFactoryBuilder();

// 2. 加载SqlMapConfig.xml配置文件

InputStream inputStream = Resources.*getResourceAsStream*("SqlMapConfig.xml");

// 3. 创建SqlSessionFactory对象

**this**.sqlSessionFactory = sqlSessionFactoryBuilder.build(inputStream);

}

@Test

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

// 4. 创建SqlSession对象

SqlSession sqlSession = sqlSessionFactory.openSession();

// 5. 执行SqlSession对象执行查询，获取结果User

// 第一个参数是User.xml的statement的id，第二个参数是执行sql需要的参数；

Object user = sqlSession.selectOne("queryUserById", 1);

// 6. 打印结果

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

// 7. 释放资源

sqlSession.close();

}

}