# Java Mybatis配置

1. 公共数据源见JdbcTemplate配置第一点
2. 在pom文件中添加依赖, 以及对应数据库驱动jar包

<properties>

<mybatis.version>3.4.0</mybatis.version>

</properties>

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>${mybatis.version}</version>

</dependency>

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis-spring</artifactId>

<version>1.3.0</version>

</dependency>

1. 在resources包下创建mybatis的spring bean配置文件Mybatis-beans.xml, 内容如下:

<import resource="classpath\*:/common-beans.xml" />

<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">

<property name="configLocation" value="classpath:beans/Mybatis-config.xml" />

<property name="dataSource">

<ref bean="dataSource"></ref>

</property>

</bean>

其中添加mybatis配置文件Mybatis-config.xml， 内容如下(注意，这个配置文件不是bean文件)

<?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>

</configuration>

1. 新建实体类User, 内容如下

package entities;

/\*\*

\* @author gacl

\* users表所对应的实体类

\*/

public class User {

//实体类的属性和表的字段名称一一对应

private int id;

private String name;

private int age;

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;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@Override

public String toString() {

return "User [id=" + id + ", name=" + name + ", age=" + age + "]";

}

}

1. 方法一(使用配置文件方式)
   1. 添加实体映射文件userMapper.xml, 如添加在entitiies/mapping包下面, 内容如下

<?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">

<!-- 为这个mapper指定一个唯一的namespace，namespace的值习惯上设置成包名+sql映射文件名，这样就能够保证namespace的值是唯一的

例如namespace="mappings.userMapper"就是mappings(包名)+userMapper(userMapper.xml文件去除后缀) -->

<mapper namespace="mappings.userMapper">

<!-- 在select标签中编写查询的SQL语句， 设置select标签的id属性为getUser，id属性值必须是唯一的，不能够重复 使用parameterType属性指明查询时使用的参数类型，resultType属性指明查询返回的结果集类型

resultType="entities.User"就表示将查询结果封装成一个User类的对象返回 User类就是users表所对应的实体类 -->

<select id="getUser" parameterType="int" resultType="entities.User">

select \* from MyUser where UserId = #{id}

</select>

b) 将实体映射文件添加到mybatis配置文件mybatis-config.xml中configuration节点下添加

<mappers>

<!-- 注册userMapper.xml文件位于resources资源包下的mappings这个包下，所以resource写成mapping/userMapper.xml -->

<mapper resource="mappings/userMapper.xml" />

</mappers>

1. 方法二, 使用接口方式(此方式可以省略xml映射)
   1. 添加接口services.interf.IUserService, 实现如下:

package services.interf;

import org.apache.ibatis.annotations.Select;

import model.UserInfo;

public interface IUserService {

//@Select({"select Id, UserName, Age from UserInfo where id = #{id}"})

UserInfo getUserInfoById(int id);

}

* 1. (若上面使用Select注解, 则此步可以省略)在mapper文件中配置如下

<?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">

<mapper namespace="services.interf.IUserService">

<select id="getUserInfoById" parameterType="int" resultType="UserInfo">

<![CDATA[select \* from UserInfo where id = #{id}]]>

</select>

</mapper>

1. 调用代码执行

WebApplicationContext wac = ContextLoader.getCurrentWebApplicationContext();

SqlSessionFactory sessionFactory = (SqlSessionFactory)wac.getBean("sqlSessionFactory");

//SqlSessionFactory sessionFactory = new SqlSessionFactoryBuilder().build(is);

// Reader reader = Resources.getResourceAsReader(resource);

// SqlSessionFactory sessionFactory = new

// SqlSessionFactoryBuilder().build(reader);

SqlSession session = sessionFactory.openSession();

// 命名空间下的userMapper.xml文件下ID为getUser的节点信息

String statement = "mappings.userMapper.getUser";

// 查询user

User user = session.selectOne(statement, 1);

System.out.println(user);

方法二执行代码如下:

WebApplicationContext wac = ContextLoader.getCurrentWebApplicationContext();

SqlSessionFactory sessionFactory = (SqlSessionFactory) wac.getBean("sqlSessionFactory");

SqlSession session = sessionFactory.openSession();

Map map = new HashMap();

map.put("id", 1);

UserInfo userab = session.selectOne("services.interf.IUserService.justSql", map);

IUserService mapper = session.getMapper(IUserService.class);

UserInfo user = mapper.getUserInfoById(1);

# 其它配置