### 1、主配置文件 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>  <properties resource="jdbc.properties"/> //加载资源文件    <!-- <typeAliases>  <typeAlias alias="Student" type="com.java1234.model.Student"/>  </typeAliases> -->  <typeAliases>  <package name="com.java1234.model"/> //配置该包下面的所有实体bean的别名  </typeAliases>  <environments default="development">  <environment id="development">  <transactionManager type="JDBC" />  <dataSource type="POOLED">  <property name="driver" value="${jdbc.driverClassName}" />  <property name="url" value="${jdbc.url}" />  <property name="username" value="${jdbc.username}" />  <property name="password" value="${jdbc.password}" />  </dataSource>  </environment>  </environments>  <mappers>  <!-- <mapper resource="com/java1234/mappers/StudentMapper.xml" /> -->  <!-- <mapper class="com.java1234.mappers.StudentMapper"/> -->  <package name="com.java1234.mappers"/> //加载该包下面的所有映射  </mappers>  </configuration> |

**资源文件——别名——数据源——映射文件**

### 数据库配置文件 jdbc.properties

|  |
| --- |
| jdbc.driverClassName=com.mysql.jdbc.Driver  jdbc.url=jdbc:mysql://localhost:3306/db\_mybatis  jdbc.username=root  jdbc.password=123456 |

### 日志配置文件 log4j.properties

|  |
| --- |
| log4j.rootLogger=info,appender1,appender2  log4j.appender.appender1=org.apache.log4j.ConsoleAppender  log4j.appender.appender2=org.apache.log4j.FileAppender  log4j.appender.appender2.File=C:/logFile.txt    log4j.appender.appender1.layout=org.apache.log4j.TTCCLayout  log4j.appender.appender2.layout=org.apache.log4j.TTCCLayout |

### 接口配置文件(xml配置方式)

**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">  <mapper namespace=*"com.java1234.mappers.StudentMapper"*>  <resultMap type=*"Student"* id=*"StudentResult"*>  <id property=*"id"* column=*"id"*/>  <result property=*"name"* column=*"name"*/>  <result property=*"age"* column=*"age"*/>  <association property=*"address"* column=*"addressId"* select=*"com.java1234.mappers.AddressMapper.findById"*></association>  <association property=*"grade"* column=*"gradeId"* select=*"com.java1234.mappers.GradeMapper.findById"*></association>  </resultMap>    <select id=*"findStudentWithAddress"* resultMap=*"StudentResult"* parameterType=*"Integer"*>  select \* from t\_student t1,t\_address t2 where t1.addressId=t2.id and t1.id=#{id}  </select>    <select id=*"findByGradeId"* resultMap=*"StudentResult"* parameterType=*"Integer"*>  select \* from t\_student where gradeId=#{gradeId}  </select>  <insert id=*"add"* parameterType=*"Student"* > //使用了别名  insert into t\_student values(null,#{name},#{age})  </insert>  <update id=*"update"* parameterType=*"Student"*>  update t\_student set name=#{name},age=#{age} where id=#{id}  </update>    <delete id=*"delete"* parameterType=*"Integer"*>  delete from t\_student where id=#{id}  </delete>    <select id=*"findById"* parameterType=*"Integer"* resultType=*"Student"*>  select \* from t\_student where id=#{id}  </select>    <select id=*"find"* resultMap=*"StudentResult"*>  select \* from t\_student  </select>  </mapper> |

**AddressMapper.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">  <mapper namespace=*"com.java1234.mappers.AddressMapper"*>  <select id=*"findById"* parameterType=*"Integer"* resultType=*"Address"*>  select \* from t\_address where id=#{id}  </select>  </mapper> |

**GradeMapper.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">  <mapper namespace=*"com.java1234.mappers.GradeMapper"*>  <resultMap type=*"Grade"* id=*"GradeResult"*>  <result property=*"id"* column=*"id"*/>  <result property=*"gradeName"* column=*"gradeName"*/>  <collection property=*"students"* column=*"id"* select=*"com.java1234.mappers.StudentMapper.findByGradeId"*></collection>  </resultMap>    <select id=*"findById"* parameterType=*"Integer"* resultMap=*"GradeResult"*>  select \* from t\_grade where id=#{id}  </select>  </mapper> |

### 接口方法

**StudentMapper.java**

|  |
| --- |
| package com.java1234.mappers;  import java.util.List;  import com.java1234.model.Student;  public interface StudentMapper {  public int add(Student student);  public int update(Student student);  public int delete(Integer id);  public Student findById(Integer id);  public List<Student> find();  public Student findStudentWithAddress(Integer id);  public Student findByGradeId(Integer gradeId);  } |

**GradeMapper.java**

|  |
| --- |
| **package** com.java1234.mappers;  **import** com.java1234.model.Grade;  **public** **interface** GradeMapper {  **public** Grade findById(Integer id);  } |

**AddressMapper.java**

|  |
| --- |
| **package** com.java1234.mappers;  **import** com.java1234.model.Address;  **public** **interface** AddressMapper {  **public** Address findById(Integer id);  } |

### 实体类

**Student.java**

|  |
| --- |
| **package** com.java1234.model;  **public** **class** Student {  **private** Integer id;  **private** String name;  **private** Integer age;  **private** Address address;  **private** Grade grade;    **public** Student() {  **super**();  // **TODO** Auto-generated constructor stub  }    **public** Student(Integer id, String name, Integer age) {  **super**();  **this**.id = id;  **this**.name = name;  **this**.age = age;  }  **public** Student(String name, Integer age) {  **super**();  **this**.name = name;  **this**.age = age;  }  ........get和set方法  @Override  **public** String toString() {  **return** "Student [id=" + id + ", name=" + name + ", age=" + age  + ", address=" + address + ", grade=" + grade + "]";  }  } |

**Address.java**

|  |
| --- |
| **package** com.java1234.model;  **public** **class** Address {  **private** Integer id;  **private** String sheng;  **private** String shi;  **private** String qu;  ..........get和set方法  @Override  **public** String toString() {  **return** "Address [id=" + id + ", sheng=" + sheng + ", shi=" + shi  + ", qu=" + qu + "]";  }  } |

**Grade.java**

|  |
| --- |
| **package** com.java1234.model;  **import** java.util.List;  **public** **class** Grade {  **private** Integer id;  **private** String gradeName;  **private** List<Student> students;    ......get和set方法  @Override  **public** String toString() {  **return** "Grade [id=" + id + ", gradeName=" + gradeName +"]";  }  } |

### 获取访问数据库的SqlSessionFactory和SqlSession方法

|  |
| --- |
| public class SqlSessionFactoryUtil {  private static SqlSessionFactory sqlSessionFactory;    public static SqlSessionFactory getSqlSessionFactory(){  if(sqlSessionFactory==null){  InputStream inputStream=null;  try{  inputStream=Resources.getResourceAsStream("mybatis-config.xml");  sqlSessionFactory=new SqlSessionFactoryBuilder().build(inputStream);  }catch(Exception e){  e.printStackTrace();  }  }  return sqlSessionFactory;  }    public static SqlSession openSession(){  return getSqlSessionFactory().openSession();  }  } |

### 测试，执行Mapper接口中的方法

|  |
| --- |
| package com.java1234.service;  public class StudentTest2 {  private static Logger logger=Logger.getLogger(StudentTest.class);  private SqlSession sqlSession=null;  private StudentMapper studentMapper=null;    /\*\*  \* 测试方法前调用  \* @throws Exception  \*/  @Before  public void setUp() throws Exception {  sqlSession=SqlSessionFactoryUtil.openSession();  studentMapper=sqlSession.getMapper(StudentMapper.class);  }  /\*\*  \* 测试方法后调用  \* @throws Exception  \*/  @After  public void tearDown() throws Exception {  sqlSession.close();  }  @Test  public void testAdd() {  logger.info("添加学生");  Student student=new Student("王五",12);  studentMapper.add(student);  sqlSession.commit();  }  @Test  public void testUpdate(){  logger.info("修改学生");  Student student=new Student(8,"王五2",13);  studentMapper.update(student);  sqlSession.commit();  }  @Test  public void testDelete(){  logger.info("删除学生");  studentMapper.delete(8);  sqlSession.commit();  }  @Test  public void testFindById(){  logger.info("通过ID查找学生");  Student student=studentMapper.findById(1);  System.out.println(student);  }  @Test  public void testFind(){  logger.info("查找所有学生");  List<Student> studentList=studentMapper.find();  for(Student s:studentList){  System.out.println(s);  }  }  } |

### 重要知识点总结

9.1 typeAliases 别名的配置和作用（在mapper.xml映射时，可以起作用）

9.2 mappers 统一地引入映射文件

9.3 insert、delete、selete、update这四个简单的映射语句（在mapper.xml中）

9.4 一对一关系实现 （association ）

9.5 一对多关系实现 （collection ）