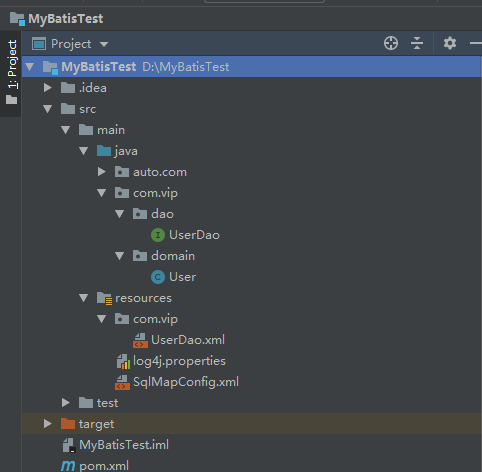
工程项目结构:



1.导入依赖

<?xml version="1.0" encoding="UTF-8"?>  
  
<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>auto.com</groupId>  
 <artifactId>MyBatisTest</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <name>MyBatisTest</name>  
 <!-- *FIXME change it to the project's website* -->  
 <url>http://www.example.com</url>  
  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <maven.compiler.source>1.7</maven.compiler.source>  
 <maven.compiler.target>1.7</maven.compiler.target>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.11</version>  
 <scope>test</scope>  
 </dependency>  
 <!--mybatis依赖-->  
 <dependency>  
 <groupId>org.mybatis</groupId>  
 <artifactId>mybatis</artifactId>  
 <version>3.4.6</version>  
 </dependency>  
 <!--mysql依赖-->  
 <dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>8.0.21</version>  
 </dependency>  
 <!--日志依赖-->  
 <dependency>  
 <groupId>log4j</groupId>  
 <artifactId>log4j</artifactId>  
 <version>1.2.17</version>  
 </dependency>  
  
 </dependencies>  
  
 <build>  
 <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->  
 <plugins>  
 <!-- clean lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean\_Lifecycle -->  
 <plugin>  
 <artifactId>maven-clean-plugin</artifactId>  
 <version>3.1.0</version>  
 </plugin>  
 <!-- default lifecycle, jar packaging: see https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_jar\_packaging -->  
 <plugin>  
 <artifactId>maven-resources-plugin</artifactId>  
 <version>3.0.2</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.0</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-surefire-plugin</artifactId>  
 <version>2.22.1</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-jar-plugin</artifactId>  
 <version>3.0.2</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-install-plugin</artifactId>  
 <version>2.5.2</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-deploy-plugin</artifactId>  
 <version>2.8.2</version>  
 </plugin>  
 <!-- site lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#site\_Lifecycle -->  
 <plugin>  
 <artifactId>maven-site-plugin</artifactId>  
 <version>3.7.1</version>  
 </plugin>  
 <plugin>  
 <artifactId>maven-project-info-reports-plugin</artifactId>  
 <version>3.0.0</version>  
 </plugin>  
 </plugins>  
 </pluginManagement>  
 </build>  
</project>

2.创建Mybatis全局配置文件SqlMapConfig.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>  
 <!--配置环境-->  
 <environments default="mysql">  
 <!--配置mysql的环境-->  
 <environment id="mysql">  
 <!--配置事务的类型-->  
 <transactionManager type="JDBC" />  
 <!-- 配置数据源(连接池) -->  
 <dataSource type="POOLED">  
 <!--配置连接数据库的4个基本信息-->  
 <property name="driver" value="com.mysql.jdbc.Driver" />  
 <property name="url" value="jdbc:mysql://localhost:3306/mybatis?serverTimezone=GMT" />  
 <property name="username" value="root" />  
 <property name="password" value="123456" />  
 </dataSource>  
 </environment>  
 </environments>  
 <mappers>  
 <mapper resource="com/vip/UserDao.xml"/>  
 </mappers>  
</configuration>

3.创建日志配置文件log4j.properties

log4j.rootCategory=debug, CONSOLE, LOGFILE  
  
# Set the enterprise logger category to FATAL and its only appender to CONSOLE.  
log4j.logger.org.apache.axis.enterprise=FATAL, CONSOLE  
  
# CONSOLE is set to be a ConsoleAppender using a PatternLayout.  
log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender  
log4j.appender.console.Target = System.out  
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout  
log4j.appender.CONSOLE.layout.ConversionPattern=%d{ISO8601} %-6r [%15.15t] %-5p %30.30c %x - %m\n  
  
# LOGFILE is set to be a File appender using a PatternLayout.  
log4j.appender.LOGFILE=org.apache.log4j.FileAppender  
log4j.appender.LOGFILE.File=d:\\log\\axis.log  
log4j.appender.file.File = target/MyBatisTest.log  
log4j.appender.LOGFILE.Append=true  
log4j.appender.LOGFILE.layout=org.apache.log4j.PatternLayout  
log4j.appender.LOGFILE.layout.ConversionPattern=%d{ISO8601} %-6r [%15.15t] %-5p %30.30c %x - %m\n

4.创建User实体类:

package com.vip.domain;  
  
import java.io.Serializable;  
import java.util.Date;  
  
*/\*\*\*  
 \* 用户实体类  
 \*/*public class User implements Serializable {  
 private Integer id; //用户id  
 private String userName; //用户名称  
 private Date birthday; //用户生日  
 private String sex; //用户性别  
 private String address; //用户地址  
  
 public Integer getId() {  
 return id;  
 }  
  
 public void setId(Integer id) {  
 this.id = id;  
 }  
  
 public String getUserName() {  
 return userName;  
 }  
  
 public void setUserName(String userName) {  
 this.userName = userName;  
 }  
  
 public Date getBirthday() {  
 return birthday;  
 }  
  
 public void setBirthday(Date birthday) {  
 this.birthday = birthday;  
 }  
  
 public String getSex() {  
 return sex;  
 }  
  
 public void setSex(String sex) {  
 this.sex = sex;  
 }  
  
 public String getAddress() {  
 return address;  
 }  
  
 public void setAddress(String address) {  
 this.address = address;  
 }  
  
 @Override  
 public String toString() {  
 return new StringBuilder()  
 .append("User{")  
 .append("id=")  
 .append(id)  
 .append(",userName=")  
 .append(userName)  
 .append(",birthday=")  
 .append(birthday)  
 .append(",sex= ")  
 .append(sex)  
 .append(",address= ")  
 .append(address)  
 .append('}').toString();  
 }  
}

5.创建Dao接口:

package com.vip.dao;  
  
import com.vip.domain.User;  
  
import java.util.List;  
  
*/\*\*  
 \* 用户持久层  
 \*/*public interface UserDao {  
 */\*\*  
 \* 查询所有的用户  
 \** ***@return*** *\*/* public List<User> queryAll();  
}

6.创建UserDao.xml文件:

<?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.vip.dao.UserDao">  
 <select id="queryAll" resultType="com.vip.domain.User">  
 select \* from users.user  
 </select>  
  
</mapper>

6.编写测试类：

package com.vip.test;  
  
  
import com.vip.dao.UserDao;  
import com.vip.domain.User;  
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 java.io.InputStream;  
import java.util.List;  
  
public class MyBatisTest {  
 */\*\*  
 \* 入门案例  
 \** ***@param*** *args  
 \*/* public static void main(String[] args) throws Exception{  
 //1.读取配置文件  
 InputStream inputStream = Resources.*getResourceAsStream*("SqlMapConfig.xml");  
 //2.创建SqlSessionFactory工厂  
 SqlSessionFactoryBuilder sqlSessionFactoryBuilder = new SqlSessionFactoryBuilder();  
 SqlSessionFactory sqlSessionFactory = sqlSessionFactoryBuilder.build(inputStream);  
 //3.使用工厂生产SqlSession对象  
 SqlSession session = sqlSessionFactory.openSession();  
 //4.使用SqlSession创建Dao接口的代理对象  
 UserDao userDao = session.getMapper(UserDao.class);  
 //5.使用代理对象执行方法  
 List<User> users = userDao.queryAll();  
 //6.循环输出查询数据  
 for(User user : users){  
 System.*out*.println(user);  
 }  
 //.7释放资源  
 session.close();  
 inputStream.close();  
 }  
}

7.测试Mybatis运行结果:



运行常见错误:

The server time zone value ‘�й���׼ʱ��’ is unrecognized or represents more than one time zone. You must configure either the server or JDBC driver (via the serverTimezone configuration property) to use a more specifc time zone value if you want to utilize time zone support.

使用的数据库是MySQL，驱动是8.0.18，这是由于数据库和系统时区差异所造成的，在jdbc连接的url后面加上serverTimezone=GMT即可解决问题，如果需要使用gmt+8时区，需要写成GMT%2B8，否则会被解析为空。再一个解决办法就是使用低版本的MySQL jdbc驱动，5.1.28不会存在时区的问题。



MyBatis入门文章参考:

http://www.zgp520.com/index.php/2020/05/22/idea%E4%B8%AD%E6%90%AD%E5%BB%BAmybatis%E5%BC%80%E5%8F%91%E7%8E%AF%E5%A2%83%E7%A4%BA%E4%BE%8B/