Mybatis-spring配置的记录

在整合的ssm框架中，关于持久层的mybatis的配置，需要一个spring.xml文件来配置(通常命名为applicationContext.xml)

其中配置有，

dataSource

<bean id =“dataSource”class=“ ”>

<property name=“driverClassName”value=“${jdbc.driver}”/>

<property name =“url”value=“${jdbc.url}” />

<property name =“username”value=“${jdbc.username}” />

<property name =“password”value=“${jdbc.password}” />

</bean>

sqlSessionFactory

<bean id=“sqlSessionFactory”class=“org.mybaits.springs.SqlSessionFactoryBean”>

<!-- dataSource必须配置-->

<property name= “dataSource”ref= “”/>

<!—1.可以不单独配置Mybaits 的配置文件，mapperLocations指定mapper.xml路径-->

<property name=“mapperLocations”value=“classpath://mapper/\*.xml”/>

<!--2.Mybatis配置文件里面指定mapper.xml，configLocation指定配置文件路径-->

<property name=“configLocation””value=“classpath://sqlMapConfig.xml”/>

</bean>

--🡪

方法2中的sqlMapConfig.xml文件中的mapper.xml指定格式：

<mappers>

<!--一个一个mapper.xml指定-->

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

<!-- 指定一个包，自动加载所有包下所有mapper.xml-->

<package name="com.zcy.ssm.mapper"/>

<mappers/>

**总结：**

当MyBatis 映射器 XML 文件在和映射器类不在同一路径下,有两种解决方法。

第一是手动在 MyBatis 的 XML 配置文件中(sqlMapConfig.xml)使用<mappers>部分来指定类路径。

第二是使用工厂 bean的 mapperLocations 属 性。(classpath:///mapper/\*.xml)

/\*\*

\* 三种调用的对比：

\* mapper.xml 、Student.java 、StuentImpl.java 、StudentMapper.java 的关系

\*/

/\* 原始的调用方法，命名空间点. sql语句方法id( namespace . id )

这里没有mapper接口，直接使用了xml文件中namespace.id的调用\*/

// SqlSession sqlSession = sqlSessionFactory.openSession();

// Student student = sqlSession.selectOne("student.findStudentBySname", "李勇");

// System.out.println(student);

//与mapper的对比：这里没有mapper接口，直接使用了xml文件中namespace.id的调用(这里通过mybatis包实现)

/\*DAO的调用方法，一个接口Student，接口有实现类StudentImpl，实现类里写了sqlSession会话的各种方法

\* 如：sqlSession.selectone("test.findStudentBySname","李勇");

\* sqlSeesion.insert("test.insertStudent",user);

\* sqlSession.delete("test.deleteStudent",1);

\* 以上方法时通过 实现类 中的 sqlSession.seletone insert delete (namesapce.id)与mapper.xml产生对应联系

\* \*/

// UserDao userDao = new UserDaoImpl(sqlSessionFactory);

// // 调用UserDao的方法

// User user = userDao.findUserById(1);

// System.out.println(user);

/\* mapper代理方法，直接自动映射mapper接口类.方法名() \*/

/\* 通过sqlSession的getMapper( .class)方法得到实现类 ,返回给代理对象，

\* 而不是像DAO一样new一个实现类，再.点方法。

\* 也不是原始的直接sqlSession.点方法\*/

StudentMapper studentMapper = sqlSession.getMapper(StudentMapper.class);

Student student = studentMapper.findStudentBySname("李勇");

sqlSession.close();

System.out.println(student);

}