## Mybatis基本流程

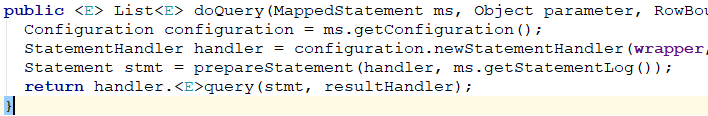
最原始的方法，在没有spring的时候，我们需要创建sqlSessionFactory，并且通过该工厂来创建一个SqlSession，SqlSession可以用来执行Sql语句，获取mapper，管理事务。是的，SqlSession也是可以执行Sql语句，一般来说我们都是通过mapper来进行sql操作，当然也可以通过获取mapper的动态代理mapperProxy来通过调用相关的方法执行sql但是mapperProxy里面还是调用的sqlSession的相关方法：

。

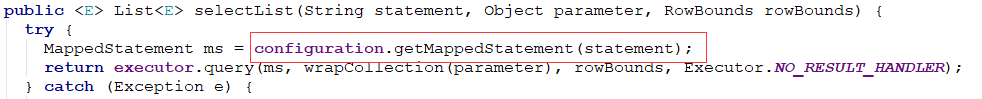
那么SqlSession里面调用的是什么？以SqlSession实现类DefaultSqlSession来说，他里面保存了一个Executor的引用，Executor是一个接口，以他的实现类ReuseExecutor来说，他里面会保存一个Transaction的引用，而Transaction当中又会保存一个connection的引用，这样子看来，SqlSession里面应该是调用了Executor的方法



那在Executor里面又是哪个对象去执行sql语句的呢？我们知道，在编写jdbc的时候，我们需要从connection中获取一个PrepareStatement(sql)，并且设置相应的值，然后再通过PrepareStatement，很巧，在Executor里面也是通过PrepareStatement的方法来执行Sql语句。例如ReuseExecutor中的doQuery方法：

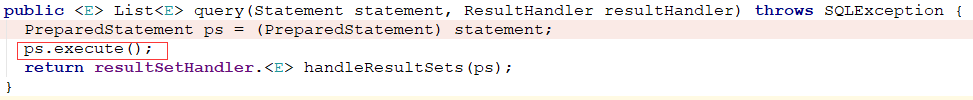


该方法里面有一个prepareStatement()方法来准备PrepareStatement（层层深入后发现是直接调用的Connection.prepareStatement方法，在此之前会把sql和参数都设置好了，Sql语句是一开始解析mapper.xml文件的时候连同<select>标签以及返回值类型等一同封装成一个对象然后被缓存—key是全类名+方法名，value是MappedStatement，执行DefaultSqlSession的selectList方法的时候会根据key值来来获取对应的MappedStatement,

）。



这样我们就准备好了prepareStatement，接下来可以调用statementHandler的query方法了。



可以看到他最终调用了PrepareStatement的execute()方法，然后进行结果集的处理，包括表属性和对象属性之间的映射。这样一套流程就下来了。

总结下:

对于每一个mapper接口，Mybatis都会创建一个动态代理proxy，然后可以通过proxy来进行相关的操作，例如，mapper接口里面有个getDepartment方法，proxy在调用这个方法的时候会调用到InvocationHandler里面的invoke方法，在invoke方法里面，会将传过来的method解析，获取到该method对应的那条sql语句是什么类型，是select还是update，然后根据不同的标签类型来做相应的处理。以select为例，会调用sqlsession的selectOne方法，而在selectOne()方法里面又调用了sqlSession内置的executor的query()方法，在executor内部，会准备好一个prepareStatement，然后调用该prepareStatement的相关方法就行。

Mapper🡪MapperProxy🡪sqlSession🡪executor🡪prepareStatement

## Mybatis整合Spring

在最原始的环境里面，Mybatis有两种事务类，一个是JdbcTransaction，它会直接从datasource中获取connection，他会直接管理事务的提交和回滚。还有一个是ManagedTransaction，该类将事务回滚和提交都交由容器（如JBOSS，Weblogic）来操作。JdbcTransaction一般是用在单独使用Mybatis的情况下，而对于Web应用，该类显得有些力不从心了。一般来说，我们会在service层里面存放一个Maper引用，MapperProxy里面会存放sqlSession应用，那就意味着，多个线程会共享这个sqlSession，如果还是使用jdbcTransaction的话，多个线程就会对同一个connection引用操作，如果还使用原先的DefaultSqlSession，这样就有线程安全的问题。

在Spring于Mybatis整合的过程里面，SqlSessionTemplate被创建添加到ioc容器里面，所以SqlSessionTemplate是一个单例，SqlSessionTemplate会代替传统的SqlSession，可以通过他来获取mapper，然后再mapper调用invoke的时候，会调用到SqlSessionTemplate里面的sqlSessionProxy的方法，sqlSessionProxy是一个代理，其invoke方法获取与当前线程相关的SqlSession，该SqlSession是传统的SqlSession，例如DefaultSqlSession。



如果没有，则通过SqlSessionFactory来新建一个，并保存到当前线程中，在创建的时候，会从当前环境中获取TransactionFactory对象，spring提供了一个SpringManagedTransactionFactory，返回的是SpringManagedTransaction，而SpringManagedTransaction是从当前线程来获取Connection的，这样前后就连贯了，一直和当前线程绑定。在此之后就是Mybatis执行的那一套了。



Mapper🡪SqlSessionTemplate🡪SqlSessionProxy🡪DefaultSqlSession🡪Executor🡪prepareStatement.

Mybatis和Spring整合的时候，Mybatis一般是不会进行事务管理的，直接托管给Spring来操作，Spring使用的DataSourceTransactionManager，利用切面的技术，在方法调用之前开启事务，在方法调用结束之后关闭事务。当然也是通过将连接和线程绑定的方式来实现线程安全。DataSourceUtils这个类可以获取到当前事务使用的connection，其内部是使用的TransactionSynchronizationManager，该类可以进行全局ThreadLocal管理。