对于mybatis配置

<settings>  
 <setting name="mapUnderscoreToCamelCase" value="true" />  
</settings>

则在mapper中

<select id="get" resultType="CcCourse">  
SELECT Id ,IS\_JUMP\_CHAPTER ,RELEASE\_STATUS ,LIMIT\_DAY ,STUDY\_HOUR,TAGS as courseTag FROM CC\_COURSE WHERE Id = #{courseId}  
</select>

会将xx\_yy映射到bean中的xxYy字段

mybatis插件配置

<configuration>

<plugins>

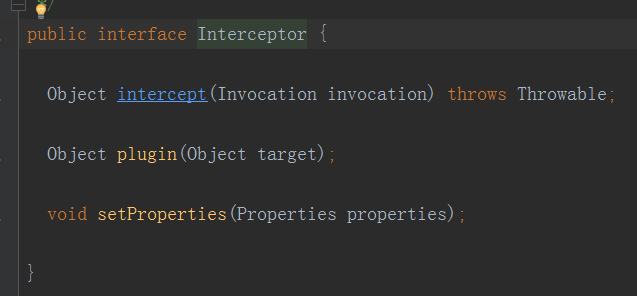
<plugin interceptor="className">

<property name=" " value=" " />

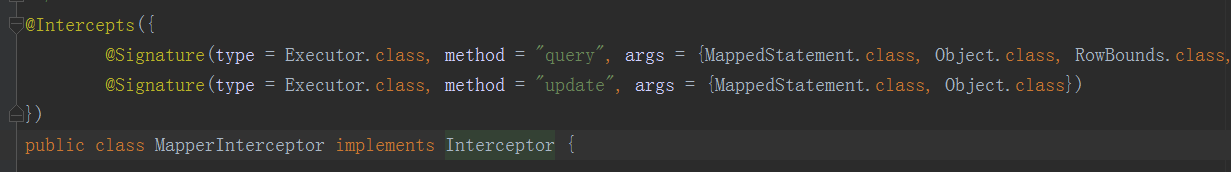
</plugin>

</plugins>  
</configuration>

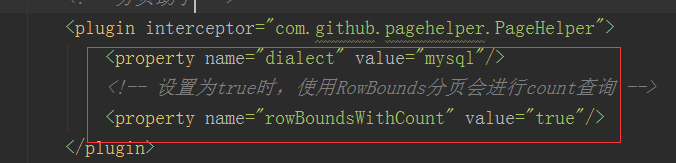
插件必须继承Interceptor接口



插件实现类一般会拦截



这个对象的@Intercepts内包含方法，具体逻辑在intercept方法里，生成代理类在plugin中

 这些配置则在setProperties中record，用于控制逻辑多态

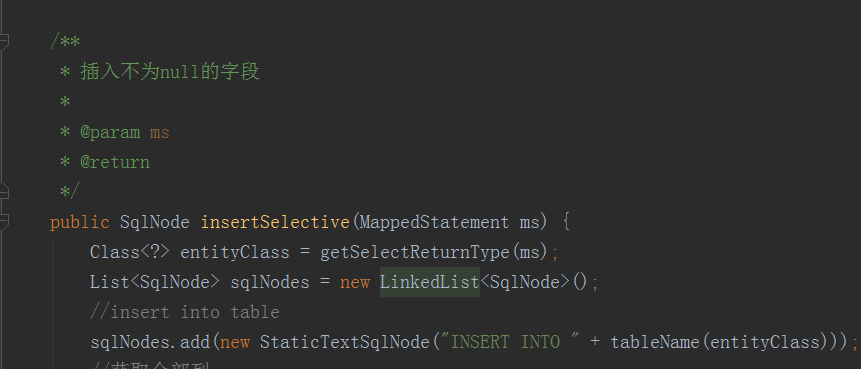
当前拥有插件，分页，注解式sql（通用mapper）

<plugins>  
 *<!-- 分页助手 -->* <plugin interceptor="com.github.pagehelper.PageHelper">  
 <property name="dialect" value="mysql"/>  
 *<!-- 设置为true时，使用RowBounds分页会进行count查询 -->* <property name="rowBoundsWithCount" value="true"/>  
 </plugin>  
 *<!-- 通用mapper插件 -->* <plugin interceptor="com.github.abel533.mapperhelper.MapperInterceptor">  
 *<!-- 主键生成方式 -->* <property name="IDENTITY" value="MYSQL" />  
 *<!-- 指定了Mapper接口 -->* <property name="mappers" value="com.github.abel533.mapper.Mapper" />  
 </plugin>  
</plugins>

在通用mapper实现中

public interface InsertSelectiveMapper<T> {  
  
 @InsertProvider(type = MapperProvider.class, method = "dynamicSQL")  
 int insertSelective(T record);  
  
}

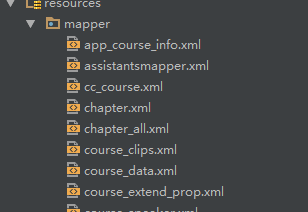
MapperProvider里有个方法insertSelective

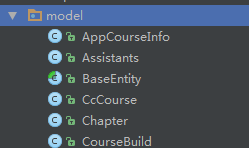


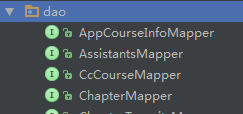
方法是添加一系列SqlNode，然后生成一个MixedSqlNode返回。

其中获取table类注解时使用的是javax.persistence-api-1.0

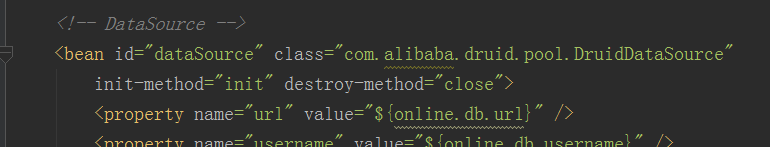
对于其配置





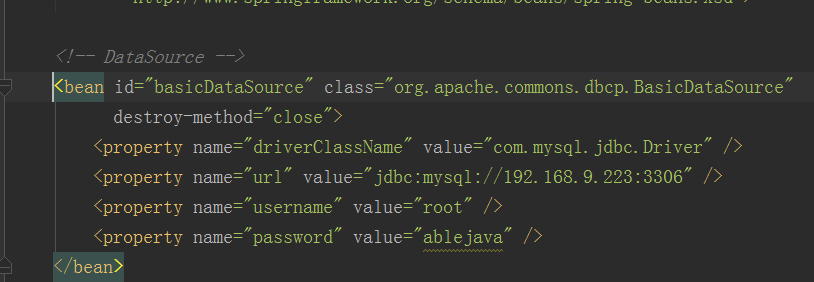


需要配置数据源



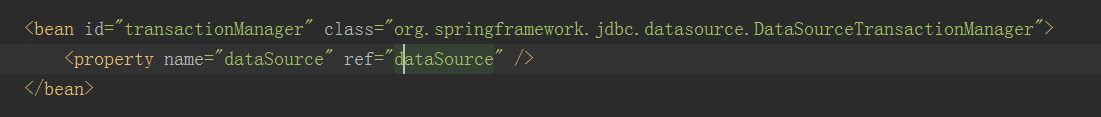


或者



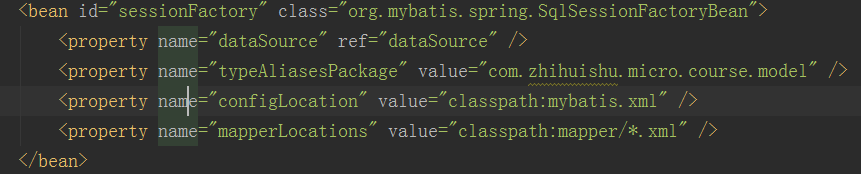


然后定义事务管理器 引用（数据源）



定义会话工厂 引用（数据源）

这样就引用到了前面的xml系列文件，model类文件



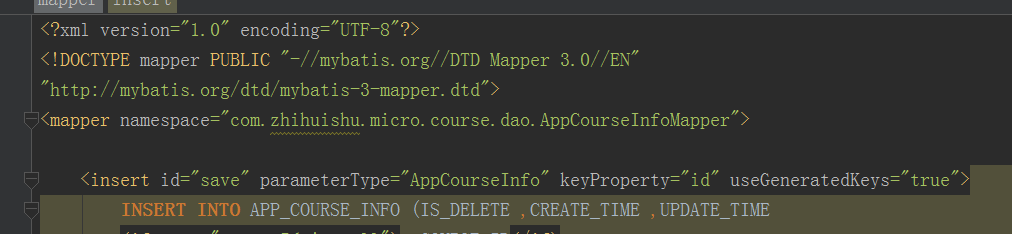
若要引用到前面的dao类文件，需要一定的配置生成其代理实现类

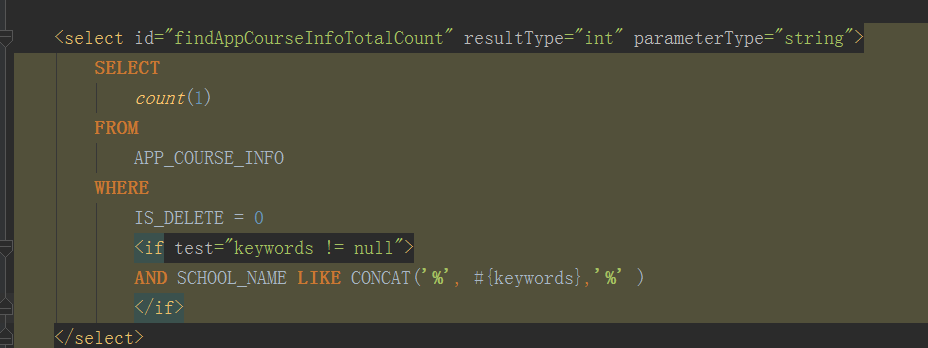
主要是配置一个Mapper扫描器，它又引用了一个会话模板类，

会话模板类又引用了会话工厂



xml

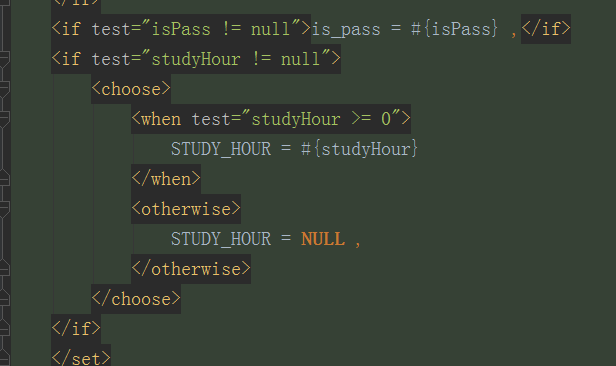




其中模糊查询用到了concat函数

<select id="getList" parameterType="list" resultType="Chapter">  
 SELECT Id ,NAME ,ORDER\_NUMBER ,PERSONAL\_COURSE\_ID ,DESCRIPTION ,LIMIT\_DAY ,STUDY\_HOUR ,is\_pass FROM CC\_CHAPTER WHERE IS\_DETELED = 0 AND ID IN  
 <foreach collection="list" item="i" open="(" separator="," close=")">#{i}</foreach>  
</select>

getList方法，参数为一个List<Id>，使用到了foreach标签



两种选择标签



使用trim标签动态生成sql，上面实现了更新不同的id的字段为不同的值

MyBatis/Ibatis中#和$的区别

1. #将传入的数据都当成一个字符串，会对自动传入的数据加一个双引号。如：order by #user\_id#，如果传入的值是111,那么解析成sql时的值为order by "111", 如果传入的值是id，则解析成的sql为order by "id".

2. $将传入的数据直接显示生成在sql中。如：order by $user\_id$，如果传入的值是111,那么解析成sql时的值为order by user\_id,  如果传入的值是id，则解析成的sql为order by id.

3. #方式能够很大程度防止sql注入。

4.$方式无法防止Sql注入。

5.$方式一般用于传入[**数据库**](http://lib.csdn.net/base/mysql)对象，例如传入表名.

6.一般能用#的就别用$.

ps:在使用mybatis中还遇到<![CDATA[]]>的用法，在该符号内的语句，将不会被当成字符串来处理，而是直接当成sql语句，比如要执行一个存储过程。

#{}相当于jdbc中的preparedstatement

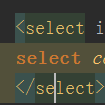
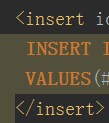
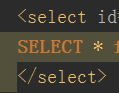
${}是输出变量的值

order by是肯定只能用${}了,用#{}会多个' '导致sql语句失效.

此外还有一个like 语句后也需要用${}

mybatis

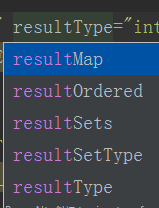
有如下标签类型:

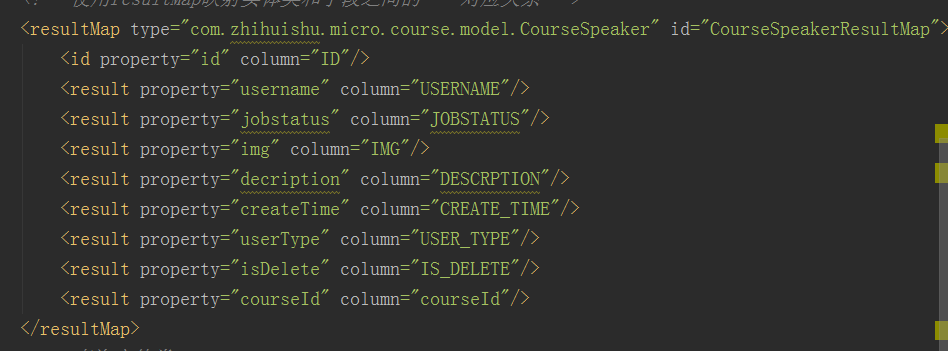
   

可能用到的属性

 主键生成



其中map对应了一个元素的id

