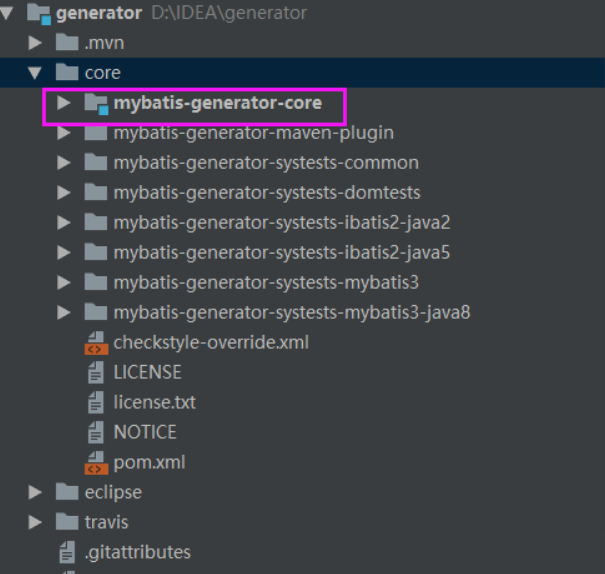
# MyBatis Generator源码分析修改和自定义插件：

从官网的GitHub上拉取源码，<https://github.com/mybatis/generator.git>



生成代码

创建表：

CREATE TABLE `users` (

`id` bigint(20) NOT NULL COMMENT '主键',

`name` varchar(255) DEFAULT NULL COMMENT '姓名',

`age` int(11) DEFAULT NULL COMMENT '年龄',

`sex` varchar(1) DEFAULT NULL COMMENT '性别',

`addr` varchar(255) DEFAULT NULL COMMENT '住址',

`birthday` date DEFAULT NULL COMMENT '出生年月',

`password` varchar(255) DEFAULT NULL COMMENT '密码',

`create\_time` datetime DEFAULT NULL COMMENT '创建时间',

`update\_time` datetime DEFAULT NULL COMMENT '修改时间',

`create\_user` bigint(20) DEFAULT NULL COMMENT '创建人',

`update\_user` bigint(20) DEFAULT NULL COMMENT '修改人',

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

创建生成配置文件的xml文件

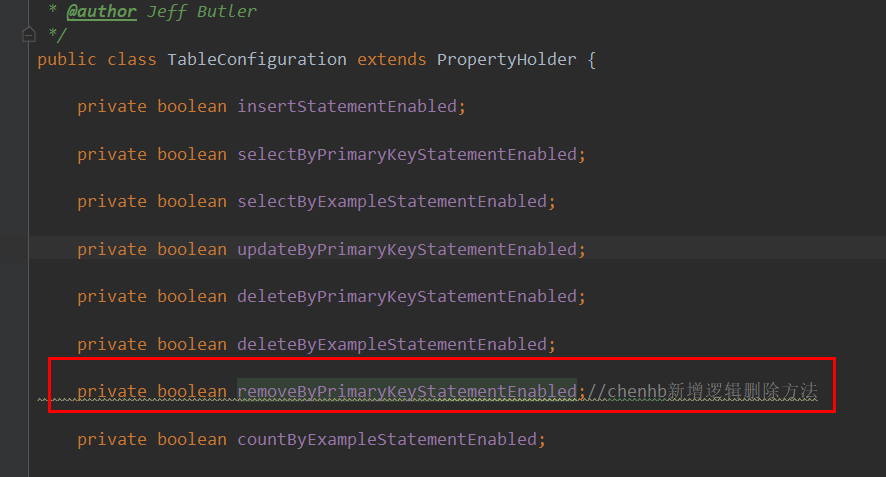
-------------

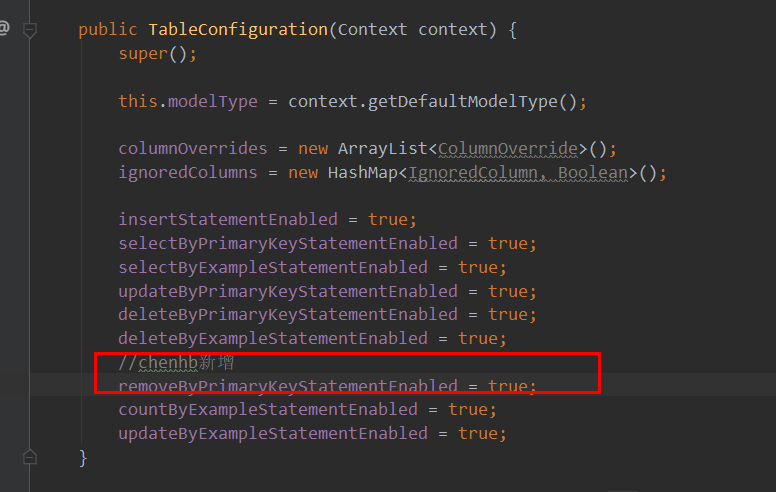
org.mybatis.generator.api.ShellRunner 运行该文件，生成代码；

自定义javaExampleGenerator标签

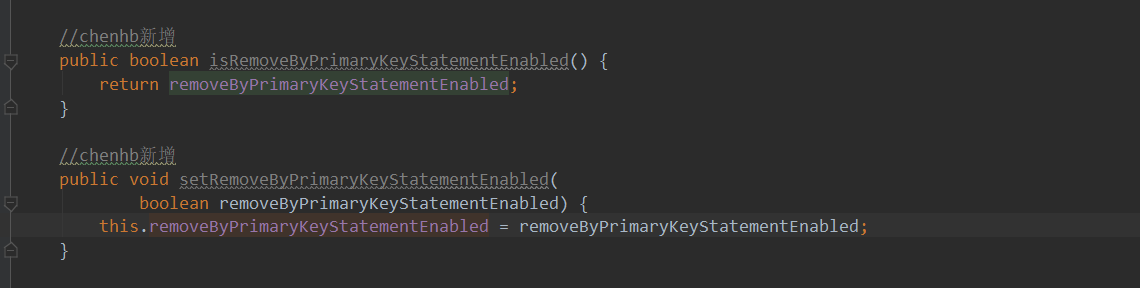
# 新增逻辑删除的方法

在类TableConfiguration中新增方法：

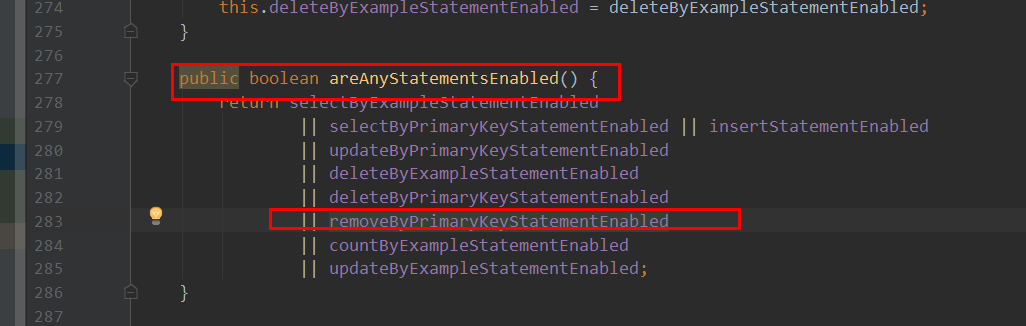




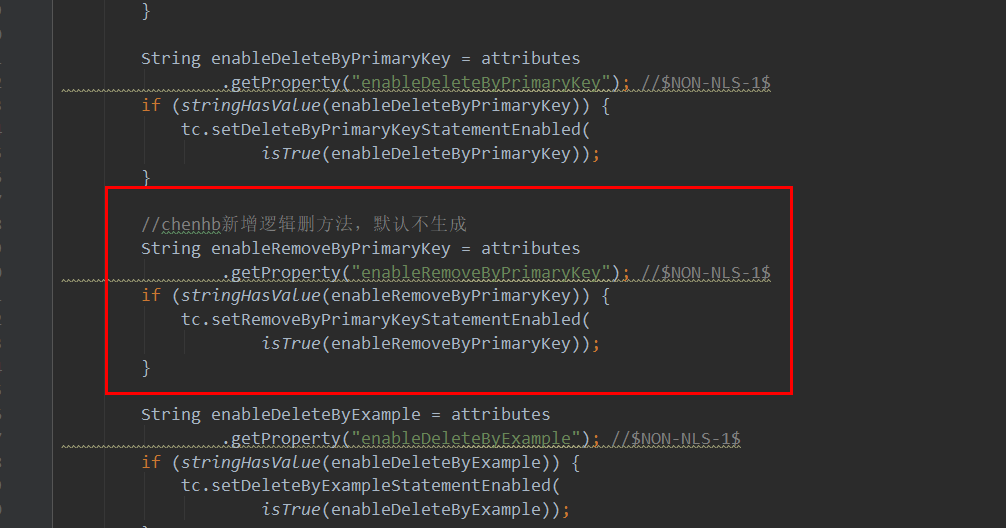
同时在类中新增对应的set和get方法如下：



同时在该类中修改该方法新增对应



在MyBatisGeneratorConfigurationParser类里注意修改parseTable方法

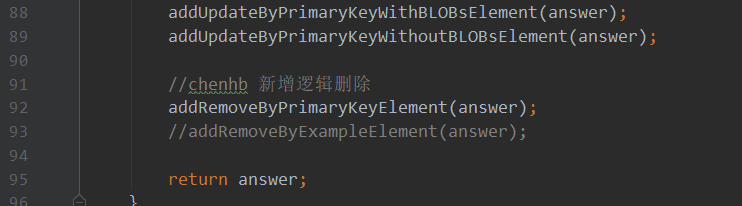


分别在XMLMapperGenerator和JavaMapperGenerator类里添加相关方法，对应方法的具体实现参考其他方法做简单修改即可，XMLMapperGenerator修改内容如下

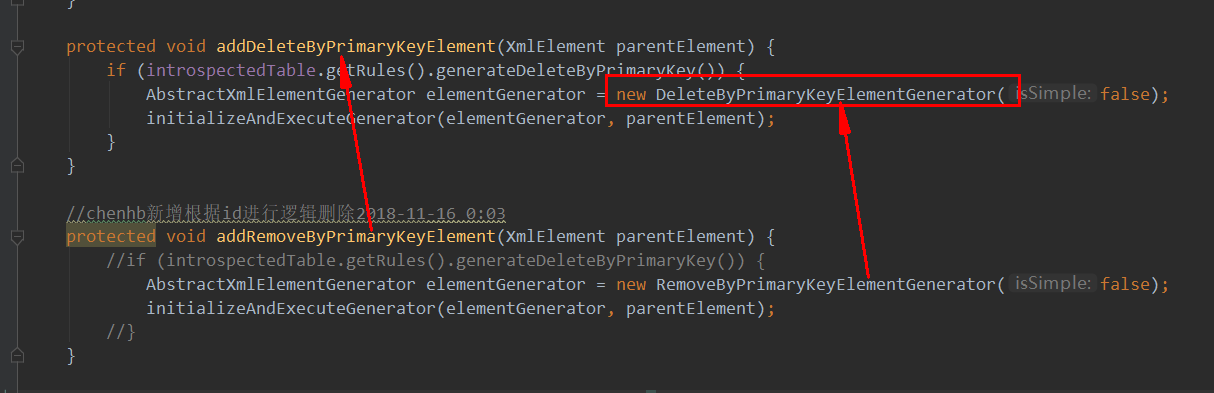
新增RemoveByPrimaryKeyExample方法

XMLMapperGenerator

getSqlMapElement方法中新增

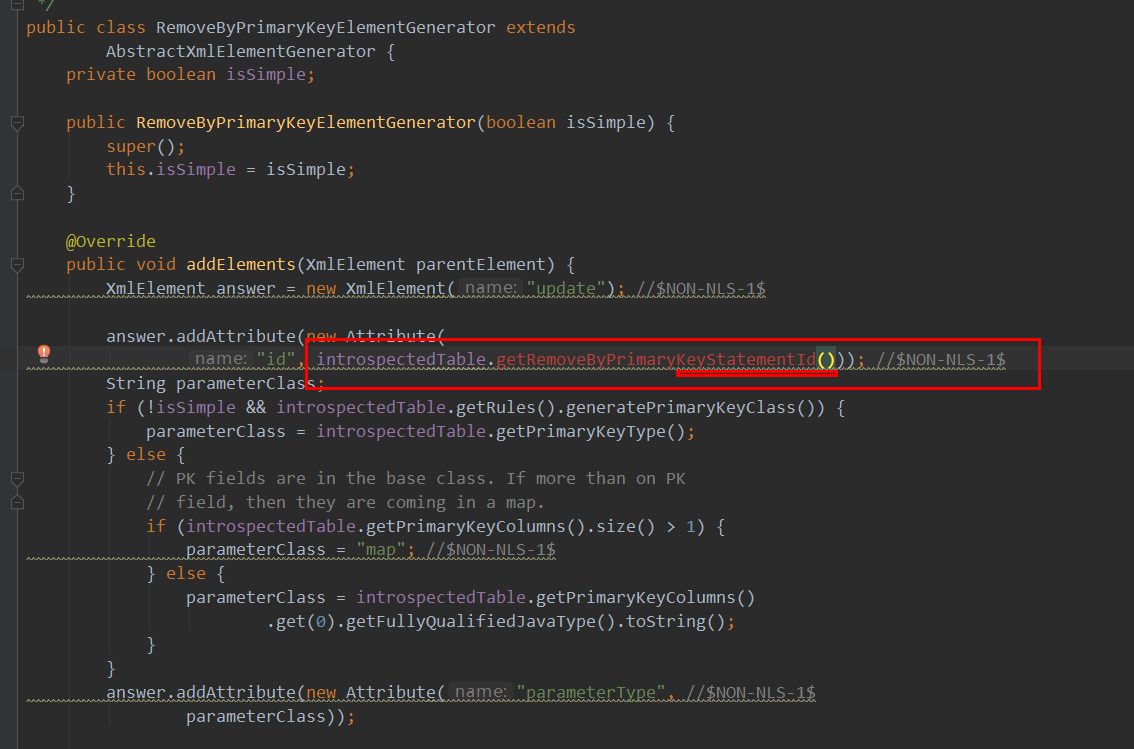


同时在该方法中新增该方法addRemoveByPrimaryKeyElement，同时新增红框中的创建类，该创建类可参考，下图

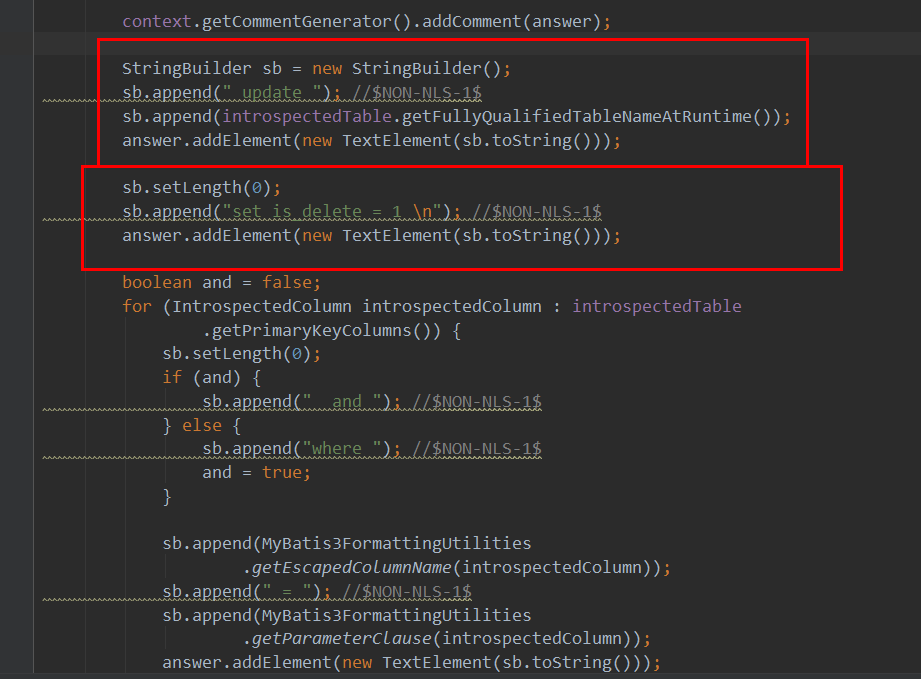


内容同样参考该类进行修改

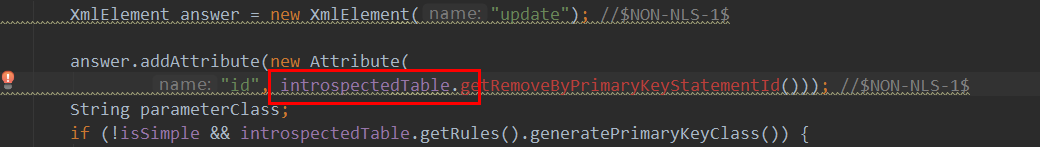
在该类中新增statementId的方法

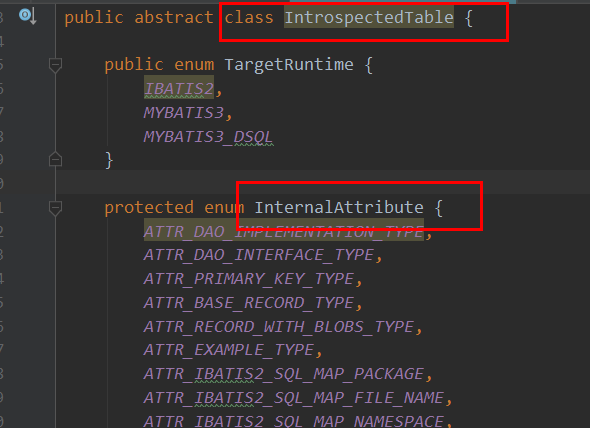


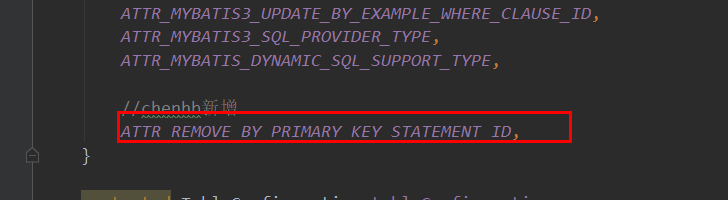
该类下sql修改，本方法实现逻辑删除，修改和添加如下

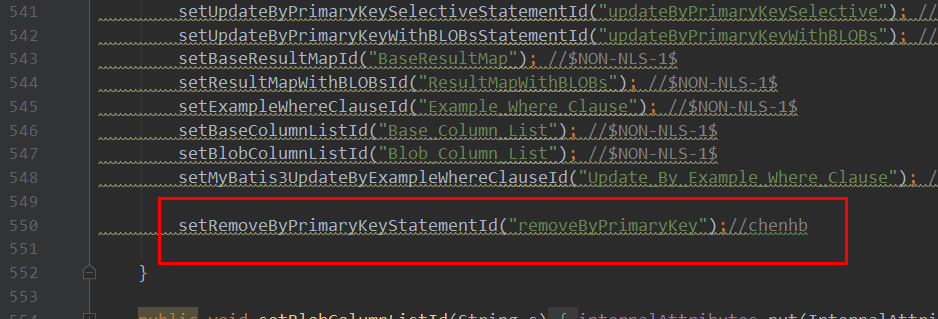


进入图中类中



在内部枚举类中新增如下枚举，同时创建刚才新增的获取相应statementId的set和get方法



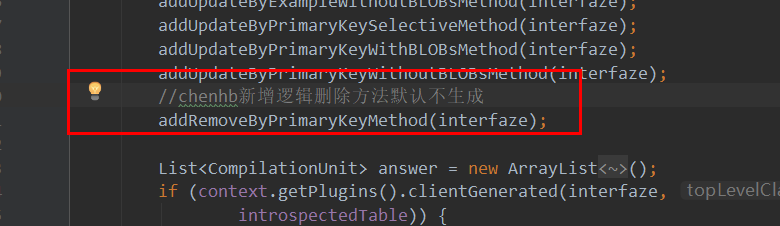


同时创建该方法，以及对应的get方法类似该类中的其他方法。

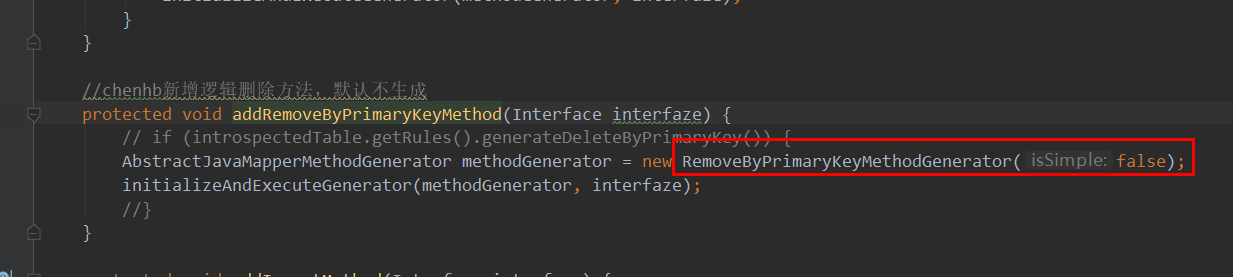
XMLMapperGenerator修改结束

------

在JavaMapperGenerator 类sgetCompilationUnits方法中新增

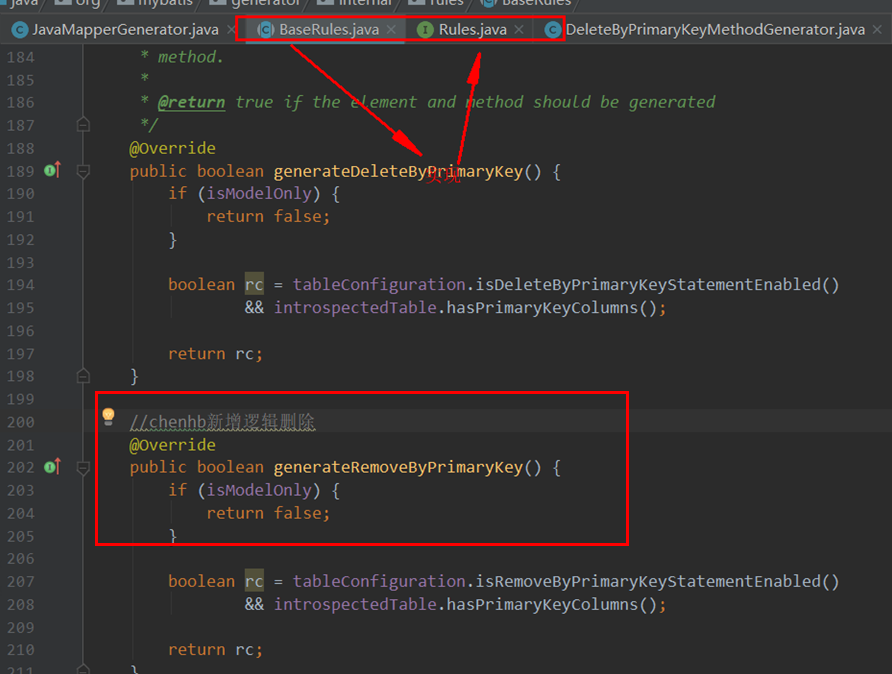


同时在该类中创建该方法，同时新增红框中的类，可安装其他方法进行修改创建

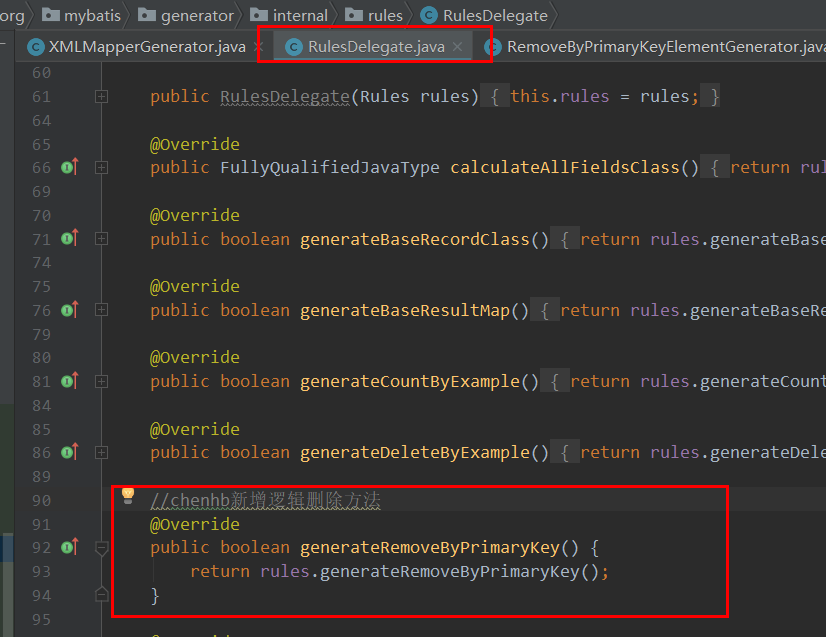


在如下类中新增

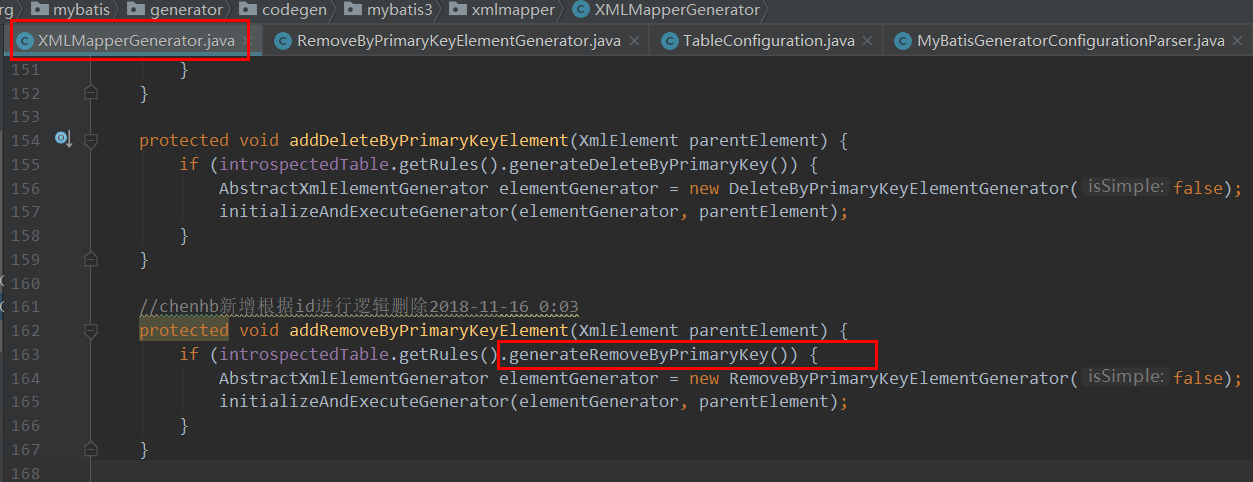




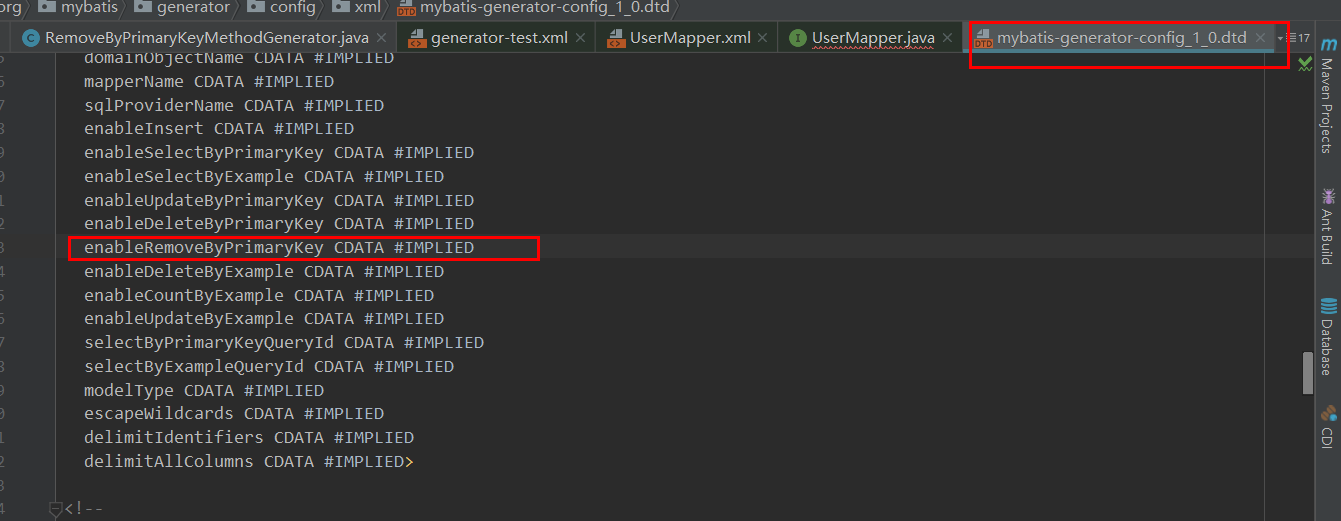
实现该方法,添加该规则后，同时如下类中也进行方法的实现：



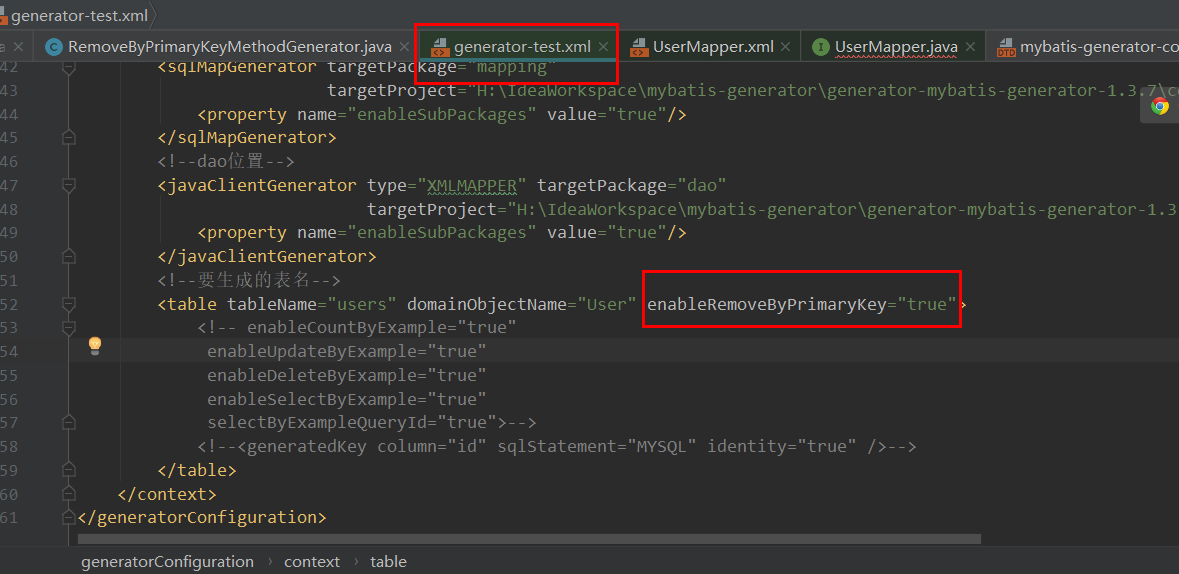
下图类中将原来注释点的内容打开并修改为刚才创建的内容



修改dtd校验文件



在创建文件配置是否生成



此时不进行配置或者为false都将不进行逻辑删除代码的生成。

# 2. MyBatis Generator 序列化处理

<plugin type="org.mybatis.generator.plugins.SerializablePlugin" /> 打开该插件类

新增方法

*/\*\*  
 \** ***@date*** *2018-11-14  
 \* chenhb 新增Example实现序列化接口  
 \** ***@param*** *topLevelClass  
 \** ***@param*** *introspectedTable  
 \** ***@return*** *\*/*@Override  
public boolean modelExampleClassGenerated(TopLevelClass topLevelClass, IntrospectedTable introspectedTable) {  
 //Example通过集成BaseExample序列化  
 //makeSerializable(topLevelClass, introspectedTable);  
  
 //内部类序列化  
 for (InnerClass innerClass : topLevelClass.getInnerClasses()) {  
 if ("GeneratedCriteria".equals(innerClass.getType().getShortName())) { //$NON-NLS-1$  
 innerClass.addSuperInterface(serializable);  
 }  
 if ("Criteria".equals(innerClass.getType().getShortName())) { //$NON-NLS-1$  
 innerClass.addSuperInterface(serializable);  
 }  
 if ("Criterion".equals(innerClass.getType().getShortName())) { //$NON-NLS-1$  
 innerClass.addSuperInterface(serializable);  
 }  
 }  
  
 return true;  
}

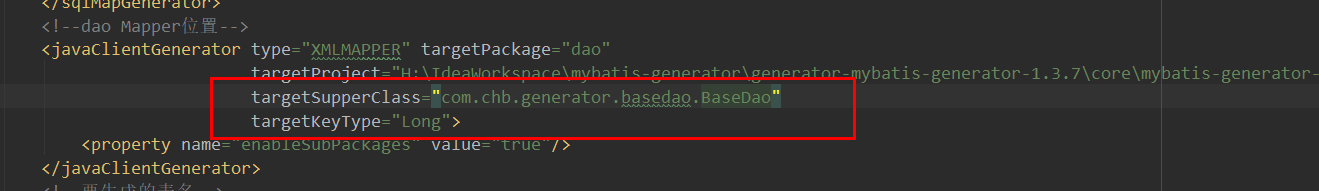
或者新建BaseExample类

package com.chb.generator.baseexample;  
  
import java.io.Serializable;  
  
*/\*\*  
 \* BaseExample  
 \*  
 \** ***@author*** *chenhb  
 \*/*public abstract class BaseExample implements Serializable {  
  
 private static final long *serialVersionUID* = 1L;  
  
}

在该方法中添加

@Override  
public boolean modelExampleClassGenerated(TopLevelClass topLevelClass, IntrospectedTable introspectedTable) {  
 //Example通过继承BaseExample序列化  
 //makeSerializable(topLevelClass, introspectedTable);  
 FullyQualifiedJavaType extendsClas = new FullyQualifiedJavaType("com.chb.generator.baseexample.BaseExample");  
 topLevelClass.addImportedType(extendsClas);  
 topLevelClass.setSuperClass(extendsClas);

# 3.Dao(Mapper)新增继承BaseDao



在dtd中增加校验，非必须

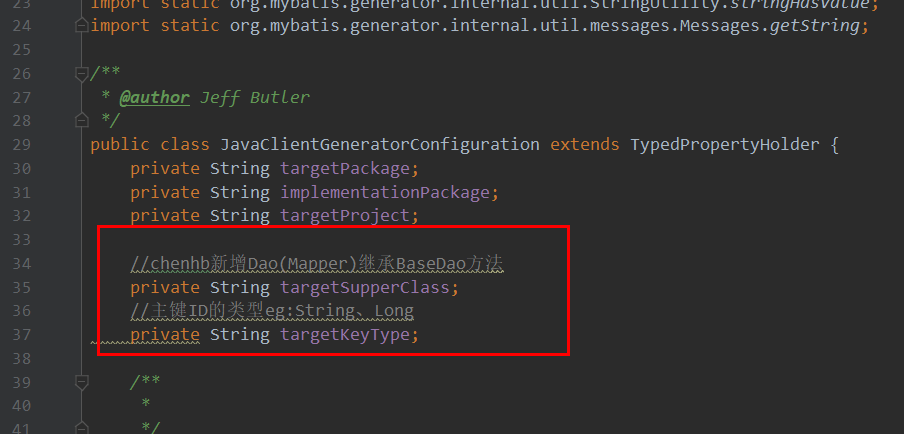
<!ELEMENT javaClientGenerator (property\*)>  
<!ATTLIST javaClientGenerator  
 type CDATA #REQUIRED  
 targetPackage CDATA #REQUIRED  
 targetProject CDATA #REQUIRED  
 targetSupperClass CDATA #IMPLIED  
 targetKeyType CDATA #IMPLIED  
 implementationPackage CDATA #IMPLIED>

KeyType为主键类型 此处用Long

在JavaMapperGenerator类getCompilationUnits中新增如下代码，在各种add方法前

//chenhb新增继承BaseDao  
JavaClientGeneratorConfiguration javaClientGeneratorConfiguration = context.getJavaClientGeneratorConfiguration();  
if (javaClientGeneratorConfiguration != null && javaClientGeneratorConfiguration.getTargetSupperClass() != null  
 && !"null".equals(javaClientGeneratorConfiguration.getTargetSupperClass())) {  
 String entity = introspectedTable.getBaseRecordType();  
 String entityExample = introspectedTable.getExampleType();  
 String superInterface = javaClientGeneratorConfiguration.getTargetSupperClass()  
 + "<" + javaClientGeneratorConfiguration.getTargetKeyType()  
 + "," + entity  
 + "," + entityExample +  
 ">";  
 FullyQualifiedJavaType fqjt = new FullyQualifiedJavaType(superInterface);  
 interfaze.addSuperInterface(fqjt);  
  
 FullyQualifiedJavaType fqjType = new FullyQualifiedJavaType("org.springframework.stereotype.Repository");  
 interfaze.addImportedType(fqjType);  
 interfaze.addAnnotation("@Repository");  
} else {  
 //所有的方法都抽到baseDao  
 addCountByExampleMethod(interfaze);  
 addDeleteByExampleMethod(interfaze);  
 addDeleteByPrimaryKeyMethod(interfaze);  
 addInsertMethod(interfaze);  
 addInsertSelectiveMethod(interfaze);  
 addSelectByExampleWithBLOBsMethod(interfaze);  
 addSelectByExampleWithoutBLOBsMethod(interfaze);  
 addSelectByPrimaryKeyMethod(interfaze);  
 addUpdateByExampleSelectiveMethod(interfaze);  
 addUpdateByExampleWithBLOBsMethod(interfaze);  
 addUpdateByExampleWithoutBLOBsMethod(interfaze);  
 addUpdateByPrimaryKeySelectiveMethod(interfaze);  
 addUpdateByPrimaryKeyWithBLOBsMethod(interfaze);  
 addUpdateByPrimaryKeyWithoutBLOBsMethod(interfaze);  
 //chenhb新增逻辑删除方法默认不生成  
 addRemoveByPrimaryKeyMethod(interfaze);  
}

在JavaClientGeneratorConfiguration类中新增



上面两个属性以及其对应的set get方法，同时在toXmlElement方法中新增代码：

//chenhb新增Dao(Mapper)继承BaseDao方法  
if (targetSupperClass != null) {  
 answer.addAttribute(new Attribute(  
 "targetSupperClass", targetSupperClass));  
}  
  
if (targetSupperClass != null) {  
 answer.addAttribute(new Attribute(  
 "targetKeyType", targetKeyType));  
}

在MyBatisGeneratorConfigurationParser类parseJavaClientGenerator方法中新增：

