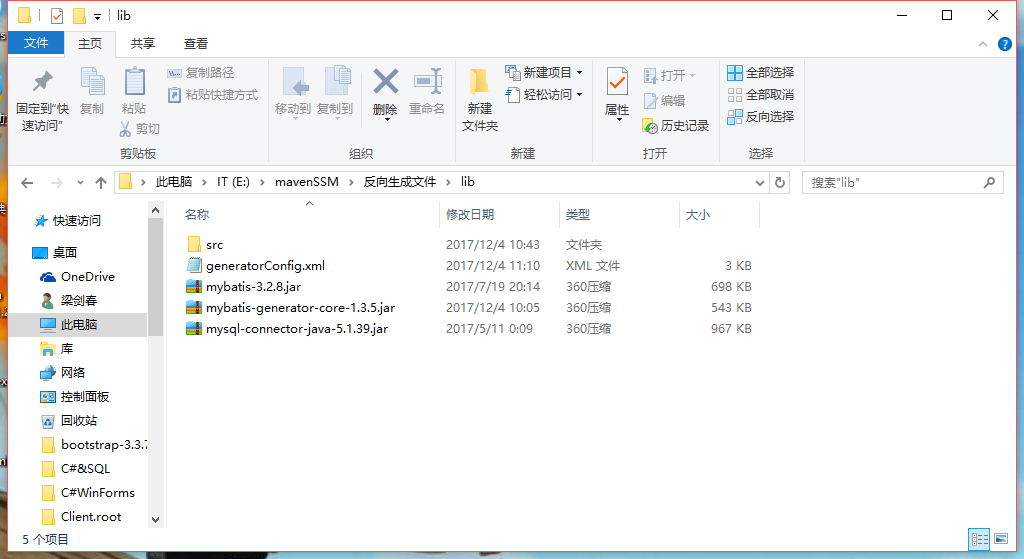
# MyBatis反向生成文件

**为什么需要mybatis反向生成实体类、dao层以及映射文件？**

是由于手写映射文件很容易出错，所以可利用MyBatis生成器自动生成实体类、DAO接口和Mapping映射文件，这样可以省去很多的功夫，将生成的代码copy到项目工程中即可。

**方法一（无插件）：**

****

生成代码需要的文件和jar包：

1. mybatis框架的jar包
2. 数据库驱动程序jar包
3. MyBatis生成器jar包
4. src文件夹
5. generatorConfig.xml 文件

**generatorConfig.xml 文件配置如下：**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE generatorConfiguration

PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"

"http://mybatis.org/dtd/mybatis-generator-config\_1\_0.dtd">

<generatorConfiguration>

<!-- mysql-connector文件路径（数据库驱动包路径） -->

<classPathEntry location="E:\mavenSSM\反向生成文件\lib\mysql-connector-java-5.1.39.jar"/>

<context id="DB2Tables" targetRuntime="MyBatis3">

<commentGenerator>

<!-- 抑制警告 -->

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

<!-- 是否生成注释代时间戳 -->

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

<!-- 是否去除自动生成的注释 true：是 ： false:否 -->

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

</commentGenerator>

<!-- 数据库链接配置 -->

<jdbcConnection driverClass="com.mysql.jdbc.Driver"

connectionURL="jdbc:mysql://127.0.0.1:3306/school"

userId="root" password="123456">

</jdbcConnection>

<javaTypeResolver>

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

</javaTypeResolver>

<!-- 生成实体类的路径，com.yang.school.entity 这个路径可以自动生成，但是必须有src这个路径-->

<javaModelGenerator targetPackage="com.yang.school.entity"

targetProject="src">

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

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

</javaModelGenerator>

<!-- 生成mapping的路径，这个路径可以自动生成，但是必须有src这个路径-->

<sqlMapGenerator targetPackage="com.yang.school.mapping"

targetProject="src">

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

</sqlMapGenerator>

<!-- 生成dao的路径，这个路径可以自动生成，但是必须有src这个路径-->

<javaClientGenerator type="XMLMAPPER" targetPackage="com.yang.school.dao"

targetProject="src">

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

</javaClientGenerator>

<!-- 要生成的表 tableName是数据库中的表名或视图名 domainObjectName是实体类名 -->

<table tableName="class\_info" domainObjectName="ClassInfo" enableCountByExample="false"

enableUpdateByExample="false" enableDeleteByExample="false"

enableSelectByExample="false" selectByExampleQueryId="false">

<!--domain字段的命名规则，false:默认为驼峰命名 true:按数据库真实命名 -->

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

<!-- 忽略列，不生成bean 字段 -->

<!-- <ignoreColumn column="FRED" /> -->

<!-- 指定列的java数据类型 -->

<!-- <columnOverride column="LONG\_VARCHAR\_FIELD" jdbcType="VARCHAR" /> -->

</table>

</context>

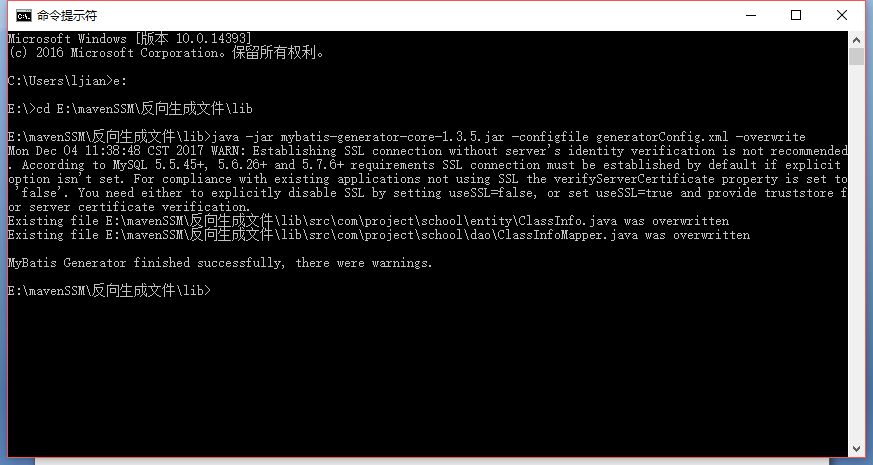
</generatorConfiguration>

**（想自动生成那个表的数据，就修改<table>标签中的tableName属性和domainObjectName属性即可，属性详情请看注释）**

配置完generatorConfig.xml 文件后打开控制台，进入相对应的lib目录下，例如我的是：E:\mavenSSM\反向生成文件\lib

**执行命令：**

**java -jar mybatis-generator-core-1.3.5.jar -configfile generatorConfig.xml –overwrite**

****

这样在生成之后，就可以在src目录下找到相应的文件夹，每个表格都会对应三个文件（实体类、接口、配置文件）。- -**《方法一结束！》**

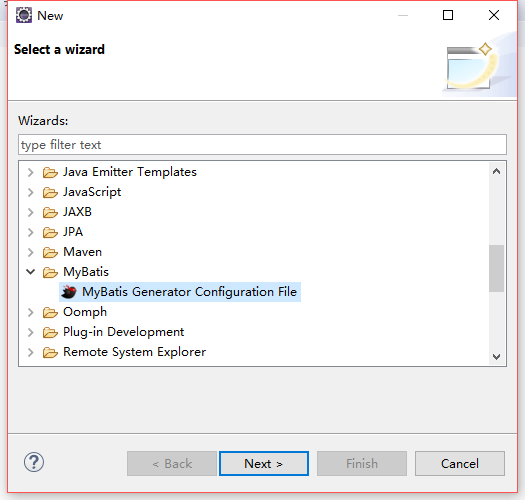
**方法二（带插件）：**

在Eclipse 使用mybatis generator插件自动生成代码：

插件下载地址: <https://github.com/mybatis/generator/releases>



请下载这个插件，然后解压，将features和plugins文件夹的内容复制到eclipse相应的features和plugins文件夹中，重启eclipse即可。

然后选中添加generatorConfig文件的项目，右键new–>other

创建生成generatorConfig.xml文件，打开然后输入**方法一**的相应generatorConfig.xml文件配置。

**注意：**

1. **如有错误请把targetProject="src"中的src改成您当前的项目名**
2. **配置数据库驱动包文件路径，如上的：mysql-connector文件路径**
3. **配置好需要链接的数据库**
4. **自动生成文件的路径**
5. **需要生成的数据库表和实体类名称**

最后点击generatorConfig.xml文件右键打开选择Generate MyBatis/IBATIS Artifacts按OK即可。- -**《方法二结束！》**

**补充说明（）：**

1. **如果是Maven项目，在POM.XML加入**

<dependency>

<groupId>org.mybatis.generator</groupId>

<artifactId>mybatis-generator-core</artifactId>

<version>1.3.5</version>

</dependency>

<dependency>

<groupId>org.mybatis.generator</groupId>

<artifactId>mybatis-generator-maven-plugin</artifactId>

<version>1.3.5</version>

</dependency>

1. **generatorConfig.xml 文件配置如下：**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE generatorConfiguration

PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"

"http://mybatis.org/dtd/mybatis-generator-config\_1\_0.dtd">

<generatorConfiguration>

<!-- targetRuntime="MyBatis3Simple":生成简单版的CRUD "MyBatis3":豪华版 -->

<context id="DB2Tables" targetRuntime="MyBatis3">

<!-- 是否去除自动生成的注释：true:是 -->

<commentGenerator>

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

</commentGenerator>

<!-- jdbcConnection：指定如何连接到目标数据库 -->

<jdbcConnection driverClass="com.mysql.jdbc.Driver"

connectionURL="jdbc:mysql://localhost:3366/mybatis?allowMultiQueries=true"

userId="root" password="orcl">

</jdbcConnection>

<!--是否强制转换 -->

<javaTypeResolver>

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

</javaTypeResolver>

<!-- javaModelGenerator：指定javaBean的生成策略 targetPackage="test.model"：目标包名

targetProject="\MBGTestProject\src"：目标工程 -->

<javaModelGenerator targetPackage="com.gdglc.mybatis.bean"

targetProject=".\src\main\java">

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

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

</javaModelGenerator>

<!-- sqlMapGenerator：sql映射生成策略： -->

<sqlMapGenerator targetPackage="com.gdglc.mybatis.dao"

targetProject=".\src\main\resources">

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

</sqlMapGenerator>

<!-- javaClientGenerator:指定mapper接口所在的位置 -->

<javaClientGenerator type="XMLMAPPER"

targetPackage="com.gdglc.mybatis.dao" targetProject=".\src\main\java">

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

</javaClientGenerator>

<!-- 指定要逆向分析哪些表：根据表要创建javaBean -->

<table tableName="tbl\_employee" domainObjectName="Employee"></table>

</context>

</generatorConfiguration>

1. **测试文件如下**

/\*\*

\* 测试MyBatis逆向工程生成文件

\*/

@Test

**public** **void** **testMbg**() **throws** Exception {

List<String> **warnings** = **new** ArrayList<String>();

**boolean** **overwrite** = **true**;

File **configFile** = **new** File("**generatorConfig.xml** ");

ConfigurationParser **cp** = **new** ConfigurationParser(warnings);

Configuration **config** = cp.parseConfiguration(configFile);

DefaultShellCallback **callback** = **new** DefaultShellCallback(overwrite);

MyBatisGenerator **myBatisGenerator** = **new** MyBatisGenerator(config,

callback, warnings);

myBatisGenerator.generate(**null**);

}