

## openTCS 持久化

---

该文件主要记录openTCS二次开发中遇到的问题：  
数据持久化问题

介绍：本次采用的框架组合是：guice，Gradle，MyBatis和MySQL。

首先，本次问题一共需要4份插件的帮助，分别是MyBatis Generator，  
MYSQL驱动，MyBatis Mapper和iBatis（如有请忽略）

整体文件结构如图1.

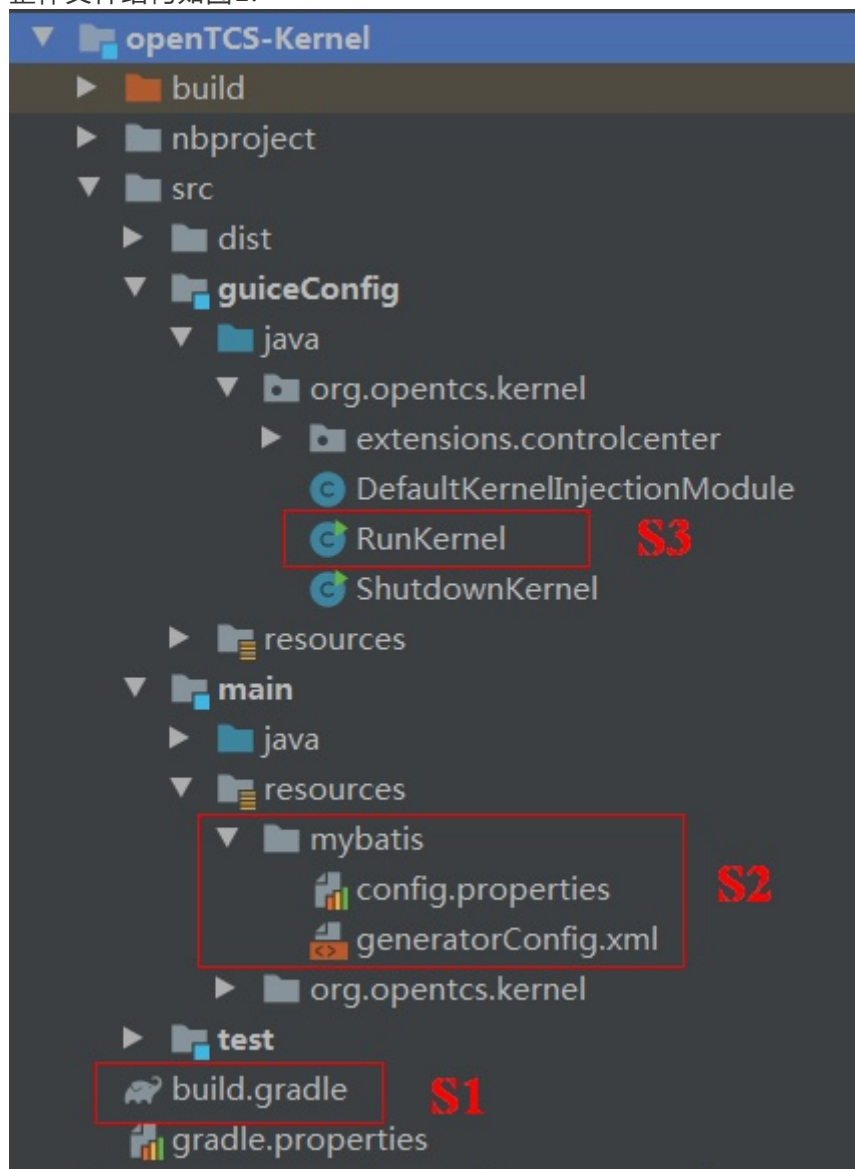


图1 文件结构

## ( 1 ) 配置MyBatis Generator

在build.gradle 首先实现mybatis Generator配置，图1 的S1文件，这部分由代码单独运行，不需要参与编译。具体操作如图2

```

11  //
12  * 配置mybatisGenerator
13  *
14  */
15  configurations {
16      mybatisGenerator
17  }

```

图2 MyBatis Generator 配置

注意：这里，很多blog都有MyBatis Generator 路径等配置信息，如图3所示。经过验证，不需要此操

```

115 // mybatis-generator.xml 配置路径
116 //mybatisGenerator {
117 //    verbose = true
118 //    configFile = 'src/main/resources/mybatis/generatorConfig.xml'
119 //}

```

图3 不需要的配置操作


## （ 2 ）在openTCS-Kernel中build.gradle文件中实现依赖注入。

本次需要的是四份三方插件，在<https://mvnrepository.com> 可以找到。

本次选择的插件具体信息如下：

1.

Home » org.apache.ibatis » ibatis-core » 3.0

 **Ibatis Core » 3.0**  
Ibatis Core


License	Apache
Date	(Apr 18, 2010)
Files	pom (399 bytes)   jar (575 KB)   View All
Repositories	Central   Mulesoft
Used By	9 artifacts

Maven | Gradle | SBT | Ivy | Grape | Leiningen | Buildr

```
// https://mavenrepository.com/artifact/org.apache.ibatis/ibatis-core
compile group: 'org.apache.ibatis', name: 'ibatis-core', version: '3.0'
```

2.

Home » org.mybatis.generator » mybatis-generator-core » 1.3.7

 **MyBatis Generator Core » 1.3.7**  
MyBatis Generator - a code generator for MyBatis.

License	Apache 2.0
Date	(Jul 04, 2018)
Files	jar (658 KB)   View All
Repositories	Central
Used By	145 artifacts

Note: There is a new version for this artifact

New Version1.4.0

Maven | Gradle | SBT | Ivy | Grape | Leiningen | Buildr

```
// https://mavenrepository.com/artifact/org.mybatis.generator/mybatis-generator-core
compile group: 'org.mybatis.generator', name: 'mybatis-generator-core', version: '1.3.7'
```

3.

Home » mysql » mysql-connector-java » 8.0.11

 **MySQL Connector/J » 8.0.11**  
JDBC Type 4 driver for MySQL

License	GPL 2.0
Categories	MySQL Drivers
Organization	Oracle Corporation
HomePage	http://dev.mysql.com/doc/connector-j/en/
Date	(Apr 18, 2018)
Files	pom (1 KB)   jar (1.9 MB)   View All
Repositories	Central
Used By	4,309 artifacts

Note: There is a new version for this artifact


New Version8.0.19

Maven | Gradle | SBT | Ivy | Grape | Leiningen | Buildr

```
// https://mavenrepository.com/artifact/mysql/mysql-connector-java
compile group: 'mysql', name: 'mysql-connector-java', version: '8.0.11'
```

4.

Home » tk.mybatis » mapper » 4.0.4

 **Mapper » 4.0.4**  
Mybatis 通用 Mapper Jar 集成

License	MIT
Date	(Aug 26, 2018)
Files	jar (227 KB)   View All
Repositories	Central   Sonatype
Used By	48 artifacts

Note: There is a new version for this artifact

New Version4.1.5

Maven | Gradle | SBT | Ivy | Grape | Leiningen | Buildr

```
// https://mavenrepository.com/artifact/tk.mybatis/mapper
compile group: 'tk.mybatis', name: 'mapper', version: '4.0.4'
```

图4 插件信息

将图4中红色框中注入语句，添加到build.gradle中，但是需要修改一下，因为前3插件与MyBatis

Generator直接链接。而最后以后是在打包后生成的文件中使用到。具体如下：

```
35      mybatisGenerator 'org.mybatis.generator:mybatis-generator-core:1.3.7'
36      mybatisGenerator 'mysql:mysql-connector-java:8.0.11'
37      mybatisGenerator 'tk.mybatis:mapper:4.0.4'
38      compile group: 'org.apache.ibatis', name: 'ibatis-core', version: '3.0'
```

图5 注入依赖操作

### (3) 设置数据库配置信息

这里主要进行数据库配置，主要修改（没有就创建）图1 的S2文件。文件具体内容如下：

图6主要配置的mysql数据库的基本信息以及打包后生成的文件路径位置设置。包文件路径可以自己根据需求修改。



```
openTCS-4.17.0-src > openTCS-Kernel > src > main > resources > openTCS-4.17.0-src:openTCS-Kernel
config.properties x
1 jdbc.driver=com.mysql.jdbc.Driver
2 jdbc.url=jdbc:mysql://localhost:3306/test?characterEncoding=utf-8&serverTimezone=UTC
3   #?useUnicode=true&useSSL=false&characterEncoding=utf-8&serverTimezone=UTC
4 jdbc.username=
5 jdbc.password=
6 jdbc.driverClassName = com.mysql.jdbc.Driver
7 #mybatis-generator 参数
8
9 # 生成实体类所在的包
10 package.model= org.opentcs.kernel.model
11 # 生成 mapper 类所在的包
12 package.mapper=org.opentcs.kernel.mapper
13 # 生成 mapper xml 文件所在的包, 默认存储在 resources 目录下
14 package.xml=mybatis
```

图6 mysql配置

图7 是数据库表配置信息，具体完整的中文配置详解，见 <https://www.cnblogs.com/ygj1ch/p/6471924.html>

```
4      "http://mybatis.org/dtd/mybatis-generator-config_1_0.dtd">
5  <generatorConfiguration>
6    <context id="Mysql" targetRuntime="MyBatis3Simple" defaultModelType="flat">
7      <commentGenerator>
8        <property name="suppressAllComments" value="true"></property>
9        <property name="suppressDate" value="true"></property>
10       <property name="javaFileEncoding" value="utf-8"/>
11     </commentGenerator>
12
13     <jdbcConnection driverClass="${driverClass}"
14                     connectionURL="${connectionURL}"
15                     userId="${userId}"
16                     password="${password}">
17   </jdbcConnection>
18
19   <javaTypeResolver>
20     <property name="forceBigDecimals" value="false"/>
21   </javaTypeResolver>
22
23   <javaModelGenerator targetPackage="${modelPackage}" targetProject="${src_main_java}">
24     <property name="enableSubPackages" value="true"></property>
25     <property name="trimStrings" value="true"></property>
26   </javaModelGenerator>
27
28   <sqlMapGenerator targetPackage="${sqlMapperPackage}" targetProject="${src_main_resources}">
29     <property name="enableSubPackages" value="true"></property>
30   </sqlMapGenerator>
31
32   <javaClientGenerator targetPackage="${mapperPackage}" targetProject="${src_main_java}" type="ANNOTATEDMAPPER">
33     <property name="enableSubPackages" value="true"/>
34   </javaClientGenerator>
35
36   <!-- sql占位符，表示所有的表 -->
37   <table tableName="%">
38     <generatedKey column="epa_id" sqlStatement="Mysql" identity="true" />
39   </table>
40 </context>
```

图7 配置信息

## ( 4 ) Gradle设置Ant Task

因为MyBatis Generator不支持Gradle，需要使用Ant Task进行代替。这里动用的还是build.gradle文件

```

def getDbProperties = {
    def properties = new Properties()
    file("src/main/resources/mybatis/config.properties").withInputStream { InputStream inputStream ->
        properties.load(inputStream)
    }
    properties
}

task mybatisGenerate << {
    def properties = getDbProperties()
    ant.properties['targetProject'] = projectDir.path
    ant.properties['driverClass'] = properties.getProperty("jdbc.driverClassName")
    ant.properties['connectionURL'] = properties.getProperty("jdbc.url")
    ant.properties['userId'] = properties.getProperty("jdbc.username")
    ant.properties['password'] = properties.getProperty("jdbc.password")
    ant.properties['src_main_java'] = sourceSets.main.java.srcDirs[0].path
    ant.properties['src_main_resources'] = sourceSets.main.resources.srcDirs[0].path
    ant.properties['modelPackage'] = properties.getProperty("package.model")
    ant.properties['mapperPackage'] = properties.getProperty("package.mapper")
    ant.properties['sqlMapperPackage'] = properties.getProperty("package.xml")
    ant.taskdef(
        name: 'mbgenerator',
        classname: 'org.mybatis.generator.ant.GeneratorAntTask',
        classpath: configurations.mybatisGenerator.asPath
    )
    ant.mbgenerator(overwrite: true,
        configfile: 'src/main/resources/mybatis/generatorConfig.xml', verbose: true) {
        propertyset {
            propertyref(name: 'targetProject')
            propertyref(name: 'userId')
            propertyref(name: 'driverClass')
            propertyref(name: 'connectionURL')
            propertyref(name: 'password')
            propertyref(name: 'src_main_java')
            propertyref(name: 'src_main_resources')
            propertyref(name: 'modelPackage')
            propertyref(name: 'mapperPackage')
            propertyref(name: 'sqlMapperPackage')
        }
    }
}

```

图8 ant task操作

至此，持久化部分的操作都已完成。但是还需要将这部分工作加入到内核启动配置中，修改图1的S3文件。如图9



```
openTCS-4.17.0-src > openTCS-Kernel > src > guiceConfig > java > openTCS-4.17.0-src
RunKernel.java x
99     ConfigurationBindingProvider bindingProvider) {
100     List<KernelInjectionModule> registeredModules = new LinkedList<>();
101     for (KernelInjectionModule module : ServiceLoader.load(KernelInjectionModule.class)) {
102         LOG.info("Integrating injection module {}", module.getClass().getName());
103         module.setConfigBindingProvider(bindingProvider);
104         registeredModules.add(module);
105     }
106     return registeredModules;
107 }
108
109 @ private static ConfigurationBindingProvider configurationBindingProvider() {
110     return new Cfg4jConfigurationBindingProvider(
111         Paths.get(System.getProperty( key: "opentcs.base", def: "."),
112             ...more: "config",
113             "opentcs-kernel-defaults-baseline.properties")
114             .toAbsolutePath(),
115         Paths.get(System.getProperty( key: "opentcs.base", def: "."),
116             ...more: "config",
117             "opentcs-kernel-defaults-custom.properties")
118             .toAbsolutePath(),
119         Paths.get(System.getProperty( key: "opentcs.home", def: "."),
120             ...more: "config",
121             "opentcs-kernel.properties")
122             .toAbsolutePath(),
123
124         Paths.get(System.getProperty( key: "opentcs.base", def: "."),
125             ...more: "config",
126             "config.properties")
127             .toAbsolutePath()
128     );
129 }
130 }
```

至此，所有操作完成！