**框架使用指南**

2018年3月7日

1. **Oracle测试环境**

**IP：**[47.98.39.45](https://47.98.39.45:5500/em/shell" \l "/security/show_users)

**账号密码：**

C##wdxc\_admin

wdxc123456

Navcat或plsql连接时，请使用oracle client 12.0.1的版本

1. **开发环境**

JDK版本：1.8

Maven：3.5+

Git

Eclipse或IntelliJ IDEA

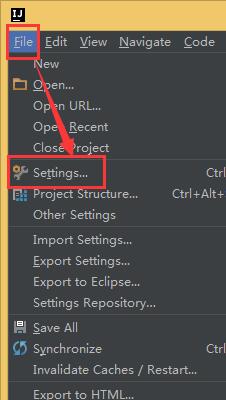
\*注：因为使用了Lomok，请根据不同的开发工具进行Lomok的集成：

Eclipse集成：

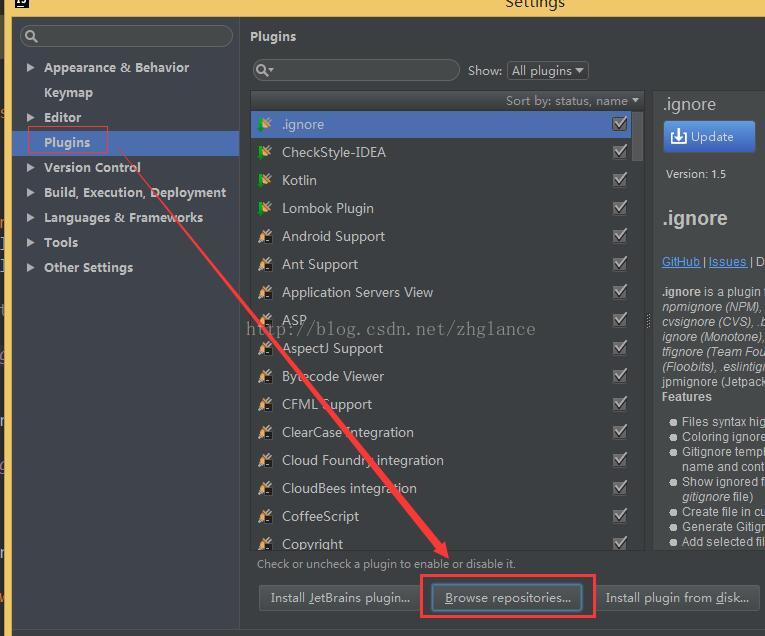
1、下载Lombok.jar <https://projectlombok.org/downloads/lombok.jar>（实际可能地址会失效，根据情况下载）   
2、运行Lombok.jar: java -jar  D:\001\_software\work\Java\libs\lombok.jar   
        数秒后将弹出一框，以确认eclipse的安装路径  
3、确认完eclipse的安装路径后，点击install/update按钮，即可安装完成  
4、安装完成之后，请确认eclipse安装路径下是否多了一个lombok.jar包，并且其  
     配置文件eclipse.ini中是否 添加了如下内容:  
           -javaagent:lombok.jar  
           -Xbootclasspath/a:lombok.jar  
     如果上面的答案均为true，那么恭喜你已经安装成功，否则将缺少的部分添加到相应的位置即可  
5、重启eclipse或myeclipse

IntelliJ IDEA集成：

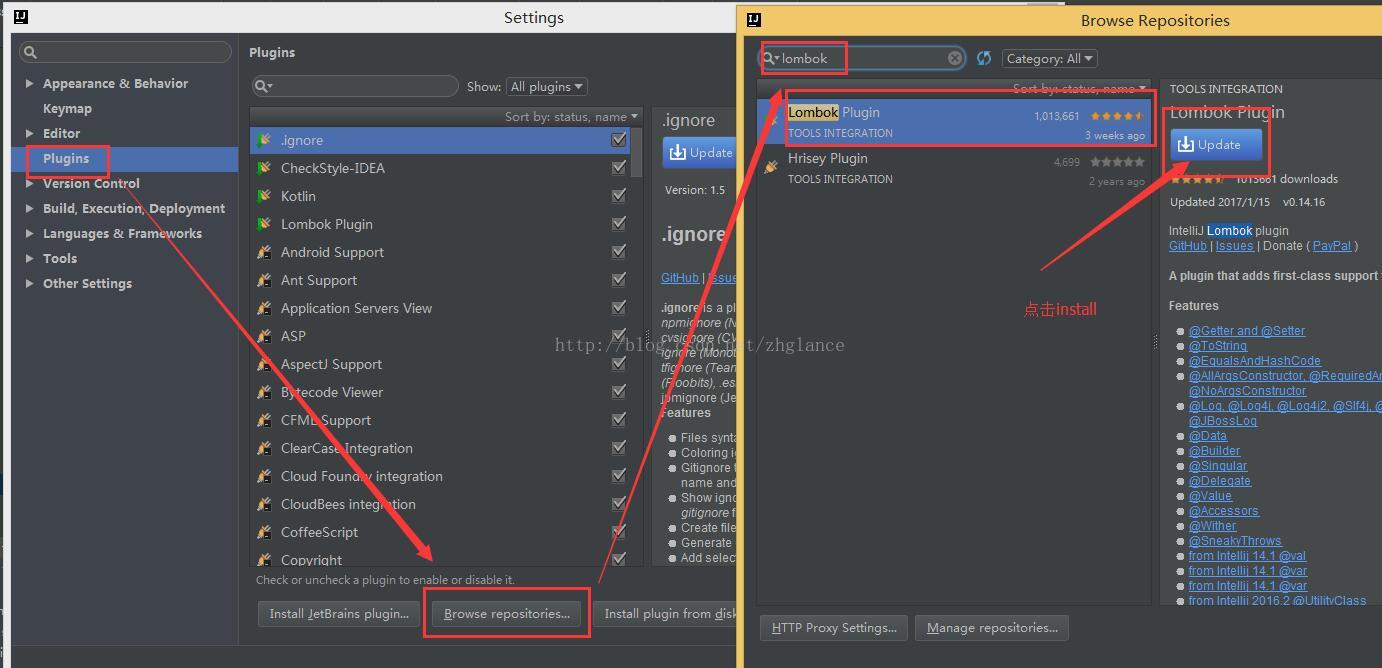
**图一**



**图二**



**图三**



项目中大部分使用的jar包，都是可以从Maven中心仓库下载的，但是不排除部分中心仓库没有，需要独立外部引用，譬如：oracle驱动包，如果项目中有这样的情况，请自行安装额外的jar库文件，Maven安装方式如下：

Mvn install:install-file -DgroupId=com.oracle -DartifactId=ojdbc6 -Dversion=12.1.0.1 -Dpackaging=jar -Dfile=ojdbc6.jar(jar实际路径)

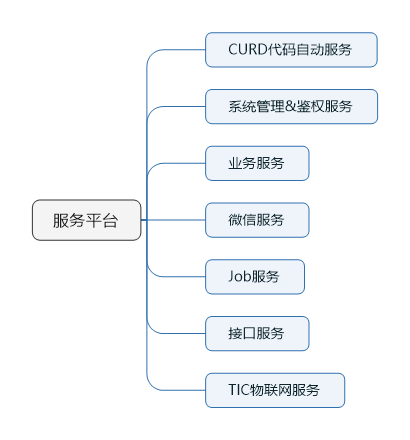
1. **框架技术栈**
2. **Springboot 1.5.6.RELEASE+**
3. **Mybatis**
4. **JWT**
5. **Log4j2**
6. **Lombok**
7. **Druid**
8. **通用Mapper（对于Mybatis基础Mapper的二次封装，按照JPA ORM思想封装的在Mapper工具类）**
9. **Pagehelper**
10. **joda-time**
11. **Quartz**
12. **Netty**
13. **Websocket**
14. **Apache Common\***
15. **weixin-java-\*（微信后台API封装代码，实现了微信大部分功能封装）**

业务服务框架图

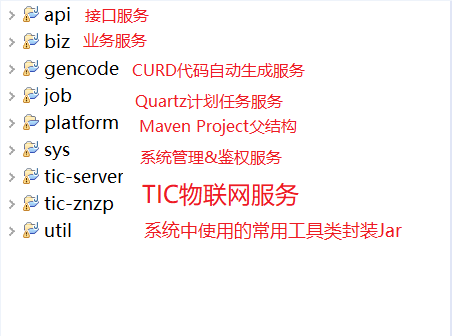


1. **目录结构说明**

*服务平台结构*



*项目目录结构*



说明：

1、接口服务、业务服务、job服务为War项目，属于独立发布

2、TIC物联网服务属于扩展服务，根据项目需求再搭建使用，为War项目，独立发布

3、sys和util为Jar工具包，项目如果需要使用该工具包，再自行引用

4、gencode为代码自动生成工具服务，自动生成后放入项目中，可实现单表CURD无需写一行代码。目前支持Mysql和Oracle数据库。自动生成代码如下：

Mybatis Entity Bean结构

Mybatis Mapper类

Mybatis xml映射文件

Bean对象的CURD Service类

Bean对象的CURD Controller类

1. **使用注意事项**
2. **单表CURD**

如果只需要单表的CURD功能，只需要使用gencode自动生成代码或是自己继承BaseController、BaseService，Mybatis Mapper接口类继承Mapper<T>即可；

Controller请求地址规则，如果需要权限控制需要加上/api/的前缀，如果是公开的可以使用/pub/前缀

\*注：如果操作的表不需要全部的CURD功能，请不要继承BaseController，因为继承了BaseController，就会自动实现CURD功能，以防程序出现漏洞

通用CURD请求地址如下：

/api/XXX/{id}根据主键查询数据；

/api/XXX/getAll查询所有数据；

/api/XXX/getCondition根据前台传递的参数查询指定的数据，前台传的参数和Bean对象里面的字段要一致；

/api/XXX/pager分页查询数据，也可以传参数进行分页查询；

/api/XXX/save新增方法，不自动生成主键，要么数据库自动生成主键内容，要么前台已生成好作为参数传递

/api/XXX/update更新方法

/api/XXX/remove/{pkid}根据主键删除数据

/api/XXX/ removeIds批量根据主键删除数据

1. **按条件查询**

常规按条件查询，只需要前台界面按Bean结构传递参数即可自动完成等值查询；

如果等值查询不能满足时，需要使用Example对象进行自定义查询，具体代码如下：

SimpleCondition condition = new SimpleCondition(SysZdxm.class);

condition.like(SysZdxm.InnerColumn.zdlmdm, value);

List<SysZdxm> results = zdxmMapper.selectByExample(condition);

SimpleCondition为封装的Example对象，SysZdxm为Mybatis的Bean对象，SysZdxm.InnerColumn的InnerColumn为gencode自动生成代码参数的一个Enum对象，主要目的是bean结构和数据库字段直接的一个转换类

LimitedCondition 是 SimpleCondition 的封装，自带机构代码查询条件，根据当前登录用户后去机构代码

BaseServiceImpl 类中有getQueryCondition方法，该方法约定在查询字段添加后缀的行使定义了一些查询条件：

In$ ：in 查询条件

Like$：like查询条件

Gte$：greater than or equals 查询条件

Lte$：less than or equals 查询条件

InRange$：between查询条件

IsNull$：is null查询条件

IsNotNull$：is not null查询条件

orderBy： order by 字段以及排序方式（例如：create\_time desc）

fillCondition 是getQueryCondition的补充，用于在约定添加后缀查询方式尚不足以满足要求的情况下使用，该方法传入参数为LimitedCondition，可根据需要添加LimitedCondition的查询条件

BaseServiceImpl类中有genId 方法，可生成主键

getOperateUser 可获取操作人信息，返回值格式为：[用户id]-[用户姓名]

findEq(String key, Object val)，效果等同于：select\*from table1 where key =val;

findByIds(Iterable iterable),效果等同于：select\*from table1 where id in(id1,id2,id3,id4)

1. **获取当前登录用户信息**

Controller类如果继承了BaseController，使用getCurrentUser()方法即可；

Service类如果继承了BaseService，使用getCurrentUser()方法即可，方法需要记录操作员信息时，使用getOperateUser()方法即可

1. **REST返回处理**

所有的Controller返回结果，统一为ApiResponse<T>

1. **异常处理**

业务过程中如果需要抛出异常，请使用项目中业务异常类抛出异常（RuntimeCheckException），在业务异常抛出后，事物回滚

1. **文件日志处理**

文件日志记录分为几种类型：

一般日志记录（info）：如果只是一般记录，可在对应的类引用注解：@Slf4j，方法里面使用log.debug即可

数据日志记录（access\_info）：对于需要记录请求参数用于以后数据分析，需要自己定义Logger类：Logger accessLog = LoggerFactory.getLogger("access\_info");

错误日志记录（error\_info）：对于需要记录错误日志，分级独立出错误日志，便于日后好查看，需要自己定义Logger类：Logger errorLog = LoggerFactory.getLogger("error\_info");

1. **Mybatis二级缓存使用**

Mybatis二级缓存使用Redis缓存，经常查询的表，数据更新不频繁的情况，建议使用二级缓存来处理。

在对象的Mapper类添加注解：

@CacheNamespace(implementation=MybatisRedisCache.class)

MybatisRedisCache为框架扩展的Redis缓存

1. **Redis使用**

常规String相关操作，使用StringRedisTemplate；

如果是存储对象，使用自定义的RedisTemplateUtil

1. **手动加载Bean，非自动注入方式**

如果需要手动加载Bean或是有的类加载无法通过@Autowired加载到Bean，可以使用SpringContextUtil.getBean(Class)

1. **连接Oracle12C方法**
   1. 使用ojdbc7.jar驱动包，自行安装到本地maven库，安装方法参考第一章中的介绍
   2. DataSource的URL配置如下：jdbc:oracle:thin:@//47.98.39.45:1521/orcl

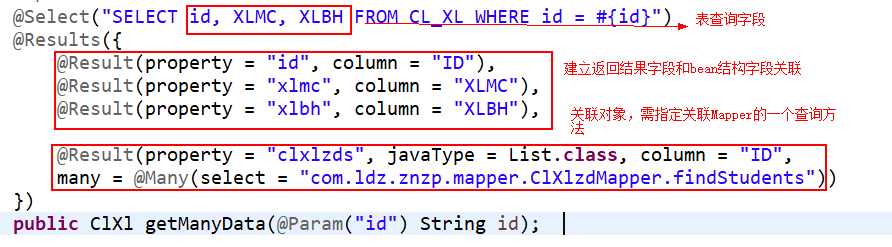
注意：oracle12c连接中间使用的是’/’，非’:’。

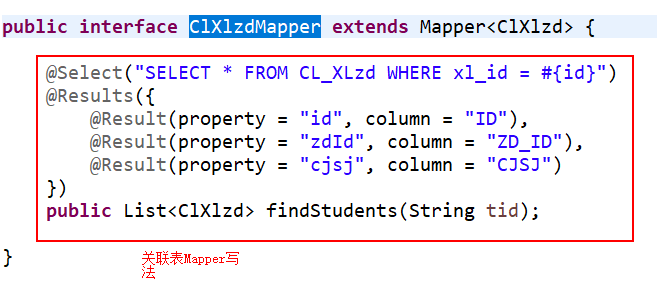
* 1. Oracle12c因为加入新规则，分CDB和PDB两种，我们创建的用户在PDB库中，PDB用户名规则前缀固定是’C##’，所以用户名形式如：’C##xxxx’，因为用户名中有特殊字符，所以在配置文件中使用用户名的时候，注意一定要加引号，如：username: 'C##wdxc\_admin'

1. **@One或@Many一对一、一对多关系**

使用注解方式在Mapper中实现一对一或一对多处理：

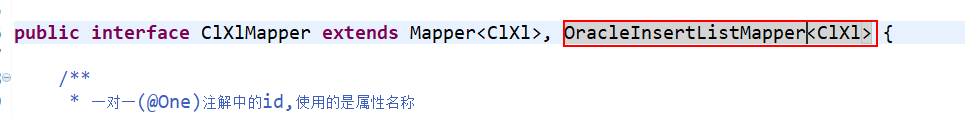
* 一对一(@One)注解中的id,使用的是属性名称
* 一对多(@Many)注解中的id,是你本对象的主键属性名称





1. **批量插入提交数据**

Oracle批量新增方法，使用如下：



在对应的Mapper类中继承：OracleInsertListMapper<T>即可，使用方法：mapper.insertList(List<T> datas)

MySQL字段为自增方式，使用如下：

在对应的Mapper类中继承：InsertListMapper<T>即可，使用方法：mapper.insertList(List<T> datas)

1. **常用工具类**
2. 如果只需要单表的CURD功能，只需要使用gencode自动生成代码或是自己继承BaseController、BaseService，Mybatis Mapper接口类继承Mapper<T>即可；
3. 日期时间转换等操作，使用org.joda.time.DateTime类进行操作；
4. 网络数据请求操作，使用com.ldz.util.commonUtil.HttpUtil类进行操作；
5. GPS相关操作（计算两点距离，GPS点是否在电子围栏内，计算路线到达时间等）类，使用：com.ldz.util.gps.\*；
6. String相关的验证方法，请使用Common-lang的工具类StringUtils;

**\*框架持续完善中，大家可以把自己平常使用过程中改进的方法或类进行补充完善，完成平台框架的真正便捷省时省事**