一、命名规范

1. 文件命名要以英文（不会拼的有道，谷歌之类的查查都行）为准，尽量简单明了，但是不要缩写，原则中文全拼也不能出现

2. 类名首字母大写，方法名和属性名首字母小写，统一遵循驼峰标识，常量统一大写用下划线进行分隔，并用 final 进行修饰

3. 接口类中的方法和属性不要加任何访问权限修饰符，尽量保持代码整洁；抽象类的属性或者方法如果只有子类能访问要用protected修饰

4. 新增的业务领域模型如没有特殊需求，直接继承基类，并尽量调用基类现有的方法或者属性

二、应用分层

1. Entity（or DO）（可根据需求继承基类 \*IdentityEntity\*）, 字段如果只有几个固定的值，尽量采用枚举类型；数据库建表统一小写，并根据驼峰标识用下划线分隔，字段映射规则写在get方法上。如：JAVA字段名：private String carId，则数据库字段名：car\_id

\* 字段注释必须加！！！字段注释必须加！！！字段注释必须加！！！

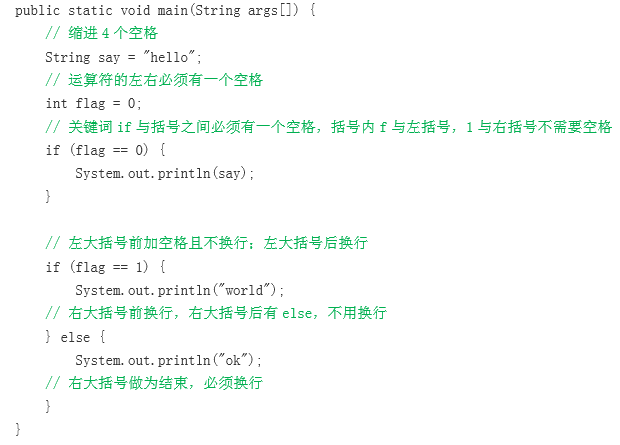
2. Dao （目前项目数据访问层基本采用JPA的形式继承 PagingAndSortingRepository 即可）。

3. Service （基类 BaseJpaService），一般情况下使用默认的方法即可。

4. Controller（基类 EnvelopRestEndpoint），该基类封装了绝大部分日常使用的返回结果为rest风格的公共泛型方法，最基础的数据包含信息和状态码，扩展的数据包含列表、分页、单个对象、以及以上全部。请根据需要返回对应数据模型，不用因为省事，所有接口都返回同一个类型的Envelop，例：{@link com.yihu.jw.base.endpoint.function.FunctionEndpoint}。

二、格式规范

1. 参考以下（缩进使用tab即可）



2. 调用方法的时候参数之间要有空格 service.method("a", "b", "c");

三、控制语句

1. 控制语句的大括号必须携带，不要用 if (condition) execute; 的形式;

2. if 语句能解决的问题，不必使用else， 如 if (condition) { ... return obj; }

3. switch 语句要用break或者return防止穿透，并以default结尾

四、注释规范

1. 非常用的新增方法名必须要加注释，注释使用 /\*\* content \*/ 的形式

2. 方法内的单行注释使用//，多行注释使用 /\* content \*/

3. 修改原有代码的时候要加修改时间，修改者的相关信息注释

4. 特别需要注意的是，拷贝的代码，如果与现在实现的业务逻辑不相符，原有的注释、(方法名、参数名)一定要修改

五、异常处理

1. 可控的运行时异常一定要尽量避免，如空指针异常，类型转换异常

2. 项目中目前在Controller层定义了一个全局错误处理器，只需要引入依赖common-web即可使用，该处理器会在捕获异常时将错误信息封装成Envelop对象作为接口的返回结果，对于不需要进行特殊处理的异常，请都将其直接抛出，方便以后的异常结果二次处理，并减少各位开发时的代码量。

3. 如果进行 try catch 的话，一定要将异常的堆栈信息打印出来，e.getMessage()能获取到的信息太少，不好定位错误

六、日志处理

1. 平台的日志统一用logback进行处理

2. 引入common-web即可使用通用的日志处理配置，该配置目前将接口访问异常的日志（仅限于发生异常且直接抛出的情况下）保存在/wlyy-logs/${spring.application.name}/api/目录底下，业务需要的日志（logger的包名必须以com.yihu.jw开头）保存在/wlyy-logs/${spring.application.name}/business/目录底下。如果该配置不能满足业务需求，请在相应的项目底下增加logback-spring.xml覆盖默认的配置即可。

3. 大量输出无效日志，不利于系统性能提升，也不利于快速定位错误，记录日志的时候请思考：这些日志是否会有人看？看到这条日志能用来做什么？能不能给问题的排查带来好处？

七、其他

1. 操作大量字符串拼接的时候用StringBuilder的append()方法， 不要用 str += str1;

2. <groupId>com.yihu</groupId>

<artifactId>common-utils </ artifactId >

有平常常用的一些公共辅助类可供使用

3. 删除数据的时候一定要检查关联表是否存在相关数据，如果存在要一并删除，避免数据库存在垃圾数据