**后端开发规范文档**

后端编码全部统一采用“UTF-8”编码，基于面向接口编程，要求代码保持干净整洁，去除没有用到的类引用，尽量消除编辑器出现的各种警告。

# 命名规范

## 1.1通用规范

* 类、变量、方法命名应具有见名知意快速地传达其用途，不要过度缩写，类和变量名推荐使用名词，方法名使用“命令性”动词
* 无特殊指明下，多个单词组成需要采用驼峰命名法(首个单词的首字母小写，其后单词的首字母大写)
* 禁止使用中文命名，拼音与英文混合命名，美元符号

## 1.2包命名规范

* 包的名字全部是小写字母
* 包名按照域名的范围从大到小逐步列出，与域名命名规则相反，用”.”连接的标识符构成

## 1.3类(接口)命名规范

* 类的名字必须由大写字母开头而单词中的其他字母均为小写
* 接口的实现类命名必须在接口名的后面加上“Impl”，例，接口名为：UserService.java，其实现类命名为UserServiceImpl.java
* 抽象类 使用Abstract 或 Base 开头
* 异常类命名使用 Exception 结尾
* 测试类 命名使用 Test结尾

## 1.4变量命名规范

* 全局变量命名时要符合 访问权限 + 变量类型 + 变量名，变量名必须小写字母开头，禁止使用is开头，变量类型如果为基本类型使用包装类型，例:Integer，Long等。不要设定默认值
* 局部变量应控制尽可能小的作用域中
* 临时变量使用下划线开头

## 1.5常量命名规范

* 常量名称必须全部大写，以下划线分隔单词，例：MIN\_VALUE

## 1.6枚举命名规范

* 枚举名称必须全部大写，以下划线分隔单词
* 命名以Enum后缀

## 1.7方法的命名规范

* 方法的名字的第一个单词应以小写字母作为开头，后面的单词则用大写字母开头。可以为动词或动词+名词组合。
* 所有的覆写方法，必须加@Override注解

## 1.8数组命名规范

* 推荐使用String[] args方式，忌讳使用String args[]方式

# 注释规范

内容要简单、明了、含义准确，防止注释的多义性，错误的注释不但无益反而有害，另外，为了保持注释风格的统一，推荐使用中文注释，以防止英文误写或理解歧义

## 2.1类（接口）注释

/\*

\* 类详细说明

\* @author: 姓名

\* @version: 版本

\* @date: yyyy-MM-dd

\*/

## 2.2成员变量注释

/\*\* 字段含义 \*/

## 2.3方法注释

/\*

\* 方法的详细使用说明

\* @param 参数

\* @return ${return\_type}

\* @author: 姓名

\* @version: 版本

\* @date: 日期

\*/

## 2.4代码段注释

// 注释说明

## 2.5构造方法注释

/\*

\* 方法的详细使用说明

\* @param 参数

\* @author: 姓名

\* @version: 版本

\* @date: 日期

\*/

# 排版规范

## 3.1代码缩进

程序块要采用缩进风格编写，缩进长度为4个空格缩进

## 3.2空行空格

* 相对独立的程序块之间、变量说明之后必须加空行
* 任何运算符左右须加一个空格

## **3.3大括号的使用**

if、for、do、while、switch等语句自占一行，且 if、for、 do、while等语句的执行语句部分无论多少行都要加括号{}，如果是大括号内为空，则简洁地写成{}即可

如果大括号内是非空代码块则，左大括号前不换行后换行，右大括号前后都换行

# 日志规范

**日志不仅对开发人员有意义，也对日志阅读者运维有意义，对错误的跟踪和代码的调试有重大意义**

* 当程序产生异常时，必须捕捉并处理异常、将异常记录到日志中（除非向上抛出异常）。
* 不要处理能够避免的异常，这是一种隐藏异常的行为，虽然程序不会报错，但是逻辑却可能错了或者存入不完整数据到数据库
* 在异常处理模块中提供适量的错误原因信息，记录应言简意赅
* 处理异常块中不要存在printStackTrace()方法

# 5.git提交日志规范

## 5.1提交原则

* 单点提交

一次提交只提交一个‘点’的修改，不要包含不相关的修改，否则git日志无法清楚的描述修改操作，代码比对麻烦。

* 完整提交

每次提交尽量保证修改的完整性，可用性；不要分多次提交同一次修改的内容；如果提交时出现特殊情况分开提交，则后续的提交在日志中应该反馈。

* 阶段提交

提交的内容在满足其他原则的基础上尽可能的少，满足最新功能需要即可。

## 5.1提交格式

**git提交格式统一使用如下描述：**

描述: [类型]具体描述内容

类型: 选择项有，新增、修改、删除、优化、撤销、补遗

具体描述内容:描述此次提交的原因

# DAO数据访问层规范

* Dao层只写接口不写实现,实现由MapperScan自动扫描生成
* Dao层接口以Dao结尾,命名方式：表名转换为驼峰+Dao，例:UserDao
* Dao层接口推荐继承BaseDao，不强制要求继承BaseDao

# Mapper编写SQL语句层规范

* Mapper层以Mapper结尾,命名方式:表名转换为驼峰+Mapper.xml例:UserMapper.xml
* \*Mapper.xml文件中的namespace 必须与相应的Dao接口相对应

# Service业务层规范

* Service层接口以Service结尾,命名方式:表名转换为驼峰+Service 例:UserServie.java
* Service层接口实现以ServiceImpl结尾，命名方式:表名转换为驼峰+ServiceImpl 例:UserServieImpl.java
* 查询类的都要以find或者get开头，返回结果为list的要以List结尾,例:findList,getList 查询方法参数使用map传递参数
* 添加使用insert或者save作为前缀，批量添加使用saveBatch作为前缀
* 修改使用update或者merge作为前缀，批量添加使用updateBatch作为前缀
* 删除使用delete或者remove作为前缀，批量添加使用deleteBatch作为前缀
* 导入使用import作为通用单词
* 导出使用export作为通用单词

# Controller访问控制层规范

* Controller层以Controller结尾,命名方式：表名转换为驼峰+Controller,例:UserController.java
* 推荐继承BaseController
* Controller层Mapping映射路径类上和方法上都以“/”开头,不能以“/”结尾,

例:/user,不推荐写法:/user/