类规范

\* 方法，行数，有效代码行数不超过60行（如果超过了，说明逻辑太多，需要拆分了）

\* 方法，参数，无用的形参要删除，参数超过3个的必须给每一个参数写注释

\* 方法，参数，参数不超过6个(超过的使用bean封装)，不能使用Map代替bean传递参数，controler方法除外

\* 类，行数，有效代码不超过1000行（如果超过了，说明这个类需要拆成2个或以上的类）

--新项目必须符合这个要求；老的项目可以不去改它，但新增方法的时候，必须考虑新设计一个类

\* 注释，创建方法或重构方法时必须写注释，注释必须简洁明了，不能描述不全，不能有歧义

\* 注释，其对应的类或方法功能发生改变，必须更新此注释（场景：方法的参数或返回值的变更，类增加父类或接口）

\* 注释，每个类新建的时候注明Created by {who}

\* 代码结构上，层次，层次要分离，具体分离规则参考 新的系统架构

\* 代码结构上，调用，相同层次的类不允许相互调用，如有调用需要，可考虑降低被调用的代码的设计层次

\* 代码结构上，功能，一个类的功能要明确且简单，一个方法也同样在某一级的处理上只干一件事

\* 书写规范，格式化，不要格式化他人的代码，避免不必要的合并冲突；但自己新加的代码段需要要格式化好，方便他人review

\* 书写规范，易读，每个方法使用一个空行隔开（多个空行的情况看到了请随手删除到一个）

**命名规范(单独拿出来，以显其重要)**

\* 命名必须含义清晰、准确，避免使用双关语，不能含有个人喜好词汇

\* 变量，驼峰命名方式，如：accessAuthorityService（也适用于Python，Scala 等）

\* 常量，大写，下划线分割单词 如：final int PAGE\_SIZE = 60

\* 定义的URL，小写，下划线分割单词 如：get\_votelist

\* url的参数名，小写，下划线分割单词，尽量简写，比如：tel，cla\_id

\* 数据库字段，小写, 下划线分割单词，注释要简洁全面（其他数据库规范请单独参考 MySQL数据库使用规范）

数据存取规范（mysql，redis，mongo，memcached）

\* 所有的写入用主，所有的读取用从（如果有特殊的使用主读取的，需要单独说明并审核）

\* 可能超过10次的循环里不能有存取操作，单独走批量存取

\* 存储的数据大小要可知，并且确定不会超过上限

\* sql查询语句每次查询条件的改动，必须explain分析，杜绝慢查询上线

日志规范

\* debug

\*\* 容易出问题的逻辑，记录需要debug的信息。

\* info

\*\* 有效信息的输出，比如重要update，delete操作。

\* warn

\*\* 有错误出现，并且机器能够自己处理，不需要人工干预。但是需要告知开发人员的信息。

\* error

\*\* 有严重错误出现，机器无法处理，需要人工处理才可以正常使用。

推荐及目标：

\* 不滥用异常（不等于不使用）

\* 不滥用日志输出

\* 日志应清晰有调理