**代码评审标准：**

|  |  |  |  |
| --- | --- | --- | --- |
| 评审对象 | |  | 作者 |
| 评审人 | |  | 评审工作量 |
| 程序结构组织 | | | |
| 1 | 代码结构清晰，外观整齐 | | |
| 2 | 所有的函数以及接口定义清晰 | | |
| 3 | 整个代码体系结构组合合理 | | |
| 4 | 模块中所有的数据结构都定义为局部的，并且通过定义好的函数进行访问 | | |
| 5 | 接口模块化，修改时不影响其他代码模块 | | |
| 6 | 提供了处理数据的策略 | | |
| 7 | 数据库操作、IO操作等是否正确关闭资源 | | |
| 8 | 多处使用相同代码时，应定义唯一方法或变量以供使用。 | | |
| 9 | 导入类时，如果仅使用包中的几个类，应导入具体类，而不是导入整个包。 | | |
| 10 | 检查代码和详细设计是否一致。 | | |
| 11 | 检查类中是否有无效的代码或者是无用的代码。 | | |
| 12 | 不要使用System.out.print()输出，需要进行日志处理 | | |
| 13 | 所有的文件名符合文件命名规范，见名知意 | | |
| 14 | 文件和模块分组清晰 | | |
| 15 | 所有的代码行在80字符以内 | | |
| 16 | 每个程序文件都小于2000行 | | |
| 代码组织 | | | |
| 1 | 数据库查询语句不要出现select \* | | |
| 2 | 对需要处理的字符串定义为StringBuffer ，常量定义成静态的。 | | |
| 3 | 所有的变量名都小于32字符 | | |
| 4 | 每行代码都小于80字符 | | |
| 5 | 类名、方法名、变量名的命名是否规范。 | | |
| 6 | 有返回值的方法是否正确返回。Return语句应定义在方法结尾处。 | | |
| 7 | 代码排版是否规范。 | | |
| 8 | 所有的行每行最多只有一句代码或一个表达式 | | |
| 9 | 复杂的表达式具备可读性 | | |
| 10 | 续行缩进 | | |
| 11 | 括号在合适的位置 | | |
| 12 | 每个顺序的小块用空行隔开 | | |
| 13 | 注释和代码对齐或接续在代码之后 | | |
| 14 | 代码的逻辑是否有问题。 | | |
| 15 | 和CSS除非必要就放到外部的单独文件中，在JSP中引用。 | | |
| 16 | 连接数据库的方法必须调用公用的方法。 | | |
| 17 | 异常要统一处理，异常处理方法是否符合项目组的约定 | | |
| 18 | 显示的文本无拼写和语法错误 | | |
| 19 | 代码与操作符的优先级无关 | | |
| 20 | 所有的表达式使用了正确的操作符 | | |
| 函数组织 | | | |
| 1 | 代码中有相关注解 | | |
| 2 | 函数的名字清晰的定义了它的目标以及函数所做的事情 | | |
| 3 | 函数中所有的部分都合理的组成函数，相关独立的语句组组成函数 | | |
| 4 | 参数遵循一个明显的顺序； | | |
| 5 | 所有的参数都被使用 | | |
| 6 | 函数的参数接口关系清晰 | | |
| 7 | 如果一个函数有返回值，在所有的出口都有返回值 | | |
| 8 | 函数的参数个数小于7个 | | |
| 9 | 使用的算法说明清楚 | | |
| 10 | 函数检查了输入数据的合法性 | | |
| 11 | 函数异常处理清楚 | | |
| 12 | 代码检查调用函数的返回值，参数和调用匹配 | | |
| 13 | 递归定义了出口 | | |
| 14 | 递归局限于一个函数 | | |
| 15 | 对接口的调用必须要有判断传入的参数不能为空和传入的参数正确。 | | |
| 数据类型与变量 | | | |
| 1 | 数据类型存在数据类型解释 | | |
| 2 | 代码避免了重新定义预先定义的数据类型 | | |
| 3 | 静态变量明确区分 | | |
| 4 | 每一个变量都初始化了 | | |
| 5 | 所有的变量都有最小的活动范围 | | |
| 6 | 所有的全局变量都描述清楚 | | |
| 7 | 所有的变量都用到了 | | |
| 条件判断 | | | |
| 1 | 条件检查和结果在代码中清晰 | | |
| 2 | If/else 使用正确 | | |
| 3 | 判断的次数降到最小 | | |
| 4 | 最常用的情况最先判断 ，所有的情况都考虑 | | |
| 5 | 判断条件使用引用时，应先判断其是否为null。 | | |
| 循环 | | | |
| 1 | 不要出现三层以上的for循环。如有必要必须写清楚注释 | | |
| 2 | 循环体不为空，循环体能够一次看清楚 | | |
| 3 | 注意while循环和for循环的使用情况 | | |
| 4 | 代码中不存在无穷次循环 | | |
| 5 | 循环体内的循环变量起到指示作用。 | | |
| 6 | 所有的循环边界是否正确； | | |
| 7 | 循环终止的条件清晰 | | |
| 8 | 循环条件、判断条件的边界值，数组、集合下标是否越界。 | | |
| 注释 | | | |
| 1 | 类、方法、判断条件、循环、主要变量是否添加注释。 | | |
| 2 | 类名、方法名、变量名的命名是否规范。 | | |
| 3 | 源代码能够自我解释 ，每个人看到代码就能很快理解 | | |
| 4 | 注解清楚正确 ，注解为用户服务 | | |
| 5 | 类、方法、变量等注释是否符合规范，注释是否能够清晰明了 | | |
| 6 | 对if和for、while等要进行必要的注释，并且方法体都要用{ }包起来 | | |
| 7 | 代码的注释不能过于简单。要清晰明了，提交的时候必须写清楚修改的功能点和为什么要修改。 | | |
| 8 | 注释是否较清晰且必要 | | |
| 9 | 函数是否已经有文档注释 |  |  |
| 总括 | | | |
| 1 | 代码直观 | | |
| 2 | 无用的代码、注解已经删除 | | |
| 3 | 除非必要，不要再循环体内定义对象。 | | |
| 4 | 代码是否与详细设计一致。 | | |
| 其他注意项 | | | |
| 1 | 尽量减少同名的类 | | |
| 2 | 不要在循环中反复定义创建变量 | | |
| 3 | 不要在循环中使用复杂的计算 | | |
| 4 | 在sql中注意敏感的单词要回避使用 | | |
| 5 | 代码与操作系统无关，不需要任何假设条件 | | |
| 特殊 | | | |
| 1 | 所有的数组访问在它们的边界内 | | |
| 2 | 类型转换明确指明 | | |
| 3 | 所有常量定义和使用替代代码中的数字 | | |
| 4 | 一个方法体的代码幅应该在一屏比较和合理；逻辑复杂的代码可以抽离出方法体 | | |