

基于B/S架构的在线数独对战平台

代

码

规

范

说

明

课 程： 软件工程

题 目： OurSudoku

成 员： SE-2020-G02

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [√] 正式发布  [ ] 正在修改 | 文件标识： | SE2020-G02-软件代码规范说明 |
| 当前版本： | 1.0.0 |
| 作者： | 刘羽佳、张鑫、潘言 |
| 完成日期： | 2021.01.20 |

**版本记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 版本 | 作者 | 参与者 | 审核人 | 起始日期 | 备注 |
| 0.0.1 | 潘言 | 潘言 | 刘羽佳 | 2020.12.20 |  |
| 0.1.0 | 潘言 | 潘言 | 刘羽佳 | 2021.01.19 |  |
| 1.0.0 | 潘言 | 刘羽佳 | 刘羽佳 | 2021.01.20 |  |

目录

[1.引言 4](#_Toc23128)

[1.1标识 4](#_Toc25042)

[1.2系统概述 4](#_Toc13095)

[1.3文档概述 4](#_Toc16783)

[2.代码规范细则 4](#_Toc8718)

[2.1排版要求 4](#_Toc25755)

[2.1.1程序块缩进 4](#_Toc6988)

[2.1.2程序块之间空行 5](#_Toc15583)

[2.1.3长语句以及长表达式 5](#_Toc1315)

[2.1.4语句对齐 6](#_Toc27539)

[2.2命名 6](#_Toc26620)

[2.2.1类命名 6](#_Toc23363)

[2.2.2方法、参数、变量命名 7](#_Toc8410)

[2.2.3常量命名 7](#_Toc9987)

[2.2.4常量使用 7](#_Toc12335)

[2.3注释 7](#_Toc8290)

[2.3.1变量、常量注释 7](#_Toc20627)

[2.3.2方法注释 8](#_Toc4569)

[2.4格式 8](#_Toc7143)

[2.5锁与并发 9](#_Toc2506)

[2.6try与异常 9](#_Toc31471)

[2.8数据库 10](#_Toc6723)

[2.8.1 建表规约 10](#_Toc14957)

[2.8.2 SQL语句 11](#_Toc7210)

[2.8.3 ORM映射 11](#_Toc14901)

[2.8测试 12](#_Toc14461)

[3.复用 12](#_Toc3549)

[style复用：前端开发中如果遇到了某几个组件调用了相同的style则应将style写在<style>中，以class=””的形式获取 12](#_Toc20675)

[4.注解 12](#_Toc25588)

# 1.引言

## 1.1标识

标题：《软件代码规范说明》

版本号：1.0.0

说明：本文档支持Office、WPS等word文档阅读软件。

## 1.2系统概述

1. **软件概述**

本小组开发的数独在线对战平台是一款能够让数独爱好者之间互相PK的平台，除了最基本的房间对战功能，我们还开放了天梯排位模块、用户出题模块、悬赏模块以及自主练习模块。

## 1.3文档概述

本文档的内容包括了开发过程中前端与后端的代码规范。

# 2.代码规范细则

## 2.1排版要求

### 2.1.1程序块缩进

所有的前端与后端代码都要按照父子关系进行缩进。



### 2.1.2程序块之间空行

不同的程序块之间要以一个空行作为分隔。



### 2.1.3长语句以及长表达式

在语句或表达式过长时需要加入适当的换行。



如果一个标签里的属性多于或等于3个，则需要换行显示。

### 2.1.4语句对齐

同级表达式的缩进格式要相同。



## 2.2命名

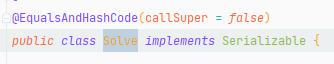
在对任何一个模块进行命名时需要遵循以下规则：

1. 禁止出现中文拼音，国际规范除外（如HangZhou）

2. 禁止以\_$作为开始或结尾

### 2.2.1类命名

1. 类名采用UpperCamelCase风格（如UserInfo）
2. 抽象类用Abstract开头，异常类用Exception结尾



### 2.2.2方法、参数、变量命名

1. 方法名、参数名、变量名采用lowerCamelCase风格（如userPwd）。



1. Boolean类型变量不要加is前缀。
2. 获取单个对象用get作为前缀。
3. 获取多个对象用list作为前缀。
4. 统计方法用count作为前缀。
5. 插入方法用insert作为前缀。
6. 删除方法用delete作为前缀。
7. 修改方法用update作为前缀。

### 2.2.3常量命名

常量命名所有字母大写，单词间用\_分隔，必须明确表达常量含义（如ID\_LENGTH）。

常量的要求是“见词知意”，因此不要担心常量的名字过长，要保证在能够明确表达常量含义的前提下再尽可能缩短长度。

### 2.2.4常量使用

在使用常数Long时，用L而不是l（小写），否则会混淆21和2l（L）。

## 2.3注释

1. 单行注释在被注释语句上一行用//单独开一行注释，多行注释用/\*\*/。
2. 注释可以用中文，但代码内定义的变量名等需要用英文表示。

### **2.3.1变量、常量注释**

在注释长度较短的情况下可以将注释添加在变量、常量的末尾，如果长度过长影响了阅读，则应该把注释加在定义的上一行。



### **2.3.2方法注释**

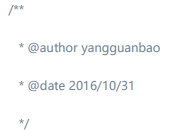
1. 不论注释的长短，注释都应加载方法的定义上方。



1. 覆写方法加上@Override注释。

说明：getObject()和get0bject()长得比较像，无法区分字母O与数字0，加上@Override可以准确判断是否覆盖成功

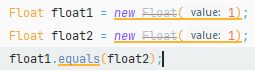
1. 所有的类都必须添加创建者和创建日期，如：



## 2.4格式

1. 浮点数之间的等值判断不能用==，包装数据类型不能用equals。

说明：浮点数采用“尾数+阶码”的编码方式，类似于科学计数法的“有效数字+指数”。



1. 日期格式转化时表示年份用y而不是Y。
2. 日期格式中分清楚H和h还有M和m，分别都有特定的含义，H表示24小时制，h表示12小时制，M表示月份，m表示分钟。



1. 获取当前毫秒用System.currentTimeMillis();（想要更精确可以使用System.nanoTime();）。



## 2.5锁与并发

1. 调用delete时，并发的时候需要加锁。
2. 线程资源必须由线程池提供。

说明：线程池的好处是减少在创建和销毁线程上锁小号的时间以及系统资源的开销，解决资源不足的问题。

1. 线程名称需要有含义（参照变量命名）。
2. 高并发时，能不用锁就不用锁，能锁一块功能就不要锁整个方法，能用对象锁就不要用类锁。
3. 获取锁必须在try外，且获取锁和try之间不会抛出异常，以免无法通过finally解锁。



1. 多线程并行处理定时任务时，Timer运行多个TimeTask时，只要其中之一没有捕获抛出的异常，其他任务就会自动终止，使用ScheduledExecutorService则不会出现这种问题。
2. 在控制线程的时候可以用volatile关键字，使用了volatile关键字后可以保证在传值的时候不会因其他原因中断而导致最终操作发生错误。

## 2.6try与异常

1. try-catch中的finally不能使用return。



1. 捕获异常与抛出异常必须完全匹配。

## 2.8数据库

### **2.8.1 建表规约**

1.表达是与否概念的字段，必须使用 is\_xxx 的方式命名，数据类型是 unsigned tinyint（1 表示是，0 表示否）。

说明：任何字段如果为非负数，必须是 unsigned。

注意：POJO 类中的任何布尔类型的变量，都不要加 is 前缀，所以，需要在<resultMap>设置

从 is\_xxx 到 Xxx 的映射关系。数据库表示是与否的值，使用 tinyint 类型，坚持 is\_xxx 的命名方式是为了明确其取值含义与取值范围。

正例：表达逻辑删除的字段名 is\_deleted，1 表示删除，0 表示未删除。

2.表名、字段名必须使用小写字母或数字，禁止出现数字开头，禁止两个下划线中间只出现数字。数据库字段名的修改代价很大，因为无法进行预发布，所以字段名称需要慎重考虑。

说明：MySQL 在 Windows 下不区分大小写，但在 Linux 下默认是区分大小写。因此，数据库名、

表名、字段名，都不允许出现任何大写字母，避免节外生枝。

正例：aliyun\_admin，rdc\_config，level3\_name

反例：AliyunAdmin，rdcConfig，level\_3\_name

3.表名不使用复数名词。

说明：表名应该仅仅表示表里面的实体内容，不应该表示实体数量，对应于 DO 类名也是单数形式，符合表达习惯。

4.禁用保留字，如 desc、range、match、delayed 等，请参考 MySQL 官方保留字。

5.主键索引名为 pk\_字段名；唯一索引名为 uk\_字段名；普通索引名则为 idx\_字段名。

说明：pk\_ 即 primary key；uk\_ 即 unique key；idx\_ 即 index 的简称。

6.小数类型为 decimal，禁止使用 float 和 double。

说明：float 和 double 在存储的时候，存在精度损失的问题，很可能在值的比较时，得到不正确的结果。如果存储的数据范围超过 decimal 的范围，建议将数据拆成整数和小数分开存储。

7.如果存储的字符串长度几乎相等，使用 char 定长字符串类型。

8. varchar 是可变长字符串，不预先分配存储空间，长度不要超过 5000，如果存储长度大于此值，定义字段类型为 text，独立出来一张表，用主键来对应，避免影响其它字段索引效率。

9.表必备三字段：id, gmt\_create, gmt\_modified。

说明：其中 id 必为主键，类型为 bigint unsigned、单表时自增、步长为 1。gmt\_create,gmt\_modified 的类型均为 datetime 类型，前者现在时表示主动创建，后者过去分词表示被

动更新。

10.表的命名最好是加上“业务名称\_表的作用”。

正例：alipay\_task / force\_project / trade\_config

11.库名与应用名称尽量一致。

12.如果修改字段含义或对字段表示的状态追加时，需要及时更新字段注释。

13.字段允许适当冗余，以提高查询性能，但必须考虑数据一致。冗余字段应遵循：

1）不是频繁修改的字段。

2）不是 varchar 超长字段，更不能是 text 字段。

正例：商品类目名称使用频率高，字段长度短，名称基本一成不变，可在相关联的表中冗余存储类目名称，避免关联查询。

14.单表行数超过 500 万行或者单表容量超过 2GB，才推荐进行分库分表。

说明：如果预计三年后的数据量根本达不到这个级别，请不要在创建表时就分库分表。

15.合适的字符存储长度，不但节约数据库表空间、节约索引存储，更重要的是提升检索速度。

### **2.8.2 SQL语句**

1.不要使用 count(列名)或 count(常量)来替代 count(\*)，count(\*)是 SQL92 定义的标准统计行数的语法，跟数据库无关，跟 NULL 和非 NULL 无关。

说明：count(\*)会统计值为 NULL 的行，而 count(列名)不会统计此列为 NULL 值的行。

2.禁止使用存储过程，存储过程难以调试和扩展，更没有移植性。

3.数据订正（特别是删除、修改记录操作）时，要先 select，避免出现误删除，确认无误才能执行更新语句。

4.如果有国际化需要，所有的字符存储与表示，均以 utf-8 编码，注意字符统计函数的区别。

说明：

SELECT LENGTH("轻松工作")； 返回为 12

SELECT CHARACTER\_LENGTH("轻松工作")； 返回为 4

如果需要存储表情，那么选择 utf8mb4 来进行存储，注意它与 utf-8 编码的区别。

### **2.8.3 ORM映射**

1.在表查询中，一律不要使用 \* 作为查询的字段列表，需要哪些字段必须明确写明。

说明：

1）增加查询分析器解析成本。

2）增减字段容易与 resultMap 配置不一致。

3）无用字段增加网络消耗，尤其是 text 类型的字段。

2.sql.xml 配置参数使用：#{}，#param# 不要使用${} 此种方式容易出现 SQL 注入。

3.不允许直接拿 HashMap 与 Hashtable 作为查询结果集的输出。

说明：resultClass=”Hashtable”，会置入字段名和属性值，但是值的类型不可控。

4.更新数据表记录时，必须同时更新记录对应的 gmt\_modified 字段值为当前时间。

5.不要写一个大而全的数据更新接口。传入为 POJO 类，不管是不是自己的目标更新字段，都进行 update table set c1=value1,c2=value2,c3=value3; 这是不对的。执行 SQL时，不要更新无改动的字段，一是易出错；二是效率低；三是增加 binlog 存储。

6.@Transactional 事务不要滥用。事务会影响数据库的 QPS，另外使用事务的地方需要考虑各方面的回滚方案，包括缓存回滚、搜索引擎回滚、消息补偿、统计修正等。

## 2.8测试

测试代码必须写在src/test/java目录下，不允许在原业务代码上进行测试（源码编译会跳过此目录）。

# 3.复用

## style复用：前端开发中如果遇到了某几个组件调用了相同的style则应将style写在<style>中，以class=””的形式获取



# 4.注解

本文档参考《阿里巴巴泰山版Java开发手册》