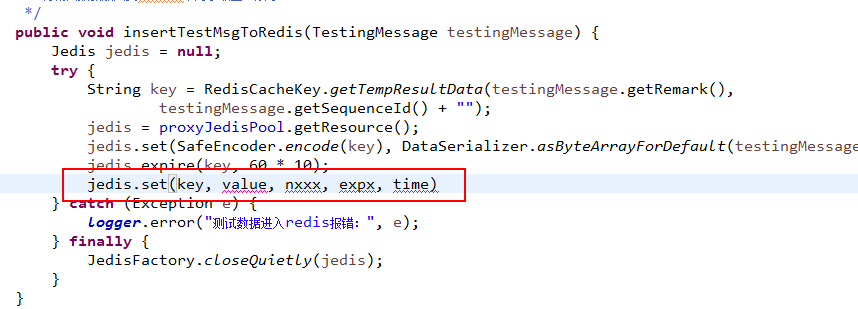
**请把项目里面不需要的代码注释掉，太多无用的类、方法，没法看！**

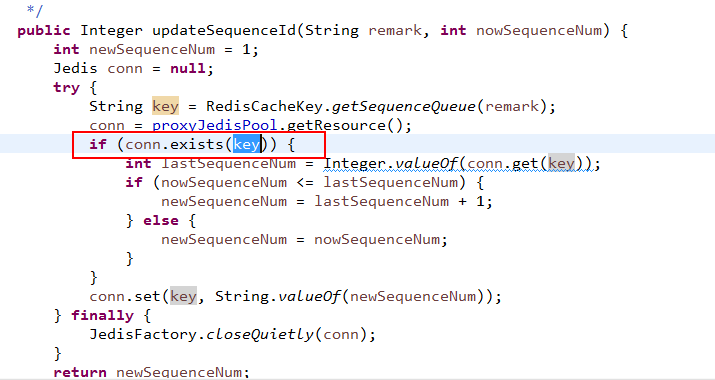
**简化逻辑，ExecuteContext简化掉，我们再来看时间到哪里去了！**

# 一



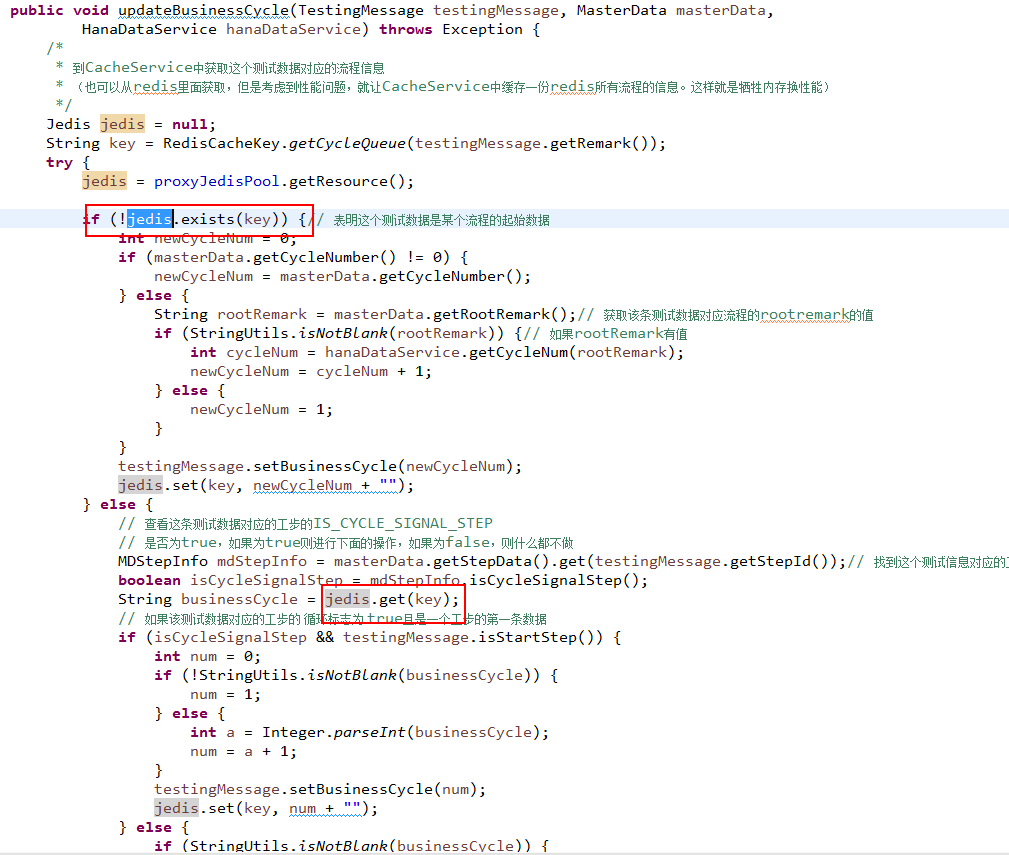
两步操作合并成一步。

# 二



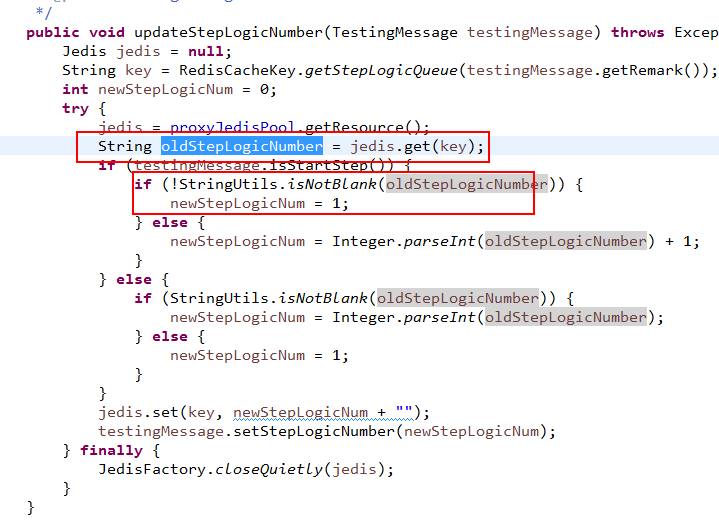
去掉exists查询，判断get的返回值是否为null。

# 三



去掉exists，用判断get返回值。

# 四



没有判断get返回值是否为null，全局禁止使用isNotBlank！用isNotEmp！

# 五



不要用exists，修改方式同上。

不用+””方式转成String，用testingMessage.getTestTimeDuration().toString()

# 六



这些都很耗时，加这些意义不大。全部去掉。

# 七



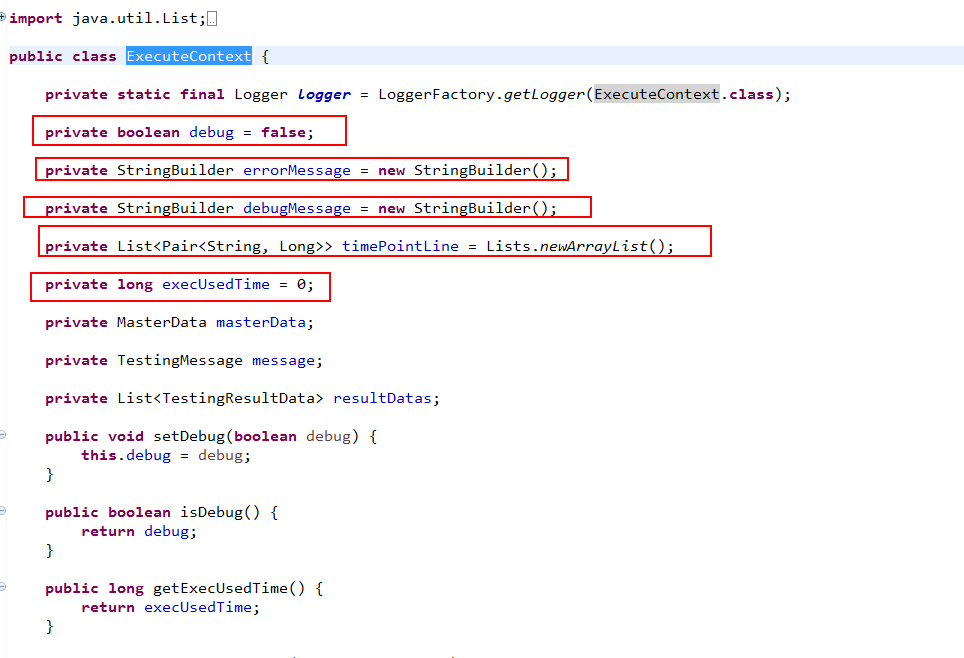
过早的获取到Redis资源！

***编程思路不对！***

***错误：方法开始前把需要的所有资源都打开，然后撸代码，最后关闭掉。***

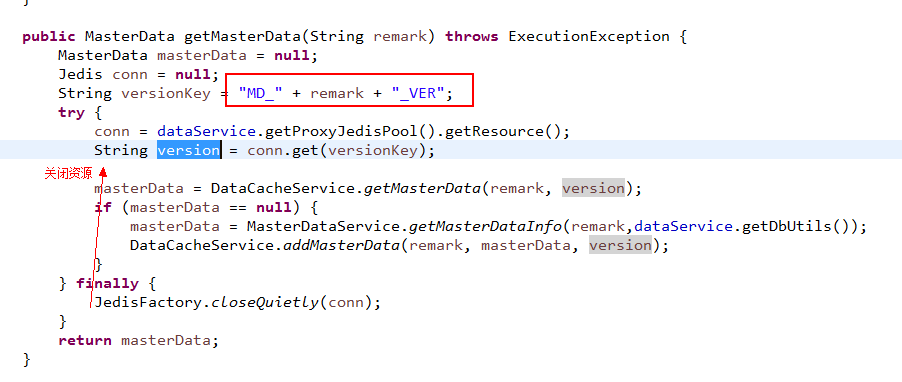
***正确：在用的时候获取资源，用完马上关闭！***

# 八



红框的内容，没用的话，全部删除掉，简化代码逻辑。

# 九



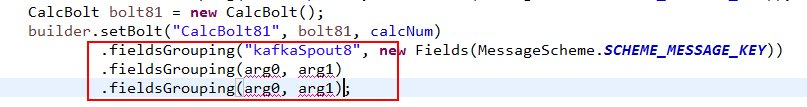
永远不要这样拼接字符串。

及早关闭资源。

# 十

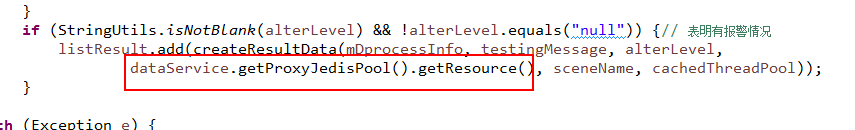


这样分组好处是哪个队列慢，逻辑清晰，但如果有的topic数据量大，有的小的话，建议全局分组。



全部包括。

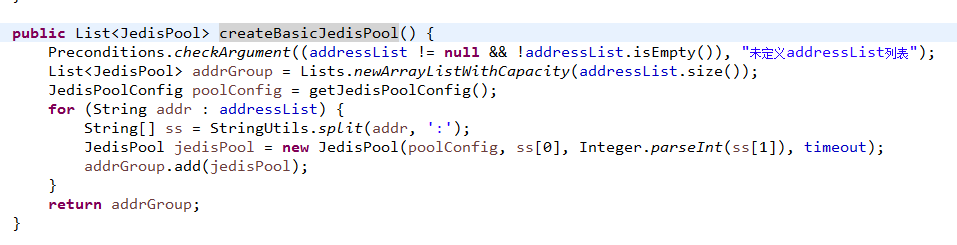
# 十一



**获取了Redis，没有关闭！可以传Pool给这个方法。不要传Jedis资源进去！**

**资源泄露！没有关闭连接！**

# 十二



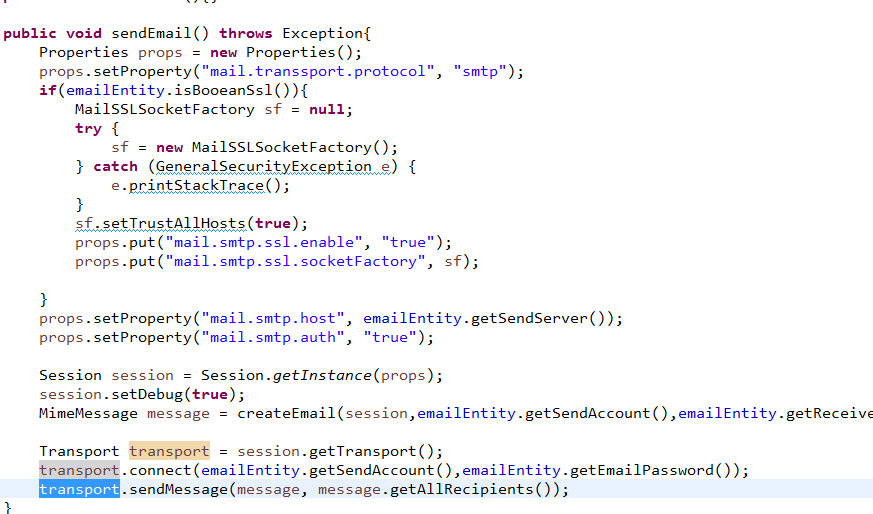
不要的代码都删除掉，如果舍不得删除，注释掉！看着烦！不知道哪些代码需要有用，需要优化！

# 十三

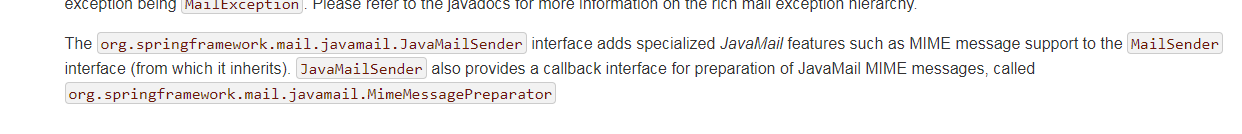


想要提速，手工解析json，解析成一个Java对象。这种代码，没有效率。

# 十四

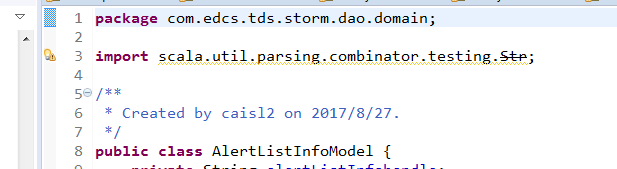


Spring有



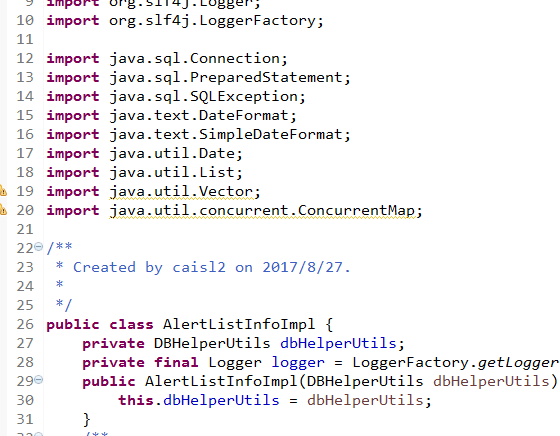
这种代码太啰嗦了！

# 十五



引入了错误的类，格式化下代码。

# 十六



最讨厌不格式化代码。

# 十六



这个类没有存在的价值！就一个DataSouce对象，封装没有任何意义！

org.springframework.jdbc.support

Class JdbcUtils

用这个类。

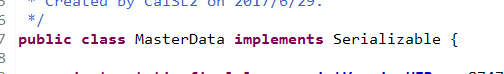
# 十七



老问题，资源使用随意。记住，早关闭连接！

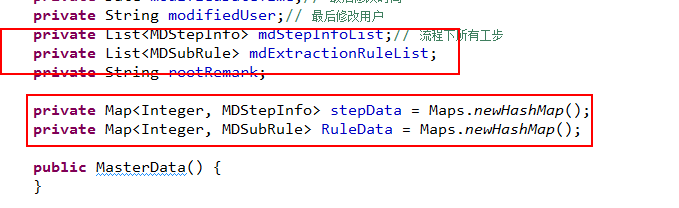
错误方法嵌套，里面那个方法没执行完毕，外面这连接没有释放，一直霸占着。

# 十八



不需要的继承！无意义的。我们用Kryo或者FST做对象序列化。

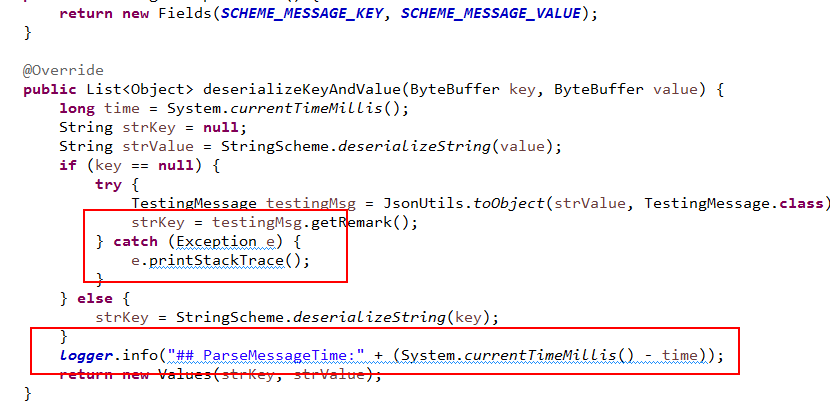
# 十九



为什么List不初始化，而下面的Map却初始化了呢？

初始化对象没有设置size！扩容的代价很高的！

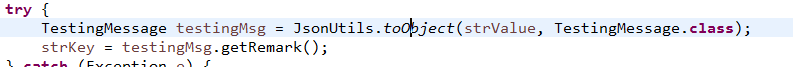
# 十九



没用的代码删除或者注释掉，里面的代码太随意了。

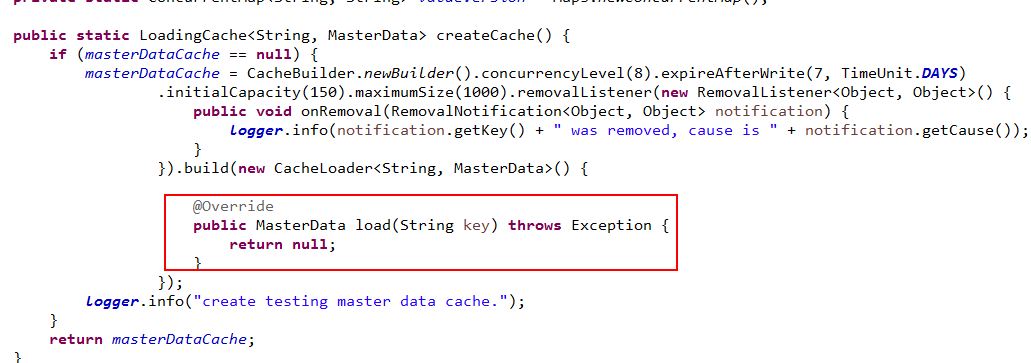
异常打印用日志。

时间计算用 System.nanoTime()，需要每个打印日志。很耗时的！建议这种日志去掉。



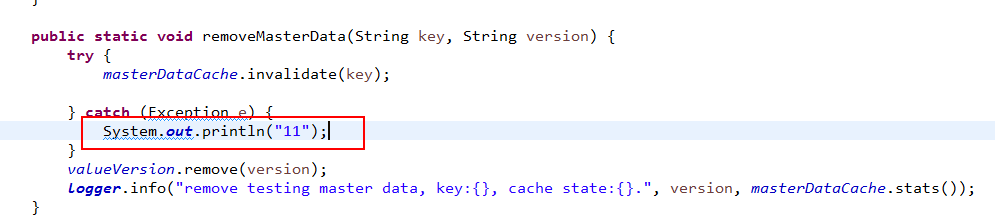
这么耗时的操作，杀了我吧！

# 二十



奇怪的逻辑代码。

# 二十一



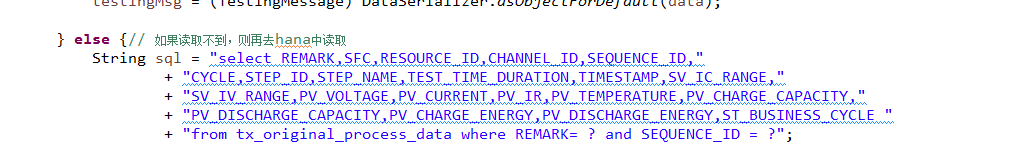
啥意思？

# 二十二



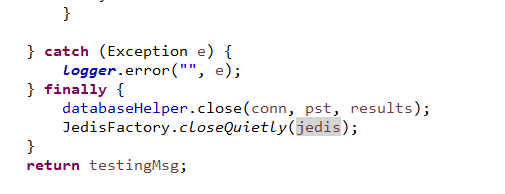
没老问题，应及早关闭资源，多于的exists判断。

# 二十三



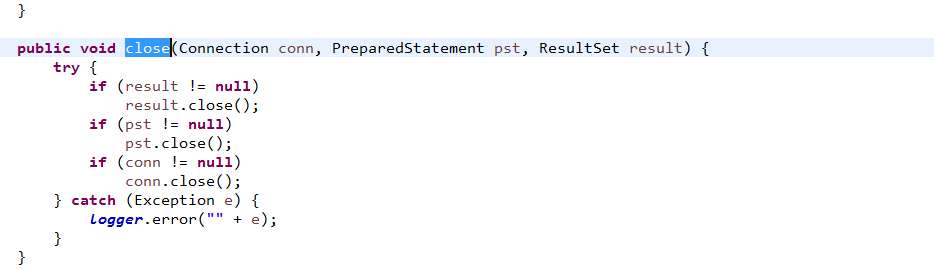
提取出去。不要每次都来一次，没用效率。

# 二十四



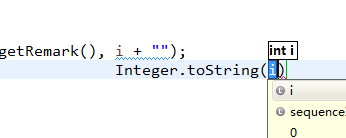
早关闭资源！

# 二十五



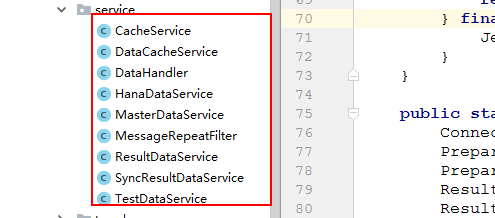
三个try关闭资源

# 二十六



修改为！Integer.toString(i)

# 二十七



这些类都是同一个毛病。

* ***资源打开过早，关闭过晚。***
* ***用简单的字符串拼接。***
* ***获取一个数据库连接方法里面再套数据数据库连接，嵌套很多层。导致一个方法需要占据最多4个连接。完全可以一步步来。占用的资源太多了。***