# 留言系统重构计划

## 重构原因

1. 原来的controller层都是继承basecontroller，在完善测试代码时由于无法隔离掉父类的方法和相关参数导致测试难以进行。
2. 不同的Controller中有自己的方法封装service层，有一些方法可以共用，比如给前台传发送留言状态信息的sendFailureMessage（），sendSuccessMessage（）方法。
3. 成功发送一条留言或回复会给通知系统发送一条通知，原系统是在主进程中调用发送通知的函数



但如果通知系统出现问题，在建立通知、发送通知的过程中阻塞，会导致留言系统无法执行后续完成发送留言后像前台传数据的流程。

1. 原老博客的发送留言或通知的接口已弃用，但仍分布在各个controller中，不便于代码的维护。

**重构目标**

1. 针对上述1和2的原因，用组合代替继承。
2. 编写ThreadManger类封装线程池，单例方式执行线程池创建和管理线程。
3. 分离出老博客接口，建立OldBlogController保管接口，不至于使老接口失效。

重构步骤

1. Controller层重构

建立ControllerUtils类，封装service层，将Basecontroller与其他controller中的部分方法改写进ControllerUtils，采用Spring注入的方式将ControllerUtils注入进到contoller中。由于打印发送留言状态信息的方法与service层无关，在测试过程中也不需隔离，又考虑到代码的重用，编写ResponseUtils，设置静态方法负责打印json成功，失败的状态信息。

1. 通知的处理

发送通知类MessageSendingTask实现Runnable，在实现run()中增加逻辑判断

**if** (!sender.equals(receiver))，如果如果sender与receiver相同，则直接打印日志并退出。

选择*CachedThreadPool*作为线程池*,*不必设置线程最大数量，*CachedThreadPool*会在程序执行过程中创建与所需数量相同的进程，然后再回收旧线程时停止创建新线程。在测试时必发执行10,000个任务时有些卡，（4G内存，i5CPU），但在执行80,000任务并发时，内存没有溢出，但CPU先扛不住了。而如果选择设置最大进程数的线程池，会执行代价高昂的线程分配，超过并发的线程时就会舍弃后来的任务。（所以在性能和风险选择上值得商榷）。

1. 建立OldBlogController分离出老博客接口发送留言，回复，删除留言，回复接口。
2. 使用easymock测试controller层，导入MockHttpServletRequest，MockHttpServletResponse，模拟放入Request中相关所需参数，隔离service层，随后assert检验Response中的JSON数据是否与预期一致。

**重构效果**

每个新建立的类修改过的类都建立相关的测试，测试通过。达到预期要求。

**重构时间点**

8.25日提测

8.