OO第二单元总结

OO的第二单元主要训练了我们多线程编程的能力，主要包括了多电梯调度，IFTTT文件监视器，城市出租车调度系统。一下我将对这三次作业写一些我自己的心得体会。

# 多电梯调度问题

## 程序结构设计图



第五次作业的程序结构比较简单，主要是使用了一个Time类进行整个程序时间的整体把控，其实这个Time类也可以直接用System.CurrentTimemillions直接搞定，只是当时年轻没有考虑到这一点，这个应该算是封装不当吧。

程序运行的主要流程就是，首先由RequestReader来读取输入，并根据合法的输入生成相应的请求。请求先被添加到Dispatcher的请求队列中，然后由Dispather根据当前三个电梯的运动状况来将请求分配给相应的电梯。当然这里的分配也仅是分配给电梯的请求队列而已，之后电梯按照自己的请求队列中的请求来一个个完成任务。为了避免冲突，这里的请求队列使用的是ArrayBlockingQueue。

## 线程协作视角图



本次作业的线程协作关系较为清晰明了，主要难点在于解决这几个RQueue之间的生产者-消费者问题。Dispatcher在是EleQueue生产者的同时还是RQueue的消费者，如何做好两个甚至多个队列之间的同步很关键。我这里采用的是ArrayBlockingQueue来构建阻塞队列，如果自己实现的话确实还是要花一些功夫。除此之外我觉得这次作业没有啥难点。

# IFTTT文件系统

## 程序结构设计图



这次作业个人认为是所有OO作业中最坑的一次。因为本次作业所要设计的功能实在太多，而且又没有之前作业的铺垫。很多关于文件的操作之前都没有学过，这次作业基本全靠现学现卖，而且时间只有一周，所以感觉效果很不好。

言归正传，这次作业的主要结构如上图。主要是在程序刚开始的时候在控制台读入所需要监视的目录或文件信息，Reader类读取后将其转化为Ifttt对象保存起来，之后启动Watch线程。其实这次的作业只有这一个线程。他负责来监视Ifttt所指向的那几个目录，并且记录下来目录中所有文件的FileInfo，包括文件的文件名，路径，大小，最后修改时间等等。每隔一定时间片，Watch就会扫描一下上述那些目录，并将其与前一次保存的FileInfo作比较，当发现满足Ifttt中的条件的时候，就执行Modifier类中相应的修改方法。其中Summary和Detail类负责创建/修改这两个文件。整体的思路大体就是这样，其实最大的难点还是在于文件信息的保存和文件操作这一块儿，而不在于多线程。由于本次作业除了主线程之外只开了一个Watch线程，因此就不单独介绍线程之间的协作关系了。

# 出租车调度系统

## 程序结构设计图



第七次作业应该是所有OO作业中设计的比较合理的，个人认为也是本人完成的比较满意的一次作业。这次作业能够很好地体现多线程思想，而且能够将之前所学到的线程同步的知识串联起来。

首先从Main类开始，首先创建一个地图类用来将文件中的字符转换为Node结点以保存必要的信息。Taxi类是线程类，其功能非常简单，说到底就是一个状态机，通过不停地转换状态就可以达到规定的功能。状态转换图如下：



之后就轮到Customer线程了，线程中包含了几乎所有测试的方法，包括添加请求的方法和用来显示车辆信息的方法等等，是用来输入Order的主要手段。其中Order本身也是一个线程，在被输入之后就会被启动，扫描3s内所有经过了窗口的出租车，并选择能够接受请求的最符合条件的一个传递给他。特别需要注意的是在分配的时候需要锁定出租车的队列，否则可能会出现线程冲突的问题。

## 线程协作示意图



协作关系很简单，主要就是通过Customer来创建新的Order，然后Order来吧自己分配给相应的出租车就可以了。