基于Android开发实现的电梯调度App

2051914 金逸

一．程序基本功能

本程序利用Android开发，使用Java语言开发基本的电梯调度系统，能够执行基本的电梯运行功能，包括切换电梯，切换楼层，针对各种响应事件的电梯正确调度系统，电梯运行的可视化展示，电梯开关门的实现等

## 二．调度程序所需变量

1. 定义电梯数量5和楼层数量20
2. 用List列表存储电梯界面显示的按钮，分别用来记录电梯所有的向上的按钮，所有的向下的按钮，以及每部电梯对应的标签，以及计时器对象
3. 对于调度状态，同样用List列表来存储，分别存储向上的楼层按钮按下的状态，向下的楼层按钮状态，电梯对象以及未能处理的任务
4. 用Map来记录那部电梯响应了哪个上升任务，哪部电梯记录了哪个下降任务
5. 对于每个外部请求任务，用Job类进行存储，里面存储三个参数，分别为发出请求的层数，向上还是向下，以及该请求是否完成
6. 接着利用两个枚举类型，第一个枚举类型用来标识电梯的三种状态，根据题意三种状态应分别为暂停，向上和向下；第二个枚举类型用来表示电梯的当前任务，分别为向上，向下和无任务
7. 对于每个电梯对象，使用类进行管理，首先设置两个变量来存储每台电梯的当前使用状态以及具体运行的状态，初值设为暂停和释放状态；接着用一个字典来存储电梯内按钮的状态，字典的键用来代表按钮代表的楼层，值用来代表按钮的状态；再设置两个变量代表电梯所处的楼层，以及电梯的编号；再用构造函数对其进行初始化，将所有按钮均设为未被按下的false的状态
8. 还需要设置两个变量option\_elevator和selected\_number,用来表示当前选中的电梯和当前选中的楼层
9. 对于电梯开关门的状态，引入类似于锁的概念，设置一个Map变量lock\_door，用于表示当前电梯开关门的状态

## 三．具体调度算法

电梯调度算法，根据实际需求应该分为两类，一类是电梯内部请求的调度，一类是电梯外部请求的调度，下面针对这两类调度进行具体分析

### 外部调度算法

1.1发生条件：当有人在某个楼层按下向上或向下的按钮时，即发出外部请求，进行电梯外部请求调度

1.2算法具体流程

1.2.1：首先选定发出请求的层数，以及请求的方向向上还是向下，再遍历list列表中存储的所有电梯对象，计算每部电梯的当前楼层和发出请求的楼层之间的差值，再计算出差值后，进行下一步分析

1.2.2：首先判断有没有空闲的电梯可以运行，假如存在空闲的电梯且该电梯对应的相应调度池中没有与发出请求楼层方向相反的调度，则选中该电梯

1.2.3：其次若电梯正在运行中，分两种情况讨论，首先时电梯有上升任务，且当前楼层发出的请求为上升，且电梯并未相应下降请求，则先加入待选；其次电梯正在执行下降任务，且当前请求为下降任务，且电梯未响应上升请求，则先加入待选

1.2.4：对1.2.3中选中的电梯进行判断，判断条件为发出请求的楼层数和选中电梯楼层之间的关系，假如电梯正在上升，则需满足当前电梯层数小于发出请求电梯的层数；假如电梯正在下降，则需要满足当前电梯层数大于发出请求电梯的层数，假如不满足则进行下一步

1.2.5：假如电梯正好处于暂停状态，且发出请求楼层和电梯当前处于同一楼层，则直接选中该电梯

1.2.6：假如找到合适的电梯，则将该任务写入对应电梯的目标楼层中，并将其加入等待队列，再检查等待队列中是否有该任务，若有的话，将其状态改为完成状态；若未找到合适的电梯，则将该任务加入到等待队列中

1.2.7：在遍历所有电梯时，每找到一部合适电梯，将距离distance改为该电梯楼层数到发出请求电梯楼层数的绝对数差值，后面合适的电梯若小于这个差值，说明该电梯距离发出请求楼层距离更近，从而保证了距离最短的调度算法

### 内部调度算法

内部调度算法比较简单，每次电梯内部按钮被按下后，将电梯内部楼层对应状态设为true即可，若当前电梯状态为free即起步状态，则相应改变电梯的上行下行状态，楼层状态则展示为open状态

### 具体调度过程

3.1：利用时间控件，每隔1s检查电梯状态，首先更新楼层信息，假如当前电梯的状态为向上，则电梯对应楼层加一，假设电梯状态为向下，则电梯对应楼层减一

3.2：其次进行状态转移过程，假设此刻电梯状态为暂停，表示当前处于停靠状态，检查是否需要继续，遍历对应电梯的所有楼层，查看是否存在下降或上升请求，再查看是否满足了电梯的外部请求，假设满足外部请求，即电梯停靠再之前发出请求的楼层，则将其移除向下请求的处理队列中，并更新标签状况

3.3最后进行电梯运行情况的状态转移，假电梯只有上升请求，则更改电梯的状态为上升；假如电梯只有下降请求，则更改电梯的状态为下降，假如电梯当前请求为上升且有上升楼层，则电梯继续上升，假如电梯当前请求状态为下降且有下降楼层，则继续下降，若无上升和下降楼层则电梯为释放状态

3.4假设电梯当前并非停靠状态，则需要检查电梯所处当前楼层是否需要停靠，假设电梯的目标楼层中含有当前楼层，则需要将当前状态改为pause进行停靠

3.5对于等待队列的调度，同样采用时间控件，每0.5s进行一次调度，尝试为调度池内的目标楼层分配电梯，假设分配成功则将其从调度池内移除

### 具体开关门实现

对于开关门的实现，首先设置两个button，分别代表开门和关门事件，并分别绑定相应的点击相应函数

4.1开门

在点击关门按钮时，首先检查电梯是否处于pause状态，当电梯处于pause状态时，才能够进行下一步的开门操作，当前电梯假如处于pause状态，首先暂停两秒，并利用lock\_door变量对当前选中电梯上锁，即不能执行其他操作(实质为堵塞相应电梯对应的timer控件正常执行)，在两秒后再将lock\_door变量对应当前电梯key值设为true，表示电梯可以正常工作

4.2关门

在点击开门按钮时，首先检查电梯是否处于pause状态，当电梯处于pause状态时，才能继续下一步的关门操作，关门操作将lock\_door变量对应的key值设为false即可，使电梯解除开门状态，可以继续正常调度

## 五．程序具体执行过程

5.1 程序的执行过程，依靠Java中的Timer时间控件，分别设置5个时间控件，分别对五部电梯进行定时控制，时间间隔设置为1s，每一秒进行一次调度，检查电梯的状态并进行电梯状态转移

5.2对于进入等待队列的任务，同样用一个Timer时间控件进行管理，每0.5s检查一次等待队列，对其中的任务尝试进行外部调度，假如任务完成则将其移除等待队列

5.3对于电梯内部按钮，先将电梯内部button存入List列表中，随后对每个内部按钮利用Android自带的事件响应机制，为其绑定相应的响应事件buttoninner\_click，代表进入内部调度

5.4对于每个楼层的上下键button，先将其存入List列表中，随后同样为其绑定响应事件的触发函数 buttonout\_click，代表进行外部调度

5.5对于电梯的选择，因为页面每次只能选择一个电梯内部，因此五部电梯用radioButton来表示，即单选按钮，每次选中电梯时，将选中电梯的编号赋值给option\_elevator，方便其他函数调用

5.6对于楼层的选择，使用下拉列表spinner来实现，在每次切换楼层后，将当前楼层编号赋给selected\_number，表示当前选中楼层

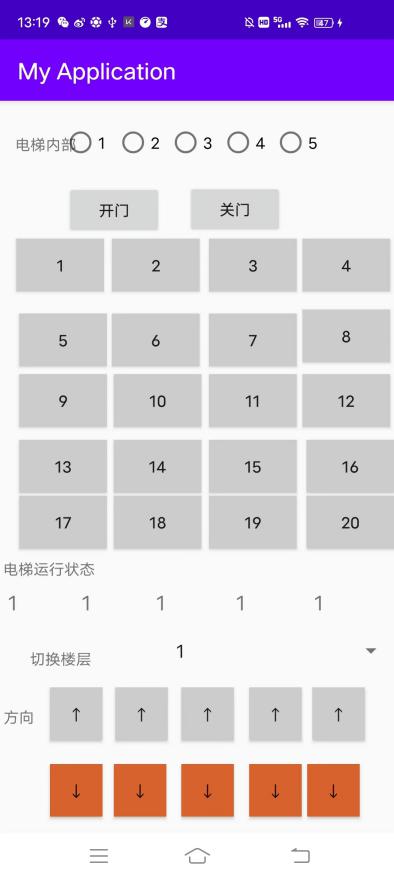
### 六．调度过程中电梯状态变量的变化情况

6.1：电梯运动状态status：status代表每部电梯当前的运动状态，分为up，down，pause三种情况，每当电梯到达需要依靠的楼层时，都需要将status变量改为pause状态，使电梯到指定楼层后停靠，每次停靠后，再执行2.2和2.3中的操作，保证电梯的执行

6.2：电梯当前请求状态order：order代表每部电梯的当前任务，分别为上升，下降和释放，其改变发生在三种情况，一是当电梯内部按钮按下时，若电梯当前任务为free状态，则根据内部按钮和当前楼层改变order状态；二是每次发生停靠时，电梯当前任务完成，因此执行2.2和2.3中的操作，重新给电梯赋予相应任务，保证电梯持续运行

### app运行截图和情况展示

7.1初始界面展示



初始界面包含四大部分

1. 电梯选择按钮，通过选择1-5的按钮，切换到相应电梯内部
2. 电梯内部按钮，包含1-20的楼层键，用于选择电梯要到达的楼层，红色代表当前电梯要去的楼层
3. 电梯运行状态显示区：实时显示每部电梯的运行状态和所处楼层
4. 楼层控制区：下拉列表用于选择楼层，方向键用于每层的上下行控制，红色代表已被按下(或代表1楼不能往下，20楼不能往上)

7.1运行状态展示

这是调度过程中的运行状态截图，此图的含义为第五部电梯内有7，9，11，13层要去，五部电梯运行状态四部上行，一部处于开门状态，第18层按下了上下层按钮



7.2 开关门显示



当前1电梯处于9楼，处于暂停状态，点击开门按钮后，处于open状态，该电梯暂时不能上下行，在点击关门按钮后，电梯关门，电梯可以继续正常工作。