同濟大學
TONGJI UNIVERSITY

项目说明文档

电梯调度

指导教师：王冬青

1751984　王舸飞

1.综述

1.1 背景分析

某一栋楼有20层, 有5部互连的电梯, 要求在编程过程中体现出多线程的思想

在每部电梯内应有对应的功能键, 包括任意楼层的呼叫信号, 主动开关门信号和呼救、通话信号

在每一层外部也有对应的上下键按钮用于呼叫电梯

此外 ,还要求有楼层的显示器和上行下行的显示器

1.2 功能分析

根据上述背景, 得知本次项目的主要难点如下:

1、并发呼叫: 在传统的电梯中, 每部电梯是相互独立的, 也就是说用户需要自己观察多部电梯的运行状况和当前楼层来决定呼叫哪一部电梯 ~~(当然, 他也可以选择同时呼叫所有的电梯这种缺乏素质的行为)~~ 为了避免这种行为的发生并优化用户体验, 我们将选择最优调度的任务交由内部算法处理, 由算法决定哪一部电梯为用户服务

2、运行逻辑: 需要考虑到电梯在运行过程中如何判断何时运行、何时停下、何时改变运行方向的问题

3、主动开关门逻辑: 电梯内部的开关门会直接影响电梯的服务时间, 需要考虑到用户可能在某一层反复点按开关门的情况,并保证在任何情况下,电梯的内部时钟不崩溃

4、界面辅助逻辑: 需要考虑到显示屏的更新, 和电梯上下行状态按钮的更新

5、界面布局与运行动画设计: 需要拥有简单易用的图形化界面, 让用户直观的体验到程序运行的同时, 也可以看到电梯内部的部分算法处理

2.设计

2.1 解决方案设计初步

1、并发呼叫: 调度的过程中采用效率最高的 LOOK 算法, 它的核心逻辑是, 在用户发出请求后计算一次在当前状态下(不包括未来的点按状态), 哪一部电梯到呼叫层的理论时间最短, 找到相应编号后, 将它加入编号电梯的处理队列中

2、运行逻辑: 运用多线程的思想, 让五部电梯每隔一秒同时执行一个函数, 该函数将判断电梯在当前层需不需要停: 如果需要, 播放开关门动画, 否则根据运行方向更新楼层信息

3、主动开关门逻辑: 严格规定运行过程中不许开关门, 而现实中也是这样的: 如果你想尽早的离开电梯, 比起点按开门按钮, 你更应该点击运行方向上离你最近的楼层按钮; 另外, 不允许主动开关门按钮私自更新内部电梯时钟而只是播放动画, 将更新的权力严格限制在判定函数内, 保证每部电梯的内部时钟稳定性

4、界面辅助逻辑: 设定相应全局变量或数组, 保存相应电梯的运行状态和当前楼层, 在每次并发执行判定函数时进行更新

对于通话和求救信号,在这里认为他们不对界面的运行逻辑造成影响而只是给出提示

5、界面布局与运行动画设计: 整个界面拟采用html+css+javascript的经典web布局实现, 其中采用了jquery框架用于简化代码和制作动画, 并调用了font-awesome字体库来制作按钮图标

在电梯布局的下方, 借用了mScrollBar库生成了一个可以即时更新的状态提示框

3.实现

3.1 相关变量的实现

为了实现上述功能, 首先需要一些常变量, 增加代码可读性

变量名	值	功能
TOTAL_ELE	5	指示电梯的数量,方便了创建相关数组和更改电梯数量
INSIDE	0	这三个变量仅在一个地方发挥作用在关灯函数执行时,它们很好的只指示了应该关掉哪里的灯
OUTSIDE_UP	1	
OUTSIDE_DOWN	2	

变量名	值	功能
MAX_FLOOR	20	它们方便了楼层的更新,并在计算距离时增加了代码可读性
MIN_FLOOR	0	

在设置变量时, 由于有五部电梯, 故对所有的变量基本都用了数组的形式定义

只要把数组长度设为5, 就可以方便的对称定义五部电梯

下面列出了相关变量、功能和default值

变量名	功能	电梯0	电梯1	电梯2	电梯3	电梯4
Requests[]	存储请求	空队列	空队列	空队列	空队列	空队列
Inside[]	指示第n部电梯的第m层是否有内部请求	空队列	空队列	空队列	空队列	空队列
OutsideUp[]	指示第n部电梯的第m层是否有外部上行请求	空队列	空队列	空队列	空队列	空队列
OutsideDown[]	指示第n部电梯的第m层是否有内部上行请求	空队列	空队列	空队列	空队列	空队列
Timer[]	javascript特性,存储电梯时钟	run(),1000	run(),1000	run(),1000	run(),1000	run(),1000
NeedToStop	判断电梯是否要停	TRUE	TRUE	TRUE	TRUE	TRUE
IsGoingUp	判断电梯的运动趋势	TRUE	TRUE	TRUE	TRUE	TRUE
IsRunning	只要有请求就是在运行	FALSE	FALSE	FALSE	FALSE	FALSE
currentfloor	指示当前层	1	1	1	1	1

在运行到某一时刻时可能出现的变量情况如下:

变量名	功能	电梯0	电梯1	电梯2	电梯3	电梯4
Requests[]	存储请求	[4,7,11,3]	[]	[2]	[5,18,3]	[20]
Inside[]	指示第n部电梯的第m层是否有内部请求	Inside[4]=true	[]	Inside[2]=true;	Inside[5]=Inside[18]=true	[]

变量名	功能	电梯0	电梯1	电梯2	电梯3	电梯4
OutsideUp[]	指示第n部电梯的第m层是否有外部上行请求	OutsideUp[7]=OutsideUp[11]=true	[]	[]	[]	OutsideUp[20]=true
OutsideDown[]	指示第n部电梯的第m层是否有内部上行请求	OutsideDown[3]=true;	[]	[]	OutsideDown[3]=true;	[]
Timer[]	javascript特性,存储电梯时钟	run(),1000	NULL(可能在开关门)	NULL(可能在开关门)	run(),1000	run(),1000
NeedToStop	判断电梯是否要停	FALSE	TRUE	TRUE	FALSE	FALSE
IsGoingUp	判断电梯的运动趋势	TRUE	FALSE	TRUE	FALSE	FALSE
IsRunning	只要有请求就是在运行	TRUE	FALSE	TRUE	TRUE	TRUE
currentfloor	指示当前层	2	12	1	19	19

3.2 主要函数及逻辑的实现

函数名	功能	调用时间	备注
init()	初始化所有变量为default值	页面加载完毕后	同时初始化滚动提示条
dial(ele_number,floor)	呼叫电梯并更新状态	外部算法计算得出编号后,或内部点按事件发生后	
$(“outside_up”).click() $(“outside_down”).click()	外部点击后,计算得出最优电梯并加入该电梯的请求队列中	外部点击事件发生后	默认一层1秒,开关门共计5秒
$(“.dial”).click()	内部点击后,加入本电梯的请求队列	内部点击楼层按钮后	无论如何都加入
run(ele_number)	最关键的函数,五部电梯反复调用,判断当前层需不需要停并给出相应的动作	除开门关门外,每秒一次	
updateFloor(ele_number)	播放移动动画	run()后	

函数名	功能	调用时间	备注
moveDown(ele_number) moveUp(ele_number)	更新currentFloor[]和屏幕指示灯	run()判断要移动后	与上面的函数产生冗余甚至冲突,可以合并
updateStatus(ele_number)	更新Isrunning[] & IsGoingUp[]和上下行指示灯	run()判断在运行后	

3.3 辅助函数的实现

函数名	功能	调用时间
openDoor(ele_number) closeDoor(ele_number)	播放开关门的小动画	停下或主动开关门时
$(“.cal”l).click() $ (“.emergency”).click()	给出相关提示	点按通话、应急按钮时
$(.close).click() $(.open).click	判断是否为合法开关,如果是,主动开、关门,否则给出提示信息	点按主动开关按钮时
shutLights(ele_number,floor,method)	关掉相应的灯	run()判断该停时
getMaxInQueue(n)	获取队列最大值	更新状态和计算最优解时
getMinInQueue(n)	获取队列最小值	更新状态和计算最优解时
removeFromQueue(n,floor)	在n号队列删除所有为floor的请求	在该层停止时
countGaps(higher,lower,n)	获取higher和lower间要停几次	计算最优解时

3.4 总体流程说明

进入界面后, 立即初始化所有变量并为下方的指示条设定移动

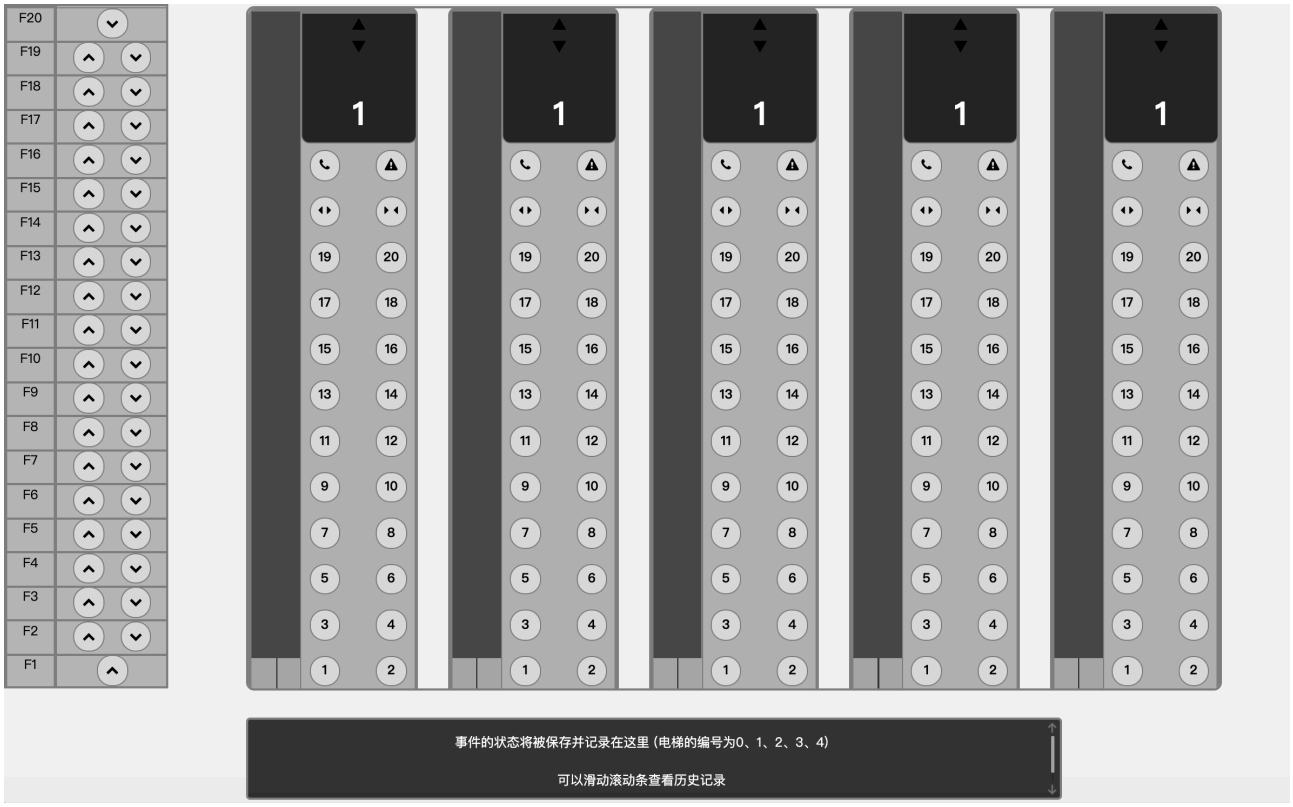
这里的初始化包括了反复执行run()函数, 对电梯进行实时监控

javascript在处理事件(event)驱动的问题时有巨大的优势, 之后对应不同的事件反复执行不同的函数, 并在每次更新状态后将执行的操作打印到下方提示条中,即完成了整个项目的实现

上述所有的代码及look算法的具体实现可以在带有详细注释的 elevator.js 文件中找到, 在此不再复制粘贴

4.测试(基于Chrome)

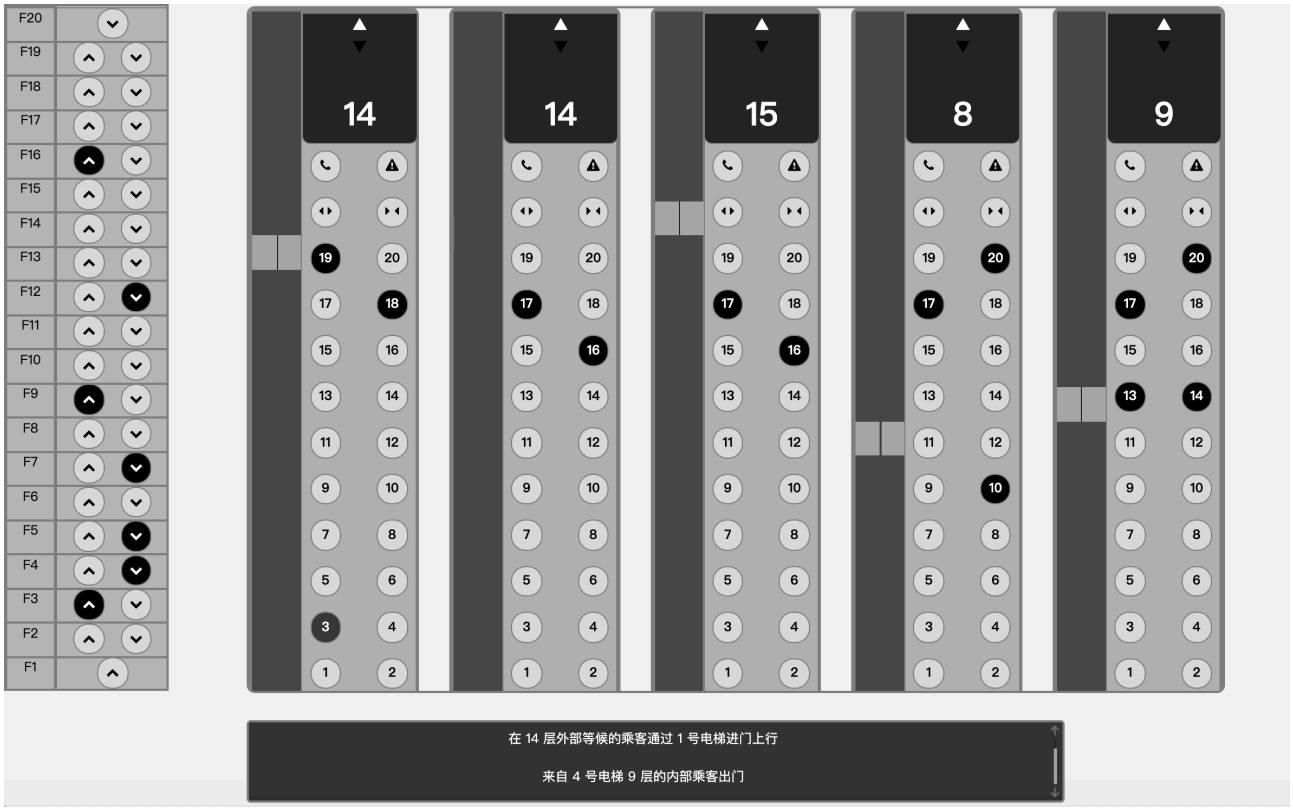
4.1 初始状态测试



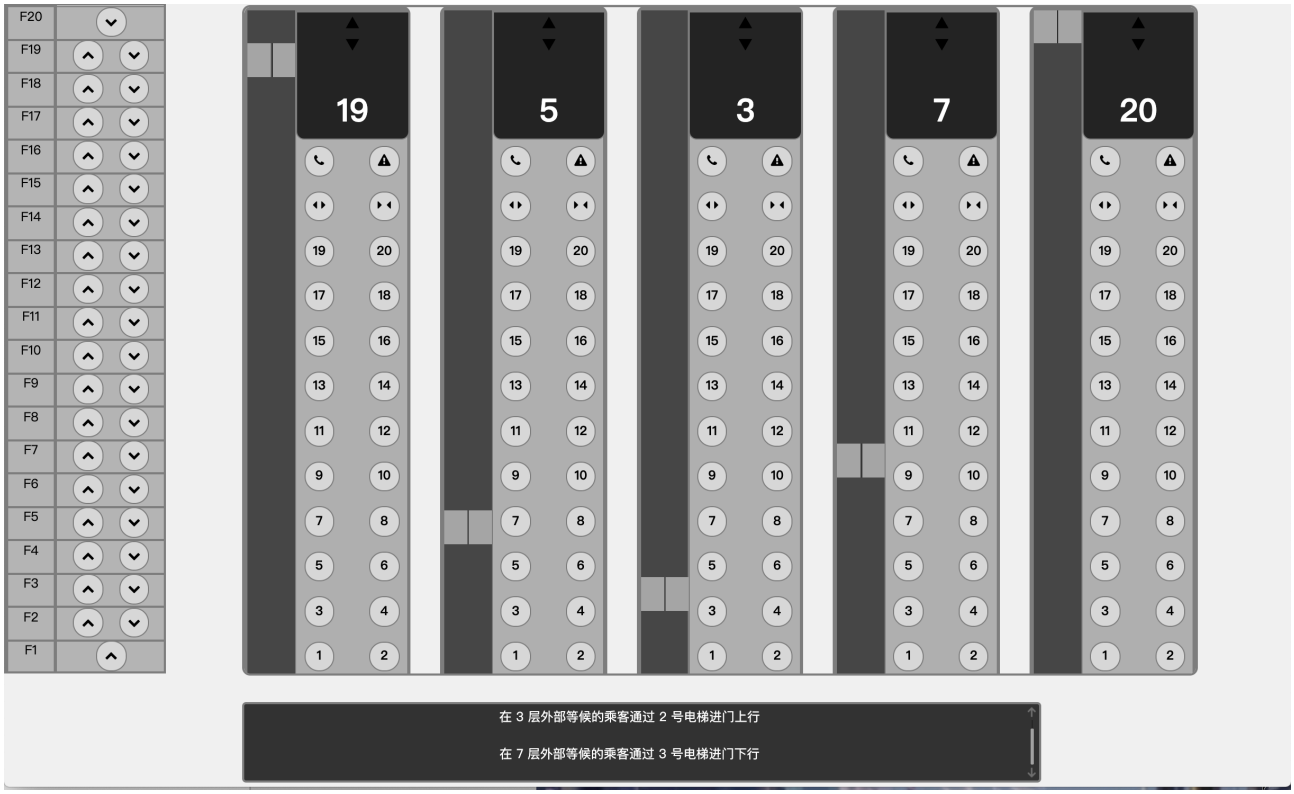
(程序的初始界面)

4.2 运行状态测试

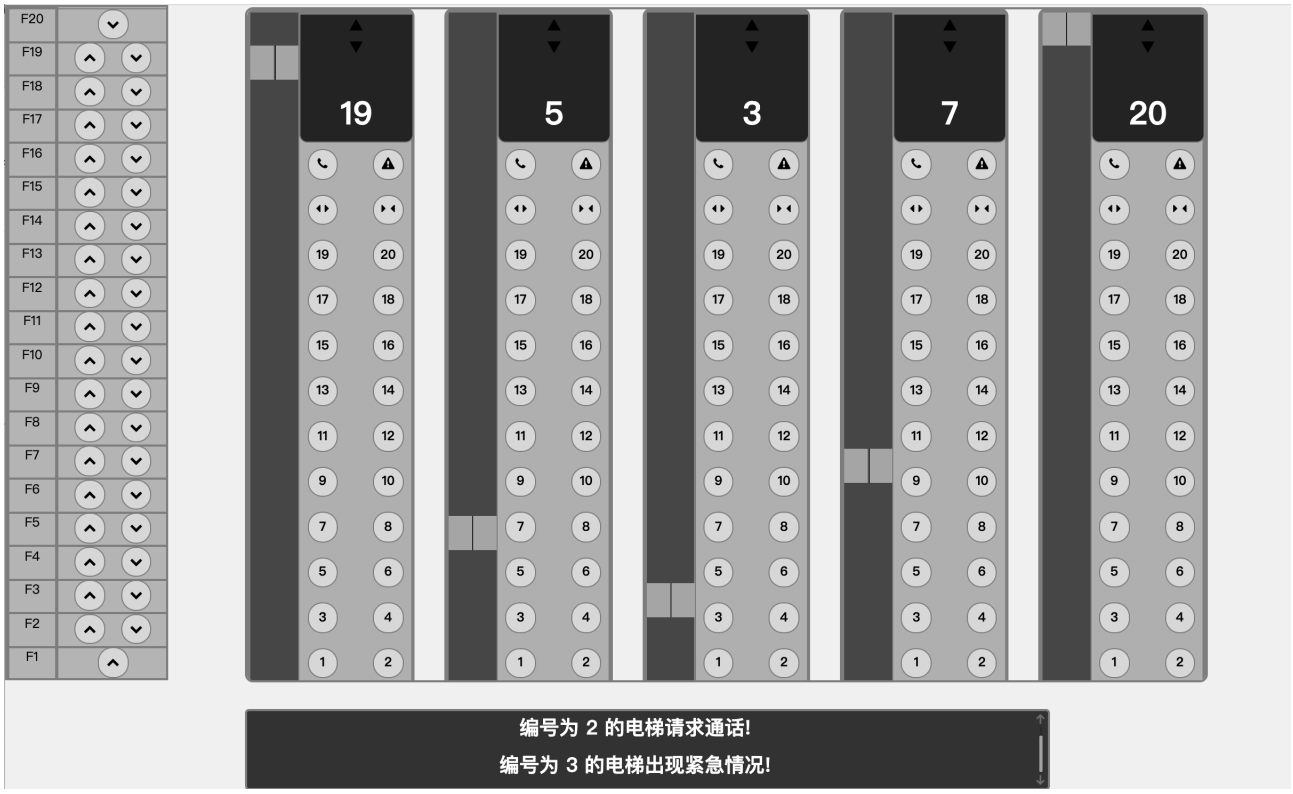
4.2.1 大量请求测试



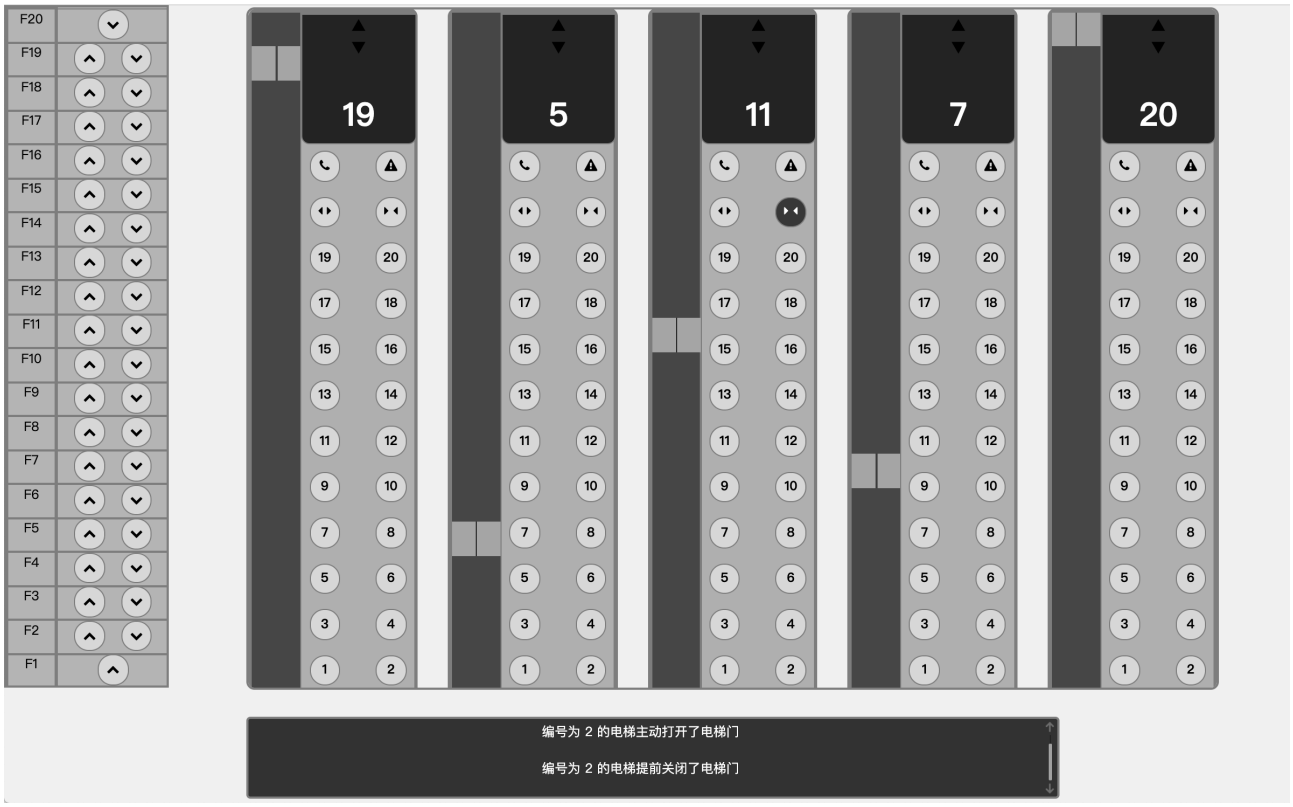
4.2.2 无请求停顿测试


F20
F19
F18
F17
F16
F15
F14
F13
F12
F11
F10
F9
F8
F7
F6
F5
F4
F3
F2
F1
19
5
3
7
20
在 3 层外部等候的乘客通过 2 号电梯进门上行
在 7 层外部等候的乘客通过 3 号电梯进门下行

4.2.3 警告&通话测试

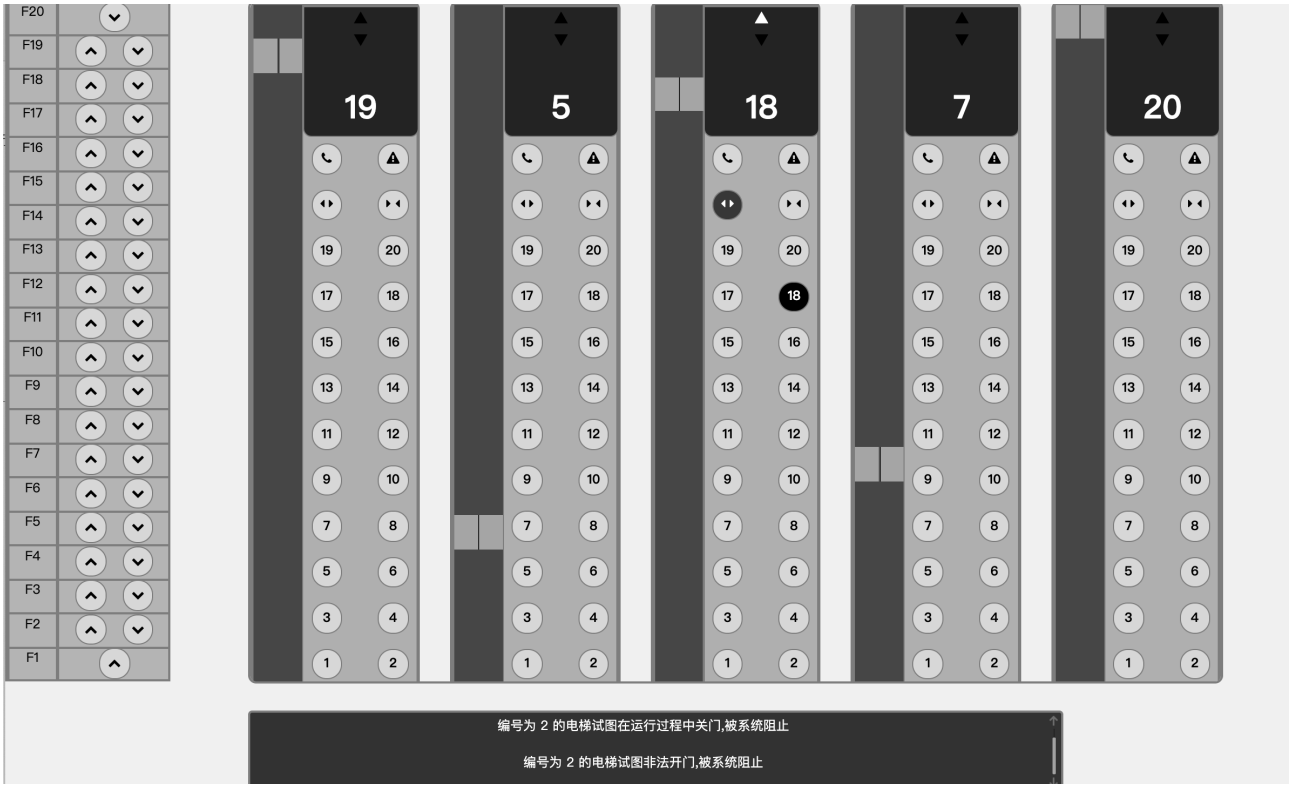

F20
F19
F18
F17
F16
F15
F14
F13
F12
F11
F10
F9
F8
F7
F6
F5
F4
F3
F2
F1
19
5
3
7
20
编号为 2 的电梯请求通话!
编号为 3 的电梯出现紧急情况!

4.2.4 反复开关电梯门测试



无论怎么反复开关门,电梯的timer仍然保持正确

4.2.4 非法开门测试



5.总结

5.1 开发环境总结

采用了Atom 与 Chrome 的开发者工具

5.2 开发过程总结

在设计之初,首先利用良好的前端布局和命名习惯对html和css文件进行了部署

之后, 分析了主要的设计难点, 并初步定义了大量变量和用于处理问题的函数

在具体实现的过程中, 对现存的变量和函数的必要性进行了反思, 并将重复的变量进行了删除

初步的原型完成后, 进行了大量的边界测试, 进一步完善了代码的逻辑

全部逻辑测试通过后, 开始进一步思考界面的布局, 调整部分美工后, 运用mScrollBar对每一次更新状态进行了指示

5.3 相关参考和问题提示

本次设计参考了 https://github.com/ChenCyl/ElevatorScheduling

但上述参考存在许多问题,(在实现的过程中将他们全部修复了), 列出几点主要的如下:

1、调度算法中, 上行下行判断后的计算间隔变量明显带入错了, 这将直接导致不能找到最好的一部电梯进行调度, 该问题目前已经修复

2、由于将修改时钟的权力赋给了主动开门函数, 反复开门后, 电梯的时钟彻底紊乱, 甚至可能出现从1层瞬移到20层的问题

将修改时钟的权力限制在run()之后, 该问题得到了解决

3、初始化时未将一层初始化为需要停止的楼层, 这导致在一开始就开门时, 会提示电梯在运行
该问题已修复

4、变量命名不统一, javascrip代码风格存在危险(jslint检查器给出了12个警告),修改后,代码更加直观简洁, 且清除了所有警告

5、html和css的类和id命名不规范, 导致在反复点按一个按钮时, 由于添加了表示开关亮灯的 on 类, 使得程序无法准确读到请求而造成错误

通过彻底重写html的界面和重新命名对象, 该问题得到了解决

6、由于采用百分比布局, 导致浏览器在缩放时按钮会出现错位

我将它更新为了基于 absolute 和 flex 的布局, 即使浏览器比例改变, 电梯的布局也维持不变

5.4 仍然可以完善的部分

在经历了多次迭代和优化后, 目前代码的框架和结构已**几乎解决了所有的问题!**

基础功能得到解决的同时, 面临各种边界测试, 代码也表现出了较好的健壮性, 但仍然存在一个可以优化的点,即在主动开门关门时, 移除了对时钟的控制固然是好事, 但在一种情况下会出现差强人意的地方,这种情况需满足以下两个条件:

1、电梯的请求队列不为空, 即仍有运行的趋势

2、用户在某个经停层反复开门超过5秒

在这种情况下,电梯会忽略掉用户后续的开门请求而直接去执行下一个任务,即不允许用户反复开门

这样是不太人道的, 因为如果用户在某些特定情况下(比如说搬家过程中)需要电梯停留很久, 而我们的电梯不会进行等待
针对上述问题, 有一个初步的解决方案

1、首先, 将moveUp()、moveDown() 两个函数与 updateFloor() 函数合并,完成ui与逻辑的统一

2、之后,在updateFloor()前判断电梯门是否是关着的,只有关着的情况下才会进行移动

这样,即使反复执行run()函数, 逻辑和ui的层数都不会更新,就可以实现停顿了

由于时间关系, 未能将它实现, 希望后续可以有进一步的改进