**Program of Synchronized Elevators System**

Liu Yitao Computer Science Technology International School. Jinan University

**Abstract**

Elevator system is the common system used in our daily life. In most cases, there are more than one elevator working at the same time. So we need an efficient elevator management. In this program, we simulate the UI of elevator and use muti-thread in java to make every elevator moves independently.

**Introduction**

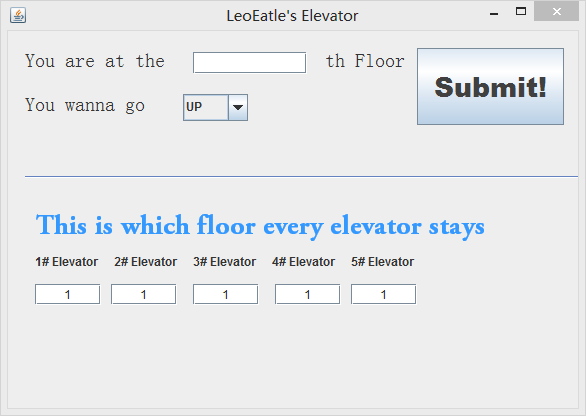
We use Java swing to simulate the UI of buttons in elevator. And use the java.lang.thread to make every elevator as a independent thread. And we need an algorithm to choose the elevator which is nearest to the user’s floor.

**Requirement**

1. Each elevator: There should be some buttons: the numeric keys, opened the door, closed key, up key, down key, alarm key, ascending signal, descending signal etc. There should be a digital display to indicate the current state of the elevator.
2. Each floor of the elevator door, there should be up and down buttons and digital display the current state of the elevator.
3. Five of elevator door button is interconnected, Five elevator door button interconnection junction, that is, when an elevator button is pressed the corresponding button of the other elevator also lit at the same time pressed.
4. Scheduling algorithm is based on the state of the five elevators, looking for the nearest and request the same direction of the current floor elevator and give a response.
5. All elevator initial state on the first floor. Each elevator if there is no corresponding request in its upper or lower case, it should be in place to keep still.

**Main Class**

1. Eleframe:



Eleframe is the class including the UI for users out of the elevator. It accept the user’s floor and user’s direction. And it will decide the elevator to carry user by a static method called *findelevato*r.

*findelevato*r:

*If*

*elevator’s direction is the same as user’s direction and the user’s floor is in front of elevator.*

*Or*

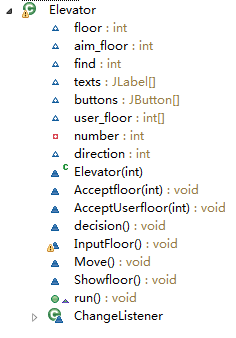
*the elevator is stopped.*

*Select the elevator and compare their floors with user’s floor to get the nearest one.*

And in this class I create a new thread to update the data in TextFiled at every second.

1. Elevator

This class is the main part in the program.

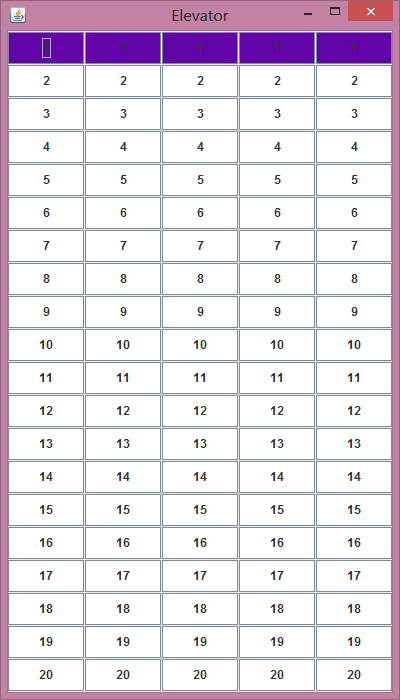


The Move method is to check if the “floor” is the same as “aim-floor” in every second. When the aim-floor is changed in Eleframe.class. The elevator will use “Move” method to change its floor number.

And the int array called “user\_floor” will stored the user’s input in the elevator. It will update the number of aim-floor in elevator object to keep it the biggest(when direction is up) or smallest(when direction is down).

1. Eleframe3

This class is to show the buttons in the elevator and accept the user’s input.



**Core Algorithm**

N

Y

The user input the location floor and the direction out of the elevator

Use findfloor method to select the correct elevator

The user input the aim-floor out of the elevator

Store the aim-floor(in a int[])

in the elevator object

Insure the aim-floor is the biggest(when direction is up) or smallest(when direction is down)

Aim-floor = this.floor?

Stop

Move

**Conclusion**

This paper shows that the muti-thread is a very useful tool to simulate some conditions in daily life. But we should also pay attention to the resources shared by all the threads.