Mini-assignment 3: Threads: The spaceship airlock

Handed out in Week 9: 24th March 2014

Introduction

This mini-assignment is designed to give you some experience of playing with threads. You will use Java threads to control the opening of airlock doors. In the spaceship an airlock is used to allow astronauts to space walk. We have to make sure that only one door into the airlock can be open at any one time, to stop the whole spaceship from depressurising.

This mini assignment (worth 10% of your mark) is to encourage your familiarity with Threads.

YOUR WORK MUST BE SIGNED OFF BY YOUR DEMONSTRATOR IN YOUR ASSIGNED PRACTICAL SLOT. YOU NEED TO HAVE THIS SHEET SIGNED OFF AT THE LATEST DURING WEEK 11 THE **WEEK BEGINNING 7th April.**

Worksheet Steps

There are several tasks to complete in this worksheet. Work through each task in turn.

Task 1: Download the code and load into an Eclipse project or equivalent

On Blackboard, you will find the starting code for this project. You need to download the files containing the AstronautSimulator class and the Door interface.

Note: you can use Eclipse or a different IDE, or none, if you want. You will need to modify this code to complete the assignment.

Task 2: Look at the existing code and fill in the blanks

Examine the Door interface. Then take a look at the AstronautSimulator class. This is the one you need to update. It will not compile because you will need to create an AirlockDoor class and an Airlock Class. The following steps you need to follow are also listed in the AstronautSimulator class:

Step 1: Create an empty AirlockDoor class. Note that you will also need to create an appropriate constructor that takes a String and an Airlock.

Step 2: Create an empty Airlock class. This one can remain empty for the reasons discussed in the code comments.

Step 3: Create two thread objects using the Thread class: one for each of the two doors. Also start the threads. You will need to do something to the AirlockDoor class, so that your code compiles. Just do enough to get it to compile.

Step 4: Make a request to open a door (hint: see the interface) and then add an 800 millisecond delay before trying to open another door. You will need to add a stubbed out method (empty and will need implementing in Task 3) to the AirdoorLock class.

Task 3: Implement the Door interface

Step 1: Make sure that the AirlockDoor class implements the Door interface. The method you are forced to add can simply record the open request in an instance variable. You will need to decide the type of this instance variable. Add a println statement to the method that says a request has been made to open the door.

Step 2: You will also need to implement the constructor. Simply store the parameter values in instance variables. These instance variables will be shared by threads. Refer back to the slides and lecture examples to see what implication this has on the way you should declare these instance variables.

Task 4: Implementing the run method

This is where most of the work is required. You will need an infinite loop. You will need to check whether a *door open* request has been made. Create a critical section of code that:

- displays a message saying that the door is about to open;
- waits for 5 seconds;
- displays a message that the door has closed.

It is important that you can "prove" with *println* or *format* statements that the two doors do not open at the same time.
Have a further 1 second delay before checking for the another *door open* request.
Run the simulator to show the program working correctly.

Task 5: Add packages and run from command line

Change all the classes to use suitable package statements. Make sure you use reverse-domain notation as discussed in lectures in Week 10. Demonstrate that you can run your program from the command line. Make sure the *AstronautSimulator* class is in a separate package to the other classes.

Task 6: Glory

For a further challenge, change the *AstronautSimulator* class to be a *Thread* class and remove the infinite *while* loop and then create several astronauts (threads) trying to gain access to the airlock.

Assessment

You must demonstrate what you have to a demonstrator AND submit your folder as a zip via BlackBoard so we have a record of people's accomplishments:

- Completing Task 2 so that everything compiles: all steps (2 marks)
- Sensible implementation of Task 3: all steps (2 marks)
- Implementing Task 4: the run method in full (4 marks)
- Task 5: the use of packages and running from command line (2 marks)

-END-