Assignment 2a

Programming Usable Interfaces - Spring 2009
Assigned: 3 Mar
Due: noon 12 Mar

Use all the widgets, alarm clock & report, and elevator simulation

The goal of this assignment is to make sure you can use all the common widgets, get you to do some simple design of an interface, and give you experience with a non-passive (real-time) interface.

Your assignment

You must write three programs and one report, which you will hand in like last time (submit via email). Please include comments in all of your code, otherwise we won't be able to understand it, and you will lose points.

Note: As in the previous assignment, use the Export feature of Flex Builder to create a zip file with all necessary files.

Program 1: Use (Almost) All the Widgets:

For useall, the widgets should be in a clear order, so that I know which one to interact with first, then second, etc. As I interact with the first, it clearly changes something about the second, then interacting with the second clearly changes something about the third, and so on. If you have multiple widgets that all want to be acted upon, but not act on anybody else, make them all be at the end of the chain, affected by the last one that handles input. You can do this in a mundane way, or you can try to be clever. I would not invest much time in it; this part of the assignment is basically pass/fail. Widgets you must include are:

- Button
- CheckBox
- HSlider
- Image
- Label
- List
- NumericStepper
- ProgressBar
- RadioButtonGroup
- TextInput

Program 2: A Hotel Alarm Clock (and a report)

Pretend Hilton is putting a PC with mouse & keyboard in every hotel room as the world's most expensive alarm clock. You must allow sleepy travellers, who will spend zero time learning your interface, to control a) whether the alarm is on, b) what time it wakes up, c) whether it wakes up to an alarm or music, d) anything else you think it should control. Naturally, the clock should show (and update) the current time, like all real alarm clocks do. As part of this assignment, you must try your designs out on 2 people and see if they can figure them out quickly. Real people are better than students. Prepare a report that describes your design(s) and your subjects' responses to it. Your grade will be based on design, user feedback received,

and implementation. The time on your clock should be accurate. Finally, the alarm may be a sound or a visual cue (changing a label to "RING RING!" or "MUSIC") is fine.

Apparently, there have been issues with people faking user tests in past years. Faking a user test is considered equivalent to copying someone else's answers, and will result in appropriate disciplinary actions.

Program 3: An Elevator Simulation

This program requires you to write an interface that isn't "passive," to use arrays, to use event handlers for multiple controls, to write a fun simulation, to handle boundary conditions (no elevators going through the roof, please!), and to use subroutines. Your program should show the user the current state of, and allow the user to interact with, a running elevator simulation. The user is given a "God's eye view," meaning that they are able to generate events that all the people in the building would generate. Specifically, the rules of the simulation are:

- There are three elevators.
- There are ten floors.
- Each elevator is always on a specific floor, and always "headed up" or "headed down"
- All elevators start on floor four, headed down.
- On each floor, there is a single "call button," and a light that shows if it's currently been pressed, but not serviced. When the user presses that button, the light goes on. When the floor is "serviced" the light goes off. If the user presses a light that's on, nothing happens (just like in real life :-) The user can press buttons at any time.
- Once per second, a single elevator should run the following algorithm. The leftmost elevator should go first, then a second later the middle one, then the rightmost one, then the leftmost one goes again, and so on.

IF the elevator is headed up AND there are pending calls above the elevator THEN

    move up one floor

    turn off the light on the new floor, if it's on (since it has now been serviced)

ELSEIF the elevator is headed down AND there are pending calls below the elevator

    move down one floor

    turn off the light on the new floor, if it's on (since it has now been serviced)

ELSE

    keep elevator on the current floor

    switch the direction it's going

    turn off the light on the current floor, if it's on (since it has now been serviced)

ENDIF

Until you press any buttons, your elevators should happily sit on the floor they start on; each second, one of the elevators will switch its direction (which should be visible as your simulation runs). You will be graded primarily on the correct functioning of your code, but also on the appearance and usability of your interface. Note that design does not make up for functionality on this assignment. If your elevators do not function exactly as specified in this handout, you will receive a poor grade. The

algorithm must be implemented exactly as described above. See me if you are not sure about how to do that. Once you have the basics up and running, feel free to embellish the simulation in reasonable ways. The entire state of the simulation should be visible at all times.

Note: although functionality is strictly graded, you shouldn't necessarily code first and design later. If you design for usability first and plan your implementation around your design, you're much more likely to have a functional and usable interface.

For help with using arrays, find a suitable tutorial on the internet. Make sure your code is written in a general way, allowing easy modification at a later time. Starting early is highly recommended; if you have questions, email the instructor or post a message in the forum.

Turning in

Submit your assignment via email to the instructor. Be sure to name the files like this: [firstname]-[lastname]-hw2a-projectname.zip. For example, Vassilis Kostakos would submit these files:

- vassilis-kostakos-hw2a-useall.zip
- vassilis-kostakos-hw2a-alclock.zip
- vassilis-kostakos-hw2a-alclockreport.pdf
- vassilis-kostakos-hw2a-train.zip