CS 301: Assignment 3 – 15-Squares Puzzle

In this assignment, you will implement a simple one-player puzzle titled 15 Squares. In the puzzle, there are 15 sliding squares, numbered 1 through 15, placed randomly on a 4x4 grid. One square is missing. The goal is to arrange the squares so that the numbers are ordered, and the missing square is in the lower-right corner, as in:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

The only legal moves are the sliding of a numbered square to the adjacent blank square.

Basic assignment

For the basic, bare-bones assignment, you need to create an Java Android application that:

- [10%] Initializes the puzzle by randomly placing the 1-15 squares on the grid.
- [30%] Displays the puzzle and allows the user to play.
- [20%] When the numbers are correctly placed, give an indication that they are correctly placed (e.g., changing the color of the background).
- [10%] "Reset" button that randomly scrambles the puzzle.
- [10%] Play should be "intuitive".
- [10%] Code is commented and conforms to the 301 coding standard.

If you implement the basic assignment correctly, you can expect to receive a grade of 90%. You have a fair amount of freedom in the layout and method of interaction. The simplest is probably for the user to touch a square that is adjacent to the blank square, which causes that square to move to the blank square. It is probably more intuitive, however, to allow the user to drag the squares [+5%].

Enhancements

There are several enhancements you can do to improve your grade, including:

- Instead of randomly initialing the puzzle at the start, randomly initialize it with the restriction that the puzzle is winnable. (There are certain initial configurations for which the puzzle is not winnable. See http://mathworld.wolfram.com/15Puzzle.html for more information.) [+8%]
- Write your program so that the size of the puzzle can be changed. For example, a user could specify a 7x7 puzzle (with 48 numbered squares) or a 9x9 puzzle (with 80

numbered squares). It is up to you to decide how the user should specify the puzzle size (and to document it so that it can be tested during grading). [+10%]

- Instead of squares on a square board, implement the puzzle with hexagons on a hexagonal grid. [+5%]
- If the user does not name a move within in 10 seconds, have the program make a random legal move on behalf of the user. (To do this, consider using the java.util.Timer class.) [+5%]

Hints

There are a number of ways that you can choose to implement this project. Perhaps the most straightforward (but not as user-friendly) is to use 16 *Button* objects in a 4x4 *GridLayout*. An alternative would be to draw your rectangles on a *SurfaceView*, and to allow the user to move squares by dragging them.

Notes

Comment your program. Make sure that you include a comment at the beginning of the file so that it includes your name, date, etc. You should also keep the comments in your code consistent with your modifications to the program's behavior. The difference between good documentation and poor documentation can easily be a full letter grade.

If you do any enhancements, clearly describe them in the header-comment in your main activity class file, just below the name and date. I'm sure you can think of others. If you have an idea, please run it by me. (There might be some hidden "gotchas" that I can help you navigate through.)

Handing it in

Submit a link to your github repository. Be sure to have made your final commit and push before the deadline.