Assignment #4 – Minesweeper

Date assigned: Friday, January 9, 2015
Date due: Java Minesweeper, Tuesday, January 13, 2015
Android Minesweeper, Friday, January 17, 2015
Points: 100

Java Minesweeper

The game minesweeper is a single-player game with the object of clearing a minefield without detonating a mine. The purpose of this assignment is to write a complete Java version of minesweeper in a project called MinesweeperJava. The logic (and code) gained from this assignment will then be used to implement an Android version of minesweeper later in the course.

Details of the game can be found at:
http://en.wikipedia.org/wiki/Minesweeper_(video_game).

Details of our Java game:

1. Set a constant to determine the dimension of our grid. Set the dimension to 9 initially giving us a grid of 9x9 or 81 squares.
2. Ask the user to input a difficulty level (EASY is 0, MEDIUM is 1, and HARD is 2)
3. Set the number of mines based on the difficulty level:
    a. EASY (specified by 0) is the dimension of our grid or 9.
    b. MEDIUM (specified by 1) is the dimension of our grid plus 1 times 3 which is 12.
    c. HARD (specified by 2) is the dimension of our grid plus 2 times 3 which is 15.
4. Some details of the game:
    a. Bombs are placed in random positions on the grid and identified with a value of @.
    b. During a turn, the user selects a cell to reveal the cell's contents
        i. A cell that has adjacent bombs is the only cell revealed and a number indicating the number of adjacent bombs is displayed in the cell the next time the grid is output.
        ii. A cell that has no adjacent bombs is marked with a SPACE. If a cell has no adjacent bombs, then all adjacent cells (including the diagonals) are looked at. For each of the adjacent cells, then either i. or ii. applies. Yes, this is a recursive definition although you do not need recursion to solve the problem.
        iii. A cell selected by the user that has a bomb terminates the game

c. If all non-bomb cells are identified with a value other than the initial value (a . in the following example) before a cell with a bomb is selected, then the user wins; otherwise, a bomb was hit and the user loses.

Here is an example of your game play.

```
***********
Minesweeper
***********

Enter difficulty level
(0 = EASY, 1 = MEDIUM, 2 = HARD): 0

  0   1   2   3   4   5   6   7   8

  .|  .|  .|  .|  .|  .|  .|  .|  .   0
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  @   1
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  .   2
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  .   3
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  .   4
-----------------------------------
  @|  .|  .|  .|  .|  .|  .|  .|  .   5
-----------------------------------
  .|  .|  .|  @|  @|  @|  .|  .|  .   6
-----------------------------------
  .|  @|  .|  .|  .|  .|  .|  .|  .   7
-----------------------------------
  @|  @|  .|  .|  .|  @|  .|  .|  .   8


Enter X and Y Coordinate: 3 8

  0   1   2   3   4   5   6   7   8

  .|  .|  .|  .|  .|  .|  .|  .|  .   0
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  @   1
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  .   2
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  .   3
-----------------------------------
  .|  .|  .|  .|  .|  .|  .|  .|  .   4
-----------------------------------
  @|  .|  .|  .|  .|  .|  .|  .|  .   5
-----------------------------------
  .|  .|  .|  @|  @|  @|  .|  .|  .   6
-----------------------------------
  .|  @|  3|  2|  4|  .|  .|  .|  .   7
-----------------------------------
  @|  @|  2|   |  1|  @|  .|  .|  .   8
```

```
Enter X and Y Coordinate: 0 0

  0   1   2   3   4   5   6   7   8

   |   |   |   |   |   |   |  1|  .   0
-----------------------------------
   |   |   |   |   |   |   |  1|  @   1
-----------------------------------
   |   |   |   |   |   |   |  1|  1   2
-----------------------------------
   |   |   |   |   |   |   |   |      3
-----------------------------------
  1|  1|   |   |   |   |   |   |      4
-----------------------------------
  @|  1|  1|  2|  3|  2|  1|   |      5
-----------------------------------
  .|  .|  .|  @|  @|  @|  1|   |      6
-----------------------------------
  .|  @|  3|  2|  4|  .|  2|   |      7
-----------------------------------
  @|  @|  2|   |  1|  @|  1|   |      8

Enter X and Y Coordinate: 5 8
Boooom!!! You lose.
```

If the player wins, then display the message "Congratulations. You win."

Notes:

1. Make sure the user enters 0, 1, or 2 for the difficulty level. If an improper selection is made, display the input message again and continue until a proper response is entered.
2. Make sure the user enters a valid position on the Minesweeper board. If an invalid position is entered, display the input message again and continue until a valid position is entered.

Goals for Java Minesweeper:

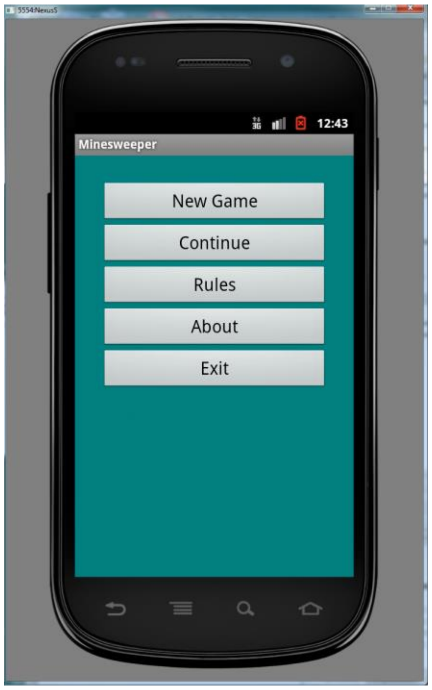
1. Write a Java application using multiple classes
2. Use packages to better organize all classes
3. Use good OOP techniques in designing your solution
4. Use the Java API which has a rich library of routines (e.g. Vector, Stack)
5. Use JUnit framework for testing classes

Android Minesweeper

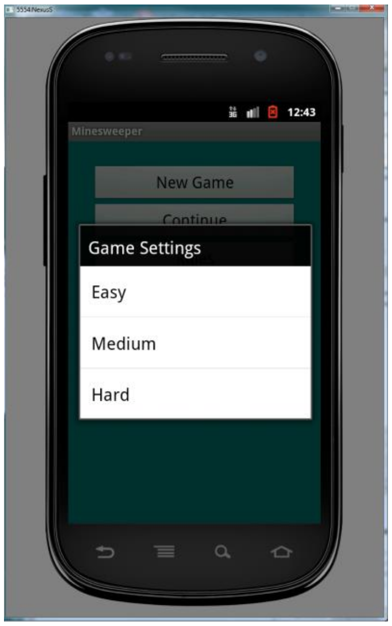
The purpose of this part of the assignment is to write a complete Android version of minesweeper. You have the game logic in Java. Now you can incorporate that game logic into a graphical version of minesweeper on Android devices.

Minimal Requirements:

1. The opening screen must look like. You can use colors you like.



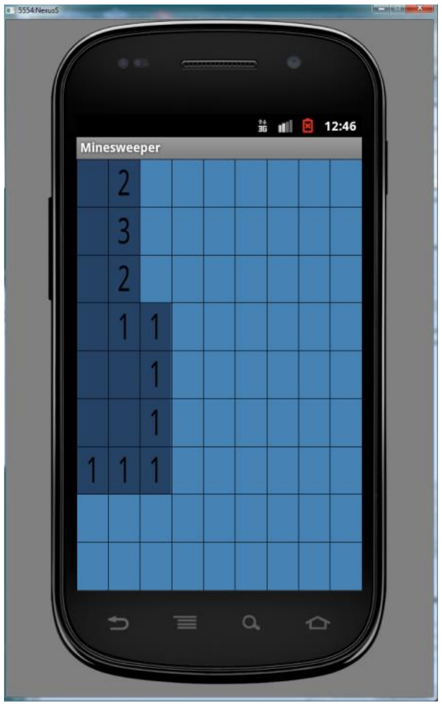
2. The **New Game** button is to create a new game of minesweeper after getting an Easy, Medium, or Hard game setting.



3. The **Continue** button does not need to be implemented.
4. The **Rules** button is to display the basic rules of minesweeper. Specifically, explain the number of bombs placed on the board. In a perfect world, I would

like the following: (a) Easy allows the user to select a tile and at least two uncovered tiles are displayed and no bomb is possible, (b) Medium displays at least one tile and no bomb, (c) Hard means the user can hit a bomb right off the bat. In this section, tell me whether you implement this or not.

5. The **About** button is to contain information you deem relevant regarding this application but you must be listed as the author with an email address and give a version number for the game.
6. The **Exit** button simply exits the game.
7. Assume that you have started a game and selected a square in the upper-left corner of the screen, you must display what happened that turn. Here is an example of what your screen might look like.



There are a couple of ways to make this happen.

a) You can draw text over a highlighted tile. I will give you some code in class for centering text in a region of the screen.
b) You can draw a bitmap into a selected region. In this case, you will need to find/create bitmaps for all of the possible numbers with appropriate highlights.

8. Visiting Tiles,
    a. If the player visits all cells without selecting a mine, then the user wins. In this case, display something (professional looking) signifying that the user won.
    b. If the player visits a mine, then the player loses. In this case, display the board showing all visited tiles and show where all mines were at. I want to be able to see all tiles on the board.
9. Create (or grab) some graphic for your minesweeper game which is to be displayed with all of the other applications. That is, don't use the default Android launcher icon. Note in the About where the icon came from.

Goals for Android Minesweeper:

1. Write an Android application using multiple classes
2. Use packages to better organize all classes
3. Use good OOP techniques in designing your solution
4. Use the Android API which has a rich library of routines
5. Use the techniques learned in Game to implement an Android game
6. Include other projects in an existing project
7. Debug with LogCat

Specifics:

1. Create a folder called your PUNetID, place a copy of JavaMinesweeper in your PUNetID folder when the assignment is due, and drop the PUNetID into CS260-01Drop on grace. Do the same for AndroidMinesweeper. Your code is to be written using the development tools specified in the syllabus and using the Java coding standards on the course Web page.

2. If you come to me with a question regarding your solution, I will have you load your project onto a machine in the CS lab. I will not look at your code on your computer or on paper as it just takes me too long to get at the problem. Further, I want you to bring in your lecture notes in case I want you to look up something. Remember, I'm not just a tell you the answer guy. Make sure you understand how to use the developer tools and that you can run your program in Eclipse.

3. If you want help with a compiler error, you must be able to tell me exactly what statement you put in your code that caused the error and be able to isolate the error. If you have typed in a bunch of code and have not tested your code as you've gone along, I'm not going to help you sort out the mess. You've been warned!!

4. IntelliJ keyboard shortcuts https://www.jetbrains.com/idea/help/keyboard-shortcuts-you-cannot-miss.html