BSC-MD Assignment 2: MineSweeper

Assignment Information

Course:	BSC
Stage: /	
Module:	MD Semester: 2
Title:	MineSweeper
Description:	Create an application to implement the Minesweeper game.
Assignment:	Assignment 2
Date of Issue:	Thu. 19. March 2020
Assignment Deadline:	Sun. 12. April 2020
Assignment Submission:	Upload to Moodle
Assignment Weighting:	This is the second of three assignments and is weighted at 17.5% out of a total of 50% for all three assignments.

Introduction:

In this assignment you will be tasked with building a custom view that will render and implement the rules of minesweeper. Assuming you are not already familiar with minesweeper, the premise of the game is simple. You are given a two dimensional grid representing a minefield. However all of the cells are covered so you cannot see what is underneath. To sweep the minefield you must uncover cells one after the other until all non-mined cells are uncovered. Non-mined cells will include a number indicating how many adjacent cells contain a mine relative to that cell. If there are no adjacent mines a number will not be displayed.

Players should be able to use a combination of a logical reasoning and a little bit of guesswork to determine what cells contain mines. If a cell contains a mine they should be able to mark it as such. Any time a player clicks on a marked cell (usually by accident) it will not uncover. Unless the user explicitly unmarks it.

You will be tasked with building a fully working game of minesweeper running on a 10x10 board with 20 mines. Players should be able to reset the game, switch between marking and uncovering modes and have some information displayed to them about how many mines they have marked and how many mines there are in total.

If you are not familiar with Minesweeper, then play it for about 20 to 30 minutes to make yourself familiar and then get to working on the application. Please read through the rest of this assignment before you start coding straight away.

Notes:

Take note of the milestones. If you skip a milestone or have serious errors with a milestone, further milestones will not be considered for correction. e.g. if you skip milestone 3 or have bad errors with it then milestones 4, 5, 6,…. will not be considered for correction. With regard to errors, depending on their severity, these will result in deductions of 5, 10, or 15% for each error. Thus it's possible that if you implement all milestones but have 20 small bugs you can end up with a poor score. This is to encourage you to build robust software free from bugs.

Assignment Deliverables
You are required to upload an archive of your complete project to moodle by the date specified on page 1 of this document. Archive the (outer or root directory of the) project using the .zip archive format. Do not use any other archiving format.

You are also expected to have a git repository with your work. Git will serve as a backup of your work and also a rollback facility if you have serious errors in the source code. You are expected to make an initial commit immediately after the project is created and at least 1 commit for each milestone (when you finish a milestone). At the beginning of your commit comment write "Milestone X:" where X is the number of the milestone.

Penalties:
There will be penalties applied to your score if any of the following conditions are not met. These are silly things that slow down my correction of your work. Thus if you want your result back in the quickest possible time make sure you do not fall foul of these penalties.

- -40% for a missing git repository. Make sure when you go to submit your work that there is a .git/ directory in your project before archiving. This will show a development history on the project. You will also lose marks for missing commits.
- -30% for non compiling code. It should be possible to open your project have it compile and run immediately.
- -10% for a non supported archive format.
- Standard late penalties

Milestones:
As noted before make sure that you complete milestones properly before moving onto the next one. The percentage in brackets indicates the maximum score you can achieve if you pass that milestone.

1) Define the shell of a custom view class and generate a layout for the main activity consisting of the custom view, two buttons, and two textviews. **(10%)**

2) Draw the initial state of the game board with all cells covered. Black should be used as the fill colour and white lines should separate the cells **(20%)**

3) Implement the basic touch behaviour that will uncover a cell. When a cell is uncovered it should change from a black colour to a grey colour.
(40%)

4) Implement methods to place 20 mines randomly in the minefield and render the mines when a cell is uncovered. A cell containing a mine should have a red background with a black M in the foreground.
(60%)

5) Modify the touch behaviour to stop accepting input when a mine is uncovered. It should only re-enable input when the user has clicked the reset button. **(80%)**

6) The other button should switch between displaying “uncover mode” or “marking mode” each time it is clicked. In uncover mode each touch should result in a cell being uncovered. In marking mode each time a covered cell is touched, it should change to yellow to denote it is marked or back to black, to denote it is unmarked. Use the two text fields to denote the total number of mines and the number of mines marked. In uncover mode, touching a marked cell should do nothing.

(100%)