The MineSweeper Game

Homework3

Başar Temiz

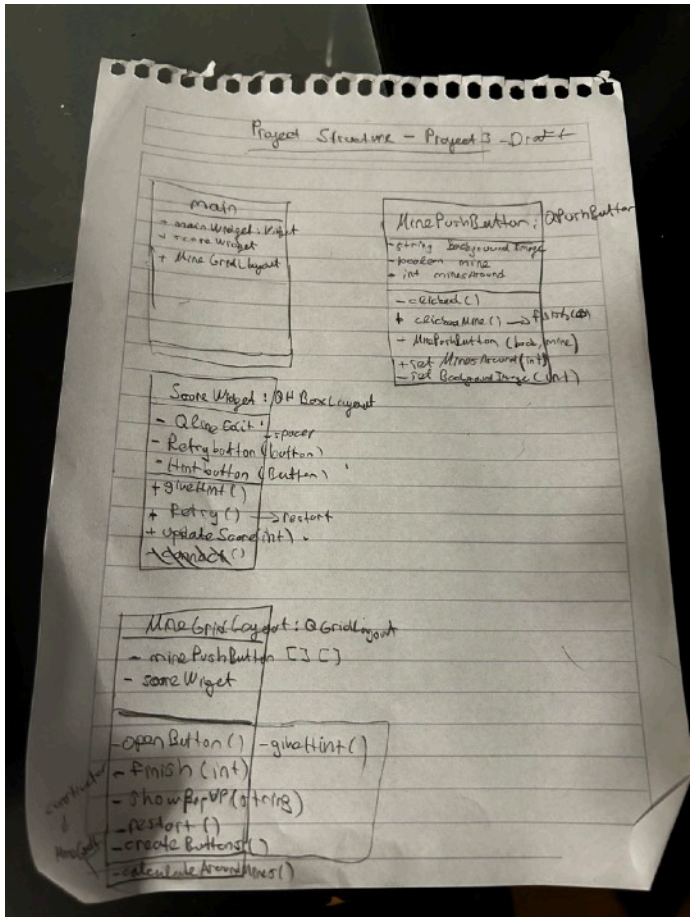
28 May 2024

The main project structure and main classes

Firstly, for this project, I have used mainly classes to implement functionalities and used the main class for instantiating the used classes.

I used to find creating a main structure before diving into coding, simplifies and makes the code more understandable and maintainable. As a result, I created a project structure on paper first. The photos used for describing the classes may not represent all the functions and the variables used, they are merely a draft of the real classes that are to be implemented.

Project Structure:



So the project structure has 4 main classes (which will be detailed later on) which are main, ScoreWidget, MineGridLayout, and MinePushButton.

The MinePushButton Class : QPushButton

This is a class for custom cell buttons. Each cell button has a certain features such as

-string backgroundImage

-boolean mine

-int minesAround...

However I want to discuss the signals, slots, and functions that are important to address because these are the features that implement relevant functionalities.

Signals:

ClickedMine:

This signal goes to our main layout class's finish function which has MinePushButtons.

ClickedEmpty:

This signal goes to our layout class's revealEmpty function which has MinePushButtons.

Functions:

openCell():

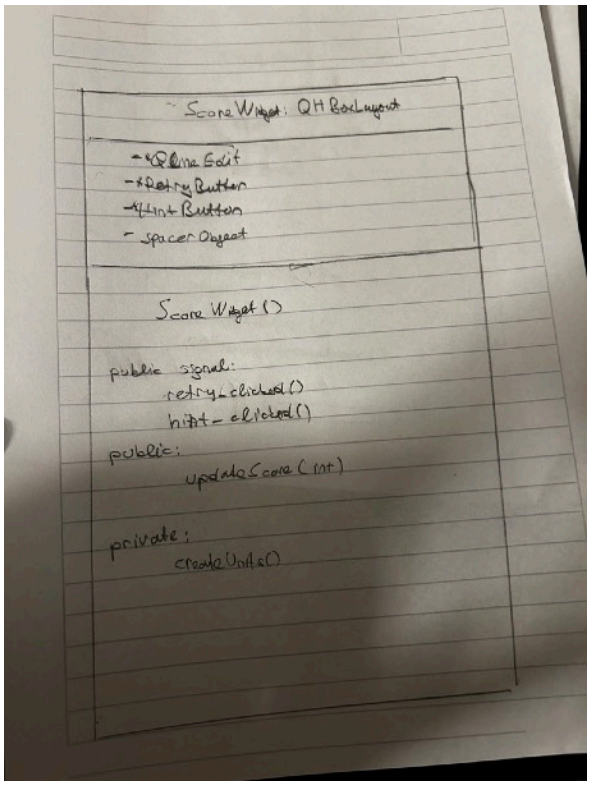
Opens the cell and reveals the number of mines around this cell.

MousePressEvent:

This function is handling the different clicks (right click and left click). So if the user right-clicks the button will set the flag to true or if the user clicks right the button will open, or if it has a mine then the game ends

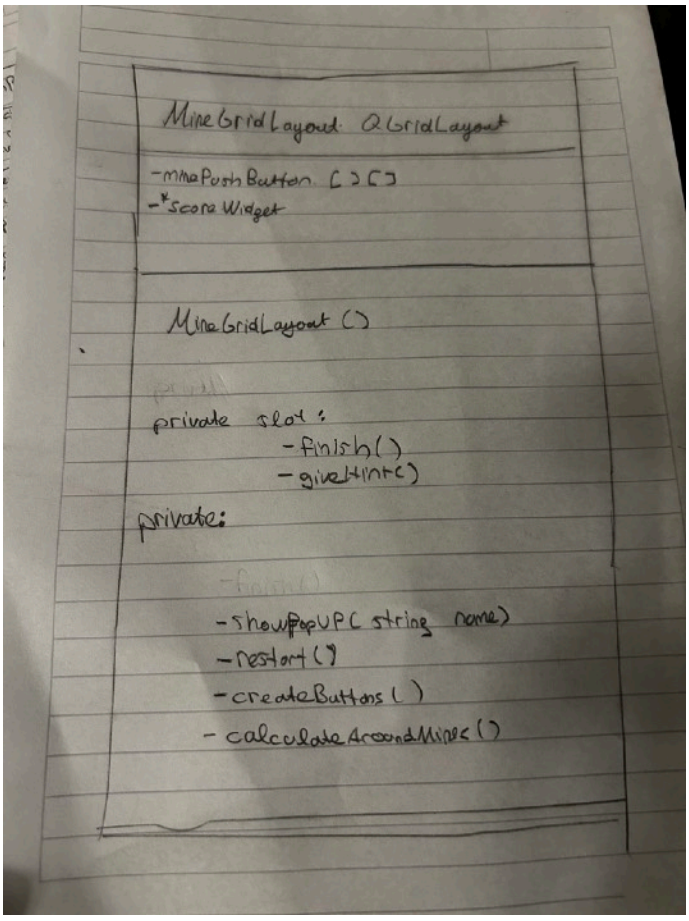
The ScoreWidget Class: QHBoxLayout

This is a class containing the hint and restart buttons, and the score label. The class inherits QHBoxLayout which helps to maintain the order of the elements. This class just uses signals to pass information to the main class that implements functionalities (MainGridLayout). The only important functions are functions that change the score label which is updateScore(int) and zeroScore().



The MineGridLayout Class: QGridLayout

This is the main class in which the important functionalities were implemented and the signals from MinePushButton objects and the ScoreWidget objects get sent. This class contains a scoreWidget and an array of MinePushButtons.



The GiveHint method

Before discussing this function, I created another function in the MineGridLayout class to simplify the giveHint function. I created the checkCell() method which does the following: Checks if the given uncertain cell has certainly a mine by checking the neighboring opened cells if the number of mines around the opened neighboring cell has the same number as neighboring uncertain cells(e.g. the opened neighboring cell has 2 mines around it and this cell 2 uncertain cells then the uncertain cells must have mines).

So the giveHint method iterates the opened cells one by one and checks the neighboring cells of the opened cells (by checkCell() function) if there are more uncertain cells than the mine count around this cell and the number of uncertain cells that has mines is equal to the number of mines count around this cell, than the cell that does not have definitely a mine is for sure does not contain one. (e.g. an opened cell with 2 as the mine count, 3 uncertain cells around it, if there are 2 mines within these 3 uncertain cells then it is for sure the cell that has not definitely a mine does not contain a

mine). The method outputs this cell and makes this cell go into the hintMode(a function in MinePushButton).

The Main Class

This is the class where I instantiated the relevant classes and objects. I used a main QVBoxLayout to contain the widgets used in the project.

Various problems and solutions

The main problem occurred when I tried to use a button size which is bigger than 15x15. The first solution for setting the image to the button was using style sheets, however, it resulted in repeated image-like behavior. Then I tried to use Pixmap, but this failed also because the images wouldn't just get bigger. Then I used scaledPixmap, and this worked. However, it was quite hard just to put an image on a button which is a basic task. Then there was the issue of spaces between the buttons which resulted in a visual slightly different from the original game. This is because of the use of scaledPixmaps. I solved this by using a fixed size and disabling the resize option of buttons. I gave a quite bit of effort into making the button size adjustable however for some reason there was always a little horizontal space between buttons, which I couldn’t solve probably because of the inner codes and structure of the pixelmap and its relation with the layout widget.