CMPE230
22.06.2020
EGE CAN KAYA - 2018400018

CMPE230 SYSTEMS PROGRAMMING HOMEWORK 3

Problem description: In this project, we were asked to implement a GUI pairs-match game using Qt, with a functional timer, scoreboard, new game and quit buttons and a 6x5 grid of cards.

Problem solution: I wrote four different classes, and an additional cpp file for the main window. These classes are: *Card*, *Grid*, *Scoreboard* and *Timer*.

- **Card:** These constitute the playing cards in the matching game.It is a subclass of the QPushButton class.

  It has two fields: the *QString hiddentext*, which holds the word to be displayed then the Card is clicked on, and the *QStack<Card*>* curclicked*, which is a stack that holds up to two last pressed Cards.

  It has two slots: *void reveal()* and *void matched()* and one signal *void checknow()*. *void reveal()* changes the display text of the Card from the string "?" to its hiddentext field. If no other Card was clicked before this one, i.e. if the curclicked stack is empty, then this change stays on indefinitely. If there does exist a Card in the curclicked stack, then this slot emits the signal *void checknow()*, which will be received by the Grid of the game to check whether the two Cards were a match.
  The other slot, *void matched()* is activated if, indeed, the last two Cards were a match. In this case, the Cards are disabled and their hiddentext is displayed for the rest of the game.

- **Grid:** A subclass of QGridLayout, this holds 30 Card items in a 6x5 layout.

  It has one field: *QStack<Card*>* curclicked*, which serves a very similar purpose to the one in Card. It holds up to 2 last pressed Card items.

  It has three slots: *void check_match(), void end_lost(), void restart()* and one signal: *void gridmatch()*. The *void check_match()* slot is triggered whenever two Cards are clicked in a row. This checks if the text on the two buttons in *curclicked* are the same, and also that these are not the same Card (otherwise, double clicking a Card would disable it). If they are a match, the signal *void gridmatch()* is emitted, which is received by the Scoreboard to increment score by 1. Also, the *void matched()* function of both of the cards are triggered to disable them. If they are not a match, a QEventLoop makes the two buttons display their text for 500 milliseconds, after which they are again covered with "?".

  void end_lost() ends the game with a loss scenario, which happens when the Timer reaches 180 seconds. It does so by disabling all of the Cards in the grid and revealing their texts.

  void restart() quits the app and restarts it from scratch. This is triggered when the New Game button is clicked.

- **Scoreboard:** This class displays the player's score. It works very similarly in principle to the Timer class. The only difference is that it is incremented not each second, but each time when the *void gridmatch()* signal is emitted by the Grid class.

It has three fields: *QString label* which is the text display for the score, *int counter* which is the numerical score and Q*Timer timer* which is the QTimer object of the Timer class, i.e. the timer of the game. This must be passed in so that the timer can be stopped when the game ends.

It has one slot: *void increment()* which does two things: It increments the *int counter* each time the *void gridmatch()* signal is received, and updates the label accordingly. Also, it checks whether the score has reached 15 each time it is called, in which case it ends the game with a win scenario. It does so by displaying a message box telling the player that they have won and stopping the timer.

- **Timer:** This class taken almost directly from the code written in the PS. It uses a QTimer, a QString and an int to keep track of and to display the number of seconds elapsed since the beginning of the game. It has a single slot *void finished()* for the increment operation and also to check at each second whether the counter has reached 180. If it has, it ends the game with a loss scenario by notifying the player with a message box telling them that they have lost, and emitting the signal *void lost()*, which is received by the Grid object of the game to disable all of the Cards.

- **MainWindow:** This is the main window of the game. It creates instances of every class implemented, puts them in a proper GUI format using box layouts and spacers, creates the necessary connections between the signals and slots (such connections are explained above), sets the dimensions of the main window etc. It also has another important function. It instantiates 30 card items and using the *void srand(int seed)* and *int rand()* functions of the C++ language, it assigns a random coordinate to each Card in the Grid, which changes with every run of the program. Using a 2-d array of boolean values of the occupied positions in the Grid, and a while loop to make sure the coordinates are randomly rerolled if the candidate position assigned is already taken by another Card, it distributes each card randomly. It also assigns 15 distinct words to the 30 cards using a switch-case construct, some of which are *"robot", "ball", "turtle"* etc.