CmpE 230 Project 3

Barış Başmak Kayacan Vesek
2016400087 , 2016400114

May 15, 2019

1 Introduction

Find the Pair is a challenging concentration memory game that requires players to spot the identical pairs in each level. Match pairs of shapes together with the help of your memory. Instead of cards in this game there is a 4x6 grid and, all the 24 buttons have a secret letter which will be shown when clicked on them. So you have to pair these 12 different letters between A-L.

2 How To Run

Option1: On command line go to the directory where the project is. On a bash command:

```
$ cd mydir/ev
$ qmake -project
$ qmake
$ make
$ ./ev
```

Please refer to https://wiki.qt.io/Getting_Started_on_the_Commandline for more details.
Option2: Open QT Creator and build the program. Then, click run button.

3 How To Play

It is pair matching game, in this game there will be a 24(6x4) button of objects on the screen. Totally 12 pairs of objects, pairs will be totally identical, and you need to find out the two as quickly as possible.

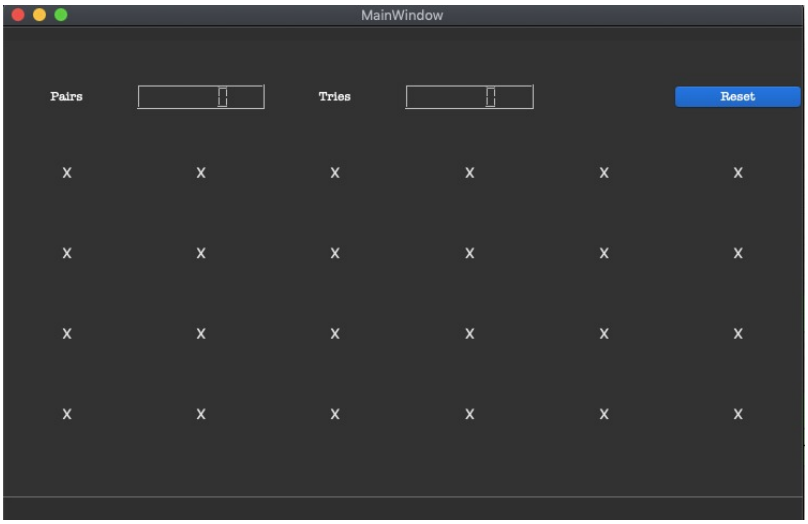

Figure 1: Example Game Play, Starting Position of The Game

At the start of the game, all the cards will be closed and you can click and open any card you want.

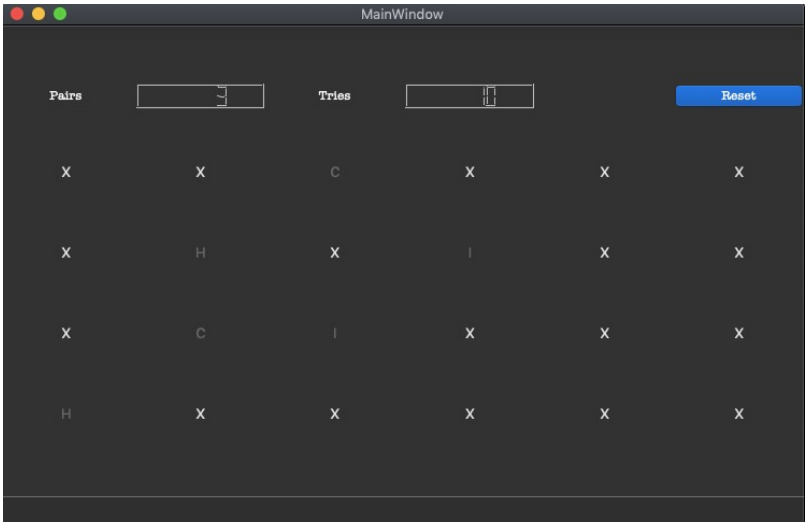

Figure 2: Example Game Play, middle of The Game

At anytime you can restart the game by clicking Reset button, and all the cards will be reshuffled, and other stats will be erased (Pairs, Tries)

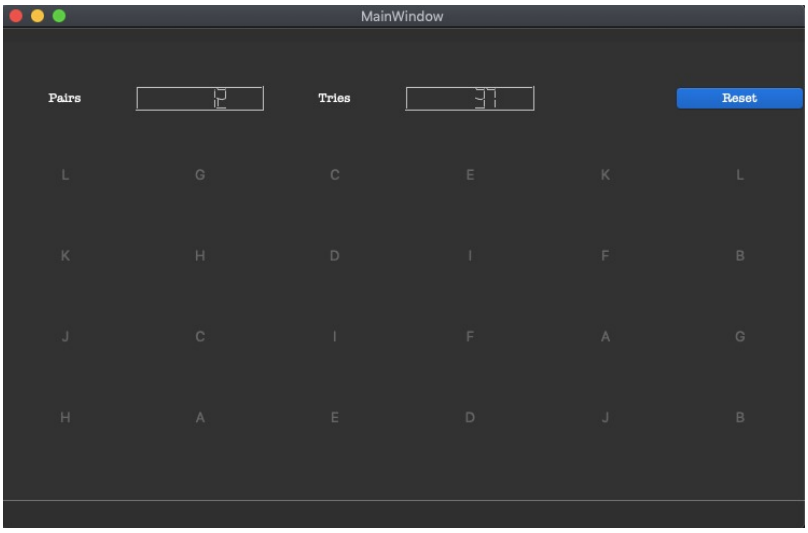

Figure 3: Example Game Play, End of The Game , GG

4 Implementation

In main.cpp only consist of one loop which callse mainwindow object.

```
do {
 QApplication a(argc, argv);
 MainWindow w;
 w.show();
 currentExitCode = a.exec();
} while( currentExitCode == MainWindow::EXIT_CODE_REBOOT );
```

In mainwindow.cpp, the interface design is connected to the code. In constructor, buttons are connected to specified slots for some actions. **There are 3 connection**

```
QObject::connect(array[i], SIGNAL(clicked()), this,SLOT(push_clicked())) ; // array goes
QObject::connect(ui->resetButton, SIGNAL(clicked()), this, SLOT(resetGame())) ;
QObject::connect(ui->resetButton, SIGNAL(clicked()), this, SLOT(slotReboot()));
```

For example: The button with name resetButton calls resetGame() slot when it is clicked.
There are 3 slots

```
public slots:
    void push_clicked();  // Slot for when a card's clicked
    void resetGame() ;    // Slot for resetting the game
    void slotReboot();    // Slot for resetting the game
```

Slots

- **push_clicked** This slot makes the actions when a card's been clicked,
  At start of the function Makes all the buttons unclickable so that a user doesn't click faster than the game can react and cheat.
  There are two options; If the click number is an odd one , then it shows the card. If the click number is an even one, it opens the second card and check if it matches with the first one, depends on the match it makes the action.

- **resetGame** Resets the game when clicked on reset button, it is connected in the constructor. Makes the parameters the correct values. Prepares the buttons for the new game and makes them clickable if they were disabled.

- **slotReboot** exits form the MainWindow with this command "qApp-¿exit(MainWindow::EXIT_CODE_REBOOT);"

RNG struct

RNG is a struct so that the random_shuffle function doesn't give the same result for all the games. And there is srand(time(0)) command at the start of the constructor gives time based seed to random function.

5 Summary

Our code gives the desired result successfully.
The game resets when pressed reset and the game part works according to the desired qualifications.