Bogazici University

CARD MATCH GAME
DOCUMENTATION

Ahmet Fırat Gamsız
2020400180
Department of Computer Engineering

Ali Tarık Şahin
2020400207
Department of Computer Engineering

CmpE 230 Systems Programming
Prof Dr Can Özturan
May 23, 2023

Introduction

Card Match Project is an implementation of a popular game where one tries to match pairs. In this project, it is implemented in QT with integrated graphical user interface using C++ language.

Purpose

This project aims to integrate a card match game in a window, which is ensured to run correctly on all systems thanks to QT framework. Also, since this project is implemented as a window-based application, the peculiar attributes of the windows of systems are preserved and provide a peculiar look for the game in different systems.

Design

The design of the project mainly consists of two parts. The first part to consider is graphical interface. In this part, the visual of the game is designed. All of the elements are placed according to the corresponding layout. 30 push buttons are placed in a grid layout and other game related indicators are placed in a horizontal layout on top of the window. On the other hand, the second part is to assign features and functionality to these elements. At this stage push buttons gain their confidential animal names, indicators get set into their initial values and in accordance with events they get updated.

Implementation

Project's implementation utilizes different concepts including slot-signal logic, default slots of Qt objects, randomization, and intrinsic fields of Qt objects. There are integers holding score and number of tries left; original card names and current order of cards are held in one dimensional vectors. Card buttons named (different than card names mentioned) as pushButton_i such that i starts from 0 to 29. UI was designed in the designer.

Initially MainWindow's MainWindow function is called for initialization. This function collects all the cards from MainWindow using findChild function. Dealing is done with rand function, cards' names are held in previously mentioned vector, and cards' released signal is connected to CardPressed function. Also score and tries in UI are reset and all cards closed with "?" mark.

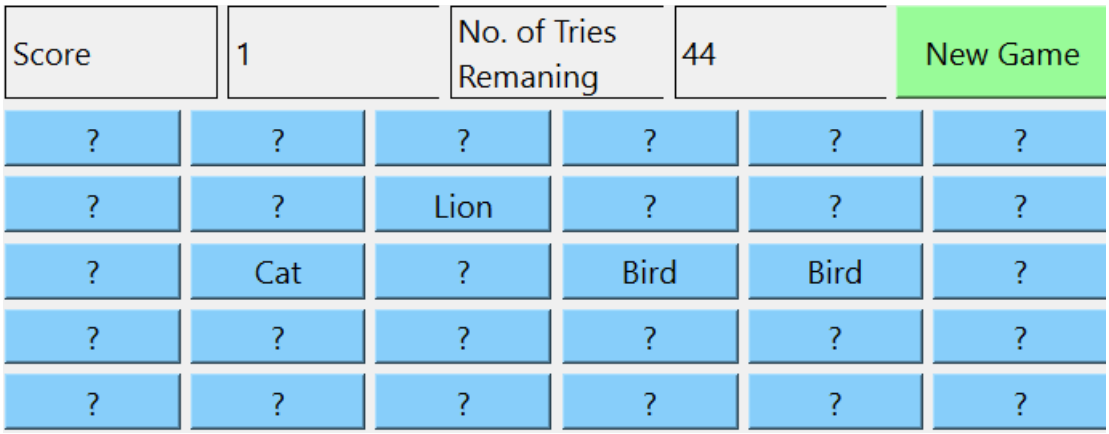
CardPressed function receives a card button object and gets its card name by using the integer in the card button's object name as index for the current card names vector. Buttons are disabled when released and their name revealed to user. The last two open cards get closed when the third one pressed. Open cards are held in a vector and if the user succeeds in finding a pair the first element of the vector is nulled. When the user opens the first card, the first element of the vector is checked to close the cards or keep them open. When the user opens the second card, number of tries is decremented. If the card names in the vector are the same a pair is detected, and score is incremented. When the user finds all pairs new game button changes its text to win message. If tries reaches zero new game button shows a loss message and all cards are opened and disabled.

New game button calls the on_newButton_released function. This function resets the message of the button and does almost the same with MainWindow function. Additionally, it nulls the open cards vector's first element, clears the current cards vector closes all the cards. (Opening a card means setting its text to its name and disabling it. Closing means the reverse.)

How to Use

- The purpose of the user is to match all pairs in at most 50 tries.
- The game includes a total of 15 animal kinds in pairs.
- All of the animals are randomly placed in the 5x6 table.
- If two opened cards are a match, then the score is incremented by one and cards remain open.
- If two opened cards are not a match, then score remains the same and cards get closed.
- Users who can successfully detect 15 matches (all the matches) before running out of tries win the game.

Gameplay Screenshots



Difficulties Encountered

The process where cards get open and close was a bit tricky in the sense that when two cards are not a match they need to be closed, but if one tries to set a time limit to solve this issue, some other bugs pop up. For instance, in the case of a time limit, a user may click cards faster than the period of closing, which breaks the game. However, we were able to solve this issue by arranging this wait process according to another event's trigger, in this case until a third card gets opened, the other two remains open.

Discussion

What is the advantage of using embedded UI?

As you might know, in earlier versions of QT user interface is mostly built using C++ and QT layout widgets. However, this was a cumbersome process which took a lot of time. Thanks to QT Creator, the UI part of the project could be done in a much shorter time using drag and drop approach in provided design mode. After setting your layout, it is easy to integrate it to the remaining of the code.

Conclusion

Using the advantages of the C++ programming language and QT graphical user interface framework, we were able to produce a window game which has an elegant look and can be executable on every type of system.