**2023 Spring OOP Assignment Report**

과제 번호 : #5

학번 : 20220923

이름 : 이윤혁

Povis ID : leeyoonhyuk0

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.

독창적인 아이디어와 추가 기능은 보너스 점수를 받을 수 있으므로, 보고서에 명확히 기재하십시오.

문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

아래 문항별 설명은 편의를 위한 것으로, 삭제하고 제출한다.

1. **프로그램 개요**

본 프로그램은 게임 2048을 그래픽 인터페이스 프레임워크인 Qt와 C++을 통해 구현한 것이다. 키보드의 화살표를 이용해 보드 위 블록들을 옮겨 충돌한 같은 숫자의 블록을 합칠 수 있고, 블럭을 합쳐 2048 블록을 만들면 승리, 2048블록을 만들기 전 보드가 가득 차 더 이상 블록을 움직일 수 없는 경우 패배하게 된다.

프로그램 디렉토리는 [표 1]과 같이 구성되어있다.

[표 ] 프로그램 디렉토리 설명

|  |  |
| --- | --- |
| 파일명 | 설명 |
| main.cpp | 메인 파일 |
| game/block.h | Block class의 헤더파일 |
| game/block.cpp | Block class의 소스파일 |
| game/board.h | Board class의 헤더파일 |
| game/board.cpp | Board class의 소스파일 |
| game/game.h | Game class의 헤더파일 |
| game/game.cpp | Game class의 소스파일 |
| game/logger.h | Logger class의 헤더파일. |
| game/logunit.h | LogUnit struct의 헤더파일 |
| ui/blockUi.h | BlockUi class의 헤더파일 |
| ui/blockUi.cpp | BlockUi class의 소스파일 |
| ui/gameui.h | GameUi class의 헤더파일 |
| ui/gameui.cpp | GameUi class의 소스파일 |
| ui/mainwindow.h | mainwindow class의 헤더파일 |
| ui/mainwindow.cpp | mainwindow class의 소스파일 |

1. **프로그램의 구조 및 알고리즘**

클래스들의 public method들을 중심으로 설명하고, class간 관계를 기술하겠다.

* 1. Block

2048 게임에서 블록을 담당하는 class이다. 블록 값을 가지고 있다.

|  |  |
| --- | --- |
| get\_value | 블록 값을 반환한다. |
| set\_value | 블록 값을 설정한다. |
| is\_mergeable | 블록이 Merge 가능한 상태인지를 반환한다. |

* 1. Board

block들이 움직이는 보드를 담당하는 class이다. 4x4보드의 블록들을 가지고 있으며, logger에게 log를 전달한다.

|  |  |
| --- | --- |
| move\_up() | 블록을 위로 움직인다. |
| move\_down() | 블록을 아래로 움직인다. |
| move\_left() | 블록을 왼쪽으로 움직인다. |
| move\_right() | 블록을 오른쪽으로 움직인다. |
| restore() | buffer의 block 배치로 restore 한다. |
| is\_movable() | 보드판의 블록들이 움직일 수 있는지 판별한다. |
| is\_2048\_exist | 2048블록이 있는지 확인한다. |
| get\_score | 점수를 반환한다. |
| get\_restore\_count | restore 가능 횟수를 반환한다. |
| get\_board | 보드판의 block들의 배열을 반환한다. |

* 1. Game

2048 게임의 내부 로직을 총괄하는 class이다. board와 logger를 contain하고 있어, board의 행동과 logger의 logging을 제어한다. GameUi와 signal을 주고 받는다.

|  |  |
| --- | --- |
| get\_score | 점수를 반환한다. |
| get\_restore\_count | restore 가능 횟수를 반환한다. |
| update\_board | board를 업데이트한다. |
| update\_score | 점수를 업데이트한다. |

* 1. Logger

게임의 progress를 logging하는 class이다. 플레이어의 입력으로 초래된 모든 행동을 logger queue로 받아, 정해진 포맷대로 정렬하여 파일에 출력하는 역할을 맡는다.

|  |  |
| --- | --- |
| write\_file | log파일에, log\_queue에 있는 모든 log를 string으로 변환하여 출력한다. |
| get\_queue | logging을 하는 object에게 log\_queue를 건내준다. log\_queue에 log를 입력하면 logger가 처리할 수 있다. |

* 1. LogUnit

logger가 사용한 log를 포맷화한 struct이다. logger는 LogUnit을 해석해 string log를 만든다.

|  |  |
| --- | --- |
| action | log의 game operation를 기록한 변수이다. |
| values | action의 argument를 담는 배열이다. |

* 1. BlockUi

GUI에 표현될 블록을 담당하는 class이다. QgridLayout에 담겨 색과 값이 설정된다.

|  |  |
| --- | --- |
| update\_value | block의 값을 업데이트하여, 값과 block의 색을 설정한다. |

* 1. GameUi

grid layout과 score board, restore button, exit button을 포함하고 있는 widget class이다. Mainwindow의 Central Widget으로 동작하여, GUI 기능의 전반을 담당한다. Game class와 시그널을 주고 받으며 사용자의 입력을 내부 로직으로 전달해 game을 진행시킨다.

* 1. Mainwindow

GameUi를 담는 Mainwindow Class이다. GameUi와 Game을 contain하고 있고, GameUi와 Game의 시그널을 connect해준다.. GUI의 main과 비슷한 역할을 하고 있다.

코드의 전체 흐름은 다음과 같다.

화살표 입력, 버튼 클릭등의 사용자 행동을 GameUI가 받아서 Game에게 game operation을 호출하고, Game은 진행된 게임 상황을 GameUi에게 전달하여 GUI를 업데이트 하는 방식이다.

이 표는 signal과 signal을 보내는 emitter, slot과 slot이 실행되는 receiver의 매핑표이다.

|  |  |  |  |
| --- | --- | --- | --- |
| emitter | receiver | signal | slot |
| Game | GameUi | board\_update | update\_board |
| Game | GameUi | score\_update | update\_score |
| Game | GameUi | restore\_response | get\_restore\_response |
| Game | GameUi | restore\_count\_response | get\_restore\_count\_response |
| Game | GameUi | win | win |
| Game | GameUi | lose | lose |
| GameUi | Game | request\_move\_up | move\_up |
| GameUi | Game | request\_move\_down | move\_dowm |
| GameUi | Game | request\_move\_left | move\_left |
| GameUi | Game | request\_move\_right | move\_right |
| GameUi | Game | request\_restore | restore |
| GameUi | Game | request\_restore\_count | send\_restore\_count |

1. **토론 및 개선**

이번 과제를 진행하면서 이벤트 기반 프로그래밍과 그래픽 인터페이스 구현을 직접 할 수 있었다. 나는 Python으로 Qt를 사용한 적이 있어 시스템이 굉장히 낯익었다. C++로 구현해보니 Qt의 클래스명과 메서드명이 Python과 c++ 모두 같고, 문법의 형식만 다르다는 것을 깨달을 수 있었다.

본 과제에서 win, lose를 출력하는 QMessagebox를 생성하는 함수에, string을 인자로 넣어주었는데, string에서 copy가 일어나므로 불필요하게 메모리를 사용하는 문제가 있다. 이는 가독성과 통일성을 위해 모든 Qmessagebox를 실행하는 함수들의 parameter로 string을 넣어주다가. 중간에 방식을 바꾸어 일어난 일이다.

1. **참고 문헌**

C++ reference : https://en.cppreference.com/w/

2023S object oriented programming slides

qt forum: <https://forum.qt.io/>

qt document: <https://doc.qt.io/>