QT 프로젝트 발표 :: Yacht Dice

로봇 게임단 휴머팀
19기 예비단원
이원준

목차

- 주제 선정 동기
- Yachtcombination.h 설명
- Mainwindow 구성
- 프로그램 시연

주제 선정 동기

- Yacht Dice는 게임 로직을 **객체지향적**으로 설계하기에 좋기 때문입니다.
- 구현 과정에서 Qt 같은 프레임워크를 활용한 UI가 필연적이고 중요하기 때문입니다.

Yachtcombination 헤더파일 구성

- Yachtcombination 헤더파일은 yacht dice 게임의 진행중 ui가 담당하는 요소를 제외한 모든 내용을 다루고 있습니다
- 크게 2개의 클라스로 구성되어 있습니다
- Player_hand 클라스에는 플레이어의 패를 관리하는 기능을, Yacht 클라스에는 족보의 조합을 바탕으로 점수를 리턴하는 기능을 구현했습니다.
- Yacht 클라스는 Player_hand 클라스를 상속을 받는 구조로 만들었습니다.
- 각 클라스의 멤버 변수를 캡슐화했습니다.

```
class Player_Hands
{
public:

    int* dice_origin;
    bool* isfixed;
    bool* is_keeped;

    int* dice_arranged; // upper arranged

    Player_Hands()
    {
        dice_origin = new int[5];
        dice_arranged = new int[5];
        isfixed = new bool[13];
        is_keeped = new bool[5];

        for(int i =0; i<13 ; i++)
        {
            isfixed[i] = {false};
        }
        for(int i = 0; i < 5; i++)
        {
            is_keeped[i] = {false};
        }

        init_origin();
        init_arranged();
    };

    ~Player_Hands()
    {
        delete[] isfixed;
        delete[] dice_origin;
        delete[] is_keeped;
        delete[] dice_arranged;
    };

    void init_origin();
    void init_arranged();

    int dice_sum() const;

    void Reroll();
};
```

Mainwindow.h의 구성

- 헤더 파일에서 Public으로 멤버 함수를 선언하고 cpp 소스파일에서 재정의하는 방법으로 UI 및 게임진행을 제어할 수 있는 함수를 구현했습니다.
- 두 명의 플레이어가 경쟁하는 방식의 게임이기에 중복되는 요소를 함수 모듈화 해서 진행하였습니다.

```
class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

    void init_checkbox();

    void indicate_box();

    void over_roll();

    void p1_scoreboard_init();

    void p2_scoreboard_init();

    void p1_scoreboard_manage(bool do_activate);

    void p2_scoreboard_manage(bool do_activate);

    void init_dice_image(bool is_p1);

    void end_game();

private slots:
    void on_p1ones_clicked();
```

Mainwindow.cpp 설명

- Mainwindow.cpp 파일은 Qt에서 제공하는 라이브러리 클래스를 활용해서 코드를 작성했습니다.

- 우측 코드는 Mainwindow.cpp 파일의 코드 중 일부를 가져온 사진으로 리소스파일의 이미지 파일의 경로를 Qstring 배열로 만들어 주사위이미지를 주사위를 굴릴때 마다 초기화 할 수 있도록 구현한 코드입니다.

```
QString diceImagePaths[6] = {
    ":/Downloads/주사위 png 사본/6면 주사위 1.png",
    ":/Downloads/주사위 png 사본/6면 주사위 2.png",
    ":/Downloads/주사위 png 사본/6면 주사위 3.png",
    ":/Downloads/주사위 png 사본/6면 주사위 4.png",
    ":/Downloads/주사위 png 사본/6면 주사위 5.png",
    ":/Downloads/주사위 png 사본/6면 주사위 6.png"};

void MainWindow::init_dice_image(bool is_p1)
{
    QLabel* diceLabels[5] = {ui->dice1, ui->dice2, ui->dice3, ui->dice4, ui->dice5};

    if(count_turn%2)
    {
        p1.Reroll();

        for (int i = 0; i < 5; ++i) //카드 이미지 초기화
        {
            diceLabels[i]->setPixmap(QPixmap(diceImagePaths[p1.dice_origin[i] - 1]));
            diceLabels[i]->setScaledContents(true);  // 이미지 크기 자동 조정
        }
    }
    else
    {
        p2.Reroll();

        for (int i = 0; i < 5; ++i) //카드 이미지 초기화
        {
            diceLabels[i]->setPixmap(QPixmap(diceImagePaths[p2.dice_origin[i] - 1]));
            diceLabels[i]->setScaledContents(true);
        }
    }
}
```

프로그램 시연