C++&Qt day3 hw1 결과 보고서

2025407012/로봇학부/송연우

목차

1. hpp파일

2. cpp파일

3. 실행 결과

1. hpp 파일

다음은 작성한 헤더파일 코드입니다.

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QTimer>
#include <QElapsedTimer>
#include <fstream>
#include <iostream>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();
    void Put_text_3(int timernum, const char *num1, const char *num2, const char *num3);
    void Put_text_4(int timernum, const char *num1, const char *num2, const char *num3, const char *num4);
```

클래스의 Public에서 슬롯 함수를 제외하고 사용할 함수를 선언하는 부분입니다. 생성자와 소멸자 밑에 있는 Put_text_3, 4함수는 채팅 박스에 출력할 문자와 타이머 번호를 입력받아 사용자 입력에 따라 자판 위의 채팅 박스에 문자들을 출력합니다.

함수의 작동 방식은 1초를 기준으로 삼아 버튼이 한 번 눌린 후 1초 전에 버튼이 다시 눌리면 전에 작성했던 글자를 지우고, 카운트를 셉니다. 1초가 지나도 버튼이 눌리지 않으면(1초가 흐른 뒤에 다시 버튼이 눌리면) 카운트를 초기화합니다. 버튼이 눌리기만 하면 무조건적으로 카운트 수에 맞는 문자가 출력됩니다.

```
24      private slots:
25          void on_pushButton_clicked();
26
27          void on_pushButton_2_clicked();
28
29          void on_pushButton_8_clicked();
30
31          void on_pushButton_7_clicked();
```

버튼과 관련한 슬롯입니다. 한 버튼당 clicked/pressed/released 슬롯이 존재하며 clicked 는 각 버

튼의 기능을, pressed/released는 버튼의 색상을 변경합니다.

```
89
90      private:
91          Ui::MainWindow *ui;
92          int count[10];
93          QElapsedTimer *timer[10];
94          bool isLargeMode;
95          bool isclicked;
96      };
97      #endif // MAINWINDOW_H
```

나머지 private 변수를 선언하는 부분입니다. 버튼이 눌린 횟수를 카운트할 배열, 시간 경과를 측정하기 위한 timer 배열, 대/소문자 구현을 위한 bool 변수가 있습니다.

2. cpp 파일

다음은 작성한 소스코드입니다.

```cpp
#include "mainwindow.h"
#include "./ui_mainwindow.h"

using namespace std;

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
    , isclicked(false)
{
    ui->setupUi(this);
    ui->textEdit->setReadOnly(true);
    for(int i=0;i<10;i++){
        timer[i]=new QElapsedTimer;
        count[i]=-1;
    }
    isLargeMode=false;
    if(isLargeMode){
        ui->pushButton_2->setText("ABC");
        ui->pushButton_8->setText("DEF");
        ui->pushButton_7->setText("GHI");
        ui->pushButton_6->setText("JKL");
        ui->pushButton_3->setText("MNO");
        ui->pushButton_4->setText("PQRS");
        ui->pushButton_5->setText("TUV");
        ui->pushButton_9->setText("WXZY");
    }
    else{
        ui->pushButton_2->setText("abc");
        ui->pushButton_8->setText("def");
        ui->pushButton_7->setText("ghi");
        ui->pushButton_6->setText("jkl");
        ui->pushButton_3->setText("mno");
        ui->pushButton_4->setText("pqrs");
        ui->pushButton_5->setText("tuv");
```

생성자 부분 코드입니다. ui->textEdit->setReadOnly함수로 사용자가 임의의 키보드 입력으로 채팅 박스에 문자를 입력하지 못하도록 합니다. 그리고 timer을 QElapsedTimer형으로 동적 할당하고, count를 초기화합니다. 처음 표시되는 자판이 소문자가 되도록 isLargeMode는 false로 설정합니다. 그리고 setText()함수를 사용해 각 버튼에 문자가 나타나도록 합니다.

```
40  MainWindow::~MainWindow()
41  {
42      delete ui;
43
44  }
45
46  void MainWindow::Put_text_3(int timernum, const char *num1, const char *num2, const char *num3){
47      //버튼 다시눌림+1초안지남->전 글자 삭제
48       if(timer[timernum]->isValid() && timer[timernum]->elapsed()<1000){
49          QTextCursor cursor = ui->textEdit->textCursor();
50          cursor.movePosition(QTextCursor::Left, QTextCursor::KeepAnchor, 1);
51          cursor.removeSelectedText();
52          ui->textEdit->setTextCursor(cursor);
53          // 글자 순환
54          count[timernum]++;
55       }
56      if(count[timernum]<0)count[timernum]=0;
57      switch(count[timernum]%3){
58          case 0: ui->textEdit->insertPlainText(num1); break;
59          case 1: ui->textEdit->insertPlainText(num2); break;
60          case 2: ui->textEdit->insertPlainText(num3); break;
61      }
62       // 버튼 눌림->무조건 타이머 초기화
63       timer[timernum]->start();
64  }
```

Put_text_3함수입니다. 만약 타이머가 유효하고 elapsed()함수를 통해 구한 버튼이 한 번 눌린 뒤 경과 시간이 1초가 넘지 않으면 채팅 박스에 남아 있는 최근 글자를 삭제합니다. 삭제를 위해 QTextcursor형 cursor변수를 선언하고 movePosition()함수를 통해 왼쪽으로 1번 이동한 값을 선택해 removeSelectedText()로 삭제합니다. 이후에 글자 순환을 위해 count를 세 줍니다.

다음으로, 혹시 카운트값이 음수가 되었을 때를 대비해서 0으로 초기화 해준 후 count를 3으로 나눈 나머지값으로 출력할 문자를 결정합니다. 끝으로 다음 클릭까지의 시간 측정을 위해 타이머를 초기화해 시작합니다.

```
87  void MainWindow::on_pushButton_clicked()
88  {
89      //버튼1
90      Put_text_4(0, ".", ",", "?", "!");
91
92  }
93
94  void MainWindow::on_pushButton_2_clicked()
95  {
96      //버튼2
97      if(isLargeMode)Put_text_3(1, "A", "B", "C");
98      else Put_text_3(1, "a", "b", "c");
99  }
```

버튼이 클릭되었을 때의 슬롯 함수입니다. 위의 Put_text_3함수의 switch값을 수정한 Put_text_4함수를 사용해 순환하는 문자가 4개일 때의 버튼을 설정합니다.

아래의 버튼 2 함수처럼 대/소문자가 존재하는 경우에는 bool 변수에 따라 출력하는 문자를 다르게 합니다. 이와 같이 9개 버튼을 설정합니다.

```
164  void MainWindow::on_pushButton_14_clicked()
165  {
166      //backspace
167      QTextCursor cursor = ui->textEdit->textCursor();
168      cursor.movePosition(QTextCursor::Left, QTextCursor::KeepAnchor, 1);
169      cursor.removeSelectedText();
170      ui->textEdit->setTextCursor(cursor);
171  }
172
173
174  void MainWindow::on_pushButton_15_clicked()
175  {
176      //enter
177      QString currentText = ui->textEdit->toPlainText();
178      ofstream save;
179
180          save.open("textbox.txt", std::ios::app); //더해서 쓰기 모드
181
182         save<<currentText.toStdString()<<"\n";
183         save<<endl;
184         save.close();
185      ui->textEdit->clear();
186  }
```

다음으로 문자 출력외의 다른 기능을 가지는 버튼의 슬롯 함수입니다. 위의 backspace 버튼 함수는 이전 글자를 지우는 기능을 가지고, endter함수는 ofstring 라이브러리를 통해 textbox.txt파일을 덮어씌우기 모드가 아닌 더해서 쓰기 모드로 오픈 후 현재 채팅 박스에 있는 문자열을 모두 읽어 저장합니다. 이후 파일을 닫고 채팅 박스의 문자를 모두 지웁니다.

```
189  void MainWindow::on_pushButton_16_clicked()
190  {
191      //shift
192      if(!isLargeMode){
193          ui->pushButton_2->setText("ABC");
194          ui->pushButton_8->setText("DEF");
195          ui->pushButton_7->setText("GHI");
196          ui->pushButton_6->setText("JKL");
197          ui->pushButton_3->setText("MNO");
198          ui->pushButton_4->setText("PQRS");
199          ui->pushButton_5->setText("TUV");
200          ui->pushButton_9->setText("WXZY");
201      }
202      else{
203          ui->pushButton_2->setText("abc");
204          ui->pushButton_8->setText("def");
205          ui->pushButton_7->setText("ghi");
206          ui->pushButton_6->setText("jkl");
207          ui->pushButton_3->setText("mno");
208          ui->pushButton_4->setText("pqrs");
209          ui->pushButton_5->setText("tuv");
210          ui->pushButton_9->setText("wxzy");
211      }
212      isLargeMode=!isLargeMode;
213  }
```

Shift 버튼에 대한 슬롯 함수입니다. 문자를 표시하고 대/소문자 키를 바꿉니다.

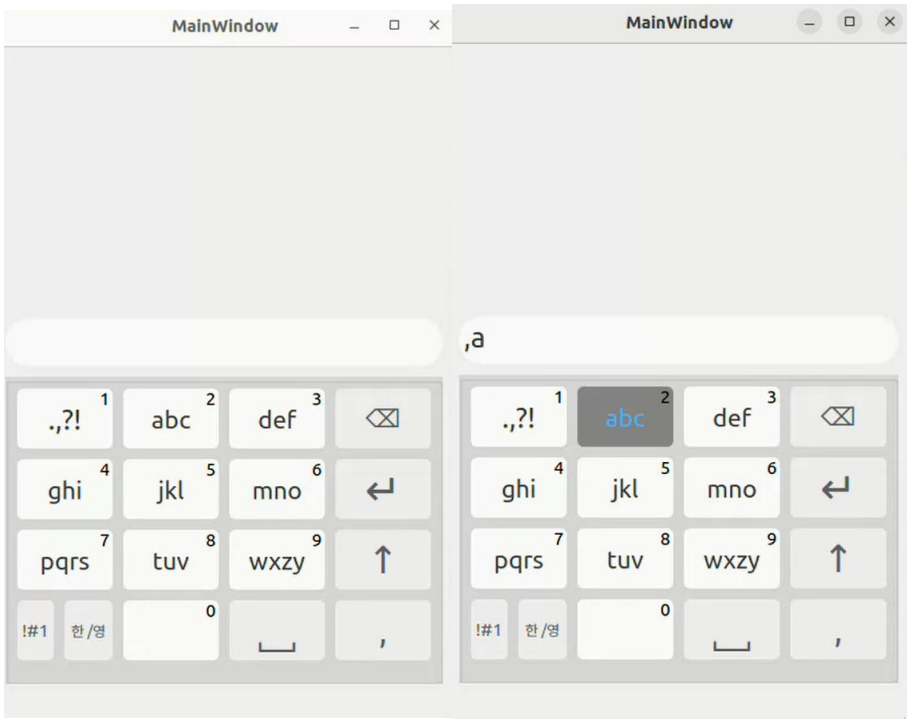
```
214  void MainWindow::on_pushButton_16_released()
215  {
216      ui->pushButton_16->setStyleSheet("background-color: rgb(237, 237, 237);color:rgb(93, 93, 93);border-radius: 5px;font: 24pt 'Ubuntu';");
217  }
218
219  void MainWindow::on_pushButton_16_pressed()
220  {
221      ui->pushButton_16->setStyleSheet("color: rgb(77, 175, 255); background-color: rgb(133, 131, 131); border-radius: 5px; font: 24pt 'Ubuntu'
222  }
223
```

버튼의 색상 변경 기능을 가지는 슬롯 함수입니다. 버튼이 눌려 있는 동안에는 짙은 회색 배경에 하늘색 글자색을 가지도록 설정하고, 버튼이 눌려 있지 않으면 밝은 회색에 검정색 글자색을 가지도록 설정합니다.

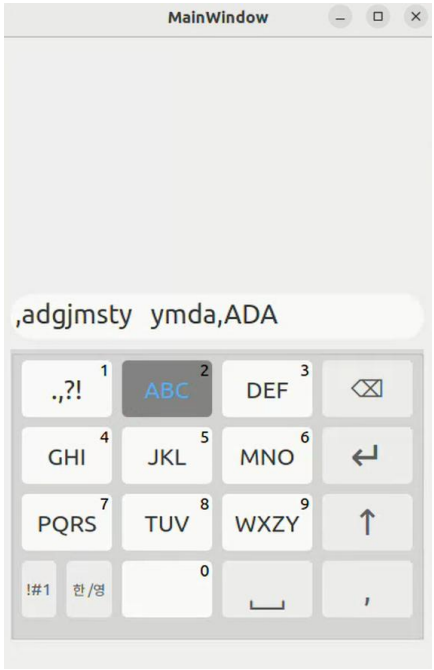
```
363  void MainWindow::on_pushButton_17_clicked()
364  {
365      //,
366      ui->textEdit->insertPlainText(",");
367  }
368
369
370  void MainWindow::on_pushButton_18_clicked()
371  {
372      //space
373      ui->textEdit->insertPlainText(" ");
374  }
```

,버튼과 space 버튼입니다. 각각 맞는 문자를 출력합니다.

3. 실행 결과



처음 실행했을 때의 화면입니다. 버튼을 눌렀을 때 버튼 색이 바뀌고 문자가 채팅 박스에 출력됩니다.



대문자 출력 화면입니다.