Министерсво образоваия Республики Беларусь

Учреждение образования

«Бресткий государтвенный технический университет»

Кафедра ИИТ

Лабораторная работа №2

за 5 семестр

По дисциплине «ООТПиСП»

Тема: «Текстовый редактор»

Выполнил:

Студент 3 курса

Группы ПО-4(2)

Кречко К.А.

Проверила:

Хацкевич М.В.

Брест 2021

**Лабораторная работа №2**

**Текстовый редактор**

**Код программы:**

**texteditor.h**

#ifndef TEXTEDITOR\_H

#define TEXTEDITOR\_H

#include <QMainWindow>

#include <QTextEdit>

#include <QString>

#include <QMessageBox>

#include <QCloseEvent>

#include <QWidget>

#include <QApplication>

#include <QSettings>

#include <QFontDialog>

#include <qstring.h>

#include <QFile>

#include <QTextStream>

#include <QFileInfo>

#include <QFileDialog>

#include <QTextEdit>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **TextEditor**; }

QT\_END\_NAMESPACE

class **TextEditor** : public QMainWindow

{

Q\_OBJECT

public:

**TextEditor**(const QString &fileName = QString(), QWidget \*parent=nullptr);

~***TextEditor***();

protected:

void ***closeEvent***(QCloseEvent \*e);

private slots:

void **on\_actionNew\_triggered**();

void **on\_actionClose\_triggered**();

void **on\_actionExit\_triggered**();

void **documentModified**();

void **on\_actionSelectFont\_triggered**();

void **on\_actionAboutQt\_triggered**();

void **on\_actionAbout\_triggered**();

void **on\_actionCut\_triggered**();

void **on\_actionCopy\_triggered**();

void **on\_actionPaste\_triggered**();

void **on\_actionUndo\_triggered**();

void **on\_actionRedo\_triggered**();

void **on\_actionOpen\_triggered**();

bool **saveFile**();

bool **saveFileAs**();

private:

void **loadFile**(const QString &fileName);

void **setFileName**(const QString &fileName);

Ui::TextEditor \*ui;

QString m\_fileName;

};

#endif // TEXTEDITOR\_H

**texteditor.cpp**

#include "texteditor.h"

#include "ui\_texteditor.h"

TextEditor::**TextEditor**(const QString &fileName, QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::TextEditor)

{

ui->setupUi(this);

setWindowTitle(QString("TextEditor[\*]"));

connect(ui->actionClose, SIGNAL(triggered()), this, SLOT(close(true)));

connect(ui->actionExit, SIGNAL(triggered()), qApp, SLOT(closeAllWindows()));

connect(ui->textEdit, SIGNAL(textChanged()), this, SLOT(documentModified()));

connect(ui->actionAboutQt, SIGNAL(triggered()), qApp, SLOT(aboutQt()));

connect(ui->actionAbout, SIGNAL(triggered()), qApp, SLOT(about()));

connect(ui->actionCut, SIGNAL(triggered()), this, SLOT(cut()));

connect(ui->actionCopy, SIGNAL(triggered()), this, SLOT(copy()));

connect(ui->actionPaste, SIGNAL(triggered()), this, SLOT(paste()));

connect(ui->actionUndo, SIGNAL(triggered()), this, SLOT(undo()));

connect(ui->actionRedo, SIGNAL(triggered()), this, SLOT(redo()));

connect(ui->textEdit, SIGNAL(cutAvailable(bool)), ui->actionCut, SLOT(setEnabled(false)));

connect(ui->textEdit, SIGNAL(copyAvailable(bool)), ui->actionCopy, SLOT(setEnabled(false)));

connect(ui->textEdit, SIGNAL(undoAvailable(bool)), ui->actionUndo, SLOT(setEnabled(false)));

connect(ui->textEdit, SIGNAL(redoAvailable(bool)), ui->actionRedo, SLOT(setEnabled(false)));

connect(ui->actionOpen, SIGNAL(triggered()), this, SLOT(open()));

connect(ui->actionSave, SIGNAL(triggered()), this, SLOT(saveFile()));

connect(ui->actionSaveAs, SIGNAL(triggered()), this, SLOT(saveFileAs()));

QFont Font = QSettings(this).value("VIEWF").value<QFont>();

ui->textEdit->setFont(Font);

loadFile(fileName);

}

TextEditor::~***TextEditor***()

{

delete ui;

}

void TextEditor::***closeEvent***(QCloseEvent \*e)

{

if (isWindowModified())

{

switch (QMessageBox::warning(this, "Document Modified",

"The document has been modified. "

"Do you want to save your changes?\n"

"You will lose and unsaved changes.",

QMessageBox::Yes | QMessageBox::No | QMessageBox::Cancel,

QMessageBox::Cancel))

{

case QMessageBox::Yes:

saveFile();

e->ignore();

break;

case QMessageBox::No:

e->accept();

break;

case QMessageBox::Cancel:

e->ignore();

break;

}

}

else

e->accept();

}

void TextEditor::**on\_actionNew\_triggered**()

{

TextEditor \*obj1 = new TextEditor();

obj1->show();

}

void TextEditor::**on\_actionClose\_triggered**()

{

close();

}

void TextEditor::**on\_actionExit\_triggered**()

{

qApp->closeAllWindows();

}

void TextEditor::**documentModified**()

{

setWindowModified(true);

}

void TextEditor::**on\_actionCut\_triggered**()

{

ui->textEdit->cut();

}

void TextEditor::**on\_actionAboutQt\_triggered**()

{

qApp->aboutQt();

}

void TextEditor::**on\_actionAbout\_triggered**()

{

QMessageBox::about(this, "Это программка", "Текстовый редактор");

}

void TextEditor::**on\_actionCopy\_triggered**()

{

ui->textEdit->copy();

}

void TextEditor::**on\_actionPaste\_triggered**()

{

ui->textEdit->paste();

}

void TextEditor::**on\_actionUndo\_triggered**()

{

ui->textEdit->undo();

}

void TextEditor::**on\_actionRedo\_triggered**()

{

ui->textEdit->redo();

}

void TextEditor::**loadFile**(const QString &fileName)

{

if(fileName.isEmpty()){

setFileName(QString());

return;

}

QFile file(fileName);

if(!file.*open*(QIODevice::ReadOnly | QIODevice::Text)){ //Создает новый QFileInfo, который дает информацию о данном файле

QMessageBox::warning(this,"Error", "Can't open file"); //Данное свойство хранит имя приложения.

setFileName(QString());

return;

}

QTextStream textStream(*&file*);

ui->textEdit->setText(textStream.readAll());

file.*close*();

setFileName(fileName);

setWindowModified(false);

}

void TextEditor::**setFileName**(const QString &fileName)

{

m\_fileName = fileName;

setWindowTitle(QString("%1[\*] - %2")

.arg(m\_fileName.isNull()?"intitled":QFileInfo(m\_fileName).fileName())

.arg(QApplication::applicationName()));

}

void TextEditor::**on\_actionSelectFont\_triggered**()

{

bool Changed;

QFont newFont = QFontDialog::getFont(*&Changed*);

if (Changed) {

ui->textEdit->setFont(newFont);

QSettings settings(this);

settings.setValue("VIEWF", newFont);

}

}

void TextEditor::**on\_actionOpen\_triggered**()

{

QString fileName = QFileDialog::getOpenFileName(this,

"Open document", QDir::currentPath(), "Text documents (\*.txt)");

if (fileName.isNull())

return;

if (m\_fileName.isNull() && !isWindowModified())

loadFile(fileName);

else{

TextEditor \*p =new TextEditor(fileName);

p->show();

}

}

bool TextEditor::**saveFileAs**()

{

QString fileName = QFileDialog::getSaveFileName(this, "Save document",

m\_fileName.isNull()?QDir::currentPath():m\_fileName, "Text documents (\*.txt)");

if (fileName.isNull()) return false;

setFileName(fileName);

return saveFile();

}

bool TextEditor::**saveFile**()

{

if(m\_fileName.isNull())

return saveFileAs();

QFile file(m\_fileName);

if(!file.*open*(QIODevice::WriteOnly | QIODevice::Text)){

QMessageBox::warning(this,"Error", "Can't open file");

setFileName(m\_fileName);

return false;

}

QTextStream textstream(*&file*);

textstream << ui->textEdit->toPlainText();

file.*close*();

setWindowModified(false);

return true;

}

**main.cpp**

#include "texteditor.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(*argc*, *argv*);

a.setApplicationName("Text Editor");

a.setApplicationVersion("0.1");

a.setOrganizationName("ExampleSoft");

a.setOrganizationDomain("example.com");

a.setWindowIcon(QIcon(":/icons/new.png"));

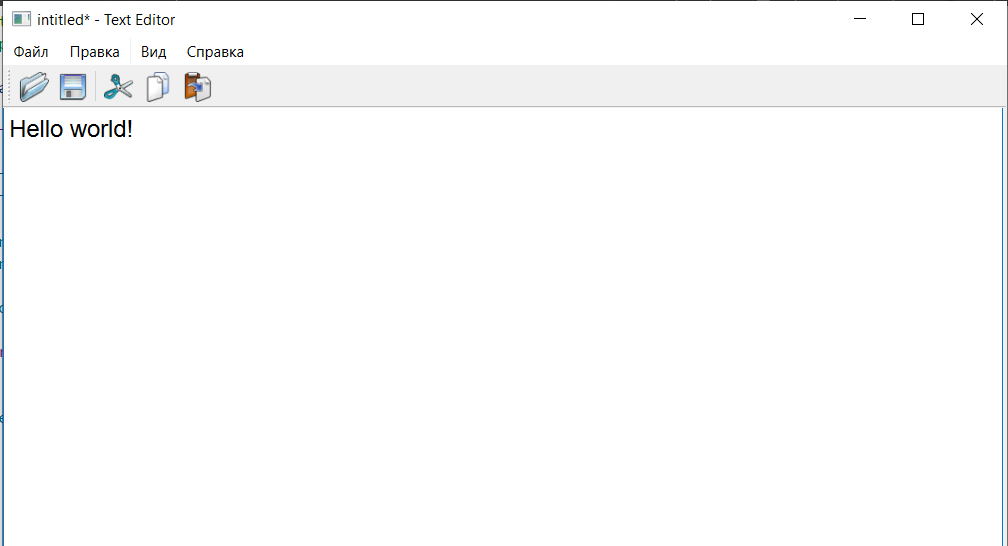
TextEditor w;

w.show();

return a.exec();

}

**Результат:**

****