**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **ІКНІ**

Кафедра **ПЗ**



### ЗВІТ

До лабораторної роботи №1-4

**На тему:** *“Компоненти Qt Creator для представлення даних”*

**З дисципліни:** *“Об’єктно-орієнтоване програмування”*

**Лектор:**

доцент каф.ПЗ

Коротєєва Т. О.

**Виконав:**

ст. гр. ПЗ-16

Шеремета А.І.

**Прийняв:**

асист. каф. ПЗ

Дивак І.В.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_ .

Львів – 2022

**Тема роботи:** Ознайомлення із середовищем розробки Qt Creator. Створення проекту та налаштування його властивостей.

**Мета** **роботи:** Засвоїти принцип візуального програмування шляхом створення та налаштування проекту.

**Теоретичні відомості**

Інтерфейс Qt Creator називають середовищем швидкої розробки застосувань RAD (Rapid Application Development) або середовищем візуальної розробки. Таку назву цей інтерфейс отримав за те, що створення застосування в ньому зводиться в основному до простого конструювання вікна майбутнього застосування із набору готових компонент, а більшу частину стандартних операцій виконує комп’ютер.

* 1. Огляд cередовища розробки Qt Creator. Середовище Qt Creator умовно розділяють на наступні частини: вікно інспектора об’єктів (Object Inspector), вікно дерева об’єктів (Object TreeView), вікно класів, палітра інструментів, вікно редактору коду , дизайнер форми.

Палітра інструментів містить закладки із набором компонент. Виділяють візуальні (видимі при запуску застосування) та невізуальні компоненти. Окрім стандартних компонент є можливість імпортувати компоненти сторонніх розробників, або створити власну компоненту.

Інспектор об’єктів містить в собі закладку властивостей (Properties) та подій (Events). Призначення інспектора об’єктів – редагування властивостей об’єктів (форм та компонент) під час розробки застосування. З його допомогою можна вказати початкове значення об’єктів та присвоїти обробники подій. Властивості та події, що дозволяють зв’язати інші компоненти із поточною, в інспекторі об’єктів виділено червоним кольором.

У верхній частині вікна інспектора розміщено поле вибору об’єктів, в якому відображається назва вибраного об’єкта. Для вибору нового об’єкта потрібно відкрити випадаючий список, вибравши кнопку із стрілкою, а потім необхідний об’єкт із списку. Після цього можна редагувати властивості вибраного об’єкта. Інший спосіб вибору об’єкта – клік мишкою на об’єкті у формі. Третій спосіб полягає у виборі потрібного об’єкту за назвою у вікні дерева об’єктів (Object TreeView).

1.2. Організація проекту в Qt Creator. Проектом називається група файлів, що необхідні для створення кінцевого виконуючого застосування. Розглянемо типи файлів що входять до звичайного проекту із користувацькими вікнами на C++ Builder. Для раціонального використання місця на диску, можна видаляти файли резервних копій, об’єктні файли та файл відлагодження, оскільки компілятор створить їх за необхідності ще раз.

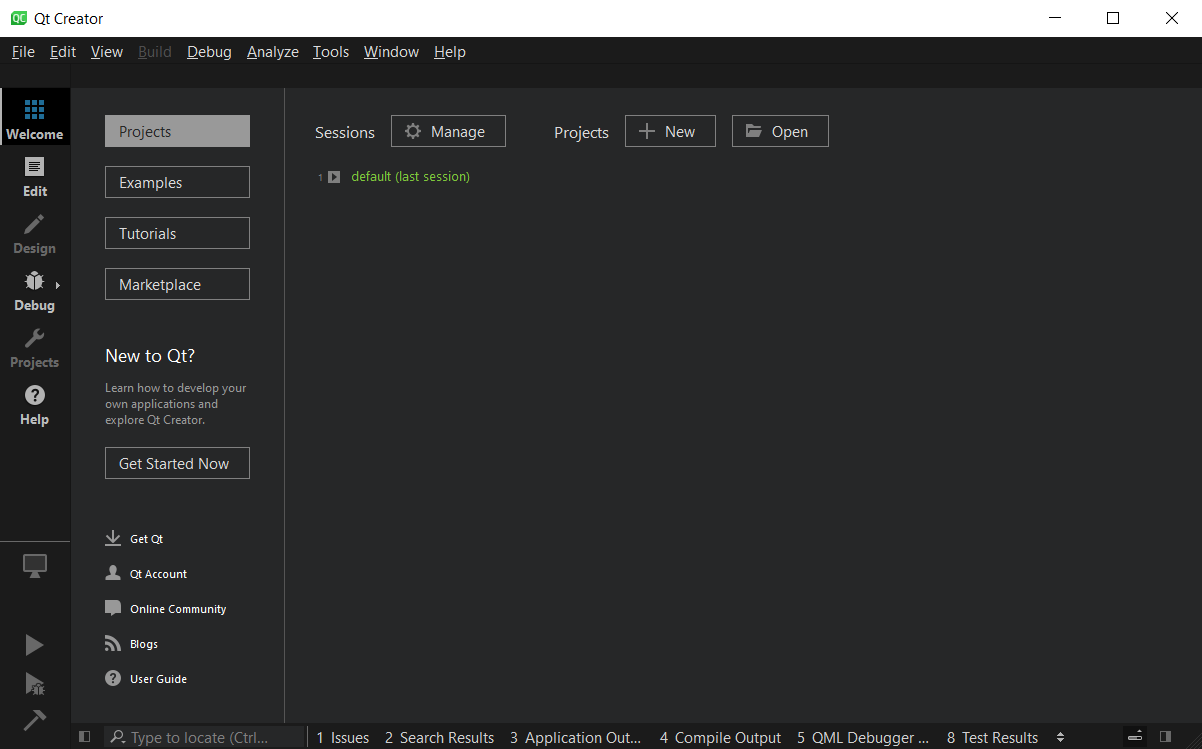
**Лабораторне завдання**

1. Ознайомитись із середовищем Qt Creator.
2. Створити новий проект. Зберегти його двома способами – через  
   комбінації швидких клавіш та через меню.
3. Проглянути у вікні інспектора об’єктів властивості форми. Змінити назву форми та її розміри.
4. Запустити на виконання застосування.
5. Відкрити опції проекту, змінити  
   налаштування на закладках Application, Compiler, Packages. Запустити на виконання застосування.

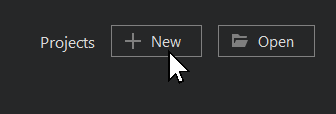
**Виконання лаборатоної роботи**

Для виконання лабораторної роботи було використано середовище розробки Qt Creator на системі Windows 10 Home (версія 21H2)

Відкривши програму Qt Creator ви побачите головне меню редактора.



Щоб створити новий проект у головному меню вибираємо ‘Projects -> New’

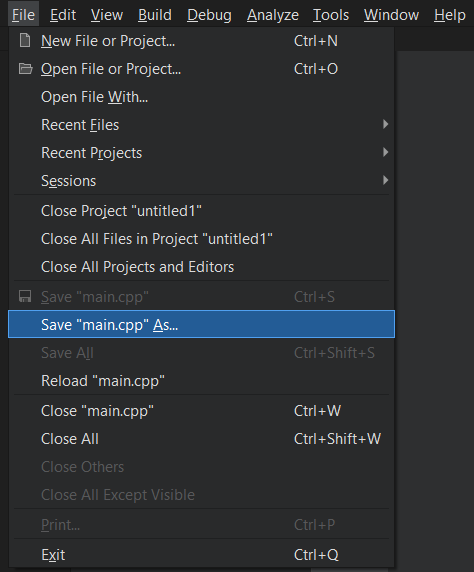


Зберегти проект можна двома способами: за допомогою комбінацій швидких клавіш та через меню.

Вибравши потрібний шаблон проекту ми можемо його зберегти.

Щоб зберегти проект натискаємо на клавіатурі комбінацію Ctrl+Shift+S.

Щоб зберегти проект за допомогою меню в панелі меню обираємо File -> Save “…” As та обираємо шлях для збереження.



**Висновки**

На даній лабораторній роботі створив віконний проект та продемонстрував використання компонент призначених для відображення та опрацювання даних.

**Тема роботи:** Базові візуальні компоненти Qt Creator.

**Мета роботи:** Створити віконний проект та продемонструвати використання візуальних компонент Qt Creator.

**Теоретичні відомості**

Клас QMainWindow забезпечує головне вікно програми.

Клас QApplication керує потоком керування та основними налаштуваннями програми GUI.

Клас QAction надає абстракцію для команд користувача, які можна додавати до різних компонентів інтерфейсу користувача.

Клас QMessageBox надає модальне діалогове вікно для інформування користувача або для того, щоб поставити йому запитання та отримати відповідь.

Клас QTextEdit надає віджет, який використовується для редагування та відображення як простого, так і форматованого тексту.

Віджет QLabel забезпечує відображення тексту або зображення.

**Лабораторне завдання**

1. Ознайомитись із палітрою компонент Qt Creator.
2. Створити віконний проект, додати розглянуті візуальні компоненти.
3. Реалізувати калькулятор.
4. Зробити висновок.

**Хід роботи**

1. Із палітрою компонент Qt Creator ознайомлений.
2. Створюю віконний проект.
3. Додаю потрібні компоненти (кнопки, текстове поле)

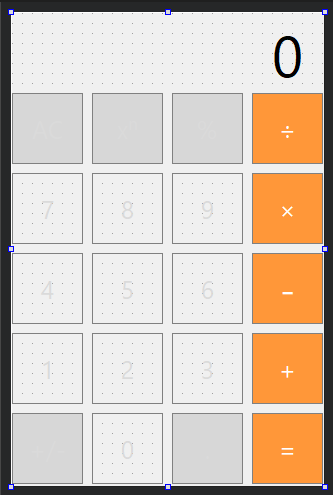


Рис. 1.Кнопки та вікно калькулятора

1. Реалізовуємо калькулятор, додаємо необхідний код:

MainWindow

connect(ui->pushButton\_0, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_1, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_2, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_3, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_4, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_5, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_6, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_7, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_8, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_9, SIGNAL(clicked()), *this*, SLOT(digits\_numbers()));

connect(ui->pushButton\_pm, SIGNAL(clicked()), *this*, SLOT(operations()));

connect(ui->pushButton\_percent, SIGNAL(clicked()), *this*, SLOT(operations()));

connect(ui->pushButton\_divide, SIGNAL(clicked()), *this*, SLOT(math\_operations()));

connect(ui->pushButton\_minus, SIGNAL(clicked()), *this*, SLOT(math\_operations()));

connect(ui->pushButton\_plus, SIGNAL(clicked()), *this*, SLOT(math\_operations()));

connect(ui->pushButton\_multiply, SIGNAL(clicked()), *this*, SLOT(math\_operations()));

connect(ui->pushButton\_square, SIGNAL(clicked()), *this*, SLOT(math\_operations()));

ui->pushButton\_divide->setCheckable(*true*);

ui->pushButton\_minus->setCheckable(*true*);

ui->pushButton\_plus->setCheckable(*true*);

ui->pushButton\_multiply->setCheckable(*true*);

ui->pushButton\_square->setCheckable(*true*)

void MainWindow::digits\_numbers()

{

QPushButton \*button = (QPushButton \*)sender();

double all\_numbers;

QString new\_label;

*if*(ui->result\_show->text().contains(".") && button->text() == "0"){

new\_label = ui->result\_show->text() + button->text();

}*else* {

all\_numbers = (ui->result\_show->text() + button->text()).toDouble();

new\_label = QString::number(all\_numbers, 'g', 15);

}

ui->result\_show->setText(new\_label);

}

void MainWindow::on\_pushButton\_dot\_clicked()

{

*if*(!(ui->result\_show->text().contains('.')))

ui->result\_show->setText(ui->result\_show->text() + ".");

}

void MainWindow::operations()

{

QPushButton \*button = (QPushButton \*)sender();

double all\_numbers;

QString new\_label;

*if*(button->text() == "+/-"){

all\_numbers = (ui->result\_show->text()).toDouble();

all\_numbers = all\_numbers \* -1;

new\_label = QString::number(all\_numbers, 'g', 15);

ui->result\_show->setText(new\_label);

} *else* *if*(button->text() == "%"){

all\_numbers = (ui->result\_show->text()).toDouble();

all\_numbers = all\_numbers \* 0.01;

new\_label = QString::number(all\_numbers, 'g', 15);

ui->result\_show->setText(new\_label);

}

}

void MainWindow::on\_pushButton\_AC\_clicked()

{

ui->pushButton\_plus->setChecked(*false*);

ui->pushButton\_minus->setChecked(*false*);

ui->pushButton\_divide->setChecked(*false*);

ui->pushButton\_multiply->setChecked(*false*);

ui->pushButton\_square->setChecked(*false*);

ui->result\_show->setText("0");

}

void MainWindow::math\_operations()

{

QPushButton \*button = (QPushButton \*)sender();

num\_first = ui->result\_show->text().toDouble();

ui->result\_show->setText("");

button->setChecked(*true*);

}

void MainWindow::on\_pushButton\_equal\_clicked()

{

double labelNumber, num\_second;

QString new\_label;

num\_second = ui->result\_show->text().toDouble();

*if*(ui->pushButton\_plus->isChecked()){

labelNumber = num\_first + num\_second;

new\_label = QString::number(labelNumber, 'g', 15);

ui->result\_show->setText(new\_label);

ui->pushButton\_plus->setChecked(*false*);

} *else* *if*(ui->pushButton\_minus->isChecked()){

labelNumber = num\_first - num\_second;

new\_label = QString::number(labelNumber, 'g', 15);

ui->result\_show->setText(new\_label);

ui->pushButton\_minus->setChecked(*false*);

}*else* *if*(ui->pushButton\_divide->isChecked()){

*if*(num\_second==0){

ui->result\_show->setText("0");

}*else*{

labelNumber = num\_first / num\_second;

new\_label = QString::number(labelNumber, 'g', 15);

ui->result\_show->setText(new\_label);

ui->pushButton\_divide->setChecked(*false*);

}

}*else* *if*(ui->pushButton\_multiply->isChecked()){

labelNumber = num\_first \* num\_second;

new\_label = QString::number(labelNumber, 'g', 15);

ui->result\_show->setText(new\_label);

ui->pushButton\_multiply->setChecked(*false*);

}*else* *if*(ui->pushButton\_square->isChecked()){

num\_first = pow(num\_first, num\_second);

labelNumber = num\_first;

new\_label = QString::number(labelNumber, 'g', 15);

ui->result\_show->setText(new\_label);

ui->pushButton\_square->setChecked(*false*);

}

**Виконання лаборатоної роботи**

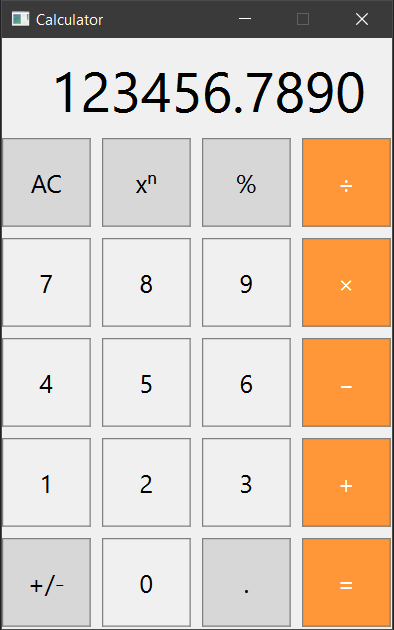


Рис. 2. Робота калькулятора

**Висновки**

На даній лабораторній роботі я створив проект з віджетами та навчився працювати з візуальними компонентами, створив користувацький інтерфейс і реалізував калькулятор.

**Тема роботи:** Огляд невізуальних компонент Qt Creator. Створення проекту із невізуальними компонентами та їх використання.

**Мета роботи:** Створити віконний проект та продемонструвати використання візуальних компонент Qt Creator.

**Теоретичні відомості**

Клас QApplication керує потоком керування та основними налаштуваннями програми GUI.

Клас QAction надає абстракцію для команд користувача, які можна додавати до різних компонентів інтерфейсу користувача.

Клас QIODevice є базовим класом інтерфейсу всіх пристроїв вводу/виводу в Qt.

Клас QToolBar забезпечує рухому панель, яка містить набір елементів керування.

Клас QFileDialog надає діалогове вікно, яке дозволяє користувачам вибирати файли або каталоги.

Клас QFile надає інтерфейс для читання з файлів і запису в них.

Клас QMessageBox надає модальне діалогове вікно для інформування користувача або для того, щоб поставити йому запитання та отримати відповідь.

Клас QTextStream забезпечує зручний інтерфейс для читання та запису тексту.

Клас QTextEdit надає віджет, який використовується для редагування та відображення як простого, так і форматованого тексту.

void QTextEdit::undo()

Скасує останню операцію.

void QTextEdit::redo()

Повторює останню операцію.

void QTextEdit::selectAll()

Виділяє весь текст.

void QTextEdit::copy()

Копіює будь-який виділений текст до буфера обміну.

void QTextEdit::paste()

Вставляє текст із буфера обміну в текстове редагування у поточній позиції курсору.

void QTextEdit::cut()

Копіює виділений текст до буфера обміну та видаляє його з редагування тексту.

void QTextEdit::clear()

Видаляє весь текст у редагуванні тексту.

void QTextEdit::setPlainText(const QString &text)

Змінює текст редагування тексту на текст рядка. Будь-який попередній текст видаляється.

bool QWidget::close()

Закриває цей віджет. Повертає true, якщо віджет було закрито; інакше повертає false.

QString QTextEdit::toPlainText() const

Повертає текст редагованого тексту як звичайний текст.

QString QFileDialog::getSaveFileName()

Це зручна статична функція, яка повертає назву файлу, вибрану користувачем. Файл не повинен існувати

QMessageBox::warning()

Відкриває вікно попередження з заданим заголовком і текстом перед вказаним батьківським віджетом.

QString QTextStream::readAll()

Читає весь вміст потоку та повертає його як QString. Уникайте цієї функції під час роботи з великими файлами, оскільки вона споживає значний обсяг пам’яті.

**Лабораторне завдання**

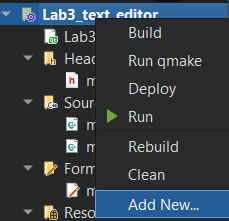
1. Створити віконний проект. Додати головне та контекстне меню, необхідні системні діалоги.
2. Реалізувати текстовий редактор і переглядач графічних файлів.

**Хід роботи**

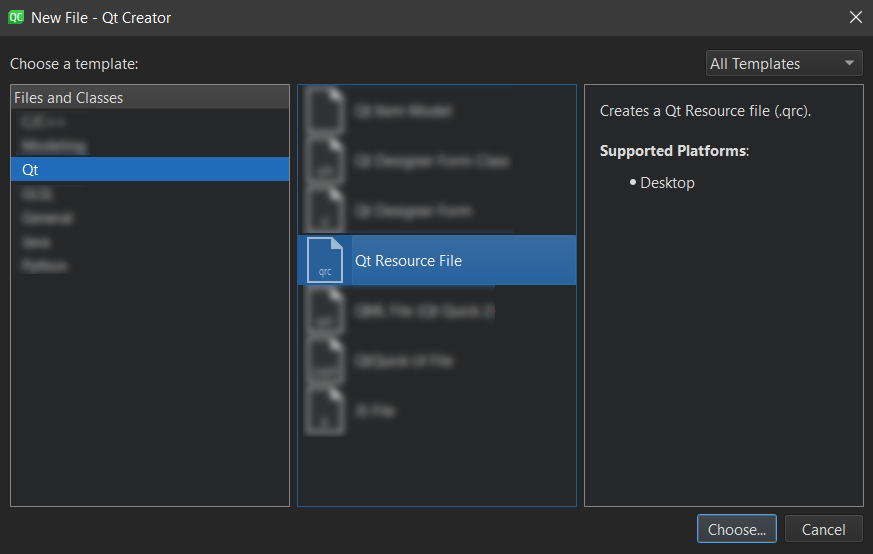
Щоб зробити текстовий редактор додаємо компоненти інтерфейсу, натиснувши на “Type Here” панелі зверху:



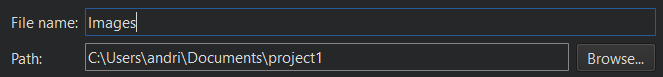
Натискаємо на назву проекту правою кнопкою миші, щоб створити файл ресурсів з метою додавання зображень до панелей. Обираємо “Add New…”:



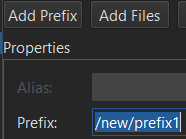
В наступному вікні обираємо Qt -> Qt Resource File:



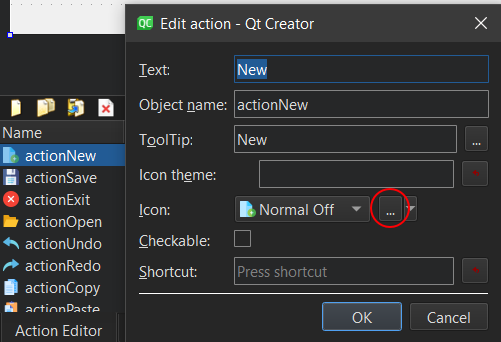
Даємо назву файлу ресурсів:



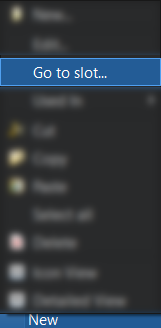
Додаємо префікс та додаємо файли:



Щоб додати зображення до активної панелі двічі натискаємо на неї в меню, у відкритому вікні обирамо “Icon”->… та виділяємо потрібні зображення:



Щоб додати дію для якоїсь кнопки в панелі інтрументів натискаємо правою кнопкою на назву кнопки, в меню обираємо “Go to slot…”



Додаємо необхідний код:

void MainWindow::on\_actionNew\_triggered()

{

ui->textEdit->clear();

}

void MainWindow::on\_actionSave\_triggered()

{

QString file\_name = QFileDialog::getSaveFileName(this, tr("Save file"), "",

tr("Text file (\*.txt);;C++ File (\*.cpp \*.h)"));

if (file\_name != "") {

QFile file(file\_name);

if (file.open(QIODevice::ReadWrite)) {

QTextStream stream(&file);

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

file.flush();

file.close();

}

else {

QMessageBox::critical(this, tr("Error"), tr("Can't save file."));

return;

}

}

}

void MainWindow::on\_actionExit\_triggered()

{

close();

}

void MainWindow::on\_actionOpen\_triggered()

{

QString file\_name = QFileDialog::getOpenFileName(this, "Open a file", "C:\\Users\\andri\\Documents\\Lab3\_text\_editor", "Text file(\*.txt)");

QFile file(file\_name);

if(!file.open(QFile::ReadOnly | QFile::Text)){

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

}

QTextStream in(&file);

QString text = in.readAll();

ui->textEdit->setPlainText(text);

file.close();

}

void MainWindow::on\_actionUndo\_triggered()

{

ui->textEdit->undo();

}

void MainWindow::on\_actionRedo\_triggered()

{

ui->textEdit->redo();

}

void MainWindow::on\_actionCopy\_triggered()

{

ui->textEdit->copy();

}

void MainWindow::on\_actionPaste\_triggered()

{

ui->textEdit->paste();

}

void MainWindow::on\_actionCut\_triggered()

{

ui->textEdit->cut();

}

void MainWindow::on\_actionSelect\_All\_triggered()

{

ui->textEdit->selectAll();

}

Щоб зробити переглядач зображень додаємо до вікна елемент “Label” та кнопку “pushButton”.



Додаємо необхідний код:

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(*parent*)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

QPixmap pix("C://Users//andri//Documents//Lab3\_img\_viewer//choose.png");

ui->Label\_picture->setPixmap(pix.scaled(1400, 700, Qt::*KeepAspectRatio*));

}

void MainWindow::on\_pushButton\_clicked()

{

QFileDialog dialog(*this*);

dialog.setNameFilter(tr("Images (\*.png \*.jpg \*.gif \*.bmp)"));

dialog.setViewMode(QFileDialog::*Detail*);

QString fileName = QFileDialog::getOpenFileName(*this*,

tr ("Open Images"), "C://Users//andri//Pictures", tr("Image Files (\*.png \*.jpg \*.gif \*.bmp)"));

*if*(!fileName.isEmpty())

{

QImage image (fileName);

ui->Label\_picture->setPixmap(QPixmap::fromImage(image));

}

ui->Label\_picture->setScaledContents(*true*);

}

**Висновки**

На цій лабораторній роботі я навчився створювати віконний проект та використовувати, створювати візуальні компоненти програм, реалізував текстовий редактор і переглядач графічних файлів.

**Тема роботи:** Компоненти Qt Creator для представлення даних.

**Мета** **роботи:** Створити віконний проект та продемонструвати використання компонент призначених для відображення та опрацювання даних.

**Теоретичні відомості**

Віджет QPushButton містить командну кнопку.

Кнопка, або кнопка команди, є, мабуть, найбільш часто використовуваним віджетом у будь-якому графічному інтерфейсі користувача. Натисніть кнопку, щоб наказати комп’ютеру виконати певну дію або відповісти на запитання. Типовими кнопками є OK, Застосувати, Скасувати, Закрити, Так, Ні та Довідка.

Віджет QLabel забезпечує відображення тексту або зображення

**Лабораторне завдання**

1. Реалізувати гру.

**Хід роботи**

1. Створюю віконний проект.
2. Додаю потрібні компоненти (кнопки, позначки)

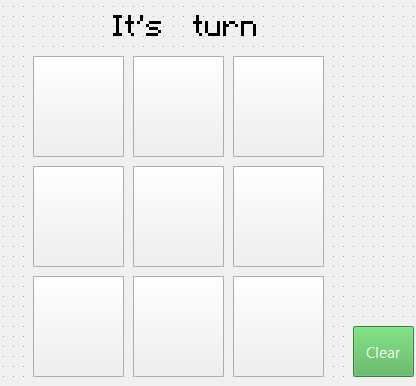


Рис.1 Кнопки та позначки

1. Реалізовую гру, додаючи необхідний код.

{

ui->setupUi(this);

ui->label\_currentPlayer->setText("X");

connect(ui->pushButton\_1, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_2, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_3, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_4, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_5, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_6, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_7, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_8, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_9, SIGNAL(clicked(bool)), this, SLOT(checkx(bool)));

connect(ui->pushButton\_1, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_2, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_3, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_4, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_5, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_6, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_7, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_8, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

connect(ui->pushButton\_9, SIGNAL(clicked(bool)), this, SLOT(checko(bool)));

}

const char\* currentSym = "X";

bool MainWindow::checkx(){

if(ui->pushButton\_1->text()=="X" && ui->pushButton\_2->text()=="X" && ui->pushButton\_3->text()=="X" ){

return true;

}

if(ui->pushButton\_4->text()=="X" && ui->pushButton\_5->text()=="X" && ui->pushButton\_6->text()=="X" ){

return true;

}

if(ui->pushButton\_7->text()=="X" && ui->pushButton\_8->text()=="X" && ui->pushButton\_9->text()=="X" ){

return true;

}

if(ui->pushButton\_1->text()=="X" && ui->pushButton\_4->text()=="X" && ui->pushButton\_7->text()=="X" ){

return true;

}

if(ui->pushButton\_2->text()=="X" && ui->pushButton\_5->text()=="X" && ui->pushButton\_8->text()=="X" ){

return true;

}

if(ui->pushButton\_3->text()=="X" && ui->pushButton\_6->text()=="X" && ui->pushButton\_9->text()=="X" ){

return true;

}

if(ui->pushButton\_1->text()=="X" && ui->pushButton\_5->text()=="X" && ui->pushButton\_9->text()=="X" ){

return true;

}

if(ui->pushButton\_3->text()=="X" && ui->pushButton\_5->text()=="X" && ui->pushButton\_7->text()=="X" ){

return true;

}

return false;

}

bool MainWindow::checko(){

if(ui->pushButton\_1->text()=="O" && ui->pushButton\_2->text()=="O" && ui->pushButton\_3->text()=="O" ){

return true;

}

if(ui->pushButton\_4->text()=="O" && ui->pushButton\_5->text()=="O" && ui->pushButton\_6->text()=="O" ){

return true;

}

if(ui->pushButton\_7->text()=="O" && ui->pushButton\_8->text()=="O" && ui->pushButton\_9->text()=="O" ){

return true;

}

if(ui->pushButton\_1->text()=="O" && ui->pushButton\_4->text()=="O" && ui->pushButton\_7->text()=="O" ){

return true;

}

if(ui->pushButton\_2->text()=="O" && ui->pushButton\_5->text()=="O" && ui->pushButton\_8->text()=="O" ){

return true;

}

if(ui->pushButton\_3->text()=="O" && ui->pushButton\_6->text()=="O" && ui->pushButton\_9->text()=="O" ){

return true;

}

if(ui->pushButton\_1->text()=="O" && ui->pushButton\_5->text()=="O" && ui->pushButton\_9->text()=="O" ){

return true;

}

if(ui->pushButton\_3->text()=="O" && ui->pushButton\_5->text()=="O" && ui->pushButton\_7->text()=="O" ){

return true;

}

return false;

}

void MainWindow::on\_pushButton\_1\_clicked()

{

ui->pushButton\_1->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_2\_clicked()

{

ui->pushButton\_2->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_3\_clicked()

{

ui->pushButton\_3->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_4\_clicked()

{

ui->pushButton\_4->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_5\_clicked()

{

ui->pushButton\_5->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_6\_clicked()

{

ui->pushButton\_6->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_7\_clicked()

{

ui->pushButton\_7->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_8\_clicked()

{

ui->pushButton\_8->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_9\_clicked()

{

ui->pushButton\_9->setText(currentSym);

if(strcmp( currentSym, "X") == 0){

currentSym = "O";

ui->label\_currentPlayer->setText(currentSym);

}else{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

}

if(checkx()){

QMessageBox::about(this, "You Win", "X won the game");

}

if(checko()){

QMessageBox::about(this, "You Win", "O won the game");

}

}

void MainWindow::on\_pushButton\_clear\_clicked()

{

currentSym = "X";

ui->label\_currentPlayer->setText(currentSym);

ui->pushButton\_1->setText("");

ui->pushButton\_2->setText("");

ui->pushButton\_3->setText("");

ui->pushButton\_4->setText("");

ui->pushButton\_5->setText("");

ui->pushButton\_6->setText("");

ui->pushButton\_7->setText("");

ui->pushButton\_8->setText("");

ui->pushButton\_9->setText("");

}

**Виконання лаборатоної роботи**

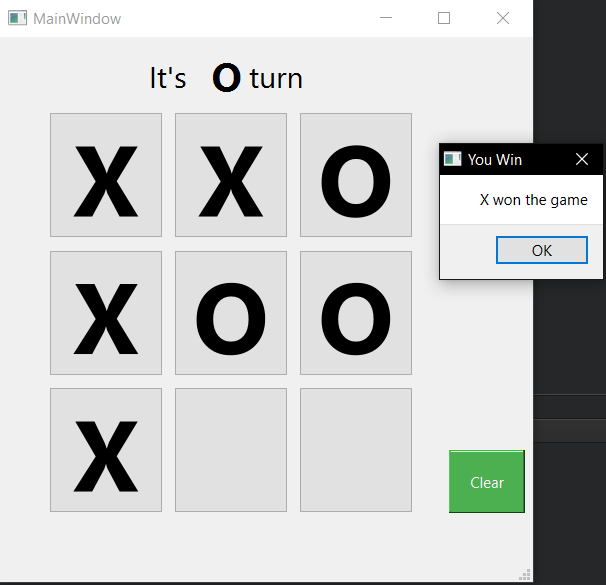


Рис. 2 Робота програми

**Висновки**

На даній лабораторній роботі створив віконний проект та продемонстрував використання компонент призначених для відображення та опрацювання даних та реалізував гру.