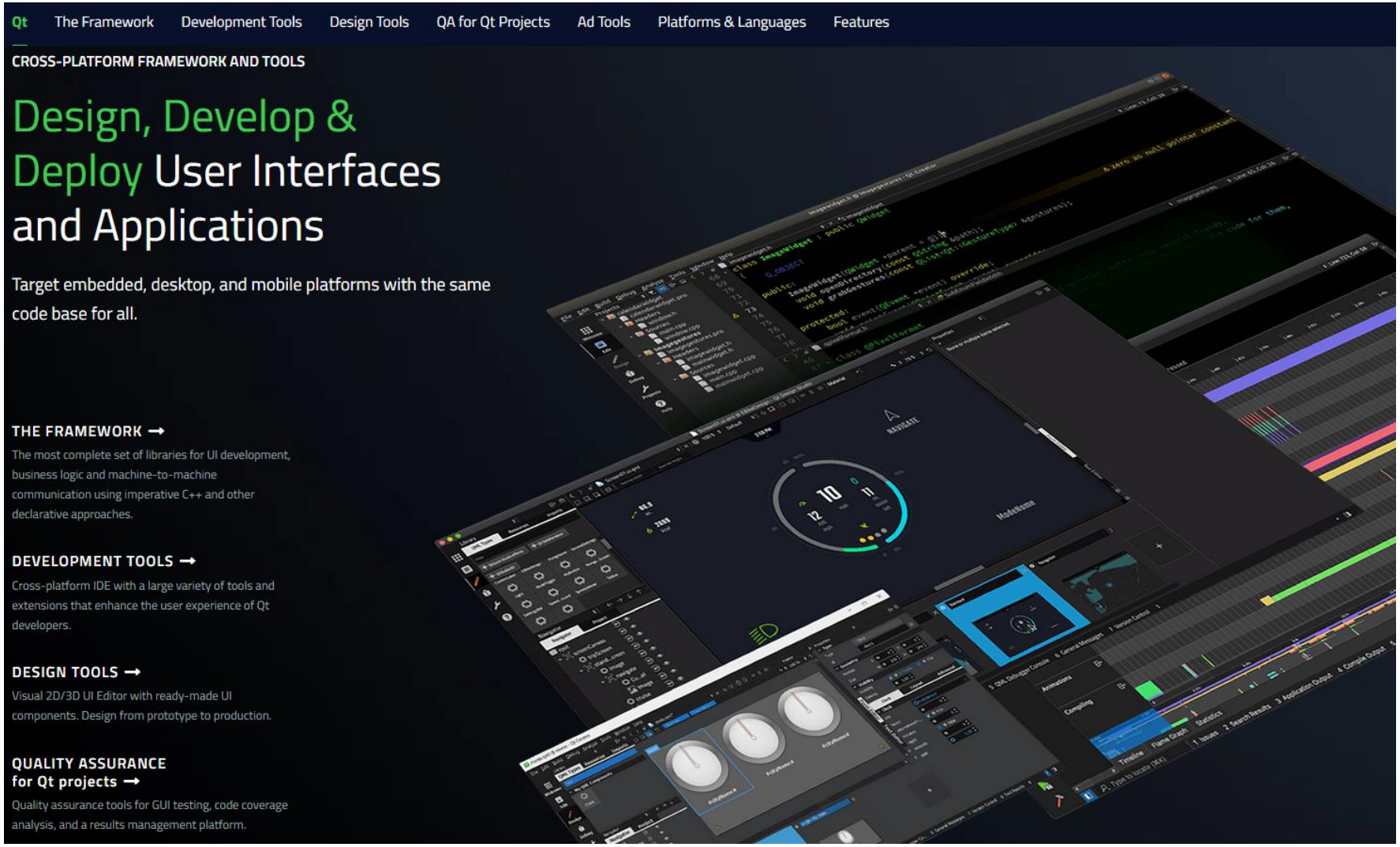
Qt

Code less, create more, deploy everywhere.
1.26

Qt

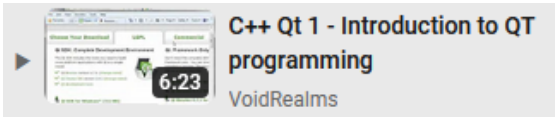
biblioteki i narzędzia do budowy m.in. aplikacji desktoowych oraz mobilnych dla środowisk Windows, Linux , Mac OS, Android, iOS



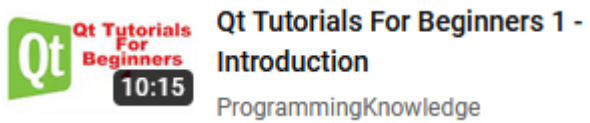
https://www.qt.io/product

tutorial

https://forbot.pl/blog/kurs-qt-1-czym-jest-qt-pierwsza-aplikacja-w-praktyce-id35549

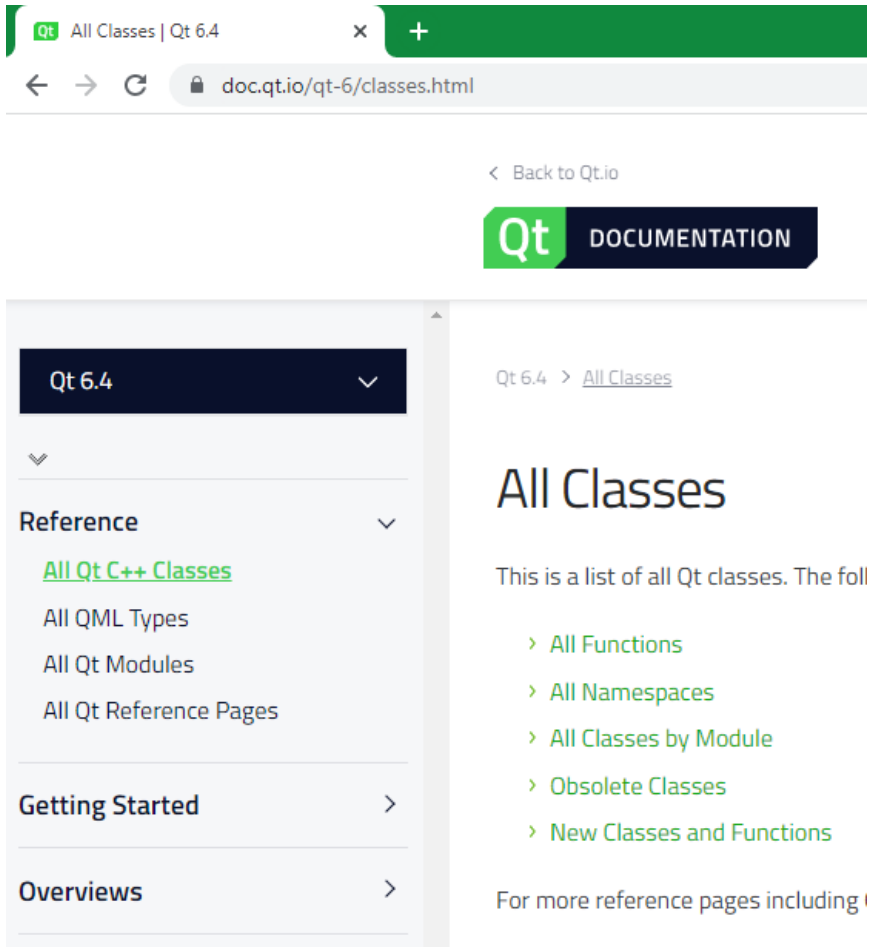


https://www.youtube.com/watch?v=6KtOzh0StTc&list=PL2D1942A4688E9D63&index=1



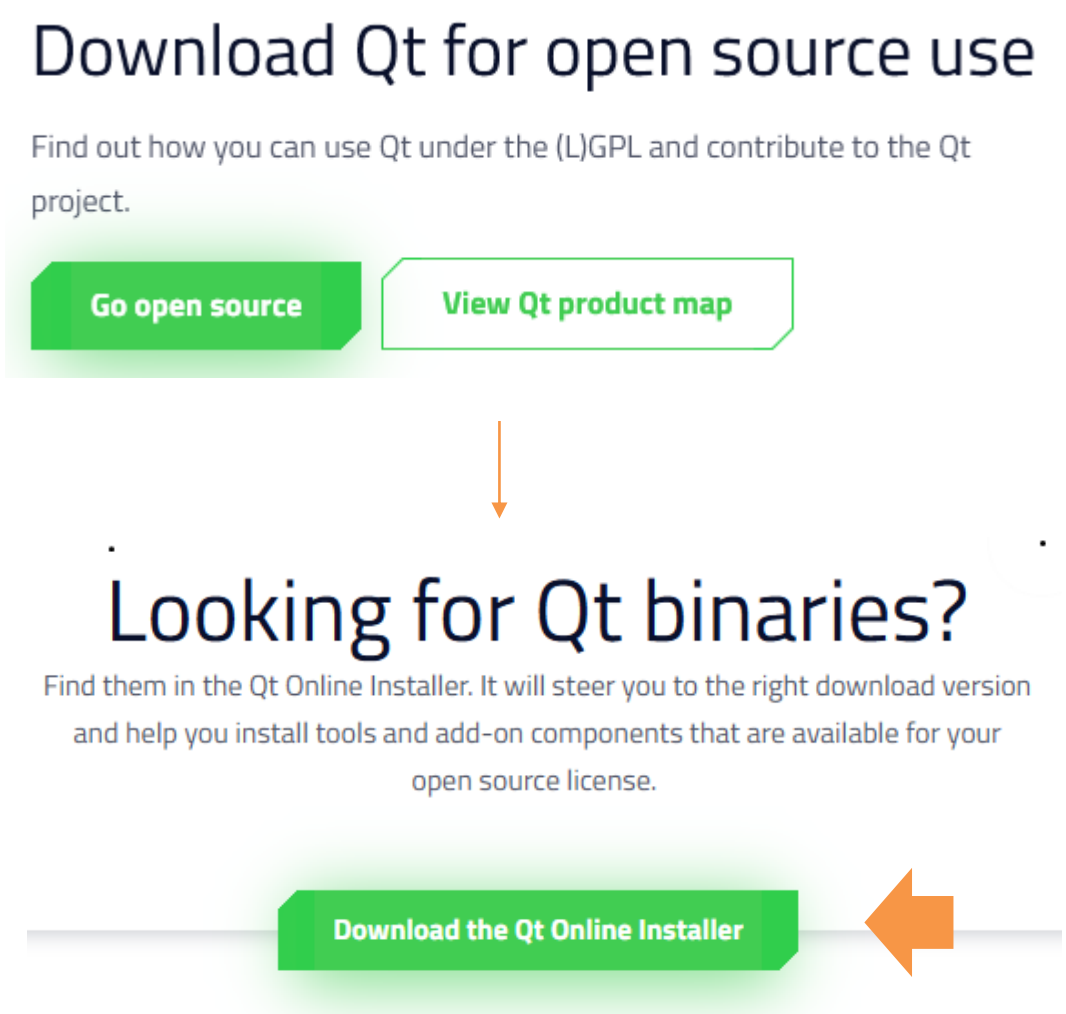
https://www.youtube.com/watch?v=EkjaiDsiM-Q&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA

documentation



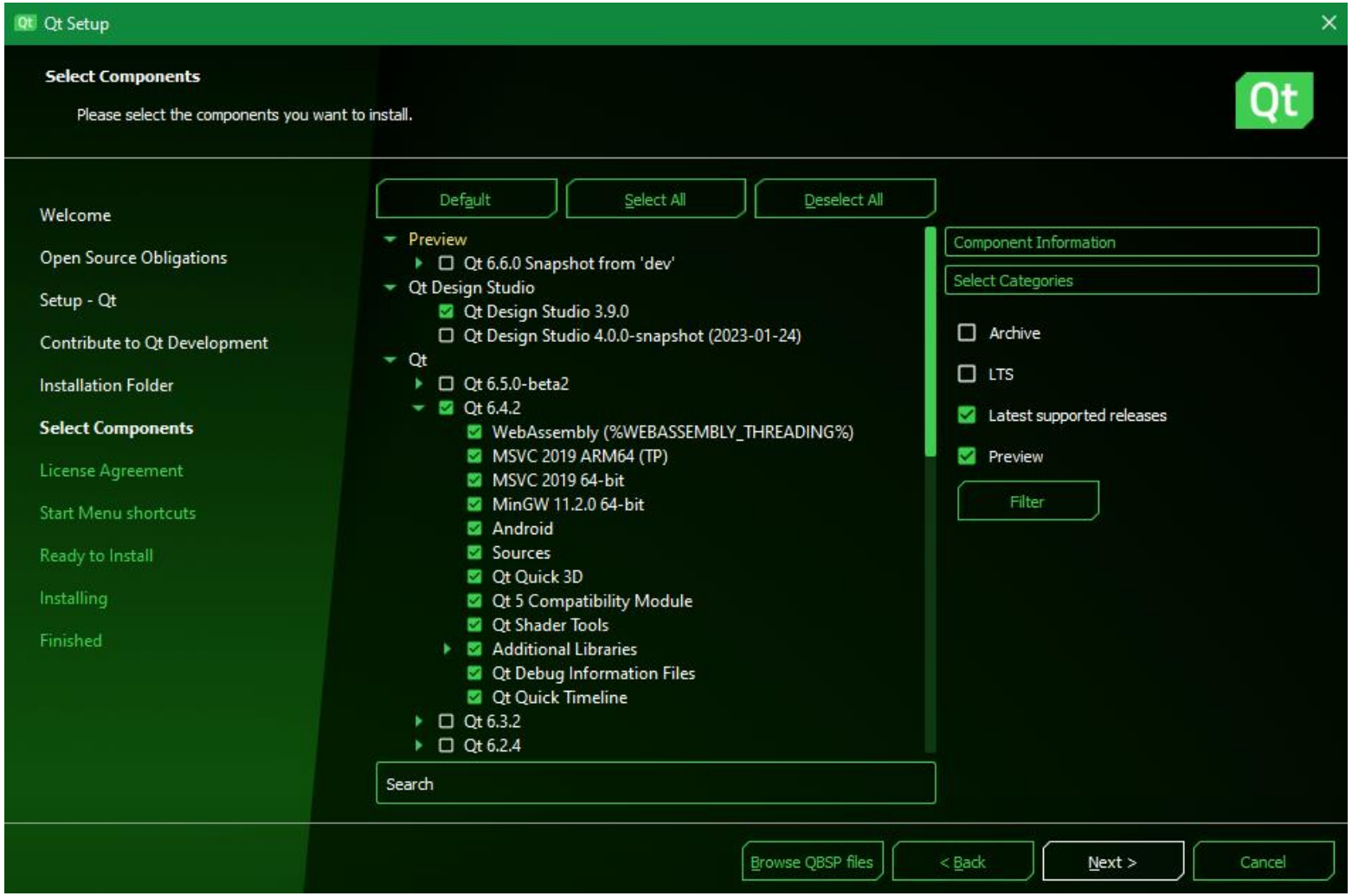
https://doc.qt.io/qt-6/classes.html

download

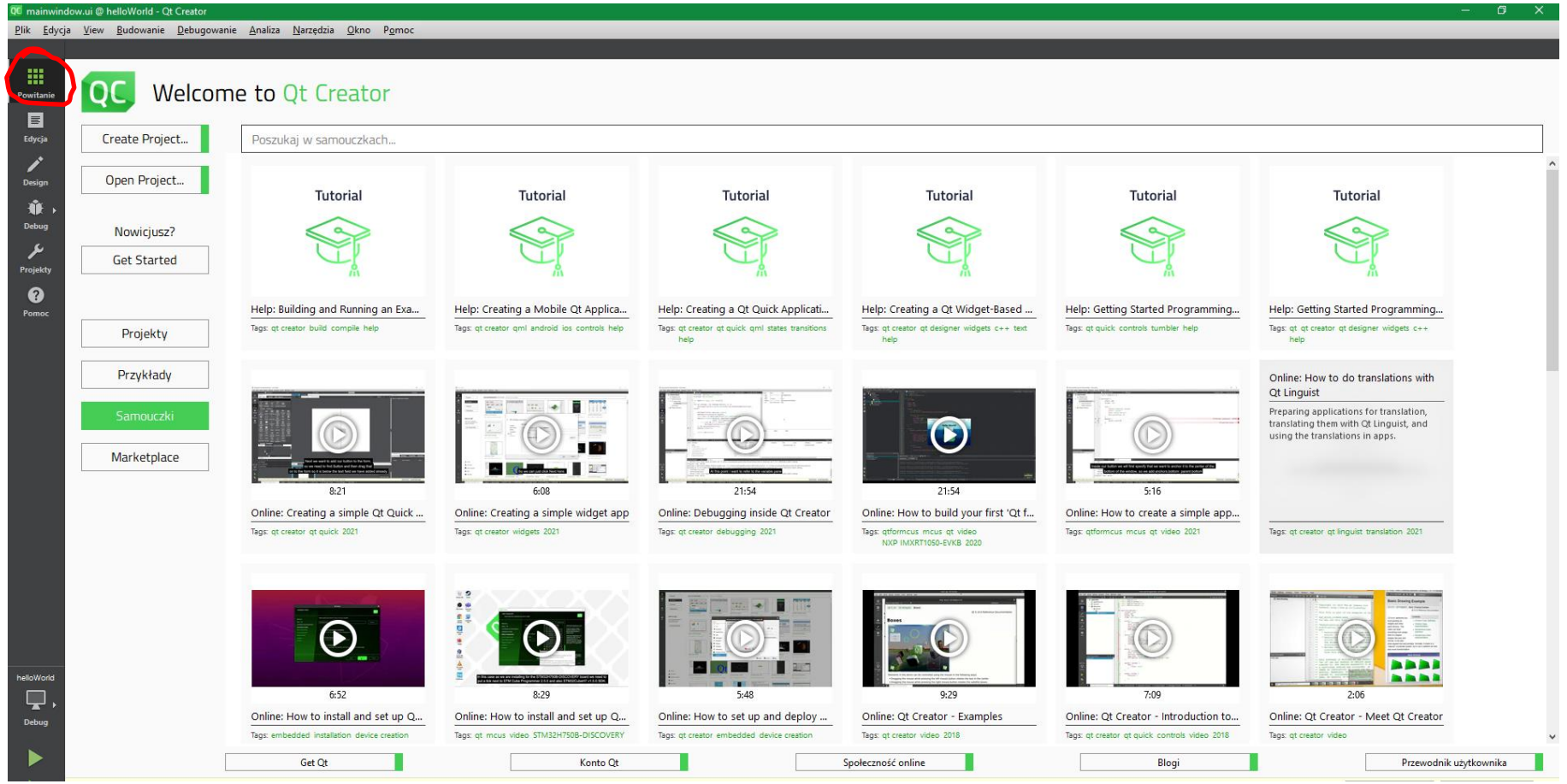


https://www.qt.io/download-open-source

download

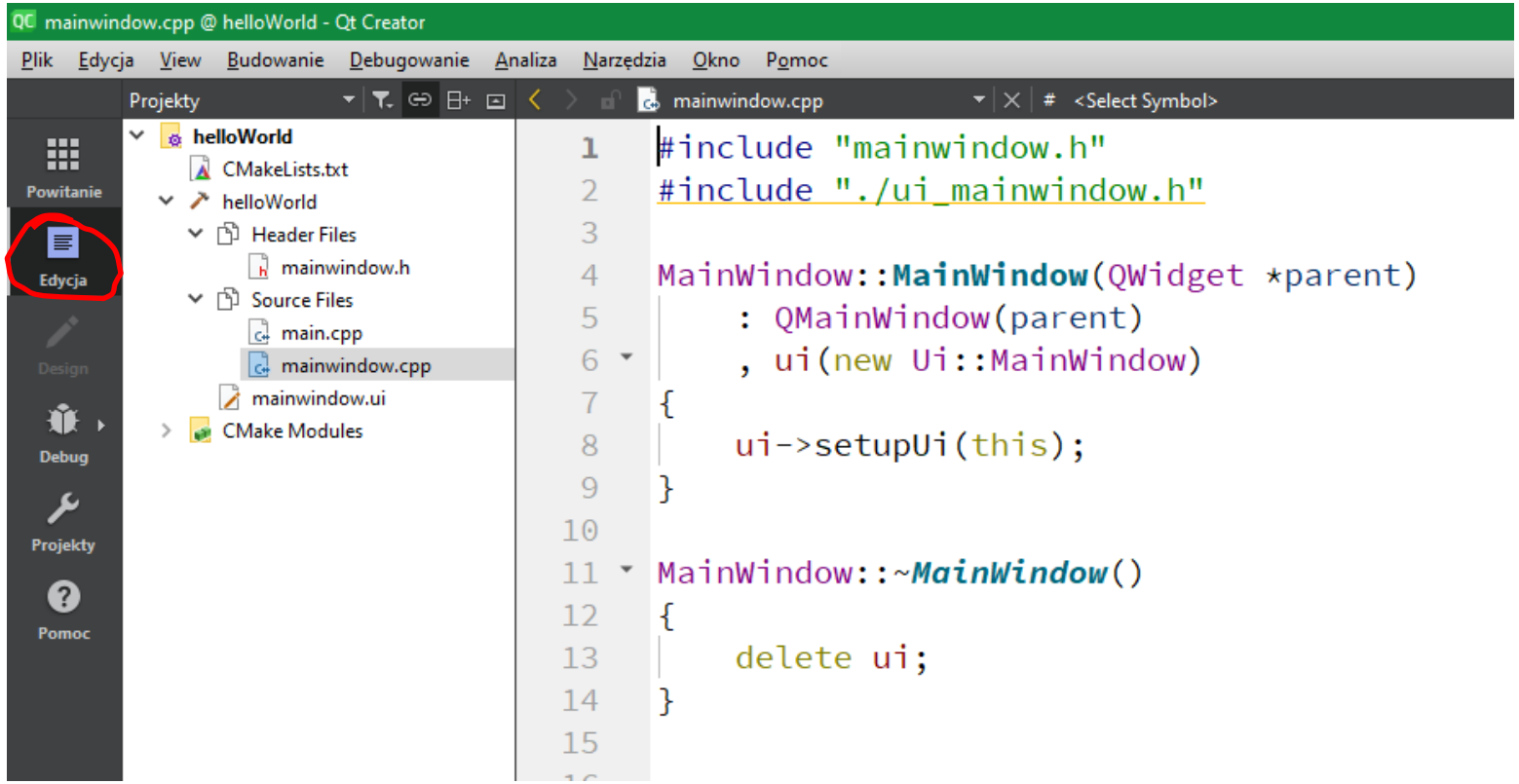

Qt Setup
Select Components
Please select the components you want to install.
Welcome
Open Source Obligations
Setup - Qt
Contribute to Qt Development
Installation Folder
Select Components
License Agreement
Start Menu shortcuts
Ready to Install
Installing
Finished
Default
Select All
Deselect All
Preview
Qt 6.6.0 Snapshot from 'dev'
Qt Design Studio
Qt Design Studio 3.9.0
Qt Design Studio 4.0.0-snapshot (2023-01-24)
Qt
Qt 6.5.0-beta2
Qt 6.4.2
WebAssembly (%WEBASSEMBLY_THREADING%)
MSVC 2019 ARM64 (TP)
MSVC 2019 64-bit
MinGW 11.2.0 64-bit
Android
Sources
Qt Quick 3D
Qt 5 Compatibility Module
Qt Shader Tools
Additional Libraries
Qt Debug Information Files
Qt Quick Timeline
Qt 6.3.2
Qt 6.2.4
Search
Component Information
Select Categories
Archive
LTS
Latest supported releases
Preview
Filter
Browse QBSP files
< Back
Next >
Cancel

Qt Creator



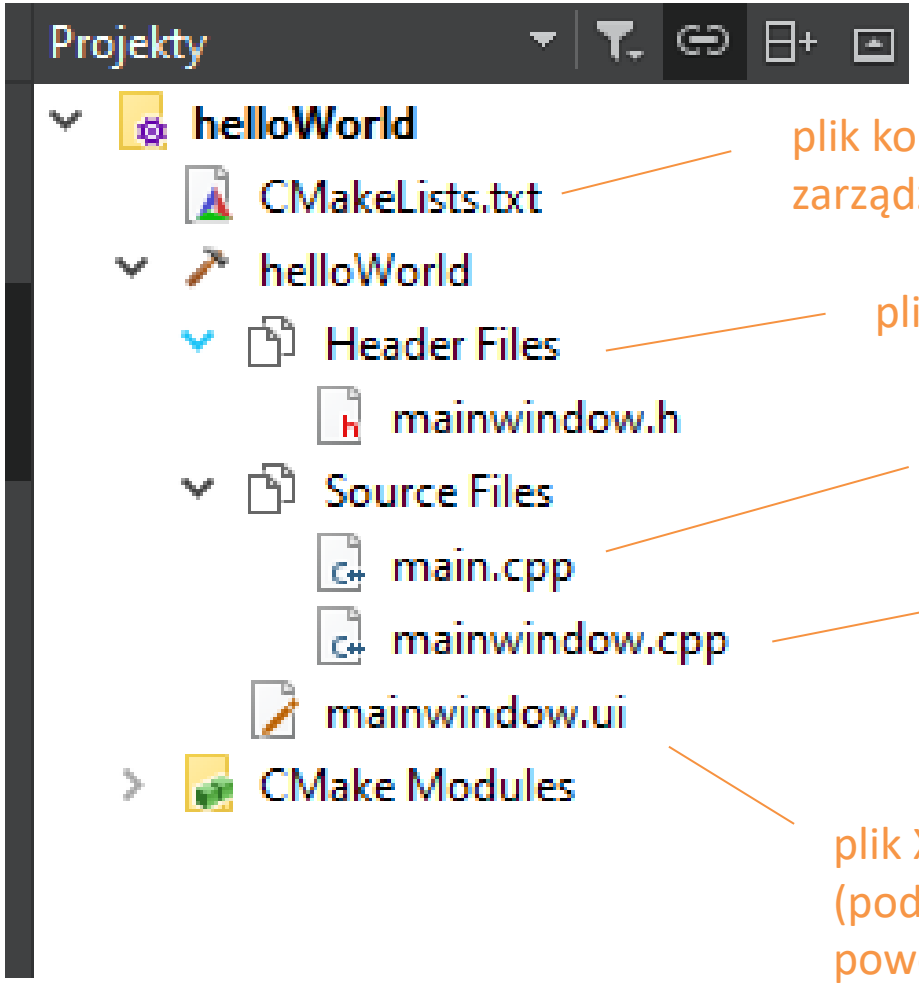
strona z tutorialami, przykładami

Qt Creator



edytor

Qt Creator



plik konfiguracyjny programu CMake zarządzającego kompilacją programu

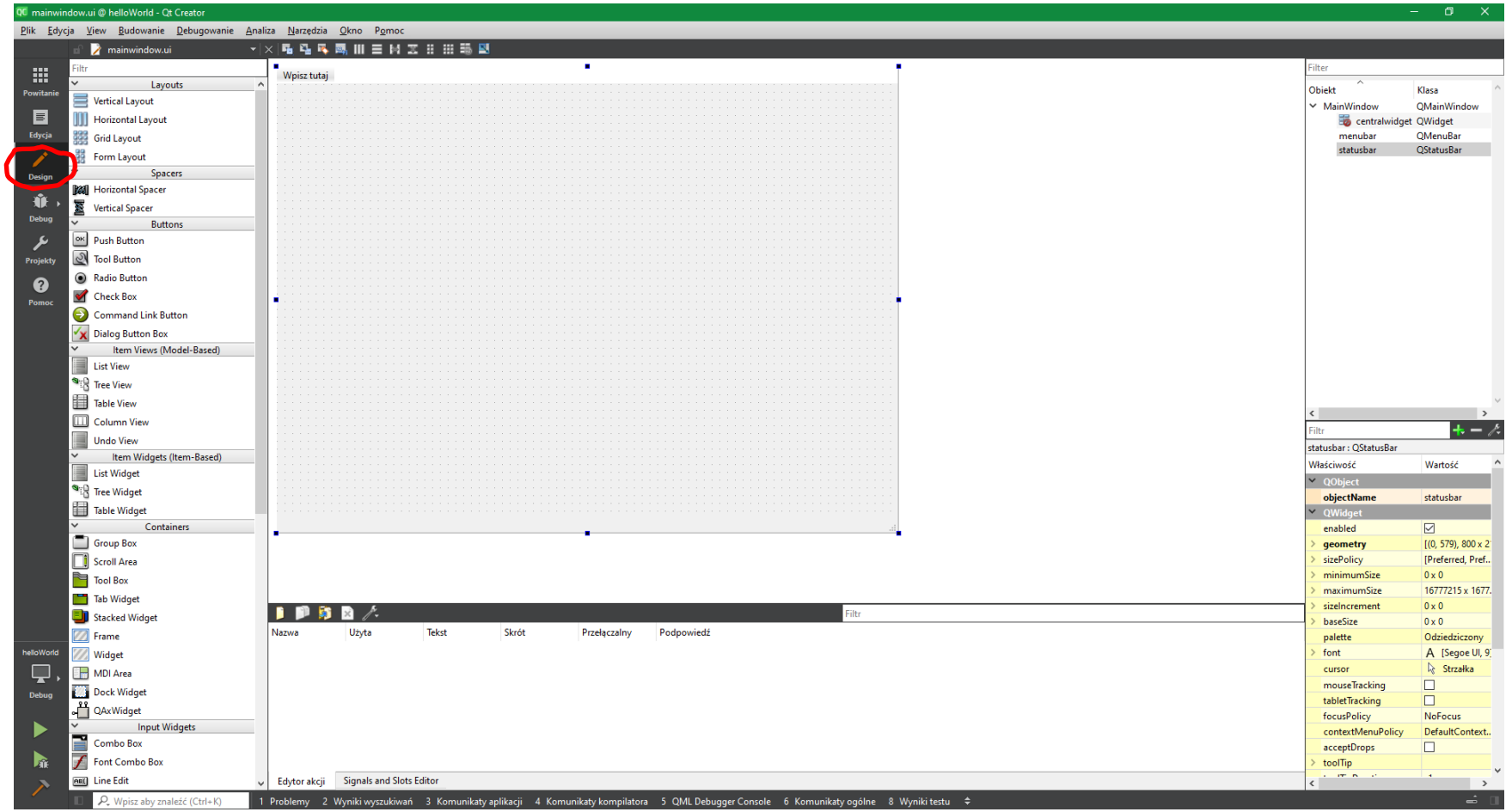
plik nagłówkowy klasy MainWindow

plik z funkcją main()

plik z klasą MainWindow głównego okna aplikacji

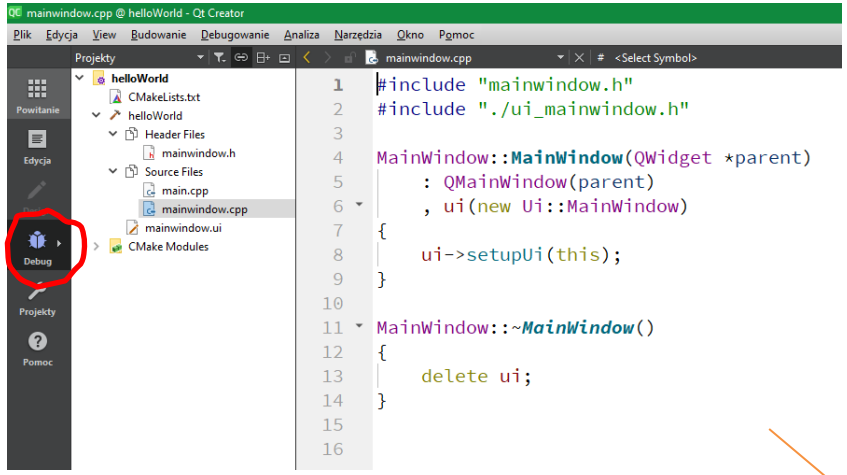
plik XML interfejsu graficznego (podwójne kliknięcie na pliku powoduje przejście do zakładki Design)

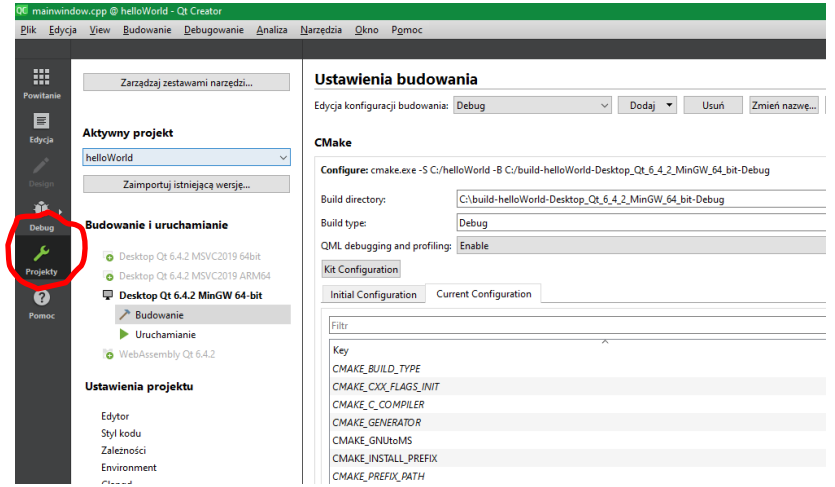
Qt Creator

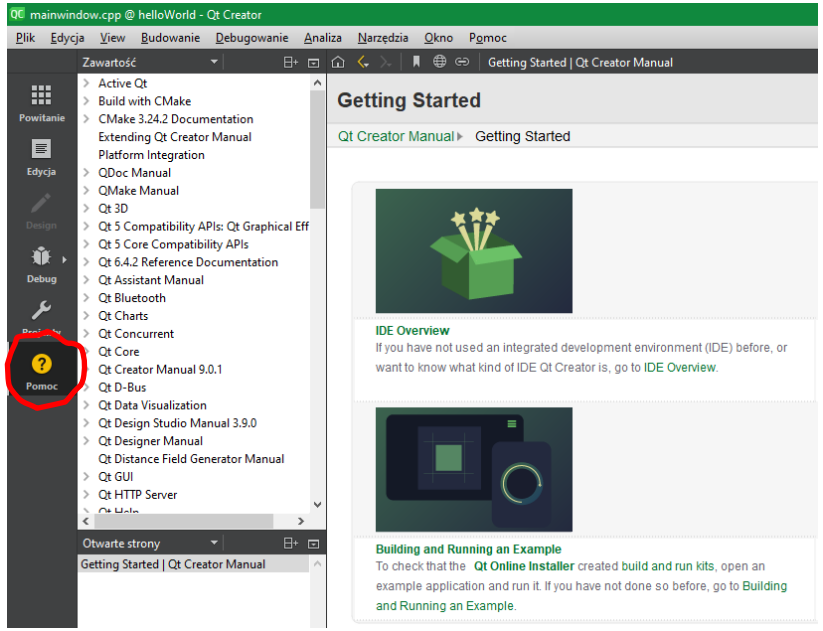


narzędzie do projektowania interfejsu

Qt Creator





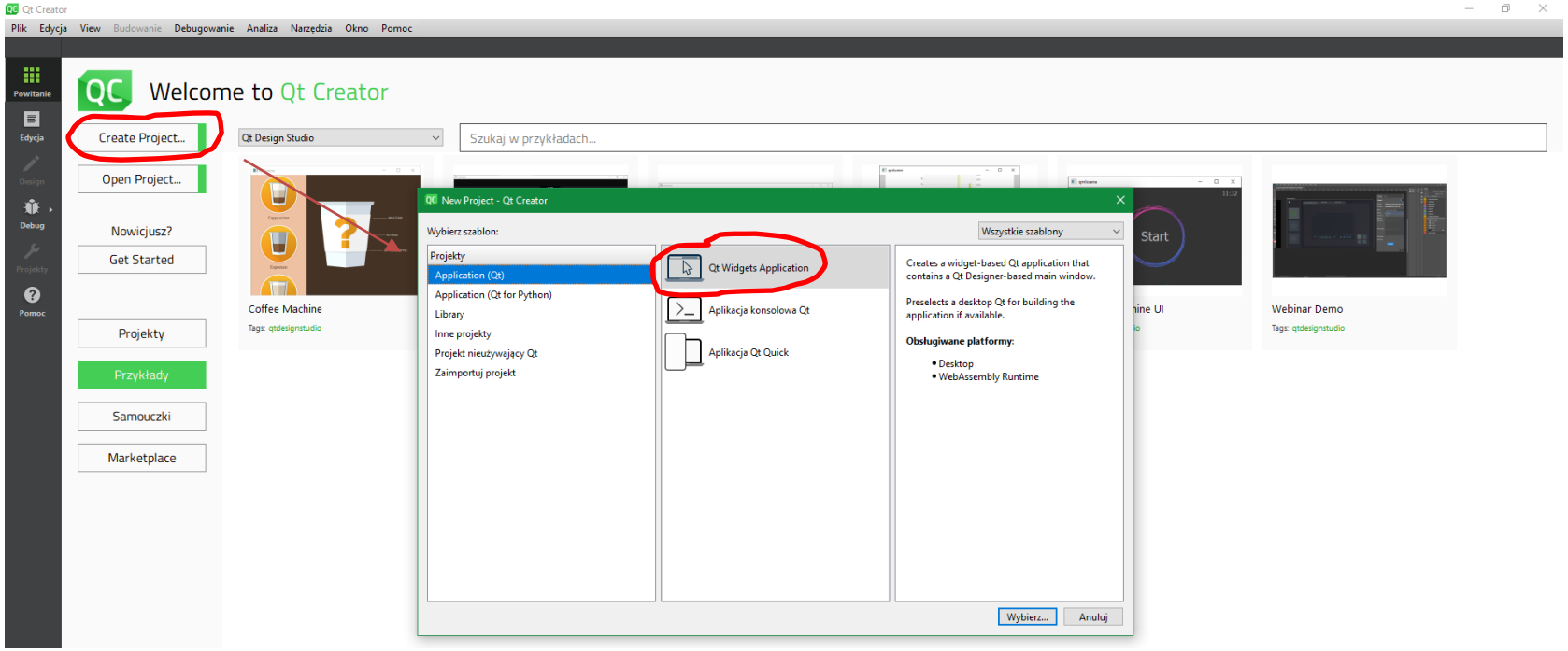


debugger
(usuwanie błędów)

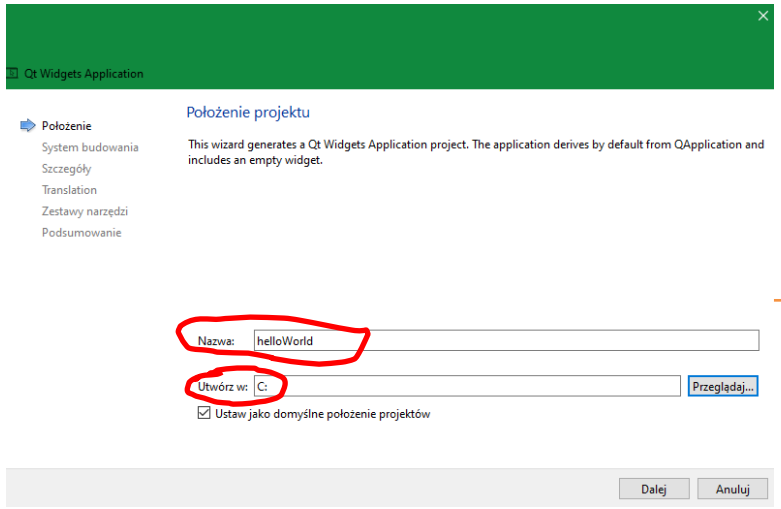
konfiguracja projektu

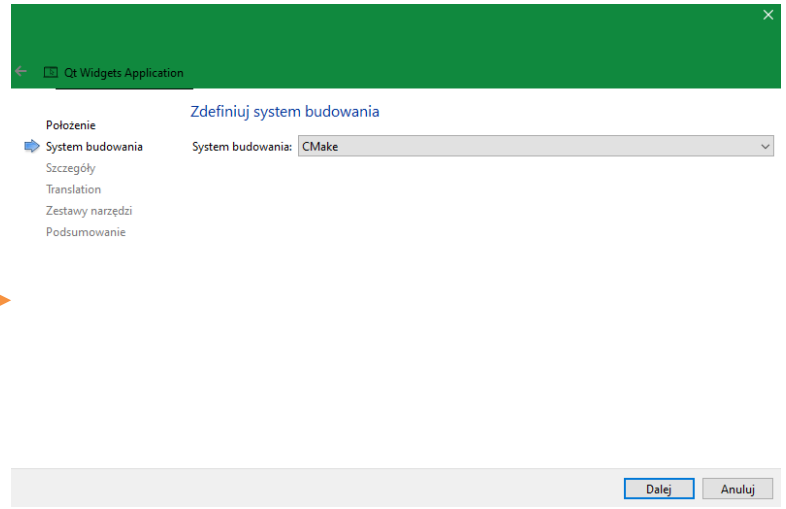
źródła pomocy

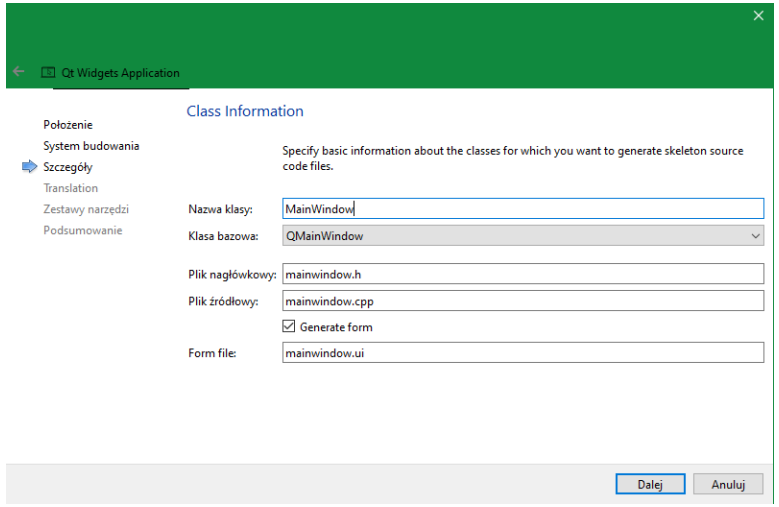
new project

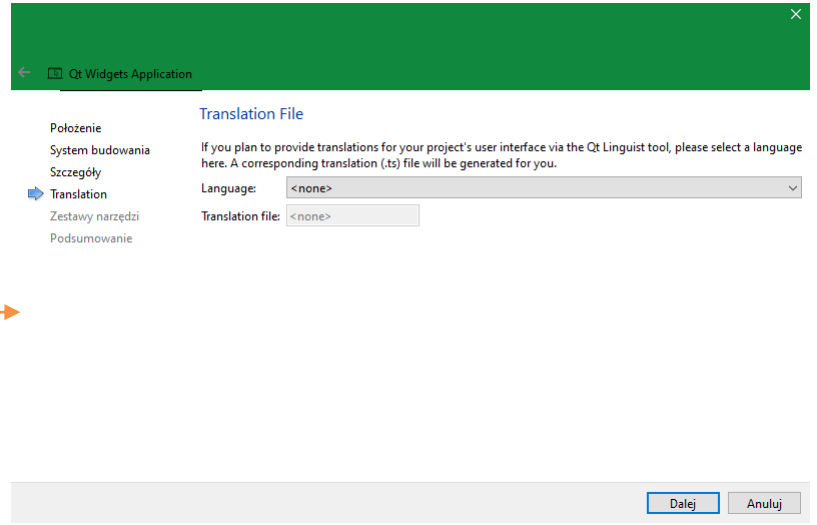

Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
Welcome to Qt Creator
Create Project...
Open Project...
Nowicjusz?
Get Started
Projekty
Przykłady
Samouczki
Marketplace
Qt Design Studio
Szukaj w przykładach...
Coffee Machine
Tags: qtdesignstudio
New Project - Qt Creator
Wybierz szablon:
Wszystkie szablony
Projekty
Application (Qt)
Application (Qt for Python)
Library
Inne projekty
Projekt nieużywający Qt
Zaimportuj projekt
Qt Widgets Application
Aplikacja konsolowa Qt
Aplikacja Qt Quick
Creates a widget-based Qt application that contains a Qt Designer-based main window.
Preselects a desktop Qt for building the application if available.
Obsługiwane platformy:
Desktop
WebAssembly Runtime
Wybierz...
Anuluj
Start
Webinar Demo
Tags: qtdesignstudio

new project

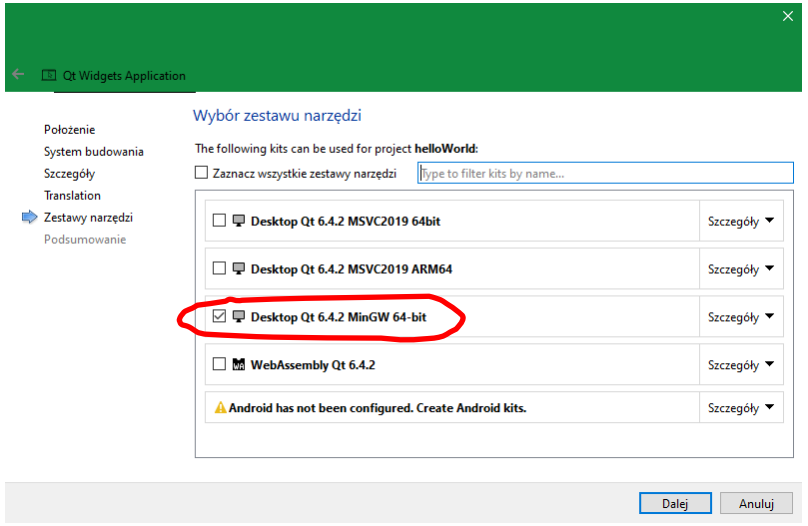

Qt Widgets Application
Położenie projektu
Położenie
System budowania
Szczegóły
Translation
Zestawy narzędzi
Podsumowanie
This wizard generates a Qt Widgets Application project. The application derives by default from QApplication and includes an empty widget.
Nazwa: helloWorld
Utwórz w: C:
Przeglądaj...
Ustaw jako domyślne położenie projektów
Dalej
Anuluj

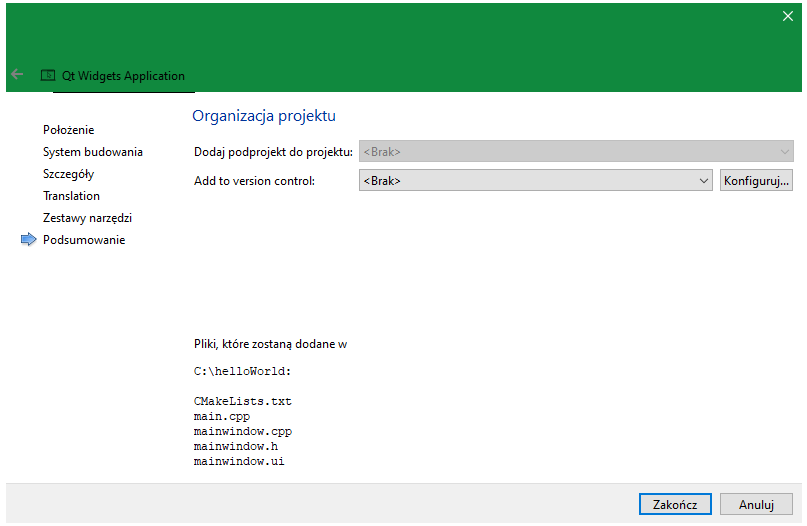

Qt Widgets Application
Położenie
System budowania
Szczegóły
Translation
Zestawy narzędzi
Podsumowanie
Zdefiniuj system budowania
System budowania: CMake
Dalej
Anuluj


Qt Widgets Application
Położenie
System budowania
Szczegóły
Translation
Zestawy narzędzi
Podsumowanie
Class Information
Specify basic information about the classes for which you want to generate skeleton source code files.
Nazwa klasy: MainWindow
Klasa bazowa: QMainWindow
Plik nagłówkowy: mainwindow.h
Plik źródłowy: mainwindow.cpp
Generate form
Form file: mainwindow.ui
Dalej
Anuluj


Qt Widgets Application
Położenie
System budowania
Szczegóły
Translation
Zestawy narzędzi
Podsumowanie
Translation File
If you plan to provide translations for your project's user interface via the Qt Linguist tool, please select a language here. A corresponding translation (.ts) file will be generated for you.
Language: <none>
Translation file: <none>
Dalej
Anuluj

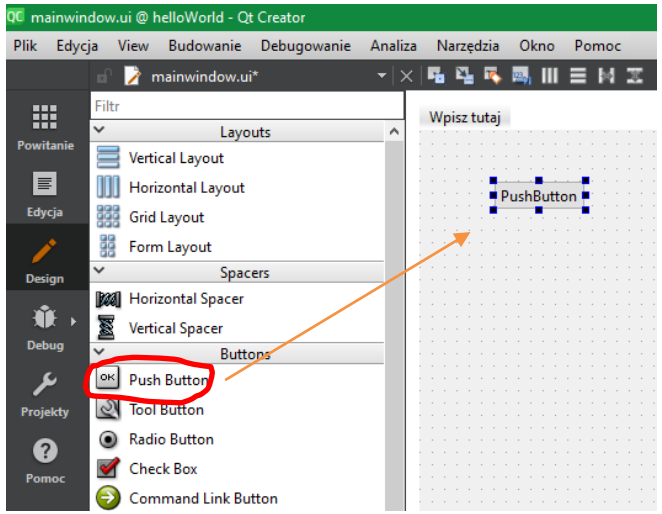
new project


Qt Widgets Application
Położenie
System budowania
Szczegóły
Translation
Zestawy narzędzi
Podsumowanie
Wybór zestawu narzędzi
The following kits can be used for project helloWorld:
Zaznacz wszystkie zestawy narzędzi
Type to filter kits by name...
Desktop Qt 6.4.2 MSVC2019 64bit
Desktop Qt 6.4.2 MSVC2019 ARM64
Desktop Qt 6.4.2 MinGW 64-bit
WebAssembly Qt 6.4.2
Android has not been configured. Create Android kits.
Szczegóły
Dalej
Anuluj

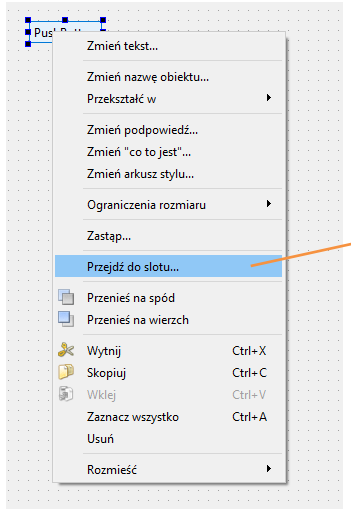

Qt Widgets Application
Położenie
System budowania
Szczegóły
Translation
Zestawy narzędzi
Podsumowanie
Organizacja projektu
Dodaj podprojekt do projektu:
<Brak>
Add to version control:
<Brak>
Konfiguruj...
Pliki, które zostaną dodane w
C:\helloWorld:
CMakeLists.txt
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
Zakończ
Anuluj

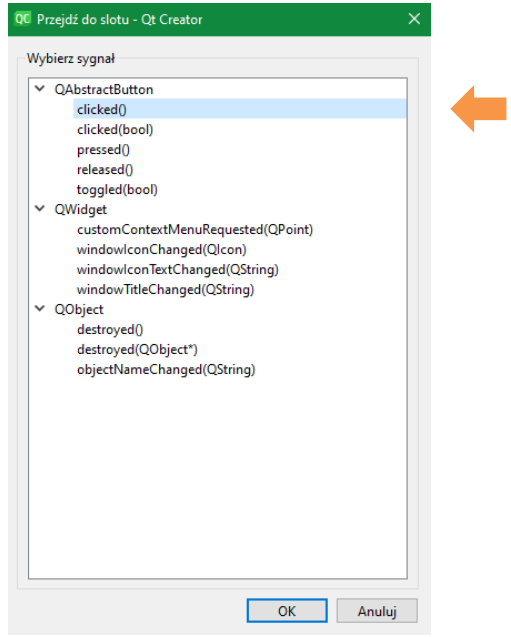
helloWorld

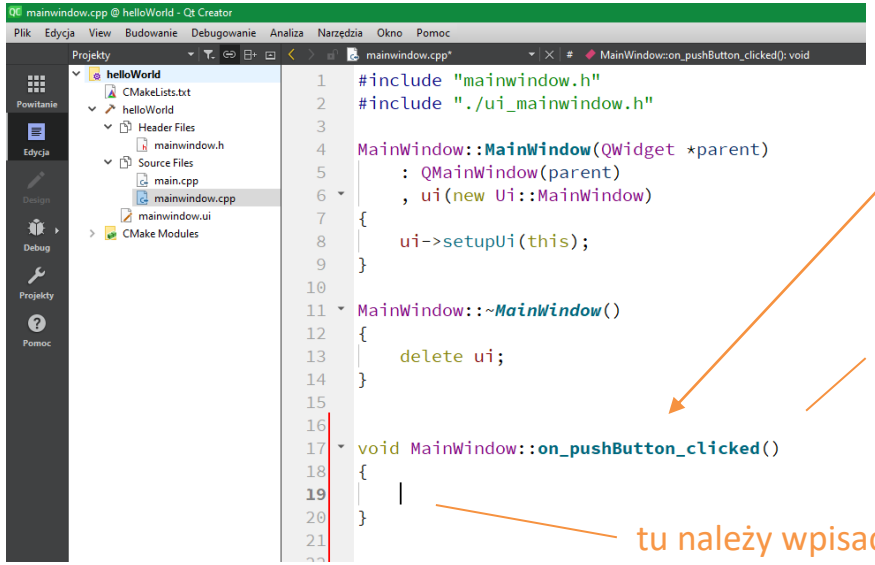
łączymy sygnał przycisku *click()*
ze slotem okna



prawy przycisk myszy na buttonie



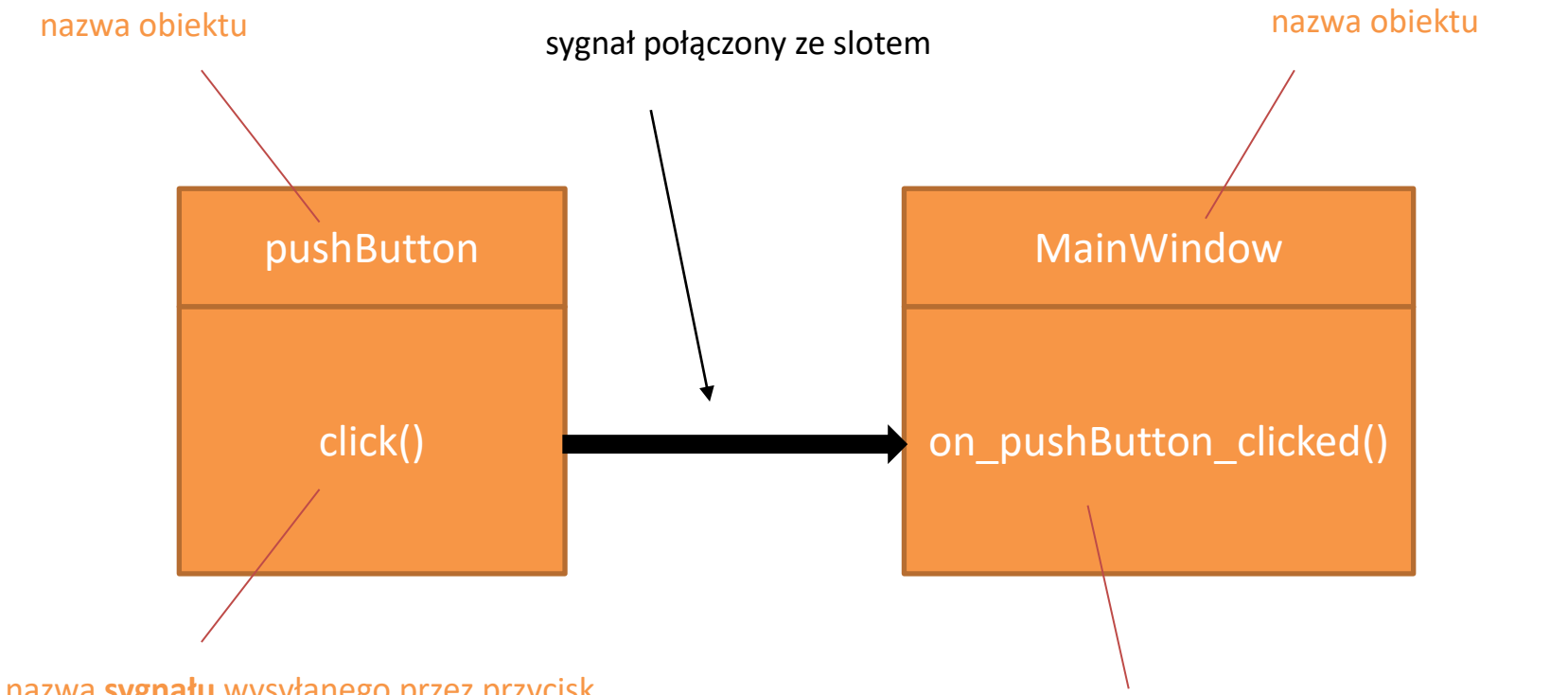




implementacja slotu okna
(metoda obsługująca kliknięcie przycisku)

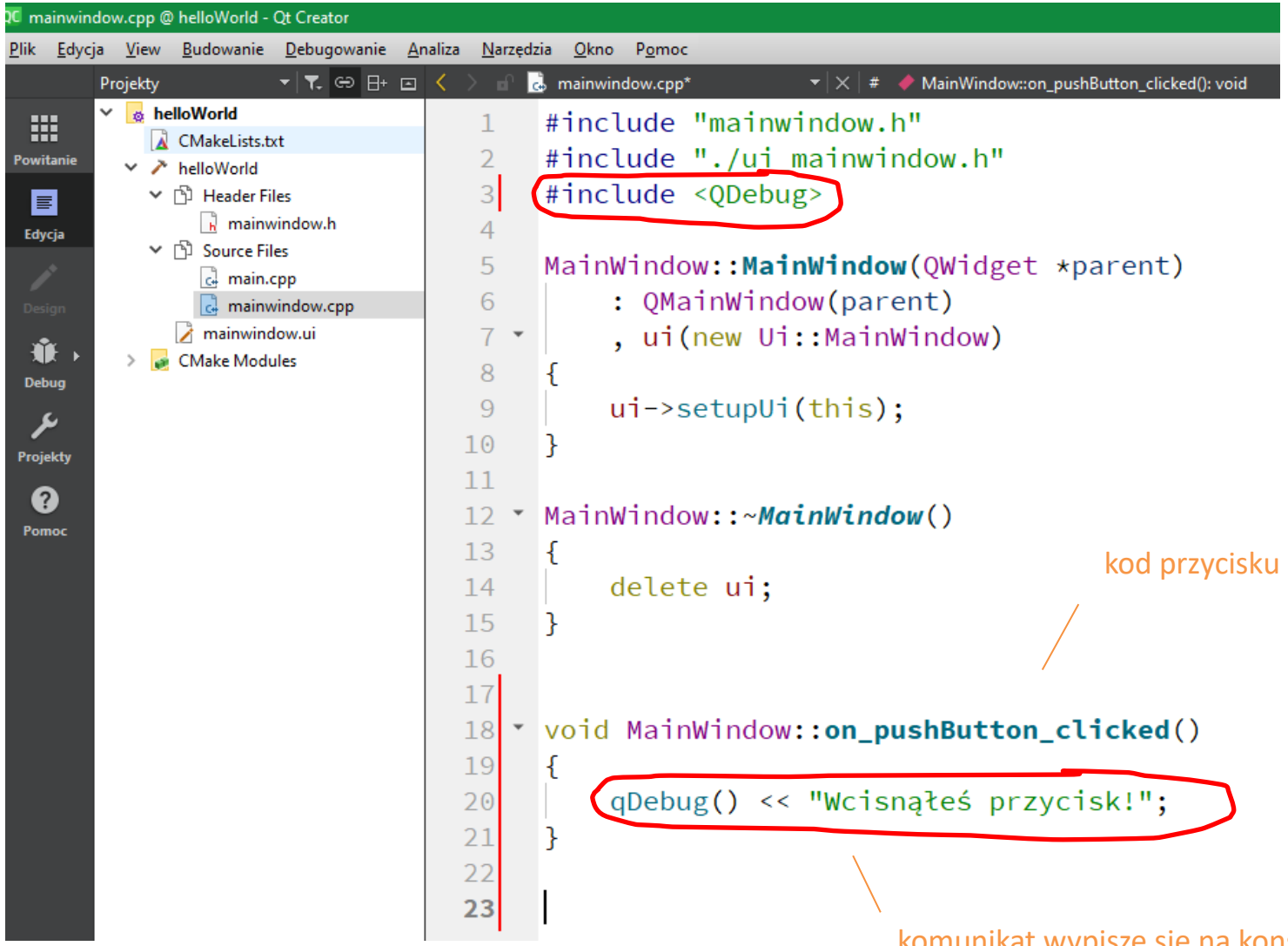
tu należy wpisać kod przycisku

helloWorld

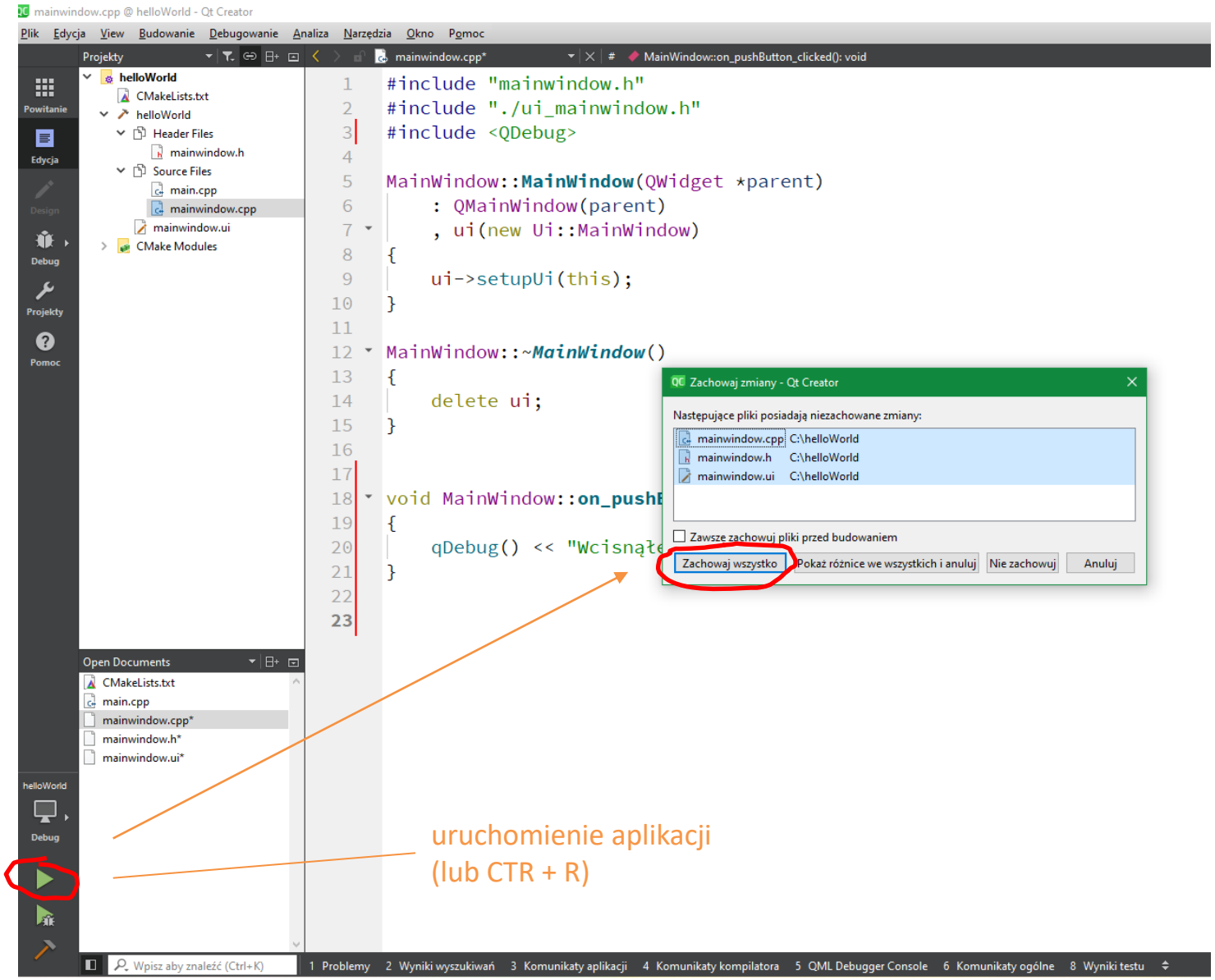


https://doc.qt.io/qt-6.2/signalsandslots.html

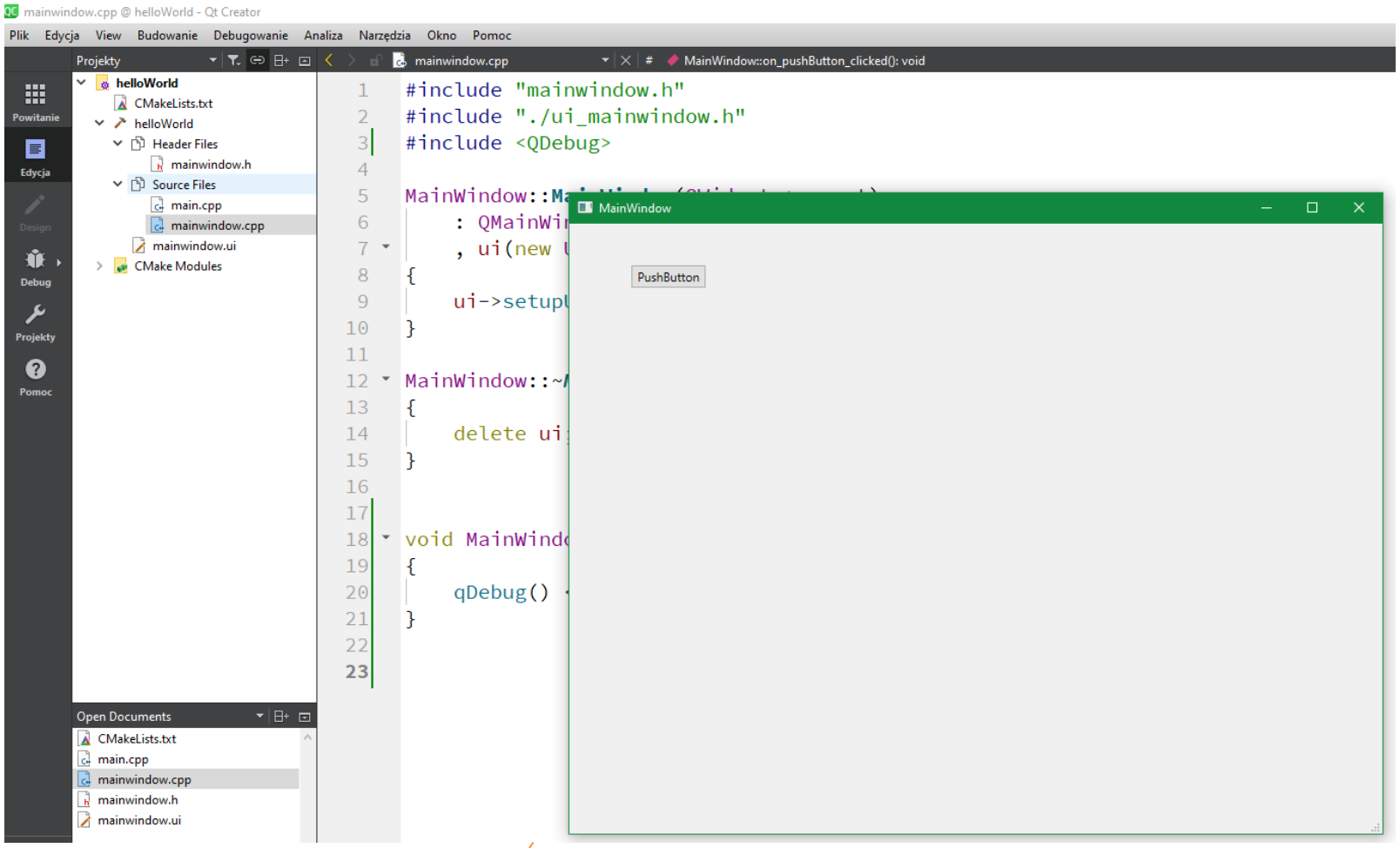
helloWorld


mainwindow.cpp @ helloWorld - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Projekty
mainwindow.cpp*
MainWindow::on_pushButton_clicked(): void
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
helloWorld
CMakeLists.txt
helloWorld
Header Files
mainwindow.h
Source Files
main.cpp
mainwindow.cpp
mainwindow.ui
CMake Modules
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QDebug>
MainWindow::MainWindow(QWidget *parent)
: QMainWindow(parent)
, ui(new Ui::MainWindow)
{
ui->setupUi(this);
}
MainWindow::~MainWindow()
{
delete ui;
}
void MainWindow::on_pushButton_clicked()
{
qDebug() << "Wcisnąłeś przycisk!";
}
kod przycisku (slot)
komunikat wypisze się na konsoli

helloWorld

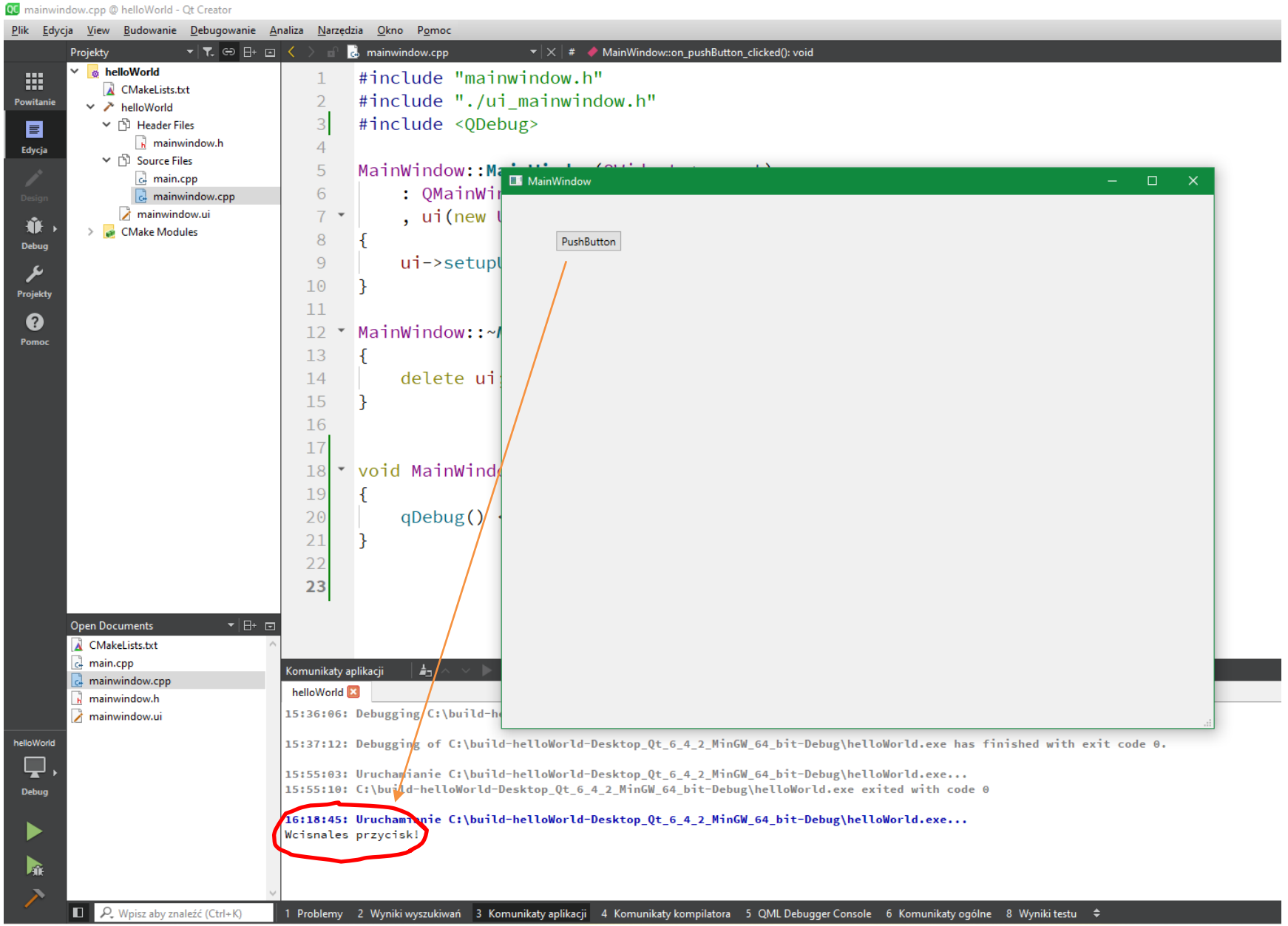

mainwindow.cpp @ helloWorld - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Projekty
helloWorld
CMakeLists.txt
helloWorld
Header Files
mainwindow.h
Source Files
main.cpp
mainwindow.cpp
mainwindow.ui
CMake Modules
mainwindow.cpp*
MainWindow::on_pushButton_clicked(): void
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QDebug>
MainWindow::MainWindow(QWidget *parent)
: QMainWindow(parent)
, ui(new Ui::MainWindow)
{
ui->setupUi(this);
}
MainWindow::~MainWindow()
{
delete ui;
}
void MainWindow::on_pushB
{
qDebug() << "Wcisnąłe
}
Zachowaj zmiany - Qt Creator
Następujące pliki posiadają niezachowane zmiany:
mainwindow.cpp C:\helloWorld
mainwindow.h C:\helloWorld
mainwindow.ui C:\helloWorld
Zawsze zachowuj pliki przed budowaniem
Zachowaj wszystko
Pokaż różnice we wszystkich i anuluj
Nie zachowuj
Anuluj
Open Documents
CMakeLists.txt
main.cpp
mainwindow.cpp*
mainwindow.h*
mainwindow.ui*
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
helloWorld
Debug
uruchomienie aplikacji
(lub CTR + R)
Wpisz aby znaleźć (Ctrl+K)
1 Problemy
2 Wyniki wyszukiwań
3 Komunikaty aplikacji
4 Komunikaty kompilatora
5 QML Debugger Console
6 Komunikaty ogólne
8 Wyniki testu

helloWorld



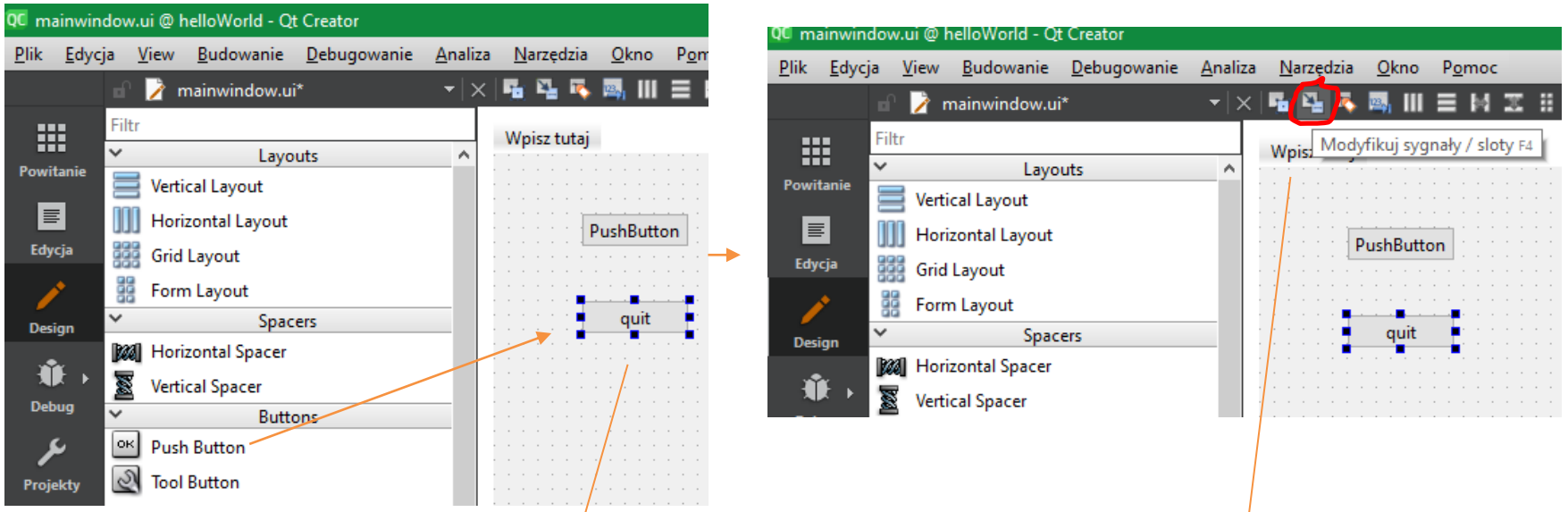
okno aplikacji

helloWorld


mainwindow.cpp @ helloWorld - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Projekty
helloWorld
CMakeLists.txt
helloWorld
Header Files
mainwindow.h
Source Files
main.cpp
mainwindow.cpp
mainwindow.ui
CMake Modules
mainwindow.cpp
MainWindow::on_pushButton_clicked(): void
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QDebug>
MainWindow
PushButton
Open Documents
CMakeLists.txt
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
Komunikaty aplikacji
helloWorld
15:36:06: Debugging C:\build-h
15:37:12: Debugging of C:\build-helloWorld-Desktop_Qt_6_4_2_MinGW_64_bit-Debug\helloWorld.exe has finished with exit code 0.
15:55:03: Uruchamianie C:\build-helloWorld-Desktop_Qt_6_4_2_MinGW_64_bit-Debug\helloWorld.exe...
15:55:10: C:\build-helloWorld-Desktop_Qt_6_4_2_MinGW_64_bit-Debug\helloWorld.exe exited with code 0
16:18:45: Uruchamianie C:\build-helloWorld-Desktop_Qt_6_4_2_MinGW_64_bit-Debug\helloWorld.exe...
Wcisnales przycisk!
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
helloWorld
Debug
Wpisz aby znaleźć (Ctrl+K)
1 Problemy 2 Wyniki wyszukiwań 3 Komunikaty aplikacji 4 Komunikaty kompilatora 5 QML Debugger Console 6 Komunikaty ogólne 8 Wyniki testu

helloWorld

dodajemy nowy przycisk

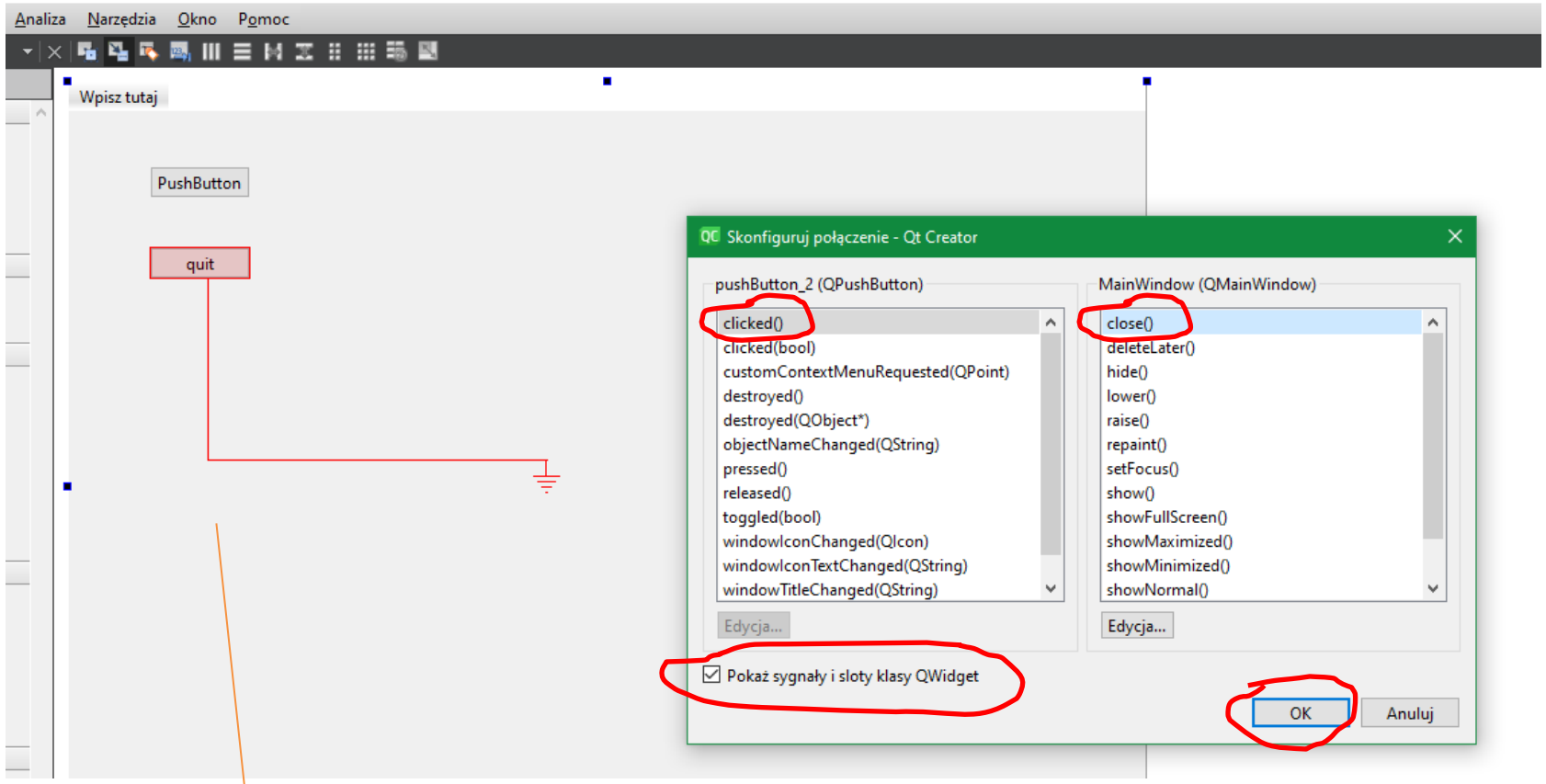


zmiana nazwy przycisku na *quit*

włączamy przycisk

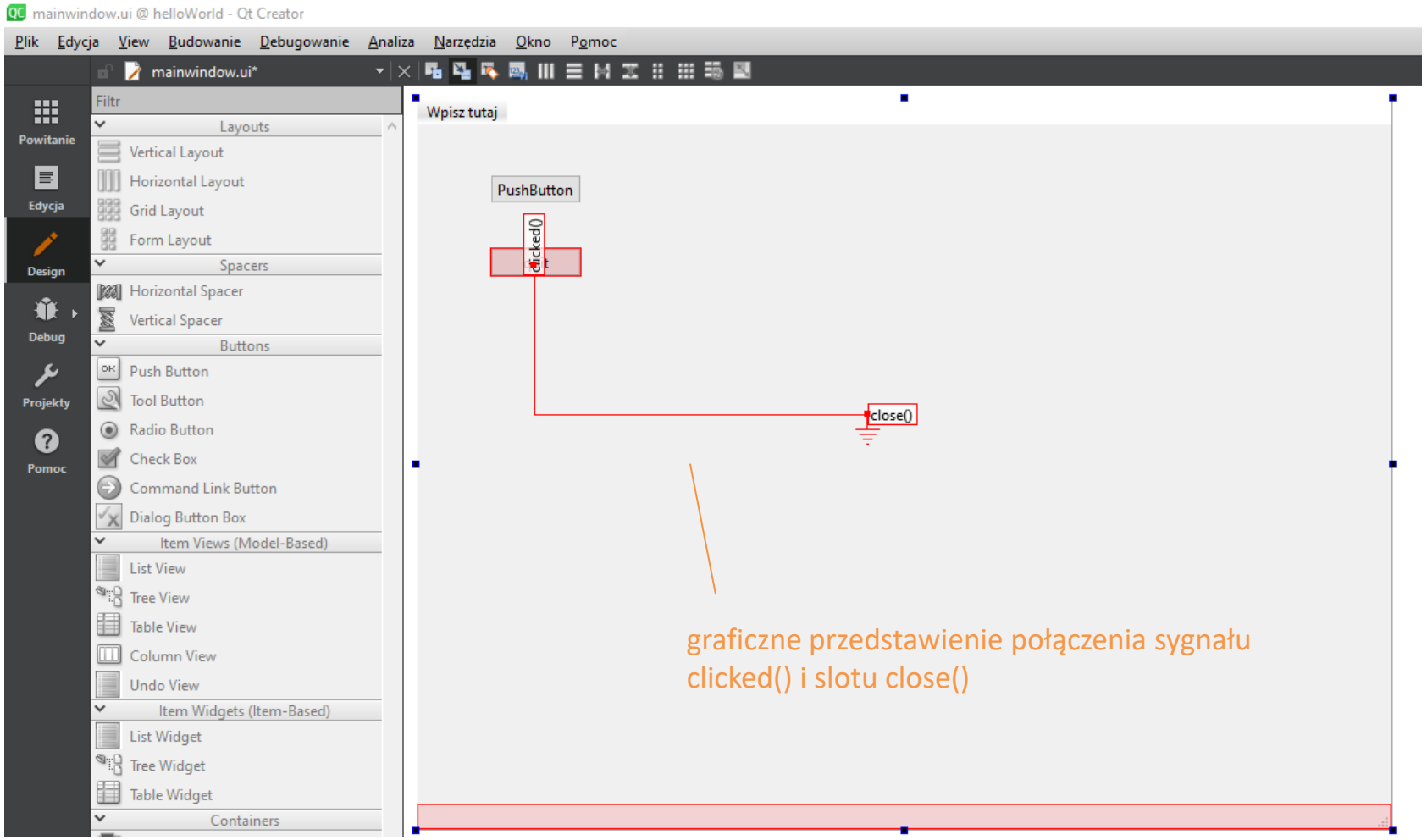
helloWorld

przypisanie do sygnału clicked() slotu close()

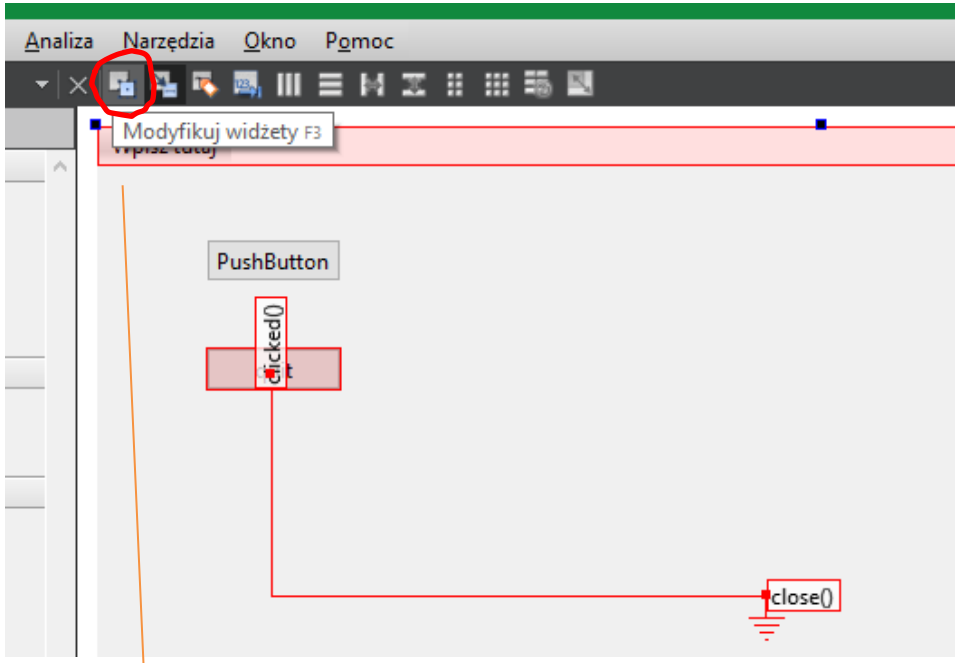


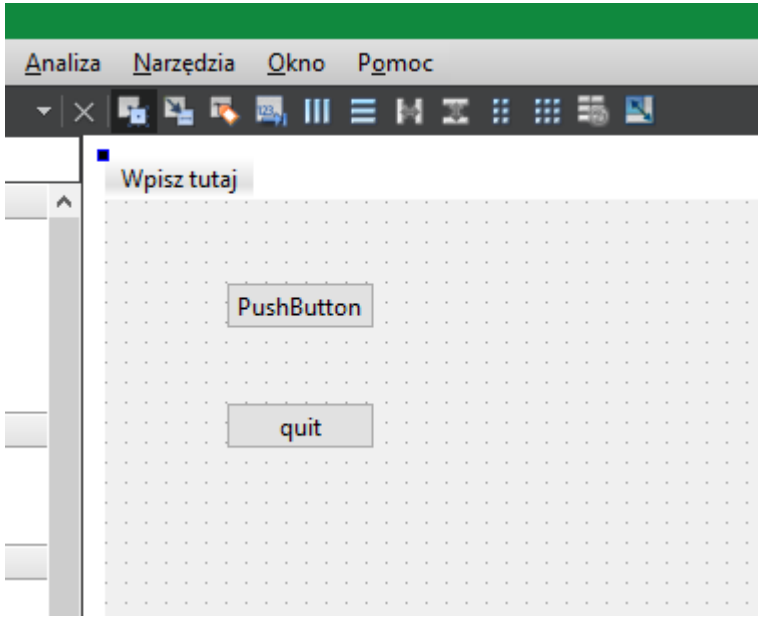
przeciągamy lewym klawiszem myszy w dół zaczynając od przycisku

helloWorld


mainwindow.ui @ helloWorld - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
mainwindow.ui*
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
Filtr
Layouts
Vertical Layout
Horizontal Layout
Grid Layout
Form Layout
Spacers
Horizontal Spacer
Vertical Spacer
Buttons
Push Button
Tool Button
Radio Button
Check Box
Command Link Button
Dialog Button Box
Item Views (Model-Based)
List View
Tree View
Table View
Column View
Undo View
Item Widgets (Item-Based)
List Widget
Tree Widget
Table Widget
Containers
Wpisz tutaj
PushButton
clicked()
close()
graficzne przedstawienie połączenia sygnału
clicked() i slotu close()

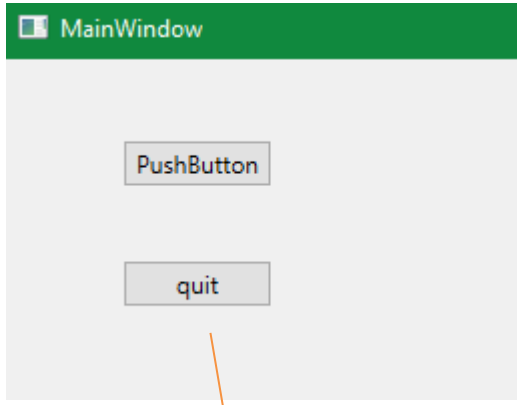
helloWorld





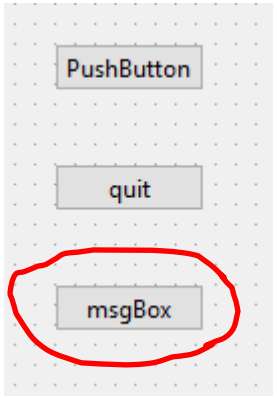
powrót do trybu modyfikowania widżeta

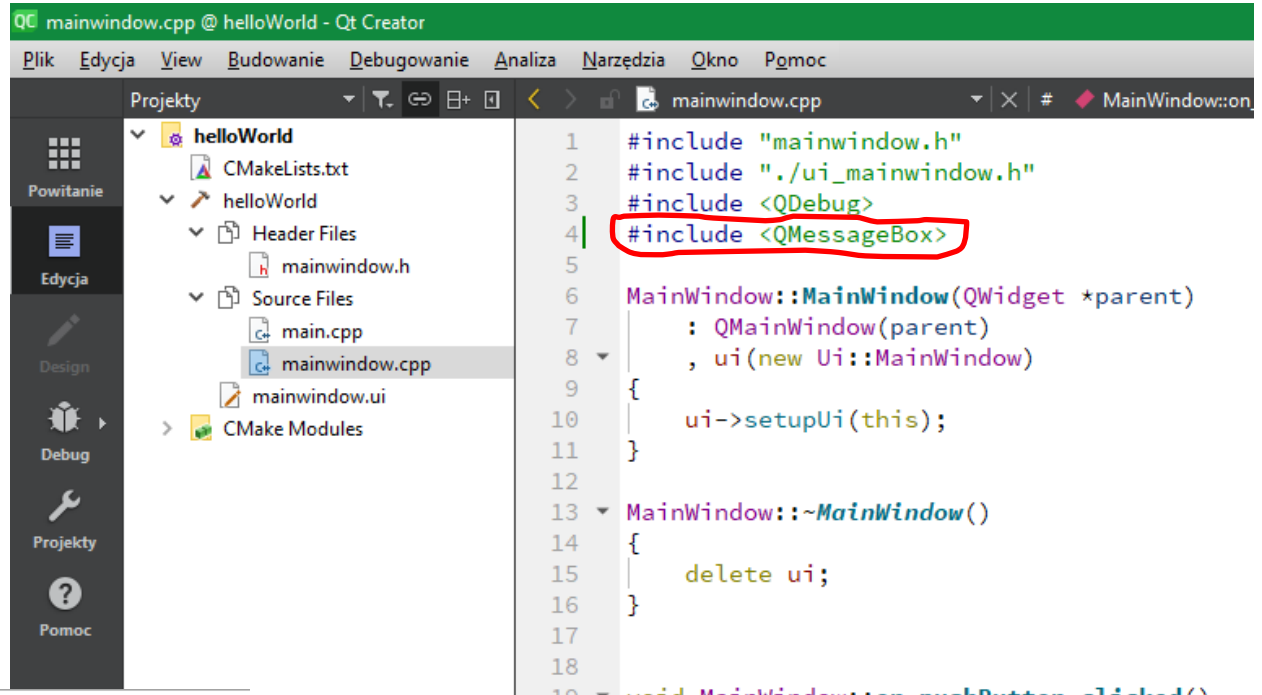
helloWorld

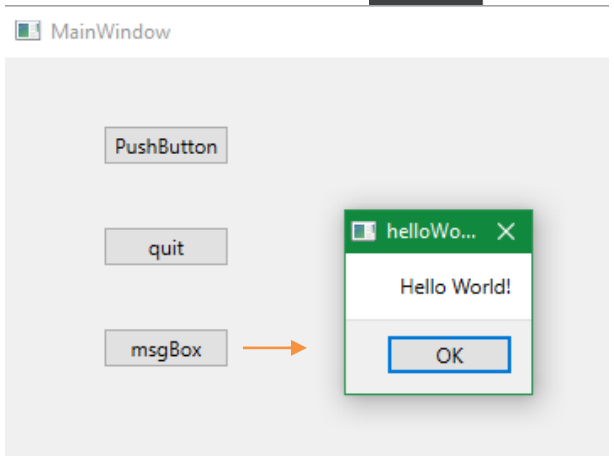


po naciśnięciu na przycisk *quit* następuje zamkniecie aplikacji

helloWorld

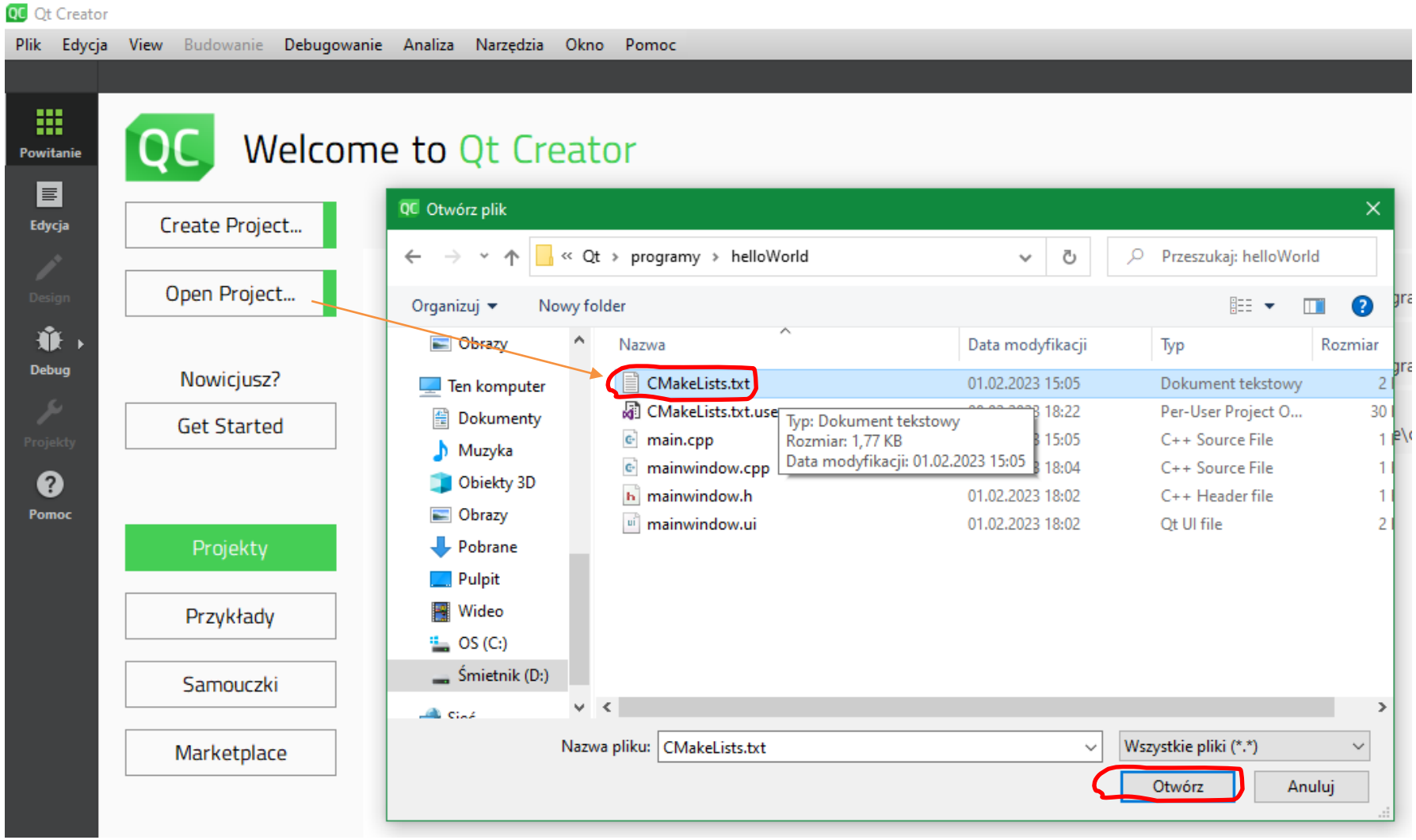






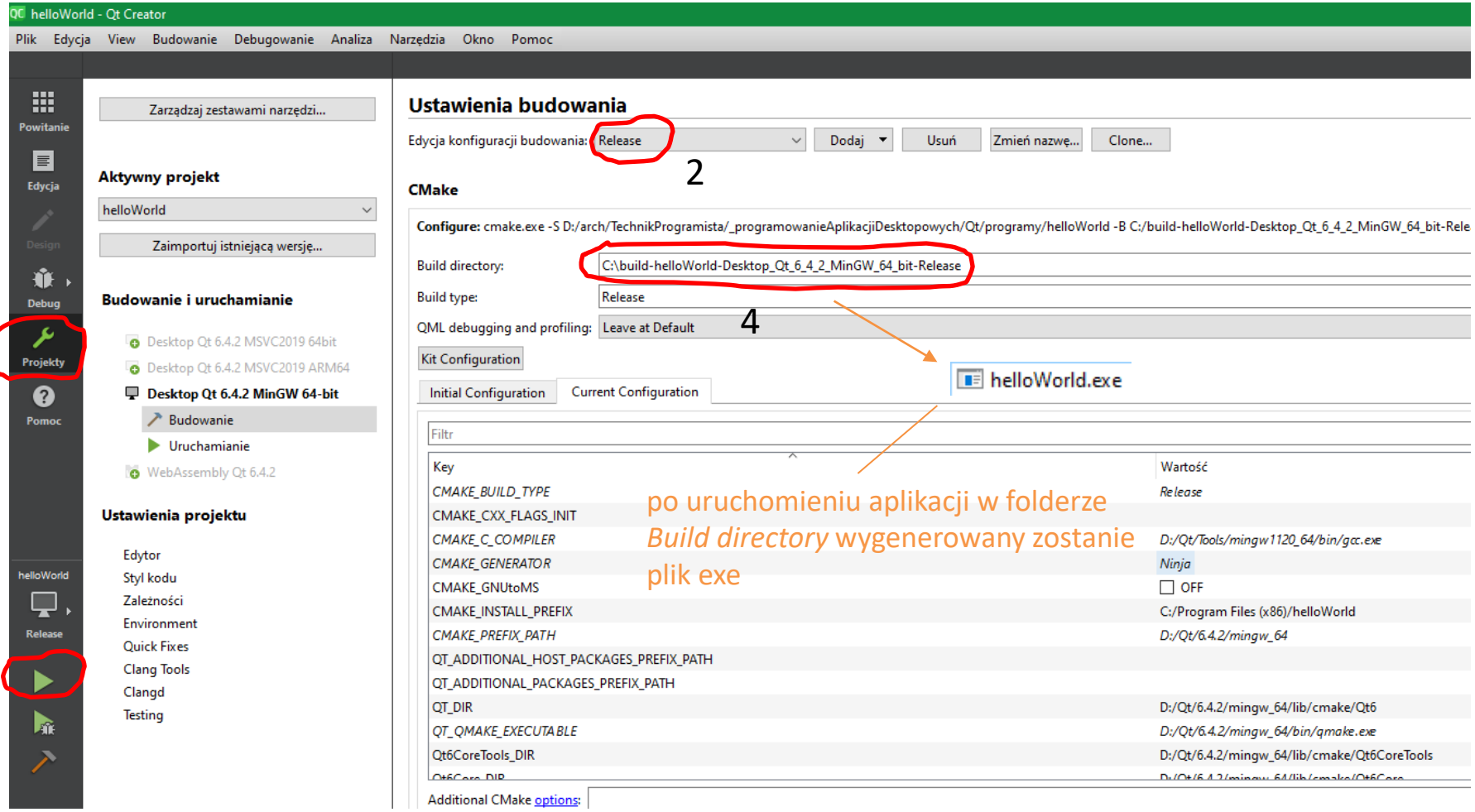
https://doc.qt.io/qt-6/qmessagebox.html

helloWorld
otwarcie zapisanego projektu



aby otworzyć zapisany projekt wybieramy plik CMakeLists.txt

deployment


helloWorld - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
helloWorld
Release
Zarządzaj zestawami narzędzi...
Aktywny projekt
helloWorld
Zaimportuj istniejącą wersję...
Budowanie i uruchamianie
Desktop Qt 6.4.2 MSVC2019 64bit
Desktop Qt 6.4.2 MSVC2019 ARM64
Desktop Qt 6.4.2 MinGW 64-bit
Budowanie
Uruchamianie
WebAssembly Qt 6.4.2
Ustawienia projektu
Edytor
Styl kodu
Zależności
Environment
Quick Fixes
Clang Tools
Clangd
Testing
Ustawienia budowania
Edycja konfiguracji budowania: Release
Dodaj
Usuń
Zmień nazwę...
Clone...
CMake
Configure: cmake.exe -S D:/arch/TechnikProgramista/_programowanieAplikacjiDesktopowych/Qt/programy/helloWorld -B C:/build-helloWorld-Desktop_Qt_6_4_2_MinGW_64_bit-Rele
Build directory: C:\build-helloWorld-Desktop_Qt_6_4_2_MinGW_64_bit-Release
Build type: Release
QML debugging and profiling: Leave at Default
Kit Configuration
Initial Configuration
Current Configuration
helloWorld.exe
Filtr
Key
Wartość
CMAKE_BUILD_TYPE
Release
CMAKE_CXX_FLAGS_INIT
CMAKE_C_COMPILER
D:/Qt/Tools/mingw1120_64/bin/gcc.exe
CMAKE_GENERATOR
Ninja
CMAKE_GNUtoMS
OFF
CMAKE_INSTALL_PREFIX
C:/Program Files (x86)/helloWorld
CMAKE_PREFIX_PATH
D:/Qt/6.4.2/mingw_64
QT_ADDITIONAL_HOST_PACKAGES_PREFIX_PATH
QT_ADDITIONAL_PACKAGES_PREFIX_PATH
QT_DIR
D:/Qt/6.4.2/mingw_64/lib/cmake/Qt6
QT_QMAKE_EXECUTABLE
D:/Qt/6.4.2/mingw_64/bin/qmake.exe
Qt6CoreTools_DIR
D:/Qt/6.4.2/mingw_64/lib/cmake/Qt6CoreTools
Additional CMake options:
1
2
3
4
po uruchomieniu aplikacji w folderze
Build directory wygenerowany zostanie
plik exe

deployment

najpierw przenosimy plik helloWorld.exe, pliki dll oraz pluginy do jednego folderu *Deployment*

Initial deployment (Quick and dirty)

1. Close Qt Creator.
2. Copy the following into C:\Deployment\
    1. The *release version* of **MyApp.exe**
    2. All the **.dll files** from C:\Qt\5.2.1\mingw48_32\bin\
    3. All the **folders** from C:\Qt\5.2.1\mingw48_32\plugins\
    4. (If you used QML) All the **folders** from C:\Qt\5.2.1\mingw48_32\qml\
3. Rename C:\Qt\ to C:\QtHidden\ (This turns your PC into a clean environment, just like one that doesn't have Qt installed).
4. Launch C:\Deployment\MyApp.exe.

If your app worked correctly, congratulations! You are almost ready for deployment. You don't want to ship a 1.5GB package though, so it's time to clean up unused files.

If it didn't work correctly, ask for help (see the Appendix)

Final deployment (Cleaned up)

Do the deletion steps below in C:\Deployment\ and all of its subdirectories. After each deletion, launch C:\Deployment\MyApp.exe and test it. If it stops working, restore the files you just deleted.

1. Launch MyApp.exe. While it is running, try to delete all DLLs. The DLLs that aren't used will go to the recycle bin, leaving behind only the DLLs that you need. (This trick doesn't work for .qml and qmldir files, however).
2. (If you used QML) Delete a few .qml files and try relaunching MyApp.exe. Repeat until you try all .qml files.
3. (If you used QML) Delete *qmldir* files from the folders that have no more DLLs or .qml files

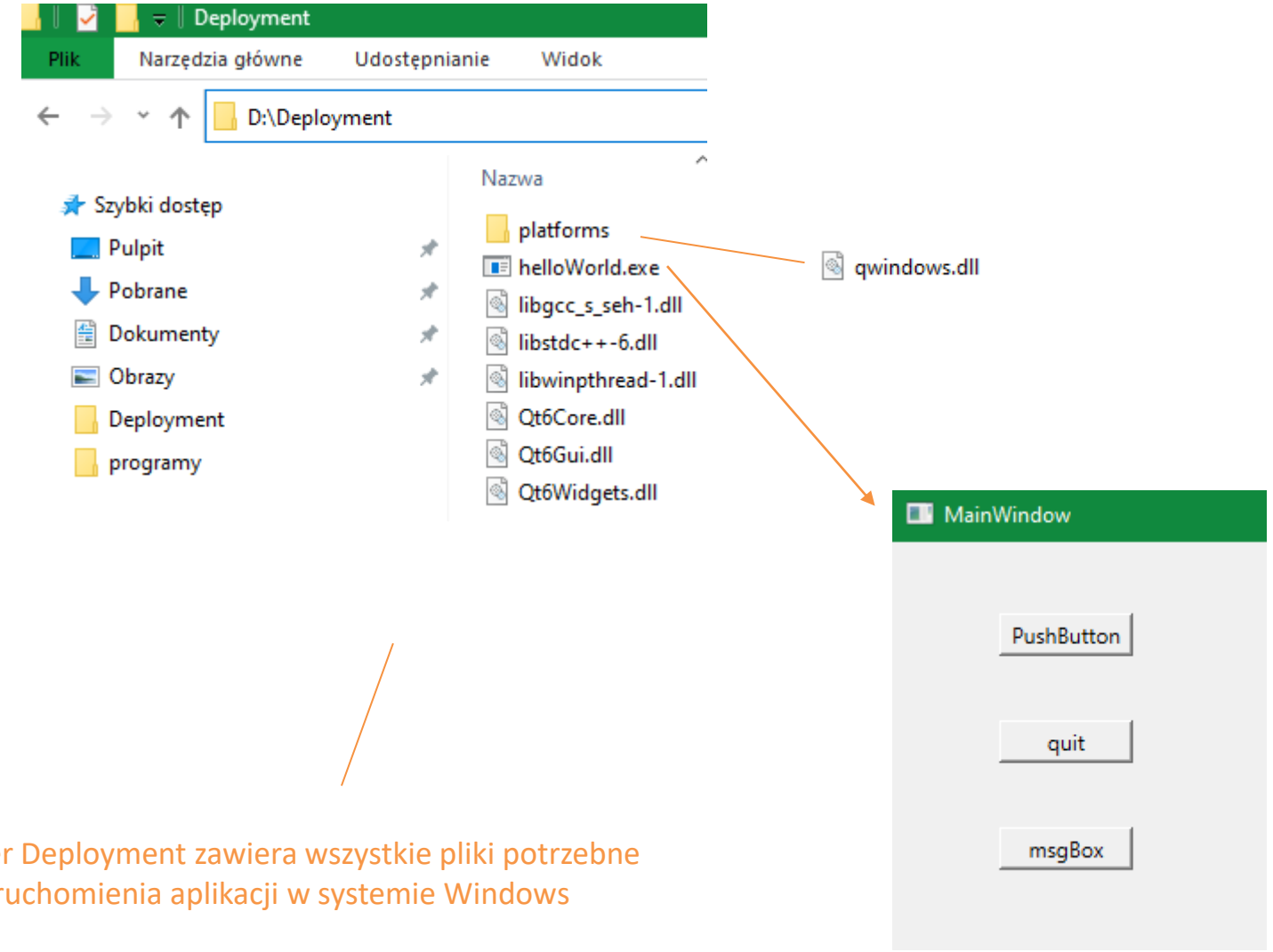
When you have removed all the files that you don't need,

1. Rename C:\QtHidden\ back to C:\Qt\ to restore your installation.
2. Distribute your app.

następnie uruchamiamy plik exe i usuwamy wszystkie pliki, które są niepotrzebne (system nie pozwoli usunąć plików, które są potrzebne)

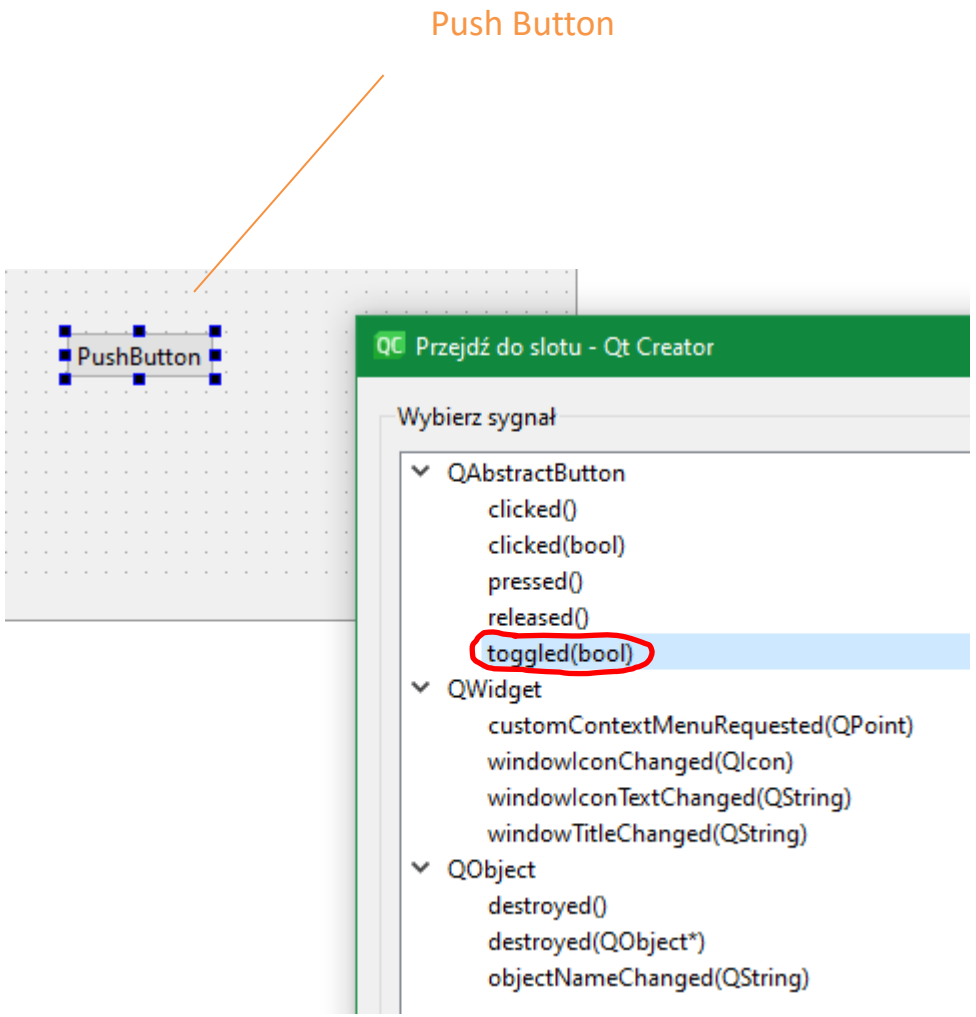
https://wiki.qt.io/Deploy_an_Application_on_Windows

deployment

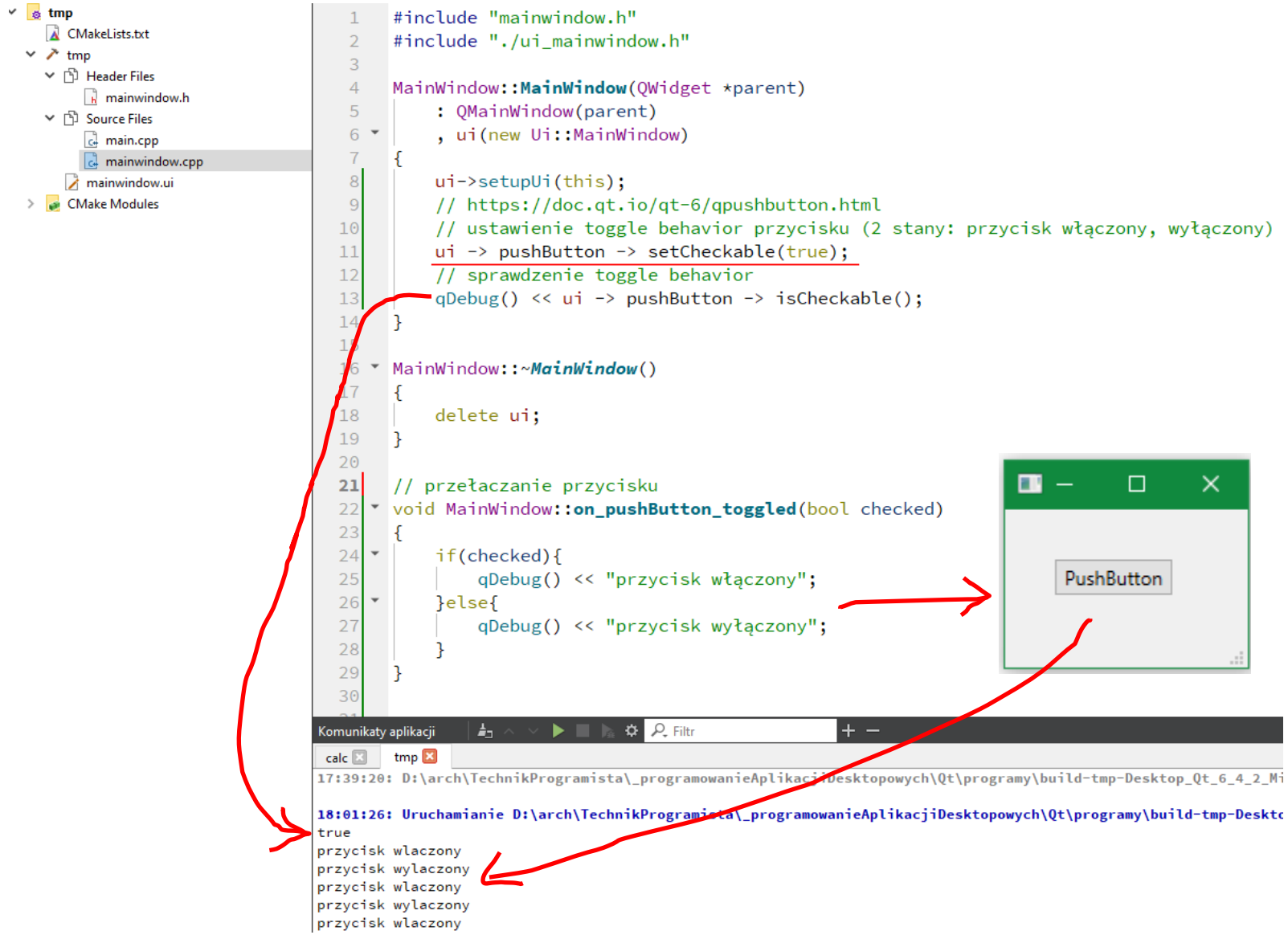


folder Deployment zawiera wszystkie pliki potrzebne
do uruchomienia aplikacji w systemie Windows

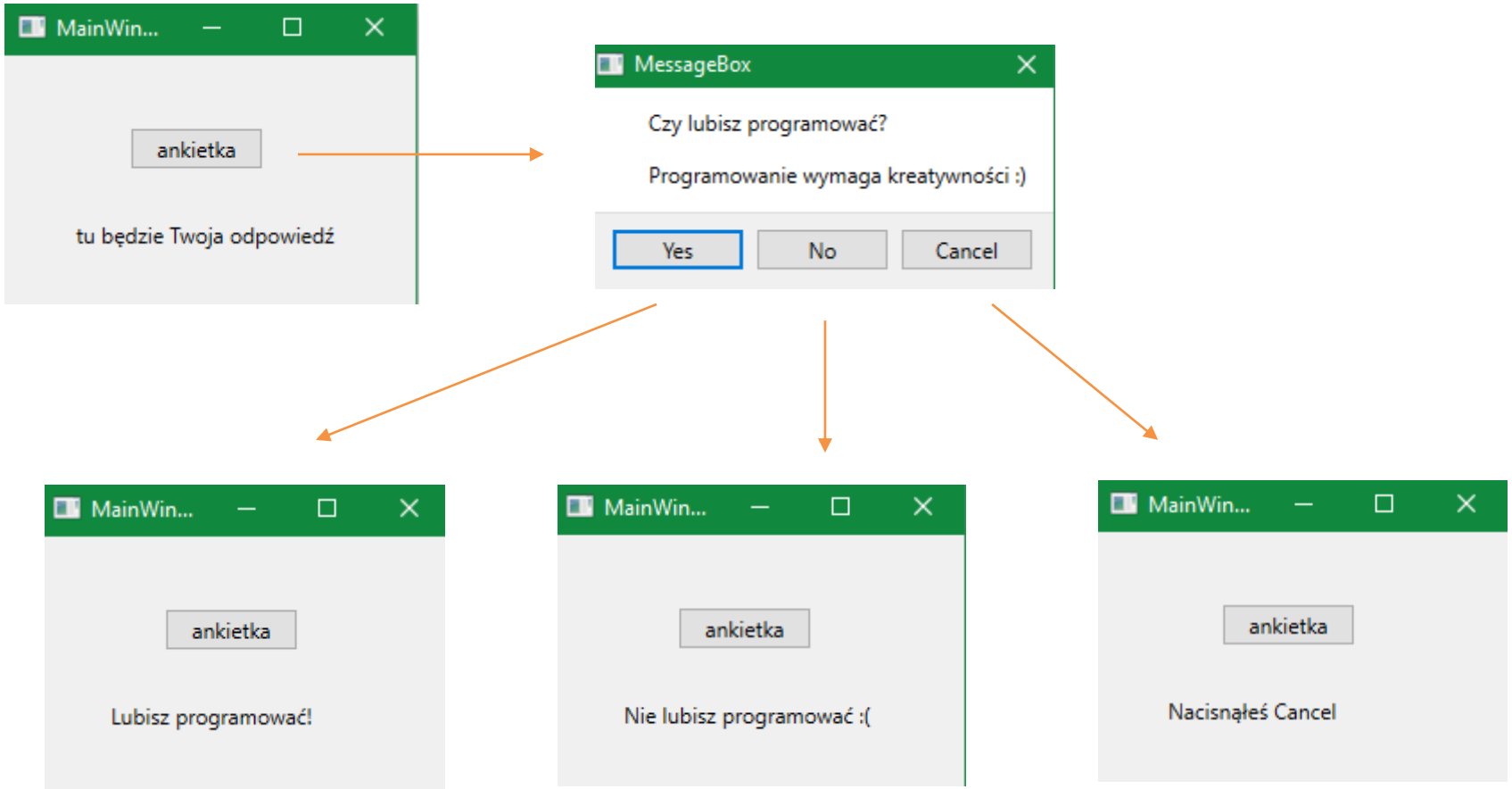
toogle button



toogle button

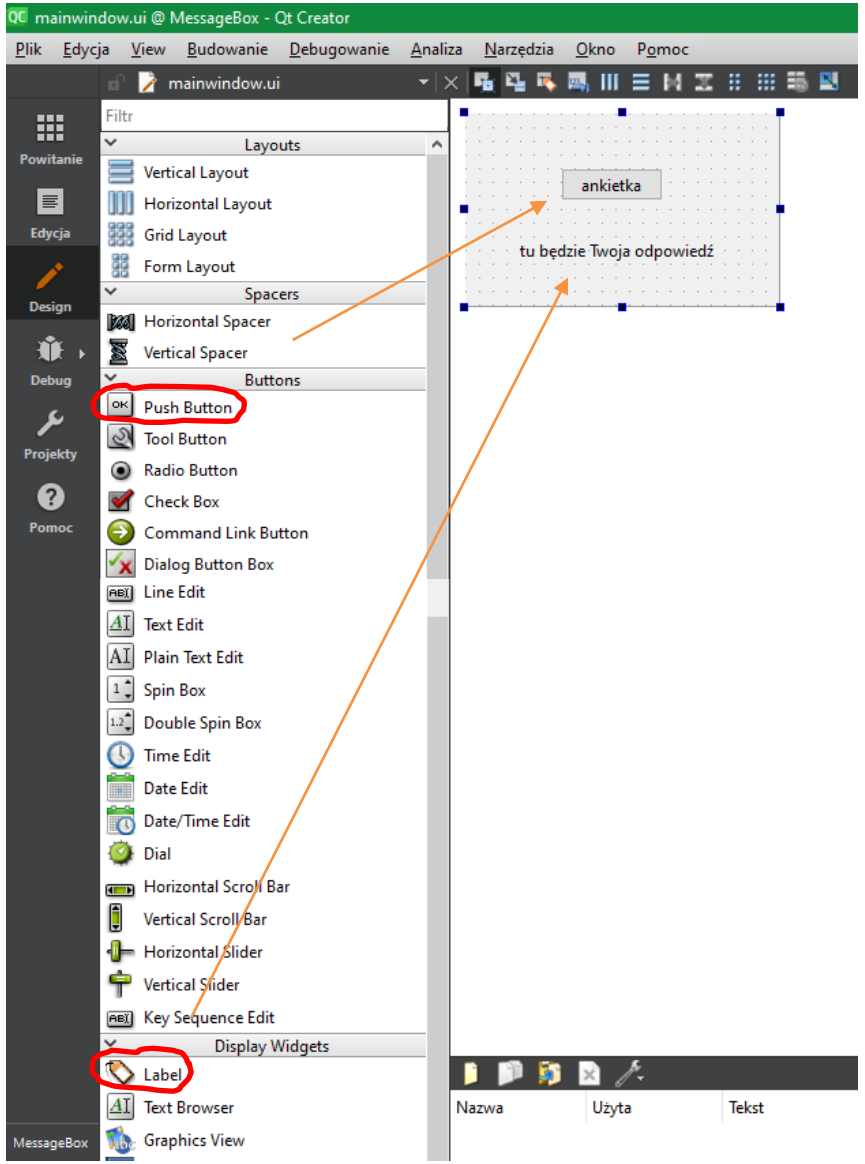


MessageBox



https://doc.qt.io/qt-6/qmessagebox.html

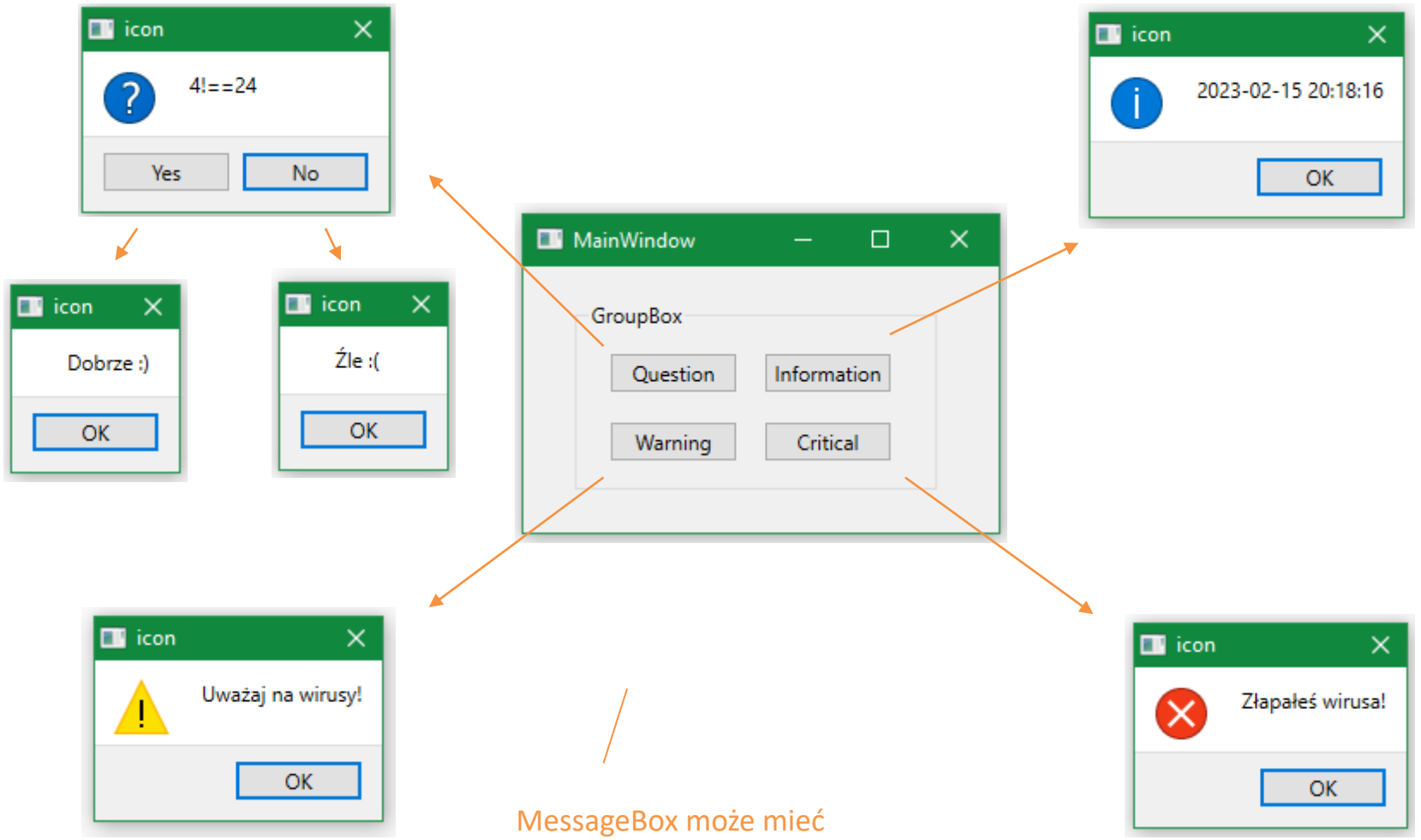
MessageBox


mainwindow.ui @ MessageBox - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
mainwindow.ui
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
MessageBox
Filtr
Layouts
Vertical Layout
Horizontal Layout
Grid Layout
Form Layout
Spacers
Horizontal Spacer
Vertical Spacer
Buttons
Push Button
Tool Button
Radio Button
Check Box
Command Link Button
Dialog Button Box
Line Edit
Text Edit
Plain Text Edit
Spin Box
Double Spin Box
Time Edit
Date Edit
Date/Time Edit
Dial
Horizontal Scroll Bar
Vertical Scroll Bar
Horizontal Slider
Vertical Slider
Key Sequence Edit
Display Widgets
Label
Text Browser
Graphics View
ankietka
tu będzie Twoja odpowiedź
Nazwa
Użyta
Tekst

MessageBox

```
void MainWindow::on_pushButton_clicked()
{
    QMessageBox msgBox;
    msgBox.setText("Czy lubisz programować?");
    msgBox.setInformativeText("Programowanie wymaga kreatywności :)");
    msgBox.setStandardButtons(QMessageBox::Yes | QMessageBox::No | QMessageBox::Cancel);
    msgBox.setDefaultButton(QMessageBox::Yes);
    int ret = msgBox.exec();
    switch (ret) {
      case QMessageBox::Yes:
          ui->label->setText("Lubisz programować!");
          break;
      case QMessageBox::No:
          ui->label->setText("Nie lubisz programować :(");
          break;
      case QMessageBox::Cancel:
          ui->label->setText("Nacisnąłeś Cancel");
          break;
      default:
          // should never be reached
          break;
    }
}
```

MessageBox icon



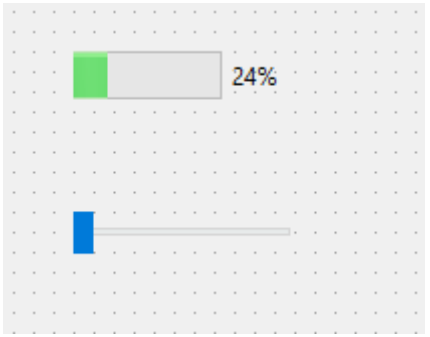
MessageBox może mieć jedną z 4 ikonek: *Question, Information, Warning, Critical*

https://doc.qt.io/qt-6/qmessagebox.html#Icon-enum

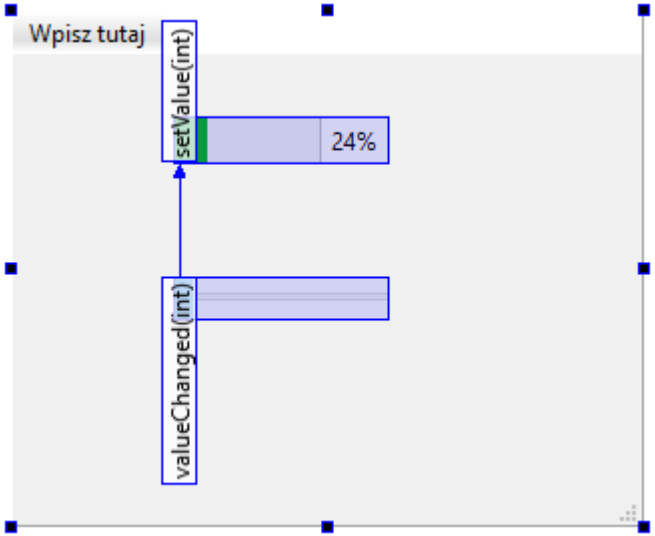
Progress Bar

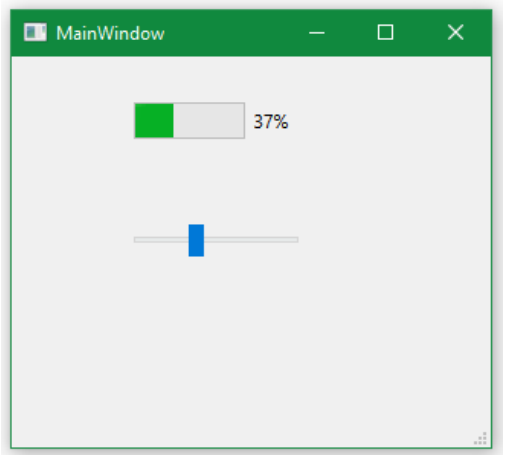
Progress bar

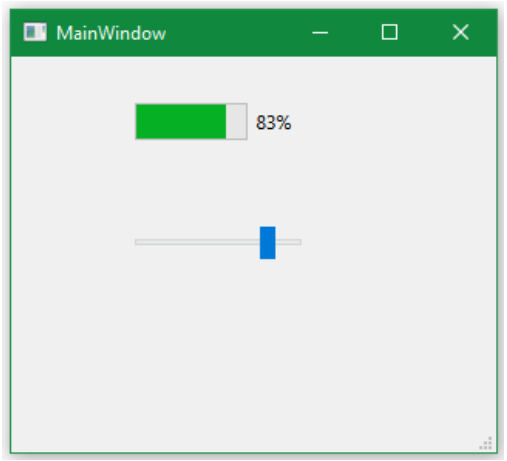
Horizontal Slider



łączymy sygnał ze Slidera ze slotem w Progress Bar



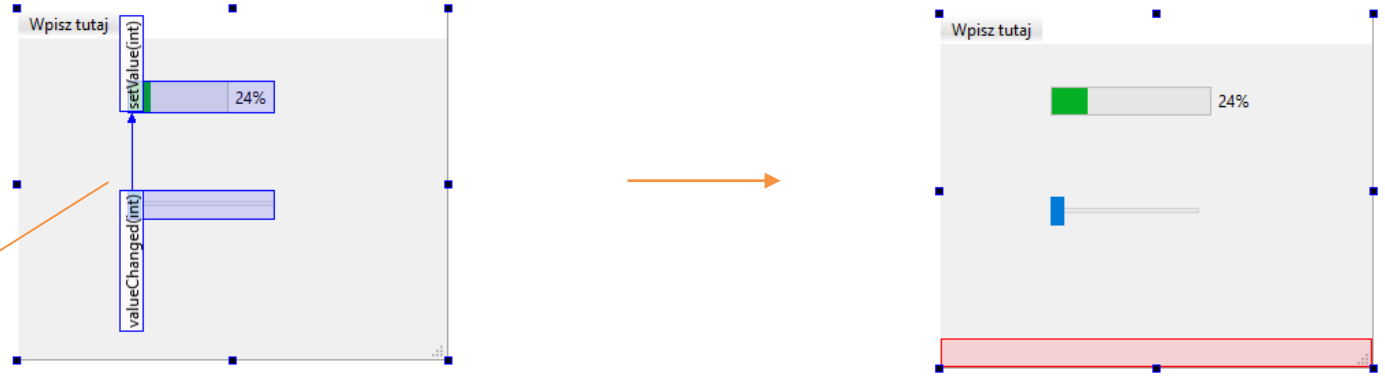




[youtube]

wielkość przesunięcia suwaka jest odwzorowana w pasku postępu

Progress Bar



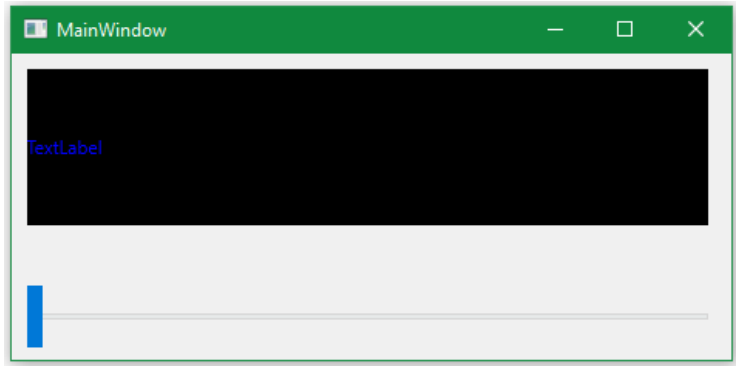
usuwamy połączenie

wpisujemy połączenie za pomocą kodu (działa tak samo)

rozłączenie sygnału ze slotem za pomocą kodu

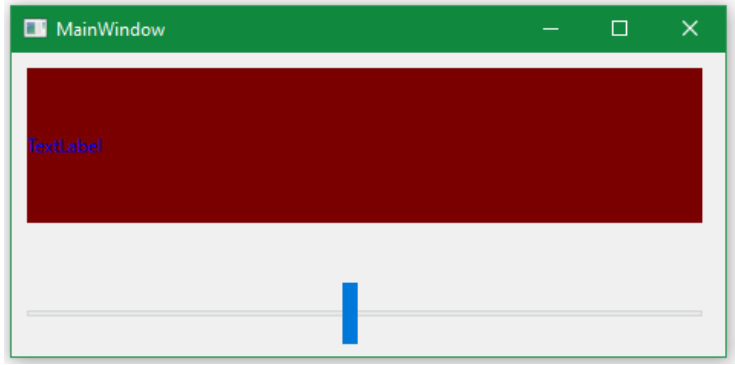
youtube

Slider



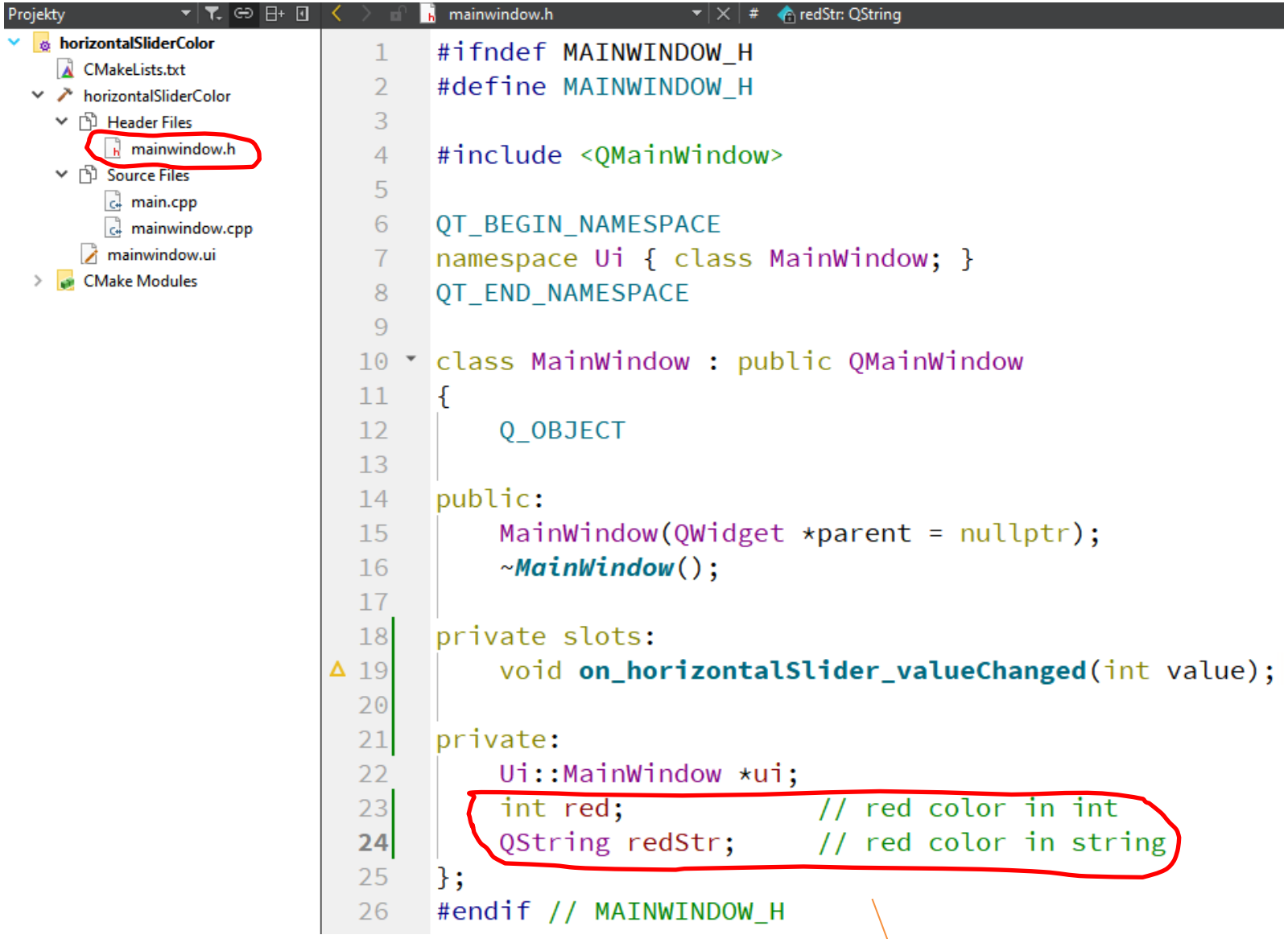
Label

Horizontal Slider (red)



kolor labela jest uzależniony
od położenia suwaka
(tylko składnik czerwony koloru
ulega zmianie)

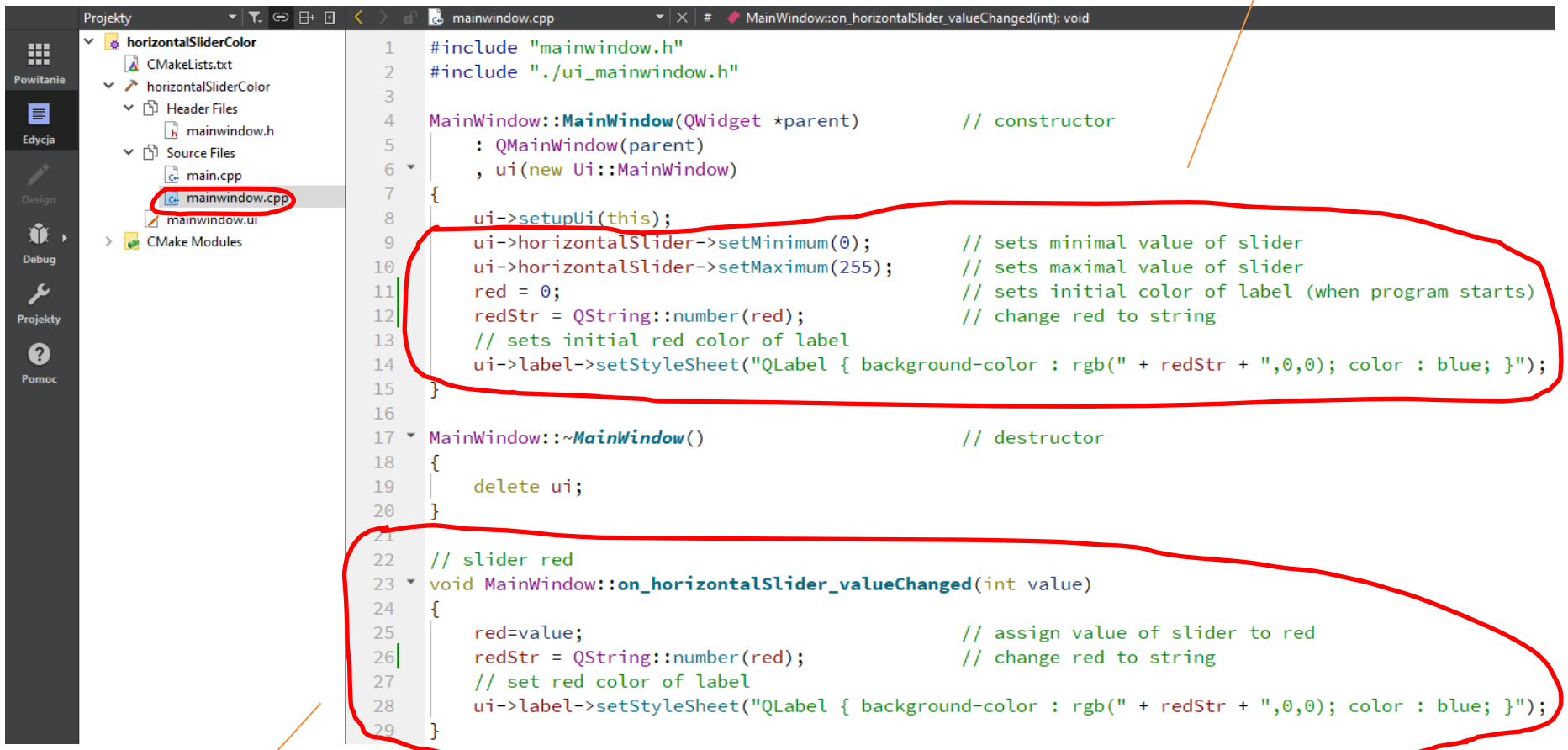
Slider



pola klasy MainWindow przeznaczone na wartości koloru czerwonego

Slider

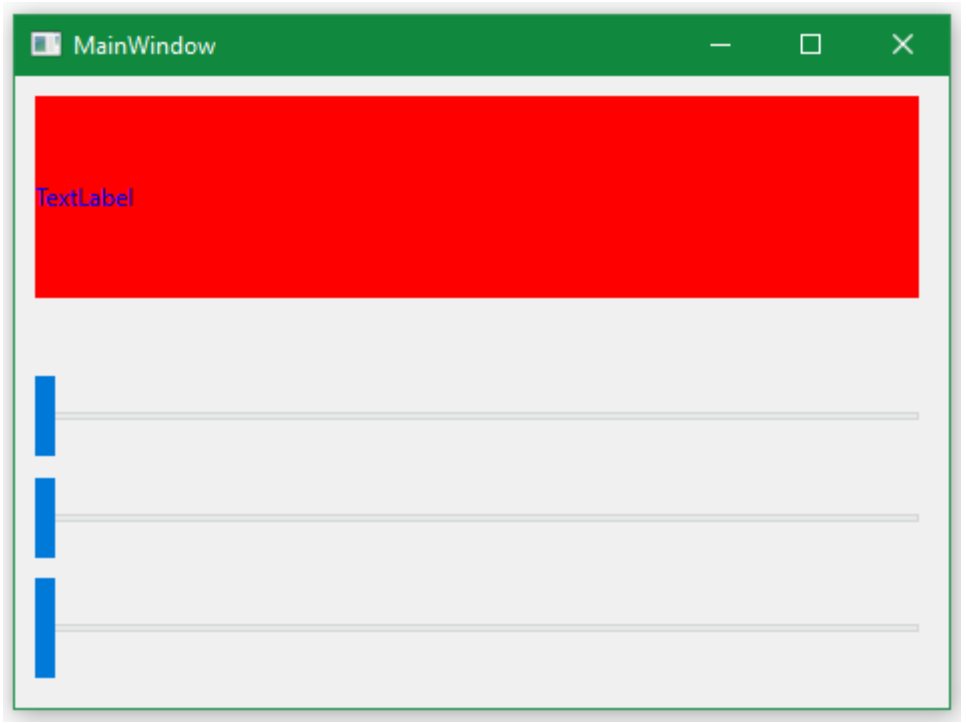
ustawienie zakresu czerwonego slidera
oraz wartości początkowej koloru czerwonego
w konstruktorze okna



zmiana wartości koloru czerwonego za pomocą slidera
(w slocie on_horizontalSlider_valueChanged()
połączonym z sygnałem valueChanged()
generowanym przez slider)

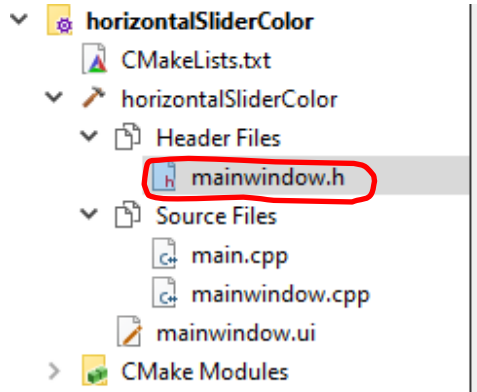
https://stackoverflow.com/questions/2749798/qlabel-set-color-of-text-and-background

Slidery RGB



Dołącz slidery zmieniające kolory green oraz blue

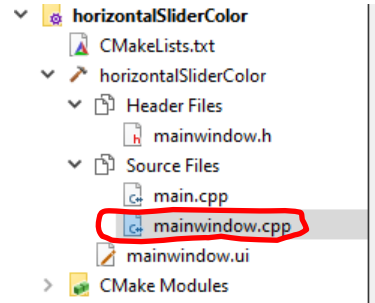
Slidery RGB



pola klasy MainWindow przeznaczone na wartości kolorów RGB

```
1  #ifndef MAINWINDOW_H
2  #define MAINWINDOW_H
3
4  #include <QMainWindow>
5
6  QT_BEGIN_NAMESPACE
7  namespace Ui { class MainWindow; }
8  QT_END_NAMESPACE
9
10 class MainWindow : public QMainWindow
11 {
12     Q_OBJECT
13
14 public:
15     MainWindow(QWidget *parent = nullptr);
16     ~MainWindow();
17
18 private slots:
19     void on_horizontalSlider_valueChanged(int value);
20
21     void on_horizontalSlider_2_valueChanged(int value);
22
23     void on_horizontalSlider_3_valueChanged(int value);
24
25 private:
26     Ui::MainWindow *ui;
27     int red;              // red color in int
28     QString redStr;       // red color in string
29     int green;            // green color in int
30     QString greenStr;     // green color in string
31     int blue;             // blue color in int
32     QString blueStr;      // blue color in string
33 };
34 #endif // MAINWINDOW_H
```

Slidery RGB

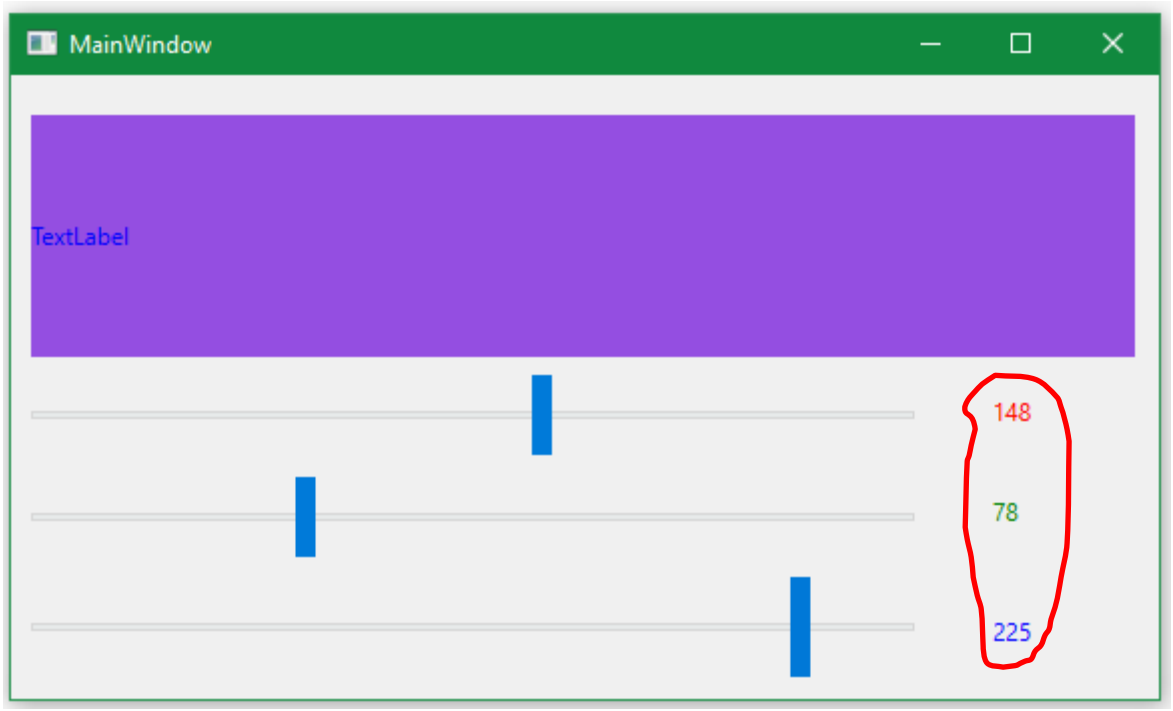


ustawienie zakresów sliderów
oraz wartości początkowej kolorów
w konstruktorze okna

zmiana wartości
kolorów RGB
za pomocą sliderów

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

MainWindow::MainWindow(QWidget *parent)          // constructor
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->horizontalSlider->setMinimum(0);         // sets minimal value of slider
    ui->horizontalSlider->setMaximum(255);       // sets maximal value of slider
    ui->horizontalSlider_2->setMinimum(0);
    ui->horizontalSlider_2->setMaximum(255);
    ui->horizontalSlider_3->setMinimum(0);
    ui->horizontalSlider_3->setMaximum(255);
    red = 0;                                     // sets initial color of label (when program starts)
    redStr = QString::number(red);               // change red to string
    green = 0;
    greenStr = QString::number(green);
    blue = 0;
    blueStr = QString::number(blue);
    // sets initial red color of label
    ui->label->setStyleSheet("QLabel { background-color : rgb(" + redStr + "," + greenStr + "," + blueStr + "); color : blue; }");
}

MainWindow::~MainWindow()                        // destructor
{
    delete ui;
}

// slider red
void MainWindow::on_horizontalSlider_valueChanged(int value)
{
    red=value;                                   // assign value of slider to red
    redStr = QString::number(red);               // change red to string
    // set red color of label
    ui->label->setStyleSheet("QLabel { background-color : rgb(" + redStr + "," + greenStr + "," + blueStr + "); color : blue; }");
}



// slider green
void MainWindow::on_horizontalSlider_2_valueChanged(int value)
{
    green=value;                                 // assign value of slider to green
    greenStr = QString::number(green);           // change green to string
    // set green color of label
    ui->label->setStyleSheet("QLabel { background-color : rgb(" + redStr + "," + greenStr + "," + blueStr + "); color : blue; }");

}

// slider blue
void MainWindow::on_horizontalSlider_3_valueChanged(int value)
{
    blue=value;                                  // assign value of slider to blue
    blueStr = QString::number(blue);             // change blue to string
    // set blue color of label
    ui->label->setStyleSheet("QLabel { background-color : rgb(" + redStr + "," + greenStr + "," + blueStr + "); color : blue; }");

}
```

Slidery RGB

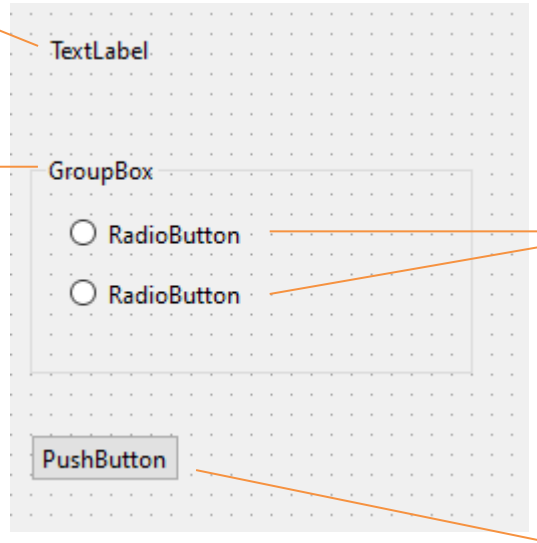


wstaw labele z wartościami kolorów RGB

QRadioButton

wstaw zagadkę,
zwiększ rozmiar czcionki

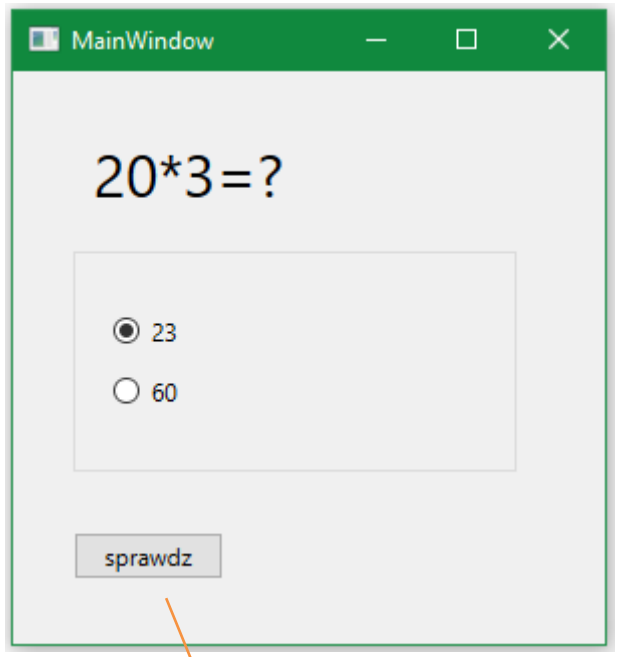
usuń napis

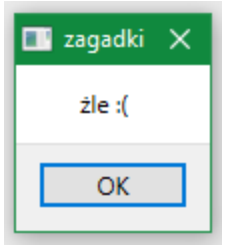


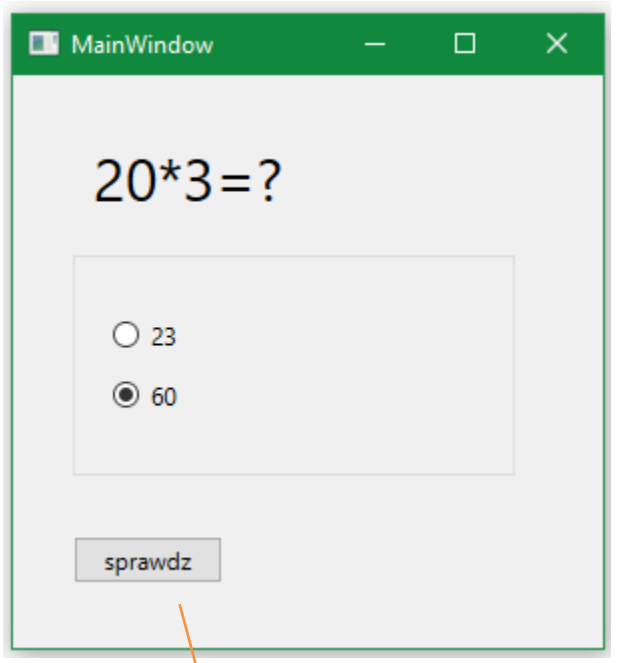
wstaw odpowiedzi

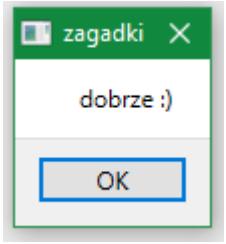
zmień napis, sprawdź odpowiedź,
wyświetl QMessageBox z wynikiem

QRadioButton









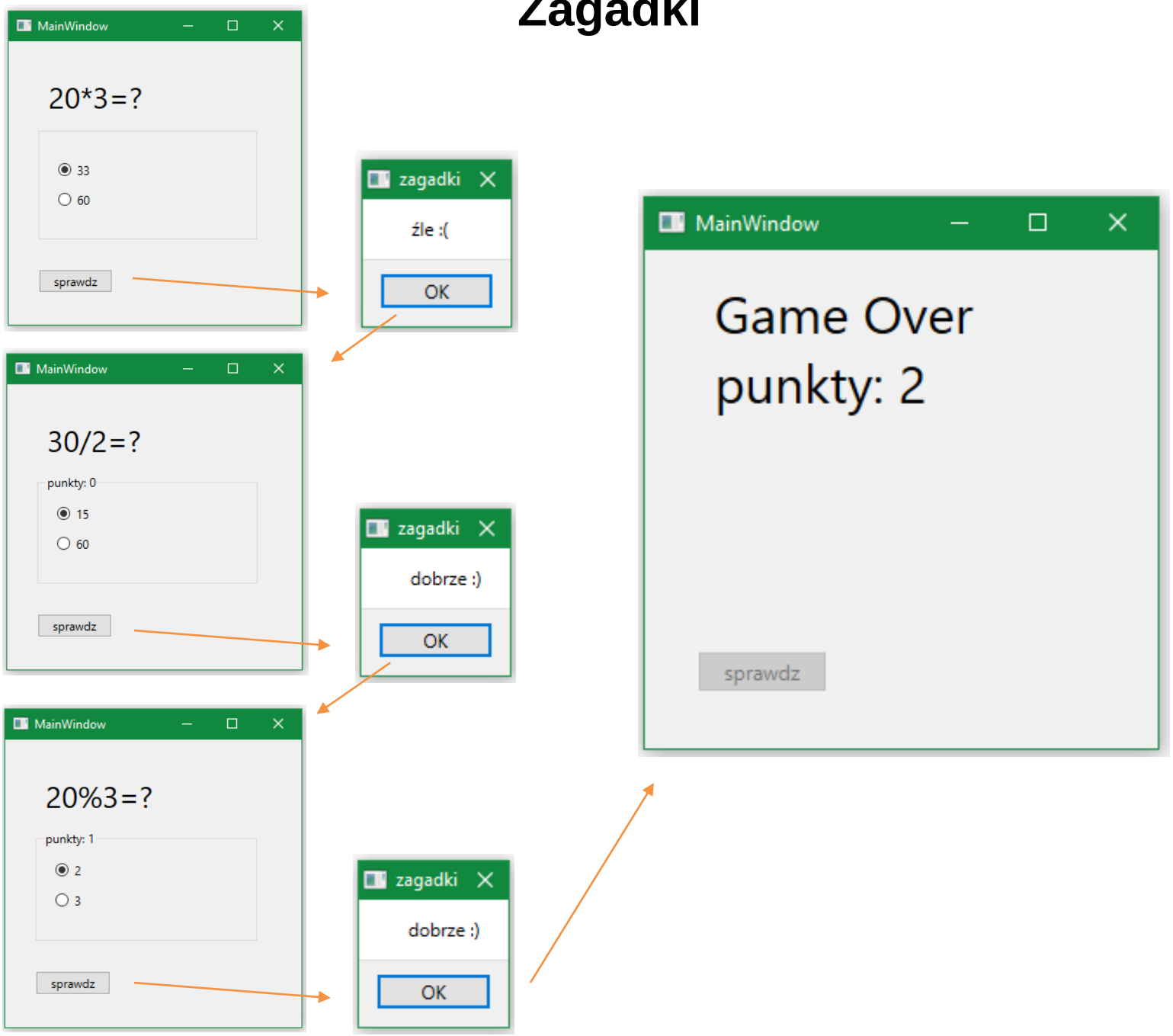
https://codeloop.org/qt5-gui-how-to-create-qradiobutton/

QRadioButton

mainwindow.cpp

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

#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent) // konstruktor
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->radioButton->setChecked(false);      // zaznacza pierwszy radiobutton
    ui->pushButton->setText("sprawdz");      // ustawia tekst na przycisku
    ui->radioButton->setText("23");          // odpowiedź 1
    ui->radioButton_2->setText("60");        // odpowiedź 2
    ui->groupBox->setTitle("");              // usuwa tytuł z groupBox
    ui->label->setText("20*3=?");            // ustawia text zagadki
    ui->label->setStyleSheet("QLabel {font-size: 30px}");  // ustawia styl zagadki
}

MainWindow::~MainWindow()
{
    delete ui;
}

// sprawdzenie odpowiedzi użytkownika
void MainWindow::on_pushButton_clicked()
{
    QMessageBox msgBox;
    if (ui->radioButton->isChecked()){
        msgBox.setText("źle :(");
        msgBox.exec();
    }else if (ui->radioButton_2->isChecked()){
        msgBox.setText("dobrze :)");
        msgBox.exec();
    }
}
```

Zagadki


MainWindow
20*3=?
33
60
sprawdz
zagadki
źle :(
OK
MainWindow
30/2=?
punkty: 0
15
60
sprawdz
zagadki
dobrze :)
OK
MainWindow
20%3=?
punkty: 1
2
3
sprawdz
zagadki
dobrze :)
OK
MainWindow
Game Over
punkty: 2
sprawdz

Zagadki

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

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

private slots:
    void on_pushButton_clicked();

private:
    Ui::MainWindow *ui;
    const static int iloscZagadek = 3;
    QString zagadki[iloscZagadek] = {"20*3=?","30/2=?","20%3=?"}; // tablica zagadek
    int i = 0;      // index tablic
    int punkty = 0; // punkty
    QString rbText[iloscZagadek] = {"33","15","2"};  //  radioButton text
    QString rb2Text[iloscZagadek] = {"60","60","3"}; //  redioButton_2 text
    QString msgBoxRbText[iloscZagadek] = {"źle :(", "dobrze :)", "dobrze :)"}; // tekst wyświetlany w msgBox przy radioButton
    QString msgBoxRb2Text[iloscZagadek] = {"dobrze :)","źle :(","źle :("};     // tekst wyświetlany w msgBox przy radioButton_2

};
#endif // MAINWINDOW_H
```

tablice z informacjami tekstowymi
jako pola prywatne klasy MainWindow

Zagadki

mainwindow.cpp 1/2

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

#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent)     // konstruktor (uruchamia się na poczatku programu)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->radioButton->setChecked(true);      // zaznacza pierwszy radiobutton
    ui->pushButton->setText("sprawdz");     // ustawia tekst na przycisku
    ui->radioButton->setText(rbText[i]);    // ustawia początkową odpowiedź 1 (pierwszej zagadki)
    ui->radioButton_2->setText(rb2Text[i]); // ustawia początkową odpowiedź 2 (pierwszej zagadki)
    ui->groupBox->setTitle("");             // usuwa tytuł z groupBox
    ui->label->setText(zagadki[i]);         // ustawia text pierwszej zagadki
    ui->label->setStyleSheet("QLabel {font-size: 30px}");  // ustawia styl zagadki - wielkość fontóW
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

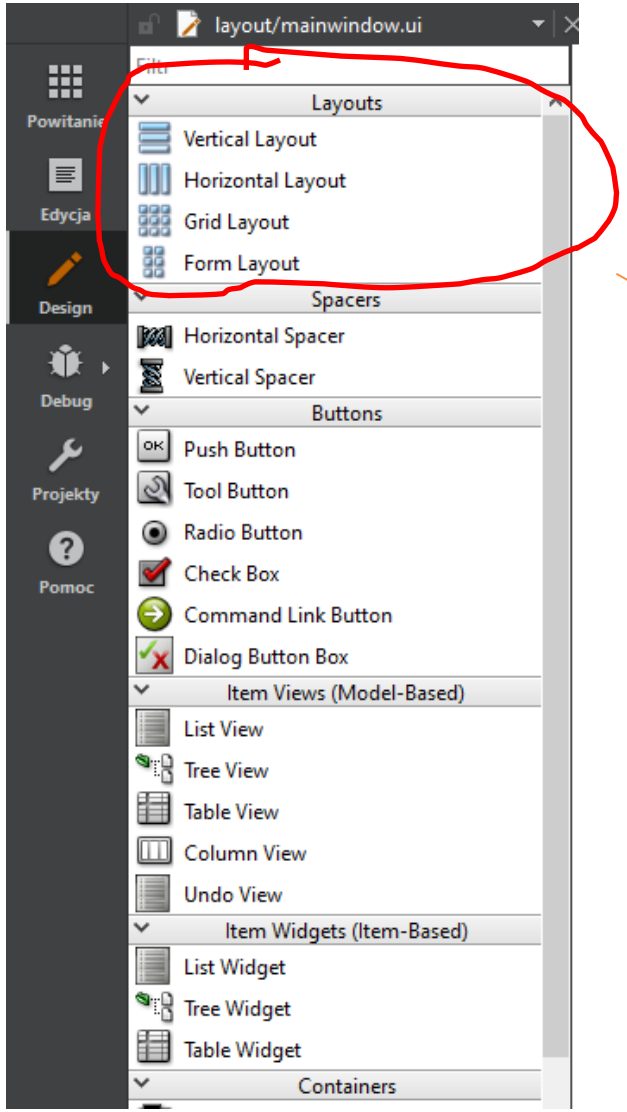
ustawienia początkowe programu

Zagadki

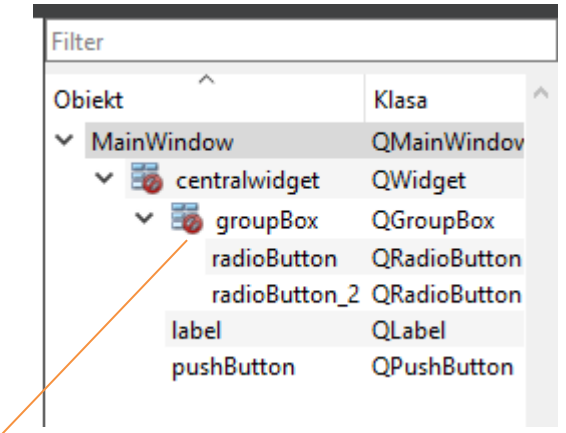
mainwindow.cpp 2/2

```
// sprawdza odpowiedzi użytkownika po kliknięciu na przycisk
void MainWindow::on_pushButton_clicked()
{
    QMessageBox msgBox;
    if (ui->radioButton->isChecked()){
        msgBox.setText(msgBoxRbText[i]);        // wyświetla w msgBox napis z tablicy msgBoxRbText[]
        msgBox.exec();
        if (msgBoxRbText[i] == "dobrze :)"){
            punkty++;
        }
    }else if (ui->radioButton_2->isChecked()){
        msgBox.setText(msgBoxRb2Text[i]);       // wyświetla w msgBox napis z tablicy msgBoxRb2Text[]
        msgBox.exec();
        if (msgBoxRb2Text[i] == "dobrze :)"){
            punkty++;
        }
    }
    i++;                                        // inkrementuje numer zagadki
    if (i >= iloscZagadek){                     // sprawdza czy zagadki się skończyły
        ui->label->setText("Game Over\npunkty: " + QString::number(punkty));  // wyswietla informację o zakończeniu gry
        ui->groupBox->setVisible(false);        // chowa groupBox (radioButtony)
        ui->pushButton->setEnabled(false);      // unieruchamia przycisk
    }else{
        ui->label->setText(zagadki[i]);         // ustawia text następnej zagadki
        ui->radioButton->setText(rbText[i]);    // ustawia odpowiedź 1 nastepnej zagadki
        ui->radioButton_2->setText(rb2Text[i]); // ustawia odpowiedź 2 nastepnej zagadki
        ui->groupBox->setTitle("punkty: " + QString::number(punkty)); // wyświetla punktację
    }
}
```

Layout



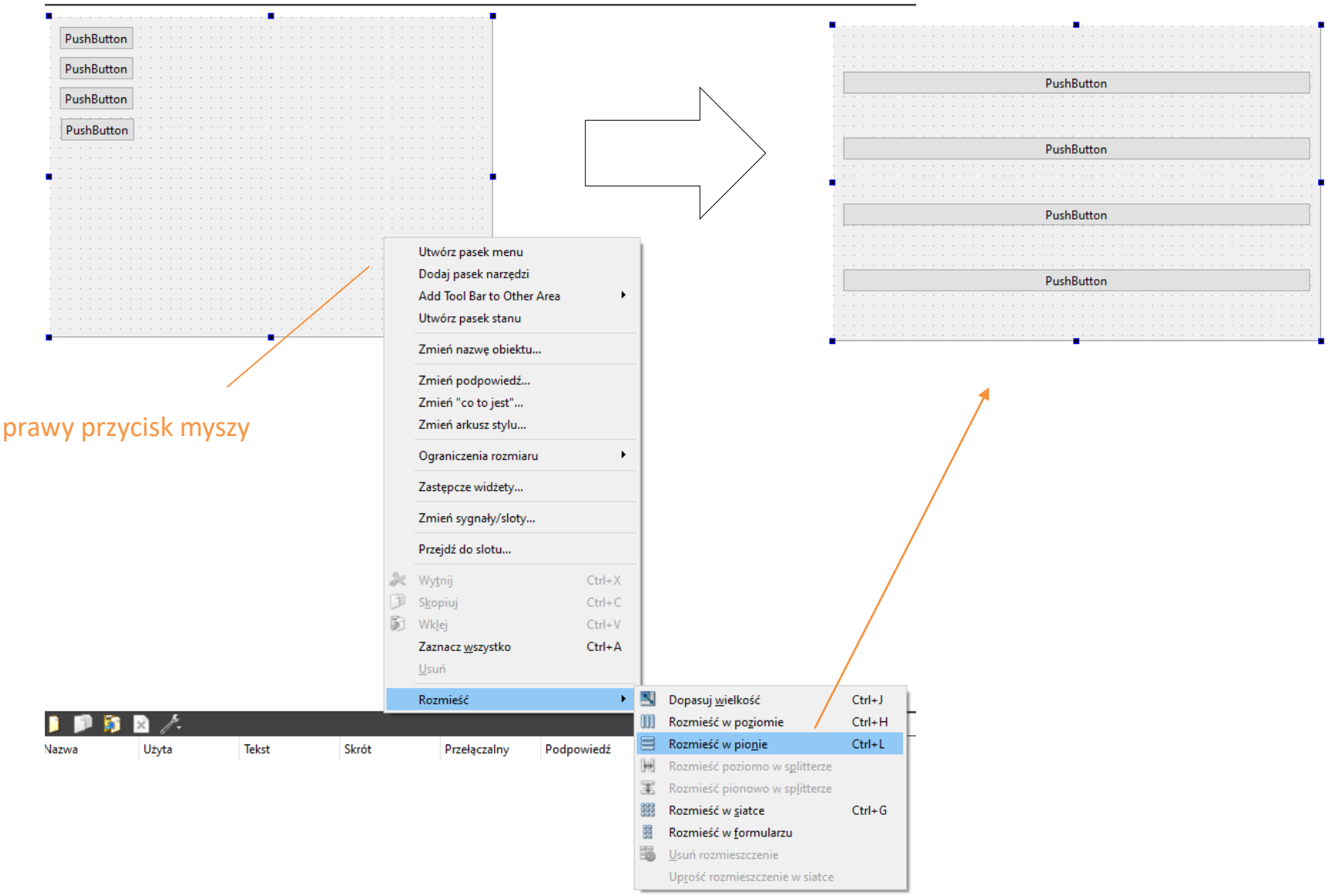
layouty do wyboru



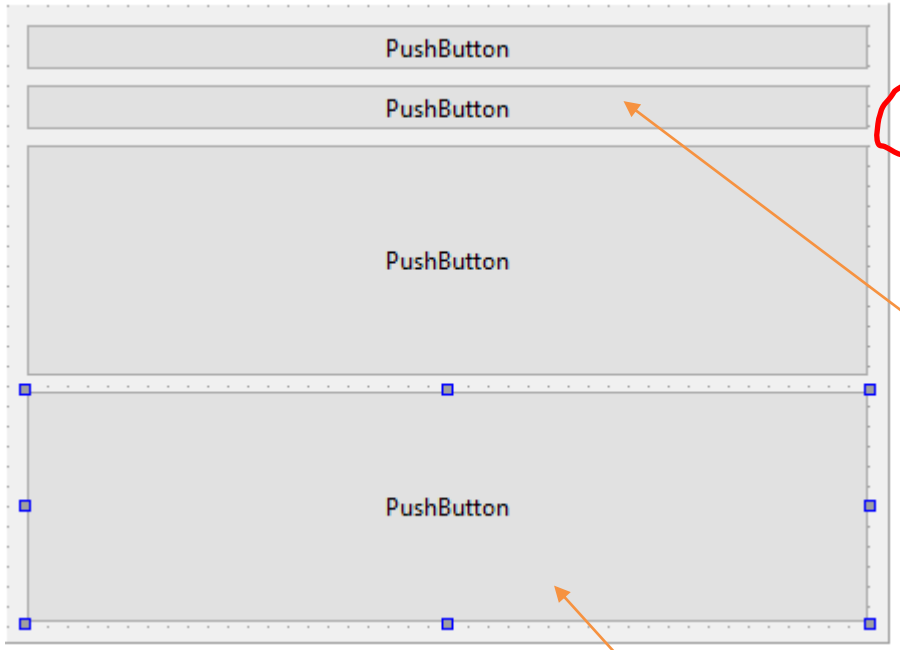
czerwony znaczek (kółeczko)
oznacza brak layoutu

https://www.youtube.com/watch?v=gWa2rqe8l6E

Vertical Layout



Vertical Layout



sizePolicy

pushButton_3 : QPushButton

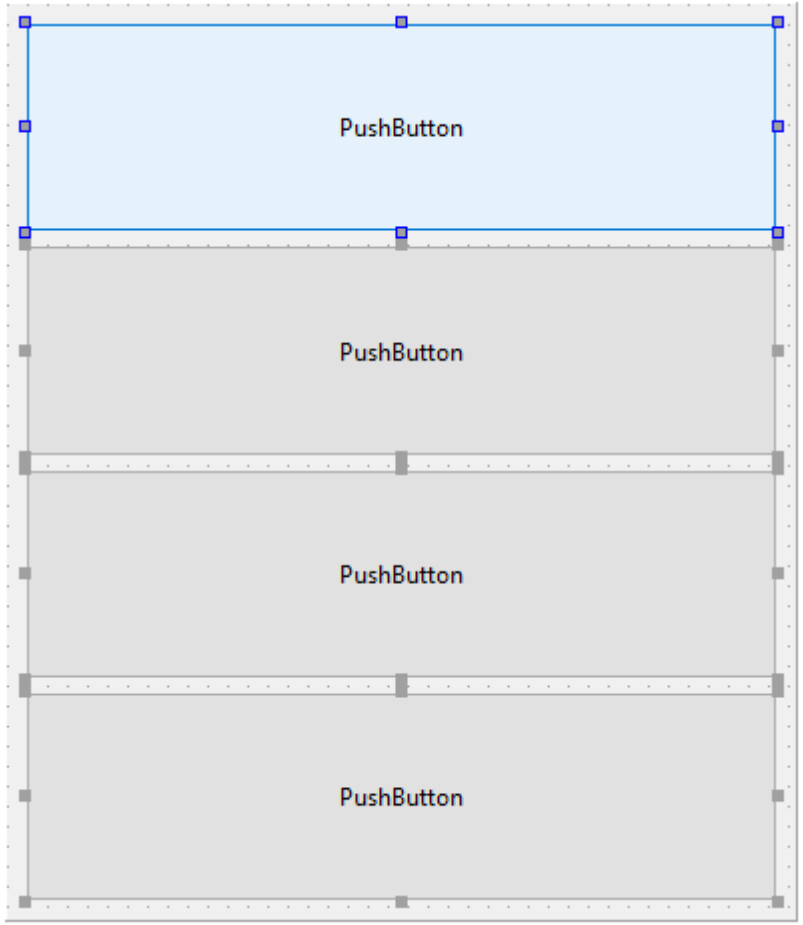
Właściwość	Wartość
QWidget	
sizePolicy	[Minimum, Fixed, 0, 0]
Strategia pozioma	Minimum
Strategia pionowa	Fixed
Rozciąganie w po...	0
Rozciąganie w pi...	0

sizePolicy

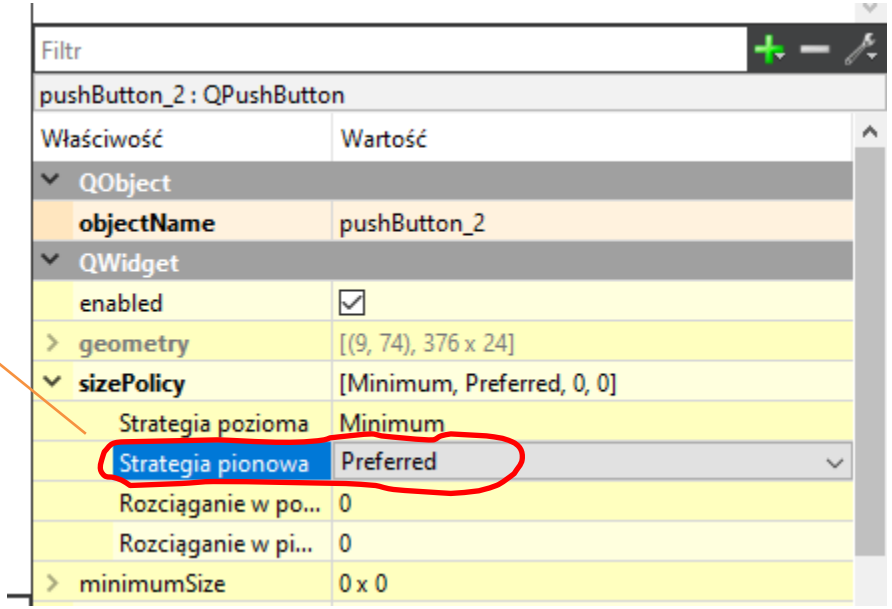
pushButton : QPushButton

Właściwość	Wartość
QWidget	
sizePolicy	[Minimum, Expanding, 0, 0]
Strategia pozioma	Minimum
Strategia pionowa	Expanding
Rozciąganie w po...	0
Rozciąganie w pi...	0

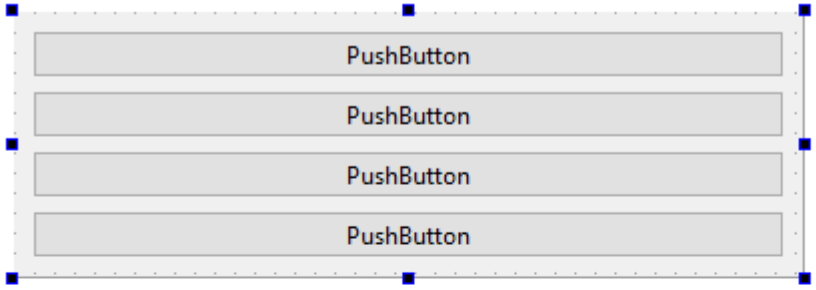
Vertical Layout



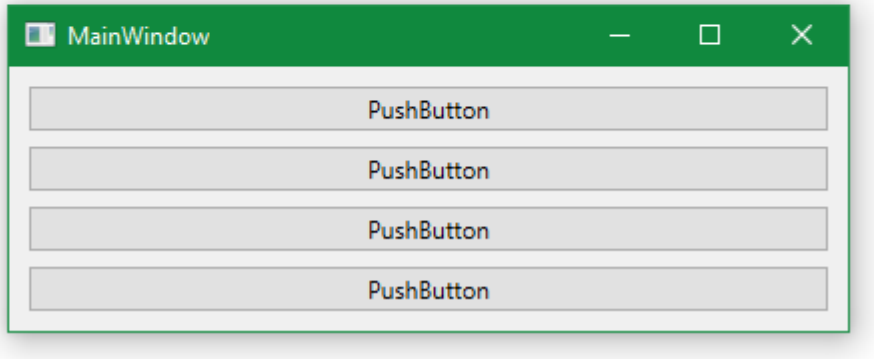
wszystkie przyciski mają Strategię pionową ustawioną na Prefered

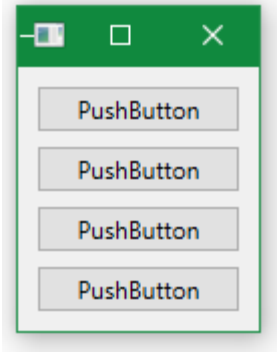


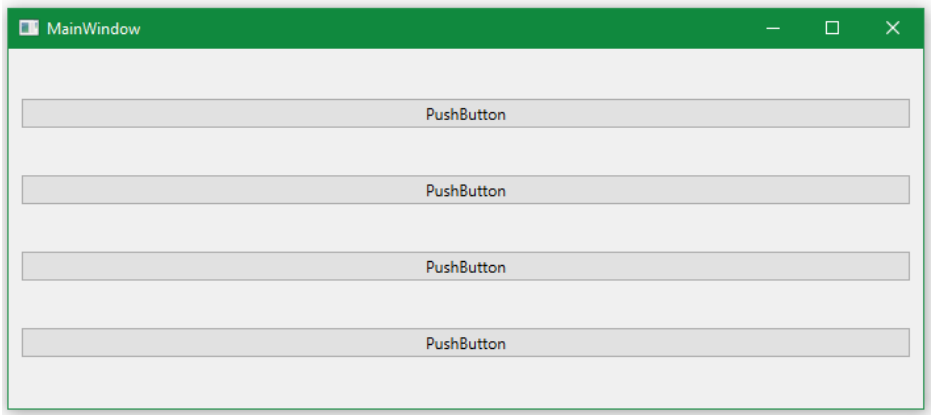
Vertical Layout



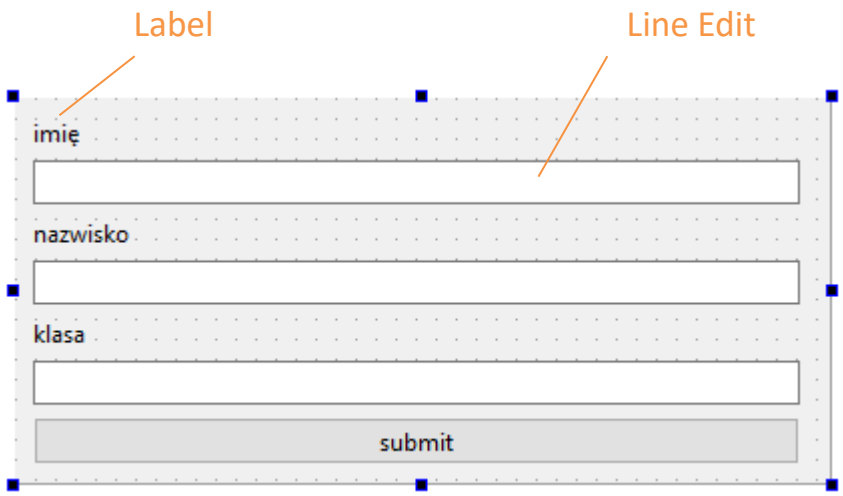
dzięki temu, że przyciski umieściliśmy w layoucie, ich wymiary dopasowują się do okna programu



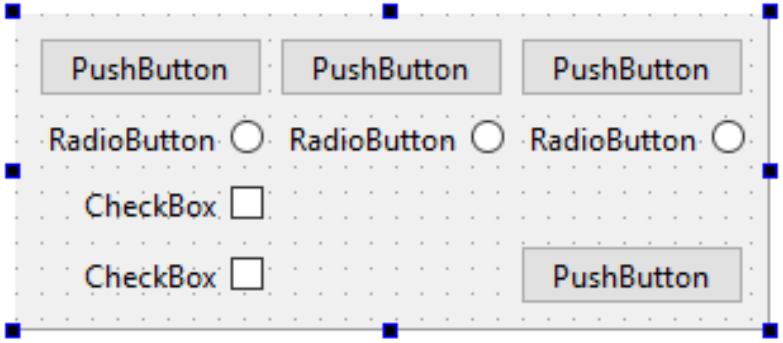




Form Layout
Grid Layout



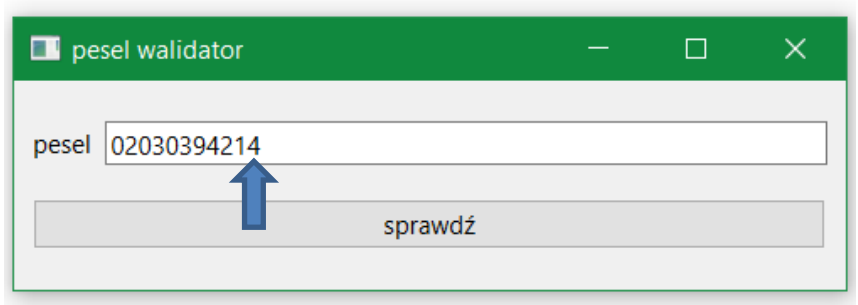
layout Formularz

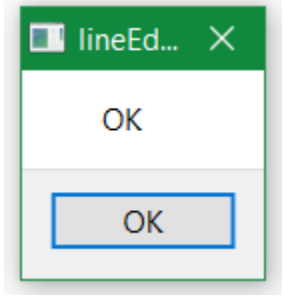


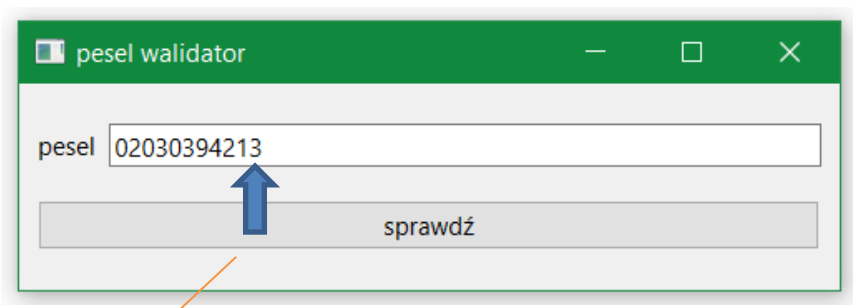
layout Siatka

pesel walidator

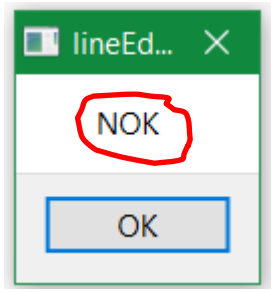
Zaprojektuj aplikację, która będzie weryfikować poprawność numeru pesel.

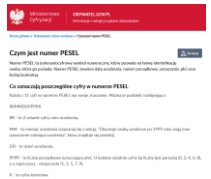






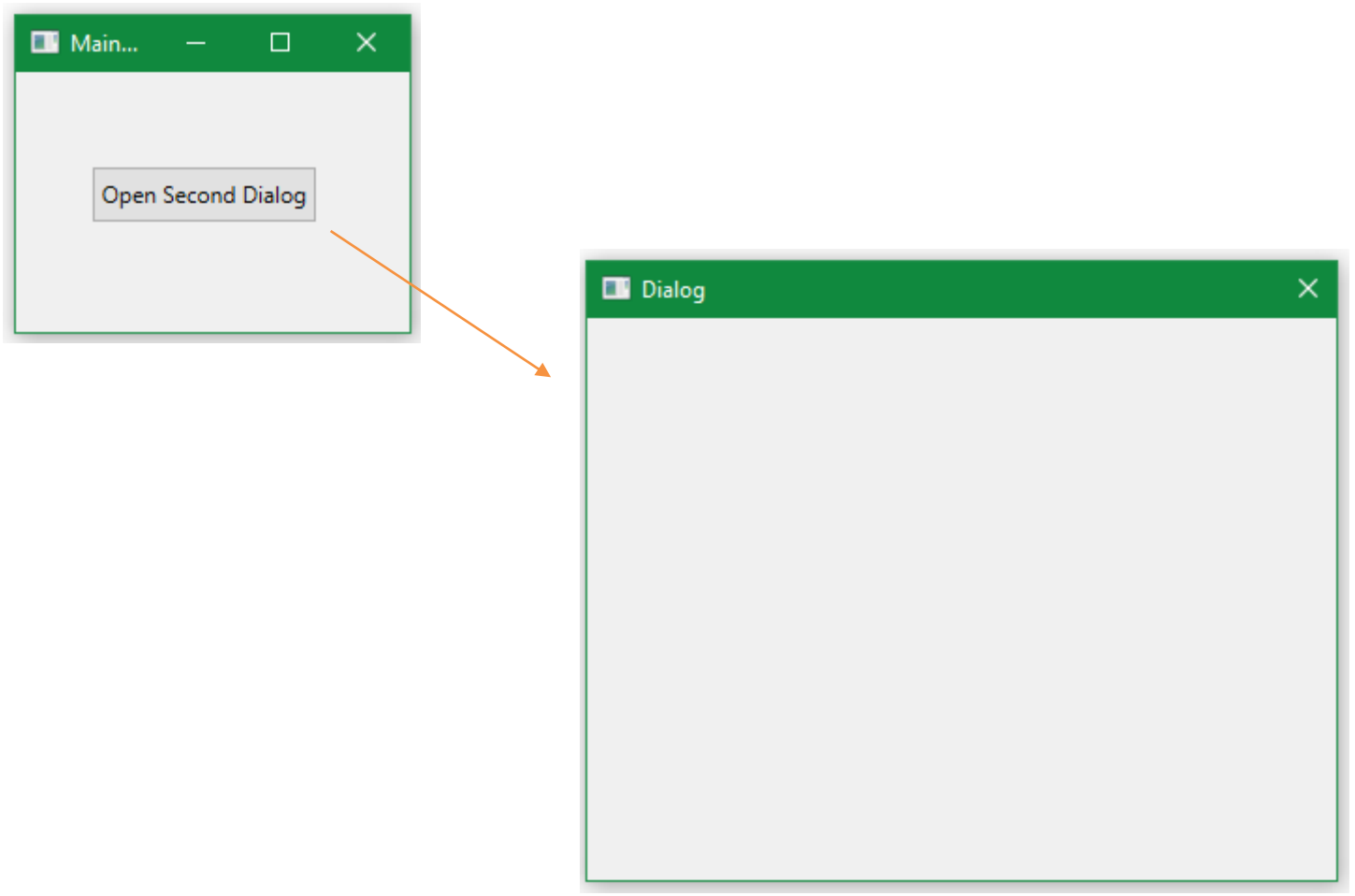
zła wartość cyfry kontrolnej
powoduje błąd weryfikacji





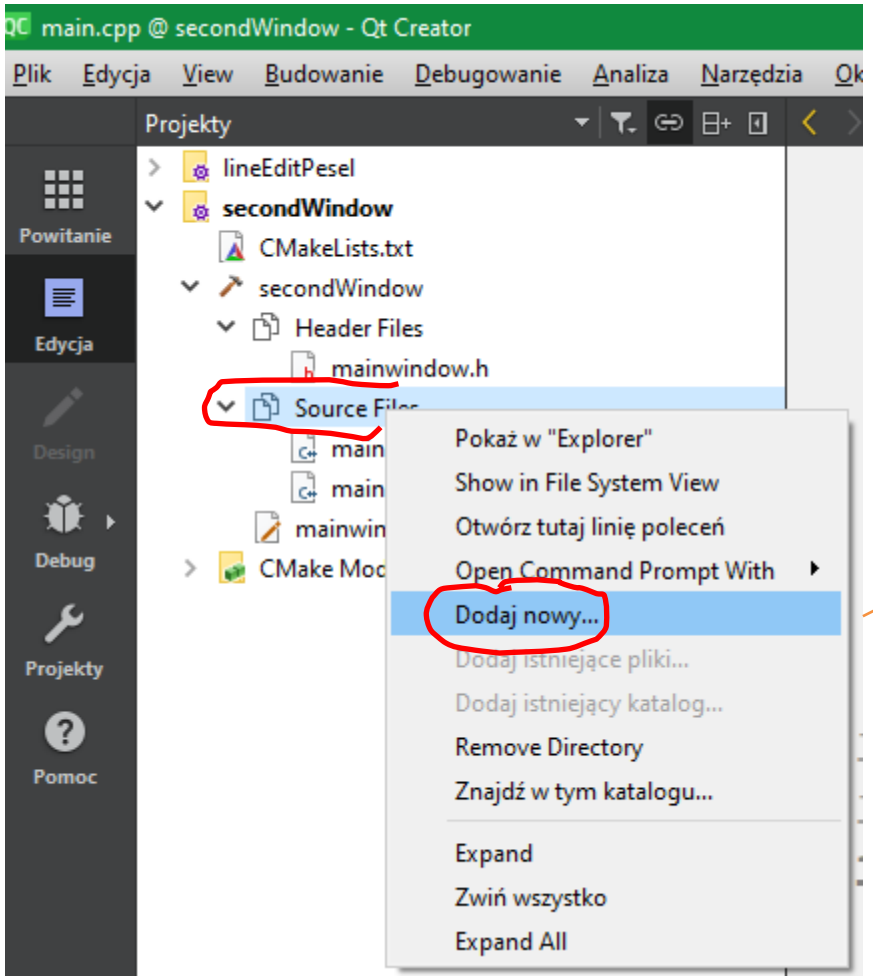
budowa numeru pesel

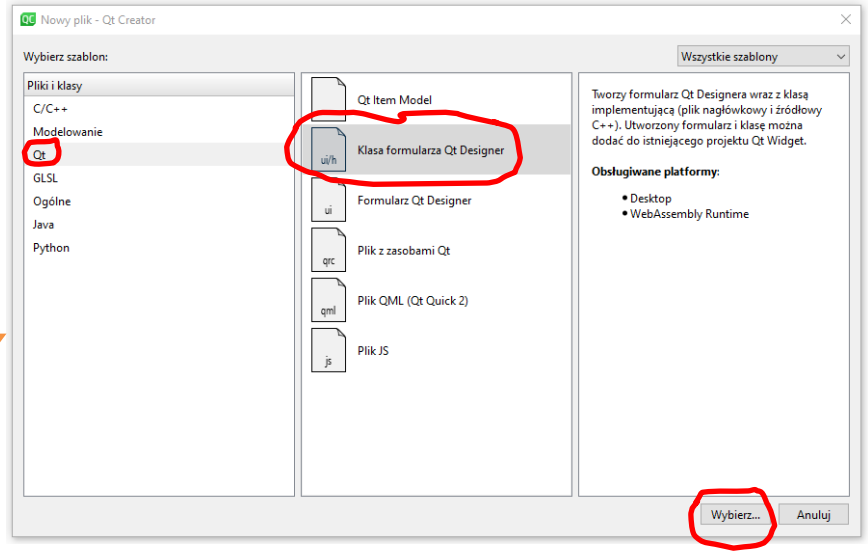
second window



second window

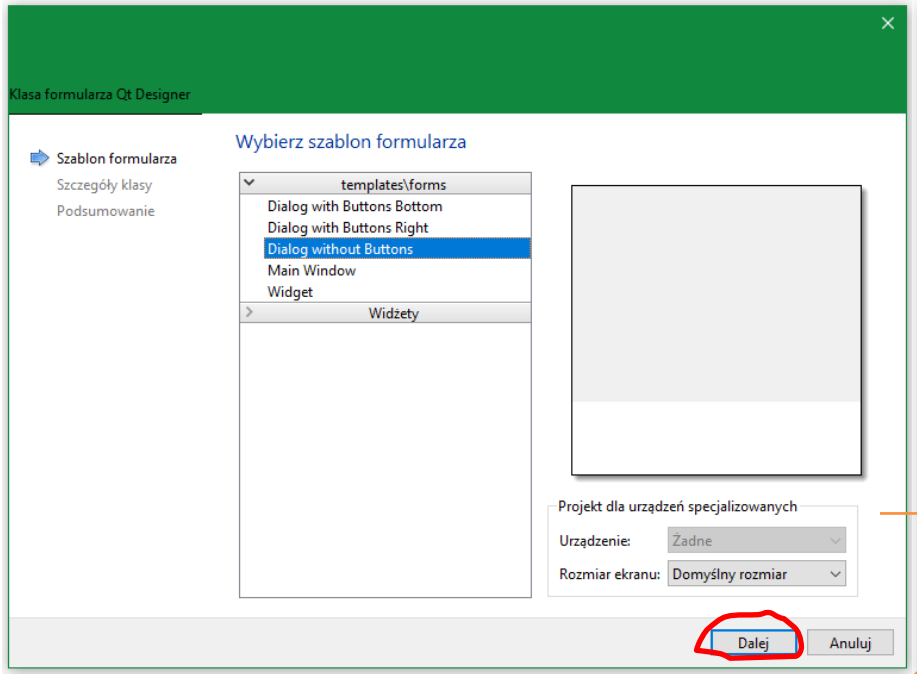
aby dodać nowe okno należy
wygenerować nowe pliki

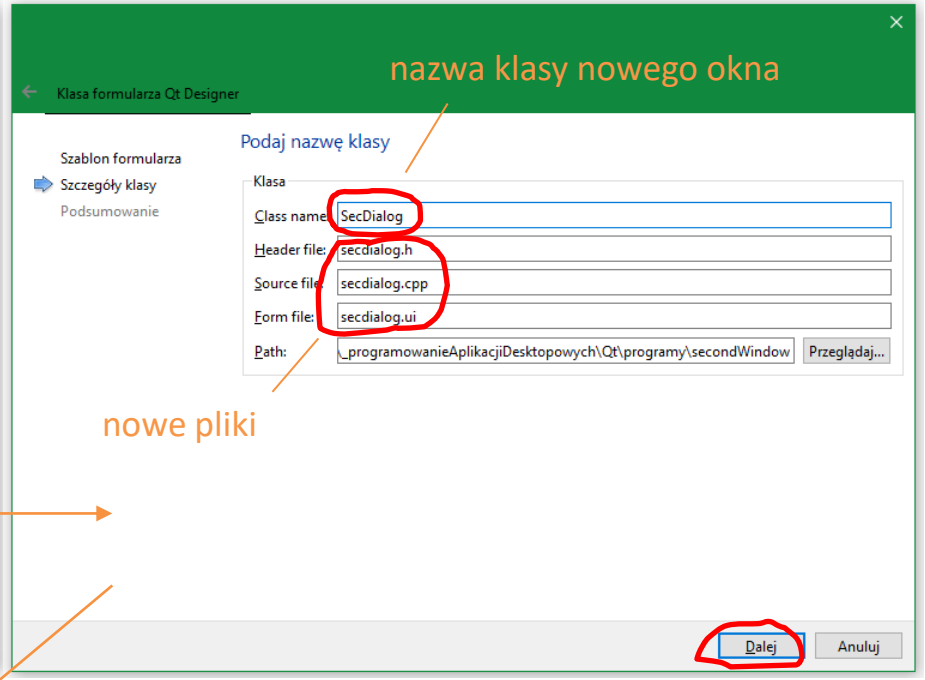




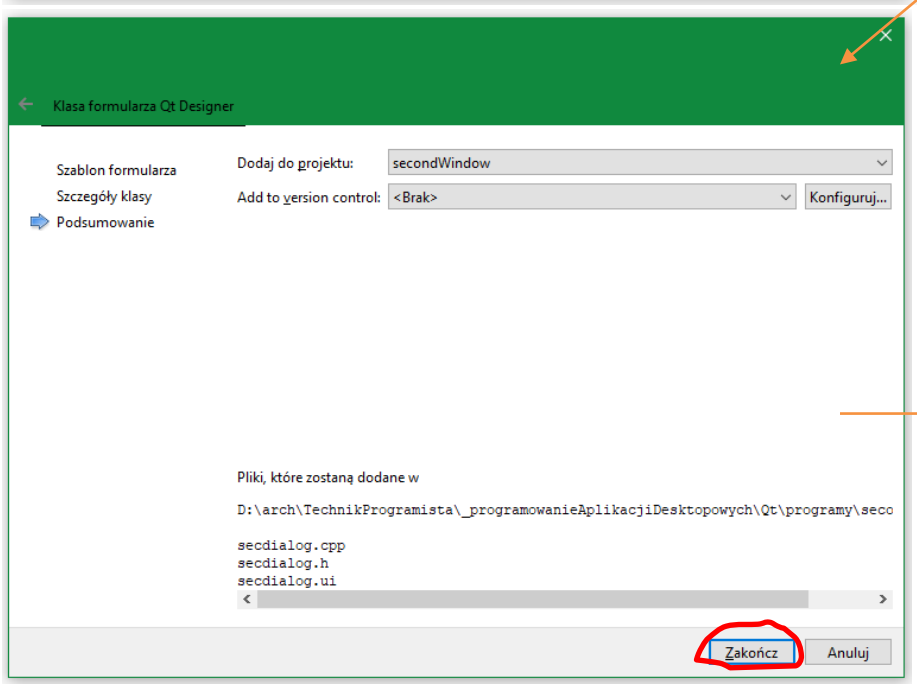
https://www.youtube.com/watch?v=VigUMAfE2q4

second window

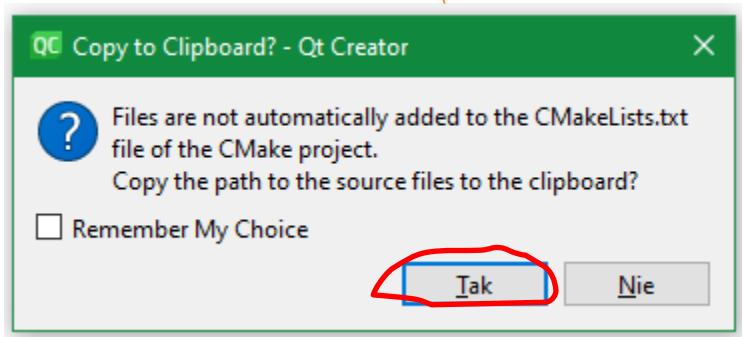




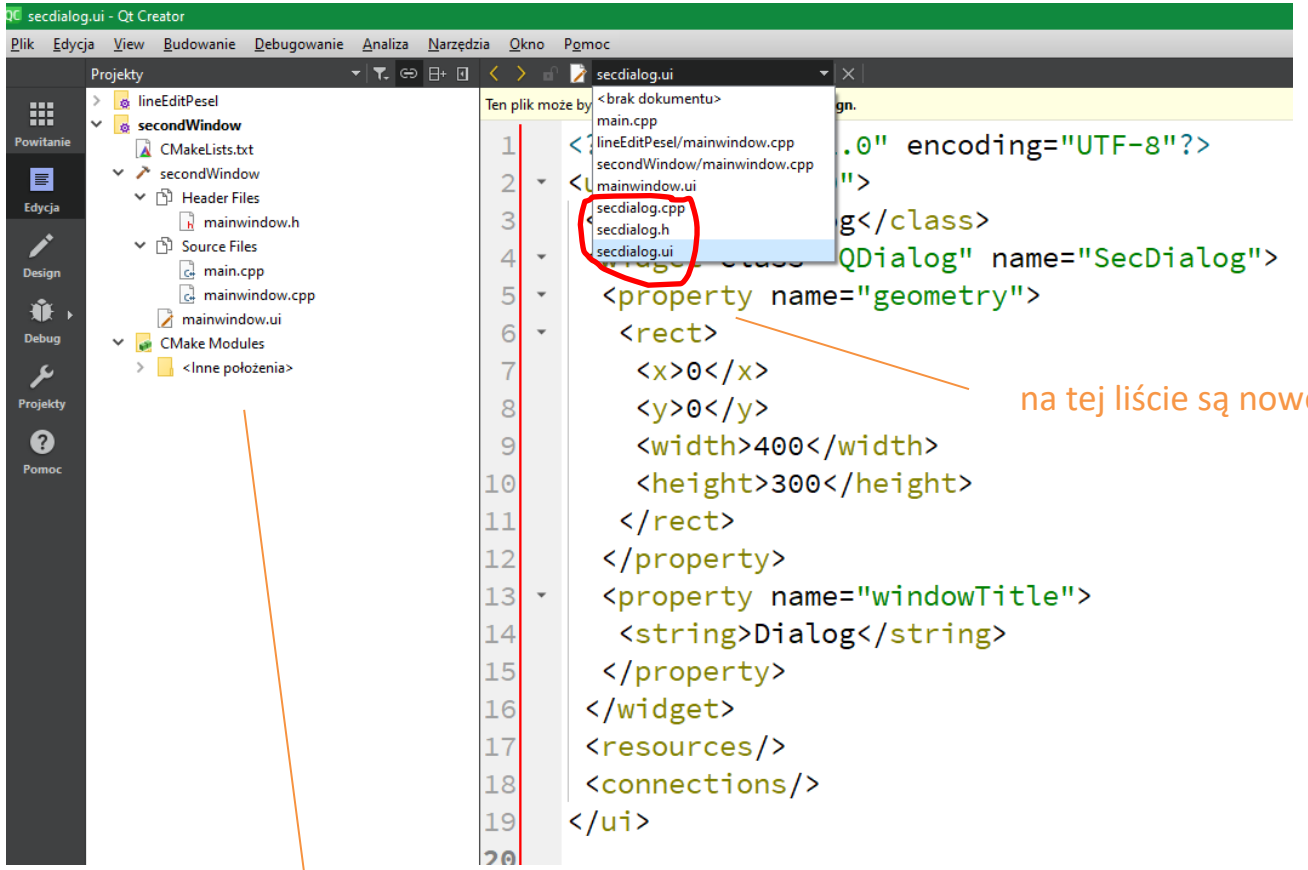
nowe pliki



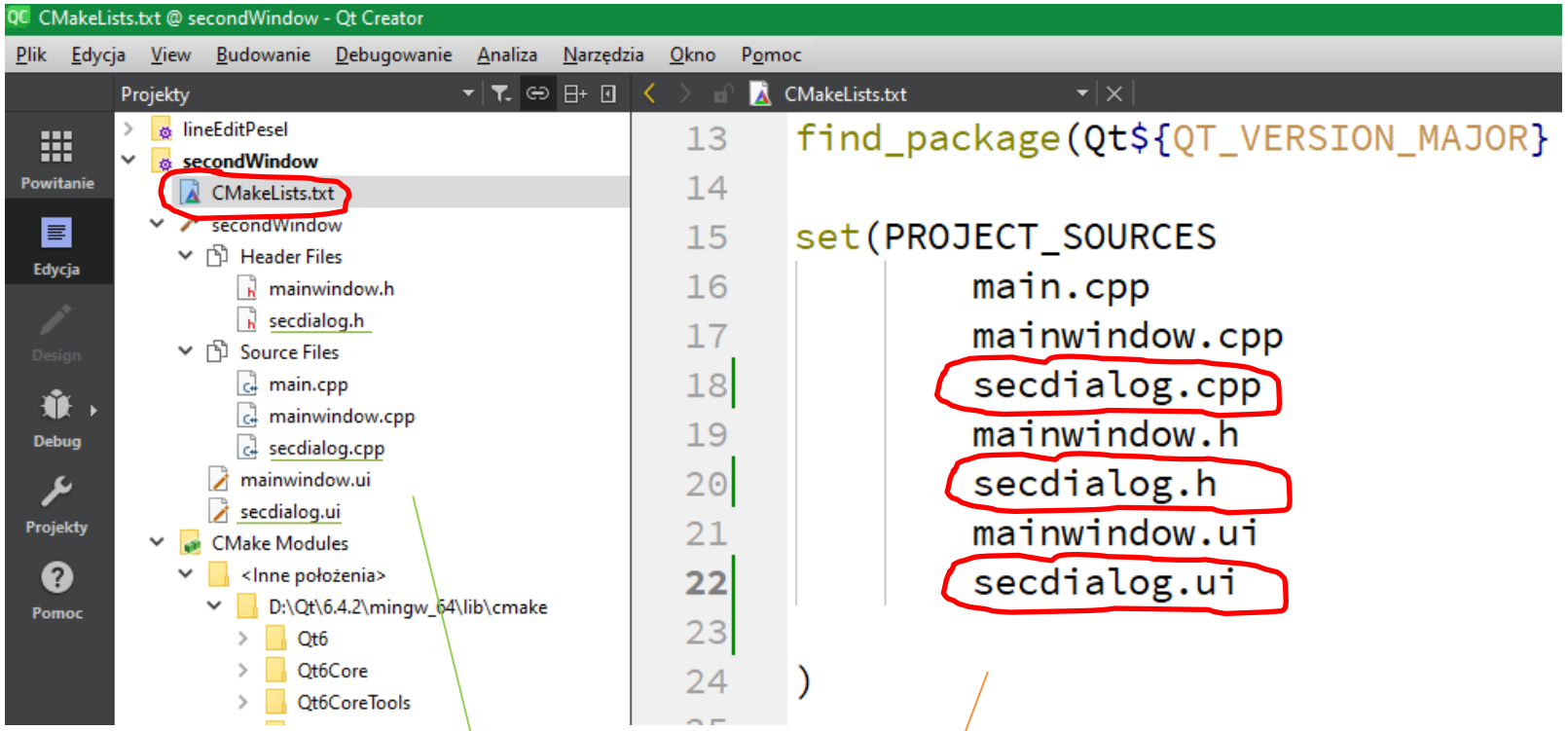
nazwy nowych plików nie zostały dodane do CMakeLists.txt (aby stworzone pliki pojawiły się w projekcie należy wpisać ich nazwy do pliku CMakeLists.txt)



second window



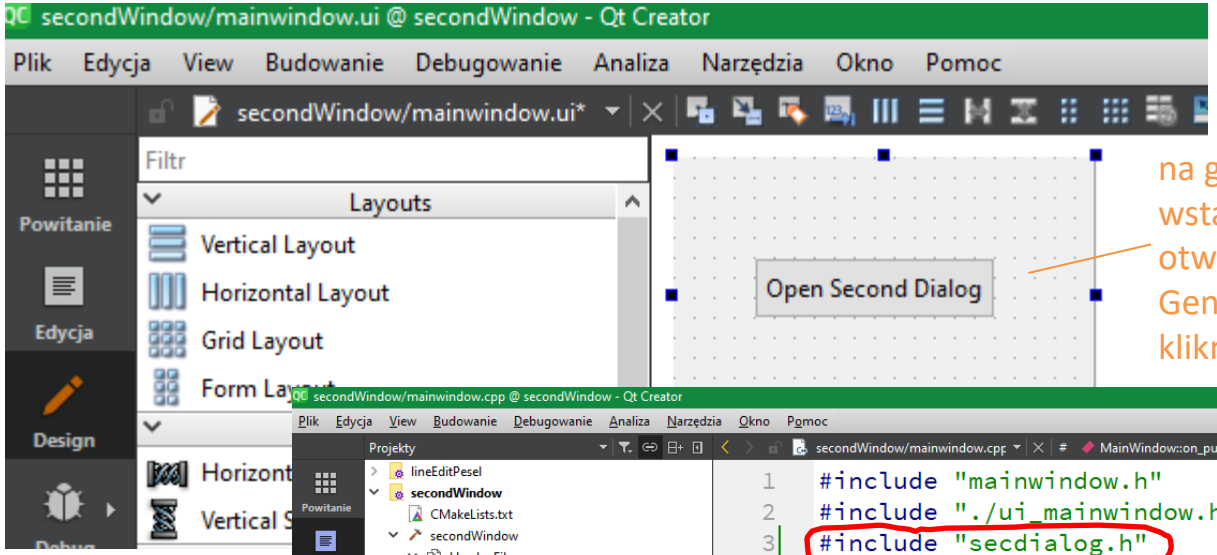
second window



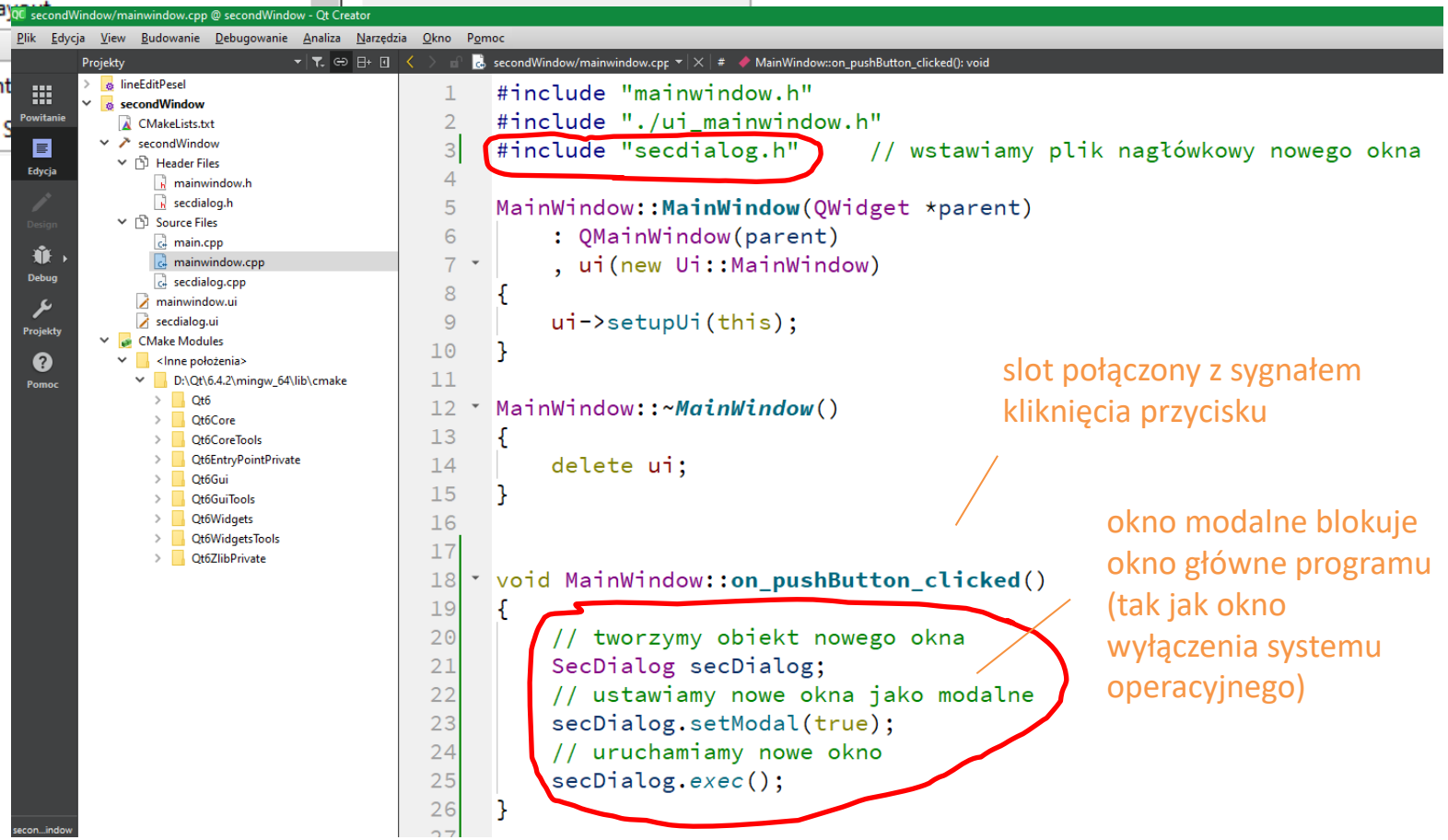
aby stworzone pliki pojawiły się w projekcie należy je wpisać do pliku CMakeLists.txt

nowe pliki pojawiły się w projekcie

second window



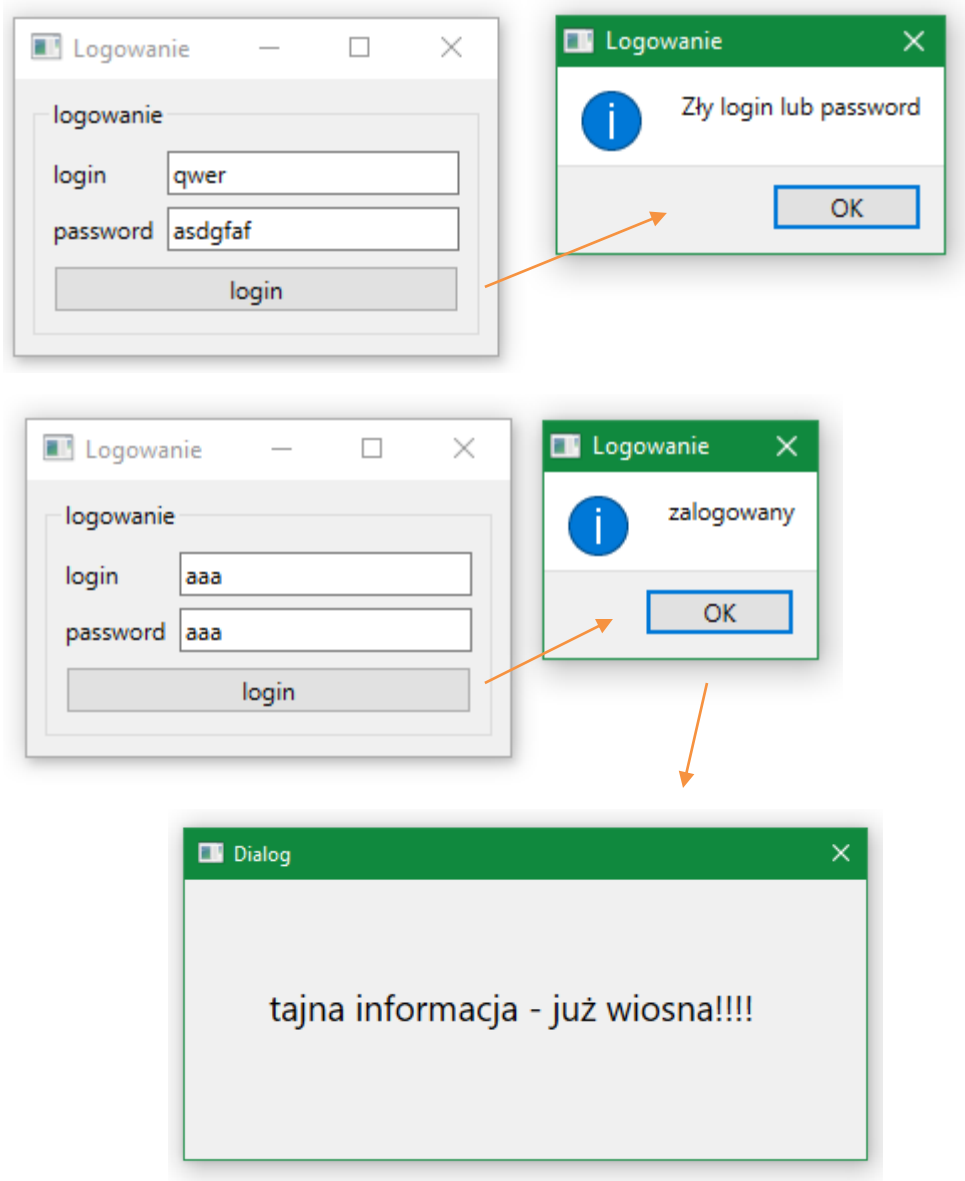
na głównym oknie (mainwindow)
wstawiamy przycisk
otwierający drugie okno.
Generujemy slot połączony z sygnałem
kliknięcia przycisku (prawy/przejdź do slotu)



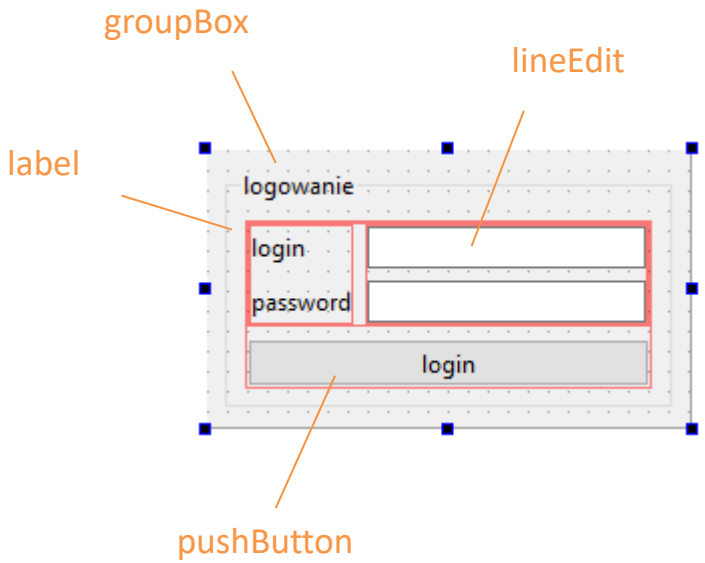
slot połączony z sygnałem
kliknięcia przycisku

okno modalne blokuje
okno główne programu
(tak jak okno
wyłączenia systemu
operacyjnego)

login

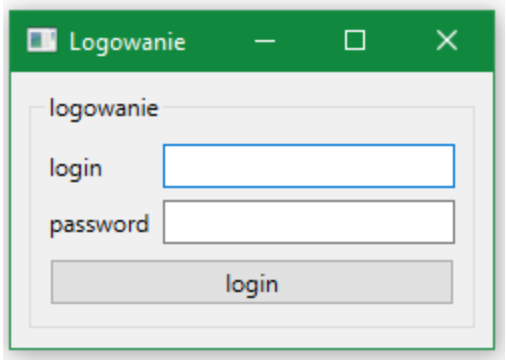

Logowanie
logowanie
login
qwer
password
asdgfaf
login
Logowanie
Zły login lub password
OK
Logowanie
logowanie
login
aaa
password
aaa
login
Logowanie
zalogowany
OK
Dialog
tajna informacja - już wiosna!!!!

login



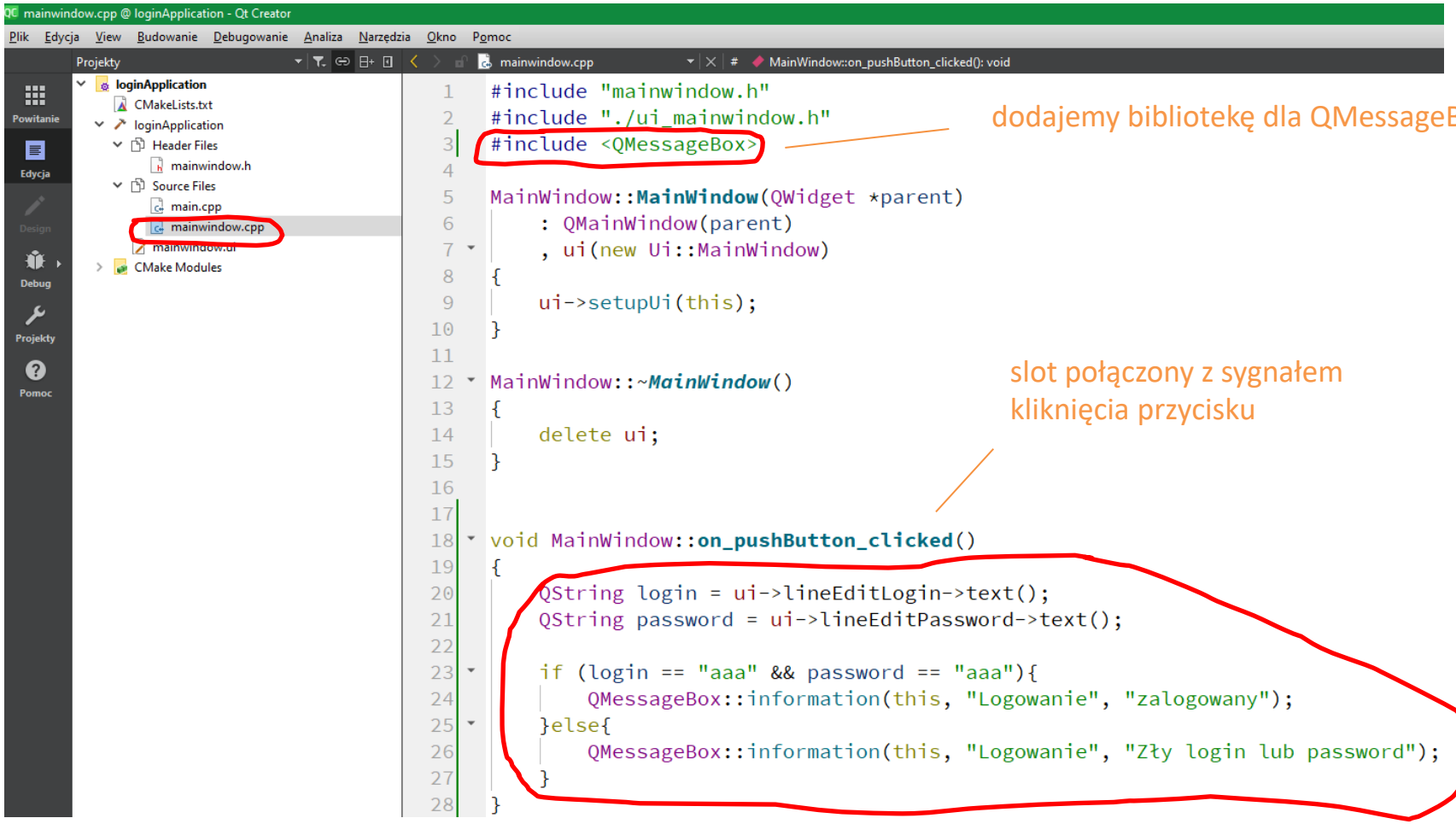
Obiekt	Klasa
MainWindow	QMainWindow
centralwidget	QWidget
groupBox	QGroupBox
verticalLayout_3	QVBoxLayout
horizontalLayout	QHBoxLayout
verticalLayout	QVBoxLayout
label	QLabel
label_2	QLabel
verticalLayout_2	QVBoxLayout
lineEditLogin	QLineEdit
lineEditPassword	QLineEdit
pushButton	QPushButton

zmiana nazwy pól lineEdit



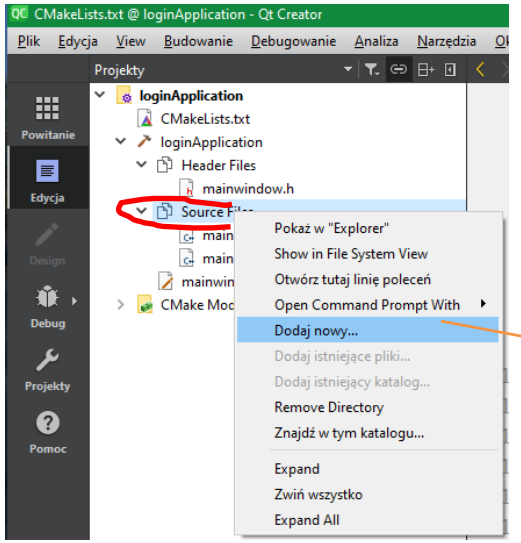
https://www.youtube.com/watch?v=6_elY8O20I8&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=10

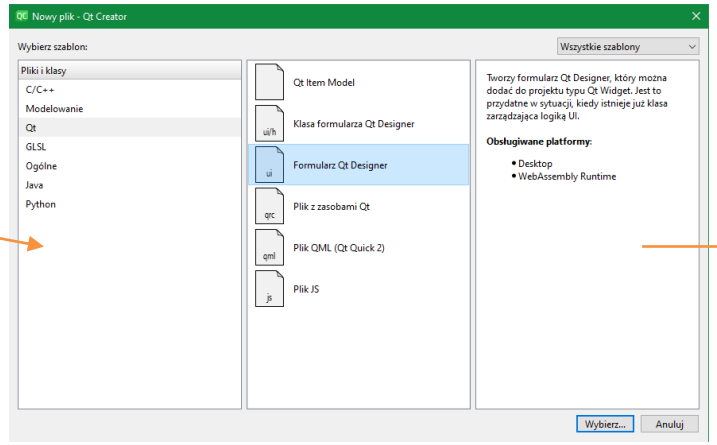
login


mainwindow.cpp @ loginApplication - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Projekty
loginApplication
CMakeLists.txt
loginApplication
Header Files
mainwindow.h
Source Files
main.cpp
mainwindow.cpp
CMake Modules
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
mainwindow.cpp
MainWindow::on_pushButton_clicked(): void
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QMessageBox>
MainWindow::MainWindow(QWidget *parent)
: QMainWindow(parent)
, ui(new Ui::MainWindow)
{
ui->setupUi(this);
}
MainWindow::~MainWindow()
{
delete ui;
}
void MainWindow::on_pushButton_clicked()
{
QString login = ui->lineEditLogin->text();
QString password = ui->lineEditPassword->text();
if (login == "aaa" && password == "aaa"){
QMessageBox::information(this, "Logowanie", "zalogowany");
}else{
QMessageBox::information(this, "Logowanie", "Zły login lub password");
}
}
dodajemy bibliotekę dla QMessageBox
slot połączony z sygnałem kliknięcia przycisku

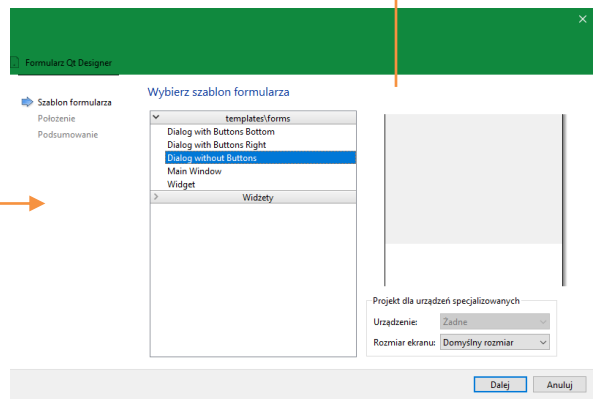
login

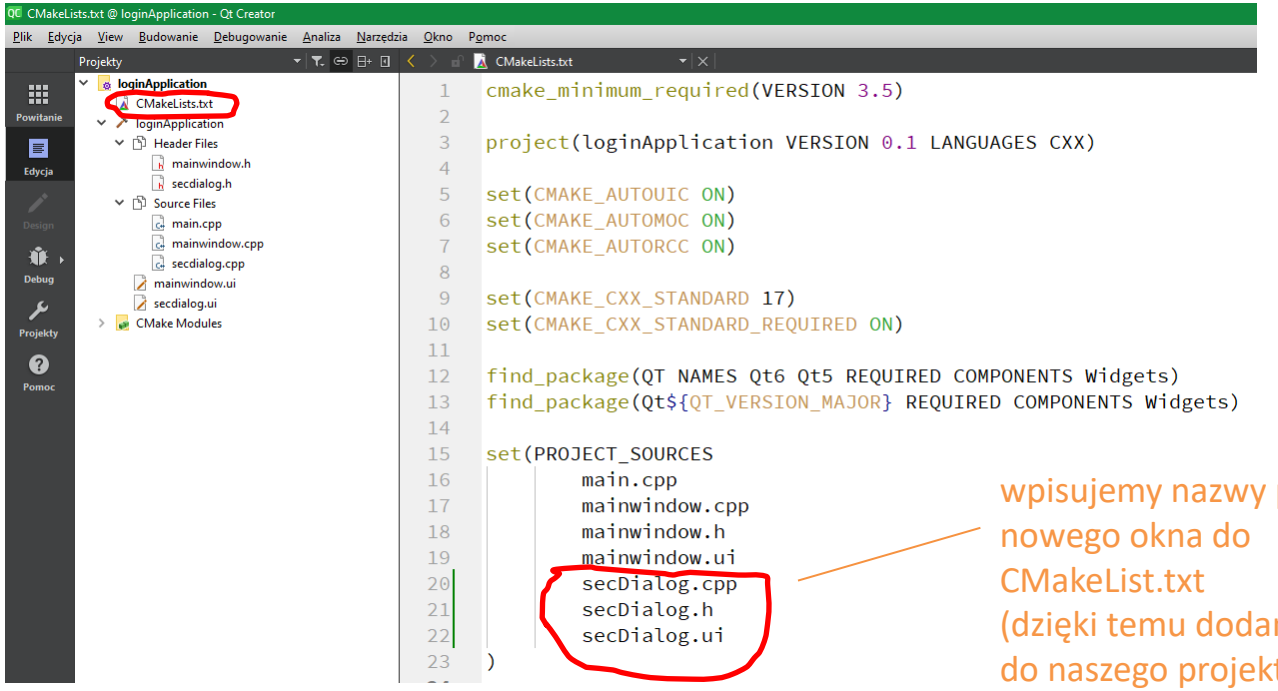
tworzymy
nowe okno





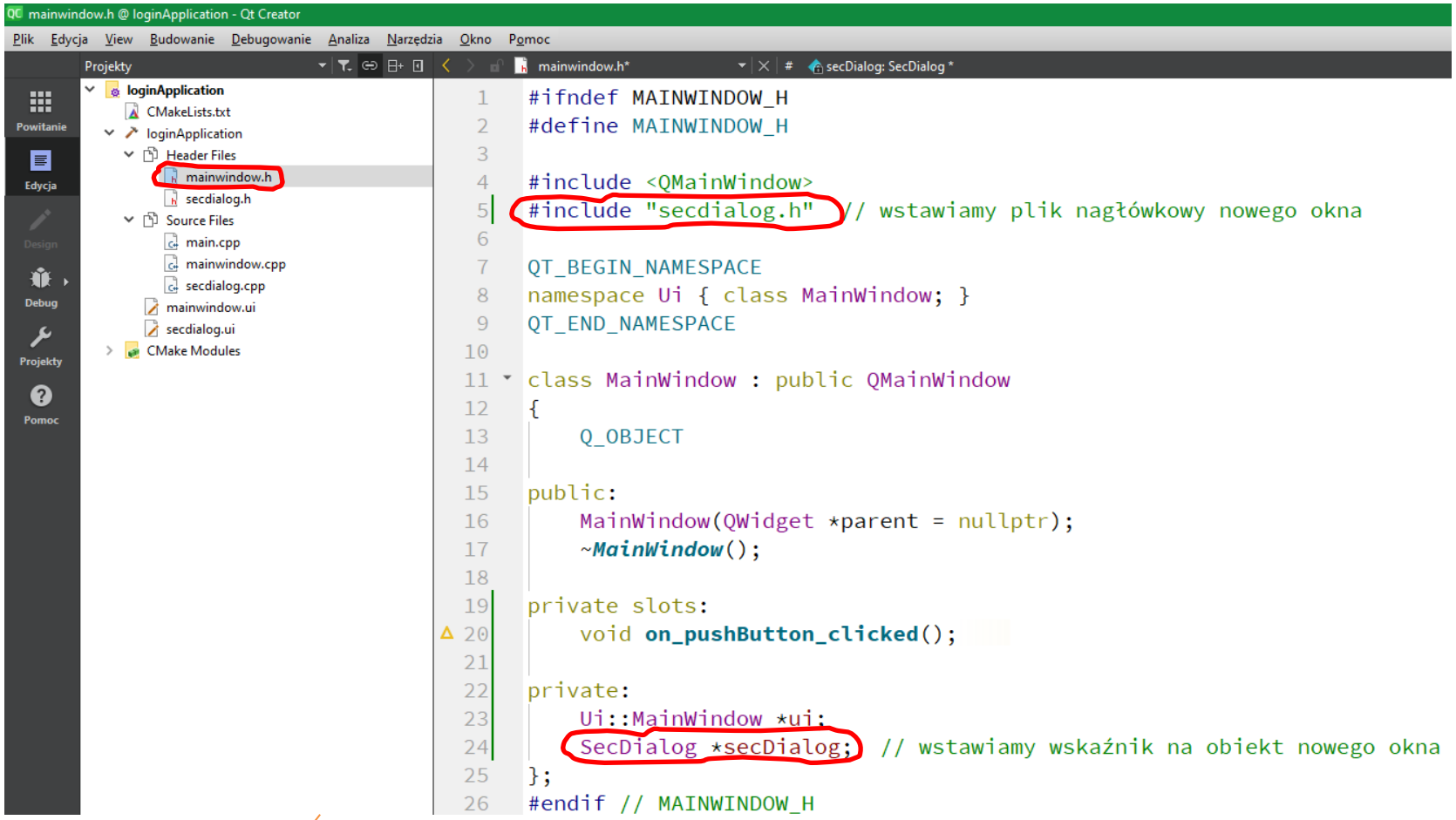
itd. ...





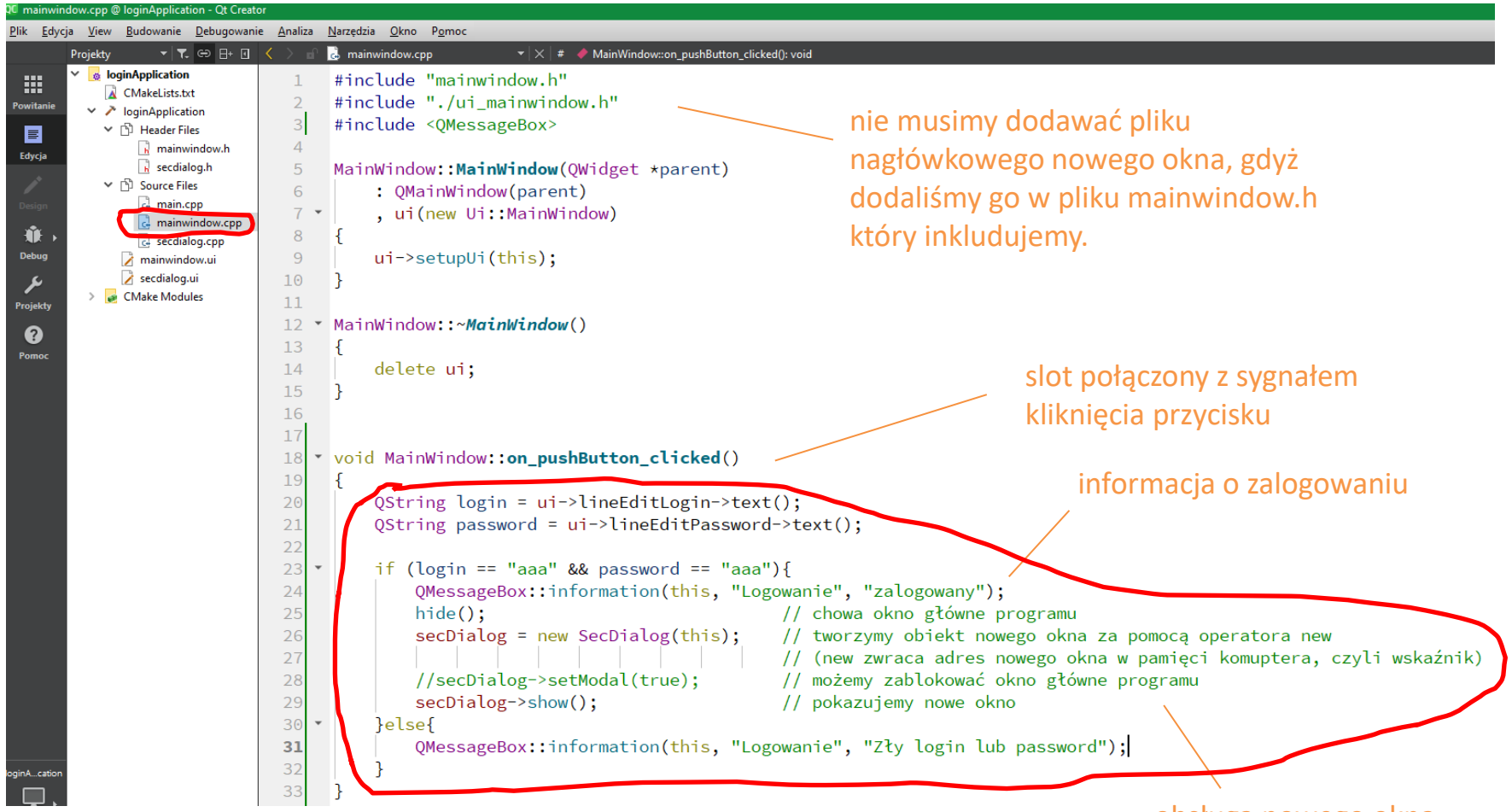
wpisujemy nazwy plików
nowego okna do
CMakeList.txt
(dzięki temu dodamy pliki
do naszego projektu)

login

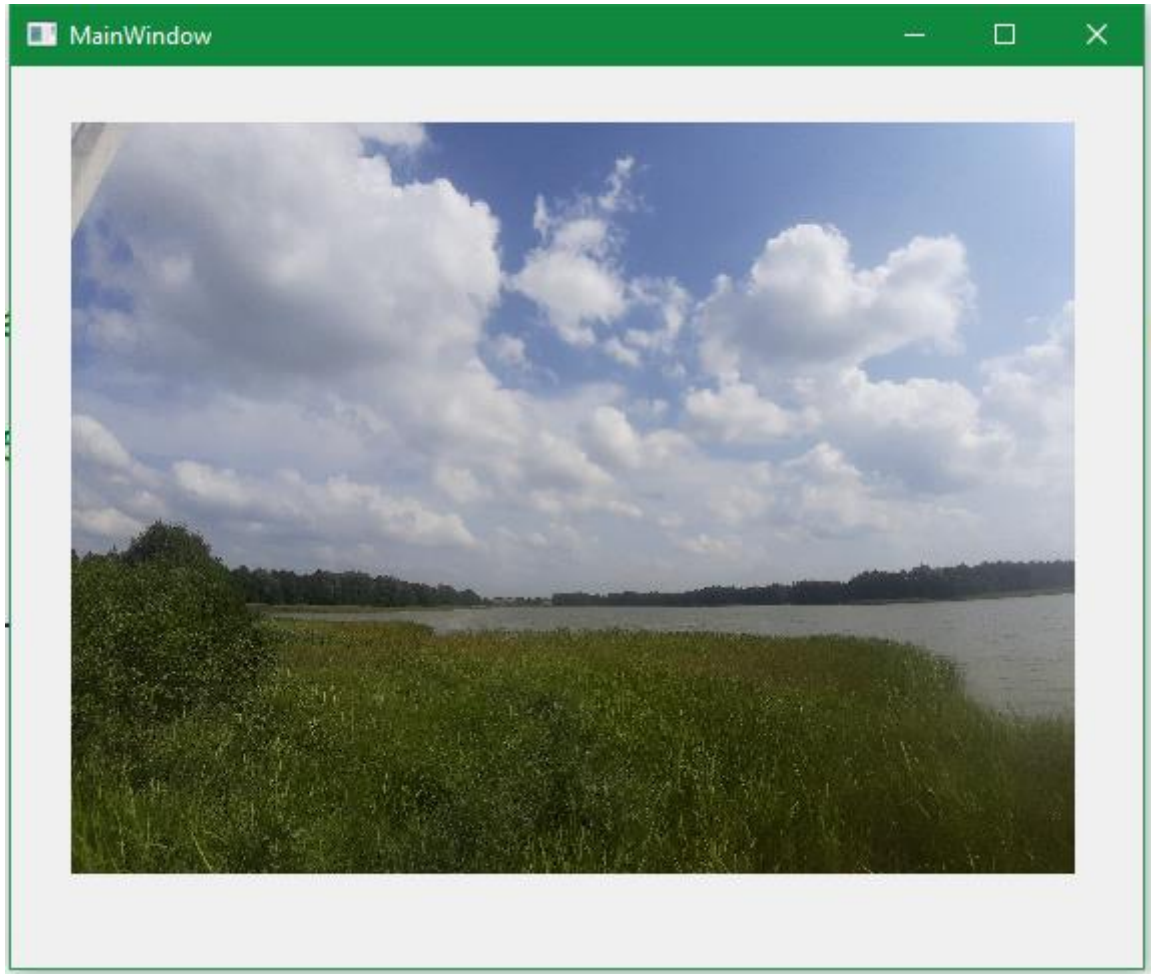


dodajemy nowe okno od klasy MainWindow, w tym celu w pliku nagłówkowym klasy MainWindow musimy zaimportować plik nagłówkowy nowego okna oraz dodać nowe prywatne pole

login


mainwindow.cpp @ loginApplication - Qt Creator
Plik Edycja View Budowanie Debugowanie Analiza Narzędzia Okno Pomoc
Projekty
mainwindow.cpp
MainWindow::on_pushButton_clicked(): void
loginApplication
CMakeLists.txt
loginApplication
Header Files
mainwindow.h
secdialog.h
Source Files
main.cpp
mainwindow.cpp
secdialog.cpp
mainwindow.ui
secdialog.ui
CMake Modules
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
1 #include "mainwindow.h"
2 #include "./ui_mainwindow.h"
3 #include <QMessageBox>
4
5 MainWindow::MainWindow(QWidget *parent)
6 : QMainWindow(parent)
7 , ui(new Ui::MainWindow)
8 {
9 ui->setupUi(this);
10 }
11
12 MainWindow::~MainWindow()
13 {
14 delete ui;
15 }
16
17
18 void MainWindow::on_pushButton_clicked()
19 {
20 QString login = ui->lineEditLogin->text();
21 QString password = ui->lineEditPassword->text();
22
23 if (login == "aaa" && password == "aaa"){
24 QMessageBox::information(this, "Logowanie", "zalogowany");
25 hide(); // chowa okno główne programu
26 secDialog = new SecDialog(this); // tworzymy obiekt nowego okna za pomocą operatora new
27 // (new zwraca adres nowego okna w pamięci komuptera, czyli wskaźnik)
28 //secDialog->setModal(true); // możemy zablokować okno główne programu
29 secDialog->show(); // pokazujemy nowe okno
30 }else{
31 QMessageBox::information(this, "Logowanie", "Zły login lub password");
32 }
33 }
nie musimy dodawać pliku nagłówkowego nowego okna, gdyż dodaliśmy go w pliku mainwindow.h który inkludujemy.
slot połączony z sygnałem kliknięcia przycisku
informacja o zalogowaniu
obsługa nowego okna

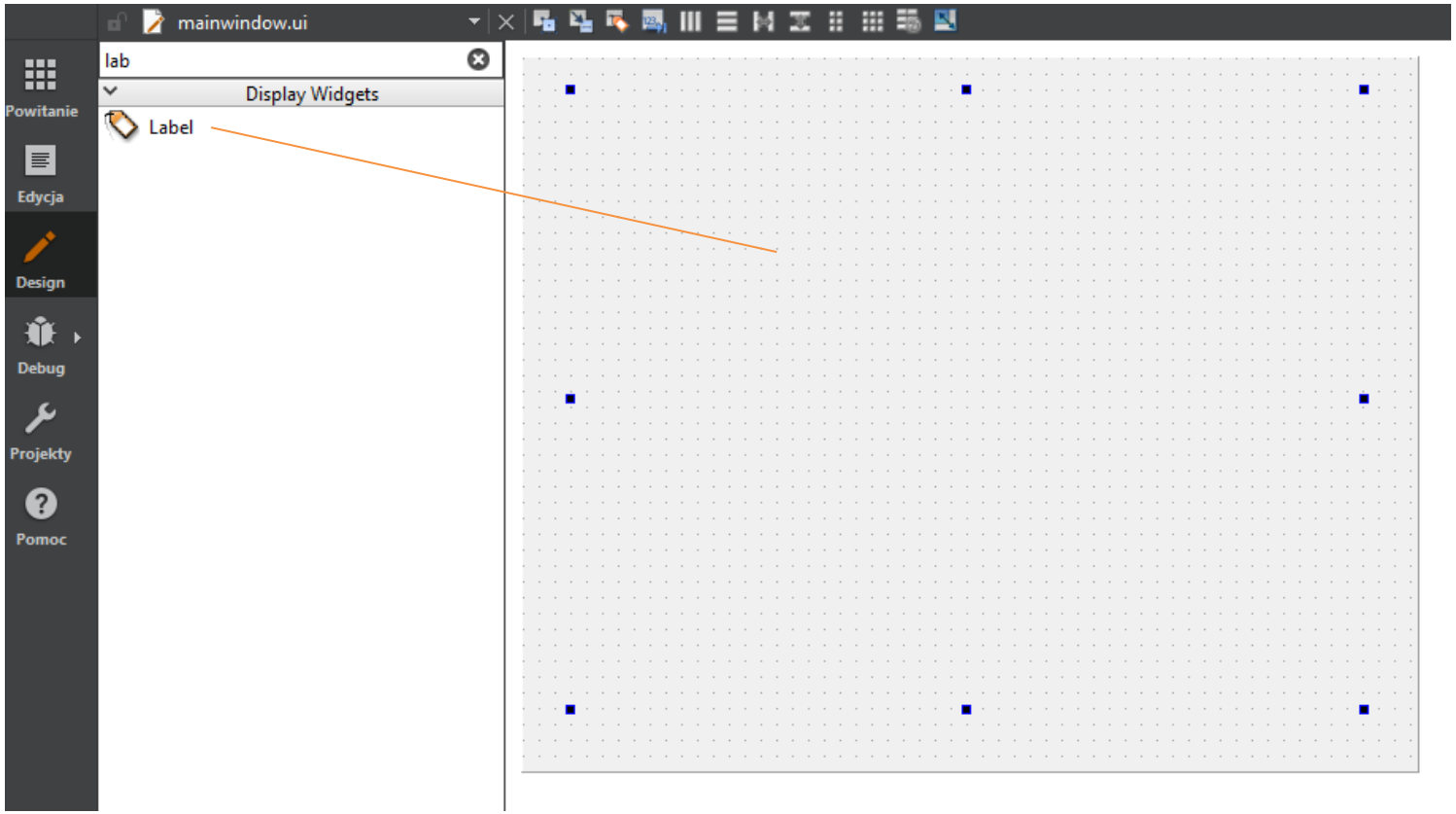
picture



https://www.youtube.com/watch?v=SyzxerD2Yps&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=11

picture

wstawiamy obrazek za pomocą komponentu Label

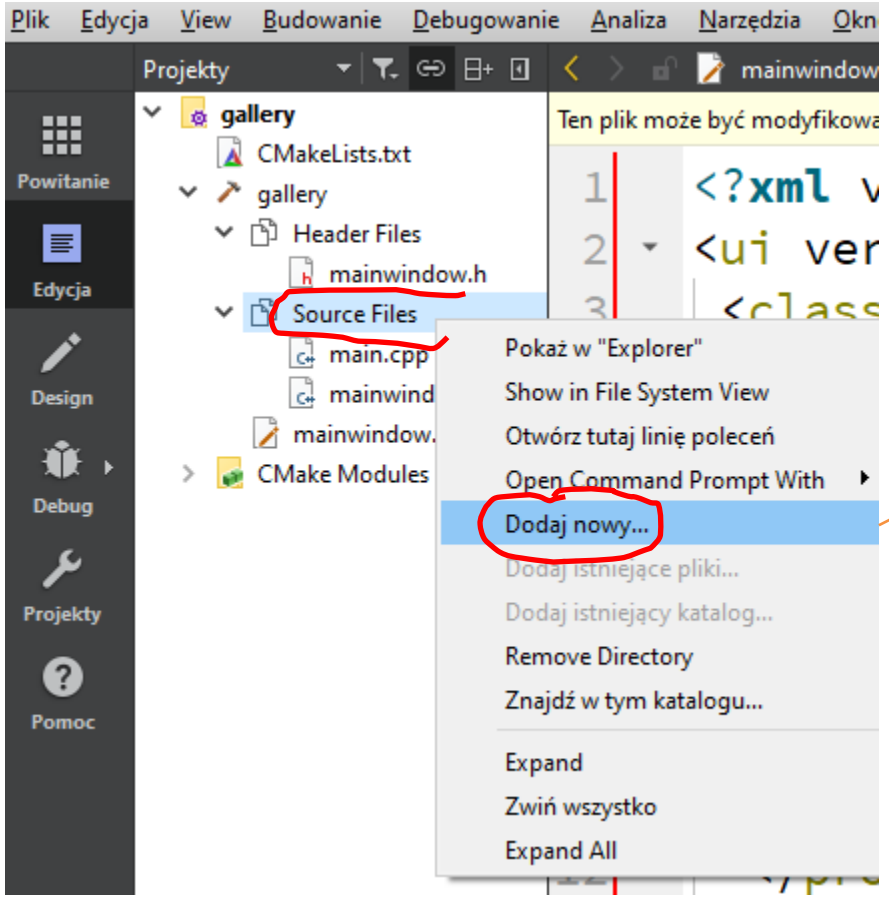


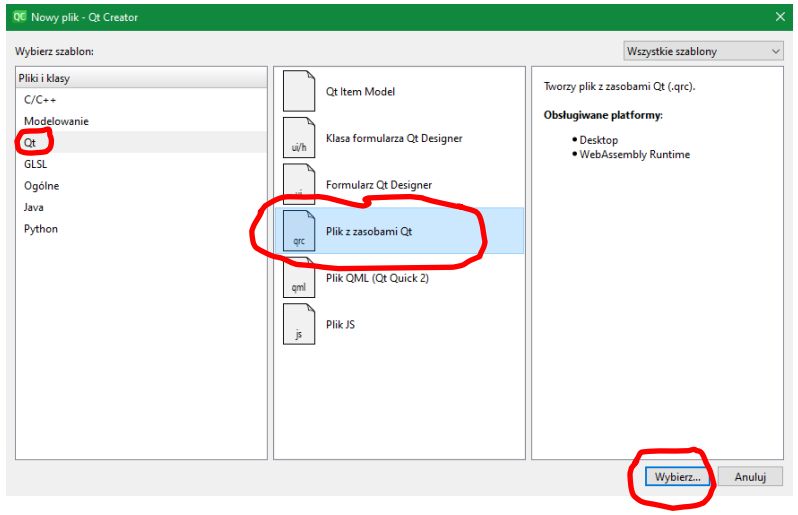
picture


Projekty
gallery
CMakeLists.txt
gallery
Header Files
mainwindow.h
Source Files
main.cpp
mainwindow.cpp
mainwindow.ui
CMake Modules
Powitanie
Edycja
Design
Debug
Projekty
Pomoc
mainwindow.cpp
MainWindow::~MainWindow()
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QPixmap>
MainWindow::MainWindow(QWidget *parent)
: QMainWindow(parent)
, ui(new Ui::MainWindow)
{
ui->setupUi(this);
QPixmap picture("D:/mazury.jpg");
//ui->label->setPixmap(picture);
//ui->label->setPixmap(picture.scaled(600,500,Qt::KeepAspectRatio));
int labelWidth = ui->label->width();
int labelHeight = ui->label->height();
ui->label->setPixmap(picture.scaled(labelWidth,labelHeight,Qt::KeepAspectRatio));
}
MainWindow::~MainWindow()
{
delete ui;
}
ścieżka bezwzględna obrazka!
(trzeba dodać folder z zasobami)
najprostszy sposób
sposób z podaniem wymiarów
wymiary obrazka dostosowują się
do wymiarów labela

picture
plik z zasobami

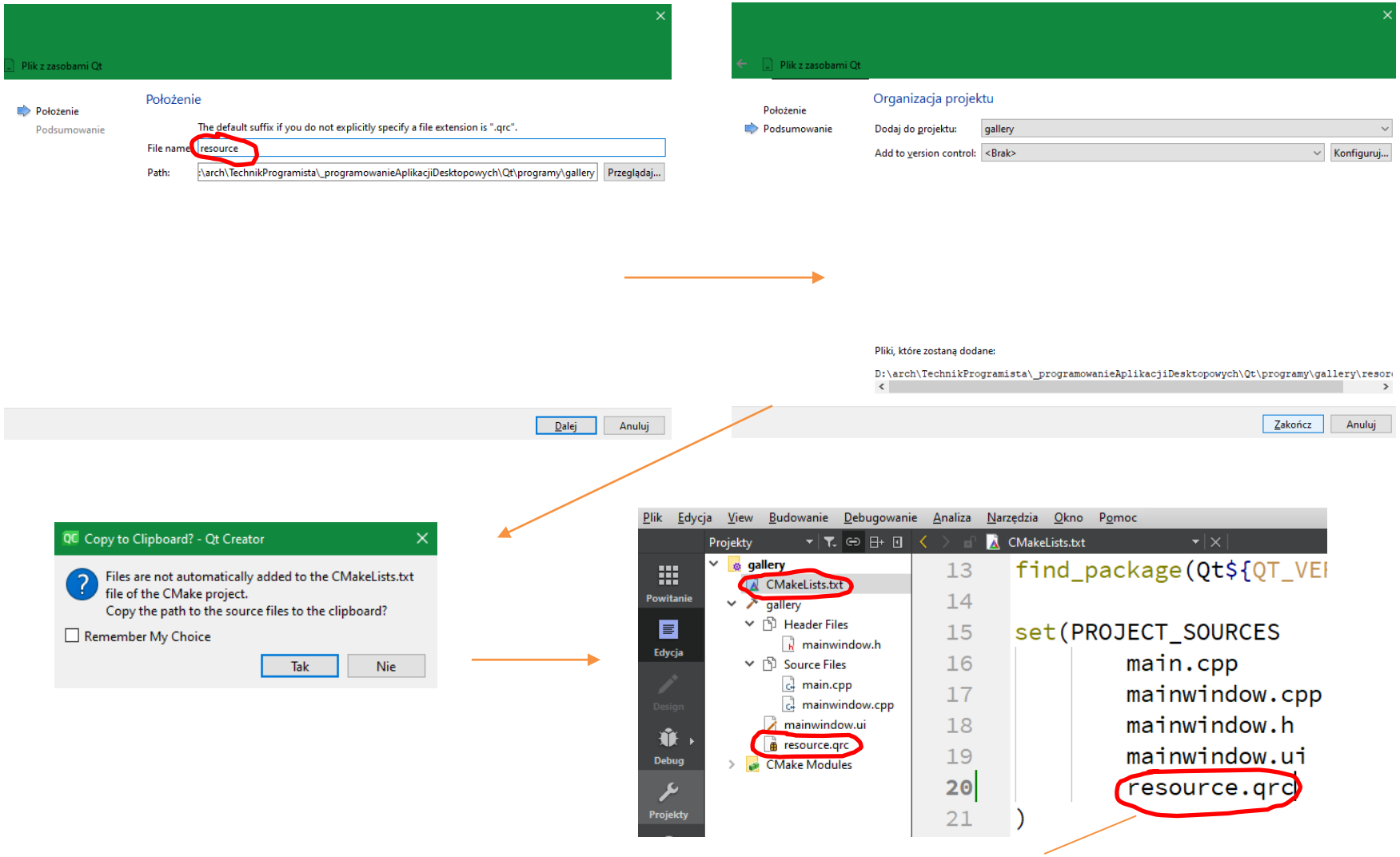
dodajemy do projektu plik resource.qrc z zasobami, aby uniknąć podawania ścieżki bezwzględnej obrazka





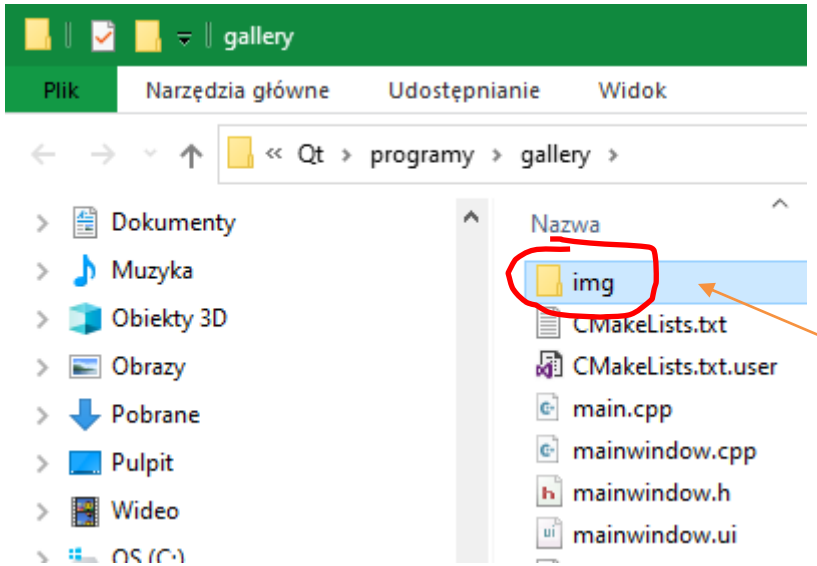
https://www.youtube.com/watch?v=CTbpS0PN-JQ&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=13

picture
plik z zasobami



wpisujemy nazwę pliku z zasobami
w pliku CMakeLists.txt

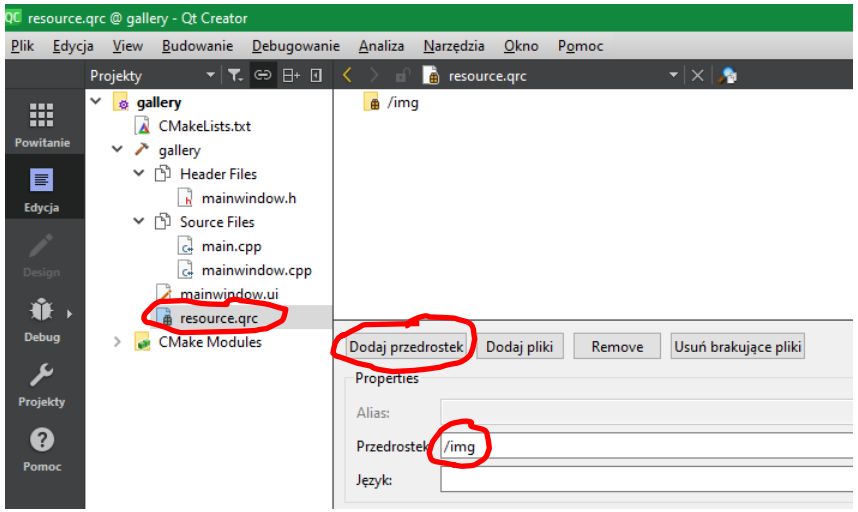
picture
plik z zasobami



w katalogu projektu tworzymy folder z obrazkiem

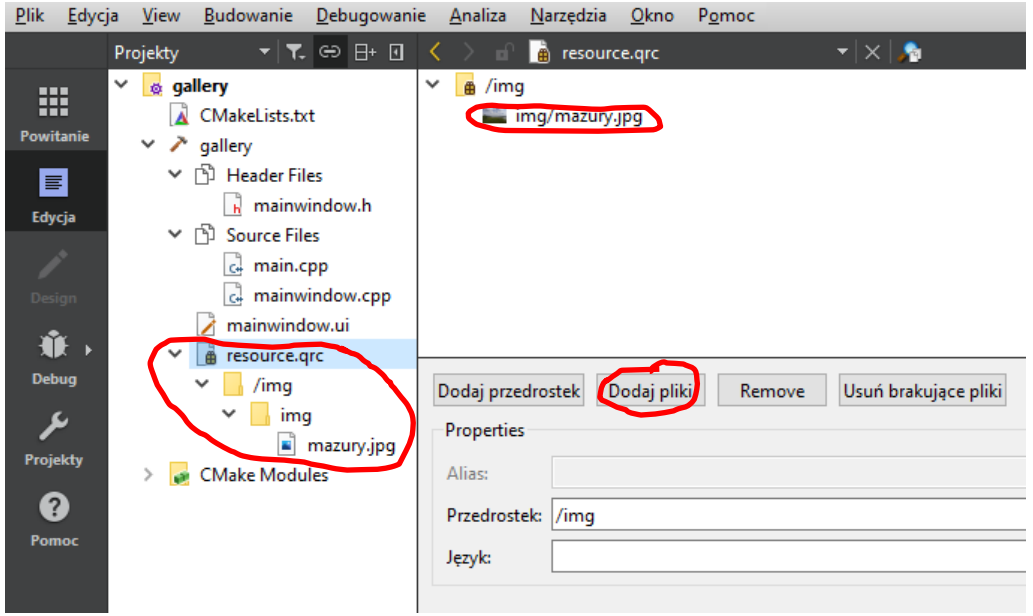
mazury.jpg

picture
plik z zasobami

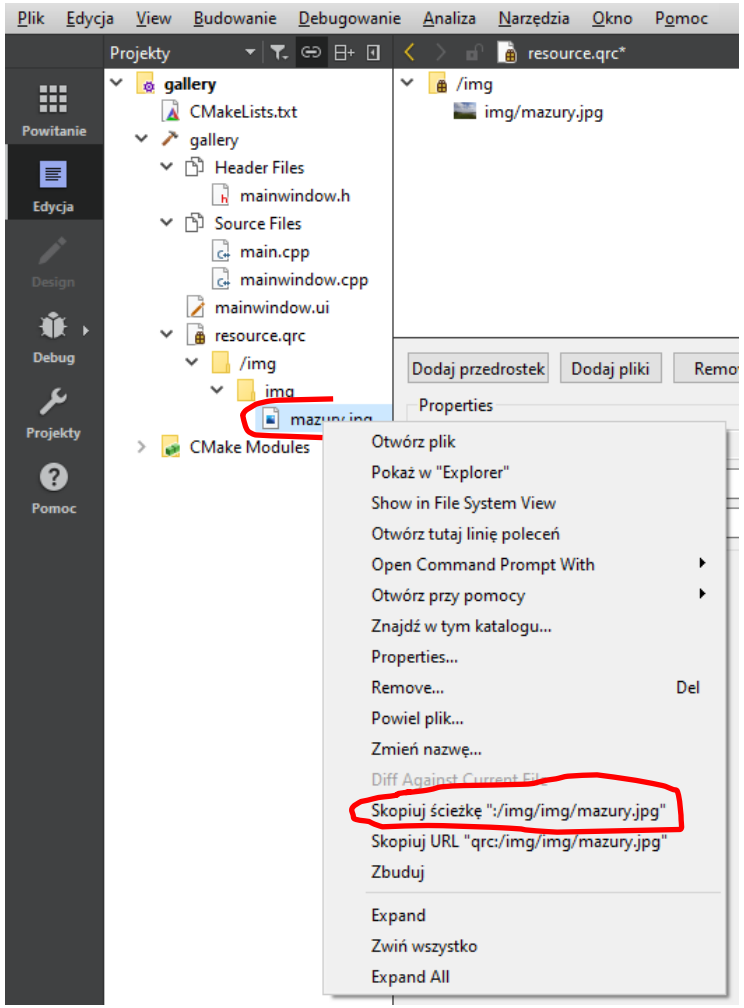


w pliku resource.qrc
dodajemy przedrostek /img

a następnie dodajemy obrazek
z katalogu img



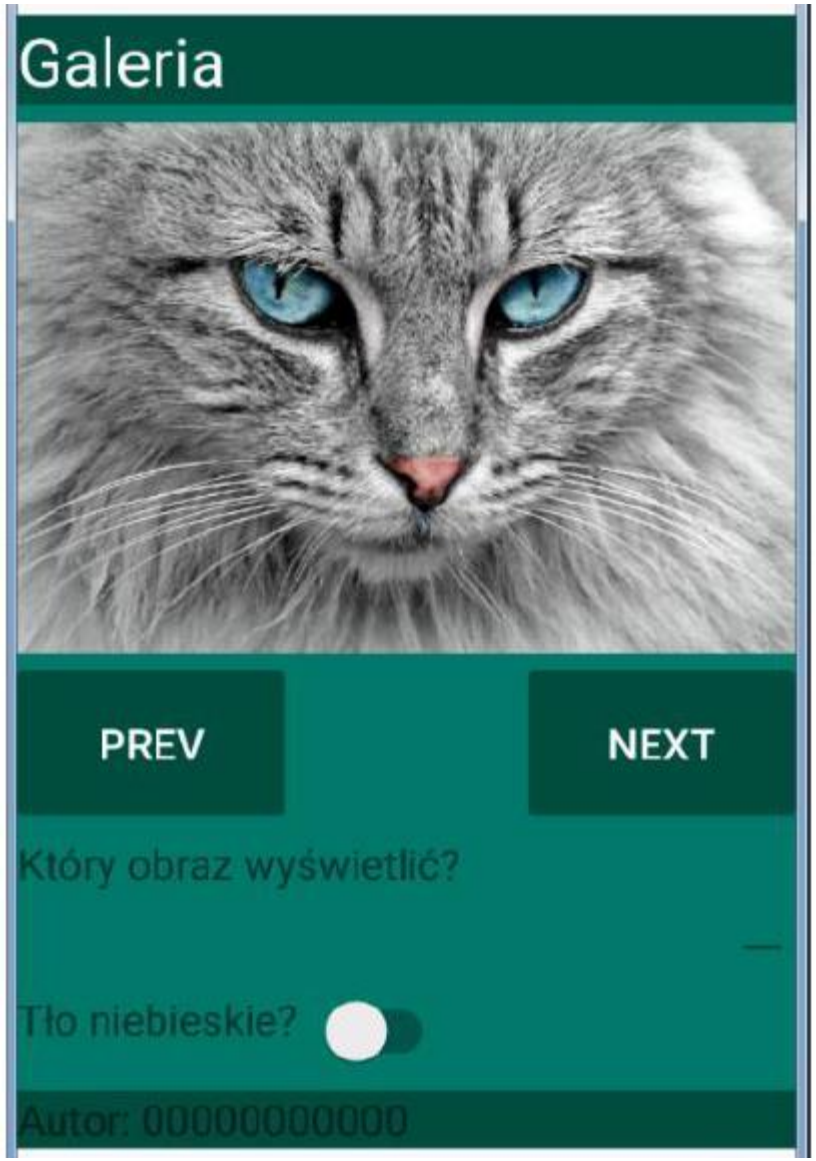
picture
plik z zasobami




```
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    QPixmap picture(":/img/img/mazury.jpg");
    int labelWidth = ui->label->width();
    int labelHeight = ui->label->height();
    ui->label->setPixmap(picture.scaled(labelWidth,labelHeight,Qt::KeepAspectRatio));
}
```

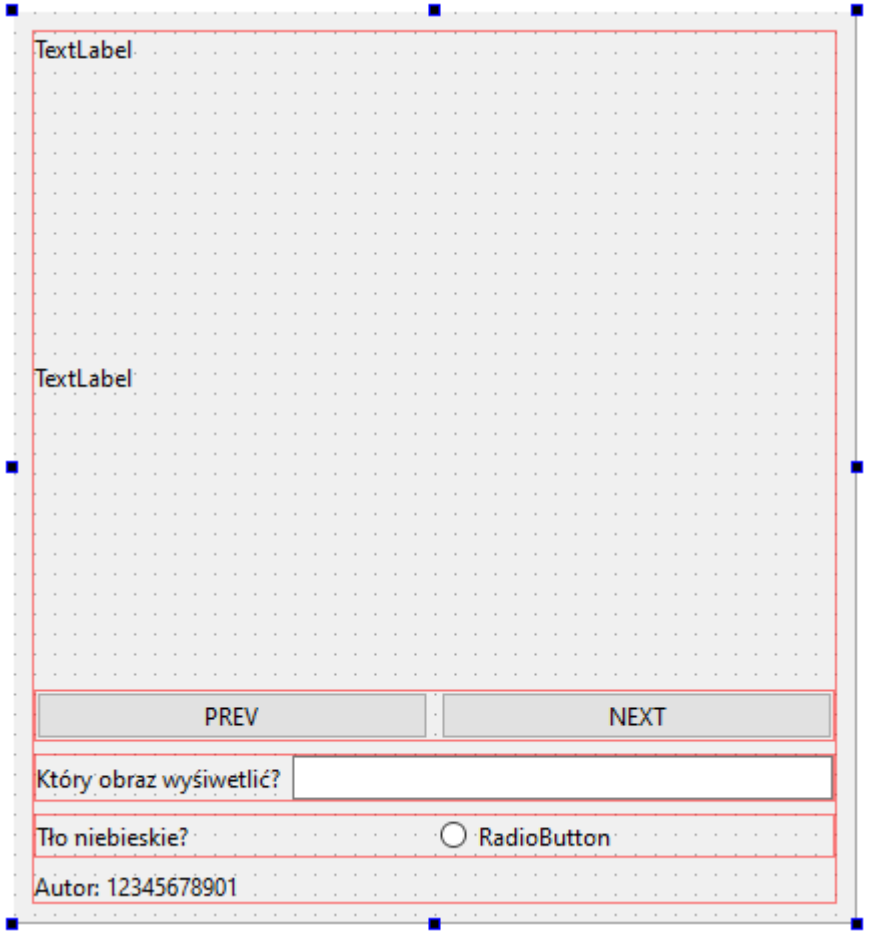
kopiujemy ścieżkę do zasobu i wklejamy na miejsce ścieżki bezwzględnej.

galeria


Galeria
PREV
NEXT
Który obraz wyświetlić?
Tło niebieskie?
Autor: 00000000000

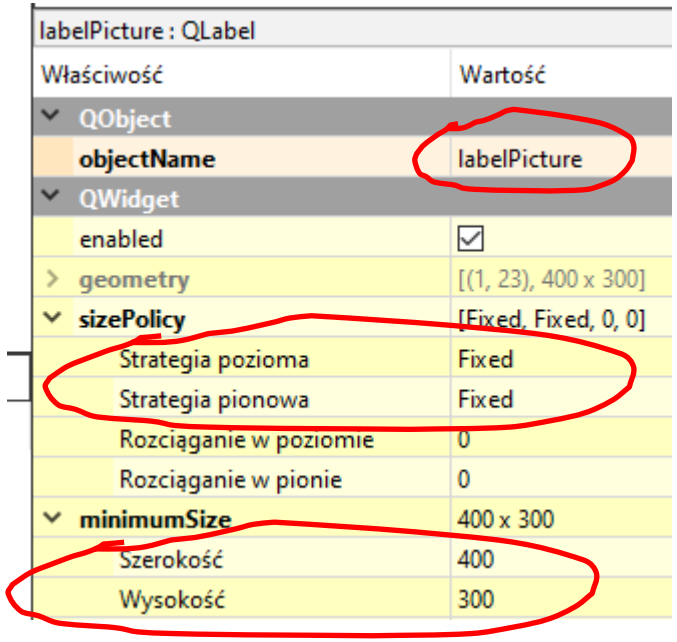

MainWindow
Galeria
PREV
NEXT
Który obraz wyświetlić?
Tło niebieskie?
tło zielone
Autor: 12345678901

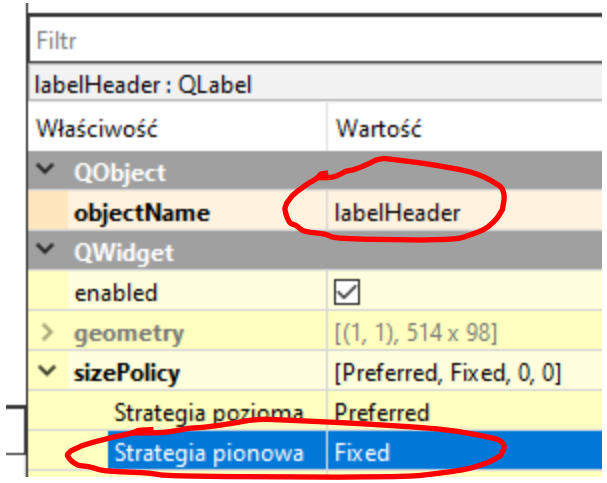
galeria



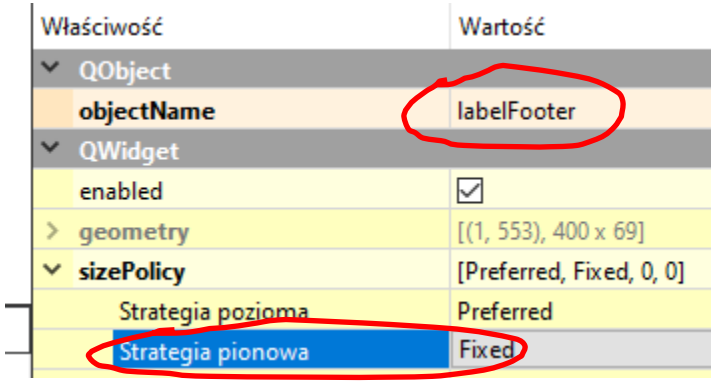
Obiekt	Klasa
MainWindow	QMainWindow
centralwidget	QWidget
verticalLayout	QVBoxLayout
horizontalLayout	QHBoxLayout
pushButtonNext	QPushButton
pushButtonPrev	QPushButton
horizontalLayout_2	QHBoxLayout
labelWhichPicture	QLabel
lineEditWhichPicture	QLineEdit
horizontalLayout_3	QHBoxLayout
labelBackground	QLabel
radioButtonBackground	QRadioButton
labelFooter	QLabel
labelHeader	QLabel
labelPicture	QLabel

galeria

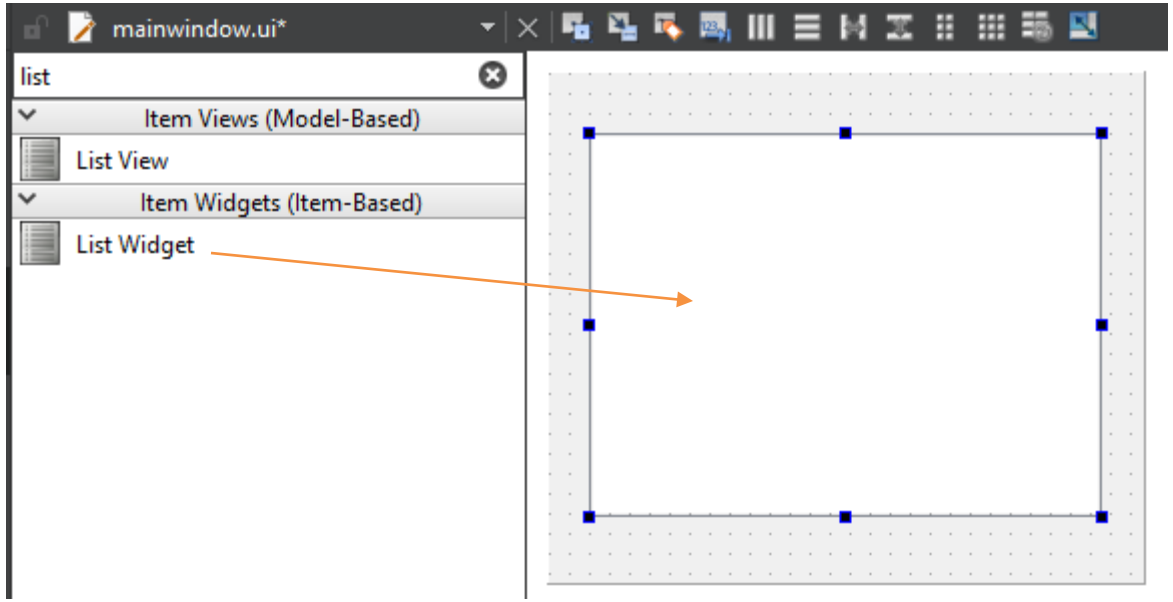




przykładowa konfiguracja labeli

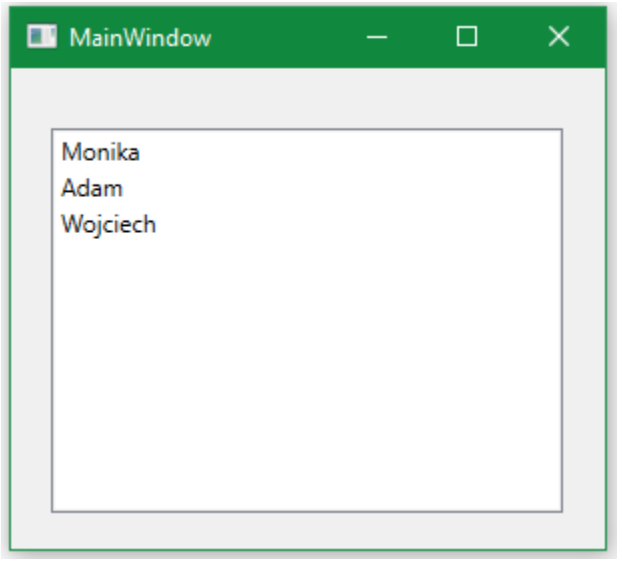


List Widget



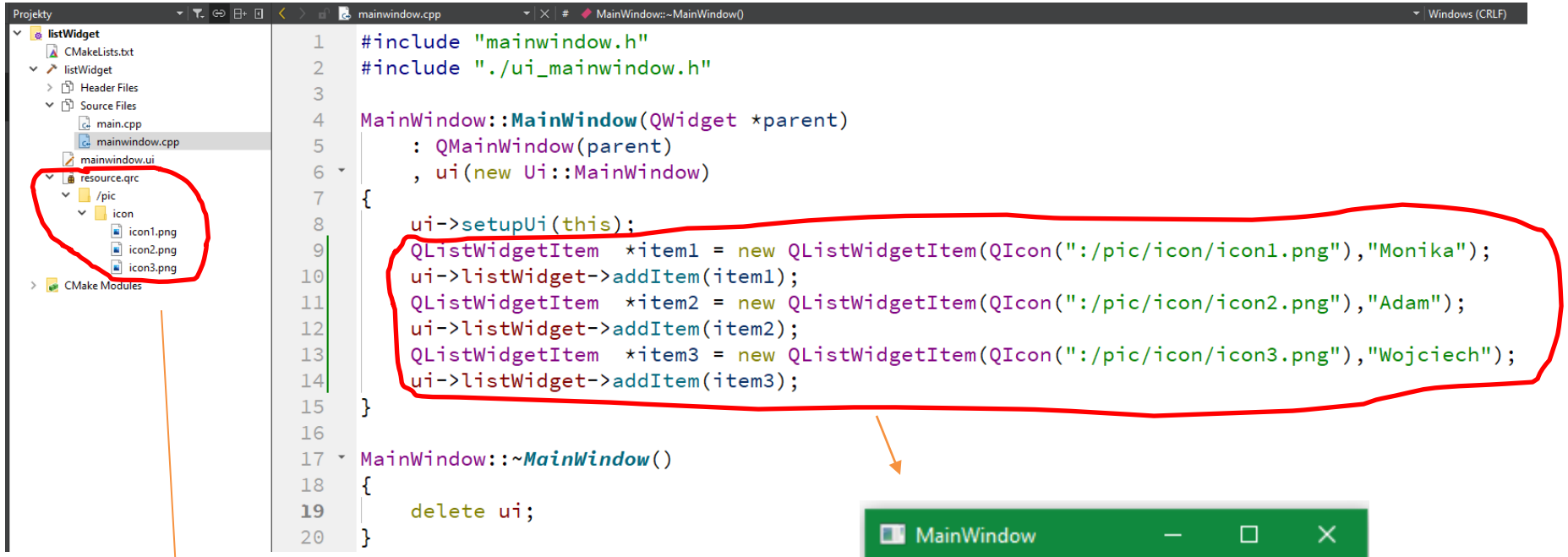
https://www.youtube.com/watch?v=2YRAJt-LbkM&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=20

List Widget



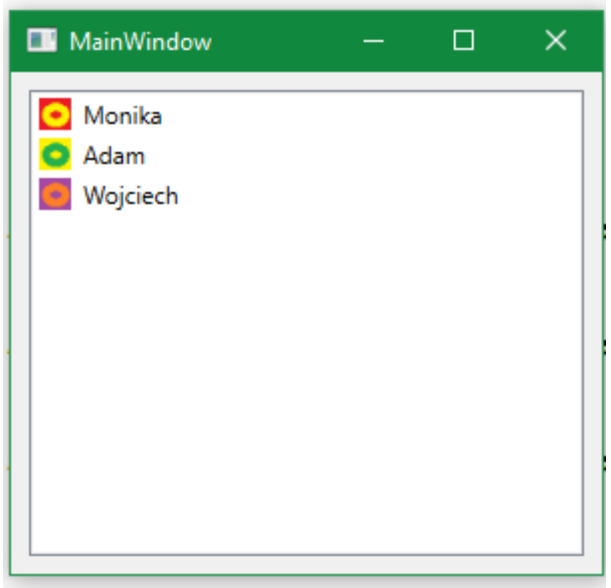
lista z imionami

List Widget



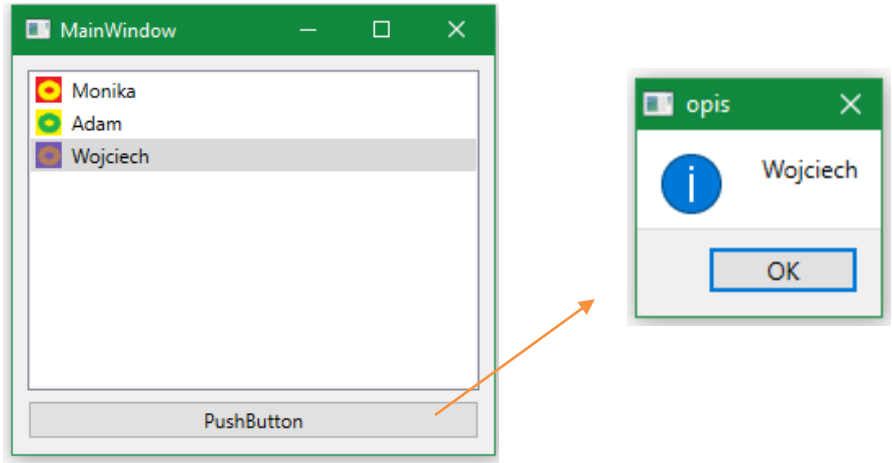
plik z zasobami w postaci ikonek

lista z imionami
oraz ikonami



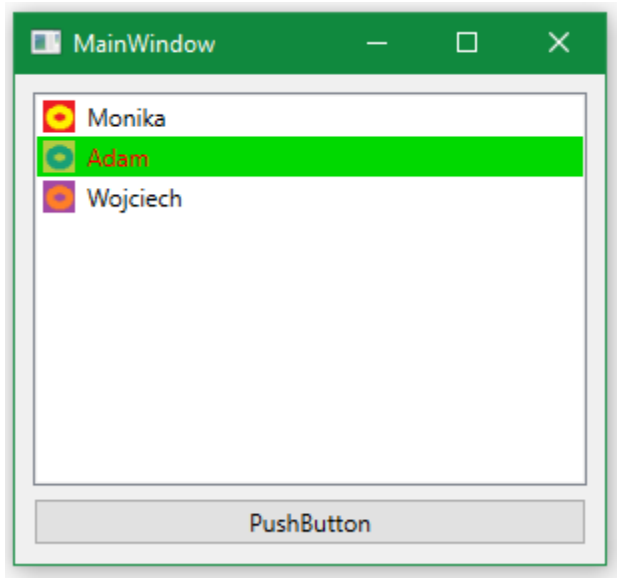
List Widget

po wybraniu imienia i kliknięciu na przycisk pojawia się okienko informacyjne z wybranym imieniem



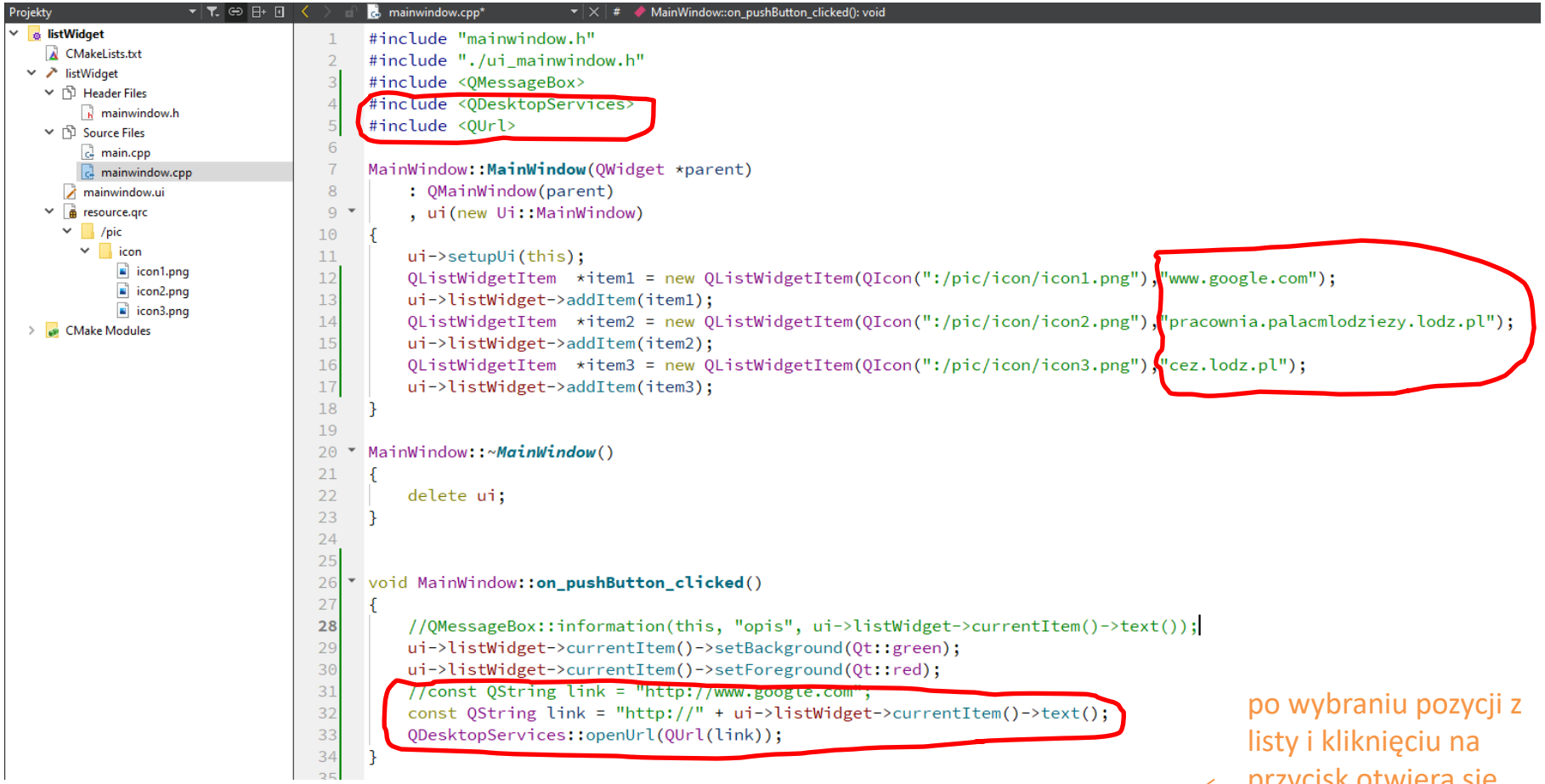
List Widget

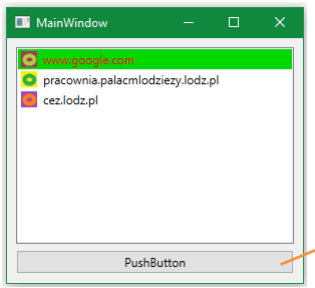
```
23
24  void MainWindow::on_pushButton_clicked()
25  {
26      //QMessageBox::information(this, "opis", ui->listWidget->currentItem()->text());
27      ui->listWidget->currentItem()->setBackground(Qt::green);
28      ui->listWidget->currentItem()->setForeground(Qt::red);
29  }
30
31
```

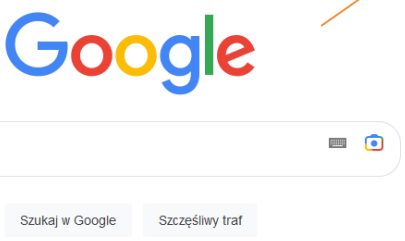


po wybraniu imienia i kliknięciu
na przycisk zmieniają się kolory
tła i czcionki wybranego imienia

List Widget



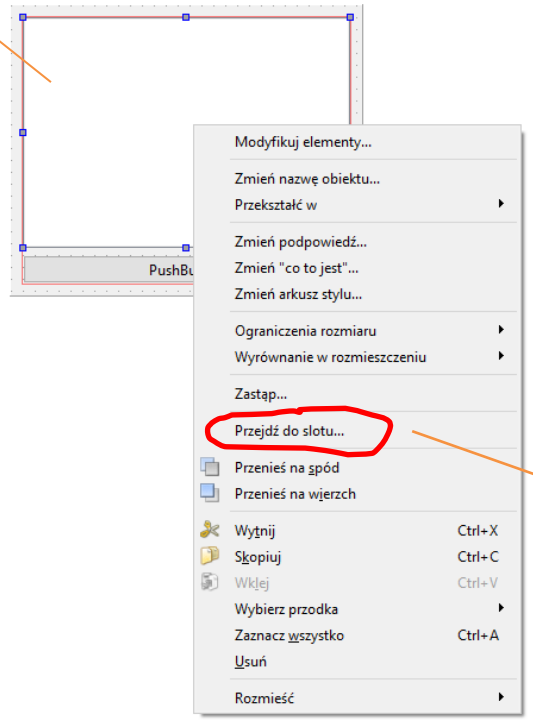




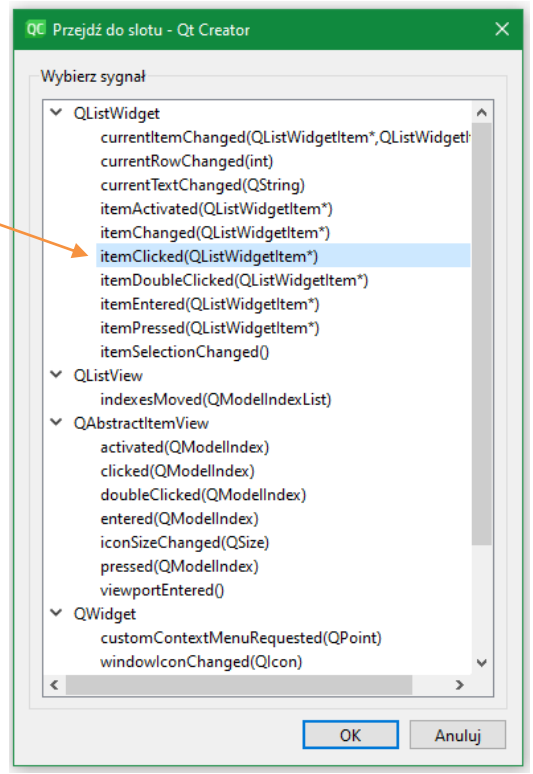
po wybraniu pozycji z listy i kliknięciu na przycisk otwiera się przeglądarka z wybranym adresem

List Widget

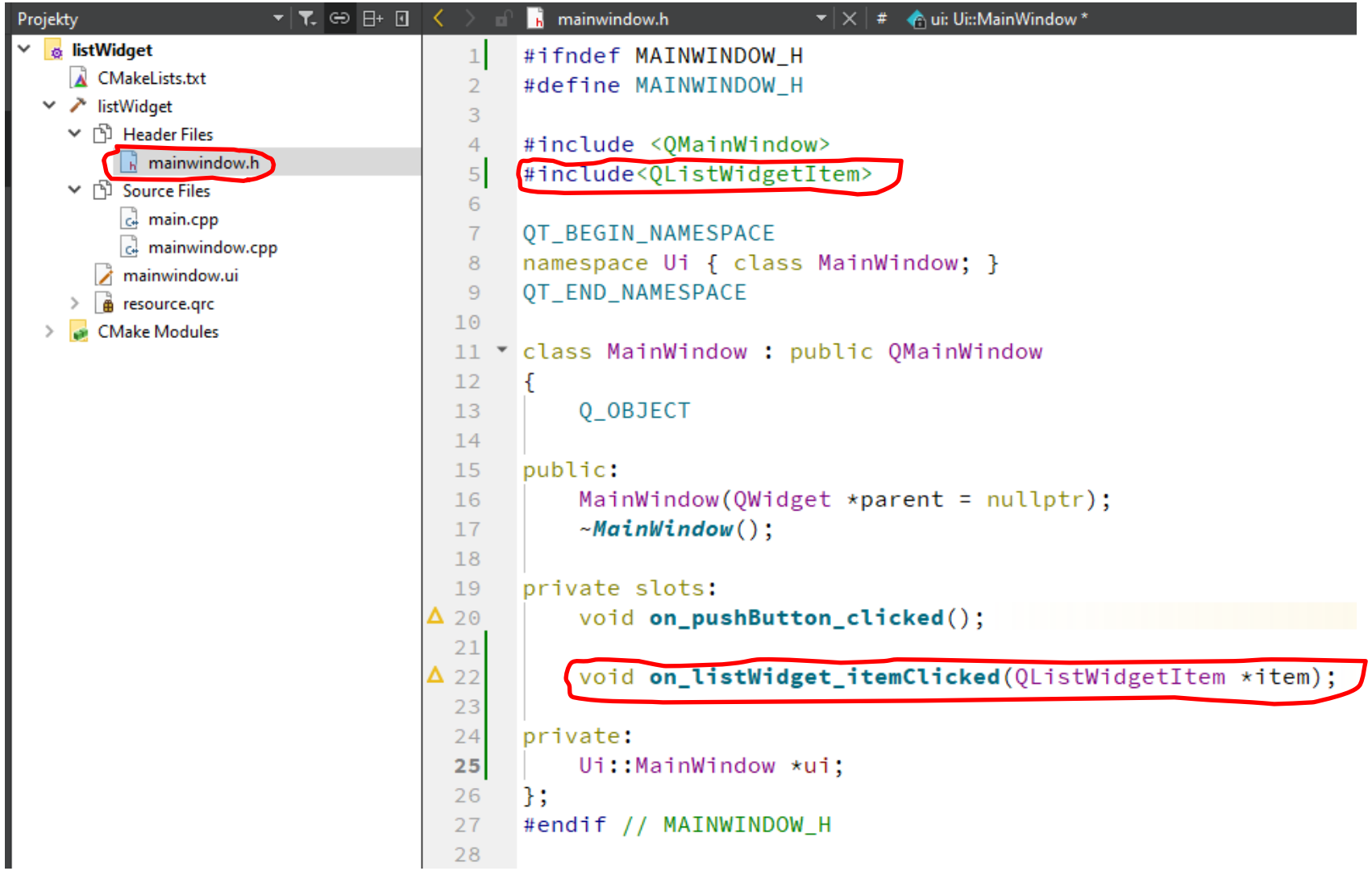
listWidget



wybieramy slot:
itemClicked()



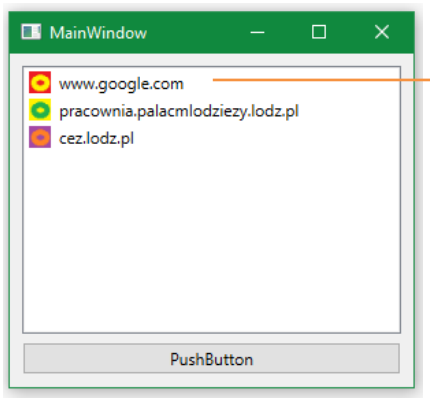
List Widget

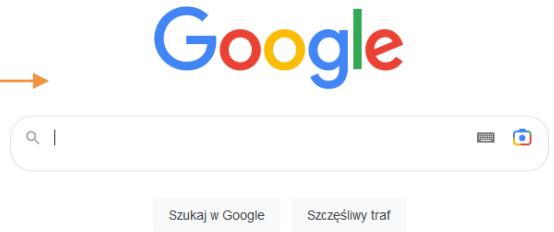

Projekty
listWidget
CMakeLists.txt
listWidget
Header Files
mainwindow.h
Source Files
main.cpp
mainwindow.cpp
mainwindow.ui
resource.qrc
CMake Modules
mainwindow.h
ui: Ui::MainWindow *
#ifndef MAINWINDOW_H
#define MAINWINDOW_H
#include <QMainWindow>
#include<QListWidgetItem>
QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE
class MainWindow : public QMainWindow
{
Q_OBJECT
public:
MainWindow(QWidget *parent = nullptr);
~MainWindow();
private slots:
void on_pushButton_clicked();
void on_listWidget_itemClicked(QListWidgetItem *item);
private:
Ui::MainWindow *ui;
};
#endif // MAINWINDOW_H

List Widget

w pliku mainwindow.cpp
dodajemy kod obsługujący
kliknięcie na pozycję listy

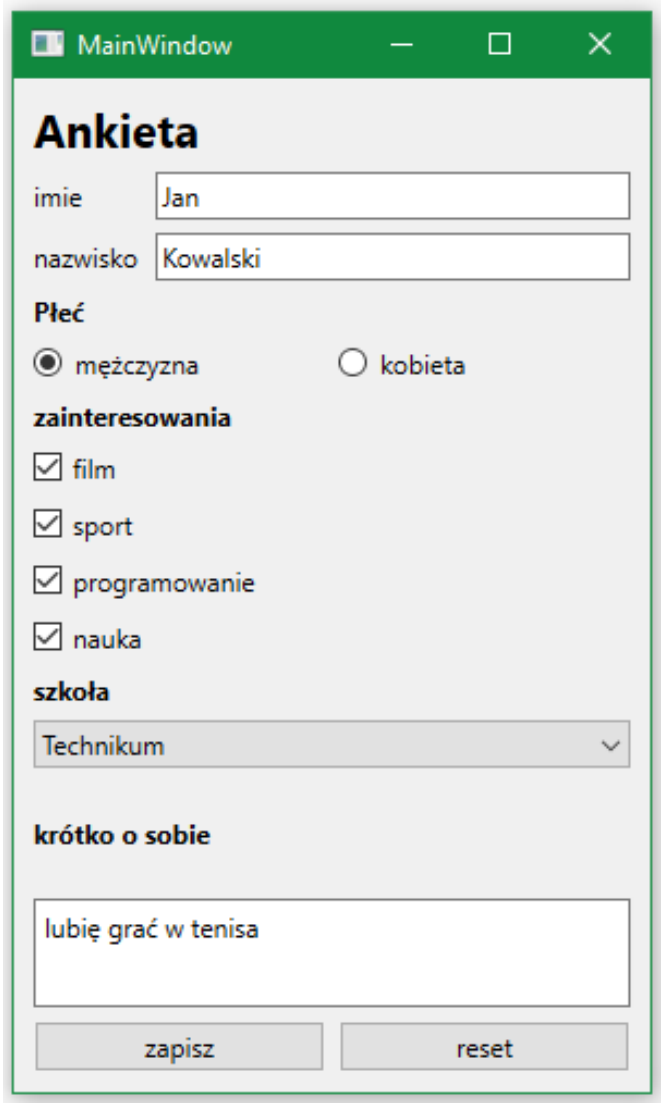
```
35
36  void MainWindow::on_listWidget_itemClicked(QListWidgetItem *item)
37  {
38      const QString link = "http://" + item->text();
39      QDesktopServices::openUrl(QUrl(link));
40  }
```

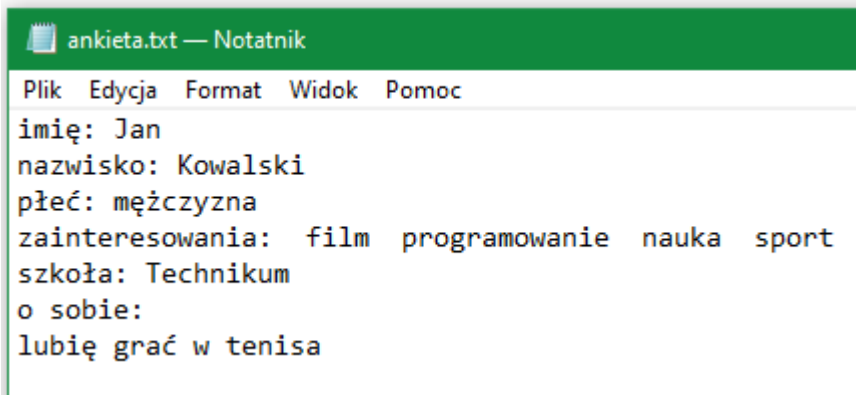




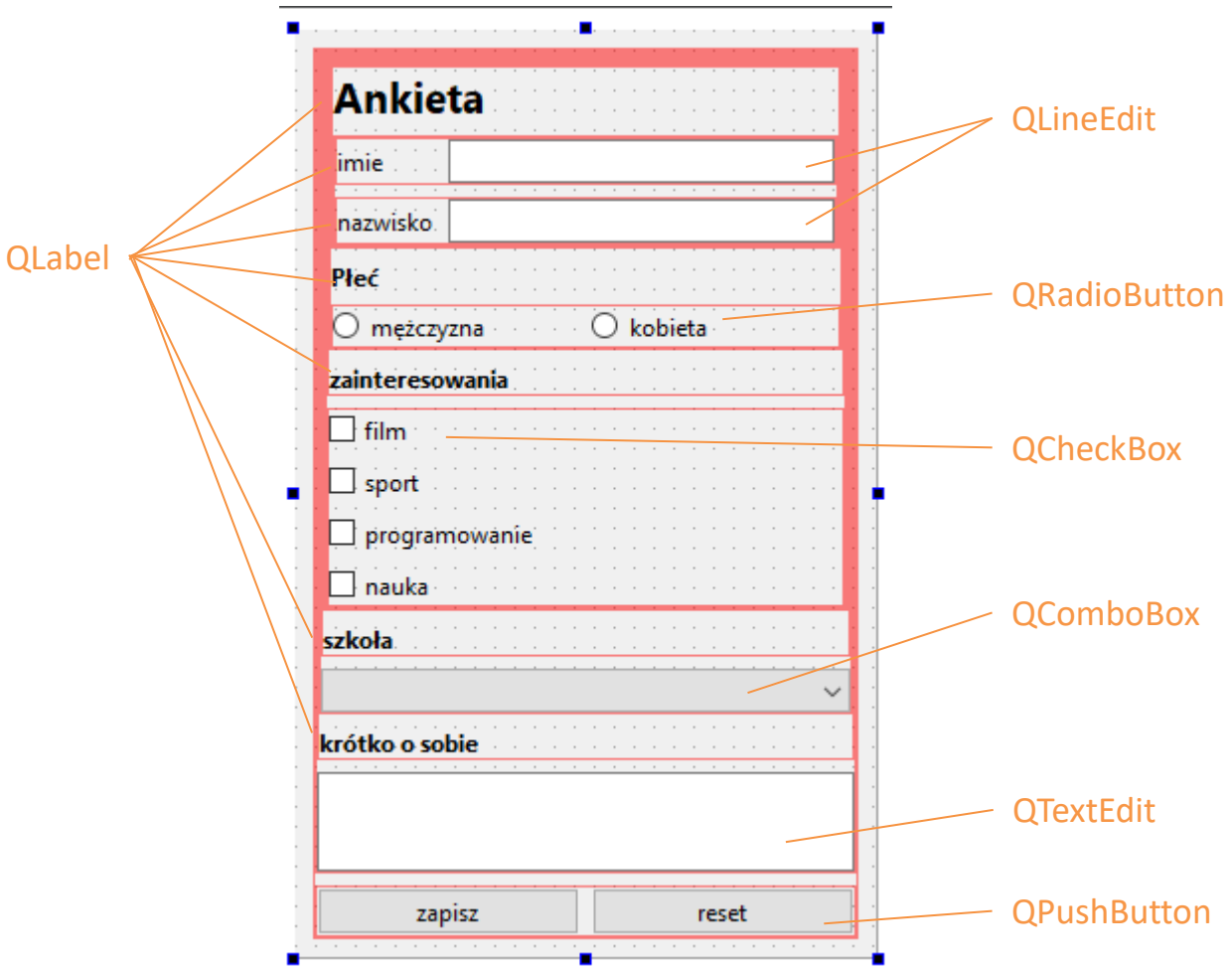
po kliknięciu na
wybranym adresie
otwiera się
przeglądarka

Ankieta


MainWindow
Ankieta
imie
Jan
nazwisko
Kowalski
Płeć
mężczyzna
kobieta
zainteresowania
film
sport
programowanie
nauka
szkoła
Technikum
krótko o sobie
lubię grać w tenisa
zapisz
reset


ankieta.txt — Notatnik
Plik Edycja Format Widok Pomoc
imię: Jan
nazwisko: Kowalski
płeć: mężczyzna
zainteresowania: film programowanie nauka sport
szkoła: Technikum
o sobie:
lubię grać w tenisa

Ankieta


Ankieta
imie
nazwisko
Płeć
mężczyzna
kobieta
zainteresowania
film
sport
programowanie
nauka
szkoła
krótko o sobie
zapisz
reset
QLabel
QLineEdit
QRadioButton
QCheckBox
QComboBox
QTextEdit
QPushButton

Ankieta

mainwindow.cpp

```cpp
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QFile>
#include <QFileDialog>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->comboBoxSchool->addItem("Podstawowa");
    ui->comboBoxSchool->addItem("Liceaum");
    ui->comboBoxSchool->addItem("Technikum");
    ui->comboBoxSchool->addItem("Politechnika");
    ui->comboBoxSchool->addItem("Uniwersytet");
}
```

w konstruktorze klasy MainWindow inicjalizujemy comboBoxSchool poszczególnymi pozycjami

Ankieta

mainwindow.cpp

```cpp
void MainWindow::on_pushButtonSave_clicked()
{
    // Dialog zapisu pliku
    QString filename= QFileDialog::getSaveFileName(this, "Save As", "ankieta.txt");

    QFile file(filename);

    // Jeśli nie powiedzie się otwarcie pliku kończymy
    if (!file.open(QIODevice::WriteOnly | QIODevice::Text))
        return;

    QString plec = "";

    if (ui->radioButtonMan->isChecked()){
            plec = "mężczyzna";
        }else{
            plec = "kobieta";

        }

    QString zainteresowania = "";

    if (ui->checkBoxFilm->isChecked()){
        zainteresowania += " film ";
    }
    if (ui->checkBoxProgramming->isChecked()){
        zainteresowania += " programowanie ";
    }
    if (ui->checkBoxScience->isChecked()){
        zainteresowania += " nauka ";
    }
    if (ui->checkBoxSport->isChecked()){
        zainteresowania += " sport ";
    }

    QString szkola = ui->comboBoxSchool->currentText();

    QString oSobie =  ui->textEditAboutYou->toPlainText();

    QTextStream out(&file);

    out << "imię: " + ui->lineEditName->text() +
           "\nnazwisko: " + ui->lineEditSurname->text() +
           "\npłeć: " + plec +
           "\nzainteresowania: " + zainteresowania +
           "\nszkoła: " + szkola +
           "\no sobie: \n" + oSobie;

    file.close();
}
```

pobieramy wpisy z ankiety i zapisujemy do pliku ankieta.txt

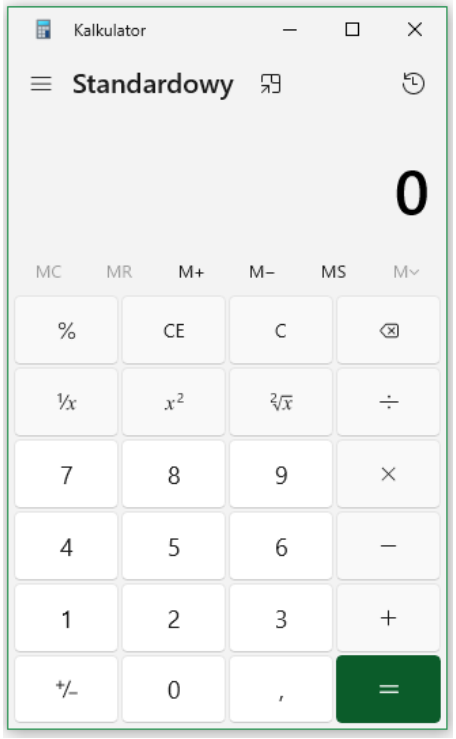
Ankieta

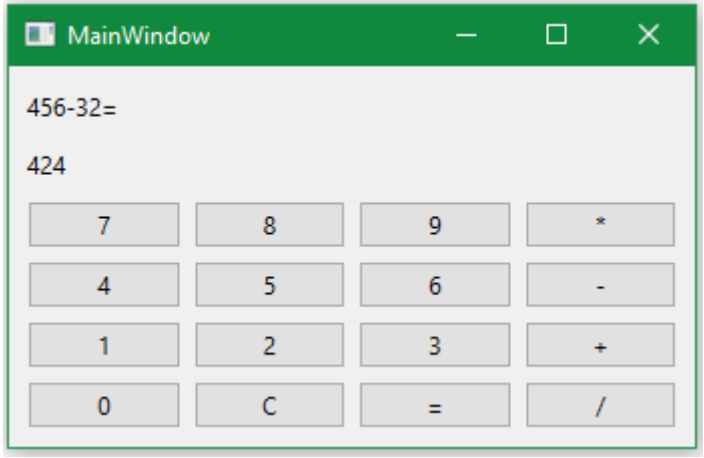
mainwindow.cpp

reset ankiety

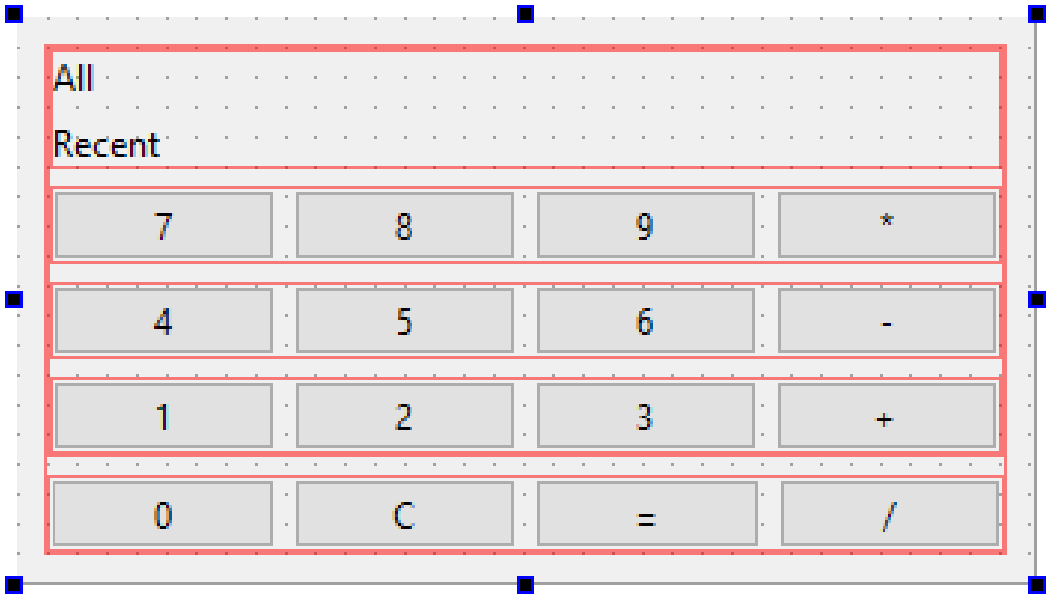
```
76  void MainWindow::on_pushButtonReset_clicked()
77  {
78      ui->lineEditName->setText("");
79      ui->lineEditSurname->setText("");
80
81      ui->radioButtonMan->setAutoExclusive(false);
82      ui->radioButtonMan->setChecked(false);
83      ui->radioButtonMan->setAutoExclusive(true);
84
85      ui->radioButtonWoman->setAutoExclusive(false);
86      ui->radioButtonWoman->setChecked(false);
87      ui->radioButtonWoman->setAutoExclusive(true);
88
89      ui->checkBoxFilm->setChecked(false);
90      ui->checkBoxProgramming->setChecked(false);
91      ui->checkBoxScience->setChecked(false);
92      ui->checkBoxSport->setChecked(false);
93
94      ui->comboBoxSchool->setCurrentIndex(0);
95
96      ui->textEditAboutYou->clear();
97  }
```

Kalkulator

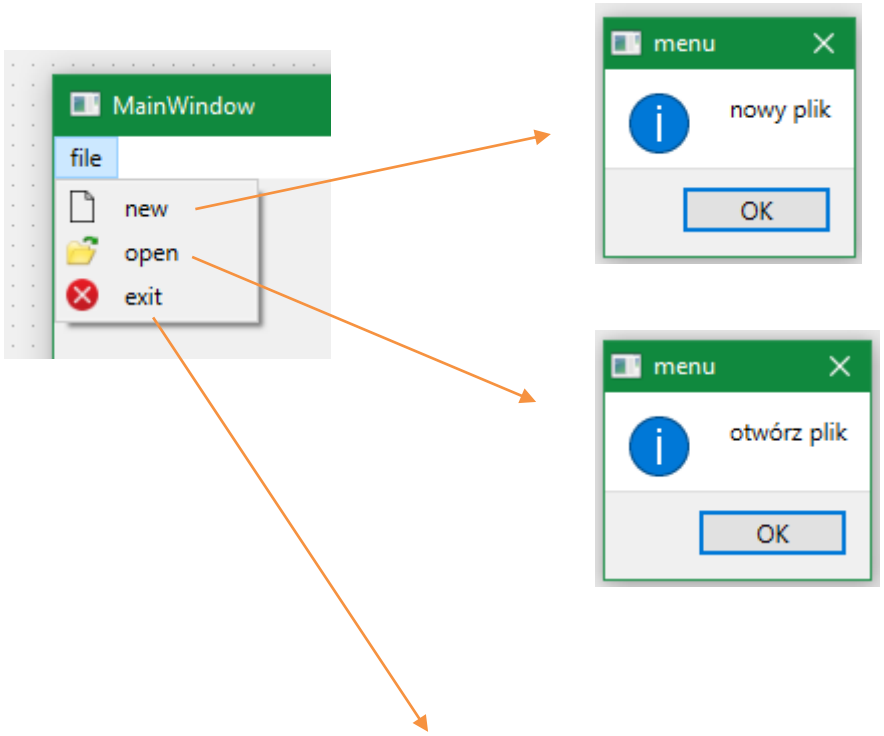

Kalkulator
Standardowy
0
MC MR M+ M− MS M⌄
% CE C ⌫
1/x x² ²√x ÷
7 8 9 ×
4 5 6 −
1 2 3 +
+/− 0 , =


MainWindow
456-32=
424
7 8 9 *
4 5 6 -
1 2 3 +
0 C = /

Kalkulator



Menu

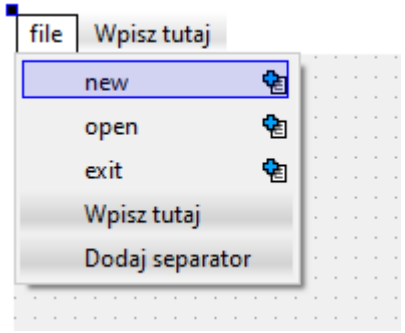


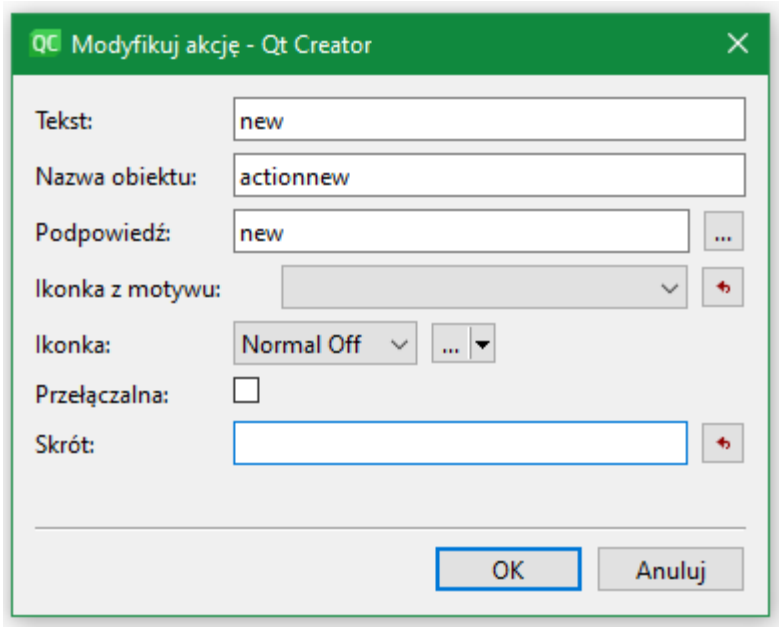
https://www.youtube.com/watch?v=E9eQO5czNrw&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=18

Menu

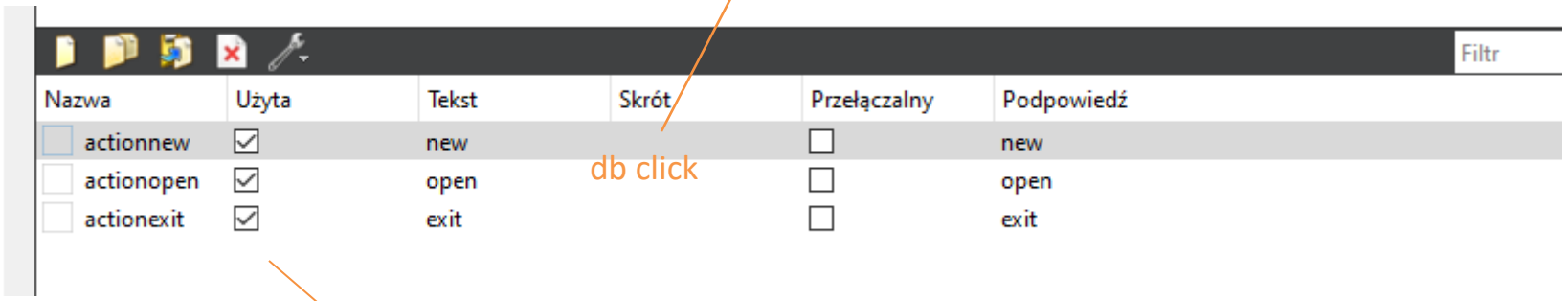
okienko modyfikacji akcji

Każdy nowy projekt ma już kontrolkę QMenuBar



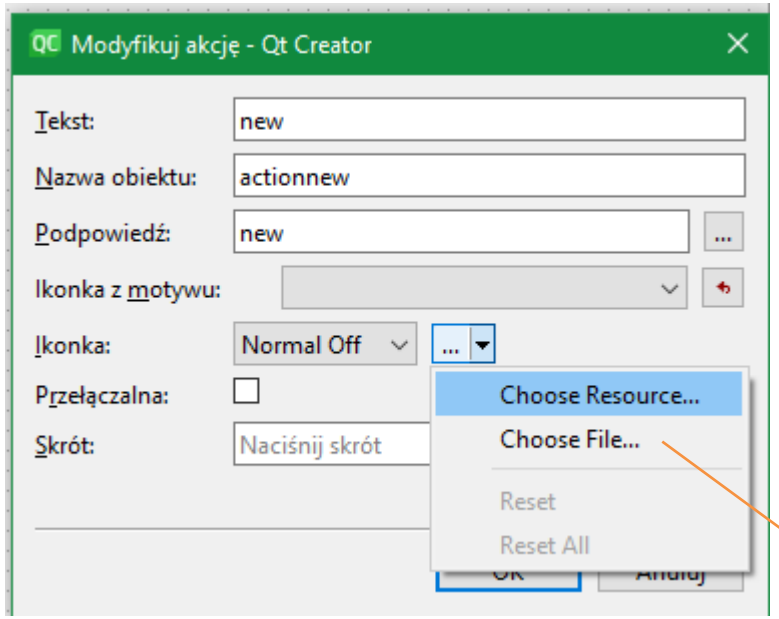


db click

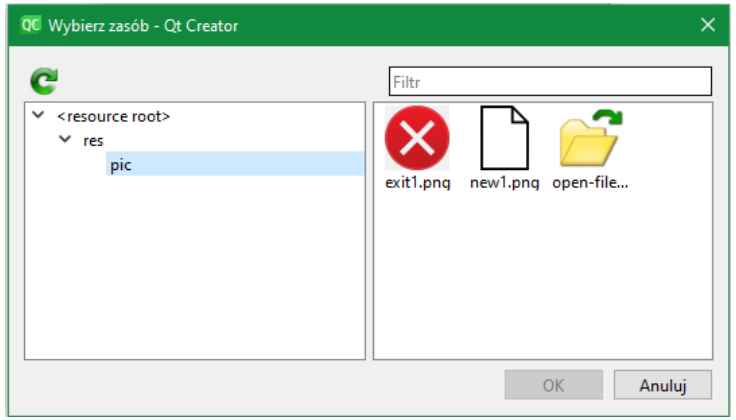


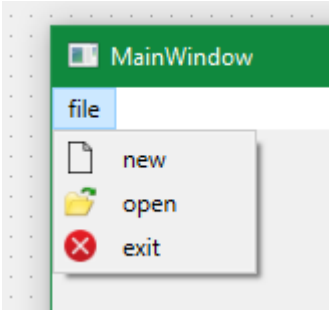
do każdej pozycji menu jest przypisana akcja

Menu



wybieranie ikonki z zasobów
(które najpierw należy utworzyć)

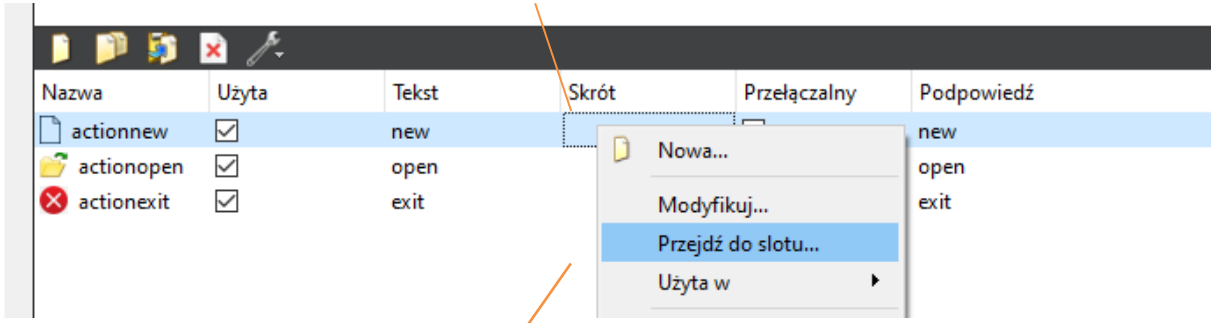


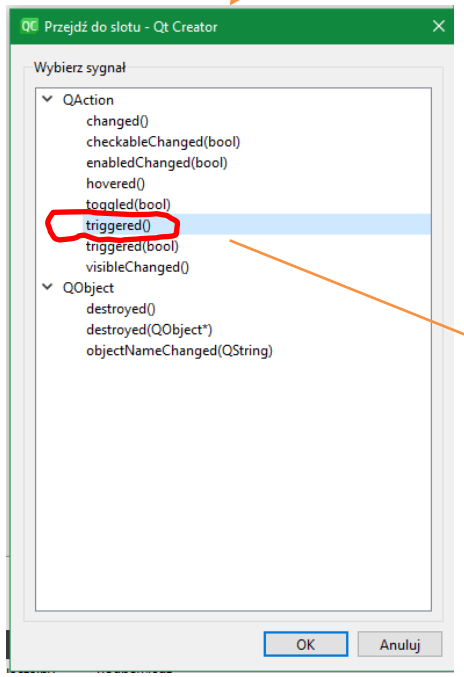


Menu

dodawanie akcji

prawy myszy






```cpp
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}

MainWindow::~MainWindow()
{
    delete ui;
}


void MainWindow::on_actionnew_triggered()
{
    QMessageBox::information(this, "menu", "nowy plik");
}


void MainWindow::on_actionopen_triggered()
{
    QMessageBox::information(this, "menu", "otwórz plik");
}


void MainWindow::on_actionexit_triggered()
{
    QApplication::quit();
}
```

QDir

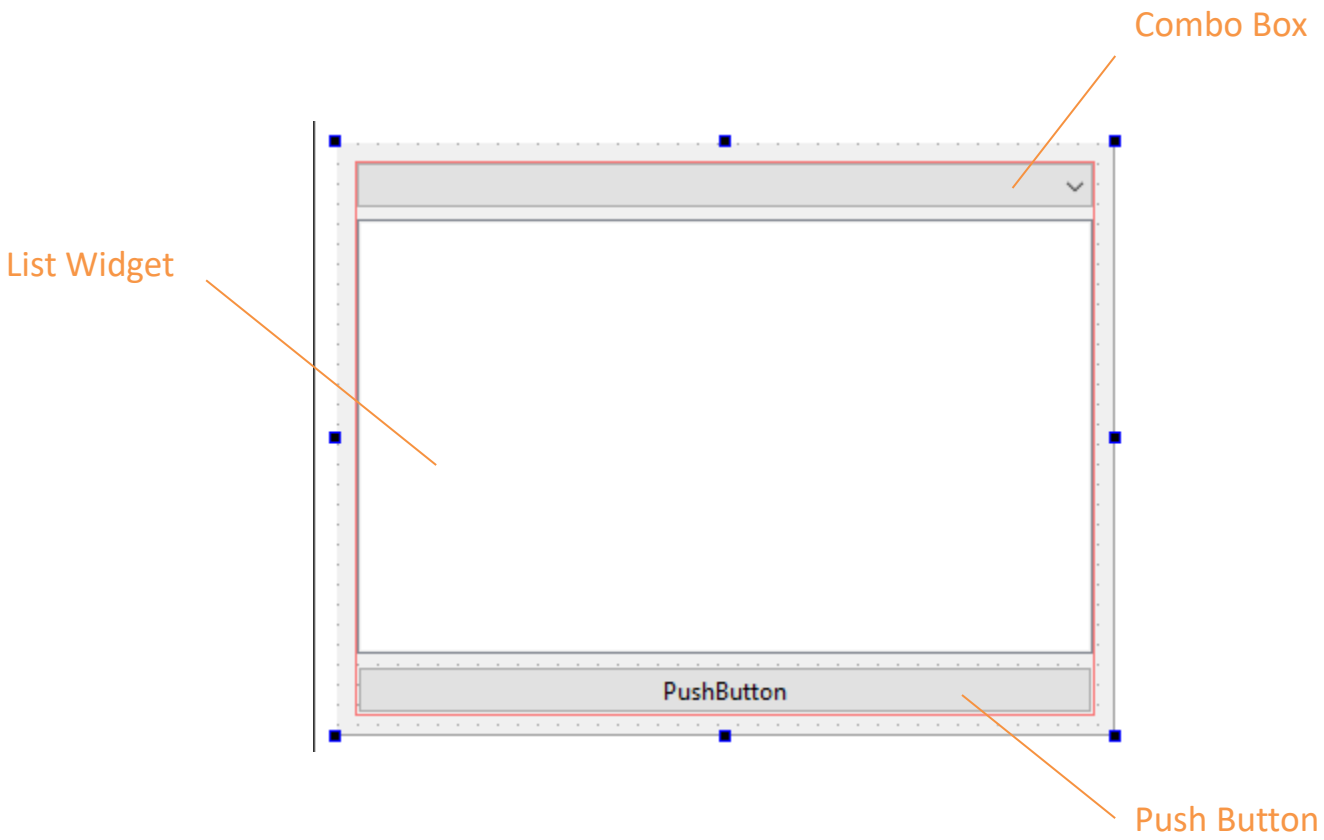
mainwindow.cpp

```cpp
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QDir>
#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    QDir dir("C:/komp");
    if (dir.exists()){
        QMessageBox::information(this,"Check","folder istnieje");
    }else{
        QMessageBox::information(this,"Check","folder nie istnieje");
    }
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

sprawdzenie czy folder c:/komp istnieje

https://www.youtube.com/watch?v=hstOtTEeW9Y&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=21

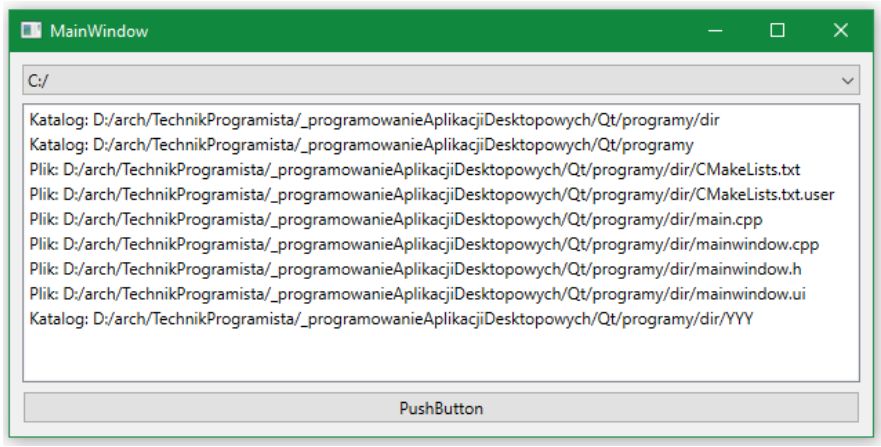
QDir



https://www.youtube.com/watch?v=hstOtTEeW9Y&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=21

QDir

```cpp
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QDir>
#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    // wstawienie liter dysków do ComboBox np: c:/
    QDir dir;
    foreach (QFileInfo var, dir.drives()) {
        ui->comboBox->addItem(var.absoluteFilePath());
    }

    // wylistowanie zawartości katalogu projektu
    QDir dir_2("D:/arch/TechnikProgramista/_programowanieAplikacjiDesktopowych/Qt/programy/dir");
    foreach (QFileInfo var, dir_2.entryInfoList()) {
        if(var.isDir()){
            ui->listWidget->addItem("Katalog: " + var.absoluteFilePath());
        }
        if(var.isFile()){
            ui->listWidget->addItem("Plik: " + var.absoluteFilePath());
        }
    }
}
```



https://www.youtube.com/watch?v=hstOtTEeW9Y&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=21

QDir

po naciśnięciu przycisku w katalogu projektu tworzony jest folder YYY (jeśli nie istnieje)

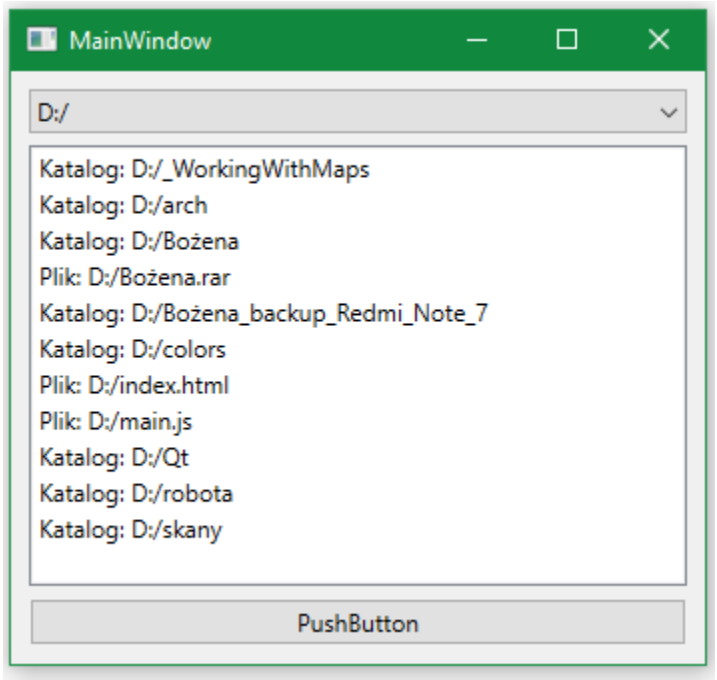
```
void MainWindow::on_pushButton_clicked()
{
    // tworzymy nowy folder w katalogu projektu
    QDir dir_3("D:/arch/TechnikProgramista/_programowanieAplikacjiDesktopowych/Qt/programy/dir/YYY");
    if(dir_3.exists()){
        QMessageBox::information(this,"Check","folder istnieje");
    }else{
        // tworzy folder YYY
        dir_3.mkpath("D:/arch/TechnikProgramista/_programowanieAplikacjiDesktopowych/Qt/programy/dir/YYY");
    }
}
```

https://www.youtube.com/watch?v=hstOtTEeW9Y&list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA&index=21

QDir

```
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QDir>
#include <QMessageBox>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    // wstawienie liter dysków do ComboBox np: c:/
    QDir dir;
    foreach (QFileInfo var, dir.drives()) {
        ui->comboBox->addItem(var.absoluteFilePath());
    }
    // wylistowanie zawartości pierwszego dysku np: c:/ (pierwszy na liście dysk ma index 0)
    comboBox_add_items(0);
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

po wybraniu litery dysku wypisują się pliki i katalogi zapisane na nim



metoda pomocnicza do wylistowania zawartości wybranego dysku (indeksy zaczynają się od 0)

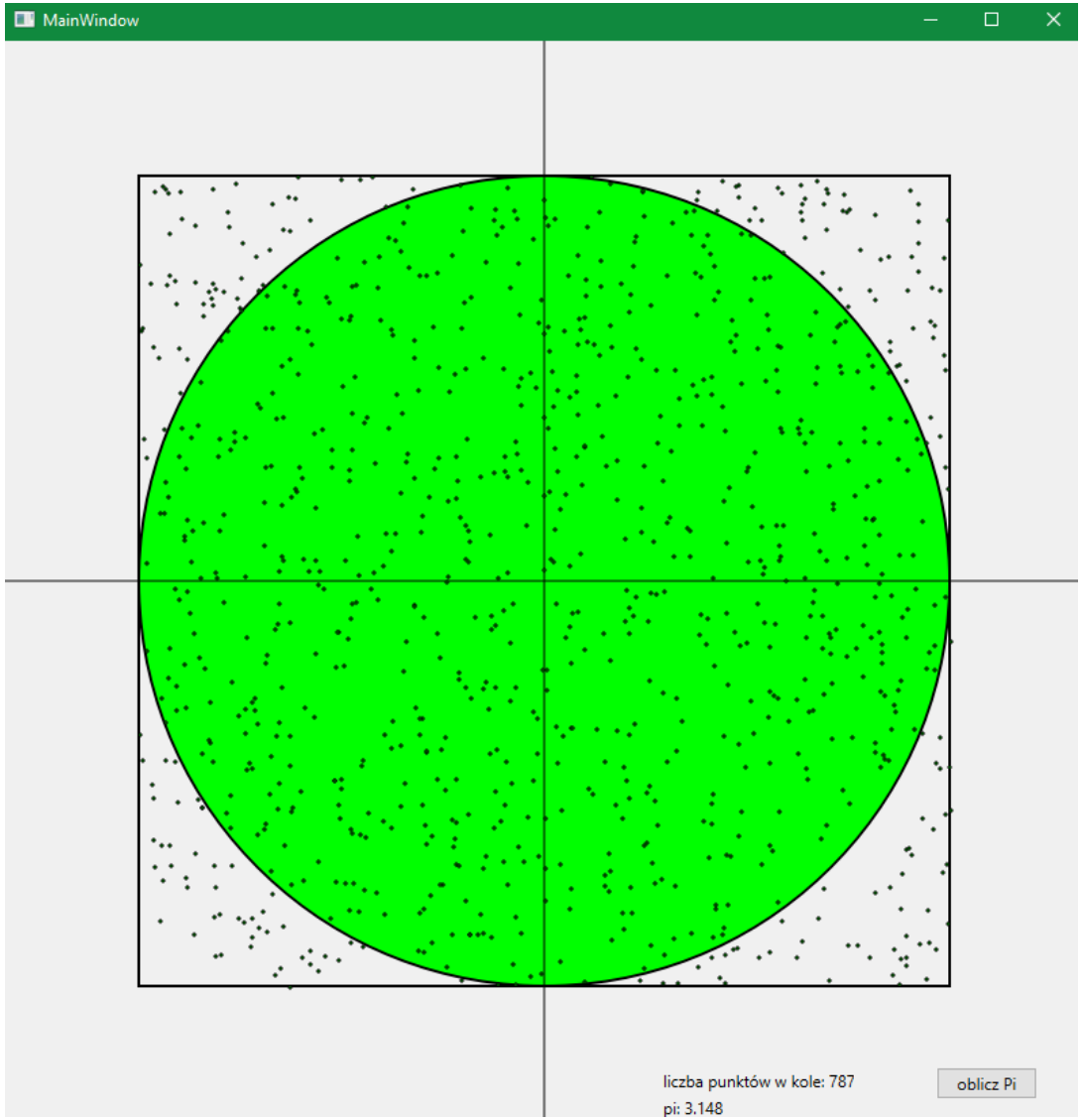
QDir

```
void MainWindow::on_pushButton_clicked()
{
    // tworzymy nowy folder w katalogu projektu
    QDir dir_3("D:/arch/TechnikProgramista/_programowanieAplikacjiDesktopowych/Qt/programy/dir/YYY");
    if(dir_3.exists()){
        QMessageBox::information(this,"Check","folder istnieje");
    }else{
        // tworzy folder YYY
        dir_3.mkpath("D:/arch/TechnikProgramista/_programowanieAplikacjiDesktopowych/Qt/programy/dir/YYY");
    }
}

// slot uruchamiany po wybraniu pozycji na comboBoxie
void MainWindow::on_comboBox_activated(int index)
{
    comboBox_add_items(index);
}

// metoda pomocnicza do wylistowania zawartości wybranego dysku (indeksy zaczynają się od 0)
void MainWindow::comboBox_add_items(int index){
    // czyszczenie zawartości listWidget (bez czyszczenia kolejne wybrane itemy dopisują się na końcu)
    ui->listWidget->clear();
    //QDir dir_2(ui->comboBox->currentText());
    QDir dir_2(ui->comboBox->itemText(index));
    foreach (QFileInfo var, dir_2.entryInfoList()) {
        if(var.isDir()){
            ui->listWidget->addItem("Katalog: " + var.absoluteFilePath());
        }
        if(var.isFile()){
            ui->listWidget->addItem("Plik: " + var.absoluteFilePath());
        }
    }
}
```

po wybraniu dysku w comboBoxie wywołujemy metodę pomocniczą do wylistowania jego zawartości.

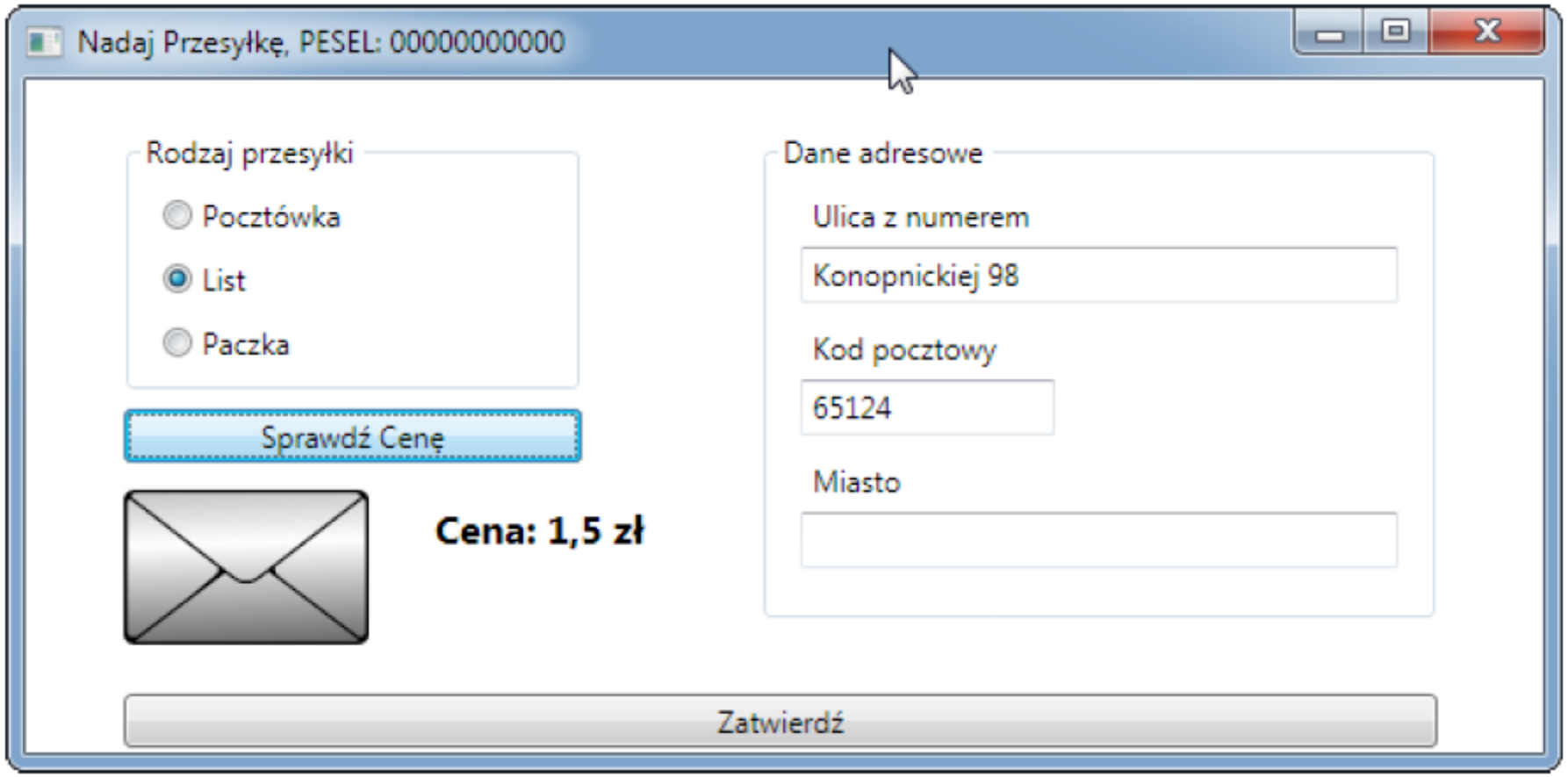
QPainter

obliczenie liczby π metodą Monte Carlo



http://programmingexamples.wikidot.com/qt-qpainter-example

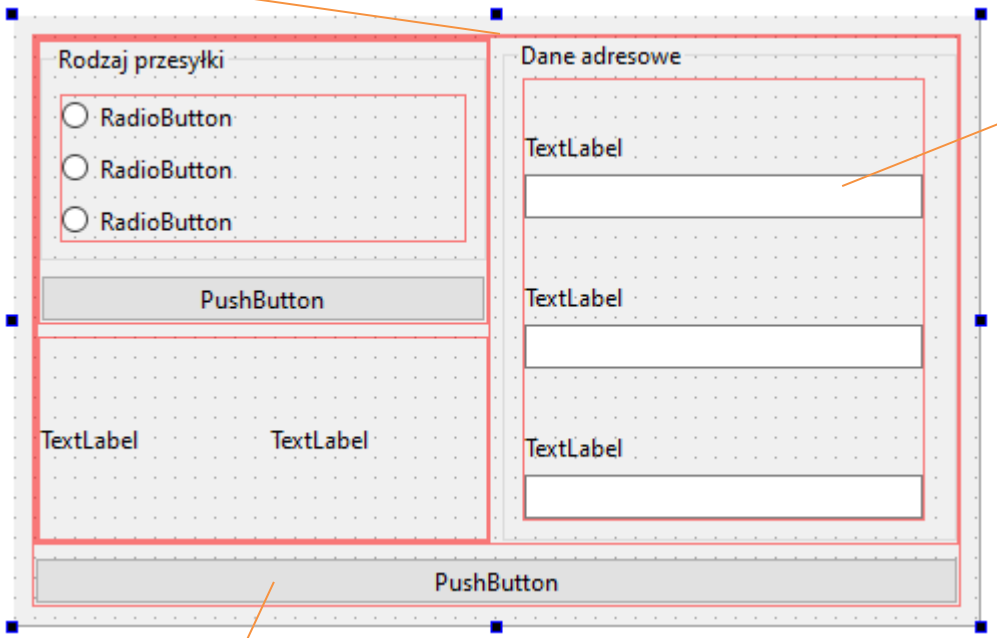
Obsługa poczty



poczta.zip

Obsługa poczty

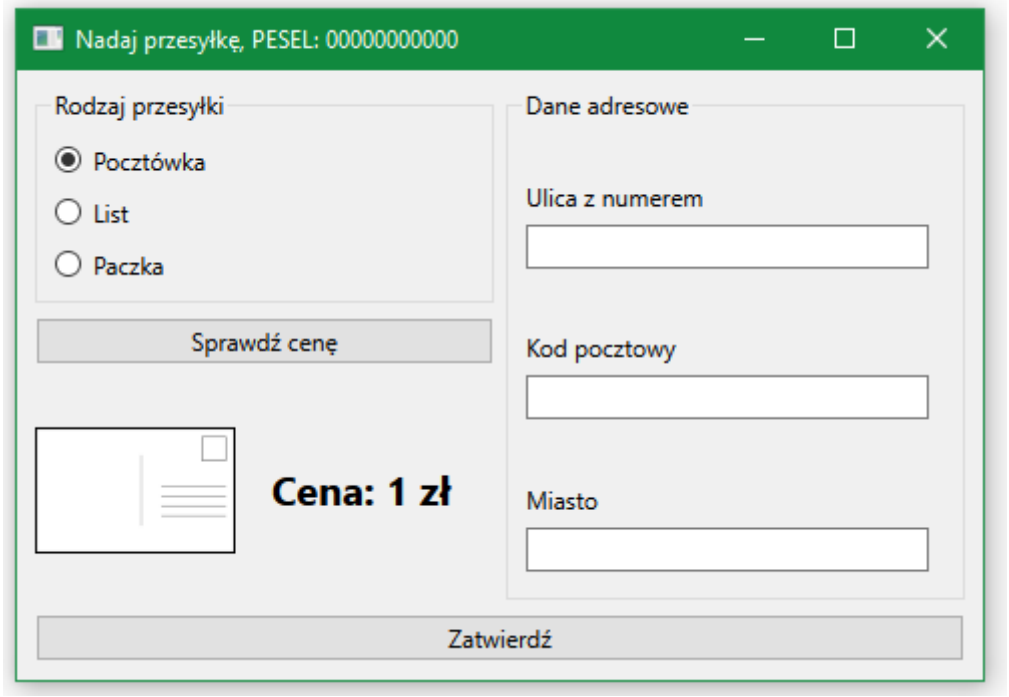
GroupBox



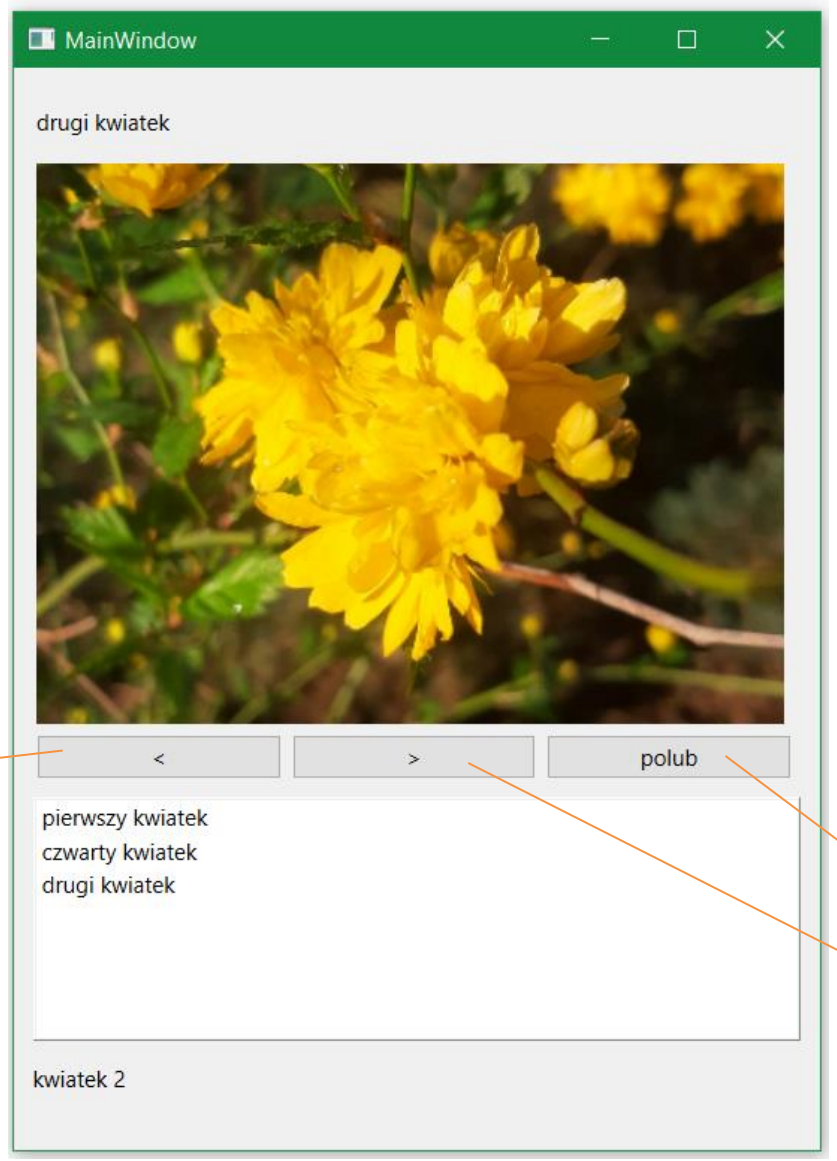
LineEdit

nazwy obiektów powinny być adekwatne do wykonywanej funkcji np, pushButton_confirm

Obsługa poczty



Galeria z polubieniami



nagłówek zdjęcia

zdjęcie

przycisk poprzednie zdjęcie

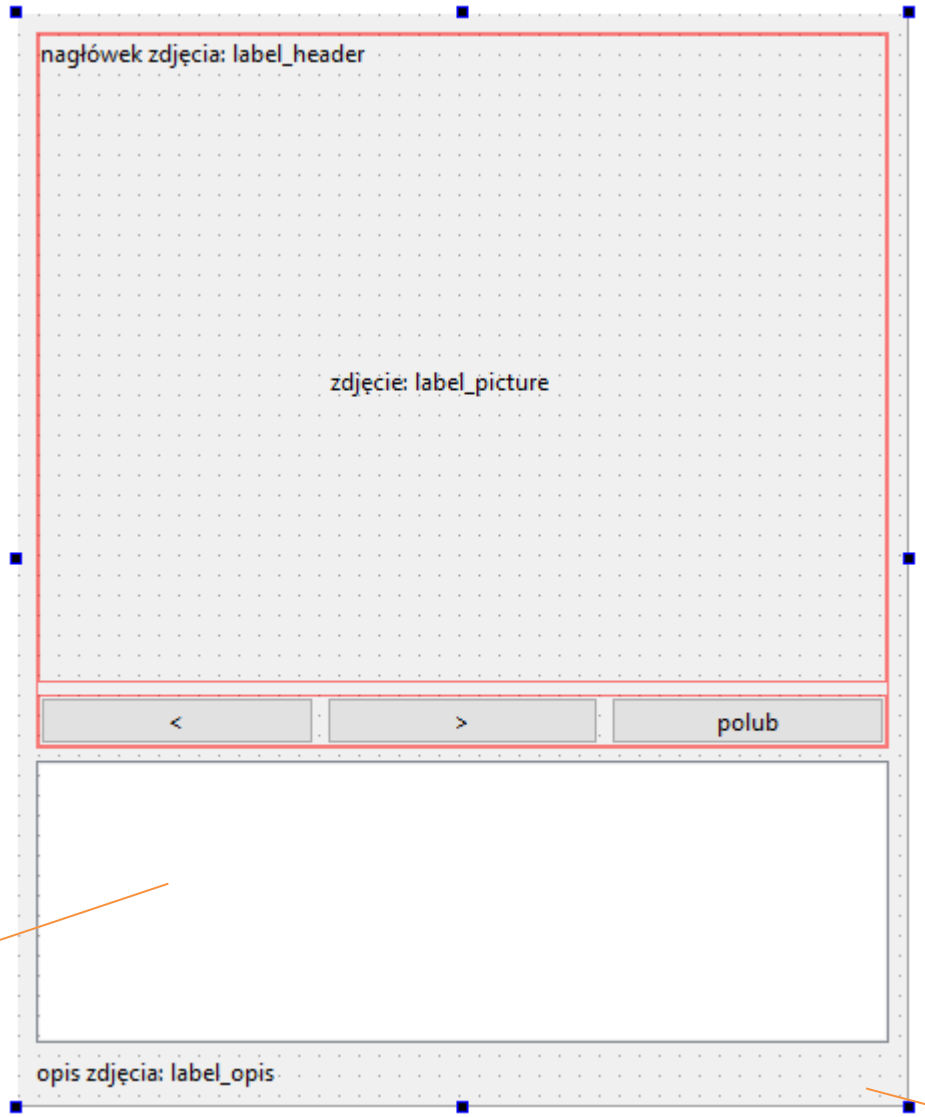
polubienia

przycisk polubienia

przycisk następne zdjęcie

opis zdjęcia

Galeria z polubieniami

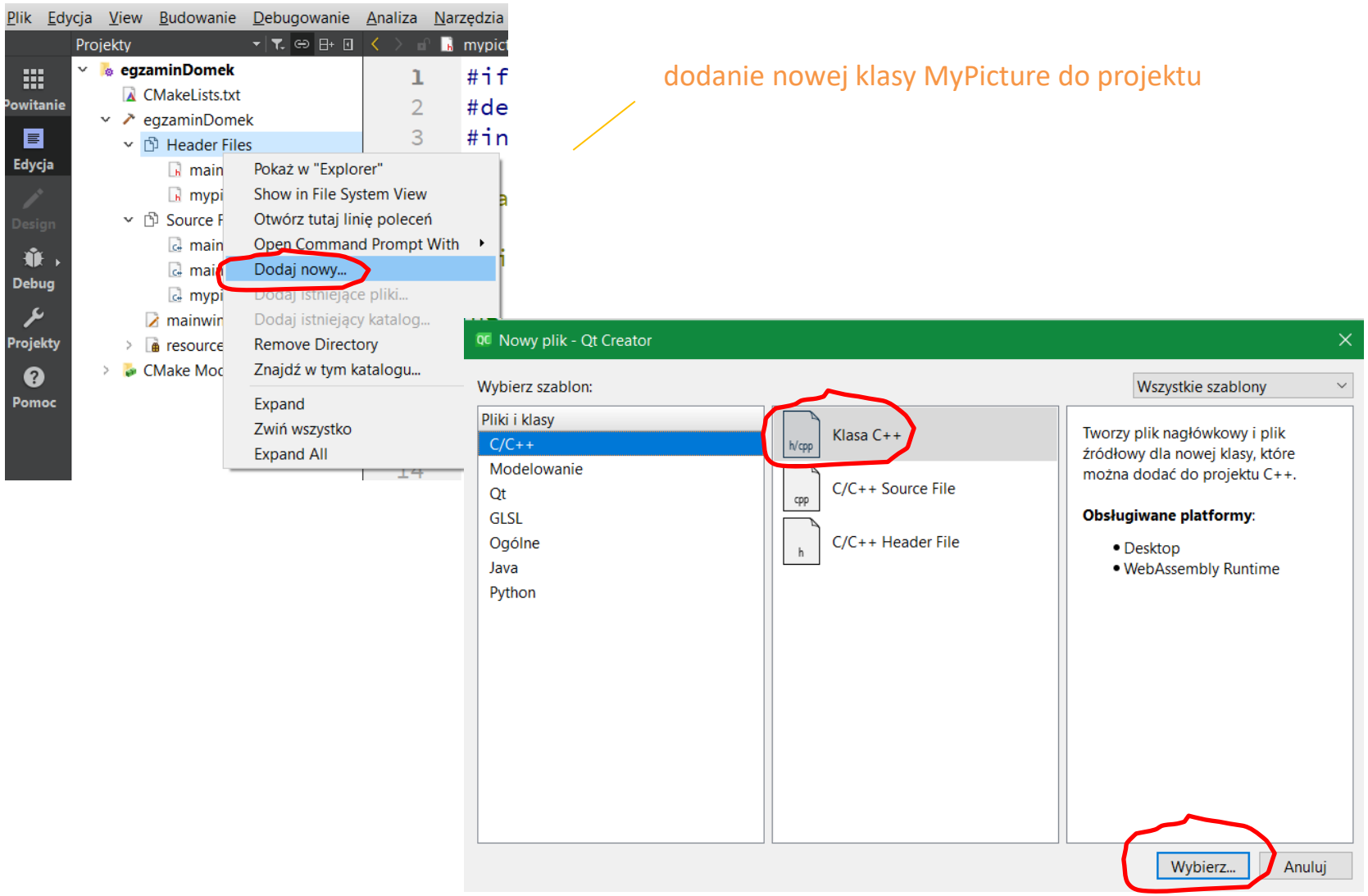


Obiekt	Klasa
MainWindow	QMainWindow
centralwidget	QWidget
label_opis	QLabel
listWidget_polubienia	QListWidget
verticalLayout_2	QVBoxLayout
horizontalLayout	QHBoxLayout
btn_lewo	QPushButton
btn_polub	QPushButton
btn_prawo	QPushButton
verticalLayout	QVBoxLayout
label_header	QLabel
label_picture	QLabel

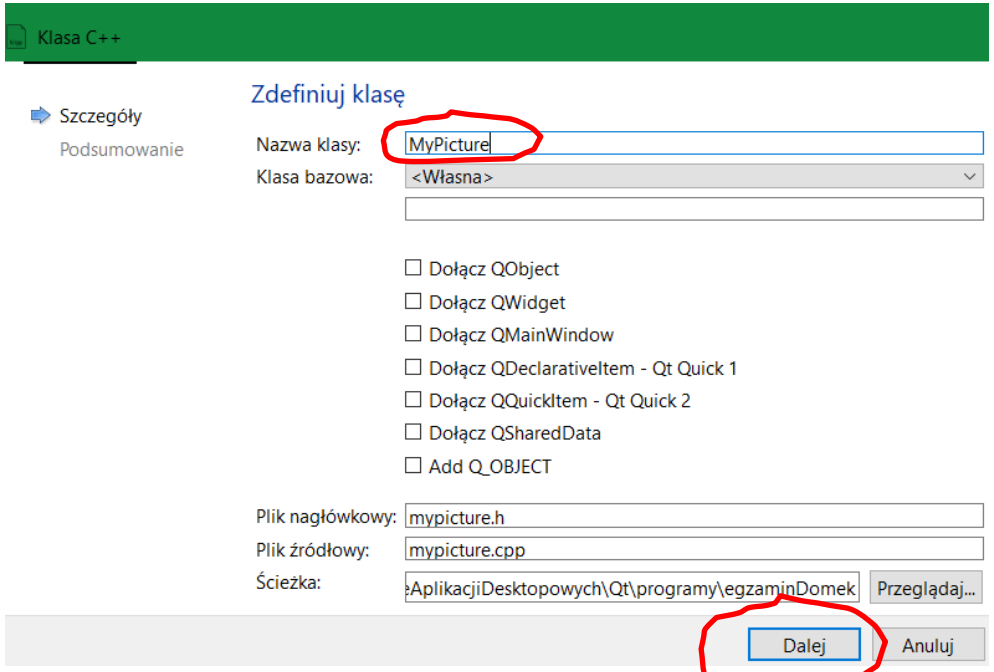
QListWidget

prawy/rozmieść w siatce

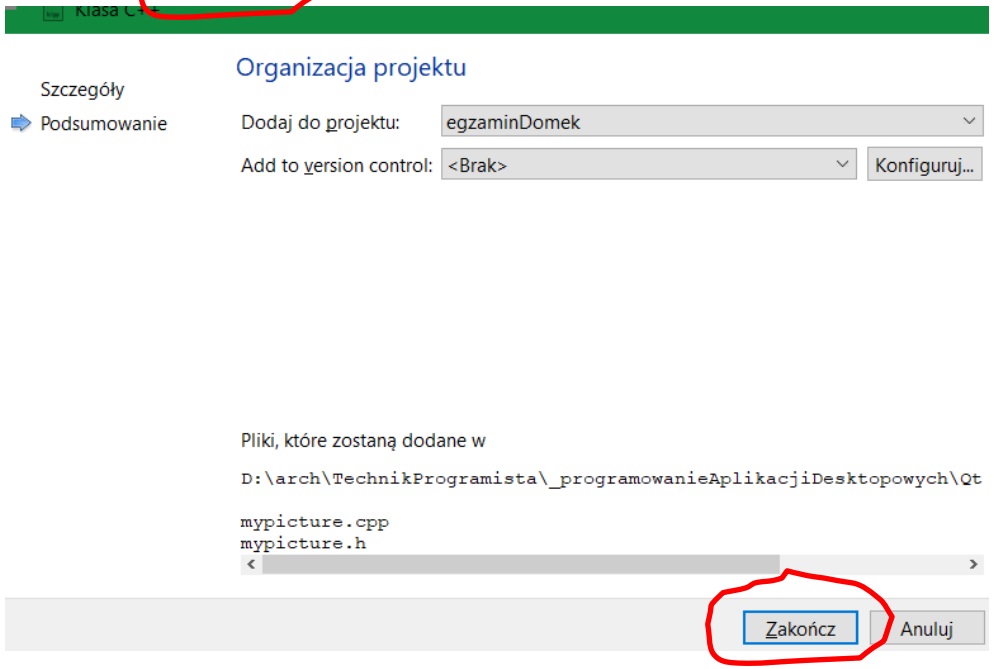
Galeria z polubieniami



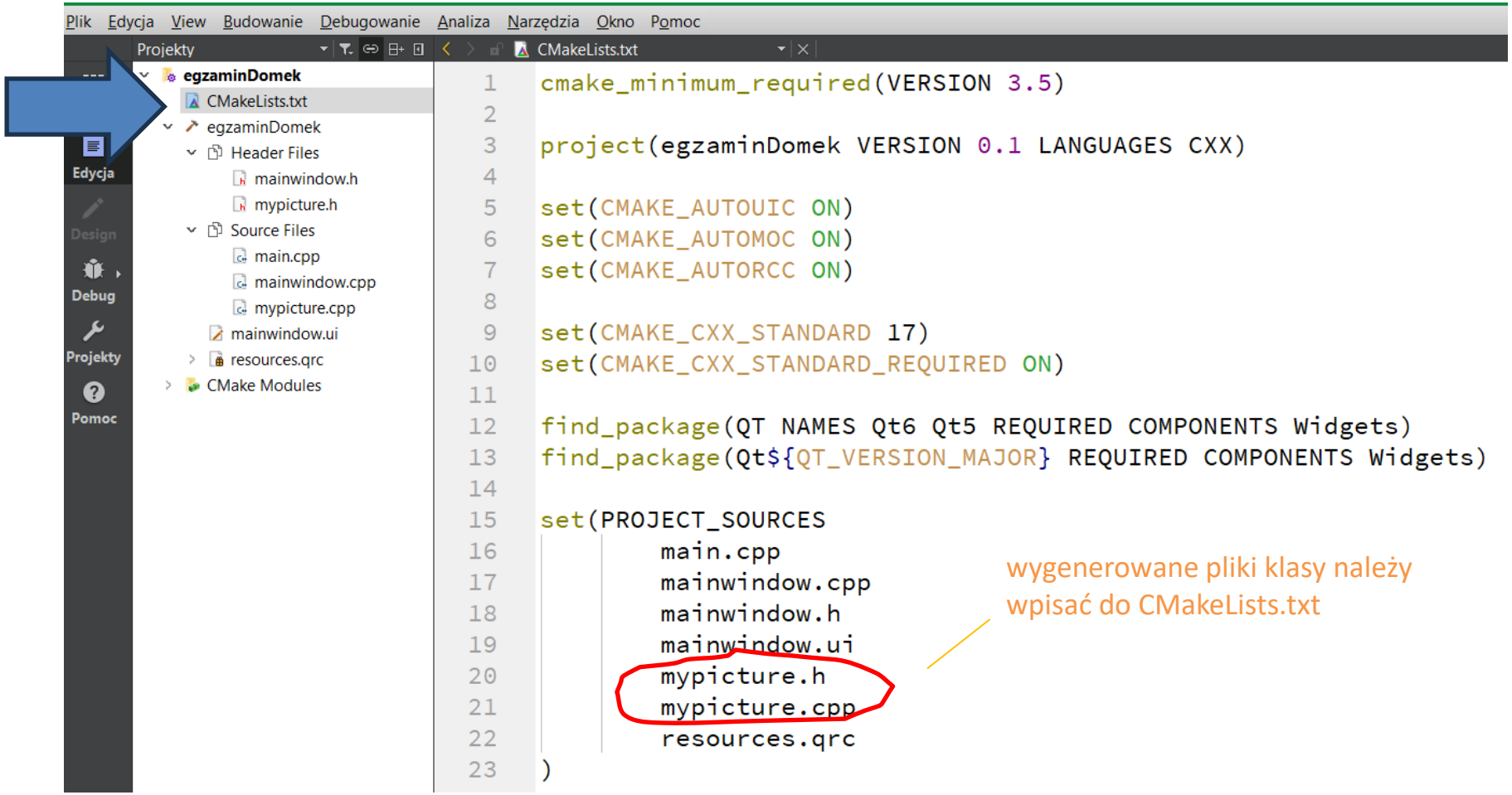
Galeria z polubieniami



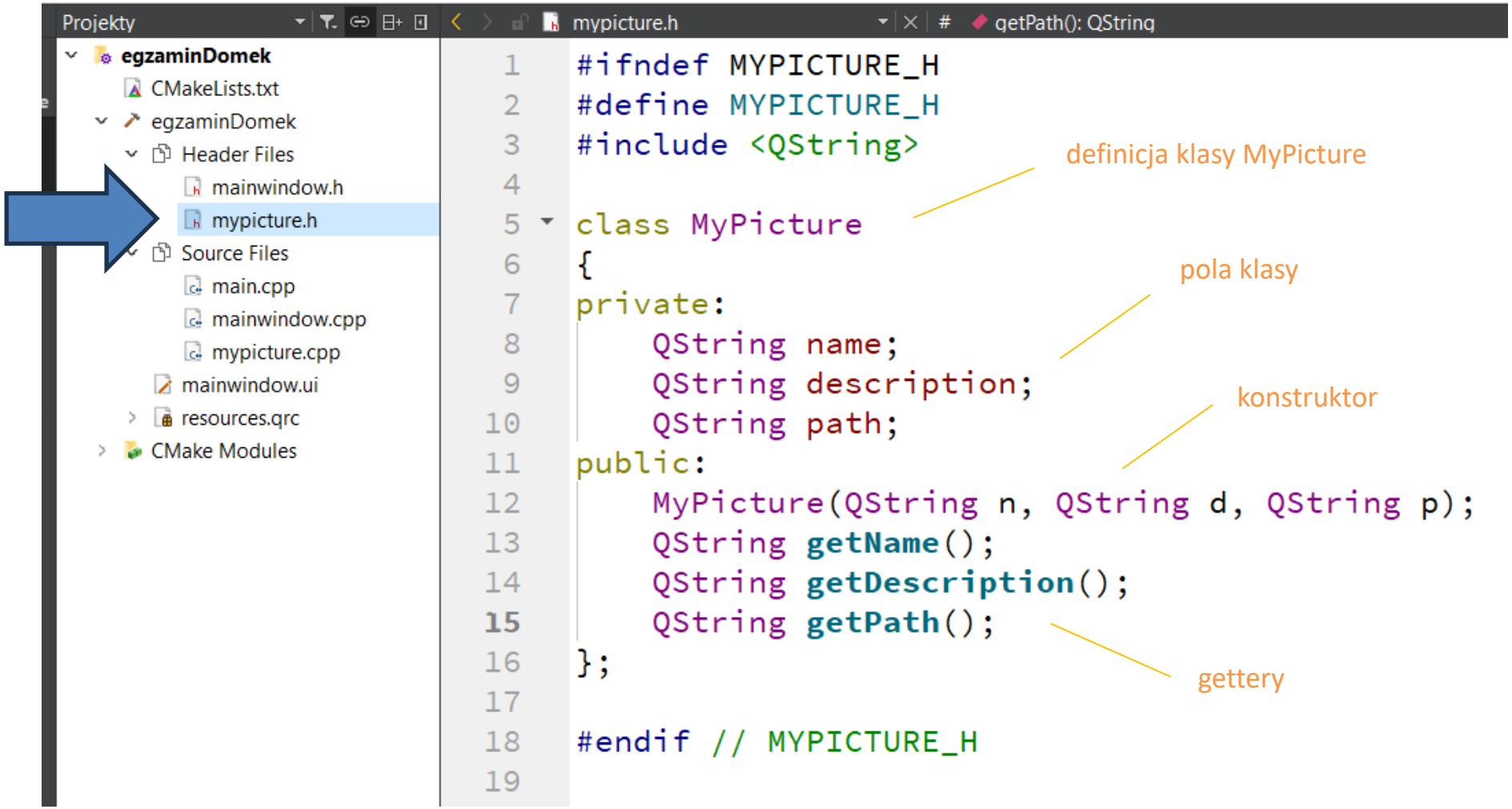
dodanie nowej klasy MyPicture



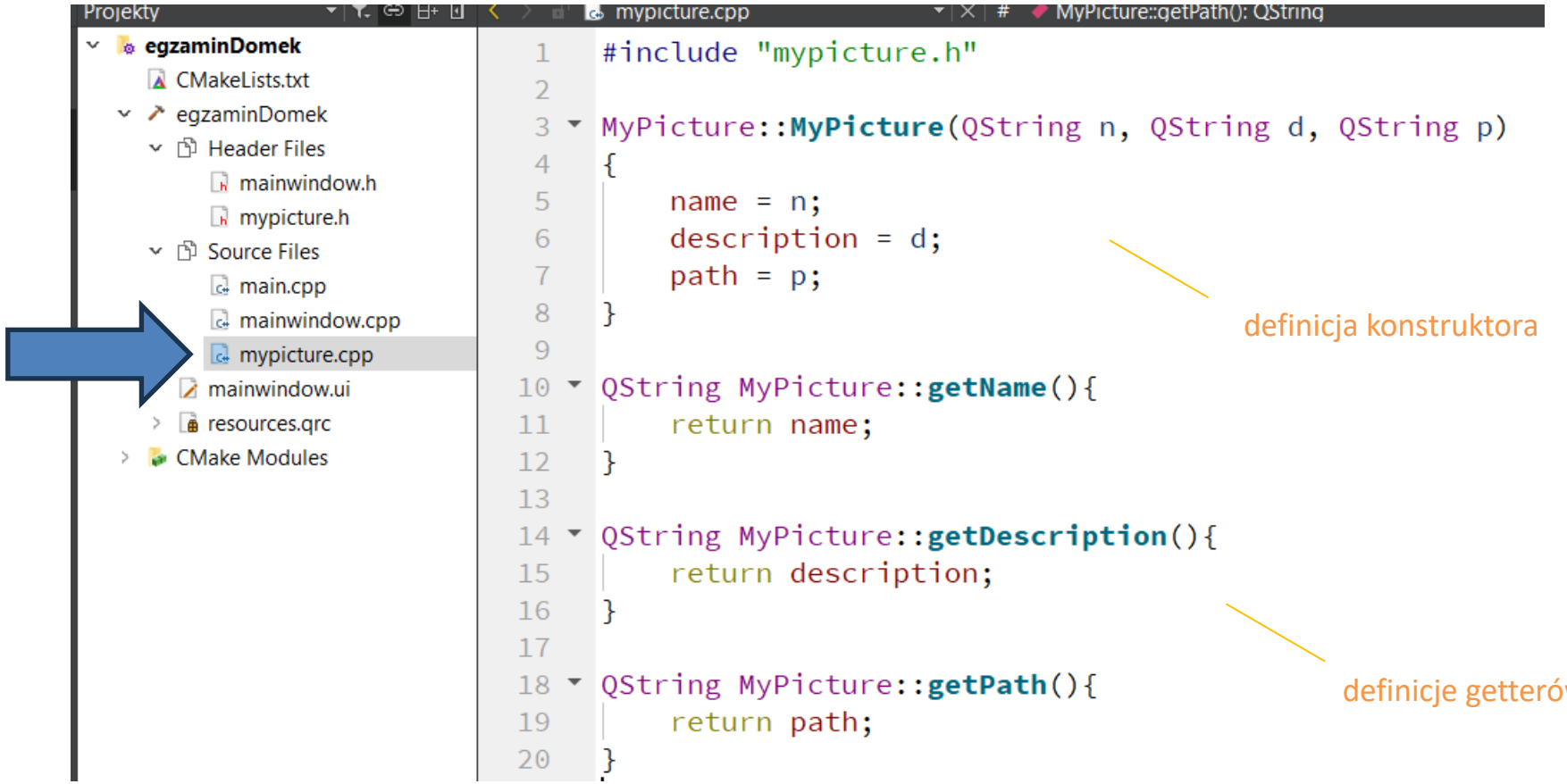
Galeria z polubieniami



Galeria z polubieniami



Galeria z polubieniami

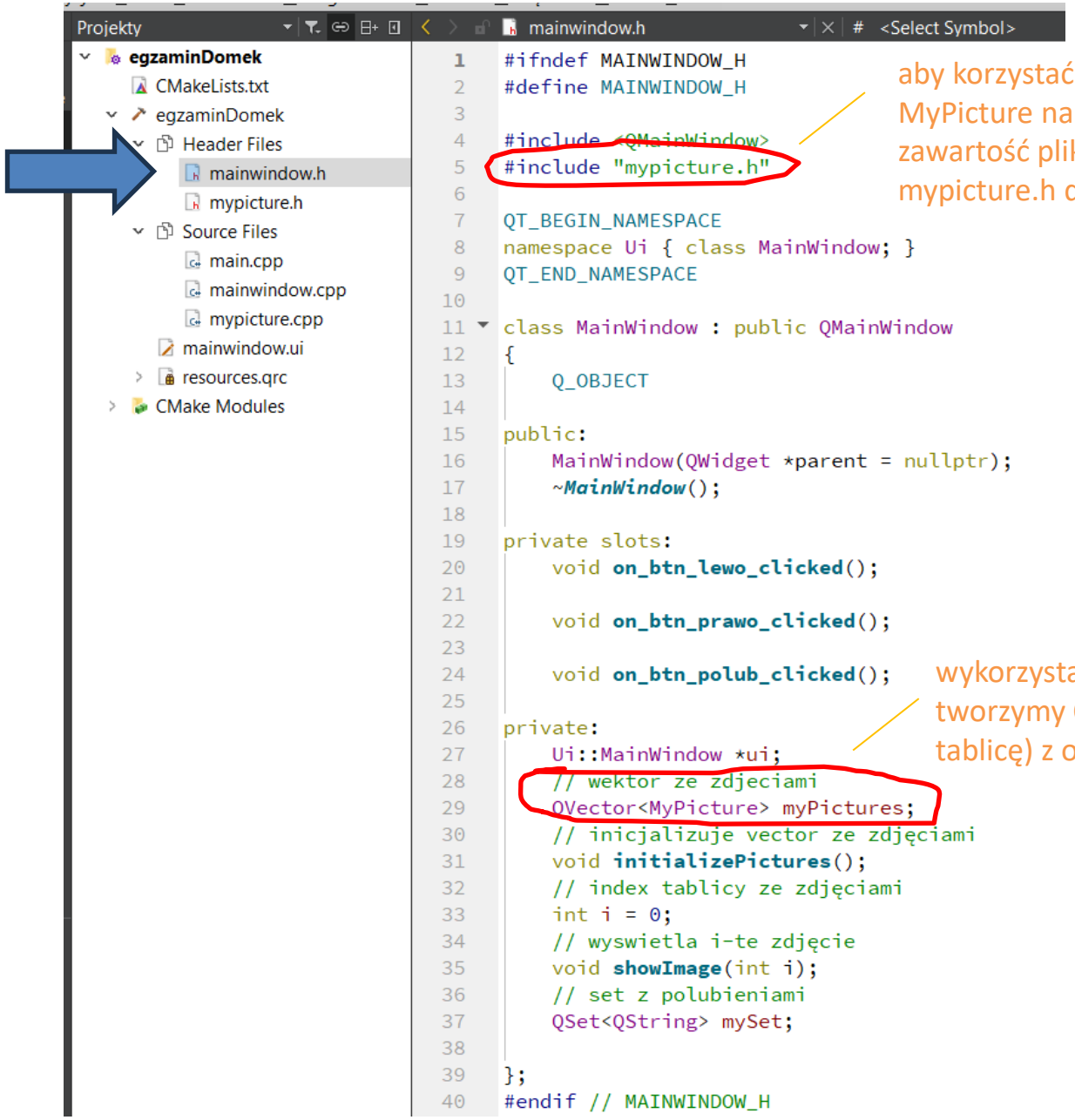


definicja konstruktora

definicje getterów

zawartość pliku mypicture.cpp

Galeria z polubieniami



aby korzystać z nowej klasy MyPicture należy dołączyć zawartość pliku nagłówkowego mypicture.h dyrektywą #include

wykorzystanie klasy MyPicture – tworzymy QVector (dynamiczną tablicę) z obiektami tej klasy

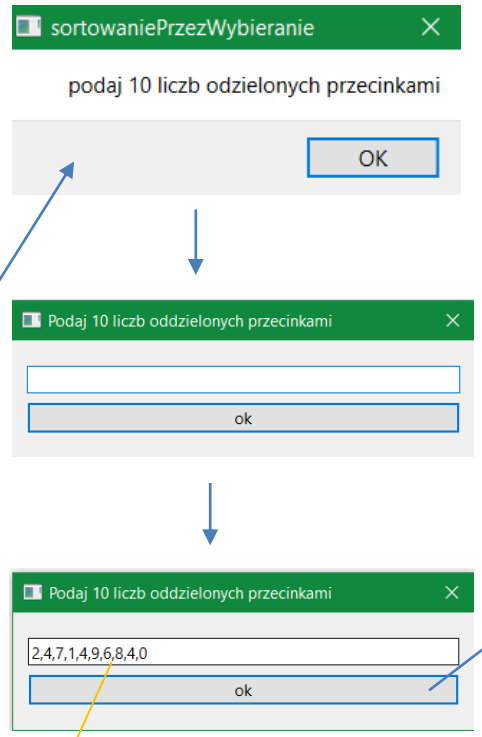
Galeria z polubieniami


```cpp
#include "mainwindow.h"
#include "./ui_mainwindow.h"
#include <QPixmap>
#include <QDebug>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    initializePictures();
    showImage(i);
    }

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::initializePictures(){
    myPictures.push_back(MyPicture("pierwszy kwiatek", "kwiatek 1", ":/pic/pictures/01.jpg"));
    myPictures.push_back(MyPicture("drugi kwiatek", "kwiatek 2", ":/pic/pictures/02.jpg"));
    myPictures.push_back(MyPicture("trzci kwiatek", "kwiatek 3", ":/pic/pictures/03.jpg"));
    myPictures.push_back(MyPicture("czwarty kwiatek", "kwiatek 4", ":/pic/pictures/04.jpg"));
    myPictures.push_back(MyPicture("piąty kwiatek", "kwiatek 5", ":/pic/pictures/05.jpg"));
}
```

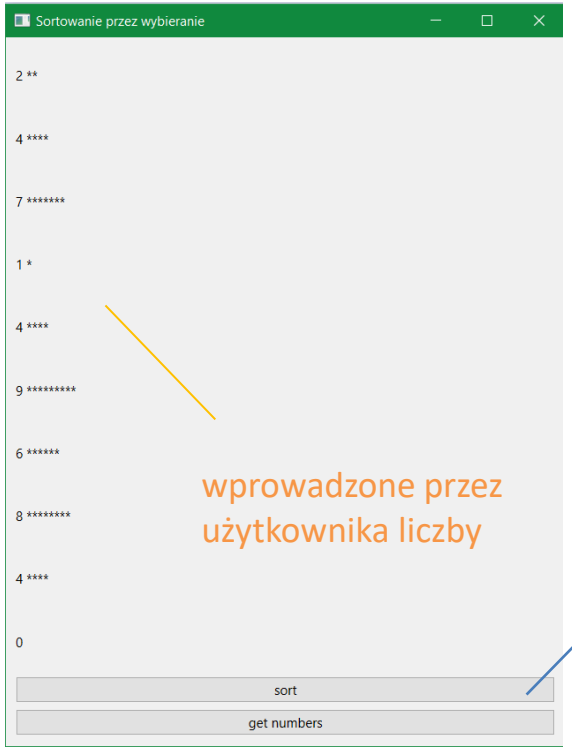
tworzenie obiektów klasy MyPicture

dodanie obiektów klasy MyPicture do dynamicznej tablicy myPictures

Sortowanie przez wybieranie



użytkownik musi wpisać właściwą ilość liczb oddzielonych przecinkami (inne znaki są ignorowane)



wprowadzone przez użytkownika liczby

liczby posortowane przez wybieranie