Wprowadzenie do biblioteki Qt w C++

SZPC++ #6

Tomasz Pietrzak

Dlaczego Qt?

- a dlaczego by nie? :)
- C++
- open source
- wieloplatformowy (Linux, Windows, OS X)
- wszechstronny

http://www.qt.io/

Wybrane programy oparte o Qt

- Adobe Photoshop Album
- Age of Wonders III
- Gadu-Gadu
- Kate
- Mathematica
- Psi
- Spotify
- VirtualBox
- Warzone 2100

https://en.wikipedia.org/wiki/Category:Software_that_uses_Qt

Instalacja Qt (Ubuntu)

```
$ sudo apt-get install qt5-default libqt5serialport5 libqt5serialport5-dev
Czytanie list pakietów... Gotowe
Budowanie drzewa zależności
Odczyt informacji o stanie... Gotowe
Zostaną zainstalowane następujące NOWE pakiety:
  libdrm-amdgpu1 libdrm-dev libegl1-mesa-dev libgl1-mesa-dev libgles2-mesa-dev
  libglu1-mesa-dev libmirclient-dev libmirclient7 libmirclientplatform-mesa
  libmirprotobuf-dev libmirprotobuf0 libprotobuf-dev libprotobuf-lite8
  libprotobuf8 libqt5concurrent5 libqt5core5a libqt5dbus5 libqt5gui5
  libqt5network5 libqt5opengl5 libqt5opengl5-dev libqt5printsupport5
  libqt5serialport5 libqt5serialport5-dev libqt5sql5 libqt5sql5-sqlite
  libqt5test5 libqt5widgets5 libqt5xml5 libwayland-dev libx11-xcb-dev
  libxcb-dri2-0-dev libxcb-dri3-dev libxcb-glx0-dev libxcb-icccm4
  libxcb-present-dev libxcb-randr0-dev libxcb-render-util0 libxcb-render0-dev
  libxcb-shape0-dev libxcb-sync-dev libxcb-xfixes0-dev libxcb-xkb1
  libxdamage-dev libxext-dev libxfixes-dev libxkbcommon-x11-0 libxshmfence-dev
  libxxf86vm-dev mesa-common-dev mircommon-dev qt5-default qt5-qmake
  qtbase5-dev qtbase5-dev-tools x11proto-damage-dev x11proto-dri2-dev
  x11proto-fixes-dev x11proto-gl-dev x11proto-xext-dev
  x11proto-xf86vidmode-dev
0 aktualizowanych, 61 nowo instalowanych, 0 usuwanych i 0 nieaktualizowanych.
Konieczne pobranie 13,8 MB archiwów.
Po tej operacji zostanie dodatkowo użyte 64,8 MB miejsca na dysku.
Kontynuować? [T/n]
```

Instalacja Qt Creator (Ubuntu)

```
$ sudo apt-get install qtcreator-dev
Czytanie list pakietów... Gotowe
Budowanie drzewa zależności
Odczyt informacji o stanie... Gotowe
Zostaną zainstalowane następujące NOWE pakiety:
  icu-devtools libbotan-1.10-0 libicu-dev libqt5clucene5 libqt5declarative5
  libqt5designer5 libqt5designercomponents5 libqt5help5 libqt5positioning5
  libqt5qml5 libqt5quick5 libqt5quickparticles5 libqt5quicktest5 libqt5script5
  libqt5sensors5 libqt5svg5 libqt5webkit5 qmlscene qtcreator qtcreator-dev
  qtcreator-doc qtcreator-plugin-cmake qtcreator-plugin-qnx
  qtcreator-plugin-remotelinux qtcreator-plugin-valgrind
  qtdeclarative5-controls-plugin qtdeclarative5-dev
  qtdeclarative5-quicklayouts-plugin
0 aktualizowanych, 28 nowo instalowanych, 0 usuwanych i 0 nieaktualizowanych.
Konieczne pobranie 48,1 MB archiwów.
Po tej operacji zostanie dodatkowo użyte 179 MB miejsca na dysku.
Kontynuować? [T/n]
```

Instalacja qwt (Ubuntu)

```
$ sudo apt-get install libqwt-qt5-dev libqwt6-qt5
```

Pakiety nie występują w repozytoriach starszych wersji Ubuntu (np. 14.04 LTS) i trzeba je ściągnąć oraz zainstalować ręcznie.

http://packages.ubuntu.com/vivid/libqwt-qt5-dev

http://packages.ubuntu.com/vivid/libqwt6-qt5

Budowanie programu z Qt (Makefile)

```
CFLAGS = -I/usr/include/qt5/
    -I/usr/include/qt5/QtWidgets
    -I/usr/include/qt5/QtCore
    -I/usr/include/qt5/QtGui
    -fPIC -std=c++11 -O2

LIBS = -lQt5Core -lQt5Gui -lQt5Widgets -lqwt-qt5
```

UWAGA! Ścieżki nagłówków są mocno niestabilne w czasie. Na chwilę obecną (dot. x86) zostały przeniesione do **/usr/include/i386-linux-gnu/qt5**

Lokalizację nagłówków, bibliotek dynamicznych itp. można sprawdzić poleceniem **qtdiag**.

Zalążek programu

```
#include <QtWidgets/QApplication>
#include <QtWidgets/Qwidget>

int main (int argc, char *arg[])
{
	QApplication app (argc, arg);
	// Tworzenie okna programu
	QWidget okno;
	okno.setWindowTitle ("Dopasowanie MNK");		// ustawia tytuł okna
	okno.setFixedSize (800, 800);			// rozmiar okna 800x800
	// (…) rysowanie na oknie
	okno.show();					// pokazuje okno

	return app.exec();
}
```

QwtPlot (wykres)

```
#include <qwt/qwt_plot.h>

// Tworzenie wykresu na konkretnym oknie
QwtPlot wykres (&okno);
// Ustawienie tytułu wykresu
wykres.setTitle ("Dopasowanie MNK");
// Ustawienia tytułów osi:  yLeft, yRight, xBottom, xTop
wykres.setAxisTitle (QwtPlot::xBottom, "Nr kanału");
wykres.setAxisTitle (QwtPlot::yLeft, "Napięcie / V");
// Ustawienie rozmiarów wykresu
wykres.setFixedSize (775, 775);
// Ustawienie koloru tła
wykres.setCanvasBackground(QBrush (QColor (0xff,0xfa, 0x6b)));
// Ustawienie zakresu osi
wykres.setAxisScale (QwtPlot::xBottom, min_x, max_x);
wykres.setAxisScale (QwtPlot::yLeft, min_y, max_y);
```

QwtPlotCurve (seria danych)

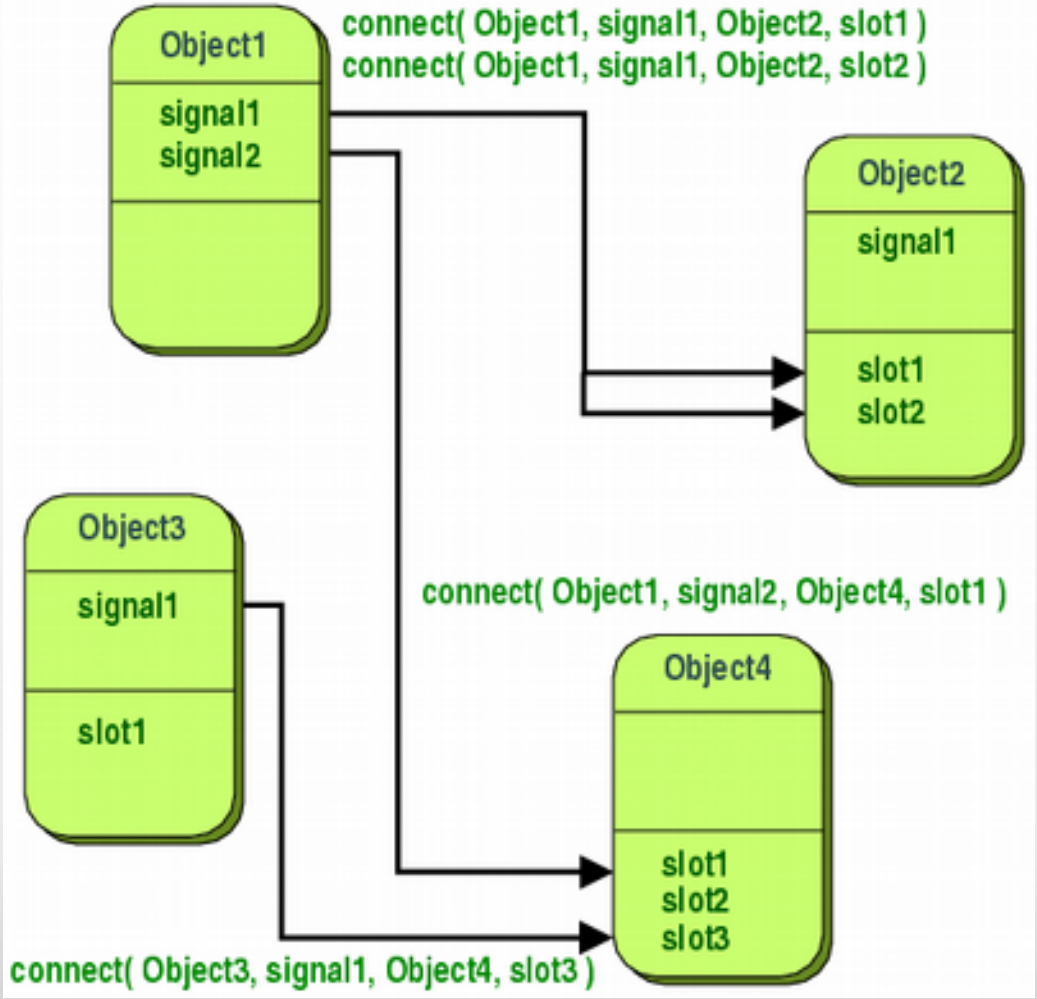
```
#include <qwt/qwt_plot_curve.h>
#include <qwt/qwt_point_data.h>

std::vector <double> x, y;
// (…) uzupełnienie wektorów z danymi

QwtPlotCurve dane_doswiadczalne;
// Wprowadzenie punktów pomiarowych (dwie możliwości)
dane_doswiadczalne.setRawSamples (x.data(), y.data(), x.size());  // oryginalne tablice
dane_doswiadczalne.setSamples (x.data(), y.data(), x.size());     // twarde kopie danych
// Ustawienie stylu QwtPlotCurve: Lines, Sticks, Steps, Dots
dane_doswiadczalne.setStyle (QwtPlotCurve::Dots);
// Ustawienie „pióra”: kolor, grubość:
dane_doswiadczalne.setPen (QPen (Qt::blue, 3));
// Dołączenie serii do istniejącego wykresu
dane_doswiadczalne.attach (&wykres);
```

QColor (kolory)


```
QColor (int r, int g, int b, int alfa = 255);
QColor (QRgb color);
QColor (const QString & name);
QColor (const char * name);
QColor (Qt::GlobalColor color);
```

```
Qt::white Qt::black Qt::red Qt::darkRed Qt::green Qt::darkGreen Qt::blue Qt::darkBlue
Qt::cyan Qt::darkCyan Qt::magenta Qt::darkMagenta Qt::yellow  Qt::darkYellow Qt::gray
Qt::darkGray Qt::lightGray Qt::transparent
```

Sygnały i sockety



Sygnały i sockety

- Każdy obiekt może posiadać wiele sygnałów i socketów
- Niektóre sygnały i sockety mogą nie był nigdzie podłączone
- Jeden sygnał może być wysyłany do wielu innych socketów
- Do jednego socketu może być „podłączonych” wiele sygnałów
- Obiekty muszą dziedziczyć (choćby pośrednio) po QObject
- Argumenty sygnału i socketu muszą być dopasowane
- Można przeciążać sockety (różne listy argumentów)
- Używanie własnych socketów wymaga parsowania przez MOC (Meta-Object Compiler), co mobilizuje do używania QtCreatora

QObject::connect()

```
QNadawca *nadawca;
QOdbiorca *odbiorca;

// QNadawca posiada sygnał          void moj_sygnal (int);
// QOdbiorca posiada slot           void moj_slot (int);

// Sposób użycia #1
QObject::connect(nadawca, SIGNAL(moj_sygnal (int)), odbiorca, SLOT(moj_slot (int)));

// Sposób użycia #2
Qobject::connect(nadawca, &QNadawca::moj_sygnal, odbiorca, &QOdbiorca::moj_slot);
```

http://doc.qt.io/qt-5/signalsandslots.html

QObject::connect()

```
QNadawca *nadawca;

void arcyfantastyczna_funkcja (int kaczor)
{
     // (…) robi super rzeczy!
}
// QNadawca posiada sygnał          void moj_sygnal (int);

// Sposób użycia #3
Qobject::connect(nadawca, &QNadawca::moj_sygnal, arcyfantastyczna_funkcja);
```

http://doc.qt.io/qt-5/signalsandslots.html

Mój slot

```
#include <QtWidgets/QLCDNumber>

class QLCDNumberMPH : public QLCDNumber
{
      Q_OBJECT  // bardzo ważne „magiczne” makro

public:
      QLCDNumberMPH(QWidget* qw) : QLCDNumber (qw) {}
      virtual ~QLCDNumberMPH() {}

// Moje sloty
public slots:
      void displayMPH (int w) { QLCDNumber::display (w / 1.6093);}
};
```

QtCreator

Dziękuję za uwagę!