Faculty of Electronics and Information Technology
WARSAW UNIVERSITY OF TECHNOLOGY

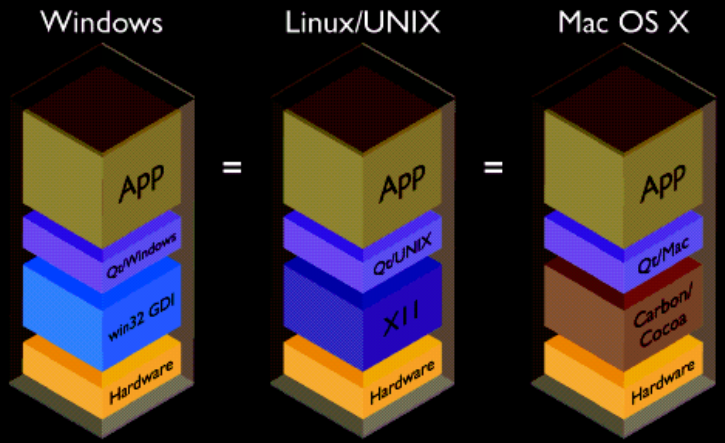
Programowanie Aplikacyjne (PAP)

Biblioteka Qt

Julian Myrcha
Institute of Computer Science
26.02.2025

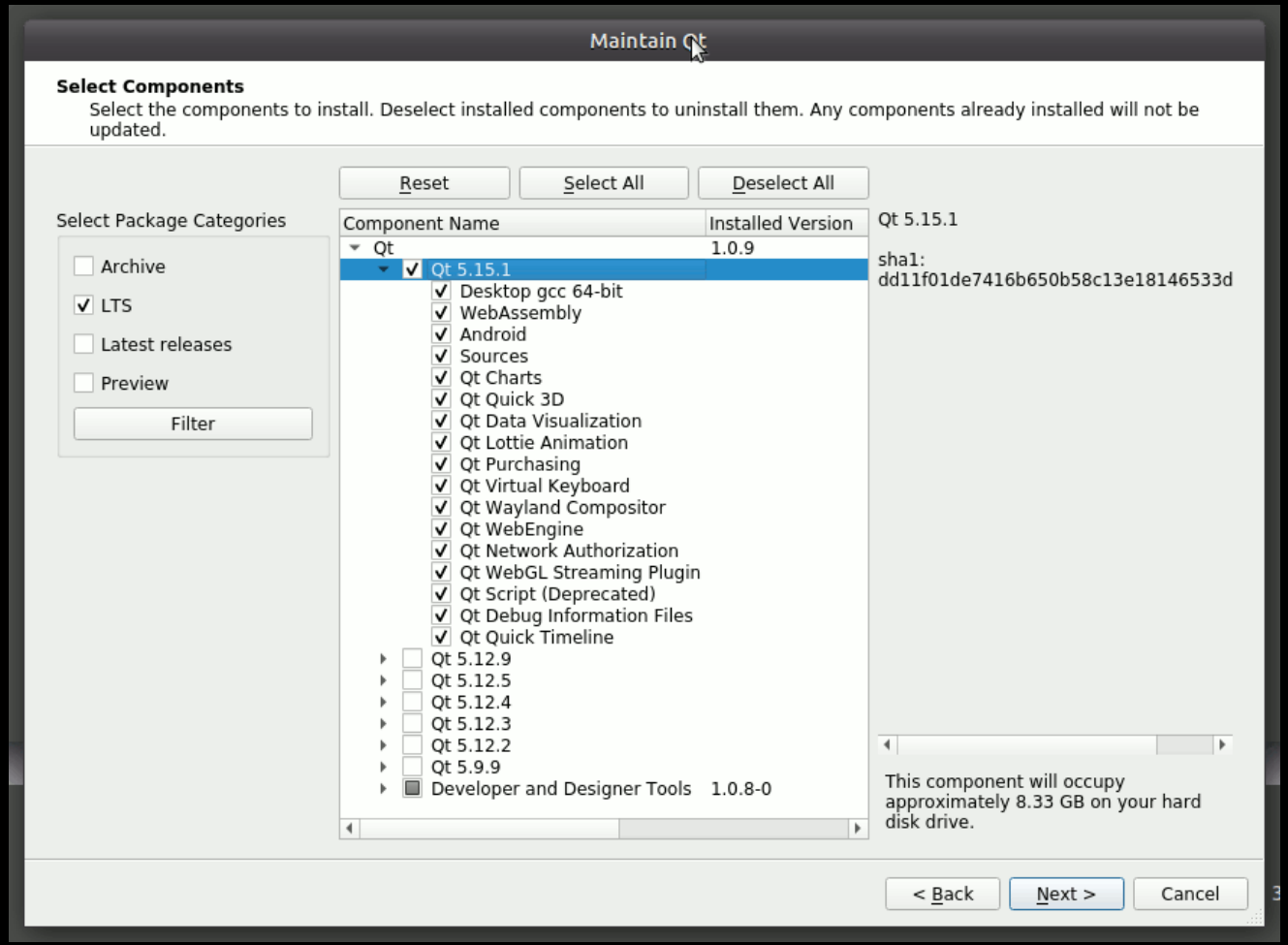
- Qt to wieloplatformowy framework aplikacji
- Napisany w C++
- Natywna aplikacja na każdej platformie
- Ukrywa biblioteki zależne od platformy przy użyciu wspólnego interfejsu API
- minimalne zmiany w kodzie
- kontrolki rysowane przez qt - więc jest to emulacja
- kompilator metaobject, który udostępnia funkcje programistyczne niedostępne natywnie w C ++ takie jak `signals` oraz `slots`
- Wersje OpenSource dostępne na licencji GPLv2, GPLv3, LGPLv3
- Licencjonowanie open source Qt jest idealne do przypadków użycia, takich jak projekty open source z dystrybucją open source, cele studenckie / akademickie, projekty hobbystyczne, wewnętrzne projekty badawcze bez dystrybucji zewnętrznej

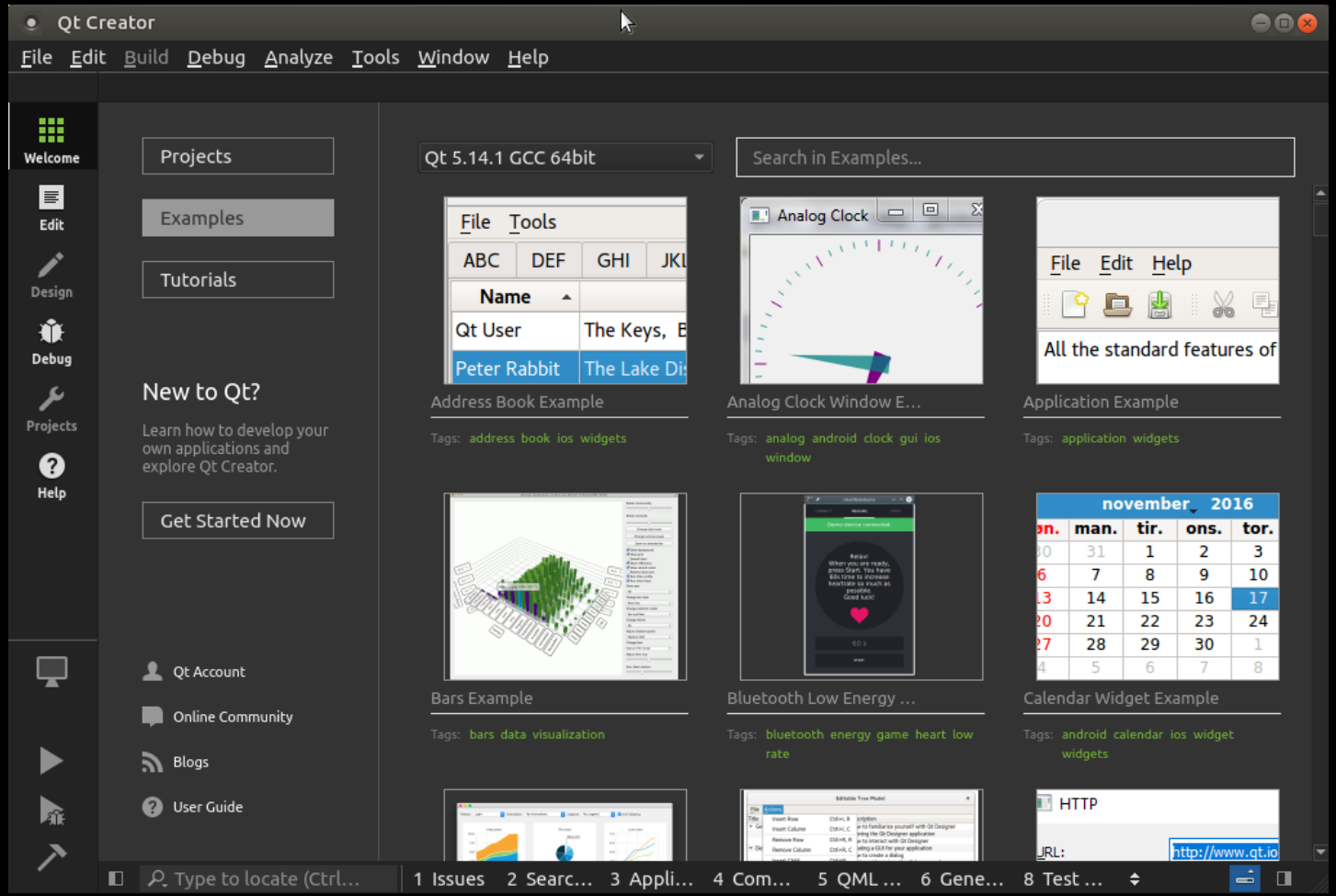
- Jednolity interfejs Qt API dla wielu platform
- Standardowe kompilatory (gcc, msvc)
- Platformy:
  - Windows XP, Vista, 7, 8, 10, WinRT
  - Unix (BSD, AIX, IRIX, HP-UX)
  - Mac OS X
  - Linux
  - Android
  - IOS

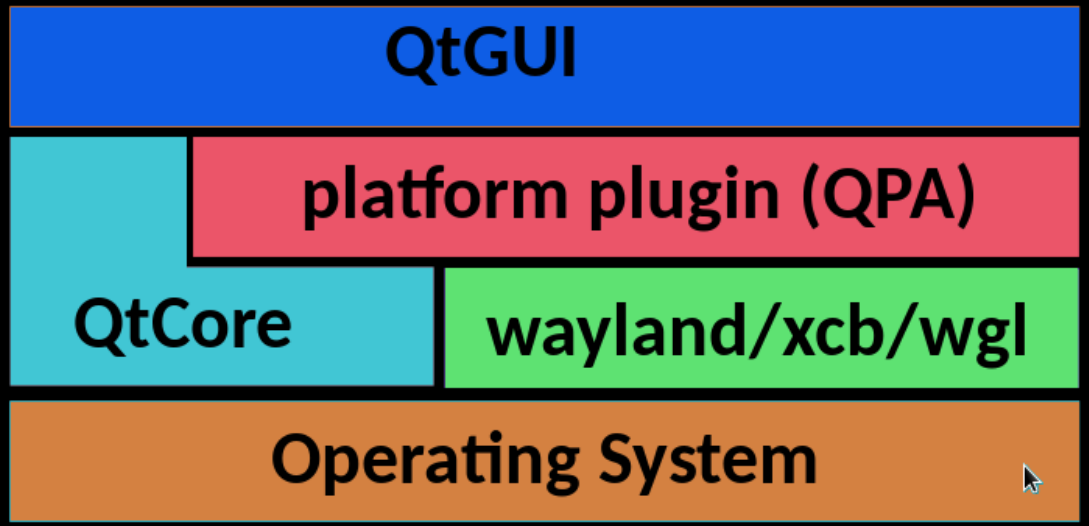


- Nie wolno łączyć oprogramowania opracowanego na licencji komercyjnej z kodem opracowanym na licencji Open Source
- Zapewnienie możliwości łączenia bibliotek Qt
- Dostarcz kopię licencji i wyraźnie potwierdź użycie Qt
- Udostępnij klientom kopię kodu źródłowego Qt
- Zaakceptuj, że modyfikacje kodu źródłowego Qt są niezastrzeżone
- Twórz „otwarte” urządzenia konsumenckie
- Zaakceptuj warunki zarządzania prawami cyfrowymi
- Zwróć szczególną uwagę podczas prób egzekwowania patentów na oprogramowanie

- 1994: Założenie Trolltech w Oslo w Norwegii
- 1996: perwsza sprzedaż Qt (Europejska Agencja Kosmiczna)
- 2000: Wydanie Qt/X11 na licencji GPL
- 2001: Qt 3
- 2005: rozszerza podwójne licencjonowanie na Windows
- 2005: Qt 4 wydane
- 2008: Qt 4.4, Nokia
- 2009: Qt 4.5 (LGPL)
- 2012: Qt 5, (QML, JavaScript, Hardware-accelerated graphics)
- 2016: Qt 5.6 LTS (usunięte Qt WebKit, Qt Quick 1.0)
- 2017: Qt 5.9 LTS (Qt WebEngine używa Chromium, ...)
- 2019: Qt 5.12 LTS (Pierwsze WebAssembly, Qt dla Pythona)
- 2019: Qt 5.14 (https://www.qt.io/offline-installers)


Maintain Qt
Select Components
Select the components to install. Deselect installed components to uninstall them. Any components already installed will not be updated.
Reset
Select All
Deselect All
Select Package Categories
Archive
LTS
Latest releases
Preview
Filter
Component Name
Installed Version
Qt
1.0.9
Qt 5.15.1
Desktop gcc 64-bit
WebAssembly
Android
Sources
Qt Charts
Qt Quick 3D
Qt Data Visualization
Qt Lottie Animation
Qt Purchasing
Qt Virtual Keyboard
Qt Wayland Compositor
Qt WebEngine
Qt Network Authorization
Qt WebGL Streaming Plugin
Qt Script (Deprecated)
Qt Debug Information Files
Qt Quick Timeline
Qt 5.12.9
Qt 5.12.5
Qt 5.12.4
Qt 5.12.3
Qt 5.12.2
Qt 5.9.9
Developer and Designer Tools
1.0.8-0
Qt 5.15.1
sha1:
dd11f01de7416b650b58c13e18146533d
This component will occupy approximately 8.33 GB on your hard disk drive.
< Back
Next >
Cancel


Qt Creator
File Edit Build Debug Analyze Tools Window Help
Welcome
Edit
Design
Debug
Projects
Help
Projects
Examples
Tutorials
New to Qt?
Learn how to develop your own applications and explore Qt Creator.
Get Started Now
Qt Account
Online Community
Blogs
User Guide
Qt 5.14.1 GCC 64bit
Search in Examples...
File Tools
ABC DEF GHI
Name
Qt User
Peter Rabbit
Address Book Example
Tags: address book ios widgets
Analog Clock
Analog Clock Window E...
Tags: analog android clock gui ios window
File Edit Help
All the standard features of
Application Example
Tags: application widgets
Bars Example
Tags: bars data visualization
Bluetooth Low Energy ...
Tags: bluetooth energy game heart low rate
november 2016
man. tir. ons. tor.
Calendar Widget Example
Tags: android calendar ios widget widgets
HTTP
http://www.qt.io
Type to locate (Ctrl...
1 Issues 2 Searc... 3 Appli... 4 Com... 5 QML ... 6 Gene... 8 Test ...



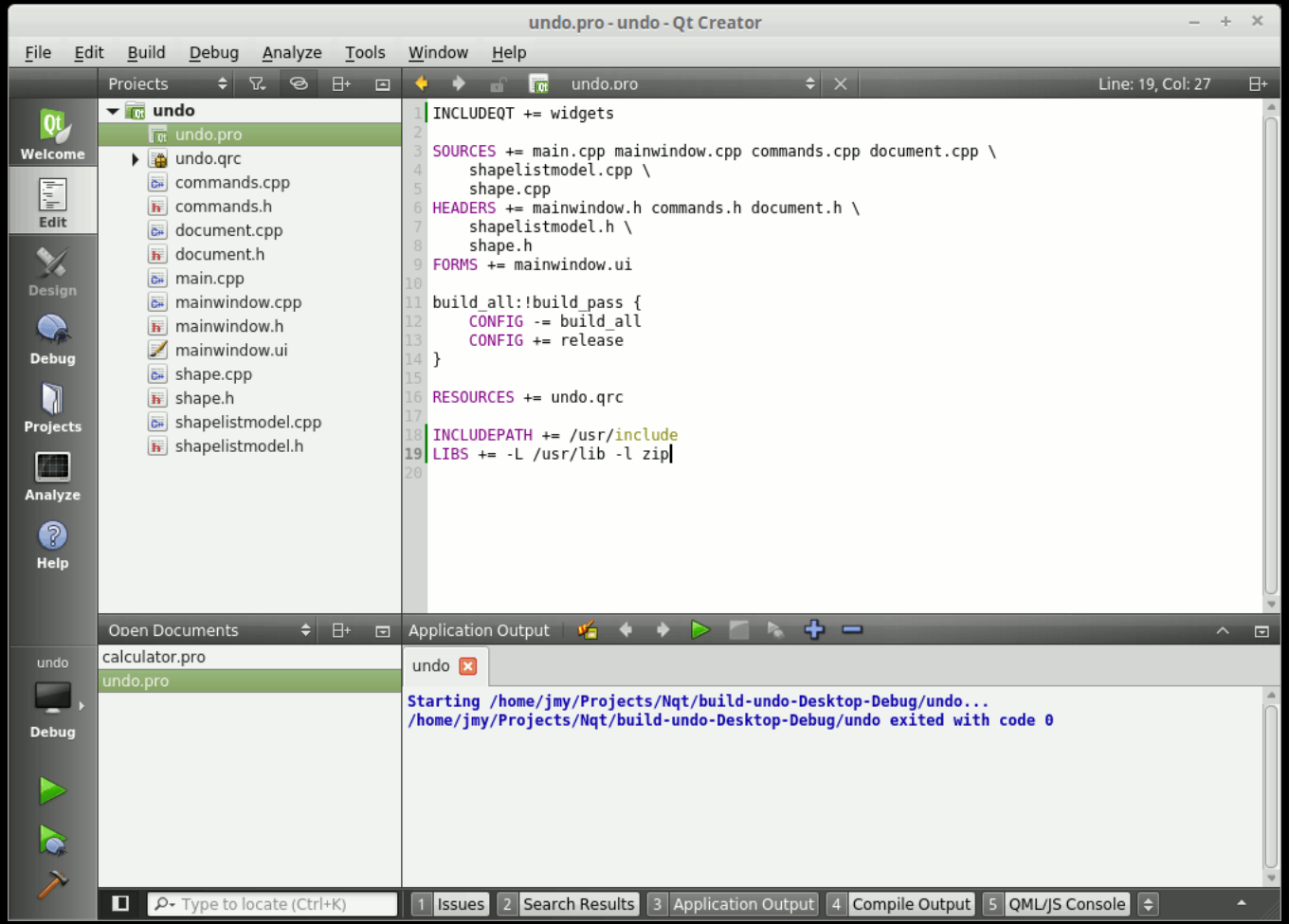

```
./program -platform <name>
```

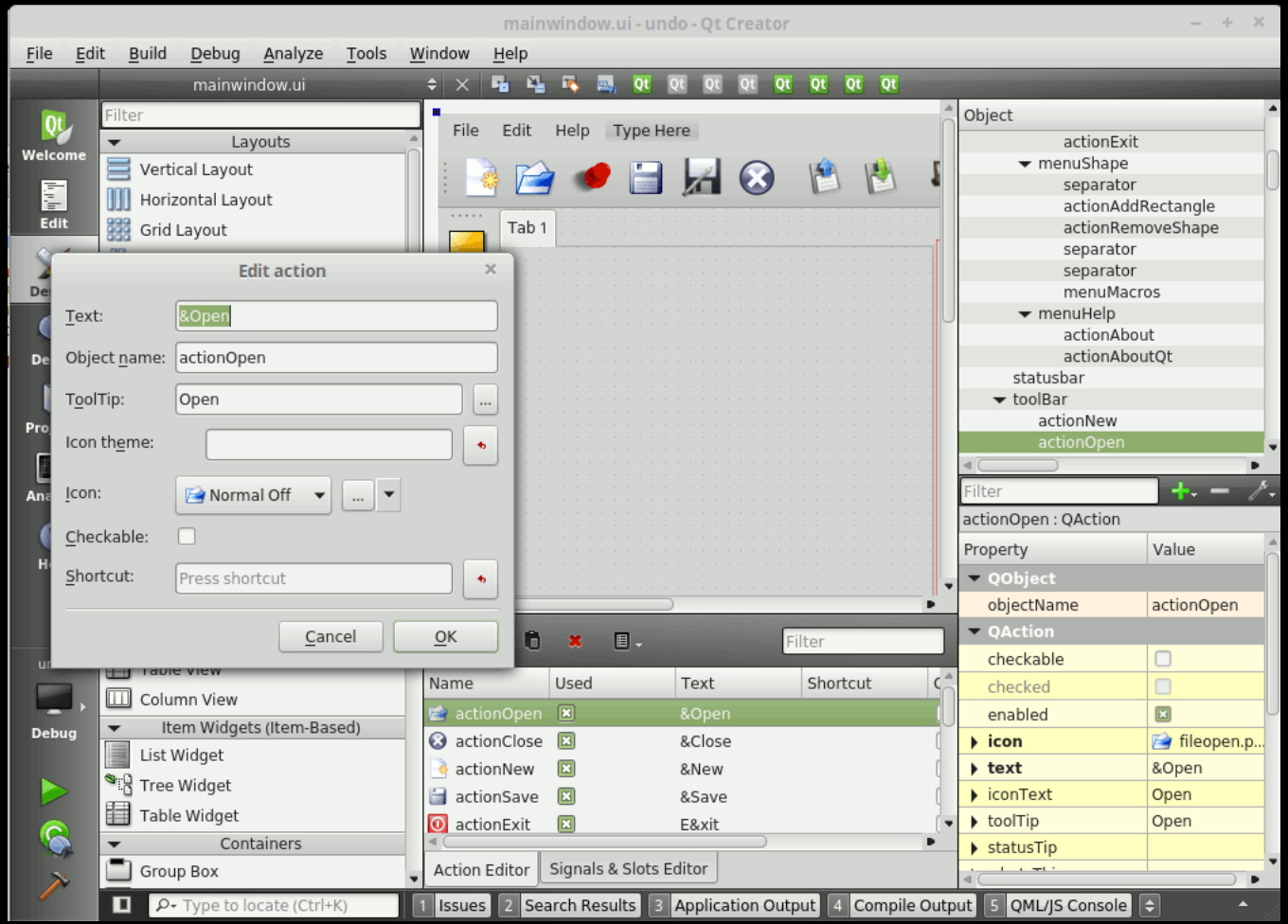
gdzie name jest jednym z: eglfs, linuxfb, minimal, minimalegl, offscreen, vnc, xcb

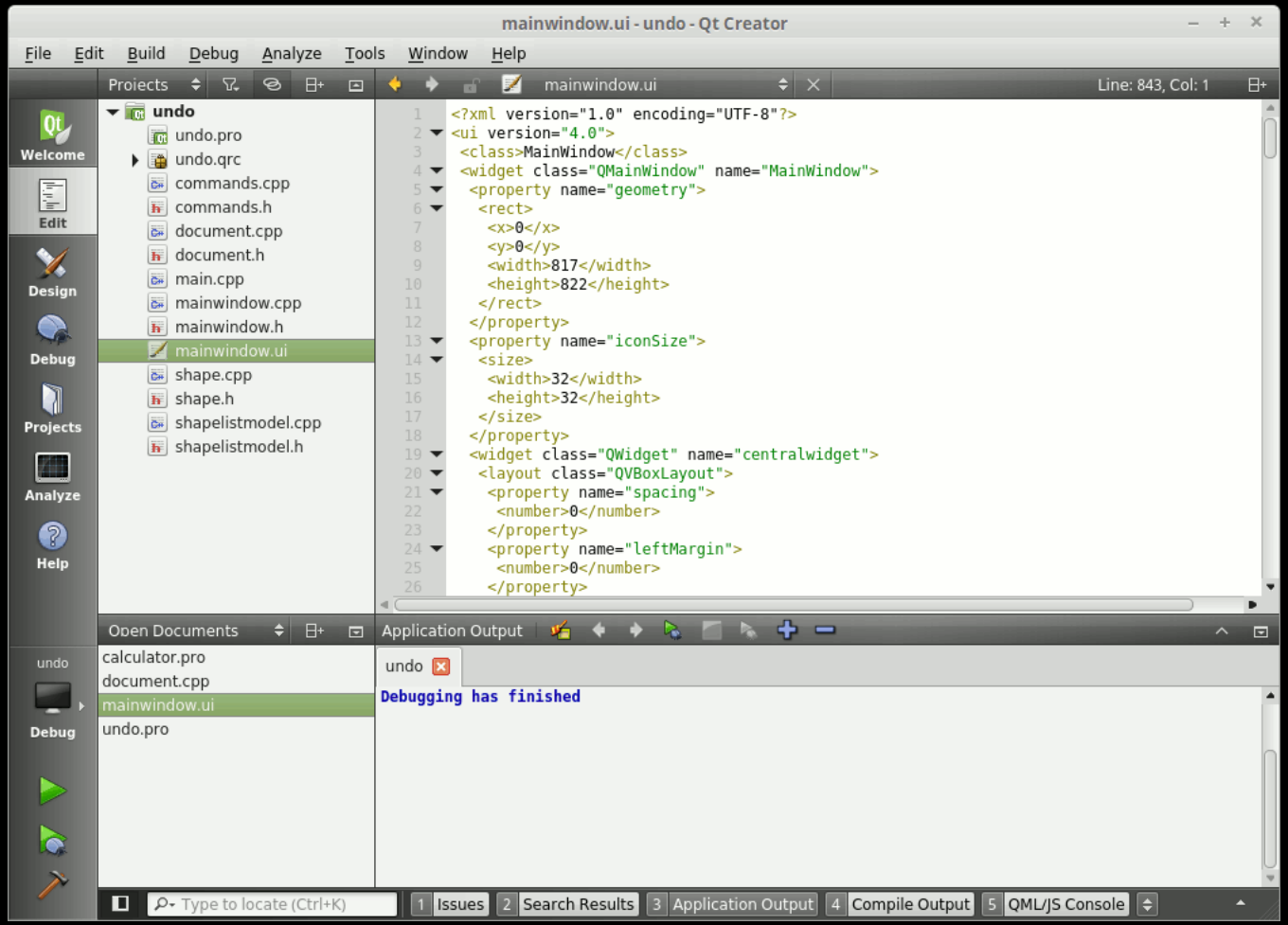
xcb - wykorzystuje X11

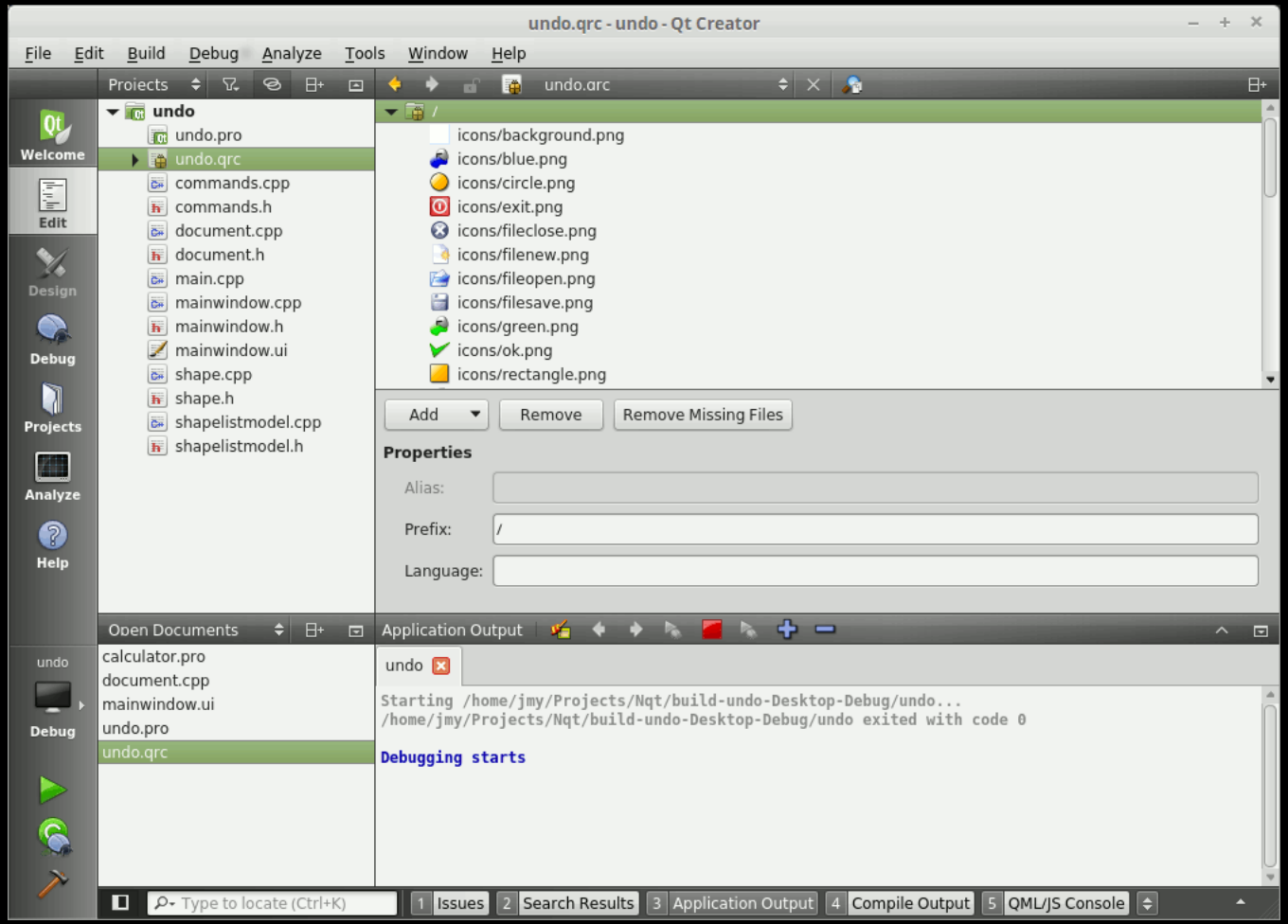
wayland - Wykorzystuje protokół wayland

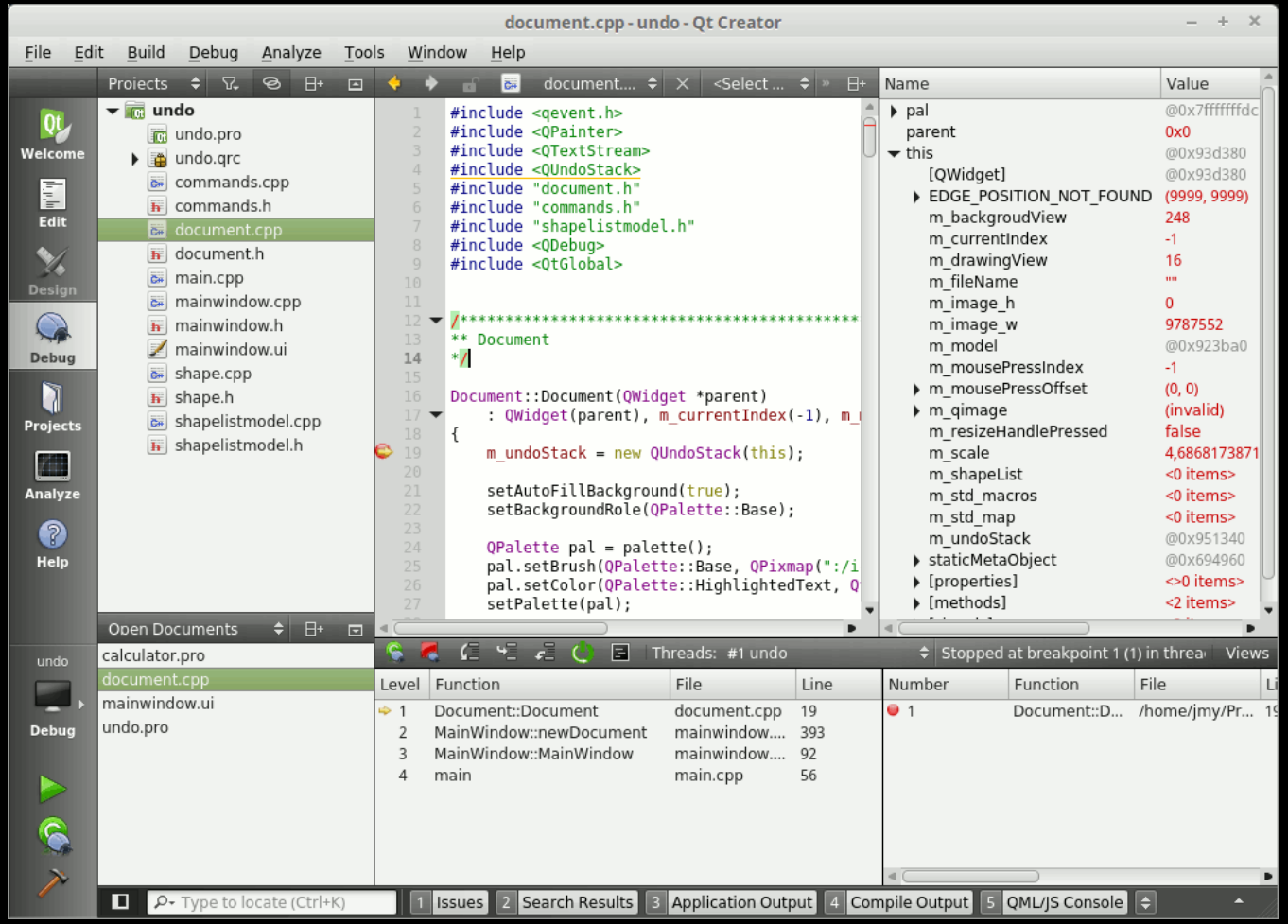
vnc - tworzy VNC dla okna aplikacji

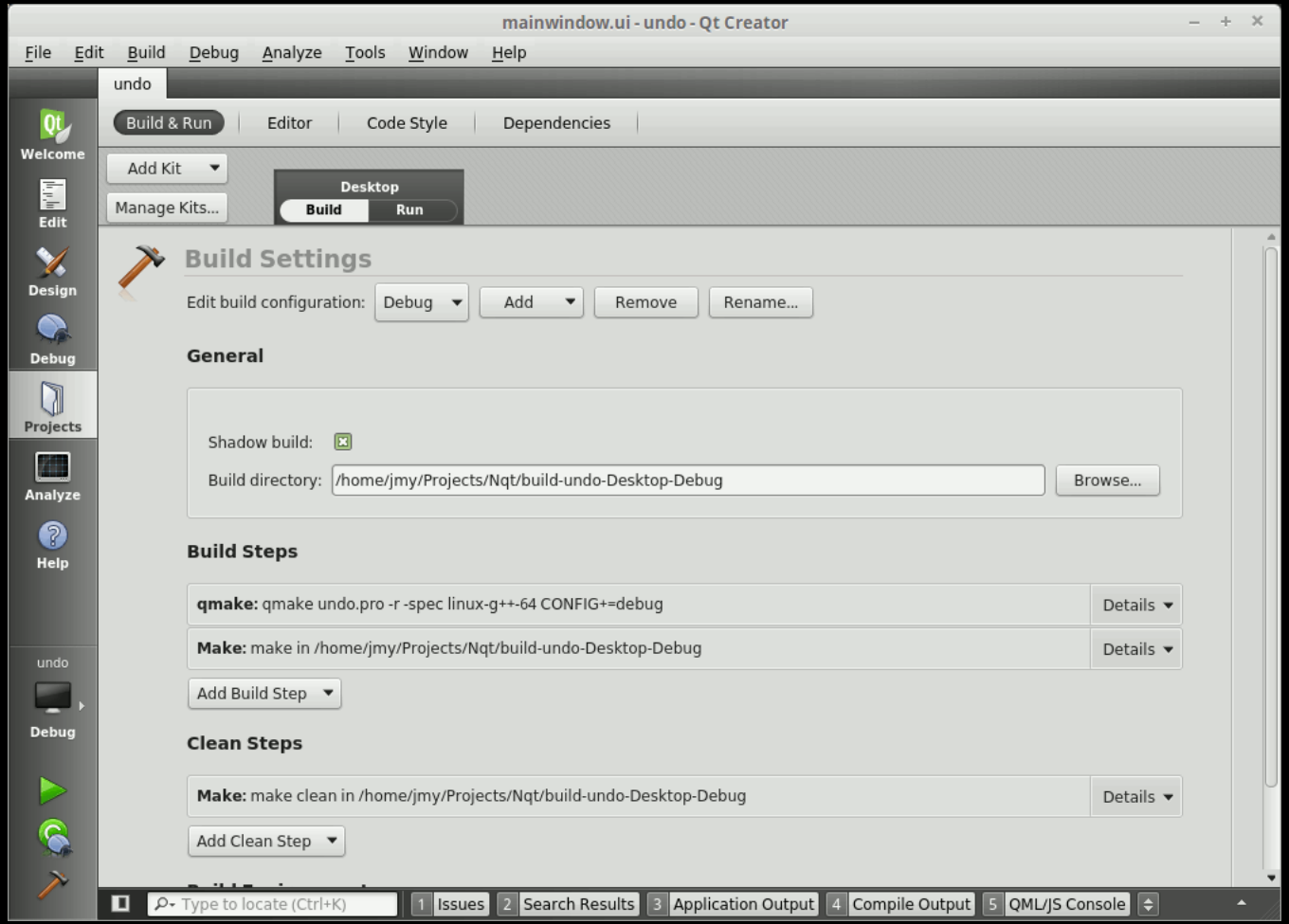

undo.pro - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Projects
undo
undo.pro
undo.qrc
commands.cpp
commands.h
document.cpp
document.h
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
shape.cpp
shape.h
shapelistmodel.cpp
shapelistmodel.h
undo.pro
Line: 19, Col: 27
INCLUDEQT += widgets
SOURCES += main.cpp mainwindow.cpp commands.cpp document.cpp \
shapelistmodel.cpp \
shape.cpp
HEADERS += mainwindow.h commands.h document.h \
shapelistmodel.h \
shape.h
FORMS += mainwindow.ui
build_all:!build_pass {
CONFIG -= build_all
CONFIG += release
}
RESOURCES += undo.qrc
INCLUDEPATH += /usr/include
LIBS += -L /usr/lib -l zip
Welcome
Edit
Design
Debug
Projects
Analyze
Help
Open Documents
calculator.pro
undo.pro
Application Output
undo
Starting /home/jmy/Projects/Nqt/build-undo-Desktop-Debug/undo...
/home/jmy/Projects/Nqt/build-undo-Desktop-Debug/undo exited with code 0
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console

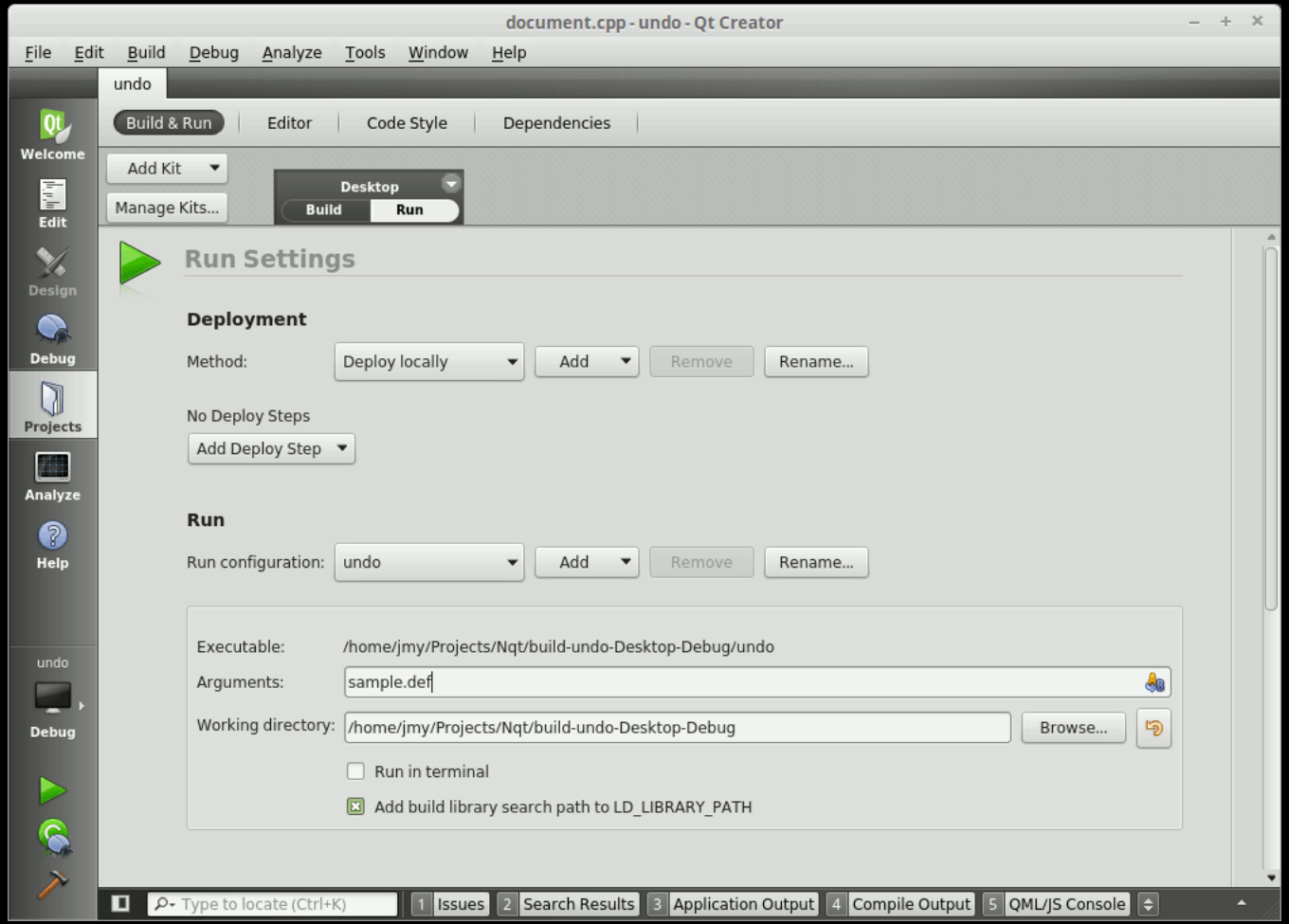

mainwindow.ui - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
mainwindow.ui
Filter
Layouts
Vertical Layout
Horizontal Layout
Grid Layout
Column View
Item Widgets (Item-Based)
List Widget
Tree Widget
Table Widget
Containers
Group Box
Edit action
Text: &Open
Object name: actionOpen
ToolTip: Open
Icon theme:
Icon: Normal Off
Checkable:
Shortcut: Press shortcut
Cancel
OK
File Edit Help Type Here
Tab 1
Name Used Text Shortcut
actionOpen &Open
actionClose &Close
actionNew &New
actionSave &Save
actionExit E&xit
Action Editor
Signals & Slots Editor
Object
actionExit
menuShape
separator
actionAddRectangle
actionRemoveShape
separator
separator
menuMacros
menuHelp
actionAbout
actionAboutQt
statusbar
toolBar
actionNew
actionOpen
actionOpen : QAction
Property Value
QObject
objectName actionOpen
QAction
checkable
checked
enabled
icon fileopen.p...
text &Open
iconText Open
toolTip Open
statusTip
Welcome
Edit
Debug
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console


mainwindow.ui - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Proiects
undo
undo.pro
undo.qrc
commands.cpp
commands.h
document.cpp
document.h
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
shape.cpp
shape.h
shapelistmodel.cpp
shapelistmodel.h
mainwindow.ui
Line: 843, Col: 1
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
<class>MainWindow</class>
<widget class="QMainWindow" name="MainWindow">
<property name="geometry">
<rect>
<x>0</x>
<y>0</y>
<width>817</width>
<height>822</height>
</rect>
</property>
<property name="iconSize">
<size>
<width>32</width>
<height>32</height>
</size>
</property>
<widget class="QWidget" name="centralwidget">
<layout class="QVBoxLayout">
<property name="spacing">
<number>0</number>
</property>
<property name="leftMargin">
<number>0</number>
</property>
Welcome
Edit
Design
Debug
Projects
Analyze
Help
Open Documents
calculator.pro
document.cpp
mainwindow.ui
undo.pro
Application Output
undo
Debugging has finished
undo
Debug
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console


undo.qrc - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Proiects
undo
undo.pro
undo.qrc
commands.cpp
commands.h
document.cpp
document.h
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
shape.cpp
shape.h
shapelistmodel.cpp
shapelistmodel.h
undo.qrc
/
icons/background.png
icons/blue.png
icons/circle.png
icons/exit.png
icons/fileclose.png
icons/filenew.png
icons/fileopen.png
icons/filesave.png
icons/green.png
icons/ok.png
icons/rectangle.png
Add
Remove
Remove Missing Files
Properties
Alias:
Prefix:
/
Language:
Welcome
Edit
Design
Debug
Projects
Analyze
Help
Open Documents
calculator.pro
document.cpp
mainwindow.ui
undo.pro
undo.qrc
Application Output
undo
Starting /home/jmy/Projects/Nqt/build-undo-Desktop-Debug/undo...
/home/jmy/Projects/Nqt/build-undo-Desktop-Debug/undo exited with code 0
Debugging starts
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console


document.cpp - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Projects
undo
undo.pro
undo.qrc
commands.cpp
commands.h
document.cpp
document.h
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
shape.cpp
shape.h
shapelistmodel.cpp
shapelistmodel.h
Open Documents
calculator.pro
document.cpp
mainwindow.ui
undo.pro
Welcome
Edit
Design
Debug
Projects
Analyze
Help
undo
Debug
#include <qevent.h>
#include <QPainter>
#include <QTextStream>
#include <QUndoStack>
#include "document.h"
#include "commands.h"
#include "shapelistmodel.h"
#include <QDebug>
#include <QtGlobal>
/**********
** Document
*/
Document::Document(QWidget *parent)
: QWidget(parent), m_currentIndex(-1), m_
{
m_undoStack = new QUndoStack(this);
setAutoFillBackground(true);
setBackgroundRole(QPalette::Base);
QPalette pal = palette();
pal.setBrush(QPalette::Base, QPixmap(":/i
pal.setColor(QPalette::HighlightedText, Q
setPalette(pal);
Name
Value
pal
@0x7fffffffdc
parent
0x0
this
@0x93d380
[QWidget]
@0x93d380
EDGE_POSITION_NOT_FOUND
(9999, 9999)
m_backgroudView
248
m_currentIndex
-1
m_drawingView
16
m_fileName
""
m_image_h
0
m_image_w
9787552
m_model
@0x923ba0
m_mousePressIndex
-1
m_mousePressOffset
(0, 0)
m_qimage
(invalid)
m_resizeHandlePressed
false
m_scale
4,6868173871
m_shapeList
<0 items>
m_std_macros
<0 items>
m_std_map
<0 items>
m_undoStack
@0x951340
staticMetaObject
@0x694960
[properties]
<>0 items>
[methods]
<2 items>
Threads: #1 undo
Stopped at breakpoint 1 (1) in threa
Views
Level
Function
File
Line
1
Document::Document
document.cpp
19
2
MainWindow::newDocument
mainwindow....
393
3
MainWindow::MainWindow
mainwindow....
92
4
main
main.cpp
56
Number
Function
File
1
Document::D...
/home/jmy/Pr...
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console


mainwindow.ui - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
undo
Build & Run
Editor
Code Style
Dependencies
Add Kit
Manage Kits...
Desktop
Build
Run
Welcome
Edit
Design
Debug
Projects
Analyze
Help
undo
Debug
Build Settings
Edit build configuration: Debug Add Remove Rename...
General
Shadow build:
Build directory: /home/jmy/Projects/Nqt/build-undo-Desktop-Debug
Browse...
Build Steps
qmake: qmake undo.pro -r -spec linux-g++-64 CONFIG+=debug
Details
Make: make in /home/jmy/Projects/Nqt/build-undo-Desktop-Debug
Details
Add Build Step
Clean Steps
Make: make clean in /home/jmy/Projects/Nqt/build-undo-Desktop-Debug
Details
Add Clean Step
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console


document.cpp - undo - Qt Creator
File Edit Build Debug Analyze Tools Window Help
undo
Build & Run
Editor
Code Style
Dependencies
Add Kit
Manage Kits...
Desktop
Build
Run
Welcome
Edit
Design
Debug
Projects
Analyze
Help
undo
Debug
Run Settings
Deployment
Method:
Deploy locally
Add
Remove
Rename...
No Deploy Steps
Add Deploy Step
Run
Run configuration:
undo
Add
Remove
Rename...
Executable:
/home/jmy/Projects/Nqt/build-undo-Desktop-Debug/undo
Arguments:
sample.def
Working directory:
/home/jmy/Projects/Nqt/build-undo-Desktop-Debug
Browse...
Run in terminal
Add build library search path to LD_LIBRARY_PATH
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML/JS Console

- komunikacja między obiektami - `signals` i `slots`
- odpytywalne i projektowalne właściwości obiektów
- zdarzenia i filtry zdarzeń
- internacjonalizacja napisów (i reszty aplikacji)
- hierarchiczne drzewo obiektów (własność obiektów)
- pointery zmieniające wartość na 0 gdy wskazywany obiekt skasowany (`QPointer`)
- dynamiczne rzutowanie działające nie tylko w ramach biblioteki

Qt Core - .

Qt GUI - .

Qt Widgets - .

Qt QML - .

Qt Quick - .

Qt Quick Controls - .

Qt Quick Layouts - .

Qt Network - .

Qt Designer - GUI design

Qt Linguist - Localization

qmake - cross-platform build system

- project
- makefiles

Qt Assistant

QtCreator

VS integration, Eclipse integration

Wartości

Aktorzy

`QObject`

Wartości

przechowują dane

Aktorzy

`QObject`

Wartości

przechowują dane
mogą być kopiowane

Aktorzy

`QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

Aktorzy

`QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

przykłady :

- QFont
- QColor
- QVariant
- QString

Aktorzy

`QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

przykłady :

- QFont
- QColor
- QVariant
- QString

Aktorzy

bierze udział w interakcji z użytkownikiem

`QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

przykłady :

- QFont
- QColor
- QVariant
- QString

Aktorzy

bierze udział w interakcji z użytkownikiem
może być kopiowane

`QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

przykłady :

- QFont
- QColor
- QVariant
- QString

Aktorzy

bierze udział w interakcji z użytkownikiem
może być kopiowane
nie mają dwóch takich samych

`QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

przykłady :

- QFont
- QColor
- QVariant
- QString

Aktorzy

bierze udział w interakcji z użytkownikiem
może być kopiowane
nie mamy dwóch takich samych
wszystkie dziedziczą z klasy `QObject`

Wartości

przechowują dane
mogą być kopiowane
gdy przechowują taką samą wartoć uważamy że są identyczne

przykłady :

- QFont
- QColor
- QVariant
- QString

Aktorzy

bierze udział w interakcji z użytkownikiem
może być kopiowane
nie mamy dwóch takich samych
wszystkie dziedziczą z klasy `QObject`
przykłady :

- QObject
- QWidget
- QGuiApplication

- Klasa bazowa dla wszystkich klas Qt

- Klasa bazowa dla wszystkich klas Qt
- Wprowadza:
  - uproszczone zarządzanie pamięcią

- Klasa bazowa dla wszystkich klas Qt
- Wprowadza:
  - uproszczone zarządzanie pamięcią
  - meta-obiekty

- Klasa bazowa dla wszystkich klas Qt
- Wprowadza:
  - uproszczone zarządzanie pamięcią
  - meta-obiekty
  - obsługa zdarzeń

- Klasa bazowa dla wszystkich klas Qt
- Wprowadza:
  - uproszczone zarządzanie pamięcią
  - meta-obiekty
  - obsługa zdarzeń
  - internacjonalizacja

- Klasa bazowa dla wszystkich klas Qt
- Wprowadza:
  - uproszczone zarządzanie pamięcią
  - meta-obiekty
  - obsługa zdarzeń
  - internacjonalizacja
- Relacja dziecko-rodzic

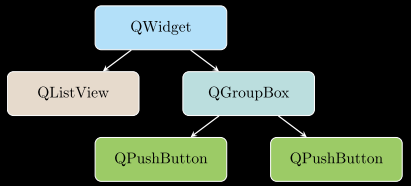
```
QObject * ob1 = new QObject;            // parent == nullptr
QObject * ob2 = new QObject( ob1 );     // parent == obj1
obj1.setParent(obj3);                   // parent == obj3
```

- Klasa bazowa dla wszystkich klas Qt
- Wprowadza:
  - uproszczone zarządzanie pamięcią
  - meta-obiekty
  - obsługa zdarzeń
  - internacjonalizacja
- Relacja dziecko-rodzic

```
QObject * ob1 = new QObject;            // parent == nullptr
QObject * ob2 = new QObject( ob1 );     // parent == obj1
obj1.setParent(obj3);                   // parent == obj3
```

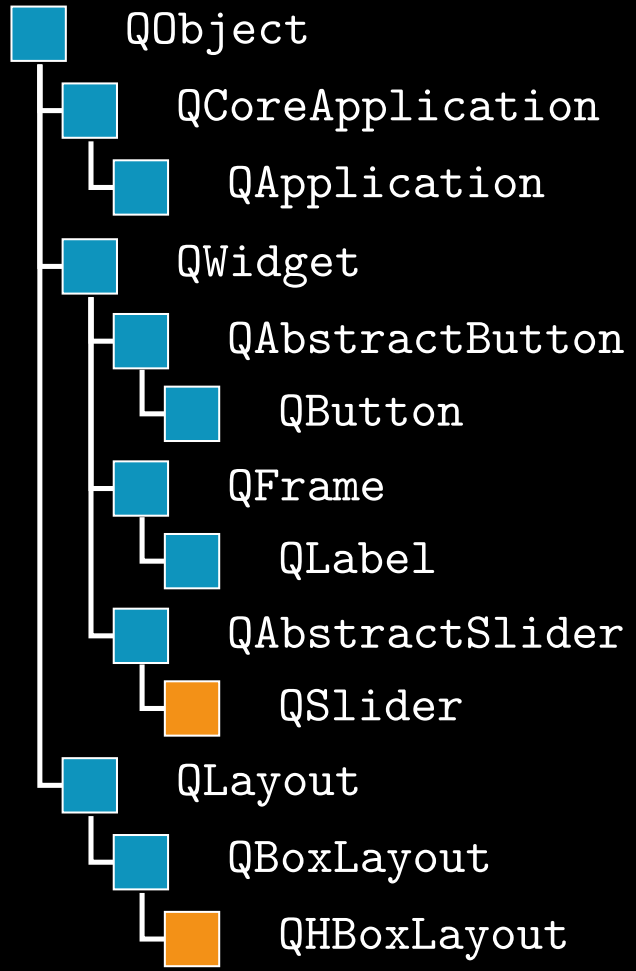
Usunięcie rodzica usuwa dzieci

- Interfejs budujemy jako drzewo, zależnosci typu dziecko-rodzic



- Usunięcie rodzica usuwa wszystkie dzieci

Drzewo dziedziczenia Qt (częściowe)


QObject
QCoreApplication
QApplication
QWidget
QAbstractButton
QButton
QFrame
QLabel
QAbstractSlider
QSlider
QLayout
QBoxLayout
QHBoxLayout

- informacja o klasie, introspekcja
- sygnały i sloty
- serializacja, tworzenie dynamiczne
- bezpieczne rzutowanie (type-safe)
- `QVariant` - typ umożliwiający przechowywanie wartości różnych typów
- dynamiczne właściwości
- dostęp po nazwie lub indeksie

- klasa `QObject`

- klasa `QObject`
  - klasa bazowa dla wszystkich meta-obiektów

- klasa `QObject`
  - klasa bazowa dla wszystkich meta-obiektów
- makro `Q_OBJECT`

- klasa `QObject`
  - klasa bazowa dla wszystkich meta-obiektów
- makro `Q_OBJECT`
  - tworzy funkcjonalność meta-obiektu

- klasa `QObject`
  - klasa bazowa dla wszystkich meta-obiektów
- makro `Q_OBJECT`
  - tworzy funkcjonalność meta-obiektu
- kompilator meta-obiektów (`moc`)

className() - zwraca nazwę klasy.

className() - zwraca nazwę klasy.

superClass() - zwraca meta-object klasy.

className() - zwraca nazwę klasy.

superClass() - zwraca meta-object klasy.

method() oraz methodCount() - zwraca informację o meta-metodach klasy (sygnałach, slotach oraz innych funkcjach klasy)

className() - zwraca nazwę klasy.

superClass() - zwraca meta-object klasy.

method() oraz methodCount() - zwraca informację o meta-metodach klasy (sygnałach, slotach oraz innych funkcjach klasy)

enumerator() oraz enumeratorCount() - zwraca informację o enumeratorach klasy.

className() - zwraca nazwę klasy.

superClass() - zwraca meta-object klasy.

method() oraz methodCount() - zwraca informację o meta-metodach klasy (sygnałach, slotach oraz innych funkcjach klasy)

enumerator() oraz enumeratorCount() - zwraca informację o enumeratorach klasy.

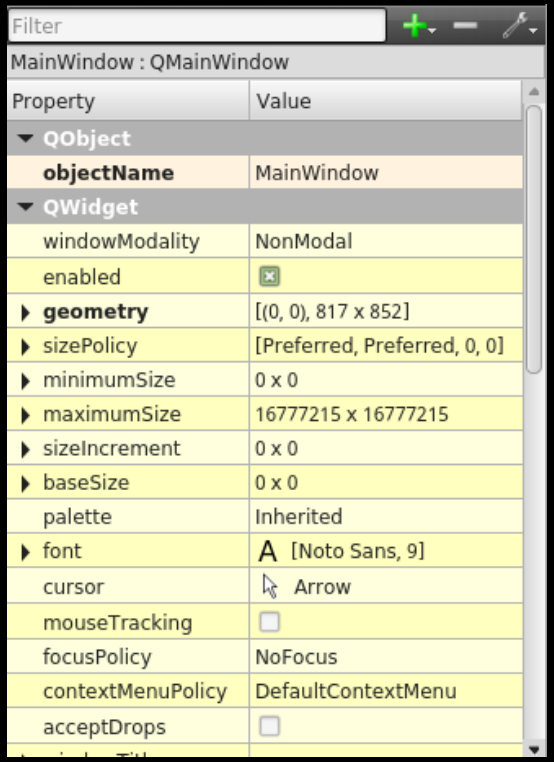
propertyCount() oraz property() - zwraca informacje o właściwościach zdefiniowanych w klasie.

```
#ifndef MYCL2_H
#define MYCL2_H
#include <QObject>

class MyCl2 : public QObject {
    Q_OBJECT
    Q_ENUMS(MyEnum)
public:
    MyCl2();
    enum MyEnum {A0 = 0, A1, A2, A3, A_E};
};
#endif // MYCL2_H
```

```
#ifndef MYCL2_H
#define MYCL2_H
#include <QObject>

class MyCl2 : public QObject {
    Q_OBJECT
    Q_ENUMS(MyEnum)
public:
    MyCl2();
    enum MyEnum {A0 = 0, A1, A2, A3, A_E};
};
#endif // MYCL2_H
```

- funkcja `read`
- (opcjonalna) funkcja `write`
- .atrybut `stored` zaznaczający trwałość
  - właściwości ulotne - wirtualne - `QWidget::minimumWidth()` nie jest zapisywane, to tylko obliczany widok `QWidget::minimumSize()`.
- funkcja `reset` (do ustawienia wartości domyślnej zależnej od kontekstu)
- atrybut `designable` określa czy właściwoś widoczna w narzędziach GUI takich jak QtDesigner

```
Q_PROPERTY(type name READ getFunction
           [WRITE setFunction]
           [RESET resetFunction]
           [DESIGNABLE bool]
           [SCRIPTABLE bool]
           [STORED bool])
```



```
class MyLevel : public QObject {
    Q_OBJECT
    Q_PROPERTY(Level level READ level WRITE setLevel)
    Q_ENUMS(Level)
public:
    MyLevel(QObject *parent = 0);
    ~MyLevel();
    enum Level { High, Low, VeryHigh, VeryLow };
    void setLevel(Level level);
    Level level() const;
};
```

```
QPushButton *button = new QPushButton;
QObject *object = button;
// button and object point to the same object
button->setDown(true);
object->setProperty("down", true);

QVariant property(const QString &name)
void setProperty(const QString &name,
const QVariant &value)
```

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`
  - i inne

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`
  - i inne
- narzędzie `moc` z Qt generuje implementację tych funkcji `Q_OBJECT`

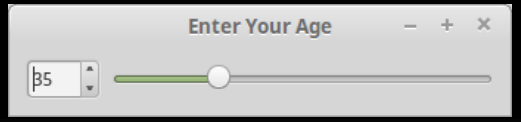
- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`
  - i inne
- narzędzie `moc` z Qt generuje implementację tych funkcji `Q_OBJECT`

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`
  - i inne
- narzędzie `moc` z Qt generuje implementację tych funkcji `Q_OBJECT`
- metody `QObject` takie jak `connect()` oraz `disconnect()` wykorzystują introspekcję

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`
  - i inne
- narzędzie `moc` z Qt generuje implementację tych funkcji `Q_OBJECT`
- metody `QObject` takie jak `connect()` oraz `disconnect()` wykorzystują introspekcję
- Wszystko powyższe robione jest automatycznie przez qmake, moc, oraz `QObject` zatem normalnie o tym nie myślimy

- Makro `Q_OBJECT` deklaruje kilka funkcji które muszą być zdefiniowane w każdym obiekcie dziedziczącym z `QObject`
  - `metaObject()`
  - `className()`
  - `tr()`
  - i inne
- narzędzie `moc` z Qt generuje implementację tych funkcji `Q_OBJECT`
- metody `QObject` takie jak `connect()` oraz `disconnect()` wykorzystują introspekcję
- Wszystko powyższe robione jest automatycznie przez qmake, moc, oraz `QObject` zatem normalnie o tym nie myślimy
- Oczywiście jak chcemy, to możemy sobie pooglądać ten wugenerowany kod

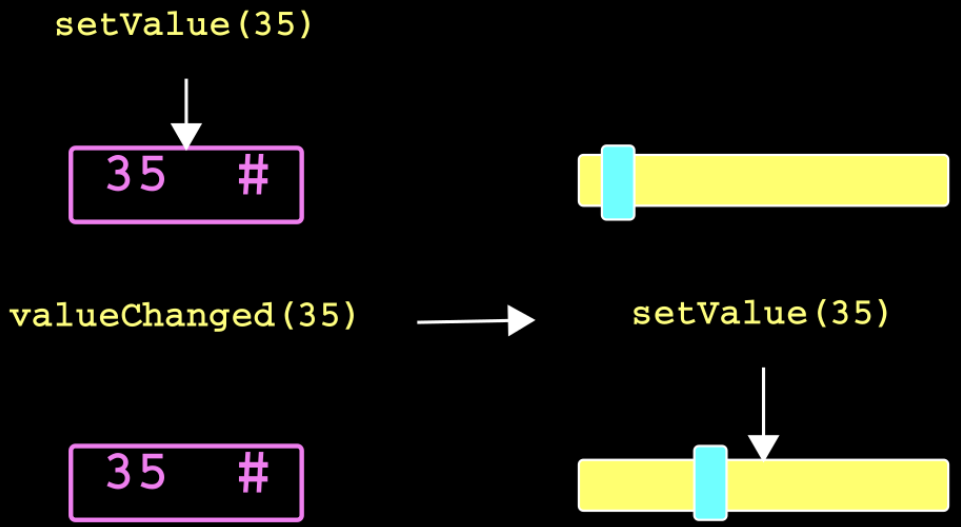
Sygnały i sloty (1)




```
#include <QApplication>
#include <QHBoxLayout>
#include <QSlider>
#include <QSpinBox>
int main(int argc, char *argv[]) {
    QApplication app(argc, argv);
    QWidget *window = new QWidget;
    window->setWindowTitle("Enter Your Age");
    QSpinBox *spinBox = new QSpinBox;
    QSlider *slider = new QSlider(Qt::Horizontal);
    spinBox->setRange(0, 130);   slider->setRange(0, 130);
    QObject::connect(spinBox, SIGNAL(valueChanged(int)), slider, SLOT(setValue(int)));
    QObject::connect(slider, SIGNAL(valueChanged(int)), spinBox, SLOT(setValue(int)));
    spinBox->setValue(35);
    QHBoxLayout *layout = new QHBoxLayout;
    layout->addWidget(spinBox);  layout->addWidget(slider);
    window->setLayout(layout);
    window->show();
    return app.exec();
}
```

- Podstawowy mechanizm komunikacji w Qt
- Łączy obiekty bez konieczności wprowadzania zależności w kodzie tych obiektów

```
1 connect(sender, SIGNAL(signal), receiver, SLOT(slot));
```



- Jak zwykłe metody C++
- Może być wirtualna
- Może być przeciążone (overload)
- Może być publiczne, protected i prrwatne
- Może być wywołane bezpośrednio (jak zwykła metoda)
- Może być podłączone do sygnału

było:

```
connect(lineEdit, SIGNAL(textChanged(QString)), label, SLOT(setText(QString)));
```

może być

```
connect(lineEdit, &QLineEdit::textChanged, label, &QLabel::setText);
```

- Wykorzystujemy pointery do metod jako sygnały i sloty
- sprawdzane przez kompilator
- argumenty rzutowane do właściwych typów
- możemy użyć wyrażeń lambda

```
connect(lineEdit, &QLineEdit::textChanged, [=](QString s) {
    qDebug() << "text changed: " << s;
});
```

- Jeden sygnał możemy podłączyć do wielu slotów

- Jeden sygnał możemy podłączyć do wielu slotów
- Wiele sygnałów możemy podłączyć do tego samego slotu

- Jeden sygnał możemy podłączyć do wielu slotów
- Wiele sygnałów możemy podłączyć do tego samego slotu
- Sygnał możemy podłączyć do innego sygnału - emisja pierwszego uruchamia emisję drugiego

- Jeden sygnał możemy podłączyć do wielu slotów
- Wiele sygnałów możemy podłączyć do tego samego slotu
- Sygnał możemy podłączyć do innego sygnału - emisja pierwszego uruchamia emisję drugiego
- Qt automatycznie usuwa połączenia w których występuje kasowany obiekt (ale można zawołać `disconnect` ręcznie)

- Jeden sygnał możemy podłączyć do wielu slotów
- Wiele sygnałów możemy podłączyć do tego samego slotu
- Sygnał możemy podłączyć do innego sygnału - emisja pierwszego uruchamia emisję drugiego
- Qt automatycznie usuwa połączenia w których występuje kasowany obiekt (ale można zawołać `disconnect` ręcznie)
- Aby poprawnie powiązać sygnał ze slotem (lub innym sygnałem) muszą mieć parametry tych samych typów w tej samej kolejności

```
1 connect(ftp, SIGNAL(rawCommandReply(int, const QString &)),
2         this, SLOT(processReply(int, const QString &)));
```

- Jeden sygnał możemy podłączyć do wielu slotów
- Wiele sygnałów możemy podłączyć do tego samego slotu
- Sygnał możemy podłączyć do innego sygnału - emisja pierwszego uruchamia emisję drugiego
- Qt automatycznie usuwa połączenia w których występuje kasowany obiekt (ale można zawołać `disconnect` ręcznie)
- Aby poprawnie powiązać sygnał ze slotem (lub innym sygnałem) muszą mieć parametry tych samych typów w tej samej kolejności

```
1 connect(ftp, SIGNAL(rawCommandReply(int, const QString &)),
2         this, SLOT(processReply(int, const QString &)));
```

- Ale gdy sygnał ma więcej parametrów niż slot to dodatkowe parametry ignorowane:

```
1 connect(ftp, SIGNAL(rawCommandReply(int, const QString &)),
2         this, SLOT(checkErrorCode(int)));
```

- Przykład Convention over Configuration

```
class ImageDialog : public QDialog, private Ui::ImageDialog {
    Q_OBJECT
public:
    ImageDialog(QWidget *parent = 0);
private slots:
    void on_okButton_clicked();
};
```

- Klasa `Ui::ImageDialog` to klasa wygenerowana przez QtDesignera z deklaracjami pól

```
class Ui::ImageDialog {
public:
    QPushButton *okButton;
    ...
    void setupUi(QMainWindow *MainWindow) {
        ...
        retranslateUi(MainWindow);
        QMetaObject::connectSlotsByName(MainWindow);
    } // setupUi

    void retranslateUi(QMainWindow *MainWindow) {
        actionOpen->setText(QApplication::translate("MainWindow", "&Open", 0));
        ...
```

- Przykład Convention over Configuration

```
void on_<object name>_<signal name>(<signal parameters>);
```

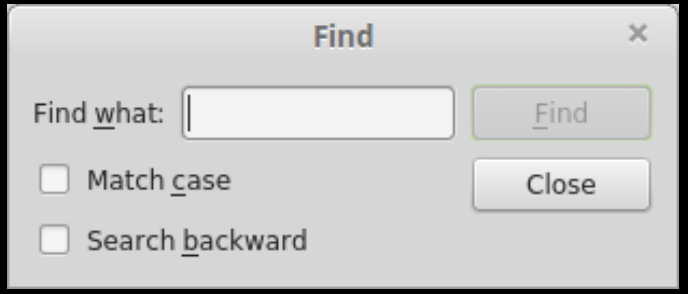
- Jeśli slot będzie miał odpowiednią nazwę, zostanie on podłączone do sygnału twojej kontrolki

```
    QPushButton *okButton;
    ...
private slots:
    void on_okButton_clicked();
```

Actions

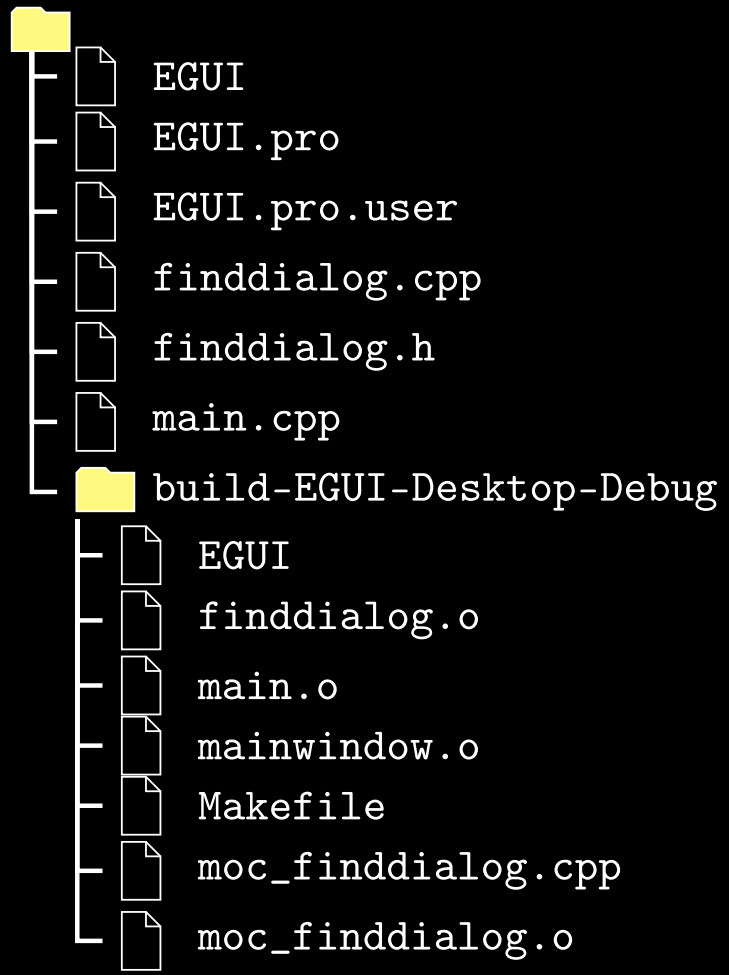
- `QAction` ma ygnały, do których podłączamy sloty
- Ma tytuł i ikony (oddzielne dla stanu wciśniętego i zwolnionego)
- Ma podpowiedzi (tooltips)
- Ta sama akcja może być połączona z kilkoma kontrolkami (takimi jak pasek narzędzi lub menu) w tym samym czasie

Przykład (1) pokazuje:



- tworzenie slotów
- podłączanie slotów do sygnałów generowanych przez standardowe kontrolki
- tworzenie własnych sygnałów
- tworzenie GUI w kodzie

```
#include "finddialog.h"
#include <QApplication>

int main(int argc, char *argv[]) {
    QApplication a(argc, argv);
    FindDialog w;
    w.show();
    return a.exec();
}
```


EGUI
EGUI.pro
EGUI.pro.user
finddialog.cpp
finddialog.h
main.cpp
build-EGUI-Desktop-Debug
EGUI
finddialog.o
main.o
mainwindow.o
Makefile
moc_finddialog.cpp
moc_finddialog.o

```
#-------------------------------------------------
#
# Project created by QtCreator 2017-03-23T11:19:54
#
#-------------------------------------------------
QT       += core gui
greaterThan(QT_MAJOR_VERSION, 4): QT += widgets
TARGET = EGUI
TEMPLATE = app
SOURCES += main.cpp\
    finddialog.cpp

HEADERS  += \
    finddialog.h
FORMS    +=
```

- Zawsze powinniśmy stosować warunkowe dołączanie, aby zapobiec wielokrotnemu włączaniu
- Widżety są deklarowane w osobnych plikach nagłówkowych

```
#ifndef FINDDIALOG_H
#define FINDDIALOG_H

#include <QtWidgets/QDialog>
#include <QtWidgets/QCheckBox>
#include <QtWidgets/QLabel>
#include <QtWidgets/QLineEdit>
#include <QtWidgets/QPushButton>

class FindDialog : public QDialog {
...
#endif
```

- Aby użyć Qt metasystemu, powinniśmy zadeklarować Q_OBJECT w klasie

```
class FindDialog : public QDialog {
    Q_OBJECT
public:
    FindDialog(QWidget *parent = 0);
signals:
    void findNext(const QString &str, Qt::CaseSensitivity cs);
    void findPrevious(const QString &str, Qt::CaseSensitivity cs);
private slots:
    void findClicked();
    void enableFindButton(const QString &text);
private:
    QLabel *label;
    QLineEdit *lineEdit;
    QCheckBox *caseCheckBox;
    QCheckBox *backwardCheckBox;
    QPushButton *findButton;
    QPushButton *closeButton;
};
```

```
/****************************************h*******************************
** Meta object code from reading C++ file 'finddialog.h'
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.5.1)
**
** WARNING! All changes made in this file will be lost!
*****************************************************************************/

#include "../EGUI/finddialog.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'finddialog.h' doesn't include <QObject>."
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.5.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
struct qt_meta_stringdata_FindDialog_t {
    QByteArrayData data[10];
    char stringdata0[95];
};
#define QT_MOC_LITERAL(idx, ofs, len) \
    Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
    qptrdiff(offsetof(qt_meta_stringdata_FindDialog_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )

static const qt_meta_stringdata_FindDialog_t qt_meta_stringdata_FindDialog = {
    {
QT_MOC_LITERAL(0, 0, 10), // "FindDialog"
QT_MOC_LITERAL(1, 11, 8), // "findNext"
QT_MOC_LITERAL(2, 20, 0), // ""
QT_MOC_LITERAL(3, 21, 3), // "str"
QT_MOC_LITERAL(4, 25, 19), // "Qt::CaseSensitivity"
QT_MOC_LITERAL(5, 45, 2), // "cs"
QT_MOC_LITERAL(6, 48, 12), // "findPrevious"
QT_MOC_LITERAL(7, 61, 11), // "findClicked"
```

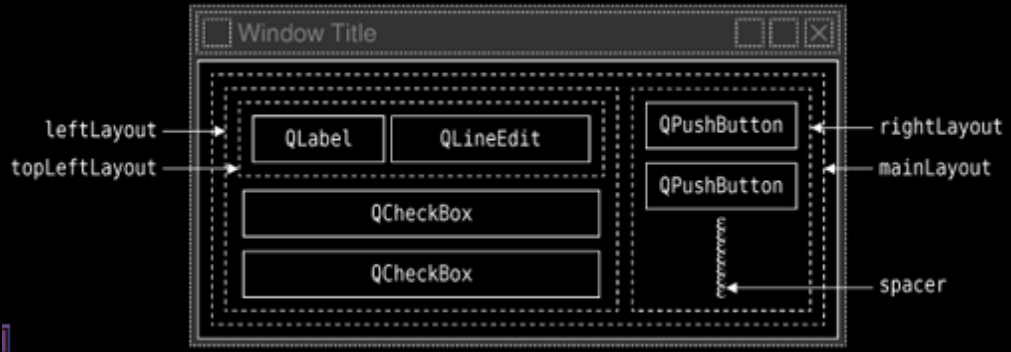
```
QT_MOC_LITERAL(8, 73, 16), // "enableFindButton"
QT_MOC_LITERAL(9, 90, 4) // "text"
    },

    "FindDialog\0findNext\0\0str\0Qt::CaseSensitivity\0"
    "cs\0findPrevious\0findClicked\0"
    "enableFindButton\0text"
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_FindDialog[] = {

 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
       4,   14, // methods
       0,    0, // properties
       0,    0, // enums/sets
       0,    0, // constructors
       0,       // flags
       2,       // signalCount
 // signals: name, argc, parameters, tag, flags
       1,    2,   34,    2, 0x06 /* Public */,
       6,    2,   39,    2, 0x06 /* Public */,

 // slots: name, argc, parameters, tag, flags
       7,    0,   44,    2, 0x08 /* Private */,
       8,    1,   45,    2, 0x08 /* Private */,

 // signals: parameters
    QMetaType::Void, QMetaType::QString, 0x80000000 | 4,    3,    5,
    QMetaType::Void, QMetaType::QString, 0x80000000 | 4,    3,    5,

 // slots: parameters
    QMetaType::Void,
    QMetaType::Void, QMetaType::QString,    9,

       0        // eod
};
void FindDialog::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
    if (_c == QMetaObject::InvokeMetaMethod) {
        FindDialog *_t = static_cast<FindDialog *>(_o);
```

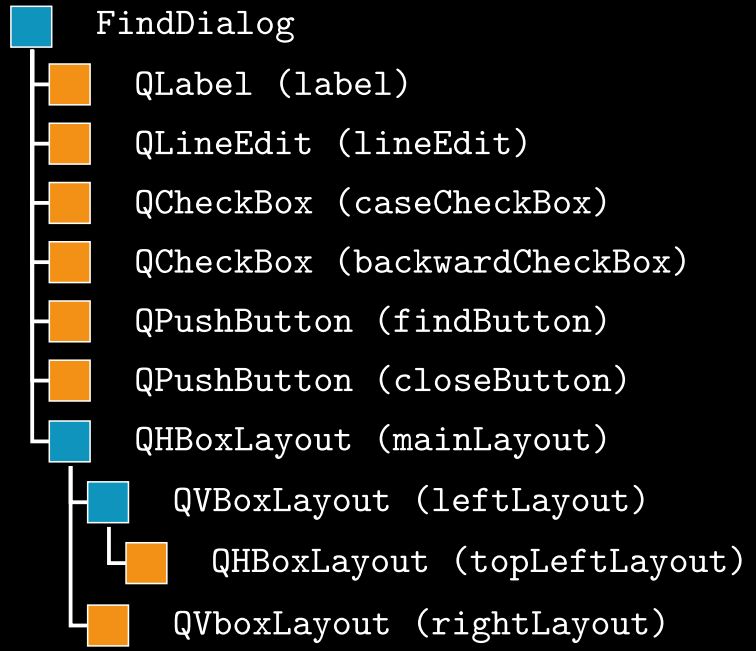
```
85          Q_UNUSED(_t)
86          switch (_id) {
87          case 0: _t->findNext((*reinterpret_cast< const QString(*)>(_a[1])),(*reinterpret_cast<
            ↪  Qt::CaseSensitivity(*)>(_a[2]))); break;
88          case 1: _t->findPrevious((*reinterpret_cast< const QString(*)>(_a[1])),(*reinterpret_cast<
            ↪  Qt::CaseSensitivity(*)>(_a[2]))); break;
89          case 2: _t->findClicked(); break;
90          case 3: _t->enableFindButton((*reinterpret_cast< const QString(*)>(_a[1]))); break;
91          default: ;
92          }
93      } else if (_c == QMetaObject::IndexOfMethod) {
94          int *result = reinterpret_cast<int *>(_a[0]);
95          void **func = reinterpret_cast<void **>(_a[1]);
96          {
97              typedef void (FindDialog::*_t)(const QString & , Qt::CaseSensitivity );
98              if (*reinterpret_cast<_t *>(func) == static_cast<_t>(&FindDialog::findNext)) {
99                  *result = 0;
100             }
101         }
102         {
103             typedef void (FindDialog::*_t)(const QString & , Qt::CaseSensitivity );
104             if (*reinterpret_cast<_t *>(func) == static_cast<_t>(&FindDialog::findPrevious)) {
105                 *result = 1;
106             }
107         }
108     }
109 }
110
111 const QMetaObject FindDialog::staticMetaObject = {
112     { &QDialog::staticMetaObject, qt_meta_stringdata_FindDialog.data,
113       qt_meta_data_FindDialog,  qt_static_metacall, Q_NULLPTR, Q_NULLPTR}
114 };
115
116
117 const QMetaObject *FindDialog::metaObject() const
118 {
119     return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() : &staticMetaObject;
120 }
121 void *FindDialog::qt_metacast(const char *_clname)
122 {
123     if (!_clname) return Q_NULLPTR;
124     if (!strcmp(_clname, qt_meta_stringdata_FindDialog.stringdata0))
125         return static_cast<void*>(const_cast< FindDialog*>(this));
126     return QDialog::qt_metacast(_clname);
127 }
```

```
128
129 int FindDialog::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
130 {
131     _id = QDialog::qt_metacall(_c, _id, _a);
132     if (_id < 0)
133         return _id;
134     if (_c == QMetaObject::InvokeMetaMethod) {
135         if (_id < 4)
136             qt_static_metacall(this, _c, _id, _a);
137         _id -= 4;
138     } else if (_c == QMetaObject::RegisterMethodArgumentMetaType) {
139         if (_id < 4)
140             *reinterpret_cast<int*>(_a[0]) = -1;
141         _id -= 4;
142     }
143     return _id;
144 }
145 // SIGNAL 0
146 void FindDialog::findNext(const QString & _t1, Qt::CaseSensitivity _t2)
147 {
148     void *_a[] = { Q_NULLPTR, const_cast<void*>(reinterpret_cast<const void*>(&_t1)),
      ↪  const_cast<void*>(reinterpret_cast<const void*>(&_t2)) };
149     QMetaObject::activate(this, &staticMetaObject, 0, _a);
150 }
151
152 // SIGNAL 1
153 void FindDialog::findPrevious(const QString & _t1, Qt::CaseSensitivity _t2)
154 {
155     void *_a[] = { Q_NULLPTR, const_cast<void*>(reinterpret_cast<const void*>(&_t1)),
      ↪  const_cast<void*>(reinterpret_cast<const void*>(&_t2)) };
156     QMetaObject::activate(this, &staticMetaObject, 1, _a);
157 }
158 QT_END_MOC_NAMESPACE
```

- Aby użyć Qt metasystemu, powinniśmy zadeklarować Q_OBJECT w klasie

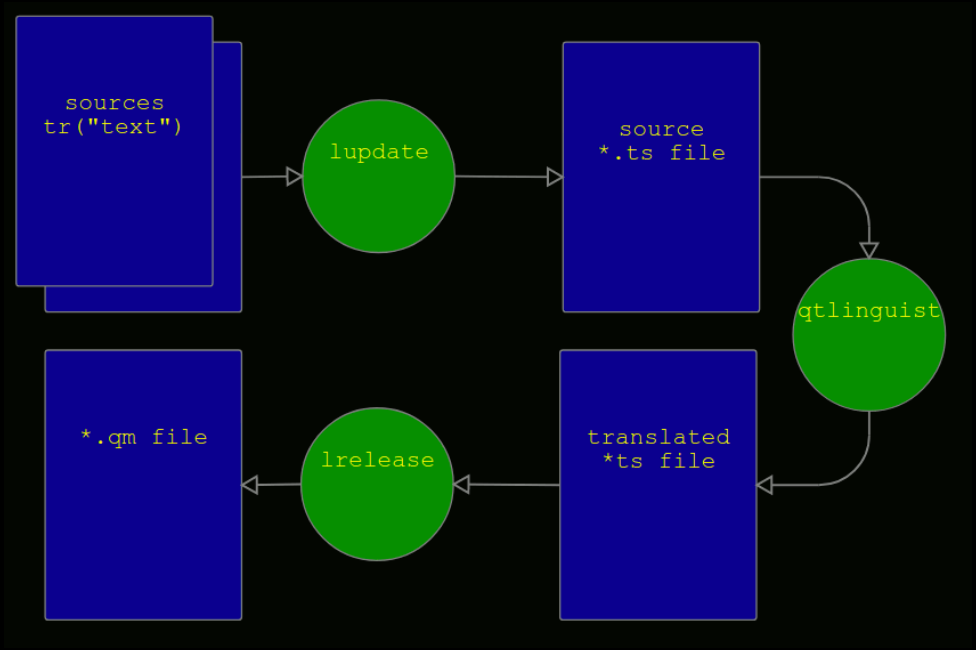
```
class FindDialog : public QDialog {
    Q_OBJECT
public:
    FindDialog(QWidget *parent = 0);
signals:
    void findNext(const QString &str, Qt::CaseSensitivity cs);
    void findPrevious(const QString &str, Qt::CaseSensitivity cs);
private slots:
    void findClicked();
    void enableFindButton(const QString &text);
private:
    QLabel *label;
    QLineEdit *lineEdit;
    QCheckBox *caseCheckBox;
    QCheckBox *backwardCheckBox;
    QPushButton *findButton;
    QPushButton *closeButton;
};
```


Window Title
leftLayout
topLeftLayout
QLabel
QLineEdit
QCheckBox
QCheckBox
QPushButton
QPushButton
rightLayout
mainLayout
spacer

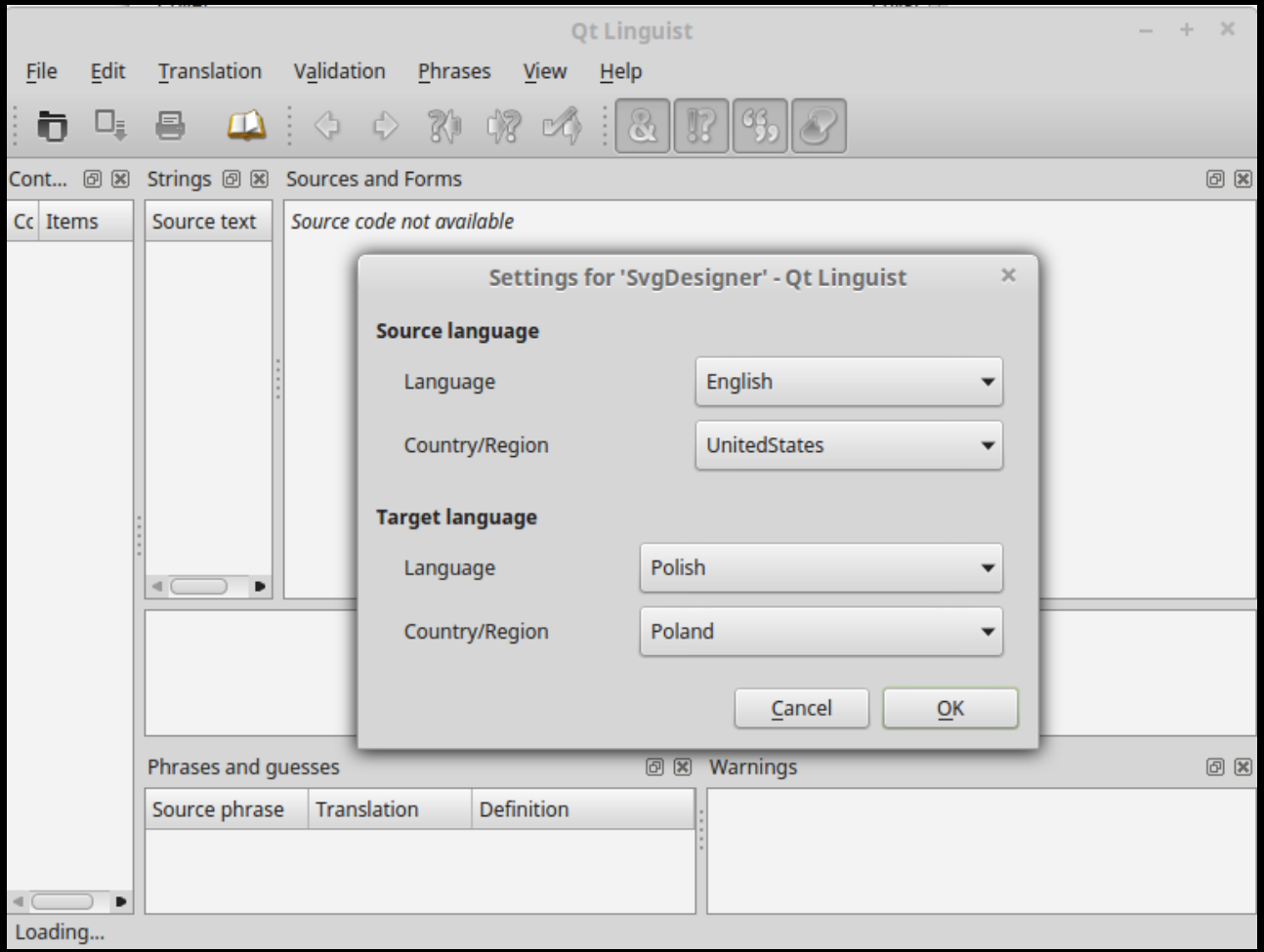

FindDialog
QLabel (label)
QLineEdit (lineEdit)
QCheckBox (caseCheckBox)
QCheckBox (backwardCheckBox)
QPushButton (findButton)
QPushButton (closeButton)
QHBoxLayout (mainLayout)
QVBoxLayout (leftLayout)
QHBoxLayout (topLeftLayout)
QVboxLayout (rightLayout)

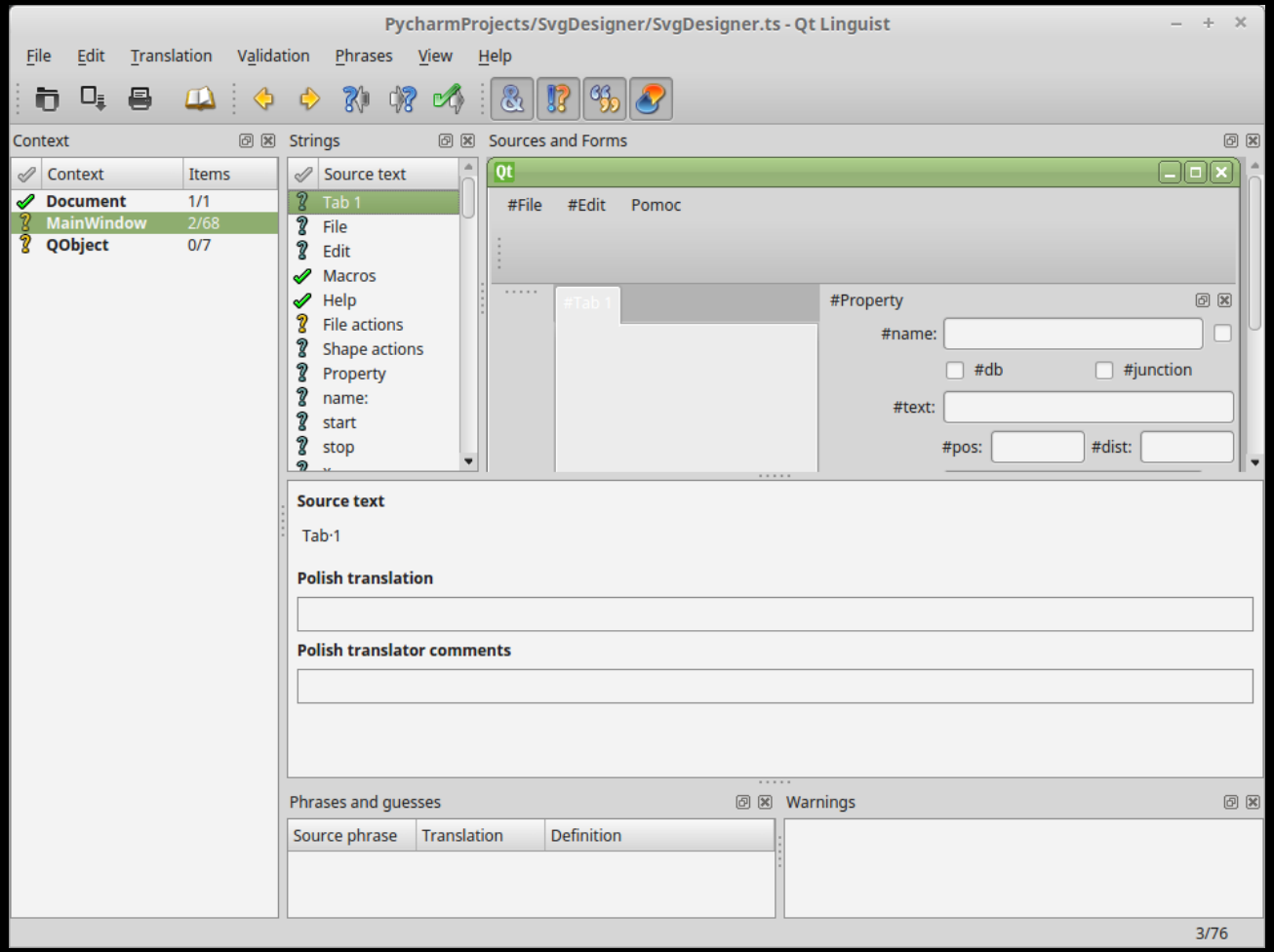
- Gdy chcemy coś przetłumaczyć:

```
1 if (!file.open(QIODevice::ReadOnly)) {
2     QMessageBox::warning(this,
3                          tr("File error"),
4                          tr("Failed to open\n%1").arg(fileName));
5     return false;
6 }
```



```
lupdate -pro SvgDesigner.pro -ts SvgDesigner-pl.ts
Updating 'SvgDesigner-pl.ts'...
    Found 76 source text(s) (76 new and 0 already existing)
```

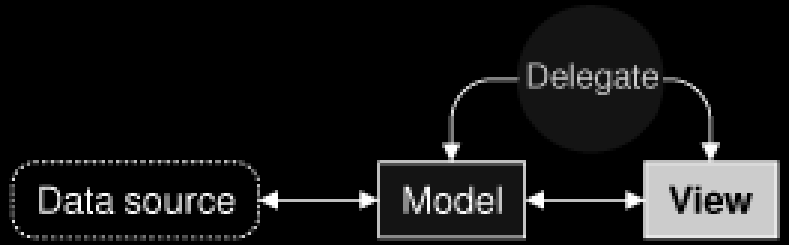

Qt Linguist
File
Edit
Translation
Validation
Phrases
View
Help
Cont...
Strings
Sources and Forms
Cc
Items
Source text
Source code not available
Settings for 'SvgDesigner' - Qt Linguist
Source language
Language
English
Country/Region
UnitedStates
Target language
Language
Polish
Country/Region
Poland
Cancel
OK
Phrases and guesses
Source phrase
Translation
Definition
Warnings
Loading...


PycharmProjects/SvgDesigner/SvgDesigner.ts - Qt Linguist
File Edit Translation Validation Phrases View Help
Context
Context Items
Document 1/1
MainWindow 2/68
QObject 0/7
Strings
Source text
Tab 1
File
Edit
Macros
Help
File actions
Shape actions
Property
name:
start
stop
Sources and Forms
Qt
#File #Edit Pomoc
#Tab 1
#Property
#name:
#db #junction
#text:
#pos: #dist:
Source text
Tab·1
Polish translation
Polish translator comments
Phrases and guesses
Source phrase Translation Definition
Warnings
3/76

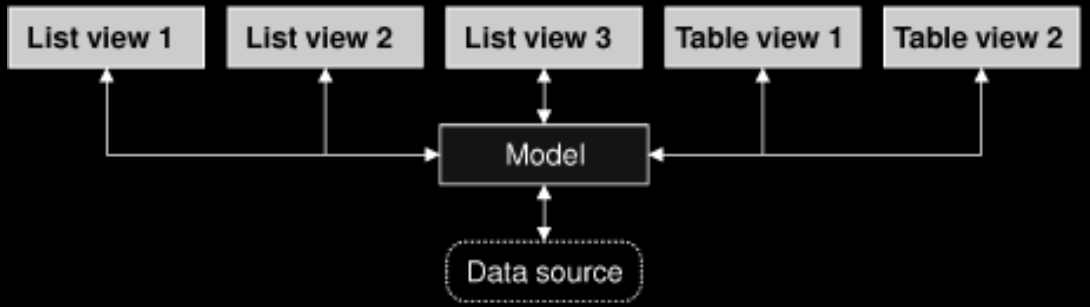
```
lrelease SvgDesigner.ts  SvgDesigner.qm
Updating 'SvgDesigner.qm'...
    Generated 4 translation(s) (3 finished and 1 unfinished)
    Ignored 72 untranslated source text(s)
Updating 'SvgDesigner.qm'...
    Generated 4 translation(s) (4 finished and 0 unfinished)
```

```cpp
#include <QtWidgets/QApplication>
#include <QtCore/QTranslator>
#include "mainwindow.h"

int main(int argc, char **argv) {
    Q_INIT_RESOURCE(SvgDesigner);

    QApplication app(argc, argv);

    QTranslator translator;
    bool result = translator.load("../SvgDesigner/SvgDesigner.qm");
    app.installTranslator(&translator);

    MainWindow win(starts);
    win.resize(800, 600);
    win.show();

    return app.exec();
}
```

- Wprowadzone w Qt4
- Separuje dane od ich prezentacji
- Umożliwia przedstawianie danych tworzonych (wyliczanych) w locie



- Tego samego modelu możemy użyć wielokrotnie

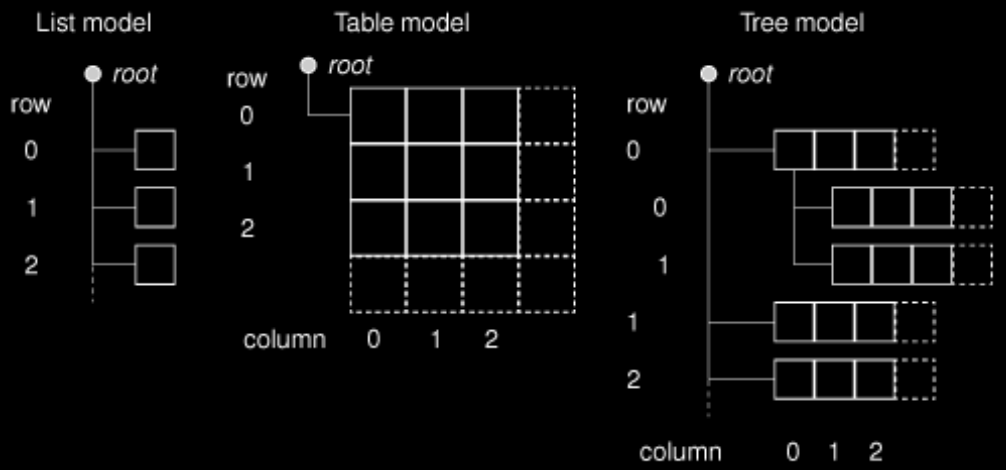


- liczność danych
  - int `rowCount`(const QModelIndex &parent) const
  - int `columnCount`(const QModelIndex &parent) const

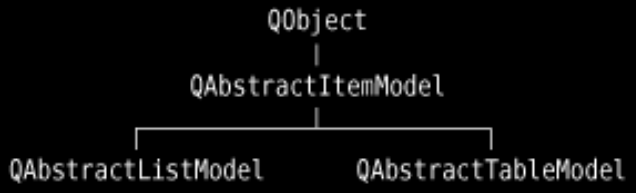
- liczność danych
  - int `rowCount`(const QModelIndex &parent) const
  - int `columnCount`(const QModelIndex &parent) const
- położenie danych
  - QModelIndex `index`(int row, int column, const QModelIndex &parent) const
  - QModelIndex `parent`(const QModelIndex &index) const

- liczność danych
  - int `rowCount`(const QModelIndex &parent) const
  - int `columnCount`(const QModelIndex &parent) const
- położenie danych
  - QModelIndex `index`(int row, int column, const QModelIndex &parent) const
  - QModelIndex `parent`(const QModelIndex &index) const
- dostęp do danych
  - Qt::ItemFlags `flags`(const QModelIndex &index) const
  - QHashint,QByteArray `roleNames`() const
  - QVariant `data`(const QModelIndex &index, int role) const
  - bool `setData`(const QModelIndex &index, const QVariant &value, int role)

- Istnieje szereg modeli predefiniowanych



- Ale wszystkie one są dziedziczone z tych samych klas bazowych



klasy bazowe:

QAbstractItemModel

QAbstractTableModel

QAbstractListModel

standardowe modele

QStringListModel -przechowuje listę stringów

QStandardItemModel -przechowuje dowolne dane hierarchiczne

QDirModel -opakowywuje odwołania do systemu plików

QSqlQueryModel -opakowywuje wynik zapytania SQL

QSqlTableModel -opakowywuje tablicę SQL

QSqlRelationalTableModel -opakowywuje tablicę SQL zawierającą odwołania do innych tablic

QSortFilterProxyModel -umożliwia sortowanie i filtrowanie innych modeli

Parę standardowych widoków

QListView -List or icon view onto a model

QTableView -Default model/view implementation of a table view

QTreeView -Default model/view implementation of a tree view

- model jest tylko interfejsem do danych - same dane nie muszą być przechowywane w modelu
- delegat musi narysować element, obliczyć jego rozmiar, zsynchronizować dane pomiędzy edytorem a modelem, obsługę zdarzenia, stworzyć odpowiedni widget edytora dla elementu
- `QDataWidgetWrapper` jest pseudo-widokiem do edycji formularzy dla elementu

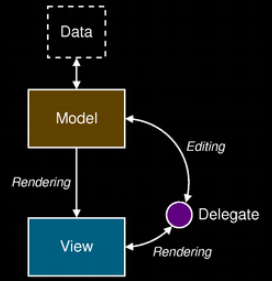
Dodawanie oraz usuwanie wierszy i kolumn

```
bool insertRows(int row, int count, const QModelIndex &parent = QModelIndex())
bool removeRows(int row, int count, const QModelIndex &parent = QModelIndex())
bool moveRows(const QModelIndex &sourceParent, int sourceRow, int count,
const QModelIndex &destinationParent, int destinationChild)
bool insertColumns(int column, int count, const QModelIndex &parent = QModelIndex())
bool removeColumns(int column, int count, const QModelIndex &parent = QModelIndex())
bool moveColumns(const QModelIndex &sourceParent, int sourceColumn, int count,
const QModelIndex &destinationParent, int destinationChild)
```

Nagłówki

```
QVariant headerData(int section, Qt::Orientation orientation, int role = Qt::DisplayRole) const
bool setHeaderData(int section, Qt::Orientation orientation,
const QVariant &value, int role = Qt::EditRole)
```

- Model komunikuje się ze źródłem danych, zapewniając interfejs dla innych komponentów architektury. Charakter komunikacji zależy od rodzaju źródła danych i sposobu implementacji modelu.
- Widok pobiera indeksy modelu z modelu; są to odniesienia do elementów danych. Dostarczając indeksy modelu do modelu, widok może pobierać elementy danych ze źródła danych.
- W widokach standardowych delegate renderuje elementy danych. Gdy element jest edytowany, delegat komunikuje się z modelem bezpośrednio przy użyciu indeksów modelu.



```
model = new QStringListModel(this);
model->setStringList(cities);
listView = new QListView;
listView->setModel(model);
listView->setEditTriggers(QAbstractItemView::AnyKeyPressed
                          | QAbstractItemView::DoubleClicked);
...
int row = listView->currentIndex().row();
model->insertRows(row, 1);
QModelIndex index = model->index(row);
listView->setCurrentIndex(index);
listView->edit(index);
```

- klasa `ModelIndex` służy do lokalizacji elementu, można go pobrać tylko z obiektu modelu
- Każdy widok ma funkcję `model()`, umożliwiającą pobranie modelu powiązanego z widokiem

atrybuty - widok pyta o nie, gdy chce coś wyświetlić

Qt ::DisplayRole - zwraca co wyświetlić
Qt ::EditRole -
Qt ::ToolTipRole -
Qt ::StatusTipRole -
Qt ::WhatsThisRole -
Qt ::FontRole -
Qt ::TextAlignmentRole -
Qt ::TextColorRole -
Qt ::BackgroundColorRole -

rowCount - liczba wierszy *

columnCount - liczba kolumn *

data - zestaw danych opisujących wyświetlanie wartości dla indeksu *

setData - modyfikację danych dla indeksu

headerData - dodatkowe informacje do wyświetlenia

flags - zarządza zachowaniem (np. edytowalnością)

```
int rowCount(const QModelIndex &parent) const;
int columnCount(const QModelIndex &parent) const;
QVariant data(const QModelIndex &index, int role) const;
bool setData(const QModelIndex &index, const QVariant &value,int role);
QVariant headerData(int section, Qt::Orientation orientation,
                    int role) const;
Qt::ItemFlags flags(const QModelIndex &index) const;
```

```
class CityModel : public QAbstractTableModel {
    Q_OBJECT
public:
  CityModel(QObject *parent = 0);
  void setCities(const QStringList &cityNames);

  int rowCount(const QModelIndex &parent) const;
  int columnCount(const QModelIndex &parent) const;
  QVariant data(const QModelIndex &index, int role) const;
  bool setData(const QModelIndex &index,const QVariant &value,int role);
  QVariant headerData(int section, Qt::Orientation orientation,
                int role) const;
  Qt::ItemFlags flags(const QModelIndex &index) const;
private:
    int offsetOf(int row, int column) const;
    QStringList cities;
    QVector<int> distances;
};
```

```
int CityModel::rowCount(const QModelIndex & /* parent */) const {
    return cities.count();
}
int CityModel::columnCount(const QModelIndex & /* parent */) const {
    return cities.count();
}
QVariant CityModel::data(const QModelIndex &index, int role) const
{
    if (!index.isValid())
        return QVariant();
    if (role == Qt::TextAlignmentRole) {
        return int(Qt::AlignRight | Qt::AlignVCenter);
    } else if (role == Qt::DisplayRole) {
        if (index.row() == index.column())
            return 0;
        int offset = offsetOf(index.row(), index.column());
        return distances[offset];
    }
    return QVariant();
}
QVariant CityModel::headerData(int section,
                               Qt::Orientation /* orientation */,
                               int role) const
{
    if (role == Qt::DisplayRole)
        return cities[section];
    return QVariant();
}
bool CityModel::setData(const QModelIndex &index,
                        const QVariant &value, int role)
{
    if (index.isValid() && index.row() != index.column()
            && role == Qt::EditRole) {
        int offset = offsetOf(index.row(), index.column());
        distances[offset] = value.toInt();
        QModelIndex transposedIndex = createIndex(index.column(),
                                                  index.row());
        emit dataChanged(index, index);
        emit dataChanged(transposedIndex, transposedIndex);
        return true;
```

```
    }
    return false;
}
Qt::ItemFlags CityModel::flags(const QModelIndex &index) const
{
    Qt::ItemFlags flags = QAbstractItemModel::flags(index);
    if (index.row() != index.column())
        flags |= Qt::ItemIsEditable;
    return flags;
}
```