

Програмування GUI

На основі мови C++ та фреймворку Qt

Лекція 1

Qt Development Frameworks (також відома як Qt Software, Trolltech та Quasar Technologies)



- Заснована в 1994 році



Eirik Chambe-Eng Haavard Nord

- офіси:
 - Осло, Норвегія (HQ)
 - Санта-Клара, Калифорнія
 - Брисбен, Австралія
- Основна продукція: Qt

Платформи



- Windows[®] 95 до Windows 10
- Mac OS[®] X
- Linux
- AIX, BSD / OS, FreeBSD, HP-UX, IRIX, NetBSD, OpenBSD, Solaris, Tru64 UNIX
- Та інші



Qt клієнти

- 
- Adobe, Agilent, Amazon, ARM, Boeing, Bosch, BMW, Cadence, Canon, CEA Technologies, ChevronTexaco, DaimlerChrysler, Deutsche Telekom, Disney, ESA, Fraunhofer, HP, IBM, Intel, Lockheed Martin, LogicaCMG, NASA, NEC, NTT, PGS, Pioneer, Rohde & Schwarz, Parallels, Pioneer, Philips, Oracle, HP, Goober, Google, Mercedes, Neonway, Rakuten, Samsung, Siemens, Sony, SUN, Tesla, Xerox, Xilinx, Yamaha
- 
- 
- 



Qt програми



- + Робочий стіл KOE Software Compilation 4, який використовується в Linux і Free8SD;

- + Редактор тривимірної графіки Autodesk Maya;

- + Додаток Viber (www.viber.com) компанії Rakuten;

- + Месенджер Telegram;

- + Програма Adobe Photoshop Album;

- + Мережева карта світу Google Earth;

- + Програма для віртуалізації операційних систем VirtualBox;

- + Вільний програвач VLC media player;

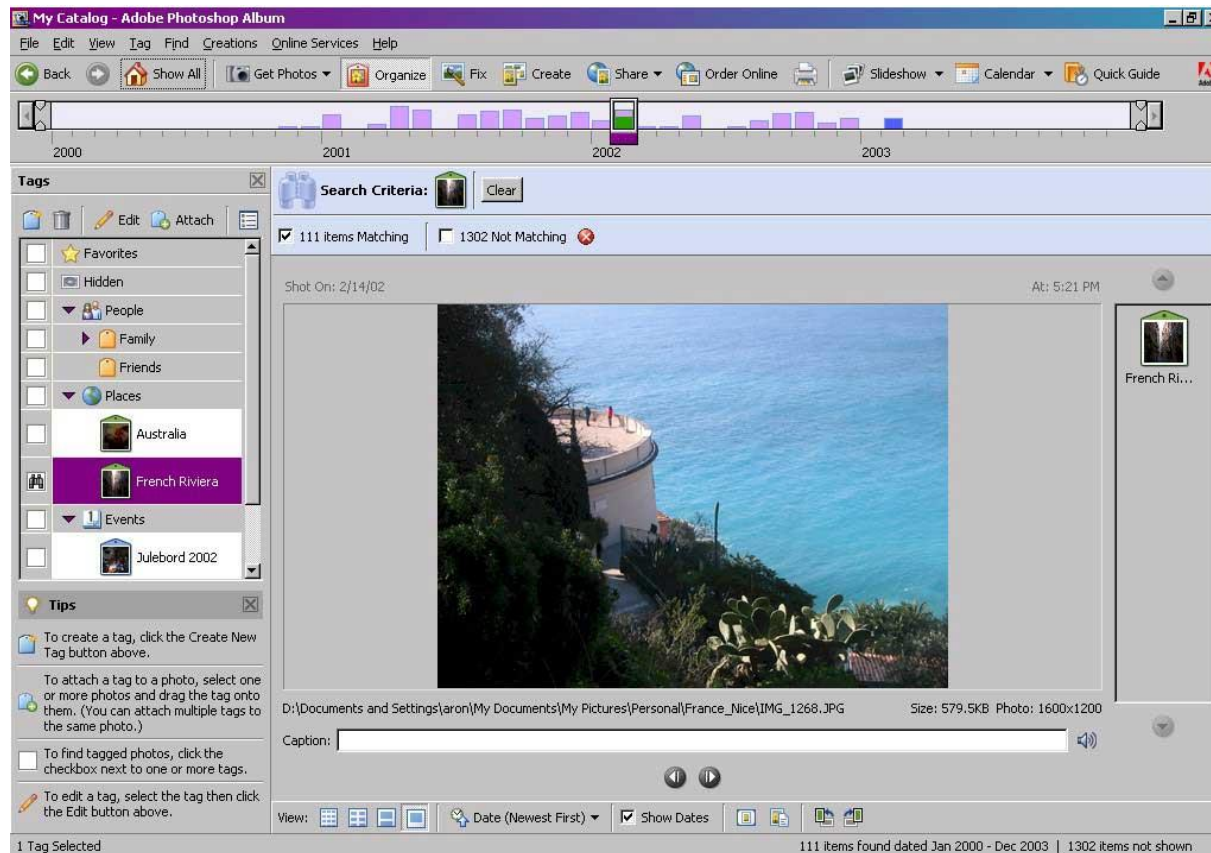
- + Програма для віртуалізації операційних систем Parallels;

- + Програма Kindle від компанії Amazon;

- + Програми офіційних клієнтів віртуальних валют Bitcoin і Litecoin



Приклад програми: Adobe Photoshop Album



Qt можливості



- підтримка двох-і тривимірної графіки, а також має свою власну альтернативну реалізацію і власний модуль Qt 3D;



- можливість інтернаціоналізації, яка дозволяє значно розширити ринок збуту ваших програм;

- використання форматів JSON (JavaScript Object Notation) і XML (eXtensible Markup Language);

- STL-сумісна бібліотеку контейнерів;



- підтримка стандартних протоколів введення / виведення;

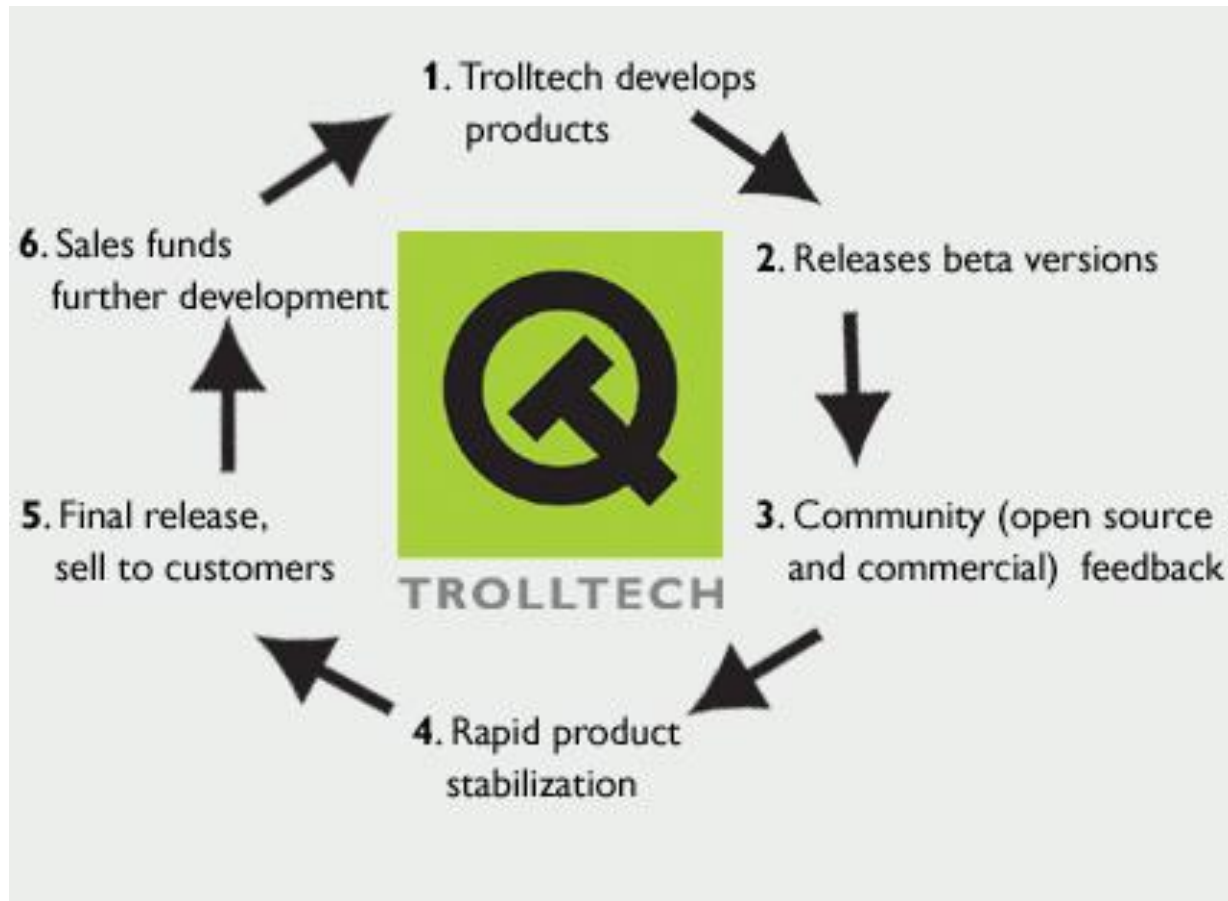
- класи для роботи з мережею;

- підтримка програмування баз даних, включаючи Oracle, Microsoft SQL Server, IBM DB2, MySQL, SQLite, Sybase, PostgreSQL;



- і багато іншого.







Ліцензування Qt



- Починаючи з версії 4.5

GPL

- Програма повинна бути відкрита, вільно поширюватися, вихідні тексти програми і всі зміни в початкових текстах Qt повинні перебувати у вільному доступі.



LGPL

- Вихідні тексти програми можуть бути як відкритими так і закритими. У разі, якщо програма є закритою і планується комерційне використання програми - Qt повинен зв'язуватися з програмою у вигляді динамічних бібліотек.



Commercial

- У разі комерційної ліцензії, крім можливості закривати, модифікувати будь-яким чином текст програми, модифікувати або закривати зміни в коді Qt і довільно вибирати ліцензію і спосіб поширення програми, надається також підтримка і консультації по використанню Qt.



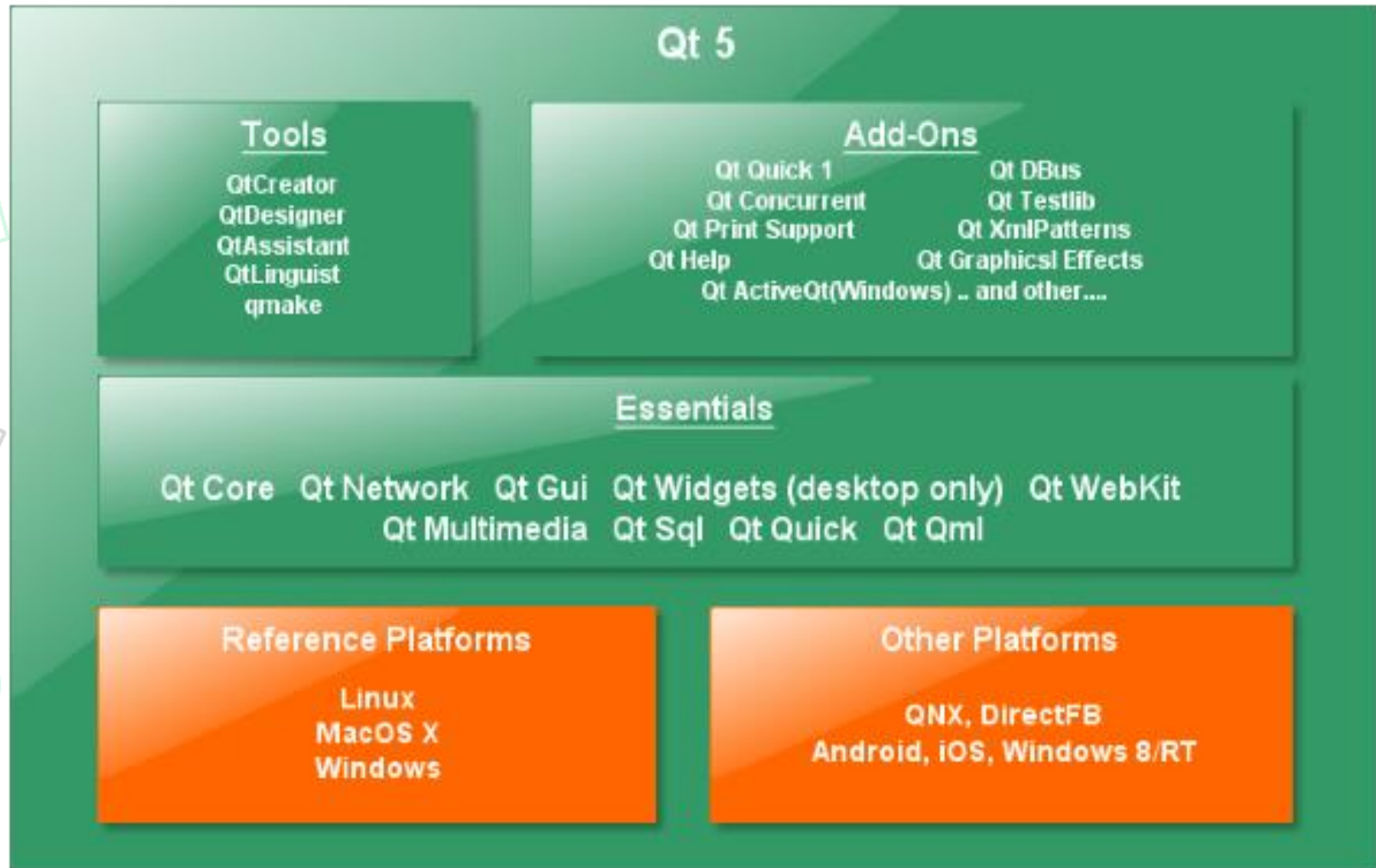
Ресурси, статті, навчальні відео







- Qt Project (<http://qt-project.org/>) - головний сайт вільного інструментарія розробки Qt;
- Qt Digia (<http://qt.digia.com/>) - офіційний сайт комерційної версії Qt;
- Planet Qt (<https://planet.qt.io/>) - сайт, який зібрав десятки блогів, присвячених Qt;
- Qt Centre (<http://www.qtcentre.org/>) - форум присвячений питанням розробки;



Основні складові Qt



Основні (Essentials) модулі

- 
- Qt Core - основний модуль;
 - Qt Network - модуль для роботи з мережевими засобами;
 - Qt Gui - модуль підтримки графічного виведення на екран;
 - Qt Widgets - модуль, який містить набір віджетів для створення графічного інтерфейсу користувача (Qt5);
 - Qt WebKit - засоби роботи з Веб;
 - Qt WebKit Widgets - віджети для роботи з Веб (Qt5);
 - Qt Multimedia - засоби роботи з мультимедійними пристроями і файлами;
 - Qt Multimedia Widgets - віджети для роботи з мультимедійними пристроями і файлами (Qt5);
 - Qt SQL - засоби роботи з базами даних;
 - Qt QML - підтримка декларативного мови QML для розробки динамічних візуальних інтерфейсів (Qt5);
 - Qt Quick - підтримка створення динамічних візуальних інтерфейсів (Qt5);
 - Qt Quick Controls - використання технології QtQuick для створення традиційного для робочих столів графічного інтерфейсу (Qt5);
 - Qt Quick Layouts - компоновка для елементів QtQuick (Qt5).
- 
- 
- 



Модуль Qt Core




Є базовим для додатків і не містить класів, що відносяться до інтерфейсу користувача.

У модуль QtCore входять понад 200 класів, ось деякі з них:



- + Контейнерні класи: QList, QVector, QMap, QVariant, QString і т. д. ;
- + Класи для введення і виведення: QIODevice, QTextStream, QFile;
- + Класи процесу QProcess і для програмування многопоточності: QThread, QWaitCondition, QMutex;
- + Класи для роботи з таймером: QBasicTimer і QTimer;
- + Класи для роботи з датою і часом: QDate і QTime;
- + Клас QObject, що є наріжним каменем об'єктної моделі Qt;
- + Базовий клас подій QEvent;
- + Клас для збереження налаштувань програми QSettings;
- + Клас додатки QApplication, з об'єкта якого, якщо потрібно, можна запустити цикл подій;
- + Класи підтримки анімації: QAbstractAnimation, QVariantAnimation і т. д. ;
- + Класи для машини станів: QStateMachine, QState і т. д. ;
- + Класи моделей інтерв'ю: QAbstractItemModel, QStringListModel, QAbstractProxyModel




Модуль Qt GUI



Цей модуль надає класи інтеграції з віконною системою, з OpenGL і OpenGL ES. Він містить клас QWindow, який є елементарною областю з можливістю отримання подій користувальницького введення, зміни фокусу і розмірів, а так само дозволяє виробляти графічні операції і малювання на своїй поверхні. Клас додатка цього модуля - QApplication. Він містить механізм циклу подій і володіє так само можливостями:



- + Отримання доступу до буфера обміну;
 - + Ініціалізації необхідних налаштувань програми - наприклад, палітри для забарвлення елементів управління;
 - + Управління формою курсору миші.
- 

Модуль QtWidgets



Цей модуль містить близько 300 класів віджетів, що представляють собою «будівельний матеріал» для програмування графічного інтерфейсу користувача:

- + Клас QWidget - це базовий клас для всіх елементів управління бібліотеки Qt.
- + Класи для автоматичного розміщення елементів: QVBoxLayout, QHBoxLayout;
- + Класи елементів відображення: QLabel, QLCDNumber;
- + Класи кнопок: QPushButton, QCheckBox, QRadioButton;
- + Класи елементів установок: QSlider, QScrollBar;
- + Класи елементів введення: QLineEdit, QSpinBox;
- + Класи елементів вибору: QComboBox, QToolBox;
- + Класи меню: QMainWindow і Qmenu;
- + Класи вікон повідомлень та діалогових вікон: QMessageBox, QDialog;
- + Класи для малювання: QPainter, QBrush, QPen, QColor;
- + Класи для растрових зображень: QImage, QPixmap;
- + Класи стилів;
- + Клас додатки QApplication, який надає цикл подій.



Модули QtQuick і QtQML



Це альтернатива віджетів - модулі, що представляють собою набір технологій для швидкої розробки графічних інтерфейсів нового покоління на базі описового мови QtQML, мови програмування JavaScript і всіх інших можливостей бібліотеки Qt



Модуль QtNetwork

Модуль QtNetwork надає інструментарій для програмування TCP- і UDP-сокетів (класи QTcpSocket і QUdpSocket), а також для реалізації програм-клієнтів, які використовують HTTP- і FTP-протоколи (клас QNetworkAccessManager).



Модуль QtSQL



Цей модуль призначений для роботи з базами даних. У нього входять класи, що надають можливість для маніпулювання значеннями баз даних



Модули QtMultimedia и QtMultimediaWidgets

Модуль QtMultimedia володіє всім необхідним для створення додатків з підтримкою мультимедіа. Він підтримує як низький рівень, необхідний для більш детальної спеціалізованої реалізації, так і високий рівень, який робить можливим програвати відео-та звукові файли за допомогою всього декількох рядків програмного коду. Модуль QtMultimediaWidgets містить корисні елементи у вигляді віджетів, які дозволяють економити час для реалізації.

Модуль QtSvg

Модуль підтримки графічного векторного формату SVG, що базується на XML. Цей формат надає можливість не тільки для виведення одного кадру векторного зображення, але може бути використаний і для векторної анімації

Додаткові модулі Qt



- QtWebEngineCore Дозволяє дуже просто інтегрувати в додаток можливості веб
- QtWebEngineWidgets Надає готові до інтеграції в додаток елементи у вигляді віджетів з можливістю також розширювати елементи веб своїми власними віджетами
- Qt 3D 3dcore Являє собою цілу колекцію з 7 модулів: Qt3DAnimation, Qt3OCore, 3dextras, Qt3DExtras, Qt3Dinput, Qt3DLogic, 3danimation, Qt3DRender і Qt3DScene2D. Мета цих модулів - надати механізми для спрощення програмування тривимірної графіки
- QtBluetooth Містить класи для використання бездротової технології Bluetooth
- QtLocation Надає класи геолокації для опрелеленія поточного місцезнаходження
- QtSensors Забезпечує доступ до сенсорам мобільних пристроїв - таким, як, наприклад, сенсор орієнтації і акселерометр.



Перша програма на Qt



```
#include <QtWidgets> //заголовковий файлQtWidgets,  
// або
```



```
//#include <QLabel>  
//#include <QApplication>
```

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



```
{
```

```
    QApplication a(argc, argv); //створення об'єкта класу  
    //QApplication
```




```
    QLabel lbl("Hello world!"); // створення об'єкта класу QLabel  
    lbl.show(); // відображення написи на екрані  
    return a.exec(); //виклик метода exec()
```



```
}
```




Сигнали і слоти





В Qt під сигналами мають на увазі методи, які можуть висилати повідомлення з інформацією про подію.



Методи сигналів можуть мати параметри, але не можуть повертати значення.



Клас, що має свої сигнали і \ або слоти, повинна успадкувати клас `QObject` і використовувати макрос `Q_OBJECT`.



Приклад



```
class A: public QObject
```

```
{  
    Q_OBJECT // для використання сигналів та слотів
```

```
...
```

```
signals: void MySignal(); // сигнал з ім'ям MySignal
```

```
...
```



```
public:  
void sendSignal()
```

```
{emit MySignal(); } // висилає сигнал
```


```
MySignal A() {}
```

```
...
```

```
};
```




Сигнали і слоти




Висилати сигнал можна за допомогою ключового слова emit:



```
emit ім'я_сигнала
```

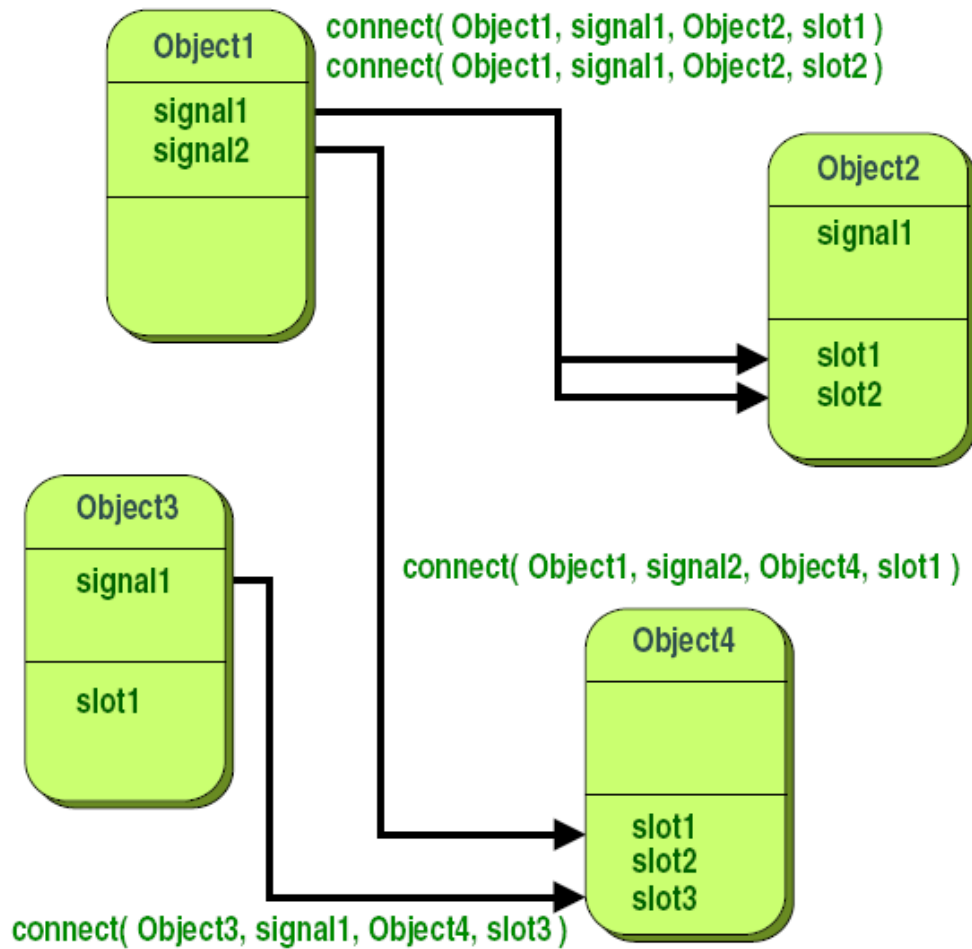


Слоти - методи, які приймають і обробляють сигнали. Слоти оголошуються в одній із секцій:
private slots, protected slots, public slots.



У слотах можна використовувати параметри за замовчуванням, слоти не можуть бути статичними.

Сигнали і слоти

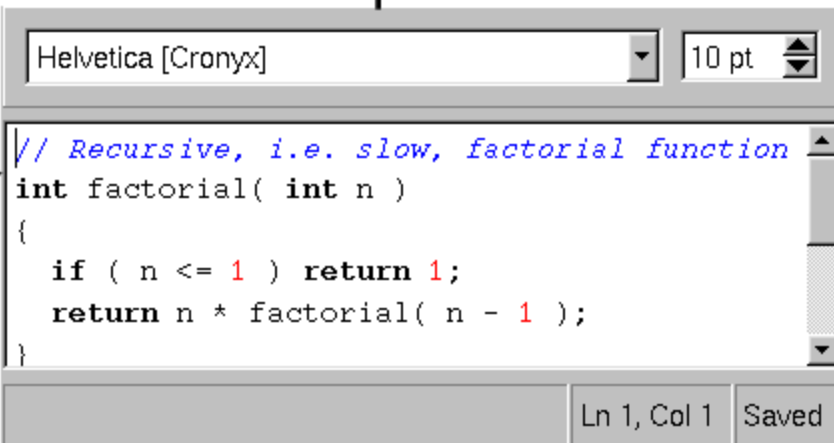


Сигнали і слоти

`connect(fontFamilyComboBox, activated(QString),
textEdit, setFamily(QString))`

`connect(fontSizeSpinBox, valueChanged(int),
textEdit, setPointSize(int))`

`connect(textEdit, modificationChanged(bool),
customStatusBar, modificationStatus(bool))`



```
// Recursive, i.e. slow, factorial function
int factorial( int n )
{
    if ( n <= 1 ) return 1;
    return n * factorial( n - 1 );
}
```

Ln 1, Col 1 Saved

Сигнали і слоти



```
class B: public QObject
{ Q_OBJECT
...
public:
    B(){}
...
public slots:
    void slotB()
    {cout<<"Hello from B";}
...
};
```



З'єднання слотів і сигналів



З'єднання слотів і сигналів реалізується за допомогою методу `connect ()` класу `QObject`



```
QObject :: connect (const QObject * sender, const char * signal,  
const QObject * receiver, const char * slot);
```

`sender` - покажчик на об'єкт, що висилає сигнал

`signal` – сигнал, що висилається

`receiver` - покажчик на об'єкт, що має слот, який приймає і обробляє сигнал

`slot` - слот, що обробляє сигнал.



Сигнатура сигналу (ім'я, список типів параметрів) повинна бути поміщена в спеціальний макрос `SIGNAL (...)`. Сигнатура слота повинна бути поміщена в спеціальний макрос `SLOT (...)`.



Сигнали і слоти



```
A *pA=new A();
```

```
B *pB=new B();
```

```
QObject::connect(pA,SIGNAL(MySignal()),pB, SLOT(slotB()));
```



Роз'єднання сигналів і слотів об'єктів здійснюється за допомогою методу `disconnect()`:

```
QObject::disconnect(const QObject *sender, const char *signal,  
const QObject *receiver, const char *slot);
```



Пример

Сигнально-слотове з'єднання між кнопкою QPushButton і віджетом-вікном QWidget.

У відповідь на натискання кнопки (сигнал clicked ()), викликається метод-слот close (), який закриває вікно.

```
connect (lPushButton, SIGNAL(clicked()),  
lWindow, SLOT(close()));
```

Тип сигнально-слотового з'єднання

- Qt :: AutoConnection _ тип з'єднання за замовчуванням. При з'єднанні об'єктів в межах потоку поводитьься як Direct Connection, інакше як Queued Connection;
- Qt :: DirectConnection _ слот викликається негайно після того, як був випущений сигнал. По суті це нагадує звичайний виклик слота як методу;
- Qt :: QueuedConnection _ слот виконується, як тільки управління перейде до черги обробки повідомлень потоку одержувача;
- Qt :: BlockingQueuedConnection _ той же, що і Queued Connection, але потік, з якого був випущений сигнал блокується, поки виконання слота не буде завершена. Цей тип з'єднання повинен використовуватися тільки коли взаємодіючі об'єкти знаходяться в різних потоках.
- Qt :: UniqueConnection тип з'єднання такий же як і Qt :: AutoConnection, але з'єднання відбувається тільки тоді, коли воно унікальне (тобто таке, що не дублює вже існуючі з'єднань)

Ієрархія об'єктів в Qt

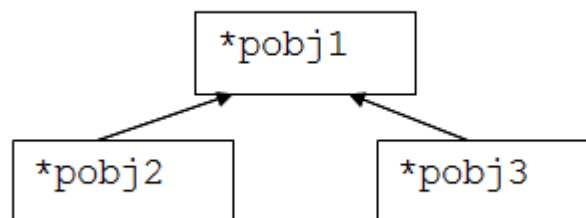


Конструктор класу QObject:

`QObject (QObject * parent),`
де `parent` - покажчик на об'єкт-предок (батьківський об'єкт).

Якщо `parent = 0` - створюваний об'єкт не має предків і є об'єктом верхнього рівня.

`QObject *pobj1=new QObject; //pobj1 - покажчик на об'єкт-предок`
`QObject *pobj2=new QObject(pobj1);`
`QObject *pobj3=new QObject(pobj1);`



`delete pobj1; // при видаленні * pobj1 автоматично видаляються * pobj2,`
`// pobj3`

