

INTRODUZIONE ALLE Qt

Salve, sono Matrix86.

Oggi vedremo le Qt, le librerie grafiche su cui si basa KDE. Naturalmente quello che affronteremo in questo tutorial è solamente una introduzione. Inizieremo con un semplice e classico HELLO WORLD.

Vediamo di iniziare...

Allora di seguito troverete il codice della applicazione che andiamo a creare, dopodiché passeremo ad analizzare ogni singolo pezzo.

Su Linux creiamo una cartella, e mettiamo il sorgente nel file main.cpp, fatto questo date i seguenti comandi per compilare:

```
$ qmake -project
```

```
$ qmake
```

```
$ make
```

Se non ci sono errori avrete la vostra applicazione funzionante!

Ecco il sorgente:

----- CUT HERE -----

```
#include <qapplication.h>
```

```
#include <qlabel.h>
```

```
#include <qpushbutton.h>
```

```
#include <qfont.h>
```

```
#include <qvbox.h>
```

```
class MyWidget : public QWidget
```

```
{
```

```
public:
```

```
    MyWidget( QWidget *parent=0, const char *name=0 );
```

```
};
```

```
MyWidget::MyWidget( QWidget *parent, const char *name )
```

```
    : QWidget( parent, name )
```

```
{
```

```
    setMinimumSize( 220, 100 );
```

```
    setMaximumSize( 220, 100 );
```

```
    QLabel *label = new QLabel("La mia prima applicazione Qt",this,"label1");
```

```
    label->adjustSize();
```

```
    label->move(25,25);
```

```
    QPushButton *quit = new QPushButton("Exit",this,"uscita");
```

```
    quit->move(50,50);
```

```
    quit->setFont( QFont( "Arial", 13, QFont::Bold ) );
```

```
connect( quit,SIGNAL(clicked()), qApp, SLOT(quit()));
}

int main(int argc, char *argv[])
{
    QApplication app(argc,argv);

    MyWidget w;
    app.setMainWidget( &w );
    w.show();
    app.exec();

    printf("Il controllo è tornato al main() =) \n");
    return 0;
}
```

----- CUT HERE -----

Passiamo ad analizzare il sorgente. Allora inizialmente come in tutti i sorgenti troveremo i soliti include:

```
#include <qapplication.h>
#include <qlabel.h>
#include <qpushbutton.h>
#include <qfont.h>
```

In questo modo includiamo le classi:

- QApplication, ovvero l'oggetto usato in ogni applicazione Qt, questa amministra alcune risorse come per esempio i font, il puntatore usato di default ecc;
- QLabel che definisce l'oggetto etichetta
- QPushButton che definisce il classico pulsante;
- QFont che permette la gestione dei font.

```
class MyWidget : public QWidget
{
public:
    MyWidget( QWidget *parent=0, const char *name=0 );
};
```

In questo modo creiamo una nuova classe che eredita da QWidget e definisce il suo costruttore (NB. La classe QWidget non richiede distruttori, quindi nemmeno la nostra MyWidget lo richiederà). Questa classe sarà un nuovo widget che noi useremo come base su cui aggiungere oggetti.

```
MyWidget::MyWidget( QWidget *parent, const char *name )
    : QWidget( parent, name )
{
```

Implementiamo effettivamente il costruttore della classe precedente.

```
setMinimumSize( 220, 100 );  
setMaximumSize( 220, 100 );
```

Con queste 2 funzioni definiamo quindi la grandezza minima e massima della finestra. Notiamo che abbiamo inserito le stesse misure, quindi questo farà sì che la finestra abbia dimensione fissa!

```
QLabel *label = new QLabel("La mia prima applicazione Qt",this,"label1");
```

Creiamo l'oggetto QLabel ovvero l'etichetta. Al suo costruttore passiamo il testo che deve contenere, il contenitore che deve contenerla (in questo caso la nostra classe padre) e il suo nome.

```
label->adjustSize();  
label->move(25,25);
```

Con queste due funzioni possiamo far sì che la grandezza della label sia definita automaticamente, e definire la sua posizione all'interno della finestra.

```
QPushButton *quit = new QPushButton("Exit",this,"uscita");
```

Come sopra creiamo il pulsante e assegniamogli il contenuto "Exit", il contenitore e il nome.

```
quit->move(50,50);
```

Spostiamo il pulsante all'interno della finestra alle coordinate (x,y) = (50,50).

```
quit->setFont( QFont( "Arial", 13, QFont::Bold ) );
```

Settiamo il font da usare per la scritta Exit sul pulsante. In questo caso abbiamo definito che si debba usare il font Arial, 13 come grandezza del carattere e che questo sia grassetto.

```
connect( quit,SIGNAL(clicked()), qApp, SLOT(quit()));
```

Arriviamo infine alla parte un po' più complessa, ma affascinante delle Qt. Naturalmente abbiamo creato la grafica, ma se premiamo il pulsante Exit, non avremo nessuna risposta. Questo perché non abbiamo definito l'operazione da eseguire quando il pulsante viene premuto.

Ogni oggetto nelle Qt ha diversi segnali per avvisare l'evento che è stato scatenato su quell'oggetto (esempio clicked() su QPushButton per il click sul pulsante) e degli SLOT per la ricezione di tale evento che eseguono una funzione.

Quindi nel nostro caso l'evento clicked() è stato associato al metodo quit(). Ovvero quando si clicca sul pulsante si esce dal programma. I segnali e gli slot sono un argomento un po' ostico che affronteremo magari più avanti nel dettaglio in un altro tutorial (se mi va e se ho tempo ;p), se volete potete naturalmente vederli da soli nei doc delle Qt (se avete installato le Qt le avrete nel sistema, per esempio sulla mia Slackware sono visualizzabili su browser in /usr/lib/qt/doc/html/).

Definito quindi il costruttore passiamo alla funzione main che andrà a creare effettivamente la nostra finestra.

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

Il classico Main, il punto di ingresso nel programma, l'entry point.

```
    QApplication app(argc,argv);
```

Qapplication è il programma, è la classe che amministra la GUI. Come potete vedere gli passiamo i valori argc e argv in modo che questo possa processare anche alcune opzioni passate da linea di comando, per esempio il parametro -display per la scelta del display X o -geometry per cambiare la grandezza della finestra.

```
    MyWidget w;
```

Istanziamo il widget che abbiamo creato prima contenente la finestra vera e propria.

```
    app.setMainWidget( &w );
```

Indichiamo quale debba essere la finestra principale della nostra applicazione. In questo caso la widget che prima abbiamo istanziato.

```
    w.show();
```

Rendiamo visibile la nostra widget (e anche tutte le widget contenute in esso). Ricordate che tutti gli oggetti che create sono di default INVISIBILI, quindi devono essere resi visibili con la funzione show().

```
    app.exec();
```

```
    printf("Il controllo è tornato al main() =) \n");  
    return 0;
```

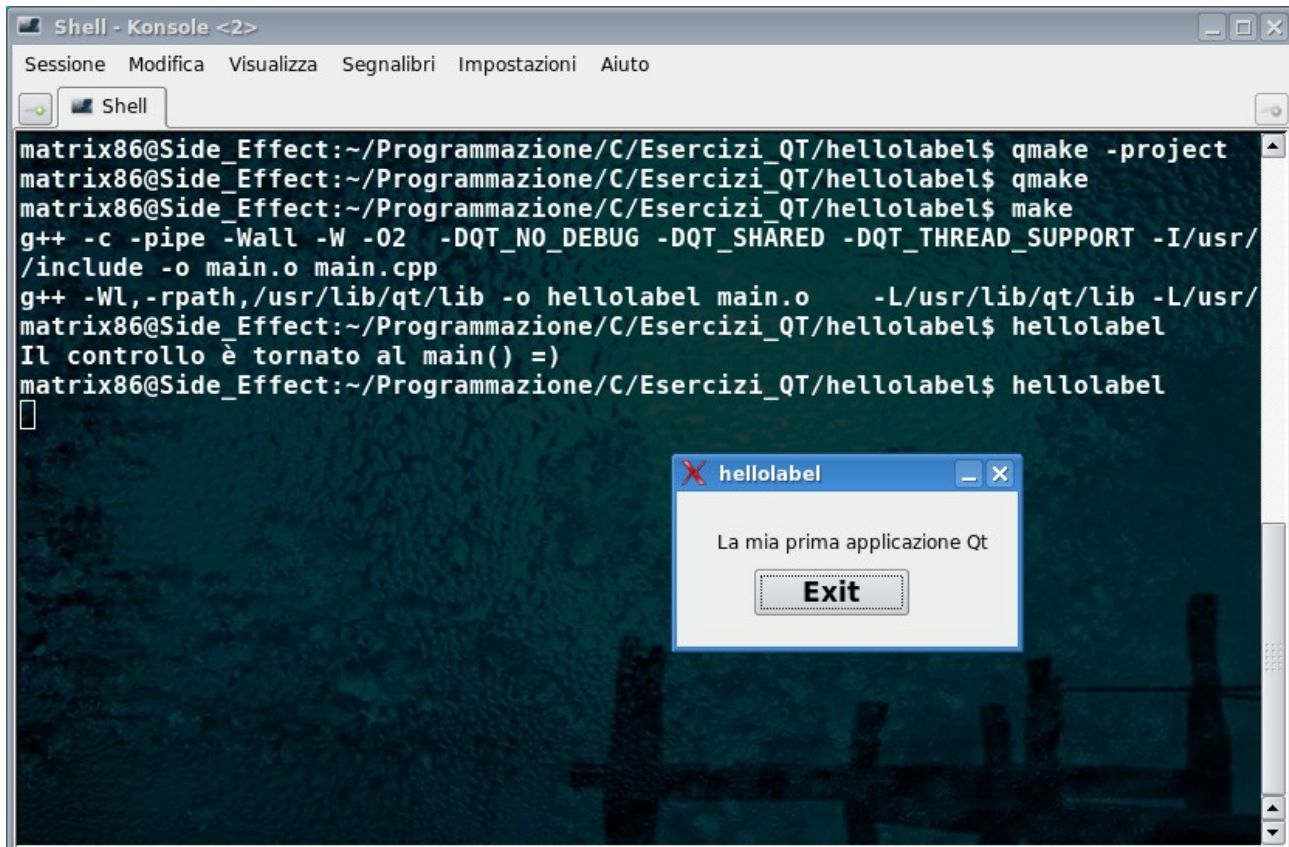
Passiamo il controllo dal main alle Qt. Questo ritornerà quando si uscirà dall'applicazione. In exec() le Qt processeranno gli eventi utente e di sistema. Al ritorno al main() possiamo vedere come verrà stampata a video la nostra stringa e infine si uscirà dal programma.

Bene eccoci giunti alla fine del nostro viaggio introduttivo sul vasto mondo delle Qt. Spero che questo tutorial vi sia di aiuto. Come ho detto all'inizio, questo è veramente una piccola introduzione. Se siete bravi in inglese date un'occhiata al sito ufficiale o ai doc sulle Qt nel vostro sistema. Ho preso spunto (molto) da lì per la stesura di questo doc. Quindi non c'è nulla di meglio di quello. Un saluto da Matrix86.

Per insulti, chiarimenti ed altro matrix86@rbt-4.net

Fonti:

- <http://doc.trolltech.com/>



```
Shell - Konsole <2>
Sessione Modifica Visualizza Segnalibri Impostazioni Aiuto
Shell
matrix86@Side_Effect:~/Programmazione/C/Esercizi_QT/hellolabel$ qmake -project
matrix86@Side_Effect:~/Programmazione/C/Esercizi_QT/hellolabel$ qmake
matrix86@Side_Effect:~/Programmazione/C/Esercizi_QT/hellolabel$ make
g++ -c -pipe -Wall -W -O2 -DQT_NO_DEBUG -DQT_SHARED -DQT_THREAD_SUPPORT -I/usr/
/include -o main.o main.cpp
g++ -WL,-rpath,/usr/lib/qt/lib -o hellolabel main.o -L/usr/lib/qt/lib -L/usr/
matrix86@Side_Effect:~/Programmazione/C/Esercizi_QT/hellolabel$ hellolabel
Il controllo è tornato al main() =)
matrix86@Side_Effect:~/Programmazione/C/Esercizi_QT/hellolabel$ hellolabel

```

The screenshot shows a terminal window with a dark background. The terminal output shows the user running 'qmake -project', 'qmake', and 'make' to compile the application. The compilation is successful, and the user then runs 'hellolabel' twice. The first run displays the message 'Il controllo è tornato al main() =)'. The second run opens a small window titled 'hellolabel' with the text 'La mia prima applicazione Qt' and an 'Exit' button.