¡Al grano con Qt!

Table of Contents

1. Introduccion 1
2. Un dialogo Qt de verdad 2

Ultima revision: 5 de noviembre de 2009

(descargar version PDF (cuant puga))

1. Introduccion

Creo que Qt es el mejor conjunto de herramientas para C++ en el desarrollo de aplicaciones multiplataforma.

Aunque empezo como herramienta para el diseño de GUI, actualemnte va mucho mas alla, permitiendo la aplicacion de las tecnologias mas productivas (SQL, XML, SVG, OpenGL, WebKIT, multi-hilo, etc.) de una manera sencilla e intuitiva.

(el rollete de abajo lo movere mas adelante a otra entrada)

Adquirida Trolltech por parte de Nokia, esta herramienta pasa al gigante de la telefonia movil con el prosito, mas o menos declarado, de recuperar el camino perdido respecto a los nuevos competidores (Google Android, Palm WebOS, Iphone) y los no tan nuevos (Microsoft Windows Mobile y Blackberry RIM) que se van haciendo poco a poco con el mercado de los moviles mas poderosos. Vamos, Nokia se durmio en los laureles y vive del cuento de momento.

Antes de continuar, aclarar que tanto Android como Palm WebOS como otros potentes moviles (los Samsung Omnia, LG Viewty, etc.) son pequeñas capas de midleware encima del sistema operativo GNU/Linux. Es decir, todo el merito de comerse el mercado de Windows y Symbian esta en las prestaciones que ofrece GNU/Linux y no al contrario. En este campo tambien ha competido Qt contra Nokia en los moviles de Samsung, LG y marcas para el mercado chino. El Iphone es tambien un *nix basado en llamdo FreeBSD Unix (que tio el Steve Jobs ese! vaya cabeza!).

La nueva e importante estrategia de Nokia ha sido cambiar los modelos de licencia de Qt. Antes se tenian dos opciones, o usabas la version Open y estabas obligado a que el software resultante fuese Open, o usabas la version comercial (pagando) y podias vender el resultado. Fruto destacado del primer caso es KDE, el mejor software de escritorio hasta hace poco, y fruto del segundo caso estan, por ejemplo, Google Earth y Skype.

En el nuevo esquema de licencias se ha añadido la licencia GNU LGPL (Lesser General Public License) que permite utilizar la version Open para producir codigo cerrado y vendible. Este importante cambio ha hecho que me decida por Qt para docencia y reserve, otros caminos como GTK y WXWidgets a otros propositos.

Volviendo a Nokia, ¿como recupera el camino perdido?. Consiguiendo que Qt genere codigo para Symbian y, a la vez, captando el esfuerzo de la comunidad open, que es la que le ha hecho el trabajo a Google Android y a Palm WebOS. Lo primero esta claro como se consigue, y lo segundo, para quien no lo sepa, Nokia ha mantenido una linea paralela de dispositivos denominados Internet Tablets que funciona con Maemo, una variacion de Debian Linux. En cualquier caso, Nokia nunca se ha atrevido a que en sus Internet Tablets se pudiesen hacer llamadas de voz ... hasta ahora, que se ha visto con el agua al cuello, y ha lanzado el Nokia N900, que ya es todo un supermovil basado en un Linux abierto y digno heredero del frustado OpenMoko Freerunner.

En resumen, si consigo que los mejores programadores (no por buenos, si no por insistentes y numerosos) trabajen para mi en el N900, pues el software podra ir directamente a engrosar la lista de software para Symbian. Ademas, se podrian sacar unas pelillas poniendo sus programas al estilo Iphone en la tienda OVI.

XML to PDF by RenderX XEP XSL-FO Formatter, visit us at http://www.renderx.com/

Tened en cuenta que lo anterior es una opinion mia, y puedo estar equivocado pero, en general, la realidad es mas retorcida que estas elucubraciones. En todo caso, el Nokia N900 es mi movil perfecto porque lleva un GNU/Linux abierto que te permite poner en practica cualquier idea por imposible que parezca, y ya he sufrido bastante con Microsoft Windows en mi PC y en mi movil.

(fin rollete)

Respecto a introducirse en su uso ... libros hay varios (ver bibliografia), pero creo que se puede simplificar muchisimo su aprendizaje si se evitan los rodeos y malos enfoques que se ven en libros y tutoriales. El proposito de este tema/tutorial es intentar transmitir la filosofia correcta sobre Qt/C++ de una manera lo mas inmediata y limpia posible.

Esto no es un sustituto de un libro buen libro de Qt (recomiendo el Molkentin), es una inmersion a pulmon libre en Qt a mi manera. A ver si resulta ;-).

2. Un dialogo Qt de verdad

Vamoas alla proponiendo crear una nueva clase C++ para una ventana de dialogo.

Como siempre en C++, primero hay que definir la clase. Esta es la que se propone:

```
 1 // Fichero: tventanuco.h
   // Primer programa Qt para DSII 2009/2010

   #ifndef TVENTANUCO_H
 5 #define TVENTANUCO_H

   #include <QDialog>  // voy a crear un QDialog!

   // declaraciones "forward" de C/C++ para acelerar la compilación
10 class QPushButton;
   class QVBoxLayout;
   class QLabel;
   class QLineEdit;

15 // se podrían cambiar por lo siguiente, pero tardaria más en compilar
   //#include <QPushButton>
   //#include <QVBoxLayout>

   // definición de la nueva clase TVentanuco que vamos
20 // a derivar de QDialog
   class TVentanuco : public QDialog
   {
       Q_OBJECT    // <-- ponlo siempre que quiers que funcione signal/slot

25 public:
       TVentanuco(char *mensaje);   // el constructor

   private:
       // vamos a meter unas cuantas cosas
30     double valor;
       QVBoxLayout *lyDistribuidor;
       QLabel *lbEtiqueta;
       QLineEdit *leEdicion;
       QPushButton *btToquetear;
35     QPushButton *btCerrar;
   };

   #endif // TVENTANUCO_H
```

La nueva clase TVentanuco va a ser una derivacion de la clase QDialog de Qt que permite la construccion de ventanas de dialogo. Si se mira el manual de Qt, se vera que se hubiese podido derivarr de otras cosa, por ejemplo, un QWidget, un QMAinWindows, un QSpalsh, etc.

XML to PDF by RenderX XEP XSL-FO Formatter, visit us at http://www.renderx.com/

En la linea X tenemos una inclusion de la cabecera QDialog (sin el .h) que es necesaria, pues vamos a derivar de QDialog. En Qt, para simplificar el tener que recordar nombres de cosas, las cabeceras y los tipos de clase tiene el mismo nombre. Asi, si necesitamos un QPushButton, simplemente tendremos que hacer #incluide <QPushButton> en el archivo que lo declare o implemente.

Destacar en la linea 23 la inclusion de la etiqueta especial Q_OBJECT. De momento, supongamos que hay que ponerla siempre.

En la zona privada del objeto entre las lineas, x-y se declaran varios punteros a clases de Qt. En el maual de Qt se puede mirar para que sirven.

Entree las lines x-y nos encontramos con declaraciones forward de C/C++ que coinciden con las clases usadas en los punteros de la zona privada. Esto sirve para que el compilador admita la cabecera sin rechistar al encontrarse conllo que hay en la zona privada.

Ahora pasamos a la implementacion. Se propone el siguiente listado:

```
// Fichero: tventanuco.cpp
// Primer programa Qt. DSII curso 2009/2010

#include "tventanuco.h"

// hay que meter la cabecera de los objetos Qt que vayas a usar
#include <QPushButton>
#include <QVBoxLayout>
#include <QLabel>
#include <QLineEdit>

// constructor de la clase, se llama también al constructor del padre
TVentanuco::TVentanuco(char *mensaje):QDialog()
{
    // enviamos el mensaje al titulo del dialog
    setWindowTitle(mensaje);

    //crear el distribuidor de objetos
    lyDistribuidor = new QVBoxLayout(this);

    // creamos una etiqueta y la metemos en el "layout"
    lbEtiqueta = new QLabel(tr("A ver si esto va o no"));
    lyDistribuidor->addWidget(lbEtiqueta);

    // parar jugar con los "layout", meto un "muelle"
    lyDistribuidor->addStretch();

    // ahora una linea de edición
    leEdicion = new QLineEdit;
    leEdicion->setText(tr("Hola, Colaloca"));
    lyDistribuidor->addWidget(leEdicion);

    // ahora un boton
    btToquetear = new QPushButton(tr("Tocar narices"));
    lyDistribuidor->addWidget(btToquetear);

    // ahora el boton de cerrar
    btCerrar = new QPushButton(tr("Salir"));
    lyDistribuidor->addWidget(btCerrar);

    // vamos a conectar cosas
    connect(btCerrar,SIGNAL(clicked()),this,SLOT(close()));
    connect(btToquetear,SIGNAL(clicked()),leEdicion,SLOT(clear()));
}
```

Lo primero es incluir la cabecera de las clases de Qt que necesitemos.

Lo unico que hemos implementado es el constructor, que es bastante intuitivo. Simplemente se van creando dinamicamente los objetos y se van introduciendo en el distribuidor (Layout).

XML to PDF by RenderX XEP XSL-FO Formatter, visit us at http://www.renderx.com/

Todos los mensajes a mostrar en pantalla van dentro de la funcion tr() (en realidad QObject::tr()) que, a partir de ahora, recomiendo usar, pues sirve para facilitar el desarroloo de versiones multilingues.

XML to PDF by RenderX XEP XSL-FO Formatter, visit us at http://www.renderx.com/