Введение

Сегодня практически невозможно представить себе приложение, не обладающее интерфейсом пользователя. Понятия *Software* (программный продукт) и *GUI* (Graphical User Interface, графический интерфейс пользователя) неразрывно связаны друг с другом.

Хотя Windows API (Application Programming Interface, интерфейс программирования приложений) обладает всем необходимым для создания графического интерфейса пользователя, использование этих доступных «инструментов» требует больших затрат времени и практического опыта. Поэтому и сегодня разработчики по-прежнему тратят массу времени на реализацию интерфейса пользователя. Но самый большой недостаток, связанный с применением таких библиотек, — это платформозависимость.

Qt предоставляет поддержку большого числа операционных систем: Microsoft Windows, Mac OS X, Linux, Solaris, NetBSD, OpenBSD, HP-UX, FreeBSD и других клонов UNIX, а так же и для мобильных операционных систем iOS, Android, Windows Phone, Windows RT и BlackBerry. Более того, благодаря встраиваемому пакету Qt Embedded все возможности Qt доступны также и в интегрированных системах (Embedded Systems). Qt использует интерфейс API низкого уровня, что позволяет приложениям работать столь же эффективно, как и приложениям, разработанным специально для конкретной платформы.

Использование в разработке разных компиляторов C++ еще больше повышает правильность и надежность кода ваших программ, поскольку предупреждающие сообщения и сообщения об ошибках вы будете получать от разных компиляторов.

Для ускорения и упрощения создания пользовательских интерфейсов Qt предоставляет программу Qt Designer, позволяющую делать это в интерактивном режиме.

На сегодняшний день Qt — это продукт, широко используемый разработчиками всего мира. Компаний, ориентированных на эту библиотеку, более четырех тысяч. В число активных пользователей Qt входят такие компании, как: Adobe, Amazon, AMD, Bosch, Blackberry, Cannon, Cisco Systems, Disney, Intel, IBM, Panasonic, Philips, Oracle, HP, Goober, Google, NASA, Nokia, Samsung, Siemens, Sony, Xerox, и др.

Используя сегодня ту или иную программу, вы, возможно, даже и не догадываетесь, что при ее написании задействовалась библиотека Qt. Приведу лишь несколько, на мой взгляд, самых ярких примеров:

рабочий стол KDE Software Compilation 4 (www.kde.org), используемый в Linux и FreeBSD;

редактор трехмерной графики Autodesk Maya (www.autodesk.com);

Linux-версия Skype (www.skype.com);

программа Adobe Photoshop Album (www.adobe.com) для обработки растровых изображений;

сетевая карта мира Google Earth (earth.google.com), которая позволяет рассматривать интересующие нас участки поверхности нашей планеты с высоты до 200 м;

программа для виртуализации операционных систем VirtualBox (www.virtualbox.org) от Sun Microsystems;

свободный проигрыватель VLC media player (www.videolan.org/vlc/);

Qt— полностью объектно-ориентированная библиотека. Новая концепция ведения межобъектных коммуникаций, именуемая «сигналы и слоты», полностью заменяет былую, не вполне надежную модель обратных вызовов. Имеется также возможность обработки событий — например, нажатия клавиш клавиатуры, перемещения мыши и т. д.

Предоставляемая система расширений (plug-ins) позволяет создавать модули, расширяющие функциональные возможности создаваемых приложений. Эти расширения пользователи вашей программы могут получать не только от вас, но и от других разработчиков.

Первая программа на Qt

Как заведено, в самом начале знакомства нужно поздороваться, и, чтобы никого не оставить без внимания, лучше всего обратиться сразу ко всему миру. Давайте для этого напишем короткую программу «Hello, World» («Здравствуй, Мир»), результат выполнения которой показан на рис. 1.

Файл Hello.pro

```
TEMPLATE = app
QT       += widgets
SOURCES  = hello.cpp
```

```
windows:TARGET     = ../Hello
```

Файл hello.cpp:

```
#include <QtWidgets>
int main(int argc, char** argv)
{
   QApplication app(argc, argv);
   QLabel lbl("Hello, World !");
   lbl.show();
   return app.exec();
}
```

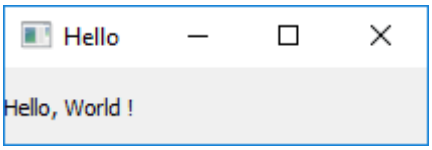


Рис. 1. Окно программы «Hello, World»

В первой строке подключается заголовочный файл Qtwidgets, представляющий собой файл модуля и включающий в себя заголовочные файлы для используемых в нашей программе классов: QApplication и QLabel.

Разберем наш пример. Сначала создается объект класса QApplication, который осуществляет контроль и управление приложением. Для его создания в конструктор этого класса необходимо передать два аргумента. Первый аргумент представляет собой информацию о количестве аргументов в командной строке, из которой происходит обращение к программе, а второй — это указатель на массив символьных строк, содержащих аргументы, по одному в строке. Любая использующая Qt программа с графическим интерфейсом должна создавать только один объект этого класса, и он должен быть создан до использования операций, связанных с пользовательским интерфейсом.

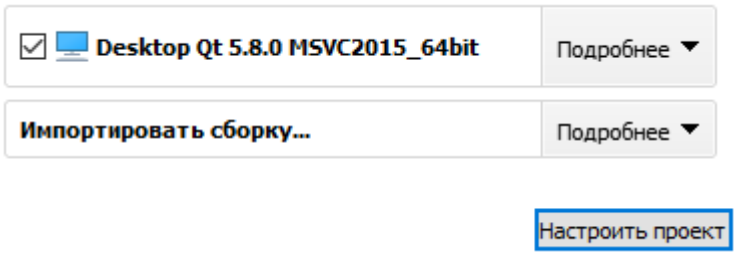
Затем создается объект класса QLabel. После создания элементы управления Qt по умолчанию невидимы, и для их отображения необходимо вызвать метод show(). Объект класса QLabel является основным управляющим элементом приложения, что позволяет завершить работу приложения при закрытии окна элемента. Если вдруг окажется, что в созданном приложении имеется сразу несколько независимых друг от друга элементов управления, то при закрытии окна последнего такого элемента

управления завершится и само приложение. Это правильно, иначе приложение осталось бы в памяти компьютера и расходовало бы его ресурсы.

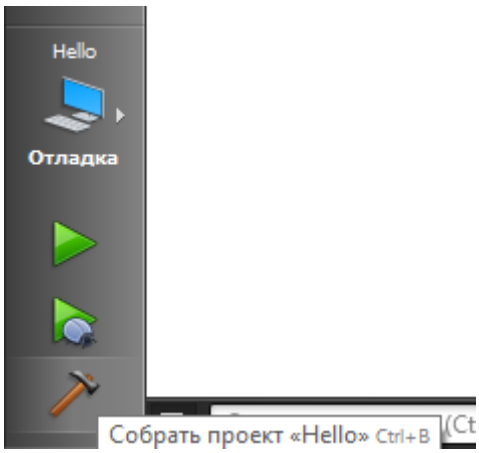
Наконец, в последней строке программы приложение запускается вызовом QApplication::exec(). С его запуском приводится в действие цикл обработки событий, определенный в классе QCoreApplication, являющемся базовым для QGuiApplication, от которого унаследован класс QApplication. Этот цикл передает получаемые от системы события на обработку соответствующим объектам. Он продолжается до тех пор, пока либо не будет вызван статический метод QCoreApplication::exit (), либо не закроется окно последнего элемента управления. По завершению работы приложения метод QApplication::exec() возвращает значение целого типа, содержащее код, информирующий о его завершении.

Для запуска программы необходимо

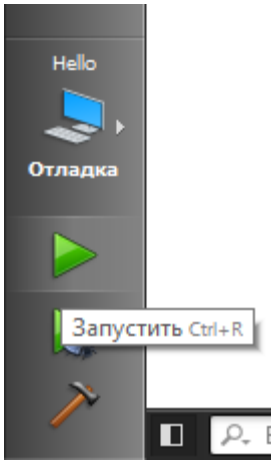
1. в QtCreator открыть файл Hello.pro
2. нажать «Настроить проект»



3. нажать собрать проект



4. для запуска приложения - нажать «Запустить»



Задание для самостоятельной работы:

1. Создать файлы «Second.cpp» и «Second.pro».
2. Написать приложение, которое выведет ваше «Имя Фамилия – студент 3 курса».