Qt 嵌入式图形开发（基础篇）

作者：深圳市优龙科技有限公司　　时间：2004/6/3

Qt 是 Trolltech 公司的一个标志性产品。Trolltech 公司 1994 年成立于挪威，但是公司的核心开发团队已经在 1992 年开始了 Qt 产品的研发，并于 1995 年推出了 Qt 的第一个商业版，直到现在 Qt 已经被世界各地的跨平台软件开发人员使用，而 Qt 的功能也得到了不断的完善和提高。

Qt 是一个支持多操作系统平台的应用程序开发框架，它的开发语言是 C++。Qt 最初主要是为跨平台的软件开发者提供统一的，精美的图形用户编程接口，但是现在它也提供了统一的网络和数据库操作的编程接口。正如微软当年为操作系统提供了友好，精致的用户界面一样，今天由于 Trolltech 的跨平台开发框架 Qt 的出现，也使得 UNIX、LINUX 这些操作系统以更加方便、精美的人机界面走近普通用户。

Qt 是以工具开发包的形式提供给开发者的，这些工具开发包包括了图形设计器，Makefile 制作工具，字体国际化工具，Qt 的 C++类库等等；谈到 C++的类库我们自然会想到 MFC，是的，Qt 的类库也是等价于 MFC 的开发库，但是 Qt 的类库是支持跨平台的类库，也就是说 Qt 类库封装了适应不同操作系统的访问细节，这正是 Qt 的魅力所在。目前，Qt 可以支持的操作系统平台如下：

◆ MS/Windows 95、Windows 98、WindowsNT 4.0、Windows 2000、Windows XP；

◆ Unix/X11 Linux、Sun Solaris、HP-UX、Compaq True64Unix、IBM AIX、SGI IRIX 和很多其它 X11 平台；

◆ Macintoshi Mac OSX；

◆ 嵌入式的，包含有 FramBuffer 的 Linux 平台。

Qt 的资源

trolltech 的主页：http://www. trolltech.com/
支持匿名访问的 FTP: ftp://ftp.trolltech.com
新闻组服务器：nntp.trolltech.com
非官方的 Qt 文档中文翻译小组： http://www.qiliang.net/qt/index.html

一、 认识 Qt/Embedded 嵌入式工具开发包

1.1 介绍

Qt/Embedded 是一个为嵌入式设备上的图形用户接口和应用开发而订做的C＋＋工具开发包. 它通常可以运行在多种不同的处理器上部署的嵌入式Linux操作系统上。如果不考虑X窗口系统的需要，居于 Qt/Embedded 的应用程序可以直接对缓冲帧进行写操作。除了类库以外，Qt/Embedded还包括了几个提高开发速度的工具，使用标准的Qt API,我们可以非常熟练的在Windows和Unix编程环境里开发应用程序。

Qt/Embedded 是一组用于访问嵌入式设备的 Qt C++ API；Qt/Embedded 的Qt/X11, Qt/Windows 和Qt/Mac版本提供的都是相同的API和工具。Qt/Embedded还包括类库以及支持嵌入式开发的工具。

Qt/Embedded提供了一种类型安全的被称之为信号与插槽的真正的组件化编程机制，这种机制和以前的回调函数有所不同。Qt/Embedded还提供了一个通用的widgets类，这个类可以很容易的被子类化为客户自己的组件或是对话框。针对一些通用的任务，Qt还预先为客户定制了像消息框和向导这样的对话框。

运行Qt/Embedded 所需的系统资源可以很小，相对X窗口下的嵌入解决方案而言，Qt/Embedded只要求一个较小的存储空间（Flash）和内存。Qt/Embedded可以运行在不同的处理器上部署的Linux系统，只要这个系统有一个线性地址的缓冲帧并支持C＋＋的编译器。你可以选择不编译Qt/Embedded某些你不需要的功能，从而大大减小了它的内存占有量。

Qt/Embedded包括了它自身的窗口系统，并支持多种不同的输入设备。

开发者可以使用他们熟悉的开发环境来编写代码。Qt的图形设计器(designer)可以用来可视化地设计用户接口，设计器中有一个布局系统，它可以使你设计的窗口和组件自动根据屏幕空间的大小而改变布局。开发者可以选择一个预定义的视觉风格，或是建立自己独特的视觉风格。使用UNIX/LINUX操作系统的用户，可以在工作站上通过一个虚拟缓冲帧的应用程序仿真嵌入式系统的显示终端。

Qt/Embedded也提供了许多特定用途的非图形组件，例如国际化，网络和数据库交互组件。

Qt/Embedded是成熟可靠的工具开发包，它在世界各地被广泛使用。除了在商业上的许多应用以外，Qt/Embedded还是为小型设备提供的Qtopia应用环境的基础。
Qt/Embedded以简洁的系统，可视化的表单设计和详致的API让编写代码变得愉快和舒畅。

1.2 系统要求

Qt/Embedded很省内存，因为它不需要一个X 服务器或是 Xlib库，相反它可以直接的写缓冲帧，它的内存消耗可以通过不编译某些不使用的功能来动态调节，它甚至可以
把全部的应用功能编译链接到一个简单的静态链接的可执行程序中，从而能够最大的节省内存。

Qt/Embedded可以运行在被Linux支持的所有的处理器上，当然我们所说的“Linux支持某个处理器”是指已经有一个Linux的c＋＋编译器支持产生该处理器的目标代码以及Linux操作系统已经顺利移植到这个处理器上。目前Qt/Embedded可以运行在Intel x86, MIPS, ARM, StrongARM, Motorola 68000 and PowerPC等处理器上。

Qt/Embedded的应用程序可以直接的写内核缓冲帧，它支持的线性的缓冲帧包括1, 4, 8, 15, 16, 24和32位深度以及VGA16的缓冲帧，任何被内核支持的图形卡也可以工作，通过对Qt/Embedded进行客户化的定制可以使得它从图形加速系统获得好处。Qt/Embedded对显示屏幕的尺寸并无限制，另外它还有许多先进的功能，例如反别名字体、alpha-blended位图和屏幕旋转等。

Qt/Embedded的库可以通过在编译时去除不需要的功能来进行精简。例如，要想不编译QlistView，我们可以通过定义一个QT_NO_LISTVIEW 的预处理标记来达到此目的；
如果我们不想编译支持国际化的功能，那么我们可以定义QT_NO_I18N 的预处理标记。.Qt/Embedded 提供了大约200个可配置的特征，由此在Intel x86平台上库的大小范围会在700KB到5000KB之间。大部分客户选择的配置使得库的大小在1500 KB 到 4000 KB之间。

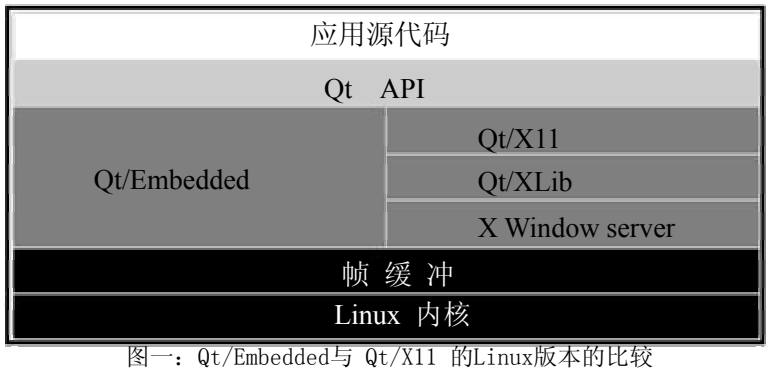
Qt/Embedded 还使用了一些节省内存空间的技术，例如隐式共享(写时复制)和缓存。 在Qt中有超过20个类，包括 QBitmap, QMap, QPalette, QPicture,QPixmap 和 QString, 使用了隐式共享技术，目的是避免不需要的复制和最小的内存需求。隐式共享过程会自动的发生，从而使编程更简单并且避免了处理指针和最优化时带来的危险。

许多的Qt组件都会被编译成为库的形式或者是插件的形式，客户视觉组件，数据库驱动，字体格式读写，图形格式转换，文本解码和窗体可以被编译成插件，从而减小核心库的大小并提供了更大的弹性。作为一种选择，如果对组件和应用了解得很深入的话，它们也可以被编译并和Qt/Embedde的库静态链接到一个简单的可执行程序，从而节省ROM,RAM和CPU的使用。

1.3 架构

Qt/Embedded 为带有轻量级窗口系统的嵌入式设备提供了一个标准的 Qt API. 。Qt/Embedded的面向对象的设计使得它一直能不断的向前支持像键盘，鼠标和图形加速卡这样的额外设备。

通过使用Qt/Embedded, 开发者可以感受到在Qt/X11, Qt/Windows和 Qt/Mac等不同的版本下使用相同的API编程带来的便利。



图一：Qt/Embedded与 Qt/X11 的Linux版本的比较

使用单一的API进行跨平台的编程可以有很多好处。提供嵌入式设备和桌面计算机环境下应用的公司可以培训开发人员使用同一套工具开发包，这有利于开发人员之间共享开发经验与知识，也使得管理人员在分配开发人员到项目中的时候增加灵活性。更远一步来说，针对某个平台而开发的应用和组件也可以销售到Qt支持的其它平台上，从而以低廉的成本扩大了产品的市场。

1．3．1 窗口系统

一个 Qt/Embedded窗口系统包含了一个或多个进程，其中的一个进程可作为服务器。这个服务进程会分配客户显示区域，以及产生鼠标和键盘事件。这个服务进程还能够提供输入方法和一个用户接口给运行起来的客户应用程序。这个服务进程其实就是一个有某些额外权限的客户进程。任何程序都可以在命令行上加上“－qws”的选项来把它作为一个服务器运行。

客户与服务器之间的通信使用共享内存的方法实现，通信量应该保持最小，例如客户进程直接访问帧缓冲来完成全部的绘制操作，而不会通过服务器，客户程序需要负责绘制它们自己的标题栏和其它式样。这就是Qt/Embedded库内部层次分明的处理过程。

客户可以使用QCOP通道交换消息。服务进程简单的广播QCOP消息给所有监听指定通道的

应用进程，接着应用进程可以把一个插槽连接到一个负责接收的信号上，从而对消息做出响应。消息的传递通常伴随着二进制数据的传输，这是通过一个QDataStream类的序列化过程来实现的，有关这个类的的描述，见28页的“非图形类”。

QProcess类提供了另外一种异步的进程间通信的机制。它用于启动一个外部的程序并且通过写一个标准的输入和读取外部程序的标准输出和错误码来和它们通信。

1．3．2 字体

Qt/Embedded支持四种不同的字体格式：True Type字体（TTF）,Postscript Type1字体，
位图发布字体（BDF）和Qt 的预呈现（Pre-rendered）字体（QPF）。Qt还可以通过增加QFontFactory的子类来支持其它字体，也可以支持以插件方式出现的反别名字体。

每个TTF或者TYPE1类型的字体首次在图形或者文本方式的环境下被使用时，这些字体的字形都会以指定的大小被预先呈现出来，呈现的结果会被缓冲。根据给定的字体尺寸（例如10或12点阵）预先呈现TTF或者TYPE1类型的字体文件并把结果以QPF的格式保存，这样将可以节省内存和CPU处理时间。QPF文件包含了一些必要的字体，这些字体可以通过makeqpf工具取得，或者通过运行程序时加上“－savefonts”选项获取。如果应用程序中使用到的字体都是QPF格式，那么Qt/Embedded将被重新配置，并排除对TTF和TYPE1类型的字体的编译，这样就可以减少Qt/Embedded的库的大小和存储字体的空间。例如一个10点阵大小的包含所有ASII字符的QPF字体文件的大小为1300字节，这个文件可以直接从物理存储格式映射成为内存存储格式。

Qt/Embedded的字体通常包括Unicode字体的一部分子集，ASII和Latin-1。一个完整的16点阵的Unicode字体的存储空间通常超过1M，我们应尽可能存储一个字体的子集，而不是存储所有的字，例如在你的应用中，你仅仅需要以Cappuccino字体、粗体的方式显示你的产品的名称，但是你却有一个包含了全部字形的字体文件。

1．3．3 输入设备

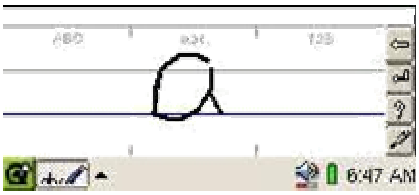
Qt/Embedded 3.0支持几种鼠标协议：BusMouse, IntelliMouse,Microsoft和MouseMan.
Qt/Embedded 还支持 NEC Vr41XX 和 iPAQ 的触摸屏。通过从QWSMouseHandler 或者QcalibratedMouseHandler派生子类，开发人员可以让Qt/Embedded支持更多的客户指示设备。

Qt/Embedded支持标准的101键盘和Vr41XX按键，通过子类化QWSKeyboardHandler可以让Qt/Embedded支持更多的客户键盘和其它的非指示设备。

1．3．4 输入方法

对于非拉丁语系字符（例如阿拉伯，中文，希伯来和日语）的输入法，需要把它写成过滤器的方式，并改变键盘的输入。输入法的作者应该对全部的Qt API的使用有完全的认识。

在一个无键盘的设备上，输入法成了唯一的输入字符的手段。Qtpia提供了四种输入方法：笔迹识别器，图形化的标准键盘，Unicode键盘，居于字典方式提取的键盘。这些键盘的样式如下所示



笔迹识别器

Unicode 输入

标准键盘

字典式提取键盘

1．3．5 屏幕加速

通过子类化QScreen 和 QgfxRaster 可以实现硬件加速，从而为屏幕操作带来好处。Trolltech提供了Mach64 和 Voodoo3 视频卡的硬件加速的驱动例子，同时可以按照协议编写其它的驱动程序。