Qt 4 初學者手冊

陳佳新◎著

應用奇步

https://leanpub.com/qt4-for-beginners

Qt 4 初學者手冊

繁體中文版

陳佳新

This book is for sale at http://leanpub.com/qt4-for-beginners

This version was published on 2014-03-17

Tweet This Book!

Please help 陳佳新 by spreading the word about this book on Twitter!

The suggested hashtag for this book is #QtForBeginners.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

https://twitter.com/search?q=#QtForBeginners

獻給內人與兒子們
你們永遠是我前進的動力

To my wife and sons:
You all are always my power to move on.

Contents

改版記錄

版本日期	版本摘要
2014/03/17	* 內容重新編排後以 Leanpub 出版
2009/02/23	* 投稿並刊登於電腦雜誌

關於作者

我是陳佳新（Jia-Sin Chen），1980 年生，A 型雙魚座。網路上常用的 ID 是 jarsing，寫部落格時則喜歡自稱「J 老闆」。諳 programming，熱中寫作與翻譯。在平面媒體刊登過超過 200 篇文章，並且翻譯了 16 本電腦書，包括《Microsoft Visual C++ .NET Step by Step》（Microsoft Press）、《Hacking and Securing iOS Applications》（O'Reilly）和《Programming Android》（O'Reilly）等。

在 2006 年到 2011 年之間，我主要的研發工作都與 Qt 脫離不了關係。涵蓋 Linux 桌面、Linux 嵌入式系統、Windows 桌面等平台，除了開發一些幫助自己日常生活的小程式，也做出了像是語音電話和影音串流機上盒等產品的 UI 介面，以及網路設備的自動化測試工具。哦，對了，期間還寫了 6 篇文章，刊登在電腦雜誌上。而且還親眼見證了 Qt 母公司從本來的 Trolltech 易主為 Nokia，最後再由 Digia 接手，有趣的是，他們 3 家都是北歐公司。

- 個人網站：http://jarsing.com
- 電郵信箱：jarsing@jarsing.com

奇步應用

近年來隨著行動加值應用的崛起，我的程式開發重心已從嵌入式系統轉往智慧型手機上的 iOS/Android App。2013 年 5 月，我與內人共同創辦了一家目前尚未進行營利事業登記的軟體公司，叫做奇步應用，歡迎從臉書粉絲專頁關注我們。

致謝

感謝《電子情報》雜誌長久以來提供了我發表電腦心得文章的版面。

關於本書

筆者曾經於 2009 年到 2010 年之間在雜誌上刊登過 6 篇 Qt 文章，其中最基礎的那篇（同時也是本書的主要內容，完成於 2009 年 2 月底）曾經被筆者拿來作為白天任職公司的內部訓練教材，協助開發團隊的成員快速上手。

筆者當初在寫作雜誌文章時使用的是稍舊的 Qt 4.5 版本，所以本書並不打算講 Qt Quick 和 QML 這些 Qt 4.7 之後才導入的技術，GUI 甚至是 Qt Designer 而非 Qt Creator 來設計的，此外將透過命令列來完成編譯的工作。

值得慶幸的是，舊版的東西並未被 Qt 現任母公司 Digia 所揚棄，因此軟體和文件都可以在下列位置找到：

- 舊版軟體專區：http://download.qt-project.org/archive/qt/
- 舊版文件專區：http://doc.qt.digia.com/archives/index.html

Qt 4 不死

既然現在都已經是 Qt 5 的時代了，為什麼 Qt 4 的書籍還有被閱讀的價值呢？筆者認為有下列 2 個理由：

1. 必須維護舊的 Qt 4 專案（現實考量，本來的東西跑得好好的，沒道理去更新它）。
2. 尚未升級 Qt 5 企業授權（成本考量，企業版的授權費用也是一筆好大的支出呀）。

所以 Qt 4 不死，只會逐漸凋零。

在 2014 年 3 月 12 日由 Qt 母公司 Digia 所舉辦的 Meet Qt in Taipei 研討會中，來自北京分公司的技術顧問講師夏春萌先生詢問了在場的台灣本地大約 70 多位的與會者，結果：

1. 開始採用 QML 的人數大約只佔了 1/3 。
2. 還在繼續使用 Qt 4 的比例則超過一半。

可以確定的是，筆者已經著手翻修本書，將會推出後繼的《Qt 5 初學者手冊》，並且提供非常優惠的折價券給所有曾經購買過本書的讀者，所以現在仍是購買本書的最佳時機。

購買本書

歡迎購買本書！如果您喜歡這本書的話，除了可以重複購買之外，也可以透過超額付款來表達支持。購買網址如下：

https://leanpub.com/qt4-for-beginners

團體優惠

針對想要學習 Qt 並有意組織讀書會的人士，筆者提供了團體優惠，3 人以上團購即可獲得這樣的優惠，歡迎來信索取折價券：jarsing@jarsing.com 。

學生優惠

針對有興趣學習 Qt 的在學學生，筆者提供了比團購更為優惠的折扣，歡迎來信索取折價券並請附上學校與科系：jarsing@jarsing.com 。

前言

貫穿本書的範例，是一個比「Hello, World!」更為進階的圖片縮小程式，屬於影像處理方面的應用。你將會看到如何透過 Qt 從無到有打造這支程式，作為初學者的導師，筆者將這個實作過程拆解成許多段落與步驟，並且融入大量的細節說明。

章節編排

- 第 1 章：快速入門指南
- 第 2 章：照片縮小程式

讀者服務

- 本書範例下載網址：http://jarsing.com/school/qt/ImageDownSizer.zip
- 您可以寄送郵件到：jarsing@jarsing.com
- 或者線上填寫表單：http://jarsing.com/form/qfb/

粉絲專頁

歡迎關注臉書「奇步應用」粉絲專頁：http://fb.com/chibuapp

交流社團

歡迎加入臉書「奇步 Qt 初學者手冊」交流社團：http://fb.com/groups/qt.for.beginners/

1 快速入門指南

本章學習重點

- 透過 5W1H 來認識 Qt
- 從 3 個面向走入 Qt
- 物件模型與事件機制
- Qt 歷史與授權模式

1.1 關於 Qt 的 5W1H

關於 Qt，我們不妨透過 5W1H 的問答方式來做個概括性的介紹。

Qt 唸做 cute，跟英文的「可愛」同音。但是筆者發現咱們華人總是喜歡把 Q 和 T 兩個字母分開來唸，即便是來自 Digia 北京分公司的技術顧問講師也一樣。:)

為何（Why）使用 Qt？因為 Qt 是跨平台的應用程式框架（Application Framework），所以基本上能夠「程式碼只寫一遍，然後在任何平台上編譯成執行檔」，再者 Qt 線上的文件和論壇都相當活躍（甚至到了讓手機大廠 Nokia 花錢將其母公司 Trolltech 給買下來的程度；但是隨著 Nokia 從智慧型手機市場敗下陣來，已經易主給 Digia 了），所以使用 Qt 的人並不會感到寂寞。

在繼續往下閱讀此之前，讓我們先來瞭解一下到底何謂「應用程式框架」？根據 Wikipedia 上面的定義，大致上是：用來實作特定作業系統上面之應用程式的「標準結構」，搭著 GUI 程式設計的順風車而變得流行起來，通常都以物件導向語言來實作這樣的框架，因為在開發應用程式時可以很容易地繼承框架內的既有類別來加以擴充使用之。知名的應用程式框架包括 Mac OS X 平台上的 Cocoa 、Windows 平台上的 MFC 和.NET，以及跨平台的 Qt 和 Java 等。

Qt 有哪些（What）特色及其應用範圍（Where）？跨平台顯然是其最大特色，在桌面和嵌入式系統領域也都相當吃得開，除此之外程式庫本身的豐富性也是吸引人們採用的主因之一，可以藉此很快速地開發 GUI 、資料庫、XML 、多執行緒、網路乃至瀏覽器等各式各樣的應用程式，而且支援多國語系。

誰（Who）該使用 Qt 及其使用時機（When）？實際上，如果你有打算讓應用程式可以跨平台的話，那麼 Qt 會是個很不錯的選擇，不過就算暫時沒有這個打算的話，剛才提到過的 Qt 所擁有的完善程式庫也可以幫助開發人員提升生產力。

最後也最重要的問題則是：如何（How）學習 Qt？大哉問，不過通常其官方的線上文件網站[1]就很足夠了，點選想要採用的平台或版本（例如 Qt 4.5）進入更詳細的分類頁面。

舉例而言，Qt 初學者可以先看〈How to Learn Qt〉這篇文章，若要安裝的話就看〈Installation〉這篇（Windows 版本在安裝完畢之後，可以接著再去啓動「程式集 → Qt 群組資料夾」內的「Examples and Demos」導覽程式，以便體驗一下 Qt 可以寫出什麼樣子的應用程式來），而如果是要查詢程式庫的細部用法，則可以點選〈All classes〉進入檢閱，或者是透過「程式集 → Qt 群組資料夾」裡面的「Qt Assistant」（中文版叫做「Qt 小幫手」）來進行離線瀏覽。

1.2 Qt 入門的 3 個面向

總的來說，Qt 入門可以從 3 個面向（或者應該說是「必備基礎」）來加以探討，包括：

1. 工具集：例如利用「Qt Designer」（中文版叫做「Qt 設計家」）來規劃 GUI 介面，以及利用「qmake」來產生建置執行檔所需的 Makefile 等。
2. 應用程式框架：例如用來處理影像圖檔的是 QImage 類別，用來進行文字編碼轉換的是 QString 類別等。
3. C++ 程式語言：例如必須知道物件導向、繼承、命名空間等。這些議題在稍後遇到實際程式碼時都會再做更進一步的介紹與演練。

另一方面，根據《Qt 4.4 Whitepaper》這份文件的介紹，Qt 的功能是奠基在個別平台的原生低階 API 之上，而非像傳統的階層式跨平台框架是將特定平台的框架（例如 Windows 平台上是 MFC、X11 平台上是 Motif）又再重新包裝了一遍，導致在速度上慢了許多，而且功能上也會多所牽制。相反地，這些與個別目標平台緊密相關的實作苦工都由 Qt 公司方面完成了，其上則是通用的類別程式庫，再之上才是 Qt 應用程式的原始碼，因此促成 Qt 應用程式的單一原始碼可以跨越多種不同的執行平台。

1.3 Qt 的物件模型與事件機制

那麼 Qt 的 C++ 框架要如何提供支援 GUI 程式設計的能力呢？最簡單的問題包括：物件如何溝通？事件如何處理與反應？其所仰賴的是 Qt 特有的物件模型，Qt 導入了一套自己的中繼物件系統（meta-object 在字面上的意思就是「關於物件的物件」），亦即由這套系統來負責建立和管理 Qt 所有衍生自 QObject 的物件，以便提供包含 signal/slot（信號/信號槽）事件處理在內的重要機制。這套中繼物件系統的實現需要搭配 3 個主角，包括：

[1] 針對 Qt 4.7 之前的舊版文件，請移駕到這裡。

1. 基礎類別 QObject：用來支援此中繼物件系統的物件。
2. Q_OBJECT 巨集：在類別宣告內用來啓用這套中繼物件系統。
3. 「中繼物件編譯器工具」moc（Meta-Object Compiler）：提供 QObject 衍生物件在實現中繼物件能力時所需之程式碼，當此工具偵測到類別宣告裡面出現了 Q_OBJECT 巨集時，便會產生檔名前綴「moc_」的額外 C++ 原始碼檔案，裡頭會包含所需的相關中繼物件程式碼，然後與原本專案裡面的程式碼一起編譯。

事實上，導入中繼物件系統的最主要原因，就是為了要提供 signal/slot 機制，好讓物件得以溝通。舉例而言，在 GUI 程式設計裡面，當我們改變了某個 Widget 的狀態時（例如按下了〔關閉〕按鈕），通常會希望其他某個 Widget 收到通知並做出反應（例如導致呼叫主程式的「結束」函式）。關於 signal/slot 機制的程式碼示範如下：

```
QObject::connect(btn, SIGNAL(clicked()), &app, SLOT(quit()));
```

透過 connect() 將發送端按鈕 btn 的 clicked() 信號與接收端主程式 app 的 quit() 函式連結在一起，所以當 btn 被按下時，程式便會結束。在本例中，可以發現 SIGNAL 和 SLOT 都是名稱為大寫字母的巨集，然後裡頭的函式 clicked() 和 quit() 都擁有相同的特徵（Signature），也就是都沒有帶參數。不過這裡必須特別留意一點，因為編譯器並不會去檢查 slot 所指定的函式是否存在，也不會檢查其參數列特徵，所以筆者曾經遇過打錯函式名稱而導致無法接收到 signal 並做出反應的窘境，花了老半天才發現問題出在拼字錯誤的經驗，實在非常冤枉。

換句話說，就是當事件發生時會發佈 signal，由所指定的 slot 函式來負責處理之。Qt 裡面有許多預先定義好的 signal 和 slot，當然程式開發人員也可以因應專案需求隨時加入自訂的版本。稍後我們會示範自訂的 signal/slot，並以此機制來傳回 3 項結果數據。

傳統上，這類物件之間的事件溝通可以透過回呼（Callback）方式實現。回呼本身是個函式指標，如果希望某個負責處理的函式在收到特定事件時能夠呼叫特定的函式，那麼就必須把這個指標傳遞給前者，由它在遭遇觸發點時呼叫後者。然而回呼方式有 2 個根本上的缺陷，既未顧及型別安全性（Type Safety），而且耦合性（Coupling）也過強：

- 未顧及型別安全性：意味著程式在執行時有可能會發生型別錯誤，因為無法肯定回呼函式所帶的參數型別是否正確，無怪乎常言道「型別定義良好的程式永遠不會出錯」。
- 耦合性過於強烈：耦合性是指關聯性，過於強烈意味著負責處理事件的函式必須清楚知道自己應該呼叫哪個回呼函式才對。

Qt 的 signal/slot 機制解決了上述這 2 個問題。首先，因為 signal 和 slot 兩者的函式參數列特徵必須相容，否則發送端的 signal 與接收端的 slot 根本無法連結在一起，所以此機制符合型別安全性。再者，物件在發出 signal 時，從來就不曉得也不在乎會由哪個 slot 去接收處理，所以此機制屬於弱耦合（Loose Coupling）。事實上，一個 signal 要被多少個 slot 接收，或者一個 slot 要連結多少個 signal 都是沒有限制的，甚至因應需求必須將多個 signal 串接

起來也是沒有問題的。總結來說，Qt 的 signal/slot 機制實現了真正的資訊封裝，確保所有物件都像獨立的軟體元件般可以被盡情地運用。

至於為什麼不透過 C++ 模板（Template）來加以擴充，反而是利用 C 巨集來實作 Qt 的中繼物件系統呢？說來話長，總之就是因為有些事情不是透過模板技巧所能夠實現的，再加上不少 C++ 編譯器對於進階的模板支援度並不高。所以儘管泛型程式設計（Generic Programming）可以讓程式碼比較漂亮、容易閱讀、節省程式大小，但是 Qt 選擇的還是在編譯器上面普遍支援度極佳的巨集搭配 C 前置處理器的技巧，畢竟這種作法很有彈性，因而犧牲一些效能也是不得已的事。

1.4 Qt 簡史

Qt 應用程式框架首次發表是在 1995 年 5 月，建構這套 C++ GUI 框架的靈感起源更是早在 1988 年受瑞典公司委託便已萌生，初始開發者是 Haavard Nord 和 Eirik Chambe-Eng 兩位挪威人，他們於 1991 年開始著手開發相關的 C++ 類別，並於 1994 年合組公司。Qt 名稱的由來是因為字母「Q」在 Haavard 的 Emacs 編輯器字型裡面看起來很漂亮，而字母「t」則是意指 toolkit。公司名稱從 Quasar Technologies 變成 Troll Tech，再演變成今天的 Trolltech，但是在 2008 年 6 月被 Nokia 併購後又更名為 Qt Software。

以 Qt 開發的知名軟體與專案包括：KDE（Unix 平台上强大的桌面系統）、Opera（號稱是地球上最快速的瀏覽器）、Google Earth（透過衛星地圖從外太空看地球）、Skype（免費撥打網路電話）、Qt Extended（Qt 在以嵌入式 Linux 系統為基礎的手持式行動裝置上的延伸平台）、Adobe Photoshop Album（相片編輯軟體）、VirtualBox（x86 虛擬化軟體），以及 OPIE（開放的行動裝置 GUI 環境）等。在 2014 年 3 月 12 日的 Meet Qt in Taipei 研討會上聽到的消息是，就連電動汽車 Tesla 中控 10 吋螢幕的 GUI 系統也是使用 Qt 開發的。

2007 年 6 月正式發表叫做 Qt Jambi 的 Java 版本框架（但是已於 2009 年 2 月宣布不再繼續發展），其他非官方推出的則還有叫做 PyQt 的 Python 版本框架、叫做 PHP-Qt 的 PHP 版本框架、叫做 QtRuby 的 Ruby 版本框架、叫做 Perl Qt4 的 Perl 版本框架、叫做 QtAda 的 Ada 版本框架，以及叫做 Qyoto/Kimono 的 C# 版本框架等，在 Nokia 併購了 Trolltech 之後，可在 Symbian 平台上面使用的 Qt/S60 版本框架也正在緊鑼密鼓地開發中。

1.5 Qt 的授權模式

Qt 4 近來演進的幾個里程碑包括：Qt 4.1 支援 SVG 向量圖形；Qt 4.2 導入套用類似 CSS 樣式表的能力；Qt 4.3 提供了 ECMAScript/JavaScript 處理引擎，可以執行腳本並呈現結果；Qt 4.4 整合了開放原始碼的 WebKit 瀏覽器引擎；以及 Qt 4.5 的開放原始碼版本由原先的 GPL 轉向 LGPL 授權方式。

針對轉向 LGPL 授權這個部分，Nokia 此舉意味著 Qt 被採用的比例將會出現大幅度的成長（比方說可能就會有些專案從採用 GTK+ 或 wxWidgets 倒戈過來改用 Qt 了），因為 LGPL 允許動態連結程式庫（就像有些 VB 6 程式是在執行期間才動態載入 VB 6 Runtime 程式庫 msvbvm60.dll 那樣，所以未來我們可能會看到不少發問缺少 Qt 程式庫該怎麼辦的帖子在網

路論壇上出現），所以只要不去更動底層的 Qt 程式庫，那麼上層用 Qt 開發的應用程式將可以不必開放其原始碼，換言之可以保持為商業性質的封閉原始碼軟體。不像以前那樣，引用了 GPL 授權的 Qt 程式庫之後，整個應用程式也會被「感染」成 GPL，導致企業必須購買商業授權，要不然就不能夠公開展示。如今授權費用的阻礙因素可謂已經移除，而且為了集中資源於原始 C++ 版本 Qt 的發展上，所以 Java 版本的 Qt Jambi 也會在 Qt 4.5 之後宣告終止，轉為由社群力量驅動的開放版本，足見 Nokia 深耕跨平台應用程式框架桌面市場（Qt/X11、Qt/Windows、Qt/Mac）以及嵌入式系統市場（Qt/Embedded Linux、Qt/Windows CE）的決心。至於主攻手持式行動裝置的 Qt Extended 版本的授權方式是否跟著改變，則仍有待觀察，至少目前 FAQ 上的答案是否定的。

此外 Nokia 也將開放 Qt 原始碼庫（Repository），鼓勵更多開發人員社群貢獻其心力。那麼到底人們還需要購買商業授權做什麼用呢？FAQ 上面的官方說法是商業授權可以獲得 Qt 團隊的電子郵件技術支援，以及針對程式庫本身的修改也會不受限制（尤其如果因而提升大量效能，而且那正好是你們公司核心競爭力的話），也就是說有付錢的話可能會比較心安（有些人覺得用開放原始碼的東西風險較高），軟體出了差錯可以寫信去罵；另外一個重點則是，在更開放的 LGPL 授權模式下，將可以吸納更多的社群力量（以往的 GPL 授權模式必然使得許多人卻步，起碼在心情上不會多麼愉快），最終 Nokia 也將可以把這些部分整合到自己的產品裡面，等於是說未來 Qt 帶來的營收雖然銳減，但是其研發能量卻是由社群共同提供的。總結來說，就像一些教育訓練和顧問機構所做的研究調查報告裡面講述的，這是一份 Nokia 送出的大禮（放棄了坐收大筆權利金的機會），有助於 Qt 成為業界標準，因此預料 Qt 未來將會因此成為嵌入式系統領域的首選 GUI 框架，而既有的舊版 Qt 應用程式也將會加速遷移至 Qt 4.5。

1.6 本章結語

既然現在都已經是 Qt 5 的時代了，為什麼 Qt 4 的書籍還有被閱讀的價值呢？筆者認為有下列 2 個理由：

1. 必須維護舊的 Qt 4 專案（現實考量，本來的東西跑得好好的，沒道理去更新它）。
2. 尚未升級 Qt 5 企業授權（成本考量，企業版的授權費用也是一筆好大的支出呀）。

在 2014 年 3 月 12 日由 Qt 母公司 Digia 所舉辦的 Meet Qt in Taipei 研討會中，來自北京分公司的技術顧問講師夏春萌先生詢問了在場的台灣本地大約 70 多位的與會者，結果：

1. 開始採用 QML 的人數大約只佔了 1/3 。
2. 還在繼續使用 Qt 4 的比例則超過一半。

所以 Qt 4 不死，只會逐漸凋零，現在仍是學習 Qt 4 的最佳時機。