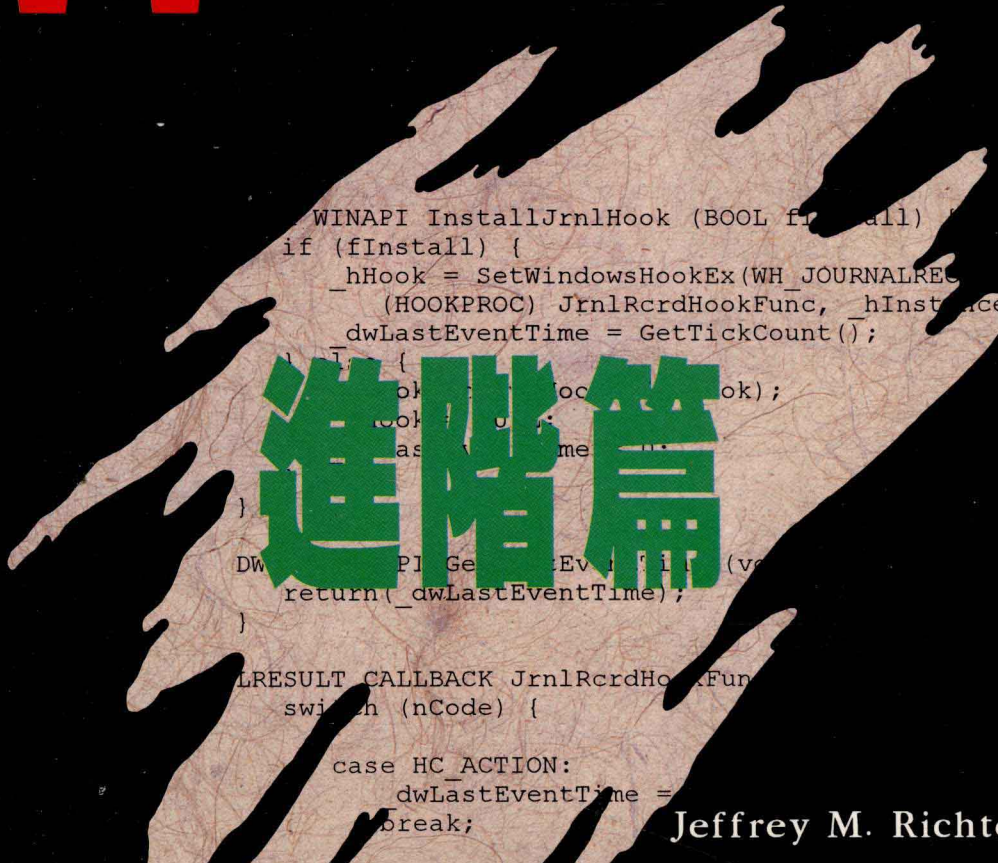
WINDOWS 程式設計實務(二)

進階篇



Jeffrey M. Richter 原著

施威銘研究室　譯

旗標出版有限公司

WINDOWS 程式設計實務(二)

Windows 3.1: A Developer's Guide

2nd Edition

Jeffrey M. Richter 原著

施威銘研究室 譯

Original English language edition published by M&T Books All Rights Reserved. Copright © 1992

旗標出版有限公司
聘任本律師爲常年法律顧問，如有侵害其信用名譽權利及其它一切法益者，本律師當依法保障之。

牛湄湄 律師

Windows 程式設計實務（二）進階篇

英文原著作人 :Jeffrey Richter
中文翻譯著作人:施威銘研究室
發 行 人 :施威銘
發 行 所 :旗標出版有限公司
台北市忠孝東路一段83號19樓
電 話:396-3257（代表號）
傳 眞:321-2545
郵撥帳號:1332727-9
帳 戶:旗標出版有限公司
印 刷 者:合豐印刷公司

定 價: 580 元（附磁片）
82 年 12 月初版

行政院新聞局核准登記: 局版台業字第 4512 號

ISBN 957-717-067-6

感　謝

首先我得聲明：如果沒有以下這些人的幫忙，您就看不到在您手上的這本書。且讓我向他們致上最高的謝意！

感謝 Susan Ramee 小姐。妳長期不辭辛苦地書寫及打字，爲本書的付出與貢獻，可見諸於書中的每一頁、每一行，每一個字。各位讀者可以在每一章節的字裡行間欣賞到妳所做的註解、建議、圖表以及漂亮的程式圖示 (icon)。

感謝我的姑母 Donna Murray 在這本書第一版時給我的協助，耐心的閱讀所有的章節、給予適當的建議、討論書中的範例程式。

感謝我的父母 — Arlene 及 Sylvan。當他們買第一部電腦 (TRS-80 Model 1) 給我的時候，還擔心電腦會像以前那些化學儀器一樣，玩了一個禮拜，就被我扔進櫥子裡。我對您們給我永遠的愛與支持感激不盡。

感謝 Jim Harkins 及 Carlos Richardson。你們遠超過一般人的工作表現，絕不是單單幾個 byte, word 或是 double word 所能形容的。不過下次我們再玩「虛擬眞實」遊戲 (virtual-reality game) 時，鹿死誰手還不知道呢！

感謝 Borland 公司的 Dan Horn, 他不斷地提供最新版的 Borland C++ 編譯器供我測試。

感謝 M&T 圖書公司的 Sarah "Here's the FedEx number again" WadsWorth 小姐及 Laura "Somebody might notice" Moorhead 小姐，感謝你們把時間跟精力花在我的書上。

感謝 Elvira Petersman 小姐，她幫我寫出 MDI 範例程式的雛形。

原作者序

Microsoft Windows 3.0 版在 1990 年 5 月 22 日正式與世人見面，不久即以強大的圖形運算能力引起 PC 界一陣風暴。Windows 3.0 固然有許多令人炫目的特性，但不容諱言的，今天它之所以能夠成功，實在是多虧了那些投注了大批金錢與精力、開發 Windows 應用程式的廠商。畢竟，人們所要的不單是一個作業環境，而是能提昇它們工作效率的應用程式。

當然，Microsoft 也瞭解這一點。因此在 Windows 3.0 正式銷售的前一年，就將 "Microsoft Windows Pre-release" 版送到一些專業的 Windows 程式設計者手上。這使得在 Windows 正式發表時，已經有許許多多 Windows 的應用軟體上市。如果沒有這些協力廠商，Windows 就不可能會獲致今天這樣的成就。

然而，令人印象深刻的 Winodws 3.0 Pre-release 程式，若是與 Windows 3.1 相比，簡直是小巫見大巫了。當 Windows 3.1 還在測試版 (Beta) 的階段，Microsoft 就已經累積了一萬名以上的測試者；這些測試者中，有許多是在開發、銷售 Windows 版軟體的廠商。最後當 Windows 3.1 正式亮相時，開發 Windows 軟體的協力廠商也已經更新了他們軟體的版本，準備開始銷售。

戲劇性地，許多公司都紛紛放棄他們所開發軟體的 DOS 版本，而全力投入 Windows 版本。由於 Windows 3.1 支援多媒體 (Multimedia)、手寫輸入 (pen recognition) 等，因此大大拓展了程式發展的空間。這些新技術可貴的地方在於它們縮短了初學者與電腦之間的距離。

Microsoft 目前正在發展 Windows-NT 及 Windows-32。Windows-NT 被定位在 "具有移植性的 Windows 版本"。也就是說，這個版本並不侷限在 Intel 的 CPU 才可以執行。可以想見，Windows-NT 其市場上潛在使用群將是不容忽視的。

此为试读，需要完整PDF请访问：www.ertongbook.com

Microsoft 也開始在計設個人家用的 Windows 版本。幾年之後，您也許可以用 Windows 來設定錄影機、CD 唱盤、電視機、微波爐等等。Windows 的影響力無遠弗屆，市場上對寫 Windows 程式的人才需求將會與日遽增，聰明的您還不從現在開始立志學習 Windows 嗎！

本書是程式設計師深入發掘 Windows 內部功能的得力助手。書中，幾乎每一章都附有一個以上的例子，以闡釋書中的觀念與作法。有些範例是可以完全不經修改即可使用、有些則教您如何活用、組合這些知識；其餘一些獨立的程式或模組，只是單純的顯示如何應用書上的概念，讀者可以自行將他們加進自己的程式中。

寫一個小型的 Windows 程式總是比寫大型程式來得容易。寫大型 Windows 程式時，必須要小心每一細節，並做完整的測試。底下列出寫作 Windows 程式必須注意的事項：

1. 測試您的程式是否能在各種常用的螢幕解析度下執行。通常問題會出在顯示交談窗時，例如文字有可能會被截掉或轉折到下一列。給您一個良心的建議：設計交談窗時，儘量採用使用者所可能用到的最低解析度。

2. 分別測試程式在彩色及黑白環境下的運作。有很多的程式，在設計時將所使用的顏色固定，或者採用預設的顏色，但這樣的作法在黑白的螢幕下，經常會產生無法辨識的情形。這樣實在是蠻糟糕的，舉例來說，如有個文字編輯器將黑色的文字寫在黑色的背景上，您覺得如何呢？

3. 測試您的程式在只用鍵盤操作的情形下，能否正常運作。沒錯，大部份的 Windows 使用者多半會備有滑鼠，但不可否認的，有些使用者還是偏好鍵盤。這樣的測試可以擴展到許多類似的情境。

4. 在交談窗中要注意按下 Tab 後，控制元件取得輸入點的順序。這一點是必須隨時注意的，使用交談窗時，如果按下 Tab 鍵，原本應該要跳到下一個控制元件，但卻跳到另一個去，那可是會令人「抓狂」的。

5. 同樣的，在交談窗中也要測試一下助憶鍵 (nmemonic key)。確定一下助憶鍵間彼此沒有重複，而且在按下助憶鍵 (nmemonic key) 後會正確的跳到所要的控制元件上。

6. 儘量與一般 Windows 程式使用相同的介面，其理由有二：

   A. 當您的程式能照一般的習慣用法操作時，可以縮短學習時間。
   B. 遵循 Windows 的運作方式來設計程式，會讓您事半功倍。

   有些公司總喜歡花上大把的時間與金錢，爲的只是讓程式用起來及看起來與衆不同。結果呢，所付出的代價是：每每會發生一些小 bug，就要再花許多時間去更正或重寫程式。

7. 交給一般的使用者去測試一下您的程式，如果有 bug，他們就會發現。通常程式設計者本身是很難發現自己程式的 bug，而使用者用您的程式時，他們的用法往往會出乎您的意料之外。但是很不幸的一點，測試版的使用者如果發現有 bug 時，通常不會主動向設計者報告；因爲他們會認爲是自己操作錯誤，爲了避免讓人覺得自己過於愚蠢，而不好意思報告。所以呢？我們這些設計者只好儘量試著去打開與使用者之間的溝通管道了。

很抱歉，我在寫上面這些事情時，情緒有點激動，所以顯的有點兒亂。無論如何，設計 Windows 程式時，我們都要對上述的事項非常小心。本書的各個章節中，我們也會時時刻刻小心這些細節的。

編譯本書範例程式所需的工具

本書中所有的範例程式，都可以用 Borland C++ V3.1 來編譯，而編譯過的結果都可以在 Windows 3.0 及 Windows 3.1 上執行（除了一些專用在 3.1 版上的特性，例如：hook 及 drag-and-drop）。當然，雖然範例程式都是用 Borland C++ 編譯的，但您也可以用其他公司的 C/C++ 來編譯。

在範例程式碼中，有些個人的寫作習慣，在此先行說明一下：

1. 所有的變數名稱皆採用標準的匈牙利命名法。
2. 所有的全域變數名稱之前都會加上一個底線字元（"_"）。這樣做可以幫助讀者瞭解某一函式的內容，因爲只要變數名稱之前沒有加底線，那麼該變數若不是傳入該函式的引數，就是函式中的 auto 或 static 區域變數。
3. 在宣告函式的敘述中，函式名稱與括弧間留下一空格，如下所示：

```
int FunctionName (int nX, int nY) { }
                  ^空格字元
```

當呼叫這一個函式時，則在函式名稱與括弧之間不留空格，如下所示：

```
nZ = FunctionName(nX, nY);
                 ^不留空格
```

這樣可以讓您比較容易在程式中找到函式的原型宣告，比如說，使用編修程式時，您可以鍵入 "FunctionName "（注意字串尾端的空白）來搜尋的字串，此時編輯器會幫您找到函式本身，而不會找到呼叫函式的敘述。

4. Windows 3.1 提供了一個新的表頭檔，叫做 WINDOWSX.H 。這一個檔案內定義了許多的巨集，讓使用者可以更容易、更快速的設計出 Windows 程式；這些巨集的大部份是在幫您做一些型別的轉換 (casting) 的工作，因此，您就不需要在您程式中做型別轉換的動作。範例程式中也用了一些 WINDOWSX.H 的巨集。在這個表頭檔中，還有一些叫做 Message Cracker 的巨集，本書只有在 MDI 的程式中用到它；其他程式則爲了讓讀者更容易瞭解，因此並沒有採用 message cracker 的寫法（譯註：爲了使讀者能順利閱讀這些程式，我們在程式設計實務（一）的附錄列有 WINDOWSX.H 與 Message Cracker 的說明，可參考之）。

我對 Message Cracker 有一種複雜的情結；如果您是正在發展一個複雜的 Windows 程式，那我會強烈建議您使用它；如果您只是寫一些簡單的 Windows 程式，那就不需要。但不管怎麼樣，Microsoft 宣稱，如果您使用 Message Cracker 的寫法來撰寫程式，那將來要把程式由 16-bit Windows 轉換成 Win-32 的規格時，會簡單許多；反正一切由您自行決定就是了。

爲什麼給你這本書

這本書是專門爲想要深入瞭解 Windows 作業環境的讀者寫的。市面上有許介紹設計 Windows 程式的書，但大多只討論到一些基本的特性、功能、以及 Windows 的資源。本書並不談這些表面的東西，而是更深入的探討 Windows 進階且威力強大的功能。同時，我們也教您如何把一些 Windows 基本的部份組合起來，而成爲一個完整的程式。

針對每一個主題，本書都會深入探討 Windows 內部的工作方式，並且以範例程式教您如何實際運用。假如您瞭解了這些主題，那將會對您運用 Windows 所提供的各種機制大有幫助，您的程式也會因而變得更爲強大且有效率。這就是本書最主要的目的：提供一個專業 Windows 程式設計者所必備的知識與工具。

本書的寫作風格與閱讀說明

本書每章主題之說明與技巧都附有完整之實例，以方便讀者印證與學習。在次序上，本書是先介紹基本原理，然後舉出程式實例。不過我們並不立即將整個程式全部列出，而是先對每一函式或程式片段加以說明，最後再列出完整的程式。因此讀者閱讀時，最好能不時將內文說明和程式交互比對參考，才能順利瞭解內文的意思。

此为试读,需要完整PDF请访问: www.ertongbook.com

目 錄

第 1 章 剖析視窗

第 2 章 視窗的 Subclassing 和 Superclassing

第 3 章 交談窗 (Dialog-Box) 的技巧

第 4 章 設計自己的子控制元件 (Custom Child Controls)

第 5 章 印表機設定

第 6 章 執行中的程式與佇列

第 7 章 攔截訊息 (HOOKS)

第 8 章 MDI 程式技巧

第 9 章 拉曳存置法 (Drag-and-Drop)

第 10 章 軟體安裝程式

附錄

1

剖析視窗

首先，我們由視窗類別 (window class) 來開始我們剖析視窗的工作。視窗類別 (window class) 在 Windows 程式設計上扮演著核心的角色，它掌管了視窗的外貌和行爲。在系統中定義類別的過程稱爲 "註册 (registering)"，註册類別時並不會建立視窗，然而，類別一旦被註册後，就可以此來建立無數同類的視窗。

一個程式可以同時註册好幾個類別，而由 Windows 來負責維護這些類別，一直到該程式的所有執行個體 (instance) 結束，或者是類別被註銷掉 (Unregister) 爲止。

1-1 註册一個視窗類別

爲了建立視窗類別，必須先做 WNDCLASS 結構體的初始化，然後再呼叫 RegisterClass 函式加以註册。WNDCLASS 結構體中最重要的項目有下面三項：

- ☐ lpszClassName: 類別的名稱。
- ☐ lpfnWndProc: 指明視窗函式 (window procedure) 的位址。
- ☐ hInstance: 程式或動態聯結程式庫 (DLL) 的代碼 (handle) 或擁有者 (owner)。

至於其他的項目則可用預設的屬性來定義。

當不再需要類別時，必須先將所有屬於該類別的視窗清除，然後呼叫 UnregisterClass 釋放其記憶體，註銷類別。一般程式或 DLL 結束時，不用呼叫 UnregisterClass，因爲 Windows 會自動的註銷所屬之視窗類別。

此为试读,需要完整PDF请访问: www.ertongbook.com