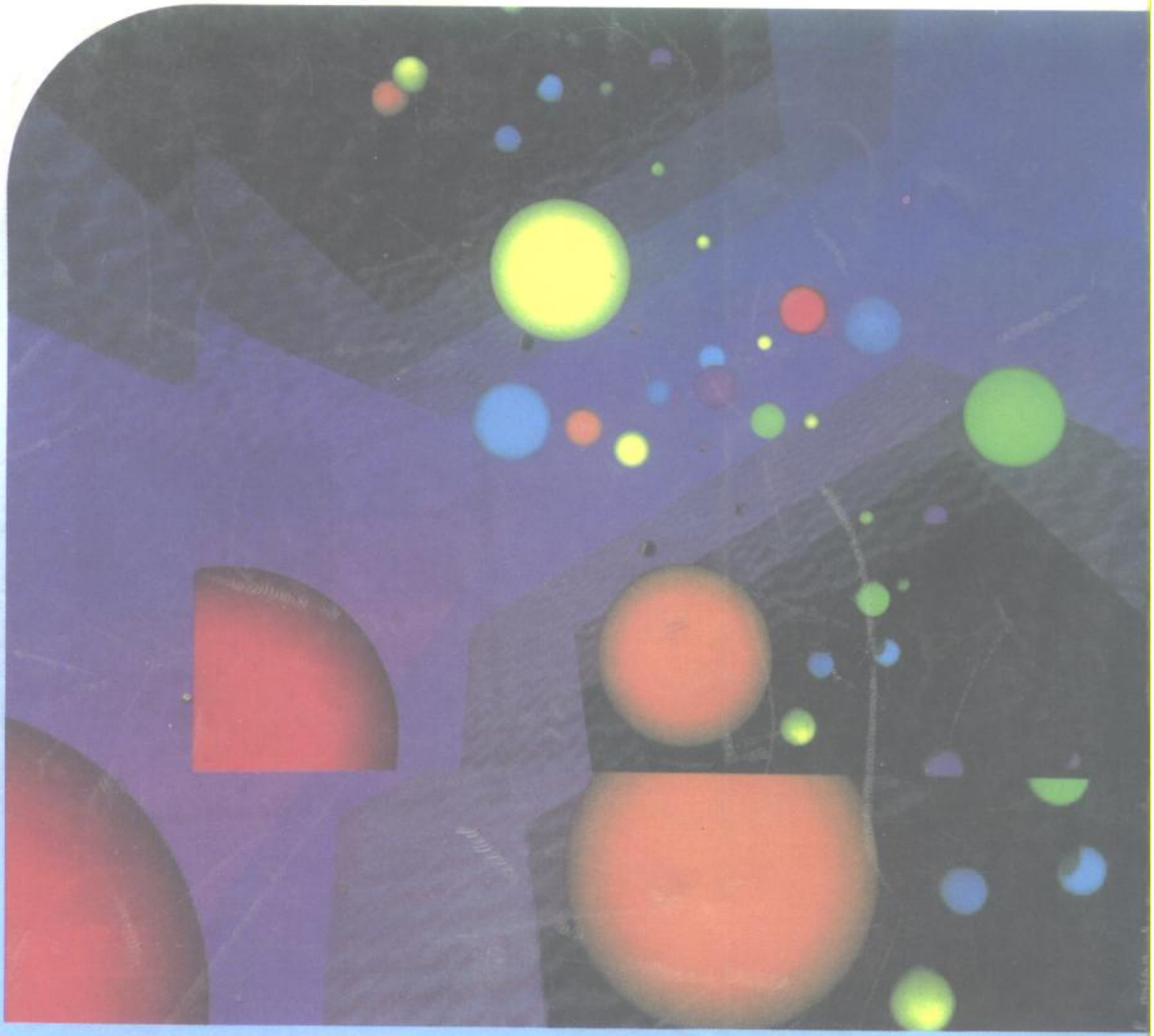
计算机软件应用系列

Visual C++ 程序开发指南（四）

——实用高级编程技术

● 黄 敏 何国辉 等 编著



科学出版社

计算机软件应用系列

Visual C++程序开发指南(四)
——实用高级编程技术

黄 敏 何国辉 等 编著

科学出版社

1995

（京）新登字 092 号

内 容 简 介

Visual C++是目前流行的面向对象的 Windows 程序设计环境，它不仅界面友好、工具繁多，而且类库也很丰富。借助 Visual C++ MFC 类库中的类和 Visual C++的交互式工具，很容易就能生成一个可以实际运行的 Windows 或 DOS 应用程序。

本书不仅讨论了在 Windows 环境下编写 Visual C++程序的各种技术，而且列举了很多可实际运行的实例程序，并对其进行了剖析。

本书可作为大专院校计算机专业学生的教材或参考书，也可供普通 Visual C++和 Windows 程序员参考。

计算机软件应用系列

Visual C++程序开发指南（四）

—— 实用高级编程技术

黄 敏 何国辉 等 编著

责任编辑 刘晓融 留 霞

科学出版社出版

北京东黄城根北街16号

邮政编码：100717

化学工业出版社印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1995 年 6 月第 一 版 开本：787×1092 1/16

1995 年 6 月第一次印刷 印张：27 1/4

印数：1—3 160 字数：635 700

ISBN 7-03-004933-0/TP·483

定价：35.50 元

前　言

面向对象程序设计(OOP)是目前流行的程序语言概念，Visual C++是汇集 Microsoft 公司技术精华的主流产品。《Visual C++程序开发指南》是《计算机软件应用系列》丛书中的一套，共四册。这套书从各个方面说明了应如何在 Visual C++环境下以 OOP 的概念设计 DOS 和 Windows 应用程序。

作为一个成熟的软件工程师，面对铺天盖地的 Word，Excel，PowerPoint，FoxPro，Visual Basic 等应用软件，不熟悉 Visual C++是很不明智的，因为借助 Visual C++，软件工程师就可以开发出很多动态链接库，从而为广大用户提供全方位的支持，其商业前景更是无可估量的。

Visual C++是一个面向对象、界面友好的 DOS 和 Windows 程序开发环境。在这个集成环境中开发应用程序时，不一定要手工设计用户界面，而只需选取菜单命令，Visual C++系统就会生成一个可实际运行的 Windows 应用程序框架。此后，程序员就可利用基于 Windows 的 C++编辑器，借助 AppWizard 建立面向对象的应用程序。

除了 AppWizard 外，Visual C++还包含 C++应用程序生成器、基于 Windows 的编辑器、基于 Windows 的面向图形的编程工具、使 C++代码和 Windows 消息及类成员函数相联系的交互式工具、可用来编写 Visual C++文本程序的 QuickWin 库以及预编译的头文件和源文件。当然，类库丰富更是 Visual C++的最大特色。

尽管目前市面上已有一些 Visual C++方面的书籍，但学习 Visual C++的人常常会有下列困惑：

(1) 对 Visual C++缺乏一致的认识，遇到问题总是缺这少那，缺乏一个值得信赖的良师益友。

(2) 只了解 OOP 的直观语法，而对于 OOP 有何好处、为什么 OOP 可取代结构化语言并成为未来程序设计风格的主流，都不太明白。

(3) 无法理解 OOP 对未来程序的管理与维护究竟有何重要性。

(4) 在 Visual C++环境下设计 OOP 程序时，对 Visual C++提供的资源不太清楚。

(5) 不知道如何用 Visual C++和 Windows 解决实际问题。

针对这些问题，这套书从各个方面介绍了 Visual C++集成环境、OOP 概念、类库以及基于 DOS 和 Windows 的 C++应用程序开发方法，并提供了丰富的类模板和大量的应用程序实例。

本套书的内容基本属于中等难度，适用于初、中级读者。不熟悉 OOP 技术和 C++语言的读者可参阅第一册的第一至十三章，这几章主要介绍 Visual C++集成环境和 OOP 概念。经常从事应用程序开发的读者大都比较关心 Visual C++类库和通用类的构造方法，可参阅第二册和第三册的第七至九章，其中第二册主要介绍通用类的构造方法，并从我们所熟悉的数据结构着手，讨论如何设计和实现自己的通用类，这为进行大程序开发的用户提供了设计开发环境的手段；第三册第七至九章则主要介绍 Visual C++基础类库，讨论如何使用 Visu-

al C++类库。比较熟悉OOP概念而对C++程序设计技术不太精通的读者可参阅第四册，其中包含大量的程序设计实例。

本书是这套书的第四册，参与编写的人员有：黄敏、何国辉、阵伯海、李友清、李杨、魏时志、李蕾、李建、欧阳红、刘崇福、李世玉、愉波、赵大年、何小小、李道雄、陈楚南、梁国、陈佳。全书由吴佳教授和陈忠敏研究员审校。此外，朱小宝为本书的编排付出了辛勤的劳动。在此对以上同志深表感谢。

限于水平和时间，书中的错误和不妥之处，敬请读者批评指正。

目　录

第一篇　Windows 程序设计基础

第二篇 子窗口控件的设计

第三篇 系统资源

第四篇 图形设计技巧

第一篇

Windows 程序设计基础

第一章　Windows 应用程序的结构

设计 Windows 应用程序比较困难的原因是，一个很简单的输出结果却需要很长的程序代码。本章将以一个最基本的 Windows 应用程序为例，深入讲解每一条语句的用法。

由于 Windows 应用程序较为复杂，本章将以 DOS 环境和 Windows 环境为例，一步一步地讲解编译、链接和运行程序的方法。本章将讲解下列内容：

(1) 软件和硬件需求。

(2) Windows 程序的结构。

(3) Windows 应用程序的编译与链接。

(4) 一个简单的 Windows 程序实例。

(5) 图标的设置。

(6) 光标外形的设置。

(7) 客户区的颜色。

自 1990 年 Microsoft 公司推出 Microsoft Windows 3.0 之后，几乎所有 PC 软件厂商都以开发 Windows 环境下的软件为目标。轻击按钮，弹指之间即可协助用户完成工作，的确为用户带来了很大的方便。尽管 Windows 环境易于使用，但如何设计可在 Windows 环境下运行的 Windows 应用程序却一直是程序员的难题，本书的主要目的就是解除此难题，使用户可以花较少的时间学会 Windows 应用程序的设计方法。

1.1　软件和硬件环境要求

设计 Windows 程序的首要条件是，用户应备有 Microsoft Windows 软件。在当前市场上，用户可以使用 Borland 公司推出的 Borland C++（含 Application Framework，简称 AF）或者 Microsoft 公司最新开发的 Visual C++软件来设计 Windows 应用程序。本书的重点是以 Vi-

sual C++为工具来说明 Windows 应用程序的设计方法。

在 Visual C++软件中,用户基本可以采用下面两种方式来设计 Windows 应用程序:

(1) 利用 C 语言并配合应用程序接口函数(Application Programming Interface,简称 API)。

(2) 利用 C++语言并配合 Microsoft Foundation Class(简称 MFC)函数库及 API。

由于 C 语言仍是当前最广泛使用的程序语言,因此本书决定以 C 语言来讲解设计基本 Windows 应用程序的方法。

过去常听人说,要设计 Windows 应用程序,需要有软件开发工具(Software Development Toolkit,简称 SDK),不过在用 Visual C++设计 Windows 应用程序时,就不必考虑这个工具了,因为 SDK 已经完全集成在 Visual C++软件中。

1.2 Windows 环境的主要特点

如果读者是 C 语言高手且精通函数库和 DOS 及 BIOS 调用,则在接触 Windows 应用程序设计之前,可能会认为利用 C 语言设计 Windows 应用程序只是多了一些有关窗口的函数调用罢了。其实并非如此。

在 DOS 环境下,程序员可以独占系统资源(即 CPU,内存,屏幕及键盘等),进行数据的输入/输出并执行一般的运算。利用系统资源和执行一般运算时,大多数情况下只需要调用几个函数。但是在 Windows 环境下,以上情况都变得无效。下面将介绍 Windows 应用程序和 DOS 应用程序开发环境之间的差异。

1.2.1 多任务(Multitasking)特性

Windows 是一个多任务的操作系统,也就是能在同一时间内执行多个应用程序。在这种情况下设计的应用程序无法独占所有系统资源。DOS 只是一个单用户的操作环境,在设计程序时,可以充分使用系统有效资源(avaliable resource)。

例如,对 DOS 应用程序而言,一定可以在屏幕上输出数据,而不可能有其他程序共享屏幕资源。但是在 Windows 环境下,屏幕上可能同时出现多个应用程序的输出窗口。因此,对于任何一个 Windows 应用程序,在输出数据时必须考虑与其他正在运行的应用程序共享屏幕资源。Windows 操作系统必须小心管理所有系统资源,以供所有应用程序分享。所有 Windows 应用程序必须根据 Windows 操作系统特有的界面来执行操作,以确保 Windows 操作系统有效地管理系统资源,保证应用程序的使用。

1.2.2 用户界面

在 DOS 环境下,程序员可以独占屏幕。只要执行几个函数调用,就可以将资源输出到屏幕上,或者接收所输入的数据。

在 Windows 环境下,若想执行输入或输出操作,首先必须在屏幕上设计一个窗口,然后通过此窗口进行输入与输出。当然,一个应用程序可以存在多个应用程序窗口,这样就可以形成多文档界面(Multiple Document Interface)。

1.2.3 数据输入方式

在 DOS 环境下，程序员可以利用 getchar()，scanf()，int86 等函数来接收用户通过键盘或鼠标输入的数据。

在 Windows 环境下，Windows 操作系统将处理用户通过键盘或鼠标输入的所有数据，然后再将接收到的数据送到适当的应用程序消息队列(application message queue，每个应用程序在运行时均产生此队列)，应用程序将自动在所属的应用程序消息队列中读取下一组数据。

1.2.4 数据输出方式

在 DOS 环境下，缺省操作模式为文本模式(Text Mode)。如果用户想输入图形，必须先切换到绘图模式(Graphics Mode)。如果不想输出图形，则必须切换回文本模式。

在 Windows 环境下，绘图模式是基本工作模式，用户可以很顺利地通过此模式进行文本或图形数据的输出。

1.3 Windows 程序的结构

设计 Windows 应用程序比较困难。尽管同样是使用 C 语言来设计程序，但程序结构却有所不同。传统 C 语言的程序结构如下所示：

```
void main()
{
    .
    .
}
```

如果存在从主程序中调用子程序的情况，则调用图 1.1 所示的子程序的基本流程。

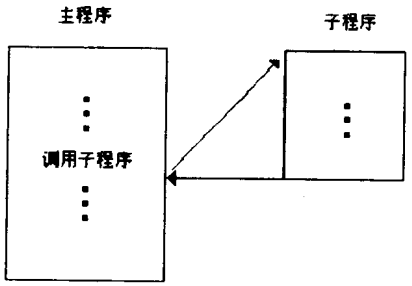
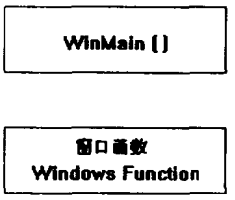


图 1.1 子程序的基本流程

在图 1.1 中，主程序调用子程序，子程序运行完毕后将立即返回主程序的原调用位置，然后继续往下运行。

尽管上述程序的逻辑概念仍然适用于 Windows 应用程序，但 Windows 应用程序还包含了一些不同的逻辑概念。

Windows 应用程序的基本结构如下：



WinMain 和 Windows Function 是构成 Windows 应用程序的两大部分。WinMain 是主程序，也就是程序的入口点(entry point)，而 Windows Function 则是 Windows 应用程序的工作核心。程序员通常利用 WinMain 来执行设计和建立窗口的操作，而利用 Windows Function 来执行应用程序的操作。

WinMain 和 Windows Function 之间的区别在于，尽管从程序员的角度来看，Windows Function 属于 WinMain 的子程序，但是在 WinMain 中找不到任何一个调用 Windows Function 的语句。这时候用户一定会问，在哪一种情况下才执行用户所设计的 Windows Function 呢？

执行 Windows Function 的时机(后面章节将以实例详细说明)很多，重要的是当时机出现时，Windows 操作系统将主动调用 Windows Function。

1.4 编译和链接

对于初学 Windows 应用程序设计的用户，另一个难点是，Windows 应用程序的编译(compile)及链接(link)过程比较复杂。一般 C 语言程序的编译及链接过程如图 1.2 所示。

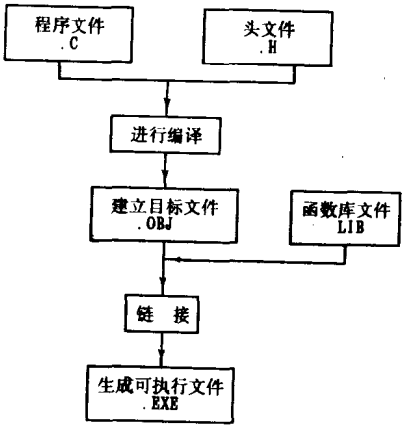


图 1.2 C 语言程序的编译及链接过程

Windows 应用程序的编译及链接过程如图 1.3 所示。

对于 C 语言程序员，通常只要设计 C 语言程序文件即可，至于编译及链接过程，通常可利用 C 语言本身的集成环境方便地完成。Windows 应用程序则比较复杂，其步骤如下：

(1) 建立 C 语言程序文件。

(2) 建立模块定义文件(.def)，以定义程序的名称、属性(可搬移否)、堆栈(stack)和堆(heap)的大小。

(3) 利用资源编辑器 (App Studio) 的编辑程序所需的资源文件，例如光标 (.cur)、图标 (.ico)、位图(.bmp)、对话框(.dia)及菜单等。

(4) 建立资源定义文件(.rc)，定义 Windows 应用程序所需的资源。

(5) 编译C语言程序文件,同时链接(配合模块定义文件和函数库文件)并建立可执行文件。

(6) 编译资源文件,产生资源可利用文件(.res)。

(7) 将资源可利用文件加到C程序语言的可执行文件内,形成Windows应用程序的可执行文件。

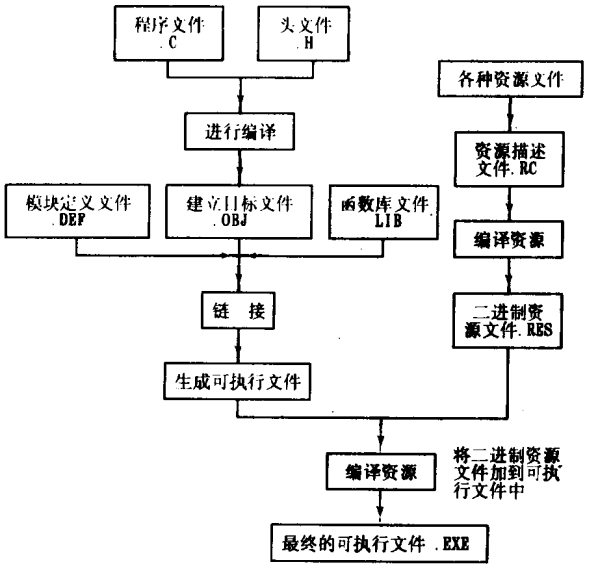


图 1.3 Windows 应用程序的编译及链接过程

以上所述对于初学者来说可能稍感复杂,但是用户不必担心,本书将以实例一步一步讲解,用户只要遵循这些步骤,就一定可以学会建立自己的Windows应用程序。

1.5 Windows资源介绍

常见的C语言数据类型列于表1.1中。

表 1.1 C语言数据类型

数据类型	含 义
int	16位整数
unsigned int	16位无符号整数
long int	32位长整数
unsigned long int	32位符号长整数
char	字符
float	32位浮点数
double	64位双精度浮点数

对于以C语言为基础的Windows应用程序,除了仍然沿用上述数据类型外,又扩充了几种常见的数据类型,见表1.2。

表 1.2 扩充数据类型

数据类型	含 义
WORD	16 位无符号整数
DWORD	32 位无符号整数
LONG	32 位整数
HANDLE	16 位无符号整数，此数据类型常用于表示 Windows 操作系统所建立的某个对象的句柄，使用此句柄相当于引用该对象
HWND	16 位无符号整数，此数据类型常用于表示 Windows 环境内所打开的窗口句柄，使用此句柄相当于引用该窗口
BOOL	16 位布尔值(Boolean value)
BYTE	8 位无符号整数
FARPROC	32 位指向函数的指针
LPSTR	32 位指向字符(字符串)数据的指针
UINT	16 位无符号整数
LPARAM	32 位整数，常用于 Windows 函数的参数
WPARAM	16 位整数，常用于 Windows 函数的参数

注意，如果未特别声明为无符号整数，则以上数据类型都代表符号整数。

以上只是一些常用的数据类型，后面将根据需要陆续配合程序实例介绍其他的数据类型。

1.6 一个简单的例子

本程序将在屏幕上建立宽 300(x 轴)、高 200(y 轴)的窗口，基本单位是像素(pixel)。当光标在窗口内显示为“I”型时，窗口的标题为 ch1_1。

ch1_1.c 程序代码的全部内容如下：

```
01  // ----------------------------
    //  程序名:ch1_1.c
    //  第一个 windows 应用程序
    // ----------------------------
    #include <windows.h>
    long FAR PASCAL WindowFun(HWND,UINT,WPARAM,LPARAM);

    int PASCAL WinMain(HANDLE CurInstance,HANDLE PreInstance,
                       LPSTR CmdParaStr,int ShowStyle)
10  {
      static char ProgName[] = "ch1_1";   // 程序名
      HWND hwnd;                          // 窗口句柄
      MSG  msg;                           // 消息结构
      WNDCLASS wndclass;                  // 窗口类型

    // 注册
    // 通常只有第一次需要注册
      if ( ! PreInstance )
      {
20        wndclass.lpszClassName = ProgName;
          wndclass.hInstance = CurInstance;
          wndclass.lpfnWndProc = WindowFun;
          wndclass.hIcon = LoadIcon(NULL,IDI_APPLICATION);
```

```
    wndclass.hCursor = LoadCursor(NULL,IDC_IBEAM);
    wndclass.lpszMenuName = NULL;
    wndclass.hbrBackground = GetStockObject(WHITE_BRUSH);
    wndclass.style = CS_HREDRAW | CS_VREDRAW;
    wndclass.cbClsExtra = 0;
    wndclass.cbWndExtra = 0;
30
    RegisterClass(&wndclass);                    // 注册
  }

  // 建立窗口
  hwnd = CreateWindow(ProgName,                  // 首先注册名称
                      ProgName,                  // 窗口标题
                      WS_OVERLAPPEDWINDOW,// 窗口类型
                      100,                       // 窗口x位置
                      100,                       // 窗口y位置
40                    300,                       // 窗口x长度
                      200,                       // 窗口y长度
                      NULL,                      // 父窗口的句柄
                      NULL,                      // 窗口菜单句柄
                      CurInstance,               // 当前窗口程序句柄
                      NULL);                     // 所用参数
  // 显示窗口
  ShowWindow(hwnd,ShowStyle);

  // 消息循环
50  while (GetMessage(&msg,NULL,0,0))
  {
    TranslateMessage(&msg);
    DispatchMessage(&msg);
  }
  return msg.wParam;
}
// ------------
//  窗口处理函数
// ------------
60 long FAR PASCAL WindowFun(HWND hwnd,
                             UNIT meessage,
                             WPARAM wParam,
                             LPARAM lParam)
{
  switch(message)
  {
    case WM_DESTROY:             // 释放所建窗口
      PostQuitMessage(0);
      return 0;
70    default:                   // 执行其他情况
      return DefWindowProc(hwnd,message,wParam,lParam);
  }
}
```

ch1_1.def 的内容如下：

```
01 ;----------------
   ; ch1_1.def
   ;----------------
   NAME          ch1_1
   DESCRIPTION   '第一个 Windows 应用程序'
   EXETYPE       WINDOWS
   STUB          'WINSTUB.EXE'
   CODE          PRELOAD MOVEABLE DISCARDABLE
   DATA          PRELOAD MOVEABLE MULTIPLE
```

```
10  HEAPSIZE        1024
    STACKSIZE       4096
```

ch1_1.mak 的内容如下：

```
01  #----------------
    #  ch1_1.mak
    #----------------

    ch1_1.exe :ch1_1.obj ch1_1.def
        link /nod ch1_1,ch1_1,NUL,\
        slibcew libw commdlg,ch1_1
        rc -t ch1_1.exe

10  ch1_1.obj :ch1_1.c
        cl -c -G2sw -Ow -W3 -Zp -Tp ch1_1.c
```

1.6.1 程序的运行

运行 Windows 应用程序有两种方法，一是直接在 DOS 提示符下进行编译与链接，另一种是充分利用 Visual C++集成环境进行编译、链接与运行。

1. 在 DOS 环境下建立和运行程序

要在 DOS 环境下编译、链接与运行程序，首先应对启动盘中的 AUTOEXEC.BAT 文件作如下修改：

(1) 将下面一行加到 PATH 设置内：

```
C:\MSVC\BIN
```

(2) 请加入下列两行数据：

```
SET INCLUDE = C:\MSVC\INCLUDE;C:\MSVC\MFC\INCLUDE
SET LIB=C:\MSVC\LIB;C:\MSVC\MFC\LIB
```

以上假设将 Visual C++安装在 C:\MSVC 目录内。如果将 Visual C++安装在 D 盘的 MSVC 目录中，则只要将驱动器编号 C 改为 D 即可。当然，在上述设置更改完之后，必须重新开机，以上设置才有效。

在含有 ch1_1.c，ch1_1.def 和 ch1_1.mak 的目录下执行下列语句：

```
nmake ch1_1.mak
```

如果一切顺利，用户将看到下列编译及链接过程：

```
Microsoft (R) Program Maintenance Utility Version 1.30
Copyright (c) Microsoft Corp 1988-93. All rights reserved.

        cl -c -G2sw -Ow -W3 -Zp -Tp ch1_1.c
Microsoft (R) C/C++ Optimizing Compiler Version 8.00
Copyright (c) Microsoft Corp 1984-93. All rights reserved.

ch1_1.c
        link/nod ch1-1, ch1_1, NUL, slibcew libw commdlg, ch1_1

Microsoft (R) Segmented Executable Linker Version 5.50
Copyright (C) Microsoft Corp 1984-93. All rights reserved.

        rc -t ch1_1.exe
Microsoft (R) Windows Resource Compiler Version 3.31
Copyright (C) Microsoft Corp 1985-1992. All rights reserved.
```

见到上述运行结果后，表示已在 DOS 环境下成功地建立了某个 Windows 应用程序的可