面向 21 世纪高等院校计算机教材系列

# Visual C++程序设计教程

黄维通　姚瑞霞　编著

机 械 工 业 出 版 社

本书从最基本的概念出发，详细讲述了 V C++的开发过程，内容涉及可视化编程过程中常用的 API 函数及 MFC 类库。由于 MFC 类库中封装了大量的 API 函数，通过把 API 函数作为专题讲解，使难点分散，有利于读者循序渐进地学习。

本书既可以作为高等学校计算机软件技术课程的教材，也适于有关科研及应用开发人员作为参考，同时也可供从事计算机软件开发的专业人员使用。

**图书在版编目（CIP）数据**

Visual C++程序设计教程/黄维通，姚瑞霞编著.
—北京：机械工业出版社，2001.7
面向 21 世纪高等院校计算机教材系列
ISBN 7-111-08470-5

Ⅰ. V... Ⅱ. ①黄... ②姚... Ⅲ. C 语言-程序设计-高等学校-教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2001）第 045692 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策 划：胡毓坚
责任编辑：田 梅
责任印制：

·新华书店北京发行所发行

2001 年 7 月第 1 版·第 1 次印刷
787mm×1092mm 1/16·20.75 印张·513 千字
0001—5000 册
定价：29.00 元

# 出 版 说 明

随着计算机技术的飞速发展，计算机在经济与社会发展中的地位日益重要。在高等院校的培养目标中，都将计算机知识与应用能力作为其重要的组成部分。为此，国家教育部根据高等院校非计算机专业的计算机培养目标，提出了“计算机文化基础”、“计算机技术基础”和“计算机应用基础”三个层次教育的课程体系。根据计算机科学发展迅速的学科特点，计算机教育应面向社会，面向潮流，与社会接轨，与时代同行。随着计算机软硬件的不断更新换代，计算机教学内容也必须随之不断更新。

为满足高等院校计算机教材的需求，机械工业出版社聘请了清华大学、北方交通大学、北京邮电大学等院校的老师，经过反复研讨，结合当前计算机发展需要和编者长期从事计算机教学的经验精心编写了“面向 21 世纪高等院校计算机教材”系列丛书。

本套教材以理论教学和实践教学相结合，图文并茂，内容实用、层次分明、讲解清晰、系统全面，其中溶入了老师大量的教学经验，是各类高等院校、高等职业学校及相关院校的最佳教材，也可作为培训班和自学使用。

# 前　言

目前，面向过程的 C 语言已经成为高校理工科学生的必修或选修课程，但随着软件工程技术的不断发展，应用面向对象的编程技术已经成为当今软件开发的重要手段之一，尤其是 Visual C++（简称 V C++）的出现，大大推进了面向对象与可视化编程技术的应用与发展。因此，掌握“面向对象与可视化程序设计”的思想与方法已经成为对大学生应用与开发能力的要求之一。

本书分 API 专题和 MFC 专题对 V C++编程方法及其体系结构进行介绍。在 API 专题中，通过具体的实例介绍消息响应的机制及其对消息的响应，内容包括读者在开发应用程序中将经常接触到的包括 Windows 绘图、文本输入/输出、资源的应用及一系列标准控件的使用等知识；在 MFC 专题部分主要介绍应用 MFC 进行可视化编程的思想方法，包括各种类在编程中的应用等知识点。本书作为面向 21 世纪高等院校计算机软件技术的教材，体现了计算机软件技术课程改革的方向之一。本课程建议授课学时为 32 小时，并要求先修 C 语言课程。

本书特点是从面向对象的基本概念出发，讲述可视化程序设计的思想与方法。对每一部分的知识点、概念、难点，都力求以较精炼的语言进行讲解，同时，对每一个知识点都配以必要的实例，实例中配以较为详细的步骤说明及语法说明（因此本书也适合自学），力求通过实例让读者全面掌握“面向对象与可视化程序设计”的思路和开发技巧与体系，本书的例子都是根据教学特点精心编排，且所有的例题都在 Windows 98 及 V C++6.0 的环境下调试运行通过的。

参加本书编写、程序调试工作的同志还有刘嘉、程楠、张伟浩、马骏锋、李阳、胡定涵、张国华、王增宏、杜春青、李静等，在此表示衷心感谢！同时感谢本书所列参考文献的作者！

由于作者水平有限，缺点和错误在所难免，恳请读者批评指正。

本书配有授课用电子讲稿，欢迎通过电子邮件（hwt@cic.tsinghua.edu.cn）联系。

感谢阅读本书的读者！

作　者

# 目　　录

# 第 1 章　Windows 应用程序

## 1.1　Windows 编程基础知识

Microsoft Windows 是一个应用于微型计算机上的图形化用户界面的操作系统，它为应用程序提供了一个由一致的窗口和菜单结构构成的多任务环境。

目前基于 Windows 风格的应用软件开发平台大多是“可视(Visual)”的，它通常集成了一系列系统可用的资源和开发工具。这些工具包括：

- 源程序编辑器和编译器；
- 可进行源程序语法检查和运行的程序调试工具；
- 系统函数库和系统函数开发工具；
- 资源管理器，包括图形化窗口及组成元素的多种对象(如菜单、图标、光标等)的编辑器；
- 例程库及例程的 Help；
- 应用程序的 Help 和 Setup 开发工具包；
- 其他功能。

编写 Windows 风格的应用程序，可以使用 Visual C、Visual Basic、Visual Java 等面向对象(Object Oriented)的程序开发语言，在 Windows 编程中，“对象(Object)”是指 Windows 的规范部件，包括各种窗口、对话框、菜单、按钮、工具栏及程序模块等等。这些多样化的“对象”能够充分满足构成应用软件操作界面的需要，因此编写 Windows 程序相当一部分工作是在创建对象和为对象属性赋值。对象既具有规范形态，又具有规范操作模式。用户可以采用传统的源程序代码编写方法编写程序，也可以采用 Windows 特色的交互式操作方法构造程序。采用交互式方法时，可视化开发平台给出了许多选用的对象，程序员可以选择所需要对象并为对象的属性确定参数值，由此搭建起应用程序的“大框架”，在这个“大框架”中，程序员根据需要进一步编写必要的细节代码段，最后构成完整的应用程序。

在用 Visual C++(以下简称 V C++)开发面向对象应用程序时，主要使用了两种方法，一种是使用 Windows 提供的 Windows API 函数，另一种是直接使用 Microsoft 提供的 MFC 类库。

API 是应用程序编程接口(Application Programming Interface)的缩写，它是 Windows 系统和 Windows 应用程序间的标准程序接口。API 为应用程序提供系统的各种特殊函数及数据结构定义，Windows 应用程序可以利用上千个标准的 API 函数调用系统功能。

根据 Windows API 函数完成的功能，可将其分为三类：第一是窗口管理函数，它主要实现窗口的创建、移动和修改功能；第二是图形设备(GDI)函数，它主要实现与设备无关的图形操作功能；第三是系统服务函数，它主要实现与操作系统有关的多种功能。

到目前为止，Microsoft 已经发布了几个 Windows API 版本，其中 Win32 用于Windows NT，Windows 98 的 API 接近于 Win32。

在 MFC 类库中，集成了大量已经预先定义好的类，用户可以根据编程的需要，调用相应的类，或根据需要自定义有关的类。由于 MFC 类的定义中封装了大量的 API 函数，为使读者能够全面掌握 V C++编程方法，本书将重点讲述 API 函数及 MFC 类库的应用，并通过一系列实例来加深对基本概念和基本方法的理解。

利用 Windows API 函数和 MFC 类库编写 Windows 应用程序必须首先了解有关面向对象编程的特点及相关概念，如窗口的概念、事件驱动的概念、消息及其在编程中的应用、句柄的概念和匈牙利表示法等。

### 1.1.1 窗口的概念

窗口是 Windows 应用程序基本的操作单元，是系统管理应用程序的基本单位，是应用程序与用户之间交互的接口环境。应用程序的运行过程即是窗口内部、窗口与窗口之间、窗口与系统之间进行数据处理与数据交换的过程。因此，编写 Windows 应用程序首先应创建一个或多个窗口。

一个应用程序的窗口一般包含窗口边界、工作区、控制菜单框、控制菜单、水平和垂直滚动条以及最大化按钮、最小化按钮和标题栏等对象。

### 1.1.2 事件驱动的概念

Windows 程序设计围绕着事件或消息的产生驱动运行消息处理函数。所谓消息是描述事件发生的信息。例如按下键盘上的某一个键时，系统就会生产一条特定的消息，标识按键事件的发生，这里的事件包含按下键的消息、字符消息和键的弹起消息。Windows 程序的执行顺序取决于事件发生的顺序，程序的执行顺序是由顺序产生的消息驱动的，但是消息的产生往往并不要求有次序之分。程序员可以针对消息类型编写消息处理程序以处理接收的消息，或者发出其他消息以驱动其他处理程序，但是不必预先确定消息产生的次序。这是面向对象编程的最显著的特点，也是与传统的面向过程编程方法的重要区别之一。

事件驱动编程方法对于编写交互式程序很有用处，用这一方法编写的程序使程序避免了死板的操作模式，从而使用户能够按照自己的意愿采用灵活多变的操作形式。

### 1.1.3 句柄

句柄(handle)是 Windows 编程的基础，它是一个 4 字节长的整数值，用于标识应用程序中不同的对象和同类对象中不同的实例，诸如一个窗口、按钮、图标、滚动条、输出设备、控制或文件等对象，都需要一个唯一的句柄来标识，应用程序通过句柄来访问相应的对象信息。

在 Windows 应用程序中，句柄的使用是很频繁的，表 1-1 是部分常用句柄类型及其说明。

表 1-1 常用句柄类型及其说明

| 句柄类型 | 说明 | 句柄类型 | 说明 |
|---|---|---|---|
| HWND | 标识窗口句柄 | HDC | 标识设备环境句柄 |
| HINSTANCE | 标识当前实例句柄 | HBITMAP | 标识位图句柄 |
| HCURSOR | 标识光标句柄 | HICON | 标识图标句柄 |

（续）

| 句柄类型 | 说　明 | 句柄类型 | 说　明 |
|---|---|---|---|
| HFONT | 标识字体句柄 | HMENU | 标识菜单句柄 |
| HPEN | 标识画笔句柄 | HFILE | 标识文件句柄 |
| HBRUSH | 标识画刷句柄 | | |

Windows 应用程序利用 Windows 消息(Message)与其他的 Windows 应用程序及 Windows 系统进行信息交换。由于 Windows 应用程序是消息或事件驱动的，因此，Windows 消息的工作机制就显得很重要了。Windows 中消息由三部分组成：消息号、字参数和长字参数。消息号由事先定义好的消息名标识；字参数(wParam)和长字参数(lParam)用于提供消息的附加信息，附加信息的含义与具体消息号的值有关。在 Windows 中消息往往用一个结构体 MSG 来表示，结构体 MSG 的定义如下：

```
typedef struct tagMSG
{    HWND    hwnd;
     UINT    message;
     WPARAM # wParam;
     LPARAM # lParam;
     DWORD   time;
     POINT   pt;
} MSG;
```

其中：

- hwnd 是获取消息的窗口句柄，若此参数为 null，则可检索所有驻留在消息队列中的消息。
- message 是代表一个消息的消息编号，每个 Windows 消息都有一个消息编号。
- wParam 和 lParam 是包含有关消息的附加信息，它随不同的消息而有所不同。
- time 指定消息发送至消息队列的时间。
- pt 指定消息发送时，屏幕光标的位置。pt 的数据类型 POINT 也是一个结构体，POINT 的定义如下：

```
typedef struct tagPOINT
                {    LONG x;      // x 和 y 分别表示屏幕的横坐标和纵坐标
                     LONG y;
                }POINT;
```

### 1.1.4　消息及其在编程中的应用

在面向对象的编程中，重点之一就是编写对消息的响应代码。

**1. 消息**

V C++中存在几种系统定义的消息分类，常用的消息有窗口管理消息、初始化消息、输入

消息、系统消息、剪贴板消息、控制处理消息、控制通知消息、滚动条通知消息、非用户区消息、文档界面消息、DDE(动态数据交换)消息、应用程序自定义的消息等。为了能够很好地识别不同的消息类型,V C++定义了不同的前缀符号以用于消息宏识别消息类属,系统定义的消息宏前缀如下:

- BM 表示按钮控制消息;
- CB 表示组合框控制消息;
- DM 表示默认下压式按钮控制消息;
- EM 表示编辑控制消息;
- LB 表示列表框控制消息;
- SBM 表示滚动条控制消息;
- WM 表示窗口消息。

应用程序自定义的消息可以供内部应用程序和系统内其他进程通信使用。不同类型的 Windows 消息的取值范围见表 1-2。

**表 1-2 不同 Windows 消息类型的取值范围**

| 消 息 类 型 | 取 值 范 围 |
|---|---|
| 系统定义消息(部分 I) | 0x0000～0x03FF |
| 用户定义内部消息 | 0x0400～0x07FF |
| 系统定义消息(部分 II) | 0x8000～0xBFFF |
| 用户定义外部消息 | 0xC000～0xFFFF |

**2. Windows 应用程序常用消息**

(1)WM _ LBUTTONDOWN

它产生单击鼠标左键的消息,其附加消息参数 wParam 包含一个整数值,标识鼠标键的按下状态。数值常数名称及其说明见表 1-3。

**表 1-3 鼠标键状态参数及其说明**

| 鼠标键状态参数 | 说 明 |
|---|---|
| MK _ LBUTTON | 标识按下鼠标左键 |
| MK _ MBUTTON | 标识按下鼠标中键 |
| MK _ RBUTTON | 标识按下鼠标右键 |

长字参数 1Param 的低字节包含当前光标的 X 坐标,高字节包含当前光标的 Y 坐标。

此外,相似的消息还有:

- WM _ LBUTTONUP:放开鼠标左键时产生;
- WM _ RBUTTONDOWN:单击鼠标右键时产生;
- WM _ RBUTTONUP:放开鼠标右键时产生;
- WM _ LBUTTONDBLCLK:双击鼠标左键时产生;
- WM _ RBUTTONDBLCLK:双击鼠标右键时产生。

(2) WM _ KEYDOWN

按下一个非系统键时将产生此消息。系统键是指实现系统操作的组合键,例如 Alt 键与某个功能键的组合以实现系统菜单操作等。其附加消息参数 wParam 为按下键的虚拟键码,虚拟键码用以标识按下或释放的键,例如功能键 F1 的虚拟键码在 V C++编程中定义为 VK _ F1,lParam 记录了按键的重复次数、扫描码、转移代码、先前键的状态等信息。此外相似的消息还有 WM _ KEYUP,它在放开非系统键时产生。

(3) WM _ CHAR

这也是按下一个非系统键时产生的消息。附加消息参数 wParam 为按键的 ASCII 码,1Param 与 WM _ KEYDOWN 的相同。

(4) WM _ CREATE

此消息是由 CreateWindow 函数发出的消息。附加消息参数 wParam 未用,1Param 包含一个指向 CREATESTRUCT 数据结构的指针,该结构是传递给 CreateWindow 函数的参数的副本。

(5) WM _ CLOSE

这是关闭窗口时产生的消息,如果有子窗口存在,也一起删除。附加信息参数 wParam 和 1Param 均未用。

(6) WM _ DESTROY

这是消除窗口时由 DestroyWiodow 函数发出的消息。wParam 和 1Param 均未用。

(7) WM _ QUIT

这是退出应用程序时由 PostQuitMessage 函数发出的消息。附加信息参数 wParam 含有退出代码,退出代码标识应用程序退出运行时的有关信息;1Param 未用。

(8) WM _ PAINT

当发生用户区移动或显示事件、用户窗口改变大小的事件以及程序通过滚动条滚动窗口时,均产生一条 WM _ PAINT 消息,此外,当需要恢复被覆盖的区域如下拉菜单关闭后需要恢复被覆盖的部分以及关闭对话框或消息框等对象并需要恢复被覆盖的区域时,均产生 WM _ PAINT 消息。

(9) WM _ COMMAND

当选择某个菜单项或控件向父窗口发送消息及翻译加速键时,均发送 WM _ COMMAND 消息。如果消息来自控件,wParam 的高字节为控件标识码;如果来自加速键,wParam 的值为 1;如果来自菜单,wParam 的值为 0。

## 1.2 Windows 中的事件驱动程序设计

大家所熟悉的 Turbo C 语言就是一种基于 DOS 的应用程序开发工具,其主要特点是面向过程的,主要使用的是顺序的、过程驱动的程序设计方法。这种程序设计方法有一个明显的开始、明显的过程和一个明显的结束,因此,程序能直接控制事件或过程的顺序。

基于 Windows 的应用程序设计方法与 DOS 程序设计方法的不同就在于 Windows 程序是事件驱动的。事件驱动的程序的特点是程序的运行不由程序的顺序来控制,而是由事件的发生顺序来控制。

假设有一个应用程序，该程序的功能是实现某一个产品的第一季度的平均销售额。在一个顺序的、过程驱动的程序中，可以设想用下面的步骤来实现该应用程序要求实现的功能：

（1）输入产品名称；

（2）输入一月份的销售额；

（3）输入二月份的销售额；

（4）输入三月份的销售额；

（5）计算并显示平均值。

其逻辑流程图可以用图 1-1 来表示。

这种设计是基于过程驱动的，它使用户只能按照程序规定好的步骤进行操作，用户不能以任意的顺序跳越性地输入数据。尽管在顺序的、过程驱动的程序中也有许多处理异常的方法，但是，这些异常处理也是顺序的、过程驱动的结构。

而事件驱动程序设计是围绕着消息的产生与处理而展开的，而消息是不会以任何预定义顺序出现的，因此，编写 Windows 应用程序，很大一部分工作就是编写消息的接收与发送的响应代码。图 1-2 是基于事件驱动的计算一季度销售平均值的程序流程示意图。

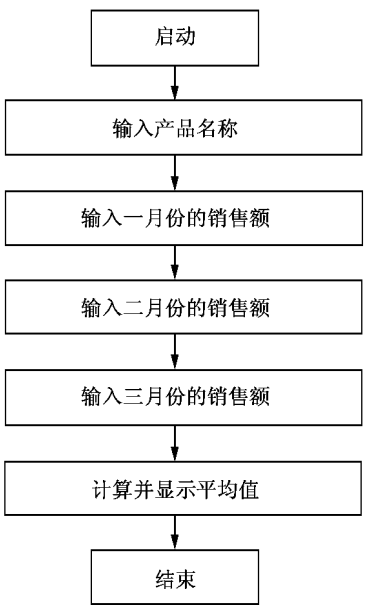


图 1-1　用过程驱动的方法来计算平均销售额

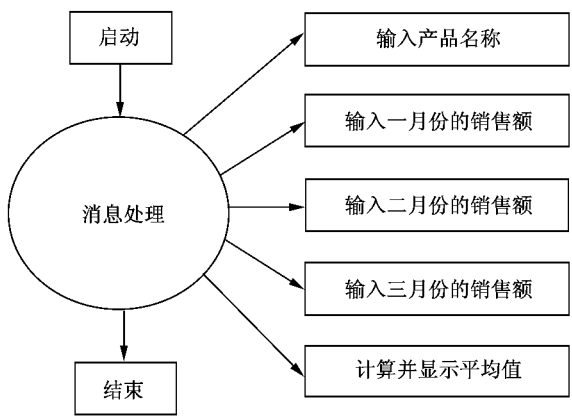


图 1-2　用事件驱动的方法来计算平均销售额

按图 1-2 流程所设计的基于事件驱动的程序所实现的功能和用图 1-1 中的流程所设计的基于过程的程序所实现的功能是相同的。但是，基于事件驱动的程序，在输入数据时，用户可以在不同的窗口中来回切换，并不需要按顺序按步骤地输入数据。例如，可以先输入三月份的销售额，然后再输入一月份的销售额，而不必以一、二、三月份的顺序输入数据。

事件驱动程序方法提供了许多的便利，对于那些需要大范围的用户干预的应用程序来说，更显其优超性。

## 1.3 Windows 应用程序的基本组成

下面通过一个窗口实例程序说明 Windows 应用程序的基本组成。一个基本的 Windows 应用程序是一个能够改变大小,能够移动、关闭、最大化、最小化的窗口。

### 1.3.1 应用程序的组成

一个完整的 Windows 应用程序通常由五种类型的文件组成,它们分别是 C 语言源程序文件(后缀为. C 或. CPP)、头文件(后缀为. H)、模块定义文件(. DEF)、资源描述文件(. RC)和工程文件(. MAK)。头文件包含源程序文件需要的外部常量、变量、数据结构和函数定义和说明;模块定义文件定义程序模块的属性,在 Windows 98/2000 中,该文件可省略;资源描述文件定义源程序使用的资源;工程文件包含各种源程序文件编译后生成的文件,经进一步编译成为可执行文件。

### 1.3.2 源程序组成结构

Windows 的应用程序具有相对固定的基本结构,其中由 WinMain 函数、窗口函数构成基本框架并包含各种数据类型、数据结构与函数等。入口函数 WinMain 和窗口函数是 Windows 应用程序的主体。

**1. WinMain 函数**

WinMain 函数是所有 Windows 应用程序的入口,类似 C 语言中的 Main 函数,其功能是完成一系列的定义和初始化工作,并产生消息循环。消息循环是整个程序运行的核心。

WinMain 函数实现以下功能:

- 注册窗口类,建立窗口及执行其他必要的初始化工作;
- 进入消息循环,根据从应用程序消息队列接受的消息,调用相应的处理过程;
- 当消息循环检索到 WM _ QUIT 消息时终止程序运行。

WinMain 函数有三个基本的组成部分:函数说明、初始化和消息循环。

(1) 函数说明

WinMain 函数的说明如下:

```
int WINAPI WinMain
    (     HINSTANCE hThisInst,      //应用程序当前实例句柄
          HINSTANCE hPrevInst,      //应用程序其他实例句柄
          LPSTR lpszCmdLine,        //指向程序命令行参数的指针
          Int nCmdShow              //应用程序开始执行时窗口显示方式的整数值标识
    )
```

由于 Windows 操作系统是多任务的操作系统,能进行多任务管理,Windows 应用程序可能被并行地多次执行,因而可能出现同一个应用程序的多个窗口同时存在的情况,Windows系统将应用程序每一次的执行称为该应用程序的一个实例(instance),并使用一个实例句柄来唯一地标识它。

(2)初始化

初始化包括窗口类的定义、注册、创建窗口实例和显示窗口四部分。

1)窗口类定义

在 Windows 应用程序中,窗口类定义了窗口的形式与功能。窗口类定义通过给窗口类数据结构 WNDCLASS 赋值完成,该数据结构中包含窗口类的各种属性,在窗口类定义过程中常用到以下函数:

- LoadCursor 函数,其作用是在应用程序中加载一个窗口光标。

  LoadCursor 函数的原型为:

```
HCURSOR LoadCursor
    (HINSTANCE hInstance,   //光标资源所在的模块句柄,为 NULL 则使用系统预定义光标
     LPCTSTR lpCursorName   //光标资源名或系统预定义光标标识名
    )
```

- LoadIcon 函数,其作用是在应用程序中加载一个窗口图标。

  LoadIcon 函数的原型为:

```
HICON LoadIcon
    (HINSTANCE hInstance,   //图标资源所在的模块句柄,为 NULL 则使用系统预定义图标
     LPCTSTR lpIconName     //图标资源名或系统预定义图标标识名
    )
```

- GetStockObject 函数,其作用是获取已经定义的画笔、画刷、字体等对象的句柄

  GetStockObject 函数的原型为:

```
HGDIOBJ   GetStockObject(int fnObject);      //fnObject 为对象的标识名
```

2)注册窗口类

Windows 系统本身提供部分预定义的窗口类,程序员也可以自定义窗口类,窗口类必须先注册后使用。窗口类的注册由注册函数 RegisterClass()实现。其形式为:

```
RegisterClass(&wndclass);                    //wndclass 为窗口类结构
```

RegisterClass 函数的返回为布尔值,注册成功则返回真

3) 创建窗口示例

创建一个窗口类的实例由函数 CreateWindow()实现,该函数的原型为:

```
HWND Create Window
            (LPCTSTR lpszClassName,          //窗口类名
             LPCTSTR lpszTitle,              //窗口标题名
             DWORD dwStyle,                  //创建窗口的样式,常用样式见表 1-4
             int X,                          //窗口左上角 X 坐标
             int Y,                          //窗口左上角 Y 坐标
             int nWidth,                     //窗口宽度
             int nHeight,                    //窗口高度
             HWND hwndParent,                //该窗口的父窗口句柄
```