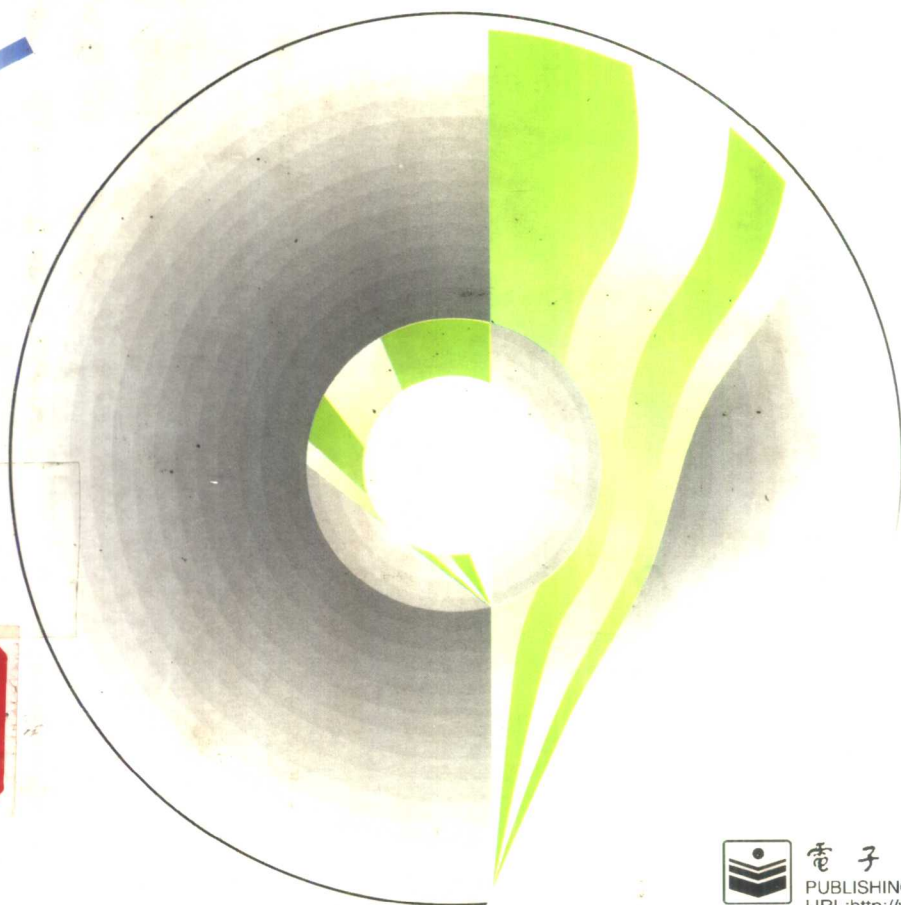


多媒体计算机系列丛书

Visual C++ 5.0

多媒体编程 技术与实例

周长发 周建欣 编著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL: <http://www.phei.com.cn>

多媒体计算机系列丛书

Visual C + + 5.0 多媒体编程技术与实例

周长发 周建欣 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书在概述 Visual C++ 5.0 编程精要和简要介绍多媒体技术与 Windows 95 多媒体体系结构的基础上,全面阐述了面向对象的多媒体程序设计方法,包括多媒体资源交换文件结构与多媒体文件输入/输出、Windows MCI 多媒体编程、利用 MCIWnd 窗口类组织多媒体、音频函数与音效处理、图像编程和动画编程技术等内容。

作者以丰富的多媒体编程实践经验为背景,详细论述了基于类设计的 CD 音频、波形音频、MIDI、AVI 数字视频、特殊音效处理、位图图像和动画的面向对象编程方法。本书不仅结合大量实例来论述每一论题,并在最后一章中以多媒体五子棋游戏为例介绍了编制多媒体游戏的过程和方法。在阅读本书的过程中,读者将经历一次次的多媒体编程旅行,真正体会多媒体编程的乐趣。

本书适合于多媒体程序开发人员、大专院校计算机及相关专业的师生参考阅读。

图书在版编目(CIP)数据

Visual C++ 5.0 多媒体编程技术与实例/周长发,周建欣编著. - 北京:电子工业出版社, 1999.2

(多媒体计算机系列丛书)

ISBN 7-5053-5138-9

I. V… II. ①周 ②周… III. C 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(98)第 40645 号

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,翻版必究。

丛 书 名:多媒体计算机系列丛书

书 名:Visual C++ 5.0 多媒体编程技术与实例

编 著 者:周长发 周建欣

责任编辑:徐 堃

特约编辑:丛 山

排版制作:电子工业出版社计算机排版室

印 刷 者:北京李史山胶印厂

出版发行:电子工业出版社 URL: <http://www.phei.com.cn>

北京市海淀区万寿路 173 信箱 邮编 100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:16.5 字数:416 千字 附磁盘:1 张

版 次:1999 年 3 月第 1 版 1999 年 3 月第 1 次印刷

书 号:ISBN 7-5053-5138-9
TP·2567

定 价:30.00 元

凡购买电子工业出版社的图书,如有缺页、倒页、脱页、所附磁盘或光盘有问题者,请向购买书店调换。

若书店售缺,请与本社发行部联系调换。电话 68279077

前 言

多媒体现已成为计算机的主要表现方式,大部分使用过计算机的人都体会过多媒体程序的丰富表现力。如果你想探究那些令人眼花缭乱、目不暇接的声音、图像、视频、动画等多媒体效果是怎样开发的,那么你一定也会对本书的内容感兴趣。本书的目的就是带领你去亲自尝试多媒体开发,在你自己的程序中增加令人羡慕的多媒体效果,让你亲身体会多媒体编程的乐趣。

Windows 95 是目前 PC 机主要的多媒体操作系统,而 Visual C++ 5.0 是进行 Windows 95 程序设计的主流编程语言。这是本书选取 Visual C++ 5.0 作为开发工具的主要原因。Windows 95 具有完善的多媒体体系结构,其 SDK 提供了对所有多媒体细节控制的编程接口,这是现代能生产出数量庞大的多媒体软件的基础。Visual C++ 涵盖 Windows SDK 中的所有 API,而且功能强大、界面友好,使用它可以实现 Windows 能实现的任何功能。

作为主要的面向对象程序设计语言,C++ 具有使用简便、扩展灵活、代码重用性好等重要优点。Visual C++ 是一个优秀的 C++ 版本。这是本书采用 Visual C++ 5.0 为开发工具的另一原因。Visual C++ 能自动生成应用程序框架,提供标准化的程序结构和用户接口,其中定义的 MFC 类库功能强大,封装了 Windows SDK 中的几乎所有函数,能实现 Windows 系统的任何功能,提供了应用程序本身的数据和操作、ActiveX、OLE、Internet、WinSock、DAO、ODBC 等操作类,生成的程序代码短、运行速度快,具有很大的灵活性。

本书不是 Visual C++ 和 Windows 编程的教科书,因此要求读者具备一定的用 Visual C++ 进行 Windows 编程的基础。阅读本书后,读者将不仅能了解多媒体编程的奥秘,而且对 Visual C++ 和 Windows 编程也将产生更加深入的认识。

全书分为九章。第一章概括了 Visual C++ 面向对象程序设计的要点和重要特征,介绍了几个常用的编程技巧,并一步步地引导读者实现了一个基于文档-视结构、使用自己设计的类的 MDI 程序实例。第二章阐述了多媒体的基本概念。第三章介绍了 Windows 的多媒体资源交换文件结构与多媒体文件输入/输出函数,说明了 RIFF、WAVE 和 AVI 文件的基本结构以及利用多媒体文件输入/输出函数进行多媒体文件操作的方法。第四章详细论述了 Windows MCI 的结构和编程方法,并设计了一个具有实用性的 CMCIClass 类,用以方便地实现 MCI 多媒体编程,并用 CMCIClass 类实现了一个媒体播放器。第五章以大量示例介绍了 MCIWnd 窗口类进行多媒体,特别是数字视频编程的方法。第六章先说明了高、低层音频函数的编程方法,然后讨论了特殊波形音频效果,包括反转、滤波、淡入、淡出等的算法和代码,并实现了一个能进行波形音频播放、录音、文件读写、音效编辑的波形音频管理器实例。第七章介绍了基于 Windows 的图像编程技术,包括位图结构、调色板处理等,设计了一个功能强大、使用方便的处理与设备无关位图的 CDib 类,并以该类为基础编制了一个 DIB 观察器。第八章详细讨论了图形、调色板、位图等不同的动画结构,并结合实例论述了不同动画的编程方法。第九章综合前面几章的内容,详细说明了设计一个多媒体五子棋游戏的技术和方法。

本书所附的所有例程都用 Visual C++ 5.0 在 Windows 95 中文 OSR2 版中调试实现。在

阅读本书时,读者最好能准备好一台计算机,以便能随时尝试本书提供的大量例程。最好在具有如下基本配置的计算机中编译或使用本书所附的例程:

- ✓ CPU:奔腾 166 或更好。
- ✓ 内存:32MB 或更多。
- ✓ 硬盘:2.1GB 或更大。
- ✓ SVGA 或更好的显示卡。
- ✓ 符合 MPC 标准的声卡和音箱。
- ✓ 符合 MPC 标准的 CD-ROM 驱动器。
- ✓ 操作系统:Windows 95(OSR2)或更高版。
- ✓ 开发工具:Visual C++ 5.0。

本书是以作者的经验体会有感而做,限于笔者的学识能力,错误、浅陋和陈旧之处在所难免,恳请读者批评指正。

最后感谢所有对本书的写作和出版有过帮助的朋友们。

作者

一九九八年六月十二日

目 录

第一章 Visual C++ 5.0 编程技术	(1)
1.1 OOP 与 Visual C++ 特性	(1)
1.1.1 Win 32 编程	(2)
1.1.2 框架和文档-视结构	(3)
1.1.3 消息映射	(5)
1.1.4 Visual C++ 可视化编程	(7)
1.2 中文程序开发环境的安装	(8)
1.3 编程风格问题	(8)
1.4 几个编程技巧	(10)
1.4.1 改变框架生成窗口的缺省风格	(10)
1.4.2 创建不规则形状窗口	(12)
1.4.3 用鼠标单击窗口标题条以外区域移动窗口	(16)
1.4.4 使用上下文菜单	(17)
1.4.5 使应用程序只能运行一个实例	(18)
1.4.6 使应用程序显示为任务条通知区中的图标	(21)
1.5 一个完整的 Visual C++ 程序剖析	(26)
第二章 多媒体基础	(37)
2.1 多媒体的基本概念	(37)
2.2 Windows 95 多媒体体系结构	(39)
2.3 MPC 标准与多媒体开发环境	(42)
2.4 多媒体程序开发涉及的主要媒体	(43)
2.4.1 数字化波形音频	(44)
2.4.2 MIDI	(45)
2.4.3 CD 音频	(46)
2.4.4 图像	(46)
2.4.5 数字视频	(47)
2.5 多媒体程序开发流程	(48)
第三章 RIFF 文件结构与多媒体文件输入/输出函数	(50)
3.1 RIFF 文件结构	(50)
3.2 WAVE 文件结构	(50)
3.3 AVI 文件结构	(51)
3.4 多媒体文件 I/O 函数	(54)
3.4.1 基本多媒体文件输入/输出	(55)
3.4.2 具缓冲区的多媒体文件输入/输出	(57)

3.4.3 RIFF 文件操作	(59)
第四章 MCI 编程	(65)
4.1 MCI 设备	(65)
4.2 MCI 函数与命令	(66)
4.3 构造 CMCIClass 类	(77)
4.4 CMCIClass 编程	(93)
4.4.1 从 CMCIClass 类派生子类	(94)
4.4.2 CMCIClass 编程示例	(98)
4.5 用 CMCIClass 编制一个媒体播放器	(103)
第五章 MCIWnd 窗口类编程	(115)
5.1 Video for Windows	(115)
5.2 Visual C++ 与 MCIWnd 窗口类	(116)
5.3 使用 AVIFile 函数	(116)
5.4 使用 MCIWnd 窗口类	(118)
5.5 MCIWnd 窗口类编程	(121)
5.6 基于 MCIWnd 窗口类的媒体播放器	(124)
第六章 音频编程	(130)
6.1 高层波形音频函数	(131)
6.2 低层音频函数	(133)
6.2.1 低层波形音频函数编程	(134)
6.2.2 波形音频音效处理	(152)
6.2.3 低层 MIDI 音频函数编程	(161)
6.3 波形音频管理器	(168)
第七章 图像编程	(172)
7.1 位图与调色板	(172)
7.2 与设备相关位图(DDB)	(175)
7.3 与设备无关位图(DIB)	(177)
7.4 构造 CDib 类	(183)
7.5 用 CDib 实现一个 DIB 观察器	(186)
第八章 动画编程	(193)
8.1 动画基础	(193)
8.2 图形动画	(193)
8.3 调色板动画	(196)
8.4 位图动画	(201)
8.4.1 无背景位图动画	(201)
8.4.2 有背景位图动画	(206)
第九章 实例分析:五子棋游戏	(219)
9.1 目标与规则	(219)
9.2 程序结构	(219)

9.3 双语界面	(222)
9.4 不同的光标提示	(224)
9.5 算法	(225)
9.6 逼真的棋盘和棋子显示	(243)
9.7 增加声音效果	(244)
9.8 系统信息显示	(247)
9.9 游戏程序清单	(252)
附录 关于本书附盘	(254)
参考文献	(255)

第一章 Visual C++ 5.0 编程技术

Visual C++ 5.0 是 Microsoft 公司推出的开发 Win 32 环境(Windows 95 和 Windows NT)程序的面向对象的可视化集成编程系统。它不但具有程序框架自动生成、灵活方便的类管理、代码编写和界面设计集成交互操作、可开发多种程序(应用程序、动态连接库、ActiveX 控件等)等优点,而且通过简单的设置就可使其生成的程序框架支持数据库接口、OLE2、WinSock 网络、3D 控件界面,因此现已成为开发 Win 32 程序的主要开发工具。

本章不是 Visual C++ 的教科书,而只是对用 Visual C++ 进行 Win 32 编程的主要功能特征做一概要说明,目的是使具有一定基础的读者能更加深入地理解 Visual C++ 编程精要。

1.1 OOP 与 Visual C++ 特性

面向对象程序设计(Object-Oriented Programming,简称 OOP)方法已出现近 30 年,90 年代已成为程序设计的主流方向。面向对象程序设计语言是现代程序开发的主要工具,如 C++、Java 是现代程序员必须掌握的编程语言。

程序包含两类基本的元素,即数据和操作数据的指令集(称为代码)。传统的程序设计语言以设计代码为核心,程序设计实际上就是指定程序指令的先后次序,数据表示必须适应代码的设计。模块化程序设计方法将完成某一功能的指令集组成一个相对独立的程序模块(即函数或过程),使程序结构清晰,便于有效地维护,对程序设计技术有了很大的促进。但由于结构化程序设计方法并不能保证各程序模块之间真正相互独立,程序设计者在设计一个模块时很难完全排除其他模块的影响,随着程序规模的增大,各模块之间的相互影响导致了一些难于测试、难以定位发现的错误,增加了程序开发和维护的困难。面向对象程序设计方法就是在这种背景下出现和发展起来的。

面向对象程序设计方法主要以数据为中心,代码是围绕着需要处理的数据而设计。面向对象程序设计语言具有如下的主要特征:

- 对象的类描述

面向对象程序设计语言将程序描述的事物看成一个整体,称之为对象(object)。事物的属性可以分为基本的两部分,即内部状态(性质)和对数据的操作方法及由此造成的对外部的影响,对象的数据用于描述内部状态,而代码完成对数据的操作。因此,对象就是包含数据和代码的完全独立的实体。类(class)就是具有相同属性的所有对象的逻辑原型,是对象的规则和设计。同一类的对象具有相同的性质和方法,每一个具体的对象都是类的一个实体,创建对象就是把类实例化。

- 封装性(Encapsulation)

封装性是 OOP 的核心技术,是指面向对象程序设计语言将数据和处理数据的方法组合在类中,具有模块化和信息隐藏的特征。类是一个独立的模块,类的内部状态描述数据从程序的其他部分是不可见的,类只向外界公布其具有 public 属性的数据和代码,这构成了类与外界

接口。外界不能直接对类的内部状态进行修改,而只能通过这个接口将信息传递给类,并由类定义的对内部数据进行操作的方法进行内部修正,外界不能决定这种修正的结果,只能得到类进行操作所做出的反应。封装性能防止类与外部的非法交互和访问,避免外界对对象内部状态的错误改变,确保类这一模块的真正独立,以保证程序的安全运行。

同时由于程序的其他部分只能访问类的接口,只要保持类的接口不变,改变类的内部结构、工作方式和实现就不会对整个程序产生非预期的影响,因此对类的内部做任何的优化都是安全的。

- 多态性(Polymorphism)

不同的类或对象对外界传入的相同信息能根据自身的性质做出不同的反应,这就是多态性。通过不同的类或对象都可设计自身的处理外界传入信息的方法就可实现多态性。在 OOP 中,多态性具有两方面的实际意义,一是具有相同名字的接口(具有 public 属性的公有数据和函数)在不同类中能具有不同的意义和实现;二是具有同一名字的函数可以具有不同的实现代码,在调用时,根据传入的参数不同,而调用不同的代码,这叫做函数的重载。

- 继承性(Inheritance)

继承性是指一个类可以派生出新的类,新类能继承原类定义的性质和方法,还能在原类定义的性质和方法之外加入自身定义的性质和方法。通过继承性能形成类之间的层次结构,在上层中已经定义的性质和方法能被下层直接继承使用,下层就不需重新定义,实现了代码的重复利用。这样,下层的类只需专注于自身的新特征描述,提高了程序设计的效率和程序组织的有效性。

不是由其他类派生的类成为基类。由其他类派生的类叫做该类的子类,该类叫做其子类的父类或超类。子类可以利用多态性特征来对继承而来的方法进行重载,使得具有相同名字的方法在子类和父类中实现不同。

C++ 是运用最广泛的面向对象程序设计语言,Visual C++ 是一个具有集成、交互和可视化编程的 C++ 实现,具备上述的所有 OOP 特征。

编写 Visual C++ 程序实际上就是一个构造类和把类实例化的过程。由于 Windows 95 和 Windows NT 是 PC 平台中应用最广泛的操作系统(Microsoft 力图用一个叫做 Win 32 的标准的 32 位应用程序接口来作为对这两个操作系统的共同开发接口,所以经常采用 Win 32 来代表 Microsoft 的这两个 32 位 Windows 操作系统),Visual C++ 5.0 主要针对的就是 Win 32 应用程序的开发。

1.1.1 Win 32 编程

Win 32 具有抢先式多任务、多线程和线性寻址内存管理等特征,Win 32 编程的基本要求包括:

- ✓ 应用程序的执行独立于硬件设备。
- ✓ 应用程序具有图形用户界面。
- ✓ 能在 Windows 95 和 Windows NT 之间透明移植,并可移植到支持 Windows NT 的 RISC 硬件平台。
- ✓ 高性能的抢先式多任务和多线程管理。

- ✓ 高级的多媒体支持。
- ✓ 通过 OLE 2 技术实现多个应用程序的对象定位。

Microsoft 为进行 Win 32 编程提供了一套名为 Win 32 SDK 的应用程序编程接口,其中包括上千个 Win 32 系统函数。Visual C++ 5.0 包括一套叫作 MFC (Microsoft Foundation Class Library) 的 C++ 类库,其中定义了进行 Win 32 编程所需的各种类,有的类封装了大部分的 Win 32 SDK 中应用程序编程接口函数,有的类封装的则是应用程序本身的数据和操作,还有的类封装了 ActiveX、OLE 和 Internet 编程特性、WinSock 网络特性和 DAO (Data Access Objects)、ODBC (Open Database Connectivity) 数据访问功能。Win 32 SDK 和 MFC 是实现 Win 32 编程的主要工具。

Visual C++ 的 AppWizard 工具能自动生成应用程序框架,该框架定义了应用程序的轮廓,并提供了用户接口的标准实现方法。运用 Visual C++ 的资源编辑器 (Resource Editor) 能直观地设计程序的用户界面,而 ClassWizard 能把用户界面和程序代码联接起来。程序员要做的就是用 MFC 类实现框架中未完成的应用程序的特定功能部分。所以,应用 Visual C++ 可以实现 Win 32 的可视化程序设计。

1.1.2 框架和文档-视结构

所谓框架 (Framework), 就是应用程序所应具备的软件模块按一定的结构组成的集合。基于 MFC 的应用程序框架是定义了程序结构的 MFC 类库中类的集合, 是 Visual C++ 编程的骨架。运用 MFC 应用程序框架能获得如下的优点:

- ✓ 标准化的程序结构和用户接口: 这对具有标准用户界面的 Win 32 程序来说, 可以极大地减轻程序员的负担, 使程序员不必过多地考虑界面, 而把主要精力放在程序设计上, 提高程序设计的效率。

- ✓ 框架产生的程序代码短, 运行速度快, 具有很大的灵活性: MFC 封装了 Win 32 SDK 中的几乎所有函数, 能实现 Win 32 系统的任何功能。

- ✓ 强大的功能: 除封装了大部分的 Win 32 SDK 函数外, MFC 还提供了应用程序本身的数据和操作、ActiveX、OLE、Internet、WinSock、DAO (Data Access Objects)、ODBC (Open Database Connectivity) 等操作类。

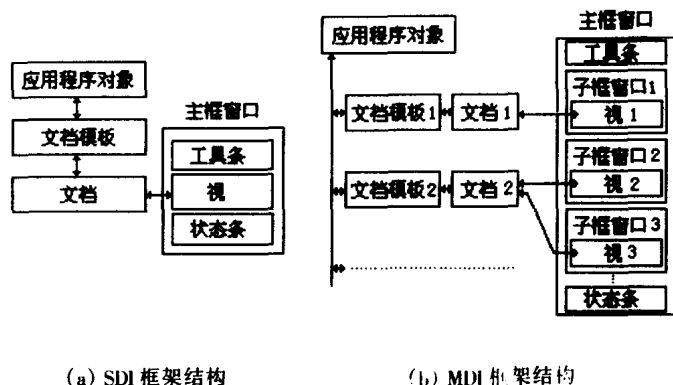
MFC 框架的核心是文档-视结构 (Document-View Architecture), 这是一个很有用, 但又往往难以入门的功能。简单地说, 文档-视结构就是将数据和对数据的观察或数据的表现 (显示) 相分离, 文档仅处理数据的实际读、写操作, 视则是显示和处理数据的窗口, 视可以操作文档中的数据。

• 框架结构

MFC 框架的基本结构包括应用程序对象、主框窗口、文档、视等, 框架通过命令和消息将它们结合在一起, 共同对用户操作做出响应。

图 1-1 表示了 MFC 框架结构和各对象间的关系。应用程序管理着一个 (SDI) 或多个 (MDI) 文档模板, 每个文档模板管理着一个或多个文档。用户通过主框窗口中的视观察和处理数据。

• 应用程序对象



(a) SDI 框架结构

(b) MDI 框架结构

图 1-1 MFC 框架结构

由 `CWinApp` 派生应用程序类对象。一个应用程序有且只有一个应用程序对象,它负责应用程序实例的初始化和进程结束时的资源清除,以及创建和管理应用程序所支持的所有文档模板。

任何的 Windows 应用程序都包含一个 `WinMain()` 函数,框架应用程序也一样,只是框架中的 `WinMain()` 函数由 MFC 类库提供,并被隐藏,在框架应用程序启动时调用。应用程序对象的 `InitInstance()` 函数就是由 `WinMain()` 调用的。

• 主窗口

这是应用程序的主窗口。MFC 框架定义了两种基本的主窗口类,即单文档接口 (Single Document Interface, 简称 SDI) 主窗口类 `CFrameWnd` 和多文档接口 (Multiple Document Interface, 简称 MDI) 主窗口类 `CMDIFrameWnd`。应用程序的主窗口应从其中之一派生出来。对 SDI, 视是主窗口的子窗口;对 MDI, 必须从 `CMDIChildWnd` 派生出主窗口的子窗口,视是该子窗口的子窗口。

• 文档

文档类由 `CDocument` 类派生而来,文档类对象由框架生成的 File 菜单中的 New 或 Open 命令创建,指定了应用程序数据的实际读、写操作。如想使应用程序支持 OLE 功能,则应从 `COleDocument` 类派生文档类。

文档由应用程序对象创建和维护的文档模板 (Document Template) 所创建。框架分别使用 `CSingleDocTemplate` 和 `CmultiDocTemplate` 文档模板来管理 SDI 和 MDI,前者可以创建和存储一种类型的文档,后者保存了一种类型的多个文档的列表。

• 视

视类从 `CView` 或其子类 (`CEditView`, `CFormView`, `CRecordView`, `CScrollView` 等) 派生而来,是显示和观察文档数据的窗口类。视类定义了用户以什么方式见到文档的数据,以及如何与其进行交互。一个文档数据可能有多个视。

• 文档-视结构

框架在响应它生成的标准用户接口的 File 菜单中的 New 和 Open 命令时将创建文档-视结构,其创建次序如下:

1. 在程序启动时, `WinMain()` 函数调用应用程序对象的 `InitInstance()` 函数,并在其中创建文档模板。

2. 程序运行过程中,用户选取了 File 菜单中的 New 或 Open 菜单项,框架将调用 `CWinApp`

::OnFileNew()或 CWinApp::OnFileOpen()函数,并使用已创建的文档模板创建文档。

3. 文档模板同时创建主框窗口(SDI)或子框窗口(MDI)。

4. 主框窗口(SDI)或子框窗口(MDI)创建文档对应的视。

文档-视结构中各对象的交互关系如图 1-2 所示。在文档-视结构中各对象创建以后,程序员应覆盖相应类的成员函数,以对各对象做具体的初始化。在视类中覆盖 OnInitialUpdate 是初始化视的最合适方法,覆盖文档类的 OnNewDocument 和 OnOpenDocument 成员函数可以为文档做特定的初始化。

1.1.3 消息映射

Windows 应用程序是消息驱动的,应用程序不能直接得到用户所做的操作事件,如鼠标按键、键盘输入、窗口移动等,这些操作由操作系统管理。操作系统检测到操作事件后,便向适当的应用程序发送消息,应用程序响应这些消息来完成用户的操作。

• 消息

Windows 中消息是操作系统与应用程序之间、应用程序之间、应用程序各对象之间相互控制、传递信息的方式。

消息的基本格式是:

Message wParam lParam

Message 是消息名称, wParam 是与消息相关的 WORD 型参数, lParam 是消息相关的 LONG 型参数。

主要有如下 3 类消息:

1. Windows 系统消息:Windows 系统向窗口发送的消息,由窗口或视进行响应处理。这类消息包括除 WM_COMMAND 消息之外的其他名称以 WM_ 开始的消息。
2. 控制通知消息:控制或子窗口传给父窗口的 WM_COMMAND 通知消息。
3. 命令消息:响应用户接口操作时,将产生 WM_COMMAND 命令消息,其参数指定了用户接口的标识号,如菜单项、按钮等的 ID 号。

• 消息映射

MFC 应用程序框架设置了相应的消息处理函数来响应消息,完成相应的操作。消息处理函数是某些类(通常是窗口类)的成员函数,程序员在其中编写响应消息时应进行操作的代码。

框架将消息和它们的处理函数联接起来就是消息映射。消息映射使应用程序在接受到消

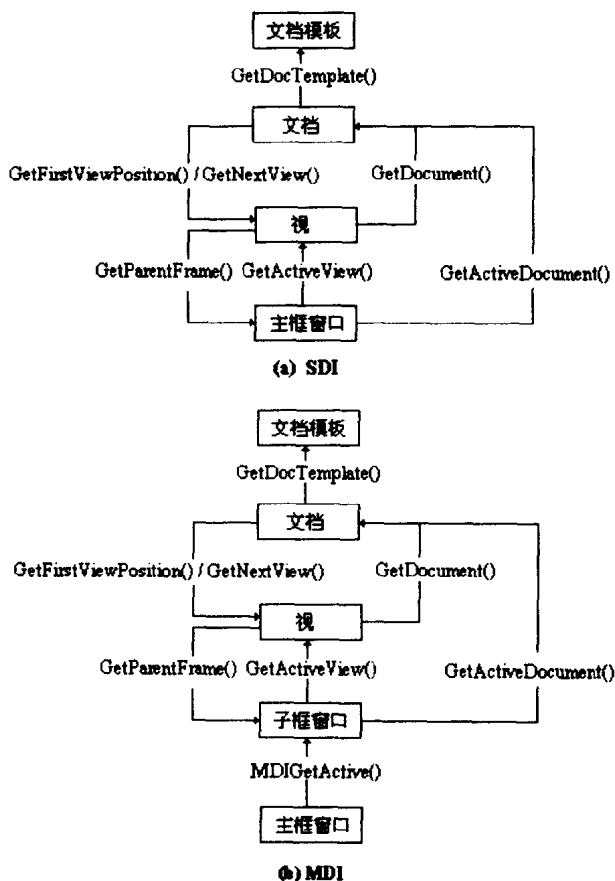


图 1-2 文档-视结构中各对象的交互关系

息时调用对应的消息处理函数来响应和处理消息。

ClassWizard 在创建新类时将其创建一个消息映射,并能为每个类能响应的消息和命令增加对应的处理函数。在源代码中,消息映射开始于 BEGIN_MESSAGE_MAP 宏,结束于 END_MESSAGE_MAP 宏,中间由一系列预定义的被称作条目宏的宏组成。其基本格式如下:

```
BEGIN_MESSAGE_MAP(classname, parentclassname)
    ///AFX_MSG_MAP(classname)
    条目宏 1
    条目宏 2
    条目宏 3
    .....
    ///AFX_MSG_MAP
END_MESSAGE_MAP()
```

其中 classname 为拥有消息映射的当前类名, parentclassname 为当前类的父类名。条目宏定义了类所处理的消息及与其对应的函数,常用的条目宏的类型如表 1-1 所示。

表 1-1 消息映射条目宏

消息类型	宏格式	说明
Windows 消息	ON_WM_XXXX	WM_XXXX 为 Windows 消息名
命令	ON_COMMAND(ID, Function)	ID 为命令标识号 Function 为处理函数名
更新命令	ON_UPDATE_COMMAND_UI(ID, Function)	ID 为命令标识号 Function 为处理函数名
控制通知	ON_XXXX(ID, Function)	ID 为控制标识号 Function 为处理函数名
用户定义消息	ON_MESSAGE(ID, Function)	ID 为消息标识号 Function 为处理函数名
用户注册消息	ON_REGISTERED_MESSAGE(ID, Function)	ID 为消息标识号 Function 为处理函数名

注意, MFC 要求所有处理函数声明为 afx_msg 类型。

Windows 消息的处理函数在 CWnd 类中进行了预定义,类库以消息名为基础定义这些处理函数的名称。例如,消息 WM_PAINT 的处理函数在 CWnd 类中声明如下:

```
afx_msg void OnPaint();
```

通过 ClassWizard 在派生类中用同样的原型定义处理函数并为该函数生成消息映射条目,然后编写处理函数代码,便在派生类中覆盖了其父类的消息处理函数。在有些情况下,必须在派生类的消息处理函数中调用其父类的消息处理函数,使 Windows 和基类都对消息进行处理。ClassWizard 将在生成的处理函数中建议是否应调用父类的消息处理函数,及调用的次序。

用户定义和注册的消息、命令和控制通知都没有默认的处理函数,需要在定义时声明,一般建议根据其 ID 名称来为函数命名。

1.1.4 Visual C++ 可视化编程

Visual C++ 的资源编辑器能以所见即所得(What you see is what you get)的形式直接编辑程序用户界面,为所有资源分配 ID 标识号。ClassWizard 能将对话框模板与生成类定义或与已有的类代码联接起来,为菜单项、控制等资源生成空的处理函数模板,创建消息映射条目,并将资源 ID 与处理函数联接起来。用 AppWizard 程序员的编程工作便简化为资源编辑器直观地设计界面、完善对话框类代码、在空的处理函数模板处填写响应用户操作的代码。这是一种完善的可视化编程方法。

用 Visual C++ 进行 Win 32 可视化编程的基本流程如下:

1. 生成框架:运行 AppWizard,并按需要指定生成应用程序的选项,指定框架中视类的基类(CView、CEditView、CFormView、CScrollView、CTreeView 等)。AppWizard 将按指定的选项生成应用程序框架和相关的文件,包括包含项目(project)的工作空间(workspace)文件和源文件,主要是应用程序(application)、文档(document)、视(view)和主框架窗口(main frame)的 C++ 代码文件(*.cpp, *.h),以及缺省包含标准界面接口的资源文件(*.rc)。

2. 设计用户界面:利用 Visual C++ 资源编辑器可视化地直观编辑资源文件,定制菜单、对话框、工具条、字符串、加速键、位图、图标、光标等接口资源。

3. 联接界面和代码:利用 ClassWizard 把资源文件中定义的界面资源标识(如菜单项、工具条和对话框中的控制等)在指定的源文件中映射成相应的函数模板。

4. 编写、修改函数代码:利用 ClassWizard 可以方便地在源码编辑器(source code editor)中跳转到指定的函数代码处。

5. 根据需要创建新类和编写代码:用 ClassWizard 创建新类,并生成相应的源文件。如新类是对话框类,可先用资源编辑器生成对话框模板,然后用 ClassWizard 创建对话框类代码,并与模板联接。编写新类相关的源代码。

6. 实现文档类:在 AppWizard 生成的框架基础上设计文档数据的数据结构、在文档类中增加相应的成员数据,实现对数据的操作和文档与数据的接口。

7. 实现框架中标准的文件操作命令,即 Open、Save 和 Save As;框架已完成标准的文件操作命令的所有接口,程序员要做的仅仅是编写文档类的串行化(Serialize())成员函数。

8. 实现视类:框架已构造好了文档与视的关系,视能方便地访问文档中的 public 数据成员。可根据文档的需要构造一个或多个视类,通过 ClassWizard 把视的用户接口资源映射成函数模板,并编写函数代码。

9. 如需要,增加分割窗口(splitter window):在 SDI 的主框架窗口类或 MDI 的子窗口类中添加一个 CSplitterWnd 对象,并在窗口类的 ONCreateClient 成员函数中对 CSplitterWnd 对象进行创建和初始化。如果用户分割了一个窗口,框架将给文档创建并增加附加的视对象。

10. 建立、调试、修改应用程序。如有问题,根据需要重复步骤 2 至步骤 10。

11. 测试应用程序。如有问题,根据需要重复步骤 2 至步骤 11。

12. 结束。

1.2 中文程序开发环境的安装

Visual C++ 5.0 安装程序安装的 Visual C++ Developer Studio 不能支持中文的资源编辑,在用 AppWizard 构造应用程序时提供的应用程序语种选项中没有中文选项。如果强行在资源编辑器中输入中文,得到的将是乱码符 。

为了使 Visual C++ Developer Studio 支持中文程序开发,就必须从 Visual C++ 5.0 安装光盘中手工安装 Visual C++ 5.0 的中文环境支撑链接库,安装步骤如下:

1. 退出已运行的 Visual C++ Developer Studio。

2. 拷贝以使中文 AppWizard 支持动态链接库:将 Visual C++ 5.0 安装光盘中 \DEVSTUDIO\SHAREIDE\Bin\Ide 目录下的 Appwzchs.dll 拷贝到硬盘中安装 Visual C++ 5.0 的相应目录中,通常是 C:\Program Files\DevStudio\ShareIDE\bin\ide。

3. 拷贝中文 MFC 资源动态链接库:将 Visual C++ 5.0 安装光盘中 \DEVSTUDIO\VC\REDIST 目录下的 Mfc42chs.dll 拷贝到 Windows 95 的系统目录 (Windows\System) 中,并将其改名为 Mfc42loc.dll。

4. 拷贝中文 MFC 资源静态链接库:将 Visual C++ 5.0 安装光盘中的 \DEVSTUDIO\VC\Mfc\Src\L.chs 目录和 \DEVSTUDIO\VC\Mfc\Include\L.chs 目录拷贝到硬盘中安装 Visual C++ 5.0 的相应目录 (通常是 C:\Program Files\DevStudio\VC\Mfc\Src\L.chs 和 C:\Program Files\DevStudio\VC\Mfc\Include\L.chs) 下。

经上述安装后,AppWizard 在生成应用程序框架时将提供中文选项,如选择中文,则生成的应用程序的菜单、对话框、控制、字符串中都将显示中文。

用同样的方法能使 Visual C++ 5.0 支持日文、韩文等其他的双字节语种。

1.3 编程风格问题

编程风格问题是一个容易引起很大争论的问题,有些人认为自己能适应的风格就是合适的风格,但无论如何,为了使程序代码易于维护,我们在编写代码时还是应遵循一定的公认准则。

- 语句缩进准则

虽然没有任何一种编程语言强制要求使用缩进准则,但在实际编程中,几乎每一个程序员都使用缩进来增强代码的可读性。使用缩进的方式是千差万别的,我们建议使用 Visual C++ 示例中所采用的缩进方式。

- 关于注释

C++ 同时支持 /* ... */ 和 // 注释方式,我们建议对于大段注释采用 /* ... */ 方式,对于较少行的注释在每行前加 // 表示注释。

虽然不要求对每行语句都进行注释,但最好是对每一源文件、每个函数、每一特殊操作语句(或语句段)、重要变量都能有注释说明文件、函数、语句或变量的用途和意义。

对源文件的注释应放在文件的最前面,一般包括如下几部分:

- ✓ 文件名

- ✓ 文件功能、用途
- ✓ 创建时间
- ✓ 作者
- ✓ 修改时间
- ✓ 修改的目的
- ✓ 修改人
- 源文件设置

每个类都应有它自己的头文件(.h 文件)来说明其类定义,而把类代码放在名称相同的 .cpp 文件中。如将类 CMyClass 的定义放在 MyClass.h 中,而类代码放在 CmyClass.cpp 中。

在项目较大时,建议设定一个总头文件来包括其他所有头文件,而在所有代码源文件(.cpp)中仅包括该总头文件。这样有利于防止互相冲突的定义发生。

使用常数时,应尽可能在头文件中使用常量定义,尽量避免直接在代码中使用常数。

在每一个头文件中应设置防止该头文件被多次包括的预编译条件语句,其形式如下:

```
// =====
// Sample.h
// 该头文件说明
// .....
// =====

#ifndef __SAMPLE_H
#define __SAMPLE_H

.....
头文件内容
.....

#endif //! __SAMPLE_H
```

• 命名准则

编写代码时如何给常量、变量和函数取名将在很大程度上影响代码的可读性,好的命名方式可减少编写代码的笔误和提高代码的维护效率。一般的命名准则包括:

- ✓ 使用能描述常量、变量或函数的意义或用途的词命名。
- ✓ 所有的常量名都使用大写字母,不同词之间用下划线(_)分开。
- ✓ 函数名中每个词的的第一个字母大写,函数名中一般不使用下划线来分隔词。

✓ 变量命名应遵循匈牙利表示法,即变量名以表示变量类型的小写前缀开始,变量名中其余词的的第一个字母大写。匈牙利表示法的常用前缀如表 1-2 所示。

表 1-2 匈牙利表示法常用前缀

前 缀	含 义
a	数组
b	BOOL
by	BYTE