精通Visual C++图像编程

周长发 著

精通 Visual C++ 图像编程

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL: http://www.phei.com.cn

精通Visual C++图像编程

周长发 著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

本书全面介绍了 Windows 95/98/NT 环境下图像处理的理论和编程技术，并基于面向对象的程序设计方法，详细讨论了图像处理算法的 Visual C++ 编程技巧。本书的主体包括基本的图像操作、调色板处理、图像的特技显示与擦除、图像变换、颜色处理、图像处理等的算法及实现技术。

本书是进行图像处理和多媒体编程的实用参考书，适合图像处理和多媒体编程人员参考阅读，也可作为大专院校计算机及相关专业师生的教学参考书。

图书在版编目(CIP)数据

精通 Visual C++ 图像编程/周长发著．－北京：电子工业出版社，2000.1
ISBN 7-5053-4563-X

Ⅰ．精… Ⅱ．周… Ⅲ．C 语言-程序设计 Ⅳ．TP312

中国版本图书馆 CIP 数据核字(1999)第 73809 号

书　　名：**精通 Visual C++ 图像编程**
著　　者：周长发
策划编辑：郭　立
责任编辑：黄志瑜
特约编辑：郭建廷
排版制作：电子工业出版社计算机排版室
印 刷 者：北京天竺颖华印刷厂
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：28.75　字数：733 千字
版　　次：2000 年 1 月第 1 版　2000 年 1 月第 1 次印刷
书　　号：ISBN 7-5053-4563-X / TP·2152
印　　数：4000 册　定价：38.00 元

前　言

数字图像处理技术与理论是计算机应用的重要基础，许多实际编程都涉及图像处理算法的编程实现。本书在分析常用的图像处理技术的基础上，基于面向对象的编程技术，结合大量的示例，向读者展示了用 Visual C++ 进行图像处理编程的基本方法和技巧。

全书分为八章。第 1 章概括了 Visual C++ 面向对象程序设计的要点和重要特征。第 2 章讨论了数字图像处理的基本概念，以及基本的位图和调色板处理技术。第 3 章在全面分析了 Windows 与设备无关位图操作的基础上，建立一套实用的与设备无关位图的处理函数集，并用面向对象的编程技术设计实现了处理与设备无关位图的 CDib 类。第 4 章详细介绍了位图的特技显示与擦除技术，包括扫描、移动、百叶窗、栅条、马赛克、渐显与渐隐和透明显示的实现方法。第 5 章说明了位图变换，即剪切、缩放、旋转和镜像的理论与实现。第 6 章介绍了颜色处理的基础理论，以及灰度化、假彩色与伪彩色处理、颜色分离与滤色、颜色调整、亮度/对比度调整、颜色量化与减色和抖动的编程技术。第 7 章研究了图像处理的理论与实现，重点阐述了图像平滑与锐化、边缘增强、Sobel 边缘检测、Hough 边缘检测、中值滤波去噪等的编程实现。第 8 章综合运用前几章的知识，以 CDib 类为基石，详细说明了一个功能较完整的图像处理程序的编制方法和过程。

本书提供了大量的例程，涉及图像处理的各主要方面。所有的例程都已用 Visual C++ 6.0 在中文 Windows 98 环境下调试实现。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例程代码。

周建欣测试了本书的示例程序；周建欣、林琳、黄家辉、邓优、李秀鹃、刘烁及何西涛等参加了本书有关内容的讨论和编写。限于笔者的能力，错误、浅陋和陈旧之处在所难免，恳请读者批评指正。

作　者

目　录

第 1 章　Visual C++编程概要

Visual C++是 Microsoft 公司推出的开发 Win 32 环境（Windows 95/98/ NT，以及即将推出的 Windows 2000）程序，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、动态链接库、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、OLE 2、WinSock 网络、3D 控件界面。因此，它现已成为开发 Win32 程序的主要开发工具。

仅用一章的篇幅来全面介绍 Visual C++几乎是不可能的，为此笔者假设本书的读者都具有一定的 Visual C++基础，这就避免了这一难题。本章既为“概要”，就对 Visual C++编程的若干要点进行重点讨论，以帮助读者更加深入地理解和掌握 Visual C++编程之精髓。

1.1　Visual C++技术主要特征

面向对象程序设计（Object-Oriented Programming 简称 OOP）方法已出现近三十年，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)，使得程序的结构清晰，便于有效地维护，对程序设计技术有很大的促进。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的相互独立，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的，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，称为对象（object)。事物的属性基本可以分为两部分，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）就是具有相同的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，并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对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，并构成了类与外界的接口。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 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，从而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称为基类。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的功能。

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 95/98/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几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 32 位 Windows 操作系统），Visual C++主要针对 Win 32 的应用程序开发。

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：

(1) 应用程序的执行独立于硬件设备；

(2) 应用程序具有图形用户界面；

(3) 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；

(4) 高性能的抢先式多任务和多线程管理；

(5) 高级的多媒体支持；

(6) 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++包括一套叫做 MFC（Microsoft Foundation Class Library）的 C++类库，其中定义了进行 Win 32 编程所需要的各种类。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连接起来。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，使用 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：

(1) 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，以提高程序设计的效率；

(2) 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；

(3) 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，及 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，但又往往较难以入门的功能。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，共同对用户的操作做出响应。

图 1-1 表示了 MFC 框架结构和各对象之间的关系。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。

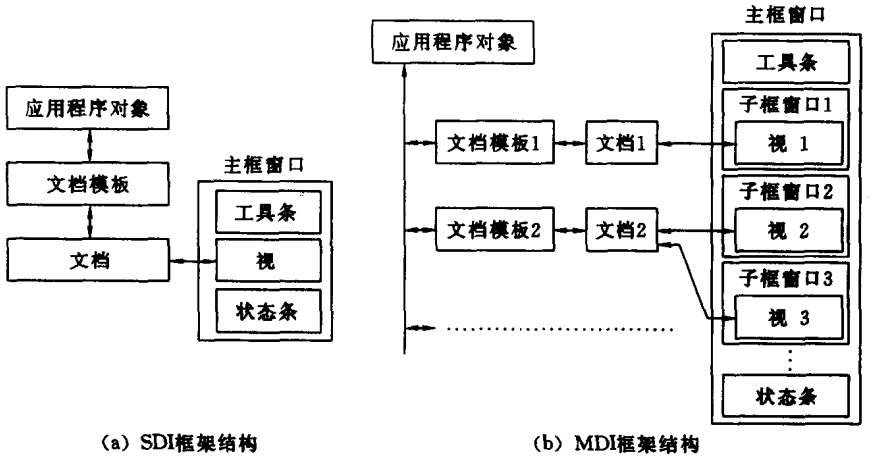

图1-1　MFC框架结构

■ **应用程序对象**

这是由 CWinApp 派生的应用程序类对象。一个应用程序有且仅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 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。

■ **主框窗口**

这是应用程序的主窗口。MFC 框架定义了两种基本的主框窗口类，即单文档接口（Single Document Interface，简称 SDI）主框窗口类 CFrameWnd 和多文档接口（Multiple Document Interface，简称 MDI）主框窗口类 CMDIFrameWnd。应用程序的主框窗口应从其中之一派生出来。对 SDI，视是主框窗口的子窗口；对 MDI，必须从 CMDIChildWnd 派生出主框窗口的子窗口，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 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，便向相关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 中的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与信息传递的方式。

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。

主要有三类消息：

(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：为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 ID 号。

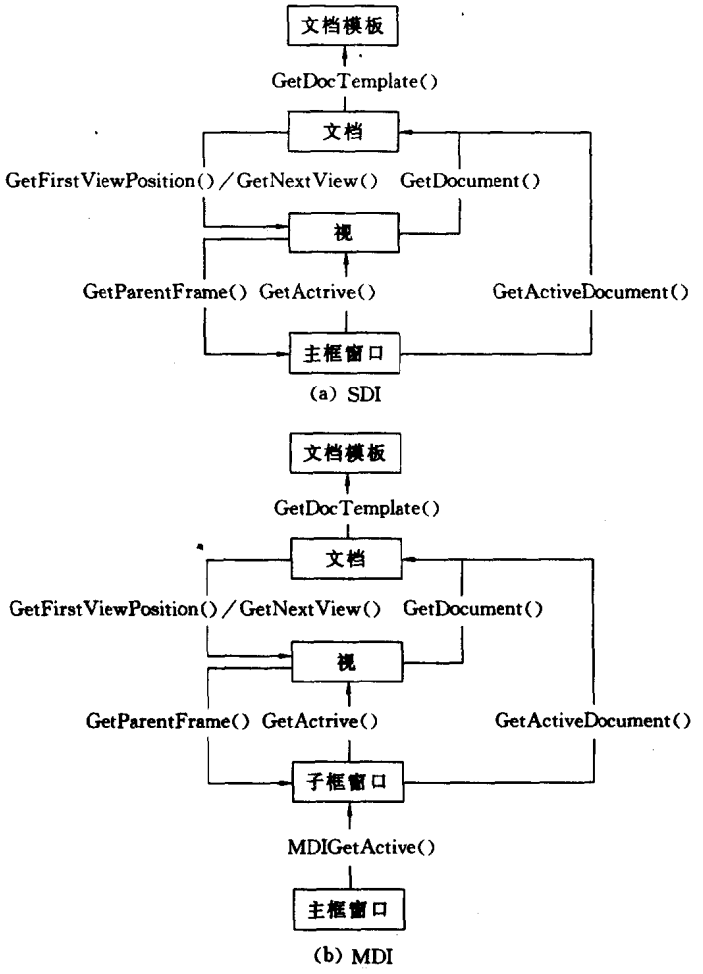

图 1-2　文档-视结构中各对象的交互关系

■ 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，以完成相应的操作。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连接起来就是消息映射。消息映射使应用程序在接收到消息时调用对应的消息处理函数来响应和处理消息。

ClassWizard 在创建新类时将为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，消息影射开始于 BEGIN_MESSAGE_MAP 宏，结束于 END_MESSAGE_MAP 宏，中间由一系列预定义的被称为条目宏的宏组成，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 为拥有消息影射的当前类名，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　消息影射条目宏

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 要求所有消息处理函数声明为 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影射条目，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 和基类能对消息进行处理。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缺省的处理函数，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 能把对话框模板与生成类定义或与已有的类代码连接起来，为菜单项、控制等资源生成空的处理函数模板，创建消息影射条目，并将资源 ID 与处理函数联接起来。通过使用 AppWizard，程序员的编程工作便简化为用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) 连接界面和代码：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，并与模板连接，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、成员函数，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 命令：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( )）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，可根据需要重复步骤 2~10。

(11) 测试应用程序。如有问题，可根据需要重复步骤（2）~（11）。

(12) 结束。

1.2 中文程序开发环境的安装

对于 Visual C++ 6.0 以前的版本，Visual C++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也没有中文选项。如果强行在资源编辑器中输入中文，得到的将是乱码符号。为了使 Visual C++ Developer Studio 支持中文程序开发，就必须从 Visual C++安装光盘中手动安装 Visual C++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 AppWizard 中文支持动态链接库：将 Visual C++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++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++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++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++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++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，以说明文件、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：

(1) 文件名；

(2) 文件功能、用途；

(3) 创建时间；

(4) 作者；

(5) 修改时间；

(6) 修改的目的；

(7) 修改人。