

软·件·工·程·师·入·门

Visual C++

项目开发 实例

■ 明日科技 编著

自学手册

- 提供开发环境搭建视频录像、实例配置录像，只需按录像操作，就可以轻松构建开发环境
- 项目涉及数据库、图像处理、网络传输、硬件开发等多种开发领域
- www.mrbccd.com 网站免费下载“Visual C++编程词典”体验版
- 编程一线人员联手打造，融合多年的项目经验和工作心得

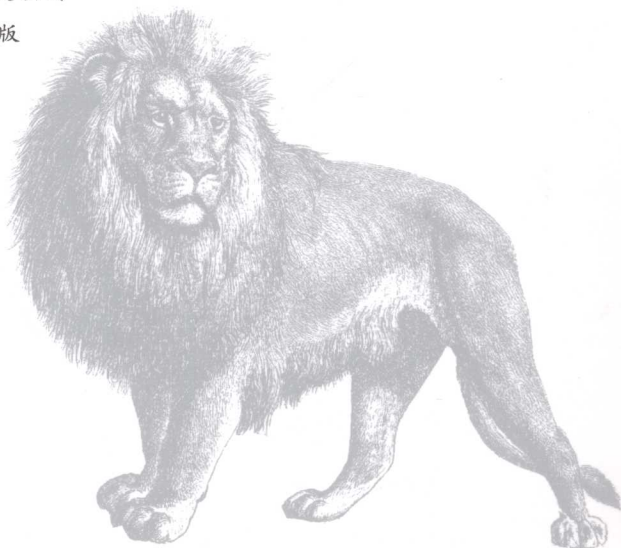


光盘内容

- 全部实例的系统文件和源程序
- 环境搭建及实例配置视频录像
- 便捷的查询工具——编程词典体验版



人民邮电出版社
POSTS & TELECOM PRESS



TP312/2992D

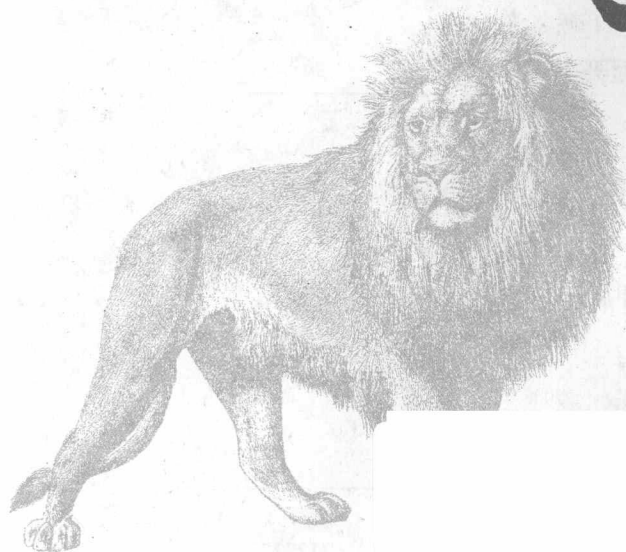
2008

Visual C++

项目开发 实例

■ 明日科技 编著

自学手册



人民邮电出版社

北京

图书在版编目 (CIP) 数据

Visual C++项目开发实例自学手册 / 明日科技编著.
北京: 人民邮电出版社, 2008.8
(软件工程师入门)
ISBN 978-7-115-18456-6

I. V... II. 明... III. C 语言—程序设计—技术手册
IV. TP312-62

中国版本图书馆 CIP 数据核字 (2008) 第 098966 号

内 容 提 要

本书以图像处理系统、局域网屏幕监控系统等 8 个实际项目开发程序为案例,从软件工程的角度出发,按照项目的开发顺序,系统、全面地介绍了程序开发流程。本书案例涉及行业广泛,实用性强,通过对本书的学习,读者可以了解各个行业的特点,能够针对某一行业进行软件开发,也可以通过光盘中提供的案例源代码和数据库进行二次开发,以节省开发系统所需的时间。

本书所附配套光盘中提供了书中案例的全部源代码,源代码都经过精心调试,在 Windows XP 和 Windows Server 2003 操作系统下全部通过,保证能够正常运行。

本书适用于使用 Visual C++进行应用程序开发的初、中级读者,一些案例对于高级用户也具有很好的参考价值。同时,本书也特别适合作为大专院校计算机专业师生的案例参考书。

软件工程师入门

Visual C++项目开发实例自学手册

- ◆ 编 著 明日科技
责任编辑 屈艳莲
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京铭成印刷有限公司印刷
- ◆ 开本: 787×1092 1/16
印张: 20
字数: 532 千字
印数: 1—5 000 册
- 2008 年 8 月第 1 版
2008 年 8 月北京第 1 次印刷

ISBN 978-7-115-18456-6/TP

定价: 42.00 元 (附光盘)

读者服务热线: (010)67132692 印装质量热线: (010)67129223
反盗版热线: (010)67171154



Visual C++是 Microsoft 公司开发的基于 C/C++语言的集成开发工具,同其他的可视化编程工具一样, Visual C++集代码编辑、编译、连接、调试等功能于一体,提供了多种有用的辅助开发工具,不但大大提高了应用程序的开发效率,而且给编程人员提供了一个完整又方便的开发环境。Visual C++强大的可视化用户界面设计,使程序员从复杂的界面设计中解脱出来,使编程成为一种享受。

本书内容

本书提供了 8 个经典案例。在这 8 个案例中,既有传统的实用案例,又有当前比较流行的案例。这些案例均采用不同的技术实现。其中,第 1 章图像处理系统,采用的是 Visual C++6.0 设计完成,同时应用了 GDI 实现图像的显示与转换技术;第 2 章局域网屏幕监控系统,采用的是 Visual C++6.0 设计实现,并使用了抓取屏幕保存为位图数据流和根据 JPEG 数据流显示图像技术;第 3 章网络五子棋,采用的是 Visual C++6.0 设计实现,并利用套接字进行网络传输;第 4 章书友会短信发送系统,采用的是 Visual C++6.0+Microsoft Access 2000 设计实现,并利用了“短信猫”发送短信技术;第 5 章企业仓库管理系统,采用的是 Visual C++6.0+SQL Server 2000 设计实现,并应用了菜单和工具栏的自绘技术;第 6 章客户管理系统,采用了 Visual C++6.0+SQL Server 2000 设计实现,并应用了加密锁加密技术;第 7 章文档管理系统,采用的是 Visual C++6.0+SQL Server 2000 设计实现,同时应用了调用 Word 文档技术;第 8 章商品销售管理系统,利用 Visual C++6.0+SQL Server 2000 设计实现,并应用了基于对话框程序的报表预览及打印技术。各章应用的核心技术如下图所示。





本书光盘内容

- ☒ 本书实例的源程序
- ☒ 环境安装视频录像
- ☒ 实例配置录像
- ☒ 编程词典体验版

本书读者

- ☒ 初学 VC 编程的自学者
- ☒ VC 编程爱好者
- ☒ VC 程序开发人员
- ☒ 毕业设计的学生
- ☒ 大中专院校的老师和学生
- ☒ 系统开发及维护人员

本书的约定

● 由于篇幅有限，本书的每一章并不能一一介绍案例中的每个模块。笔者选择的是基础和典型的模块进行介绍，对于功能重复的模块，由于技术、设计思路和实现过程比较相似，因此没有在书中体现。

● 本书光盘多媒体程序只能在光盘上执行，不能复制到计算机上执行。

技术支持

本书由明日科技的 Visual C++ 软件开发团队组织编写，主要编写人员有刘锐宁、宋坤、顾彦玲、梁晓岚、刘玲玲、刘欣、杨丽、黄锐、孙明姣、寇长梅、董大勇、张艳、郭佳博、乔敏、王敬杰、张金辉、李贺、张跃廷、房大伟、贯伟红、苏宇、刘彬彬、安剑、孙秀梅、高春艳、庞娅娟、孙明丽、吕继迪、邹天思、潘凯华、刘中华、孙鹏、王国辉、马文强、王小科、梁冰、吕双、尹相群、王毅、王殊宇、梁冰、李钟尉、赛奎春等。虽然在编写本书的过程中，我们本着科学、严谨的态度，力求精益求精，但疏漏之处在所难免，敬请广大读者批评指正。

如果读者在使用本书时遇到问题，可以访问明日科技网站，我们将通过明日科技网站全面为读者提供网上服务和支持。读者使用本书遇到的问题，我们承诺在 6 个工作日以内给您提供及时回复。

服务网站：www.mingribook.com

服务邮箱：mingrisoft@mingrisoft.com

服务电话：0431-84978981184978982

明日科技
2008 年 7 月



第1章 图像处理系统	1
1.1 概述	2
1.2 系统分析	2
1.2.1 需求分析	2
1.2.2 可行性分析	2
1.3 总体设计	2
1.3.1 项目规划	2
1.3.2 系统功能架构图	3
1.4 系统设计	3
1.4.1 设计目标	3
1.4.2 开发及运行环境	3
1.4.3 编码规则	3
1.5 技术准备	4
1.5.1 基本绘图操作	4
1.5.2 内存画布设计	5
1.5.3 自定义全局函数	6
1.5.4 自定义菜单	7
1.5.5 自定义工具栏	13
1.6 主要功能模块的设计	15
1.6.1 系统架构设计	15
1.6.2 公共模块设计	16
1.6.3 主窗体设计	18
1.6.4 显示位图模块设计	21
1.6.5 显示 JPEG 模块设计	23
1.6.6 显示 GIF 模块设计	25
1.6.7 位图转换为 JPEG 模块设计	27
1.6.8 位图旋转模块设计	30
1.6.9 线性变换模块设计	35
1.6.10 手写数字识别模块设计	37
1.7 疑难问题分析解决	44
1.7.1 读取位图数据	44
1.7.2 位图旋转时解决位图字节对齐问题	45
1.8 文件清单	45





第2章 局域网屏幕监控系统	47
2.1 概述	48
2.2 系统分析	48
2.2.1 需求分析	48
2.2.2 可行性分析	48
2.3 总体设计	48
2.3.1 项目规划	48
2.3.2 系统功能架构图	48
2.4 系统设计	49
2.4.1 设计目标	49
2.4.2 开发及运行环境	49
2.5 技术准备	49
2.5.1 套接字函数	49
2.5.2 套接字的初始化	53
2.5.3 获取套接字数据接收的事件	53
2.5.4 封装数据报	53
2.5.5 将屏幕图像保存为位图数据流	54
2.5.6 读写 INI 文件	55
2.5.7 使用 GDI+	56
2.6 主要功能模块的设计	56
2.6.1 客户端模块设计	56
2.6.2 服务器端模块设计	62
2.7 疑难问题分析解决	67
2.7.1 使用 GDI+产生的内存泄露	67
2.7.2 释放无效指针产生地址访问错误	67
2.8 文件清单	68
第3章 网络五子棋	69
3.1 概述	70
3.2 系统分析	70
3.2.1 需求分析	70
3.2.2 可行性分析	70
3.3 项目规划	70
3.4 系统设计	70
3.4.1 设计目标	70
3.4.2 开发及运行环境	71
3.5 技术准备	71
3.6 主要功能模块的设计	72
3.6.1 系统架构设计	72
3.6.2 公共模块设计	72
3.6.3 服务器端主界面	73



3.6.4	服务器设置	85
3.6.5	客户端主界面	86
3.6.6	设置服务器信息	93
3.7	疑难问题的分析与解决	95
3.7.1	如何获得 IP 地址	95
3.7.2	如何区分数据流中的不同信息	95
3.8	文件清单	96
第 4 章 书友会短信发送系统		97
4.1	概述	98
4.2	系统分析	98
4.2.1	需求分析	98
4.2.2	可行性分析	98
4.3	总体设计	98
4.3.1	项目规划	98
4.3.2	系统功能架构图	99
4.4	系统设计	99
4.4.1	设计目标	99
4.4.2	开发及运行环境	100
4.4.3	编码设计	100
4.4.4	数据库设计	100
4.5	技术准备	102
4.6	主要功能模块的设计	103
4.6.1	系统架构设计	103
4.6.2	公共模块设计	104
4.6.3	主窗体设计	105
4.6.4	系统登录	110
4.6.5	短信猫设置	112
4.6.6	会员信息设置	114
4.6.7	发送短信	118
4.6.8	投票项目	122
4.6.9	投票统计	127
4.7	疑难问题的分析与解决	130
4.7.1	如何在显示主窗体前显示登录框	130
4.7.2	如何设置按回车键实现焦点移动	130
4.7.3	如何设置 MSChart 控件的字体	130
4.8	程序调试与错误处理	131
4.8.1	插入数据不成功时如何进行调试	131
4.8.2	处理接收短信内容不正确的问题	131
4.9	测试与总结	131
4.9.1	测试的概念	131
4.9.2	测试的工作流程	132



目 录	4.9.3 测试方法·····	132
	4.9.4 系统测试的策略·····	132
	4.9.5 测试报告·····	134
	4.9.6 总结·····	134
	4.10 文件清单·····	134
	第5章 企业仓库管理系统 ·····	135
	5.1 概述·····	136
	5.2 系统分析·····	136
	5.2.1 需求分析·····	136
	5.2.2 可行性分析·····	136
	5.3 总体设计·····	136
	5.3.1 项目规划·····	136
	5.3.2 系统功能架构图·····	137
	5.4 系统设计·····	137
	5.4.1 设计目标·····	137
	5.4.2 开发及运行环境·····	137
	5.4.3 数据库设计·····	138
	5.5 技术准备·····	141
	5.6 主要功能模块的设计·····	142
	5.6.1 系统架构设计·····	142
	5.6.2 公共模块设计·····	143
	5.6.3 主窗体设计·····	144
	5.6.4 系统登录·····	159
	5.6.5 商品入库·····	160
	5.6.6 库存盘点·····	166
	5.6.7 入库查询·····	169
	5.7 疑难问题的分析与解决·····	174
	5.7.1 利用表格录入商品·····	174
	5.7.2 联想录入表格的实现·····	175
	5.8 程序调试与错误处理·····	178
	5.9 技术、经验总结·····	181
	5.9.1 技术总结·····	181
	5.9.2 经验总结·····	181
	5.10 文件清单·····	182
	第6章 客户管理系统 ·····	183
	6.1 概述·····	184
	6.2 系统分析·····	184
	6.2.1 需求分析·····	184
	6.2.2 可行性分析·····	184
	6.3 总体设计·····	184



6.3.1	项目规划	184
6.3.2	系统功能架构图	185
6.4	系统设计	185
6.4.1	设计目标	185
6.4.2	开发及运行环境	186
6.4.3	编码设计	186
6.4.4	数据库设计	186
6.5	技术准备	191
6.5.1	数据库的封装	191
6.5.2	封装 ADO 数据库的代码分析	191
6.6	主要功能模块的设计	195
6.6.1	系统架构设计	195
6.6.2	主窗体设计	196
6.6.3	系统登录	201
6.6.4	客户信息	204
6.6.5	联系人信息	208
6.6.6	客户反馈	215
6.6.7	联系人信息查询	217
6.7	疑难问题的分析与解决	221
6.7.1	使用 CtabCtrl 类实现分页的 2 种实现方法	221
6.7.2	ADO 不同属性和方法的弊端及解决方法	222
6.7.3	如何向加密锁模板写入数据	222
6.8	程序调试与错误处理	223
6.9	文件清单	224
第 7 章	文档管理系统	226
7.1	概述	227
7.2	系统分析	227
7.2.1	需求分析	227
7.2.2	可行性分析	227
7.3	总体设计	227
7.3.1	项目规划	227
7.3.2	系统功能架构图	228
7.4	系统设计	228
7.4.1	设计目标	228
7.4.2	开发及运行环境	228
7.4.3	数据库设计	228
7.5	技术准备	230
7.5.1	添加 ADO 连接类	230
7.5.2	添加数据库表的类	232
7.6	主要功能模块的设计	236
7.6.1	系统架构设计	236





7.6.2	主窗体设计	237
7.6.3	登录管理	243
7.6.4	单位档案	245
7.6.5	文档类别	250
7.6.6	文档管理	253
7.6.7	用户管理	260
7.6.8	口令修改	262
7.7	疑难问题的分析与解决	263
7.7.1	怎样将数据表中的数据添加到 ListControl 中	263
7.7.2	怎样取得文件完整路径	264
7.8	黑盒测试和白盒测试	265
7.8.1	黑盒测试	265
7.8.2	白盒测试	265
7.9	文件清单	265
第 8 章	商品销售管理系统	267
8.1	概述	268
8.2	系统分析	268
8.2.1	需求分析	268
8.2.2	可行性分析	268
8.3	总体设计	268
8.3.1	项目规划	268
8.3.2	系统功能结构图	269
8.4	系统设计	269
8.4.1	设计目标	269
8.4.2	开发及运行环境	269
8.4.3	数据库设计	269
8.5	主要功能模块的设计	272
8.5.1	系统架构设计	272
8.5.2	公共模块设计	273
8.5.3	主窗体设计	276
8.5.4	系统登录模块设计	280
8.5.5	操作员信息管理模块设计	282
8.5.6	商品入库管理模块设计	285
8.5.7	销售管理模块设计	289
8.5.8	查询管理模块设计	292
8.5.9	往来账管理模块设计	300
8.6	疑难问题分析解决	303
8.6.1	不显示文字的密码编辑框	303
8.6.2	数据库参数	303
8.7	程序调试及错误处理	303



8.8 技术、经验总结	306
8.8.1 技术总结	306
8.8.2 经验总结	306
8.9 文件清单	306





第1章

图像处理系统

程序说明

- 实例名称：图像处理系统。
- 实例位置：光盘\mr\图像处理系统\Program。
- 实例执行文件：光盘\mr\图像处理系统\Program\ImageHandle.exe。
- 注意事项：用户在使用本实例前，请仔细阅读光盘中的光盘使用说明。

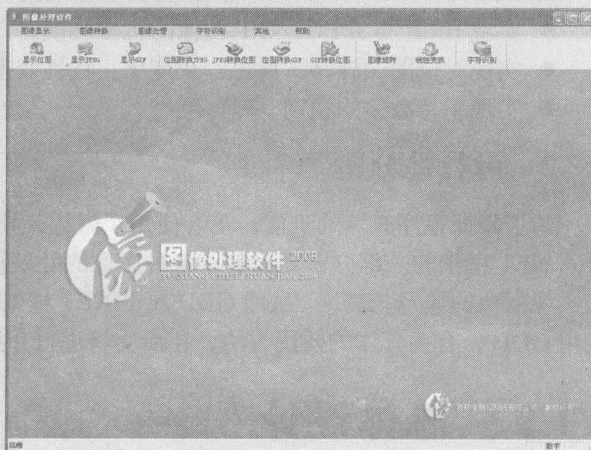
核心技术

在开发图像处理系统时，主要应用了以下技术。

- 通过从 CMenu 类派生子类来绘制彩色的菜单。
- 通过从 CToolBar 类派生子类来设计 Windows XP 风格的工具栏。
- 利用 GDI+实现图像的显示与转换。
- 设计一个内存设备上下文防止图像闪烁。

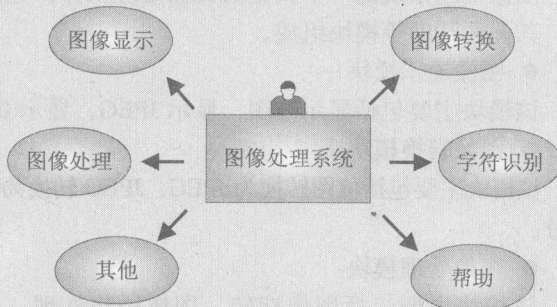
系统预览

图像处理系统的主界面如下。



系统功能

图像处理系统的系统功能结构图如下。





| 1.1 | 概述

图像处理技术一直是困扰 Visual C++ 初学者的一个难题, 在 Visual C++ 中, 没有一个完善的控件能够显示和处理各种格式的图像, 有时用户需要了解各种图像的文件格式, 这极大地增加了初学者的学习难度。基于此, 笔者设计了一个图像处理软件。

| 1.2 | 系统分析

1.2.1 需求分析

通过调查, 要求系统具有以下功能。

- 显示 BMP、JPEG、GIF 等各种格式的图像。
- 实现 BMP 与 JPEG、GIF 图像的批量转换。
- 实现对位图的旋转、反色、灰度化、线性变换等各种操作。
- 实现手写数字的识别。

1.2.2 可行性分析

为了降低程序开发的难度, 笔者使用了微软公司的 GDI+。GDI+ 又称为 GDIPlus, 是微软公司 .NET 框架的一部分, 它封装了各种图像的处理技术。通过使用 GDI+, 用户可以非常方便地实现图像的显示与转换。虽然 GDI+ 属于 .NET 框架的一部分, 但是用户可以在非 .NET 环境下使用 GDI+, 有关具体的使用方法, 在后文将进行介绍。

| 1.3 | 总体设计

1.3.1 项目规划

图像处理系统是一个典型的图像处理软件, 它由图像显示、图像转换、图像处理、字符识别、其他和帮助等模块组成。

- 图像显示模块

该模块主要包括显示位图、显示 JPEG、显示 GIF 3 个部分。

- 图像转换模块

该模块主要包括位图转换为 JPEG、JPEG 转换为位图、位图转换为 GIF、GIF 转换为位图 4 个部分。

- 图像处理模块

该模块主要包括图像旋转、图像锐化处理、图像反色处理、灰度化处理、线性变换 5 个部分。

- 字符识别模块

该模块主要包括手写数字识别。

- 其他模块

该模块主要包括设置菜单的固定颜色和随机颜色两个部分。

- 帮助模块



该模块主要包括软件的版本、开发单位等信息。

1.3.2 系统功能架构图

系统功能架构图如图 1.1 所示。

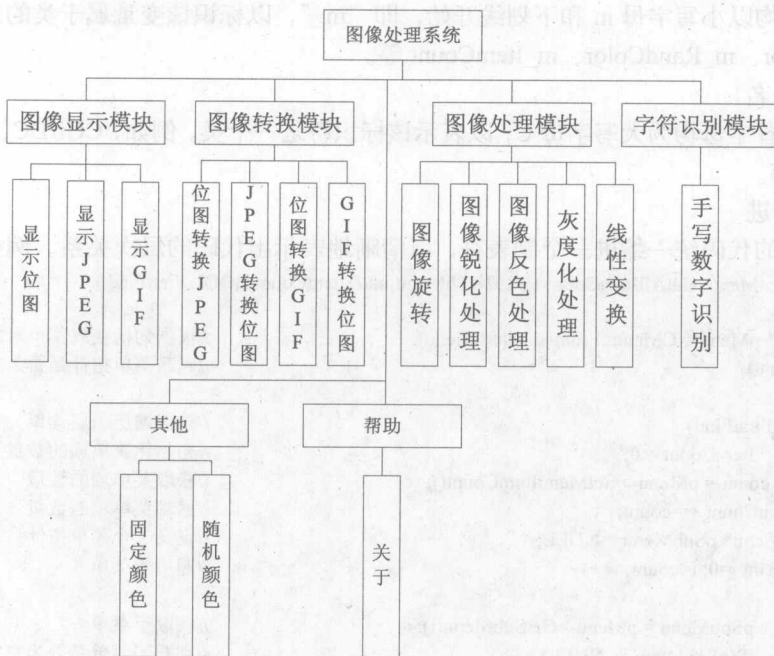


图 1.1 图像处理系统功能架构图

1.4 系统设计

1.4.1 设计目标

本系统属于小型的图像处理软件，主要用于实现图像的显示与批量转换操作。具体设计要求如下。

- 实现各种图像格式的显示。
- 系统采用良好的人机对话模式，界面设计美观、友好。
- 支持图像的单一转换与批量转换。
- 实现位图的各种常规操作。例如图像旋转、灰度化处理、线性变换等。
- 系统运行稳定、安全可靠。

1.4.2 开发及运行环境

系统开发平台：Visual C++ 6.0。

运行平台：Windows XP/ Windows 2000/ Windows 2003。

分辨率：最佳效果 1024 × 1280 像素。

1.4.3 编码规则

编码规则并不影响程序的功能，但是却关系代码的易读性。合理的编码规则能够方便开发



人员阅读和修改程序代码。因此，几乎所有的软件公司都有自己的编码规则。通常，不同语言都有各自的一些约定俗成的规则，因此，即使同一个软件公司，不同语言的编码规则也不尽相同。下面介绍图像处理系统常采用的编码规则。

- 成员变量的命名

成员变量均以小写字母 *m* 和下划线开始，即 “*m_*”，以标识该变量属于类的成员变量。例如，*m_BkColor*、*m_RandColor*、*m_ItemCount* 等。

- 类的命名

类的名称首字母均为大写字母 *C*，以表示该标识符是一个类。例如，*CGifDC*、*CGifToBmp*、*CCustomBar* 等。

- 代码缩进

不同层次的代码统一缩进一个制表符，以清晰地表示出代码的层次关系。例如：

```
void CCustomMenu::GetAllItemCount(HMENU hMenu,int &ItemCount,BOOL FirstFind)
{
    CMenu* pMenu = CMenu::FromHandle(hMenu);           //根据句柄获取菜单对象
    if (pMenu)                                           //判断菜单指针是否为空
    {
        if (FirstFind)                                  //是否遍历顶层菜单
            ItemCount = 0;                               //初始化菜单项的数量
        int count = pMenu->GetMenuItemCount();           //获取菜单项的数量
        ItemCount += count;                             //累加菜单项的数量
        CMenu* pSubMenu = NULL;                         //定义一个菜单指针
        for(int i=0; i<count; i++)                      //遍历子菜单
        {
            pSubMenu = pMenu->GetSubMenu(i);             //获取子菜单
            if (pSubMenu != NULL)                       //判断子菜单是否为空
            {
                //递归遍历所有的子菜单
                GetAllItemCount(pSubMenu->m_hMenu,ItemCount,FALSE);
            }
        }
    }
}
```

- 函数和方法的命名

函数和方法的命名使用动词或动词+名词的形式，并且首字母大写。例如，*AttatchMenu*、*GetAllItemCount*、*SetMenuItemInfo* 等。

- 指针变量的命名

指针变量以小写字母 *p* 标识。如果是指针类型的成员变量，则以 *m_p* 的形式开始。例如，*pMenu*、*m_pBmpData* 等。

1.5 技术准备

1.5.1 基本绘图操作

绘图操作通常是指在窗口的某个区域输出位图图像。窗口通常关联一个设备上下文 *CDC*，所有的绘图操作都是通过 *CDC* 来完成的。用户可以通过调用窗口类的 *GetDC* 方法获得一个 *CDC* 对象指针。*CDC* 提供 *BitBlt*、*StretchBlt* 等方法来输出位图，但是这些方法均需要借助另一个 *CDC* 对象来实现。以 *BitBlt* 方法为例，其语法为：

```
BOOL BitBlt(int x,int y,int nWidth,int nHeight,CDC* pSrcDC,int xSrc,int ySrc,DWORD dwRop);
```



其中, x、y 表示在窗口的哪个位置开始输出位图, nWidth、nHeight 表示输出位图的宽度和高度, xSrc、ySrc 表示在源位图的哪个位置开始输出, dwRop 表示图像的输出方式。这里重点介绍 pSrcDC 参数, 该参数表示源设备上下文, BitBlt 函数将依据该设备上下文来输出位图信息。因此, 在调用 BitBlt 函数之前, 源设备上下文 pSrcDC 必须具备 3 个条件, 一是源设备上下文必须是一个存在的设备上下文, 二是源设备上下文必须与窗口的设备上下文兼容, 三是源设备上下文必须选中一个位图对象。只有具备这 3 个条件, BitBlt 函数才能根据源设备上下文输出位图。下面的代码演示了如何设置源设备上下文, 并调用 BitBlt 方法输出位图。

```
CDC* pDC = GetDC();           //获取窗口的设备上下文
CBitmap bk;                   //定义位图对象
CDC memdc;                    //定义设备上下文
memdc.CreateCompatibleDC(pDC); //创建一个兼容的设备上下文
bk.LoadBitmap(IDB_CHESSBOARD); //加载位图对象
memdc.SelectObject(&bk);       //选中位图
pDC->BitBlt(0,0,600,600,&memdc,0,0,SRCCOPY); //调用 BitBlt 方法输出位图
```

1.5.2 内存画布设计

在绘制图像时, 如果频繁地进行多次绘图操作, 会出现闪烁的情况。为了防止闪烁出现, 可以设计一个临时的画布, 称之为内存画布, 让所有的绘图操作均在内存画布上进行, 然后在内存画布对象失去作用域时将其输出到目标设备上下文中, 这样就阻止了闪烁情况的出现。内存画布的设计非常简单, 所有源代码如下:

```
class CMemDC : public CDC           //定义内存画布
{
private:
    CBitmap* m_pbmp;                //定义位图对象指针
    CBitmap* m_poldbmp;             //定义位图对象指针
    CDC* m_pDC;                    //定义设备上下文指针
    CRect m_Rect;                  //定义区域对象
public:
    CMemDC(CDC* pDC, const CRect& rect) : CDC() //构造函数
    {
        CreateCompatibleDC(pDC);      //创建一个兼容的设备上下文
        m_pbmp = new CBitmap;          //创建位图对象
        //创建位图
        ① m_pbmp->CreateCompatibleBitmap(pDC, rect.Width(), rect.Height());
        m_poldbmp = SelectObject(m_pbmp); //选中位图对象
        m_pDC = pDC;                  //设置成员变量
        m_Rect = rect;                //设置成员变量
        //将目标区域绘制在内存画布上
        ② this->BitBlt(m_Rect.left, m_Rect.top, m_Rect.Width(), m_Rect.Height(),
            pDC, m_Rect.left, m_Rect.top, SRCCOPY);
    }
    ~CMemDC()
    {
        //在内存画布释放时将内存画布的内容绘制在目标区域中
        m_pDC->BitBlt(m_Rect.left, m_Rect.top, m_Rect.Width(), m_Rect.Height(),
            this, m_Rect.left, m_Rect.top, SRCCOPY);
        SelectObject(m_poldbmp);       //选中之前选中的位图对象
        if (m_pbmp != NULL)            //判断位图对象是否为空
            delete m_pbmp;             //释放位图对象
        DeleteObject(this);            //释放设备上下文
    }
};
```