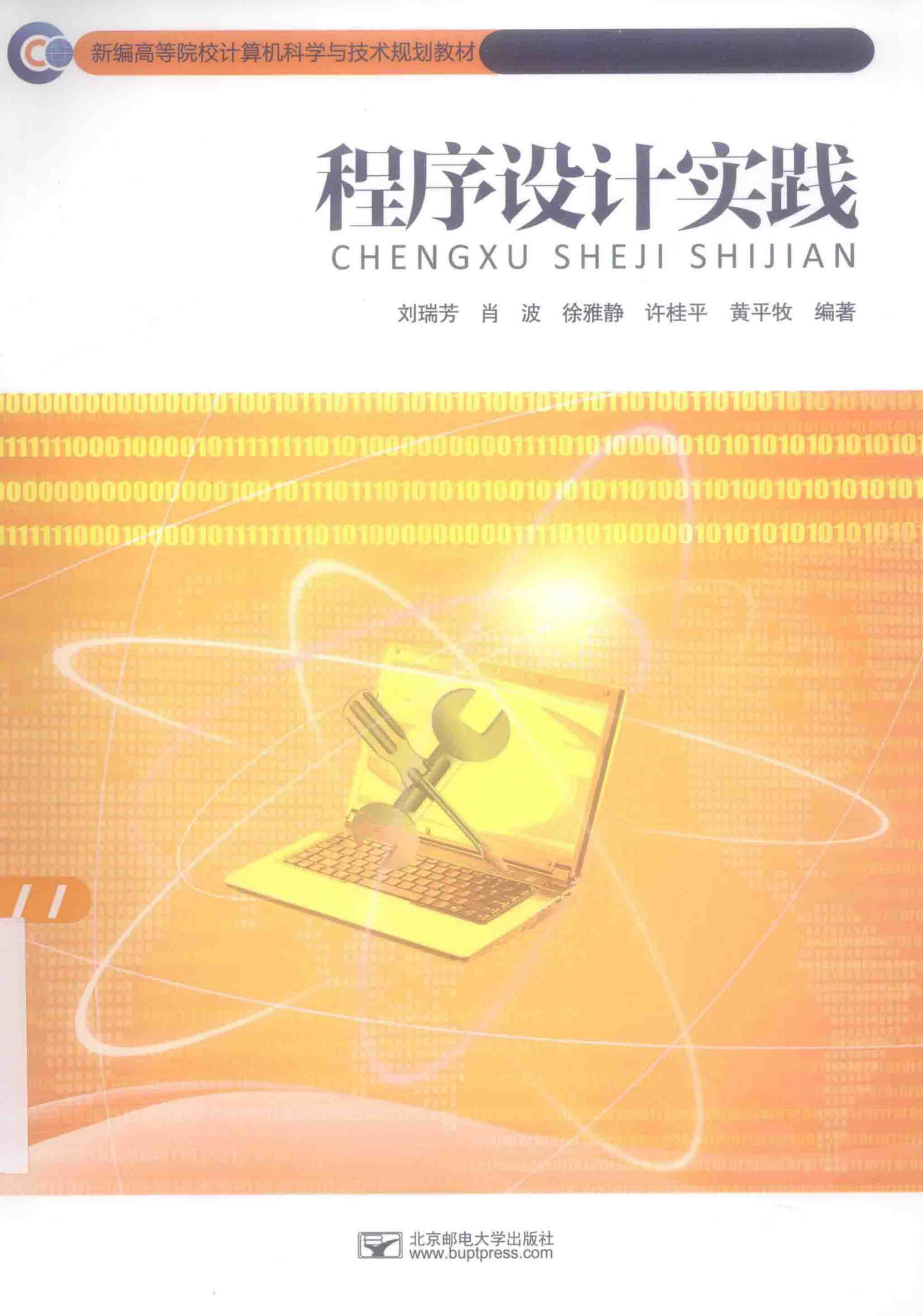


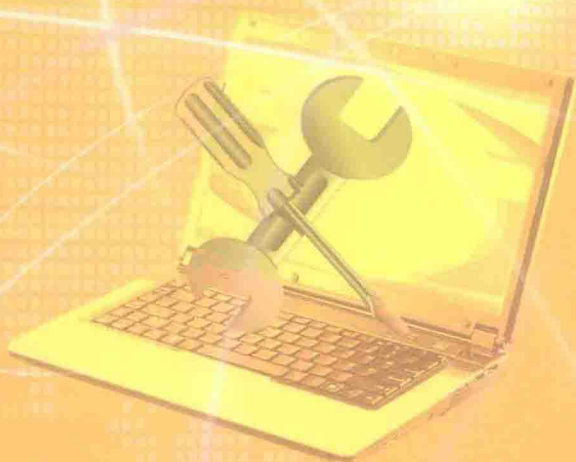


新编高等院校计算机科学与技术规划教材

程序设计实践

CHENGXU SHEJI SHIJIAN

刘瑞芳 肖 波 徐雅静 许桂平 黄平牧 编著



北京邮电大学出版社
www.buptpress.com

新编高等院校计算机科学与技术规划教材

程序设计实践

刘瑞芳 肖 波 徐雅静 编著
许桂平 黄平牧



北京邮电大学出版社

内 容 简 介

本书以 10 个章节的形式从不同侧面讲解了 10 个小型软件工程项目案例,从设计到实现,一步一步详细讲解,按照讲解过程进行操作即可达成项目目标。全书以 C++ 语言作为范例语言,在 Visual Studio 集成开发环境下进行编程实现。读者在学习 C++ 语言的基础上,完成一个项目即可,目标是学习分析问题、解决问题的方法,具备简单应用程序的设计、实现能力。

本书内容丰富,每章有项目拓展方向和要求,可作为电子工程、通信工程、信息工程类专业的程序设计实践课程的教材,也可供从事软件开发和应用的工程技术人员阅读和参考。

图书在版编目(CIP)数据

程序设计实践 / 刘瑞芳等编著. -- 北京:北京邮电大学出版社, 2015. 3

ISBN 978-7-5635-4290-1

I. ①程… II. ①刘… III. ①程序设计—教材 IV. ①TP311.1

中国版本图书馆 CIP 数据核字(2015)第 018226 号

书 名: 程序设计实践

著作责任者: 刘瑞芳 肖波 徐雅静 许桂平 黄平牧 编著

责任编辑: 刘 颖

出版发行: 北京邮电大学出版社

社 址: 北京市海淀区西土城路 10 号(邮编:100876)

发行部: 电话:010-62282185 传真:010-62283578

E-mail: publish@bupt.edu.cn

经 销: 各地新华书店

印 刷: 北京鑫丰华彩印有限公司

开 本: 787 mm×1 092 mm 1/16

印 张: 18.75

字 数: 486 千字

版 次: 2015 年 3 月第 1 版 2015 年 3 月第 1 次印刷

ISBN 978-7-5635-4290-1

定 价: 38.00 元

• 如有印装质量问题,请与北京邮电大学出版社发行部联系 •

前 言

本书的内容涉及:模块化程序设计、面向对象的程序设计、算法设计的基本方法;图形用户接口、窗口程序设计的基本知识;软件开发过程中文档的书写;集成开发环境的使用。

全书的内容分10章,讲解了10个小型软件工程项目案例,针对电子工程、信息工程等专业,希望有C++语言基础的读者,能够通过阅读本书把专业知识和程序设计实现结合起来,练就开发一定规模的工程项目的本领。每章以软件工程项目的形式进行讲解,有项目目标,有基础知识补充,有项目设计和实现,也有深入思考和拓展要求。本书内容全面涵盖了相关专业的各个研究方向,第1、2章讲解如何编写窗口程序,第3~6章讲解文本处理、通信协议、音频处理、图像处理,第7章讲解数据库应用程序,第8~10章讲解压缩编解码、加密解密、通信编码。

第1章和第6章采用Windows API接口实现窗口程序设计,开发了简单的绘图程序和图像处理程序。

第2章采用Windows API接口实现Win32控制台下的图形编程,编写多线程游戏程序。

第3章、第4章和第7章设计基于MFC对话框的应用程序,分别讨论文本处理、通信协议和数据库编程。

第5章设计基于MFC单文档的应用程序,实现了音频分析和处理程序。

第8章和第9章分别讨论压缩编解码和加密解密,以算法学习为主,以简单的控制台应用程序为实现方法。

第10章讨论通信中的编解码问题,以理解为主,以简单的控制台应用程序为实现方法。

全书以C++语言作为范例语言,在Visual Studio集成开发环境下进行编程实现,但丝毫不影响其他语言的爱好者以此作为升级读本,编程实现感兴趣的项目案例。

本书第1章和第5章由黄平牧老师编写,第2章和第9章由徐雅静老师编写,第3章和第6章由肖波老师编写,第4章和第10章由刘瑞芳老师编写,第7章和第8章由许桂平老师编写。感谢研究生张羽同学帮助编写加密解密程序实例和完成校对工作。

本书的示例程序和项目程序电子版可以通过北京邮电大学出版社的网站获得。

由于作者水平有限,书中难免有错误和缺点。在此欢迎广大读者和同行专家多提宝贵意见和建议,对书中错误疏漏之处批评指正,可直接将意见发送至lrf@bupt.edu.cn,作者将非常感谢。

作 者

目 录

第 1 章 绘图程序	1
1.1 项目分析和设计	2
1.1.1 需求分析	2
1.1.2 界面设计	2
1.1.3 总体设计	2
1.2 窗口程序基础知识	3
1.2.1 Windows API 基础	3
1.2.2 资源文件	8
1.3 绘图基础知识	9
1.3.1 图形处理技术	9
1.3.2 图形设备接口 GDI	11
1.4 绘图程序 Draw 的实现	21
1.4.1 基本图元类	21
1.4.2 绘图类	23
1.4.3 事件响应	27
1.4.4 钟表例子	29
深入思考	31
第 2 章 游戏程序	32
2.1 理论基础	32
2.1.1 相关数据结构	32
2.1.2 通用的系统函数	33
2.1.3 控制台相关的 API	34
2.1.4 编程技巧	38
2.1.5 多线程编程	41
2.2 俄罗斯方块游戏的设计和实现	44
2.2.1 需求分析	44
2.2.2 系统设计	45
2.2.3 详细设计	45
2.2.4 游戏实现	49
深入思考	56

第3章 中文机械分词	57
3.1 项目分析和设计	57
3.1.1 需求分析	57
3.1.2 界面设计	58
3.1.3 总体设计	58
3.2 分词基础	59
3.2.1 分词技术概述	59
3.2.2 词散列表构建设计	61
3.2.3 散列函数设计	62
3.2.4 散列处理类设计	64
3.2.5 词库文件处理类设计	68
3.3 后向最大匹配分词算法设计	69
3.3.1 分词类设计	69
3.3.2 分词算法设计	70
3.4 基于 MFC 对话框的分词程序实现	72
3.4.1 建立工程	72
3.4.2 对话框界面的实现	76
深入思考	83
第4章 简单通信协议	85
4.1 项目分析和设计	85
4.1.1 需求分析	85
4.1.2 界面设计	86
4.2 通信协议基础知识	86
4.2.1 基本概念	86
4.2.2 Socket 编程基础	89
4.3 SMTP	94
4.4 聊天程序的实现	99
4.4.1 总体设计	99
4.4.2 客户端的设计与实现	101
4.4.3 服务器端的设计与实现	105
深入思考	110
第5章 声音信号分析与处理	111
5.1 项目分析和设计	111
5.1.1 需求分析	111
5.1.2 界面设计	112
5.1.3 总体设计	114
5.2 理论基础	115

5.2.1	声音信号的数据结构	115
5.2.2	声音数据分析和处理	116
5.2.3	C++和 Matlab 混合编程	121
5.3	wavTool 的设计与实现	125
5.3.1	wavTool 的类关系图	125
5.3.2	自定义类 CWaveFile	126
5.3.3	文档类 CWaveDoc	137
5.3.4	视图类 CWaveView	140
	深入思考	144
第 6 章	图像处理程序	145
6.1	项目分析和设计	145
6.1.1	需求分析	145
6.1.2	界面设计	145
6.1.3	总体设计	145
6.2	BMP 图像基础知识	146
6.2.1	图像基本概念	146
6.2.2	BMP 图像基础	146
6.3	图像处理基础操作	149
6.3.1	BMP 文件的打开与保存功能实现	149
6.3.2	BMP 图像显示功能实现	152
6.3.3	图像处理类设计	153
6.4	简单图像处理程序的实现	163
6.4.1	绘图类	164
6.4.2	事件响应	166
6.4.3	消息响应函数	168
	深入思考	172
第 7 章	学生信息管理	173
7.1	项目分析和设计	173
7.1.1	需求分析	173
7.1.2	界面设计	174
7.1.3	总体设计	174
7.2	数据库基础知识	176
7.2.1	关系数据库常用术语	177
7.2.2	关系数据库设计	178
7.2.3	数据库管理系统	179
7.2.4	SQL 语言简介	181
7.3	数据库编程基础知识	182
7.3.1	数据库访问技术简介	182

7.3.2	MFC ODBC 类简介	184
7.3.3	使用 MFC ODBC 类访问数据库	185
7.4	学生信息管理程序 StudentInfo 的实现	204
7.4.1	主界面的设计与实现	204
7.4.2	学生基本信息管理模块	206
7.4.3	课程基本信息管理模块	207
7.4.4	学生选课信息管理模块	209
7.4.5	学生成绩信息查询模块	210
	深入思考	214
第 8 章	加密解密程序	215
8.1	加密技术简介	215
8.2	古典加密算法	216
8.2.1	凯撒密码	216
8.2.2	置换密码	218
8.3	对称密码 DES	222
8.3.1	DES 加密原理	222
8.3.2	DES 算法的实现	224
8.4	非对称密码算法 RSA	228
8.4.1	公钥密码体制	228
8.4.2	RSA 加密原理	228
8.4.3	RSA 加密算法的实现	229
8.5	消息摘要算法 MD5	232
8.5.1	消息摘要算法	232
8.5.2	消息摘要算法 MD5	233
8.5.3	MD5 消息摘要算法的实现	236
8.6	时域隐藏算法 LSB	240
8.6.1	信息隐藏技术	240
8.6.2	LSB 算法	241
8.6.3	LSB 算法的实现	241
	深入思考	248
第 9 章	压缩编解码	249
9.1	Huffman 压缩算法	249
9.1.1	Huffman 编码	249
9.1.2	Huffman 算法实现	250
9.2	基于字典的压缩算法	257
9.2.1	LZ77 算法	257
9.2.2	LZ78 算法	258
9.2.3	LZW 算法	260

9.2.4 LZW 算法实现	263
深入思考	269
第 10 章 通信编码	270
10.1 项目分析和设计	270
10.1.1 需求分析	270
10.1.2 系统设计	271
10.2 信源编码基础知识	273
10.2.1 0 和 1 的世界	273
10.2.2 理解信源编码	274
10.3 信道编码基础知识	275
10.3.1 理解信道错误	275
10.3.2 理解信道编码	275
10.4 信道模拟和编解码程序的实现	278
10.4.1 类的设计	278
10.4.2 类的实现	279
10.4.3 信源编码和汉明码类的使用	283
深入思考	285
附录 报告书模板	286

第1章 绘图程序

本章演示如何设计、实现一个可用于简单绘图的程序 Draw。Draw 是使用 Windows API (Application Programming Interface) 技术开发的。Windows API 是 Windows 系统和 Windows 应用程序间的标准程序接口,它实际上是为应用程序提供的一组标准函数(存储在 user32.dll, gdi32.dll 等.dll 文件中),利用这些函数可以调用 Windows 的系统功能,如窗口管理、图形设备接口、网络编程等。Windows API 是很多 Windows 开发语言封装的基石,剖析它就能了解到最基本的 Windows 编程技术,这对以后开发实际的 Windows 应用程序很有帮助。

Draw 在 Windows API 框架程序的基础上,定义了一组图元类,还有一个绘图类,利用这些新定义的类把 Windows API 的一些绘图功能进行了封装,可以为 Windows 应用程序提供高级绘图接口,简化用户程序的编写。

Draw 可以绘制由图元(点、线、圆、椭圆)构成的各种图形或动画,可以显示文字,可以响应菜单事件。后续可以在 Draw 的基础上,扩充其他的绘图功能。本章最终要实现的界面如图 1-1 所示,在主函数中编写简单的代码就可以实现一个钟表,如图 1-9 所示。本章涉及的知识包括:

Draw 在 Windows API 框架程序的基础上,定义了一组图元类,还有一个绘图类,利用这些新定义的类把 Windows API 的一些绘图功能进行了封装,可以为 Windows 应用程序提供高级绘图接口,简化用户程序的编写。

Draw 可以绘制由图元(点、线、圆、椭圆)构成的各种图形或动画,可以显示文字,可以响应菜单事件。后续可以在 Draw 的基础上,扩充其他的绘图功能。本章最终要实现的界面如图 1-1 所示,在主函数中编写简单的代码就可以实现一个钟表,如图 1-9 所示。本章涉及的知识包括:

- 窗口程序设计;
- 绘图接口函数;
- 类的派生与多态性。

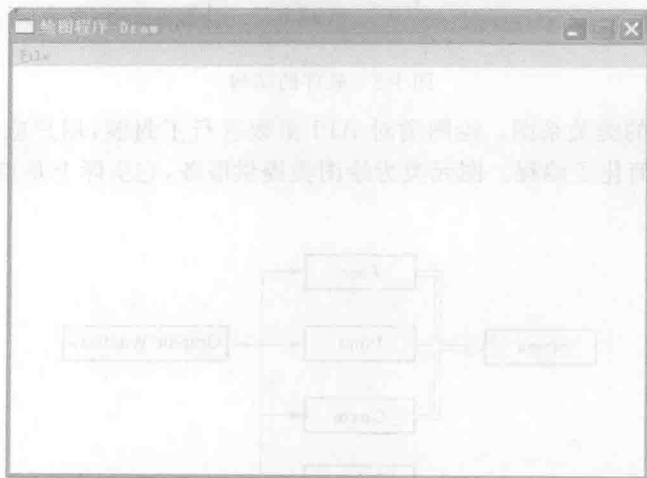


图 1-1 Draw 界面

1.1 项目分析和设计

1.1.1 需求分析

1. 功能需求

能绘制由基本图元(点、线、圆、椭圆、文字)组成的各种图形或动画,可以响应鼠标事件(菜单项)。

2. 界面要求

应用程序提供用户区供绘图使用,用户区可以显示坐标轴、原点等。

1.1.2 界面设计

在界面上可设置菜单,布置菜单项,按用户需求完成特定绘图功能,在绘图客户区可以显示坐标轴、原点等。

1.1.3 总体设计

为简化绘图应用程序的编写,Draw 在 Windows API 框架程序的基础上,定义了一组图元类,还有一个绘图类。利用这些新定义的类对 Windows API 的基本绘图功能进行了封装,目的在于为用户应用程序提供高级别的绘图接口,同时 Draw 还可以响应用户的键盘和鼠标操作,完成菜单功能、鼠标移动等。图 1-2 是软件的结构示意图。

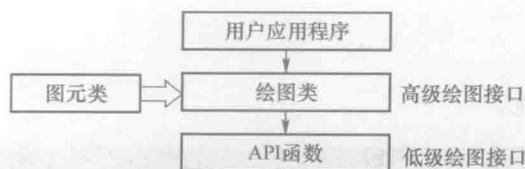


图 1-2 软件的结构

图 1-3 是 Draw 的类关系图。绘图类对 API 函数进行了封装,用户应用程序可以使用绘图类进行图形绘制,简化了编程。图元类为绘图类提供服务,它实际上是点类、线类、圆类和文本类的统称。

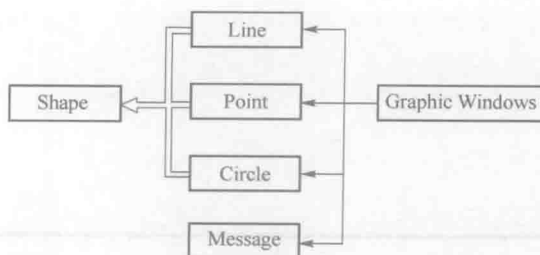


图 1-3 Draw 的类关系图

Shape 和 Line、Point、Circle 是继承关系,Shape 是基类,其他三个类为派生类; Point、

Line、Circle、Message 和 GraphicWindow 是依赖关系,GraphicWindow 依赖于其他几个类。

1.2 窗口程序基础知识

1.2.1 Windows API 基础

先介绍几个重要的概念——句柄、事件和消息,然后介绍 Windows 应用程序的基本架构,及其资源管理。

1. 句柄、事件和消息

句柄(handle)是整个 Windows 编程的基础,用于标识应用程序中不同的对象。比如一个窗口、图标、输出设备或文件,均对应着一个句柄。

单从概念上讲,句柄是指一个对象的标识,而指针是一个对象的首地址。但从实际处理的角度讲,既可以把句柄定义为指针,又可以把它定义为同类对象数组的索引。

句柄通常是一个 32 位的整数(32 位计算机系统)。

Windows 应用程序是基于事件驱动的,对于每个事件(如鼠标事件、窗口改变事件、定时器事件等),系统都将产生相应的消息。消息会被放入应用程序的消息队列中,然后应用程序将从消息队列中取出消息,最后分发给相应的窗口过程函数进行处理。所以事件和消息的概念是密切相关的,发生一个事件就会产生相应的消息,出现一个消息就意味着一个事件发生了。

从事件的角度讲,事件驱动程序的基本结构应由事件收集器、事件发送器和事件处理器组成。事件收集器收集所有发生的事件,包括来自用户的(如鼠标、键盘事件),来自硬件的(如时钟事件)和来自软件的(如操作系统、应用程序等);事件发送器将收集器收集到的事件分发到目标对象;事件处理器则完成最后的工作——具体事件的响应。

对于 Windows API 应用程序的编写者,他所能看到的是事件的分发(仅一点点)和事件的响应,而事件的响应则是编程的主要工作所在。

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;           //窗口句柄:用以检索消息的窗口句柄
    UINT    message;        //消息号:以事先定义好的消息名标识
    WPARAM wParam;         //字参数:消息的附加信息
    LPARAM lParam;         //长字参数:消息的附加信息
    DWORD   time;           //发送消息的时间
    POINT   pt;             //消息发送时,光标的位置
} MSG;
```

可以看出消息由三部分组成:消息号(message)、字参数(wParam)和长字参数(lParam)。消息以消息宏表示,如前缀为 WM 的消息宏表示的是窗口消息。窗口消息是最常用的 Windows 消息,主要有:

(1) 命令消息 WM_COMMAND

选择一个菜单项时,将产生这个消息。

(2) 鼠标消息

WM_LBUTTONDOWN,按鼠标左键产生的消息;

WM_RBUTTONDOWN,按鼠标右键产生的消息。

(3) 键盘消息

WM_KEYDOWN,按下一个键时,产生的消息;

WM_CHAR,字符消息。

(4) 窗口管理消息

WM_CREATE,CreateWindows 函数发出的消息;

WM_CLOSE,关闭窗口时产生此消息;

WM_DESTROY,清除窗口时,由 DestroyWindow 函数发出此消息。

(5) 其他窗口消息

WM_QUIT,退出应用程序时,由 PostQuitMessage 函数发出的消息;

WM_PAINT,窗口刷新时产生的消息。

Windows 消息处理的过程如图 1-4 所示。Windows 监视着所有的设备并将输入的消息(事件)放入系统的消息队列,然后将系统消息队列中的消息复制到相应应用程序的消息队列中。应用程序的消息循环便从消息队列中检索消息并将每一个消息发送到相应的窗口过程函数。

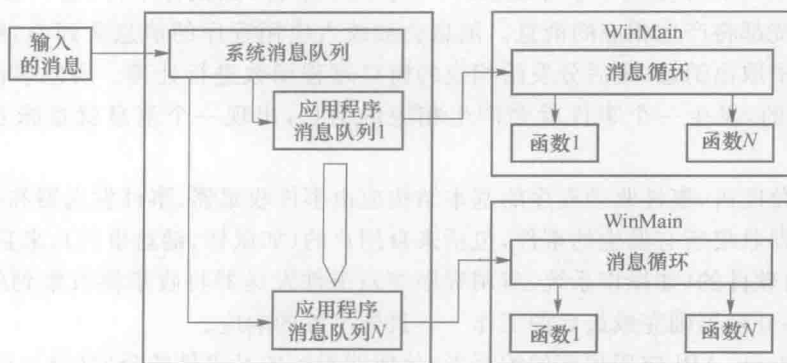


图 1-4 Windows 的消息循环

2. 基本的 Windows 应用程序

Windows 应用程序具有相对固定的基本结构,由入口函数 WinMain 和窗口过程函数等构成,WinMain 和窗口过程函数构成了 Windows 应用程序的主体。WinMain 函数中的消息循环部分,负责从应用程序消息队列中获取消息并将消息发送给相应的窗口过程函数,而窗口过程函数则决定接收一个消息时所采取的动作。

(1) WinMain 函数

WinMain 函数类似 C 语言中的 main 函数。其功能主要是完成一系列的定義和初始化工作,并产生消息循环。下面是 WinMain 函数的典型代码:

```
int APIENTRY WinMain(HINSTANCE hInstance,           //当前实例句柄
                    HINSTANCE hPrevInstance,        //前实例句柄
                    LPSTR      lpCmdLine,           //命令行参数
                    int         nShowCmd)           //指明窗口如何显示
{
    MSG msg;    //声明一个保存消息的变量
    WNDCLASS wndclass;
    //初始化窗口类,并向操作系统注册该窗口类
```

```

if (!hPrevInstance)
{
    wndclass.style = CS_HREDRAW | CS_VREDRAW;
    wndclass.lpfnWndProc = ccc_win_proc;           //指定窗口函数(消息处理入口)
    wndclass.cbClsExtra = 0;
    wndclass.cbWndExtra = 0;
    wndclass.hInstance = hInstance;
    wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION); //指定窗口类的图标
    wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);   //指定窗口类的光标
    wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
//指定填充窗口背景的画刷
    wndclass.lpszMenuName = NULL;
    wndclass.lpszClassName = "CCC_WIN"; //窗口类名称

    RegisterClass(&wndclass);
}
//初始化窗口,生成窗口并显示更新窗口
char title[80];
GetModuleFileName(hInstance, title, sizeof(title)); //得到当前运行程序的全路径
HWND hwnd = CreateWindow("CCC_WIN",
    title,
    WS_OVERLAPPEDWINDOW & ~WS_MAXIMIZEBOX,
    CW_USEDEFAULT,
    CW_USEDEFAULT,
    GetSystemMetrics(SM_CYFULLSCREEN) * 3 / 4,
    GetSystemMetrics(SM_CYFULLSCREEN) * 3 / 4,
    NULL,
    NULL,
    hInstance,
    0); //用注册的窗口类生成窗口实例

//显示窗口
ShowWindow(hwnd, nShowCmd);
UpdateWindow(hwnd); //若 Update Region 不空,则发送 WM_PAINT 消息

while (GetMessage(&msg, NULL, 0, 0)) //消息循环,对用户操作进行响应
{
    TranslateMessage(&msg); //将虚拟键消息转换成字符消息,再送入消息队列
    DispatchMessage(&msg); //分发消息,将消息队列中消息发送到窗口处理函数
}
return msg.wParam;
}

```

(2) 窗口过程函数

窗口过程函数是一个对每一个消息都进行处理的函数。其一般形式为带有多个分支的 switch 结构。通过把传送来的消息和系统中预定义的消息常量进行比较,从而判断消息的类型,执行不同的操作。其在应用程序中的一般形式如下:

```

LRESULT CALLBACK ccc_win_proc(HWND hwnd,
    UINT message,
    WPARAM wParam,
    LPARAM lParam)
{
    PAINTSTRUCT ps; //the display's paint struct

```

```

HDC mainwin_hdc;
switch (message)
{
case WM_PAINT: //窗口刷新消息
    mainwin_hdc = BeginPaint(hwnd, &ps); //置 Update Region 为空
    //ccc_win_main(); //调用用户的绘图程序

    EndPaint(hwnd, &ps);
    break;
case WM_DESTROY:
    PostQuitMessage(0);
    break;

default:
    //调用默认的消息处理函数，对其他的消息进行处理
    return DefWindowProc(hwnd, message, wParam, lParam);
}
return 0;
}

```

【例 1-1】编写一个简单的 Windows 程序，只有基本框架代码，没有具体功能。

解 使用 Visual Studio 2005 集成开发环境，创建项目的过程如下。

第一步：建立一个新的工程。

启动 Visual Studio，选择菜单项“文件”，单击“新建”→“项目”，在弹出的窗口的项目类型中选择“Win32”，继续选择“Win32 项目”，为项目起名“Draw”，如图 1-5 所示。



图 1-5 新建项目初始界面

单击“确定”后，在弹出窗口中进行应用程序设置，选择“Windows 应用程序”，选中“空项目”，然后单击“完成”，如图 1-6 所示。

第二步：设置字符集。

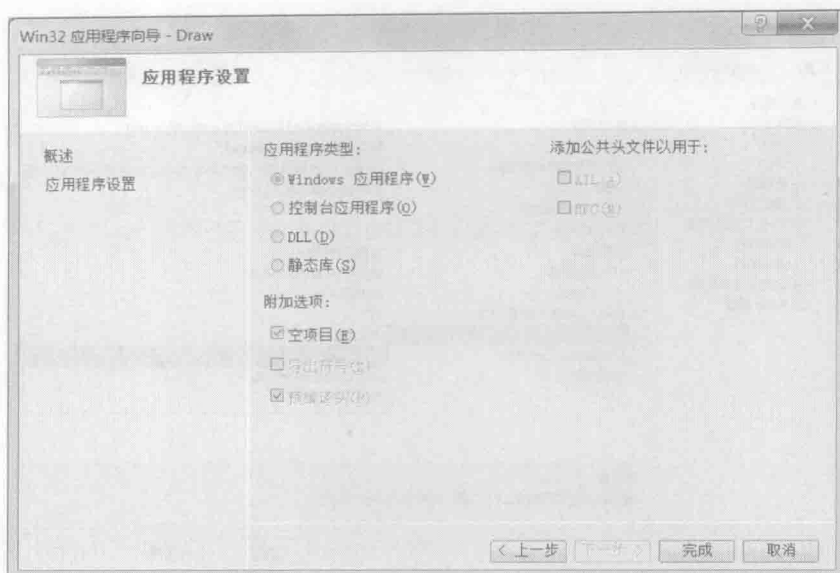


图 1-6 新建项目完成界面

在生成的解决方案名称上右击,在弹出的菜单选项中,选择“属性”,如图 1-7 所示。

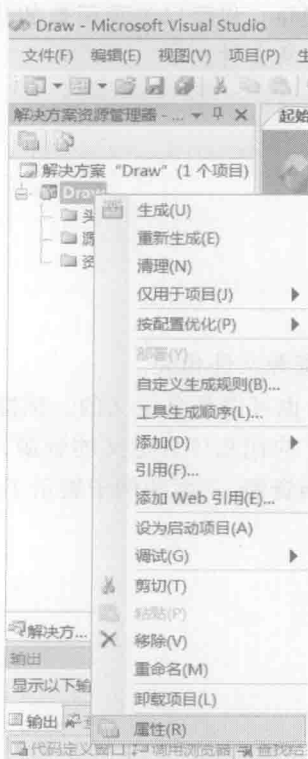


图 1-7 项目的属性菜单

在弹出的窗口中,选择“配置属性”→“常规”,单击“字符集”右侧下拉框,选中“未设置”,然后单击“确定”按钮,如图 1-8 所示。



图 1-8 设置字符集

第三步:为工程添加源文件/头文件/资源文件。

可以新建源文件,把上面 WinMain 和窗口过程函数的代码复制,也可以直接把本书附带的程序添加到工程里。要记得包含头文件。

```
#include <cstdlib>
#include "windows.h"
using namespace std;
#include "Resource.h"
```

然后就可以编译、运行了!

1.2.2 资源文件

Windows 应用程序的界面和资源文件相关。

Windows 资源可以是标准的,也可以是自定义的。标准资源中的数据描述了图标、光标、菜单、对话框、位图和加速键表等。应用程序自定义的资源,则包含任何特定应用程序需要的数据。Draw 仅涉及 Windows 标准资源,下面的例子展示了一个资源文件 fig.rc :

```
#include "resource.h"

//Menu
IDC_TEST2 MENU
BEGIN
    POPUP "&File"
    BEGIN
        MENUITEM "时钟",          IDM_FIG1
        MENUITEM "Fig2",          IDM_FIG2
        MENUITEM "E&xit",        IDM_EXIT
    END
END
```

资源文件 fig.rc 定义了一个 File 菜单,它包含有三个菜单项:时钟、Fig2 和 Exit。其中的