

实用第一 智慧密集 *Practicability First, Intelligence Intensive*

电脑编程技巧与维护

COMPUTER PROGRAMMING SKILLS & MAINTENANCE

<http://www.comprg.com.cn>

2003 年
合订本
(下)

《电脑编程技巧与维护》杂志社 编

内附光盘



学编程武艺技巧——跟我学编程、编程与应用起步、编程语言、编程课堂
可视化专栏、数据库

听编程专家论道——专家论坛（李维等）、编程疑难问题解答、博士信箱

解骇客点穴良药——计算机安全、计算机维护

和高手轻松过招——图形图像处理、网络技术、附录

```
void CTextViewerView::OnInitialUpdate()
{
    CView::OnInitialUpdate();
    CClient rectClient;
    GetClientRect( rectClient );
    if ( m_ctrlEdit ) delete m_ctrlEdit;
    m_ctrlEdit = new CEdit();
    m_ctrlEdit->Create( WS_MULTILINE | WS_CHILD | WS_VISIBLE
        | WS_VSCROLL | WS_HSCROLL | WS_AUTOVSCROLL,
        rectClient, this, 201 );
    CTextViewerDoc* pDoc = GetDocument();
    // 获取当前视图的客户区大小
    // 以下是将文档中的 m_strContent 内容全部赋给 str
    CString str;
    int nLines = (int)pDoc->m_strContent.GetCount();
    for ( int i=0; i<nLines; i++ )
    {
        str = str + pDoc->m_strContent.GetAt( i );
        str = str + "\r\n";
    }
    m_ctrlEdit->SetTabStops( 16 );
    m_ctrlEdit->SetWindowText( str );
}
```



机械工业出版社
China Machine Press

电脑编程技巧与维护

Computer Programming Skills & Maintenance

2003 年合订本（下）

《电脑编程技巧与维护》杂志社 编



机械工业出版社

《电脑编程技巧与维护》杂志始终以“实用第一，智慧密集”为宗旨，在业内获得一致好评。

《电脑编程技巧与维护》2003 年合订本仍保留该杂志的风格，按杂志的栏目重新编排，整套书包括跟我学编程、编程与应用起步、编程课堂、编程语言、专家论坛、可视化专栏、数据库、网络技术、图形图像处理、计算机维护、计算机安全、编程疑难问题解答、博士信箱等 13 个栏目；并添加了杂志没有的内容——以附录的形式给出 2003 年首届 Borland 程序员大赛精选文章及 VC/C++、VB、Delphi、VFP 等最新应用技巧和解决方案共 43 篇。本套合订本分为上、下两册，上册覆盖前 5 个栏目，文章 190 篇；下册覆盖后 8 个栏目和附录 B、C、D，文章 192 篇；并随书附赠一张光盘。

本书为广大的电脑编程爱好者、软件开发人员和专业计算机系统维护人员提供第一手的技术资料、编程技巧和维护经验，是广大计算机爱好者编程和维护人员的首选读物和必备参考资料。

图书在版编目 (CIP) 数据

《电脑编程技巧与维护》2003 年合订本/《电脑编程技巧与维护》杂志社编.

-北京: 机械工业出版社, 2004.1

ISBN 7-111-13684-5

I. 电… II. 电… III. ①程序设计-普及读物 ②电子计算机-维修-普及读物
IV. TP3-49

中国版本图书馆 CIP 数据核字 (2003) 第 119752 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

特邀策划: 曾 刚

责任编辑: 姜淑欣

版式设计: 郭新义

北京京丰印刷厂印刷·新华书店北京发行所发行

2004 年 1 月第 1 版第 1 次印刷

890mm×1240mm A4·70.25 印张·2903 千字

0 001—6 000 册

定价: 59.00 元 (上、下册, 含 1CD)

凡购本图书, 如有缺页、倒页、脱页, 由本社发行部调换

本社购书热线电话 (010) 68993821、88379646

封面无防伪标均为盗版

出版说明

◇ 杂志简介

《电脑编程技巧与维护》杂志自 1994 年创刊以来,始终以“**实用第一,智慧密集**”为宗旨,坚持“**质量第一**”的原则,为广大的电脑编程爱好者、软件开发人员和专业计算机系统维护人员提供第一手的技术资料、编程技巧和维护经验,努力为中国的软件业提供切实可行的技术解决方案,在业内获得一致好评,是广大编程和维护人员的首选刊物。在**栏目内容**上,选题覆盖面广,涉及技术领域宽、信息量大,帮助程序员开阔视野;在**技术水平**上,始终把握计算机技术发展的大方向,提供先进、详尽、准确的技术指导,并在长期的工作中与国际型大公司建立了良好的合作关系,为读者提供全球最新、最全的实用信息;在**实用性**上,稿源来自专业开发和维护人员的实践经验,以及普通书籍难以获得的编程心得体会与技巧。

◇ 内容提要

《电脑编程技巧与维护》2003 年合订本保留杂志的原有风格,不是 12 期内容的简单相加,而是按杂志的栏目重新编排,包括跟我学编程、编程与应用起步、编程课堂、编程语言、专家论坛、可视化专栏、数据库、网络技术、图形图像处理、计算机维护、计算机安全、编程疑难问题解答、博士信箱等 13 个栏目。另外,添加了杂志没有的内容——以附录的形式给出 2003 年首届 Borland 程序员大赛精选文章及 VC/C++、VB、Delphi、VFP 的最新应用技巧和解决方案共 43 篇。本套合订本分为上、下两册,上册覆盖前 5 个栏目和附录 A,文章共 190 篇;下册覆盖后 8 个栏目和附录 B、C、D,文章共 192 篇;并随书附赠一张光盘。

跟我学编程 / 编程与应用起步

两个栏目均是为广大初学电脑编程的读者提供一个相互学习、相互交流编程心得体会的园地,主要讲解 VB、VC/C++、Delphi、PB 等编程语言的基本编程知识;帮助读者学习使用这些编程语言,掌握其基本编程思路和方法;通过具体实例提供编程步骤;提供需要掌握的编程中的实战经验和小技巧等。

编程课堂

本栏目以专题讲座的形式,针对 Delphi、PowerBuilder (PB)、JBuilder 等内容进行系统全面的讲解,让编程开发人员“学之有道”,使其知识水平上升一个高度。

编程语言

本栏目主要介绍当今比较流行的编程语言(如 VB、VC/C++、Delphi、PB 等)的编程技术、技巧以及它们所提供的函数、变量等的灵活应用技术,旨在通过实用程序来讲解所应用的方法、技术与技巧。

专家论坛

本栏目提供一些在软件开发及应用的某一领域具有权威性的文章,编程人员可以在新的技术开发和应用领域有一定提高,同时也为编程人员相互学习、相互交流提供了一个论坛空间。

可视化专栏

本栏目主要介绍比较流行的可视化编程语言的发展动态和编程技巧,如 VB、VC/C++ 等应用开发的经

验和体会。通过提供一系列用 VB、VC++ 等可视化语言开发的实例，介绍其在系统开发、各种控件的制作、Web 库的管理等方面的应用。读者通过该栏目的学习将真正成为在可视化编程方面的高手。

数据库

本栏目主要介绍 Visual FoxPro、Delphi、SQL Server、Oracle 等数据库的新功能、新用法，以及利用这些数据库进行开发的经验、编程技术、技巧和方法。

网络技术

本栏目主要介绍网络的应用技术，包括网络操作系统和网络环境下的应用开发技术，即网页的制作技巧、Web 库的实现及管理、网站的建设和安全管理方面的技巧与方法，以及最新网站开发方面的实用技术。

图形图像处理

本栏目主要介绍图形图像软件的编制技术、技巧和编程的具体方法，包括软件界面的开发、各种图像格式的介绍及应用，如用 VC++ 等语言实现图像的浏览、位图的显示特技等。

计算机维护

本栏目涉及软件功能的扩充及软件故障的合理恢复方法，并介绍计算机硬件维护与维修的经验和体会，提供计算机用户自身修复硬件系统故障的实用方法。

计算机安全

本栏目主要介绍加密技术、反跟踪技术、网络安全及反病毒技术，例如具体病毒的检测和消除方法、如何有效地安装防火墙预防黑客。

编程疑难问题解答

本栏目主要对编程开发人员的实际技术难题进行答疑解惑，如在对 C++ 中如何读写文本文件和二进制文件、如何生成软件的注册号等实际问题都给出了准确、精彩的回答。

博士信箱

本栏目主要刊登计算机专家解答计算机应用方面的技术问题的文章，如 Windows 应用中常出现的问题及处理方法、各种打印机故障的维修、上网常见问题及其处理的技巧等系列文章。

附录

在囊括 2003 年杂志内容的基础上，本套合订本增加了一些杂志没有的内容，以附录的形式给出 2003 年首届 Borland 程序员大赛的精选文章，以及 VC/C++、VB、Delphi、VFP 等语言的最新应用技巧和解决方案 43 篇。

◇ 光盘内容

1. 《电脑编程技巧与维护》2003 年文章的程序源代码，所有源程序均经过调试通过。
2. 附录中，2003 年首届 Borland 程序员大赛精选文章以及 VC/C++、VB、Delphi、VFP 等语言的多篇最新应用技巧和解决方案的源代码，所有源程序均经过调试通过。

◇ 读者对象

本书为广大的电脑编程爱好者、软件开发人员和专业计算机系统维护人员提供第一手的技术资料、编程技巧和维护经验，是广大计算机爱好者编程和维护人员的首选读物和必备参考资料。只要您有志于深入研究计算机的高级技术，相信本书是您正确的选择。

目 录

出版说明

6. 可视化专栏

6.1	利用 VC 实现 DDE 服务/客户应用	1
6.2	借助内嵌资源实现 VC 对 Flash 动画的播放	3
6.3	VC 对多显示器系统的编程实现	5
6.4	用 VC 6.0 实现汉字到区位码的批量转化	6
6.5	VC 平台下实现 Foxmail 邮件转化器	8
6.6	在 VC 6 中利用 ClassWizard 调用 COM 组件	10
6.7	用 VC++ 实现与 USB 驱动程序的通信	11
6.8	用 VC++ 实现自画式菜单	13
6.9	用 VC++ 开发基于 ToolHelp32 的进程监控程序	19
6.10	用 VC++ 6.0 实现屏幕截取	21
6.11	深入剖析 VC++ 实现计时器的两种方法	23
6.12	VC++ 6.0 子用户界面线程及托盘的实现	26
6.13	Visual C++ 模态对话框消息处理机制的分析	29
6.14	用 VC++ 实现内存使用率的动态图形显示	31
6.15	在 Visual C++ 中使用托盘图标功能编写计算机定时关机程序	34
6.16	Visual C++ 中窗口子类化技术的实现及其应用	40
6.17	利用 VC++ 删除程序自己的方法	42
6.18	利用 Visual C++ 6.0 编写多功能数据处理显示程序	43
6.19	Visual C++ 6.0 实现语音采集与播放的控件	53
6.20	图像滤镜处理的一种 VC++ 实现方法	55
6.21	利用 VC++ 实现自定义工具栏	59
6.22	用 Visual C# 实现数字图像处理	61
6.23	在 VC#.NET 中利用 ADO.NET 实现树视图浏览编辑数据库	65
6.24	实现带图像和提示的 .NET 组合框组件	68
6.25	用 VC 设计实现可多次扩展的对话框	71
6.26	多线程中实现 MFC 成员函数的调用	74
6.27	如何在对话框中使用用户界面对象	76
6.28	多线程网络文件传输的设计与实现	79
6.29	利用 BCB 编写具有“磁性”特征的窗口	84
6.30	实现 Visual Studio 风格的窗口	86
6.31	VB.NET 查看文件中的图标——浅谈 VB.NET 中的 WinAPI 调用	88
6.32	Pop-Up Menu 在非模态对话框中实现 UPDATE_COMMAND_UI 机制	92

7. 数据库

7.1	利用 Visual Basic 6.0 编写通用的数据库访问组件	96
-----	--	----



7.2	VB 6 在远程拨号网络环境下实现数据库的连接与数据传输	99
7.3	在 VB 6/VB.NET 中灵活设置数据库连接	105
7.4	使用 VC++ 实现三层架构的分布式数据库应用框架	109
7.5	利用 VC++ 实现动态 ODBC 应用	111
7.6	实现 Visual C++ 与各种数据库系统共享	113
7.7	用 VC++ 的 ADO 技术访问数据库实例	117
7.8	OCI 接口简介及其在 VC++ 中的应用	123
7.9	用 ADO 实现大型二进制数据在数据库中的存取	132
7.10	基于 COM 的 ADO 数据库技术深入编程	133
7.11	Visual C#.Net 数据库编程之五——实现数据图表	136
7.12	Visual C#.Net 数据库编程之六——使用 Crystal Report 做报表	138
7.13	Visual C#.Net 数据库编程之七——通过 Web Service 来更新数据	143
7.14	Delphi 中通过 ADO 组件使用原生 ADO 对象	147
7.15	Delphi 6 存取 SQL Server 2000 中 JPEG 图像	148
7.16	ADO 记录集向 XML 文档的转换	152
7.17	在 ASP.NET 中如何通过 XML 实现异构数据的转换	155
7.18	Oracle 中层次查询的使用和探讨	157
7.19	基于 C/S 结构的 JAVA 网络数据库编程	162
7.20	.NET 数据库应用程序中存储过程的应用	165
7.21	创建基于 .NET 的通用数据库访问层	169
7.22	在 Kylix 中使用共享对象库技术	173
7.23	利用 SMI 表实现 INFORMIX 数据库大批量自动备份	177

8. 网络技术

8.1	运用 C# 编写 Pop3 组件	180
8.2	一种基于非对称多播网络的传输控件设计	184
8.3	基于 Linux 环境下 Sniffer 设计与实现	187
8.4	WinInet 类集开发 Internet 客户应用程序探索	192
8.5	用 SOAP Toolkit 开发 Web Service	194
8.6	实现 DOS 和 Windows 应用程序之间 IPX 实时数据通信	198
8.7	Matlab 环境下 Socket 网络功能的实现	201
8.8	用多视频文件测试网络系统带宽	203
8.9	基于客户端脚本构建多层事务性 Web 应用系统	211
8.10	基于 DHTML 页面的数据绑定技术的实现	217
8.11	IIS 集成编程技术	219
8.12	用 WebSnap 实现分布式 Web 图表在线显示	223
8.13	用 VB 制作自己的网络游戏	224
8.14	如何用 Visual C++ 编写 32 位串行通信程序	227
8.15	应用 Java 和 JSP 设计完整的文件上传系统	229
8.16	用 C++ 创建 ASP 服务器组件	233
8.17	Visual C# 实现网络对时系统——UDP 协议的实际应用	236
8.18	用 ActiveX 控件创建具有 FTP 功能的网页	243
8.19	Linux 下的套接口低层协议编程及其应用实例	246
8.20	VC++ 编程实现网络嗅探器	248



8.21 基于 URL 的 Web 服务器数据访问	251
8.22 利用 Winsock 编程实现局域网上所有 IP 包的捕获和分析	252
8.23 基于 IP 的网络状态实时监测系统的设计与实现	254
8.24 如何利用局域网在大屏幕上显示体育比赛成绩	256
8.25 多串口通信的软件实现方法	262
8.26 Linux XWindows 环境下用 Kylix 3 (C++) 实现串口通信	264
8.27 通用型 Web 文件上传 JavaBean 的实现	268

9. 图形图像处理

9.1 用 VB 实现马赛克	271
9.2 用 VB 与 VC 混合编程开发图像处理系统	273
9.3 在 VC 中研究双三次 B 样条曲面	276
9.4 VC++ 与 Matlab 结合方法的分析	279
9.5 用 VC++ 实现图像连通区域标记	282
9.6 VC++ 6.0 中利用 OpenGL 实现 3DS 模型的交互控制	285
9.7 使用 VC++ 绘制矢量图	289
9.8 VC++ 6.0 中利用 OpenGL 实现树木建模	291
9.9 多背景位图动画的原理及实现	293
9.10 基于 ObjectARX 2000 的参数化绘图	296
9.11 在 Windows 窗口中擦除位图背景的程序设计	299
9.12 利用 API 函数在 Windows 98 中实现位图的淡入淡出效果	300
9.13 BMP 位图文件的存储格式	302
9.14 SIMULINK 动态系统三维仿真结果可视化的研究	305
9.15 指纹识别的二次开发应用	307
9.16 一类基于不规则多边形的演示动画制作方法	310
9.17 基于 Linux 的文本与图像打印算法分析	313
9.18 利用 MATLAB 计算任意二元函数的极小值	316
9.19 ASP.NET 下自定义 Web 图像控件的开发	317
9.20 Win 2000 下屏幕图像的快速传输	320
9.21 全屏取色功能的钩子实现	323

10. 计算机维护

10.1 调用 Windows API 函数实现自动修复注册表中 IE 项的值	326
10.2 注册表操作技术全攻略	328
10.3 用 C# 进行注册表编程	330
10.4 利用动态链接库 DLL 加强 Oracle 数据库用户口令的安全	331
10.5 浅析硬盘的主引导扇区	335
10.6 信息系统应用软件自动升级的方法与工具	339
10.7 通过菜单实现 PowerBuilder 应用系统的安全性	343
10.8 用 VC++ 实现程序自删除的一种方法	347
10.9 利用网络高效维护机房	349
10.10 用 Web 方式维护 Windows NT 用户密码及相关组件的设计	352
10.11 网络环境下, 通过遍历注册表获取“添加/删除程序”对话框中的信息	356



10.12	为 Win 9X 系统增加一个开机密码.....	358
10.13	编程实现 CAB 文件压缩.....	363
10.14	Visual C++ 下利用 C++ Test 进行软件单元测试.....	367
10.15	获取网卡 MAC 地址、硬盘序列号控件的建立及应用.....	371

11. 计算机安全

11.1	在 Visual C++ 6.0 下利用 Rijndael 算法进行任意文件加密.....	374
11.2	用 VC++ 实现数字图像迭代混沌加密方法.....	376
11.3	Delphi 单机数据库系统的安全性研究与应用.....	378
11.4	基于 Linux 环境下的包过滤防火墙设计与实现.....	381
11.5	适用 VxD 实现 Windows 98 环境下的文件和目录隐藏.....	390
11.6	在 Linux（UNIX）中配置 sendmail 阻止垃圾邮件的发送和接收.....	396
11.7	基于文件时间信息和 IDEA 算法的数据加密.....	398
11.8	MIDAS 数据安全.....	401
11.9	通过编码方案实现系统的权限设置.....	402
11.10	软件加密之反复制技术.....	404
11.11	利用 Windows Hook 技术监控计算机的使用.....	407
11.12	com 文件的防 debug 跟踪.....	409
11.13	硬盘逻辑锁原理.....	410
11.14	也谈利用 DLL 加强用户口令安全的问题.....	416
11.15	软件加密解密之动态跟踪.....	418
11.16	.NET 框架下 DES 加密/解密程序的实现.....	420
11.17	一种基于 Web 的密码访问驱动器或文件夹的方法.....	424
11.18	基于 DES 算法一次一密加密系统的 JAVA 编程.....	426

12. 编程疑难问题解答

12.1	使用 Power Builder 开发应用程序过程中遇到的一些问题及解决.....	430
12.2	如何设计实现 Win 2000 下网络监控平台软件.....	431
12.3	如何提高 SQL 语言中连接查询和嵌套查询的效率.....	433
12.4	如何在 PB 中对数据进行实时查询.....	435
12.5	如何用 VB 6 破解 ACCESS 2000 数据库密码.....	436
12.6	解决 ORA-25251: 如何在 Oracle 的表空间中建立回滚段.....	439
12.7	如何对 MFC 树形控件（CTreeCtrl）进行扩展.....	441
12.8	如何用 VB 6.0 编程实现 Excel 2000 表格转化为 Access 2000 数据库表.....	442
12.9	如何在 Windows98 系统中使用 UCDOS 输入法.....	444
12.10	如何用 PB 实现通用的右键菜单程序.....	447
12.11	Tcl 如何与自身及 VB 通信的 6 种方式（Clipboard, DDE, Socket, FTP, SMTP, POP3）.....	448
12.12	在 ASP.NET 中通过 OWC 把数据导入到 EXCEL.....	451
12.13	怎样解决 Paradox 数据库转换过程中几个常见问题.....	454
12.14	如何用 PRO*C 程序实现对 ORACLE 数据库的访问.....	455
12.15	如何用 ASP 获得客户端的 IP 地址.....	456
12.16	在 C++ 中如何读写文本文件和二进制文件.....	457
12.17	.Net 框架下多线程编程有哪些方法和步骤.....	459



12.18 如何灵活运用 VFP 中的类.....	461
12.19 如何制作具有自动恢复功能的通知栏图标控件.....	461
12.20 PB 6.5 中如何处理 Excel 中文报表	463
12.21 如何解决关系数据库传统方法中的缺陷	464
12.22 如何自由访问局域网中的所有 SQL Server 服务器的数据.....	465
12.23 如何生成软件的注册号	466
12.24 VFP 如何按任意组合条件查询表单.....	468
12.25 如何利用控件进行日期编辑与查询	469
12.26 如何在 Win32 系统中引导 Java 程序.....	470
12.27 如何输出反像的汉字	472
12.28 如何备份与还原 SQL SERVER 2000 数据库.....	473
12.29 如何防止应用程序的多个实例运行	474

13. 博士信箱

13.1 上网技巧与常见问题处理	475
13.2 计算机基本工具软件实用技巧与常见问题处理.....	487

附录 B

B.1 VB 中程序执行进度指示器的实现	497
B.2 VB 6.0 中存取数据库中图片的方法	498
B.3 在 VB 开发的应用程序中多媒体的实现方法	499
B.4 VB 数据库编程中数据输出到 Excel 的两种实现方法	501
B.5 用 VB 实现计算机通信.....	503
B.6 VB 实现控制进程.....	505
B.7 在 VB 中如何建立独立于分辨率的 Form.....	507

附录 C

C.1 Delphi 6 中调用 DOS SHELL 命令实现网络编程.....	510
C.2 Delphi 6 内利用 API 函数对鼠标和光标操作.....	511
C.3 用 Delphi 6 实现收藏夹	513
C.4 用 Delphi 做一个简易上传工具.....	514
C.5 Delphi 动态报表中随机备注的实现.....	516
C.6 用 WebSnap 实现分布式 Web 图表在线显示	518
C.7 Delphi 中快速实现数据库树形结构并实现 Treeview 导航表数据	522
C.8 Delphi 中用多线程实现日志文件的读写与管理	526
C.9 用 Delphi 建立通信与数据交换服务器——Transceiver 的实现.....	532
C.10 利用 Delphi 6 制作多用户文件的备份和传递	538

附录 D

D.1 Visual FoxPro 6.0 中图片的输出方法	544
--------------------------------------	-----



D.2	在 VFP 中访问 DLL 获取磁盘结构信息	545
D.3	VFP 打印超宽报表的实现方法	547
D.4	VFP 编程访问 MS SQL Server 数据库-VFP SQL pass-through 技术应用	550
D.5	用 VFP 5.0 中的 OLE 绑定控件显示动态图形	552
D.6	VFP 6.0 数据访问中的 ODBC 和 ODBC 配置技术探讨	554
D.7	VFP 6 中基于照片数据处理的技巧方法	557
D.8	在 VFP 中用低级函数分割文件的方法	559
D.9	VFP 查询命令 SELECT-SQL 在多表查询中的应用	561

6. 可视化专栏

6.1 利用 VC 实现 DDE 服务/客户应用

周 江

摘 要 介绍 DDE 应用的文章很多,但大多是利用 VB、PB 等“高级”开发工具,应用也多是客户程序。本文将从使用 VC 的角度解释 DDE 的概念和工作原理,并给出实现 DDE 服务/客户应用的实例。

关键词 DDE, 服务/客户, 进程间通信

一、DDE 简介

自从 Microsoft Windows 2.1 版发布以来,动态数据交换 (DDE) 就成为 Windows 的一部分,并且许多应用程序都采用了 DDE 技术, DDE 作为一种基本机制已经应用于对象链接与嵌入 (Object Linking and Embedding, OLE) 的 1.0 版中,众所周知,OLE 已经升级到 ActiveX、COM 等,但 DDE 在许多应用程序中仍得到了普遍的支持,如 Microsoft Office、Acrobat Reader 和众多的组态软件,也许只是向下兼容的缘故,但 DDE 仍可作为应用程序间 (本地、远程) 通信的一种方式。

Windows 消息虽然是在不同程序窗口间传送信息的最佳手段,但一条消息只能包含两个参数 (wParam 和 lParam),不能传送较多的信息。DDE 正是建立在 Windows 内部消息系统、全局和共享全局内存基础上的一种协议,用来协调 Windows 应用程序之间的数据交换和命令调用。

二、DDE 原理

DDE 应用程序可以分为 4 种类型:客户、服务器、客户/服务器和监视器。DDE 会话发生在客户应用程序和服务程序之间。客户应用程序从服务器应用程序请求数据或服务,服务器应用程序响应客户应用程序的数据或服务请求。客户/服务器应用程序则是它既可发出请求又可提供信息。监视器应用程序用于调试目的。

DDE 协议使用三级树型命名:服务 (Service)、主题 (Topic) 和数据项 (Item) 来标识 DDE 所传递的数据单元。服务使应用程序具有了提供给其他程序的数据交换能力;主题类似于目录,是建立会话连接的参数,但一般无法知道服务器确切有哪些主题,除非查阅应用程序的相关技术文档;顾名思义,ITEM 才是具体通信的数据内容,例如一个字符串。

动态数据交换管理库 (Dynamic Data Exchange Management Library, DDEML) 中提供了 DDE 和应用程序级协议。使

用 DDEML 开发的应用程序无论在运行一致性方面还是在应用程序相互通信方面性能均优于没有使用 DDEML 的应用程序,而且 DDEML 的应用使得开发支持 DDE 的应用程序容易了许多,本文中采用它来实现应用。

建立 DDE 会话后,客户程序和服务器程序可通过 3 种链接方式进行数据交换,分别为:(1)冷链接 (Cold Link):客户程序申请数据,服务器程序立刻给客户程序发送数据;(2)温链接 (Warm Link):服务器程序通知客户程序数据项发生了变化,但并没有将已变化的值立即发送给用户程序;(3)热链接 (Hot Link):当数据项发生变化时,服务器程序立即把变化后的值发送给客户程序,这是最常用、最方便的方法,本文采用此方式,其流程及使用的 API 函数简介如下,详见后面的实例或 MSDN。

(1) DDE 会话初始化。使用 API 函数 DdeInitialize,在 DDEML 中注册应用。

(2) 会话建立。服务器:注册服务 DdeNameService。客户:连接 DdeConnect。

(3) 会话过程。类似于 Windows 的消息循环,会话的过程就是处理事务的过程。

客户通过 DdeClientTransaction 来发出事务请求,通过 DDE 回调函数,服务器处理客户事务请求,返回 DdeCreateDataHandle 来发送数据,同时客户可以通过调用 DdeGetData 获取数据。

(4) 会话结束。可由服务方或客户方来终止会话,退出程序时要注销服务,释放资源,调用 DdeUninitialize。

三、实例介绍

下面通过一个具体的简单实例来说明各项功能的实现。设计说明:服务器端有两个数据项,一个是输入的字符串,另一个是定时增加的整数,运行两个实例 (运行两次) 后,两实例即可建立 DDE 连接,同时在 Client Group Box 控件里的两个文本框中显示对方的数据。

用 VC 新建一个基于对话框的 MFC 应用程序, 取名为 DDEdemo, 放置编辑控件 IDC_EDIT, 用 Wizard 添加对应成员变量 m_edit (CString 类型), 添加静态控件如图 1 所示, ID 分别为 IDC_STATIC1、IDC_STATIC2、IDC_STATIC3, 将其 Cation 置空, 最后添加 Group Box。

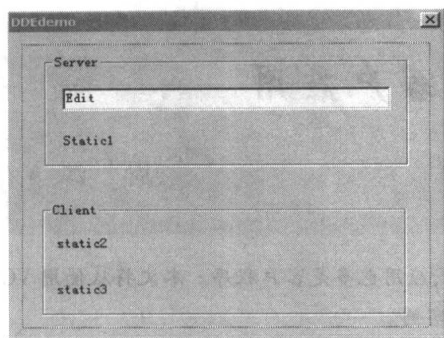


图 1

使用 Wizard 添加对话框函数分别处理消息 WM_DESTROY、WM_INITDIALOG、WM_TIMER 及 IDC_EDIT 的 EN_CHANGE, 建立对应函数。

在 DDEdemoDlg.cpp 中加入 #include "ddecl.h" 以使用 DDECL 函数。

(1) 添加以下宏定义和全局变量。

```
#define NITEM 2 //ITEM 数量
const char szApp[] = "Server"; // server DDE 服务名
const char szTopic[] = "Topic"; // server DDE 目录名
const char *pszItem[NITEM] = { "Item1", "Item2" };
// SERVER ITEM 名称字符串数组
int count = 0; //计数, 在 Static1 中显示
CString ServerData[NITEM]; //存放服务器中的数据项内容
HCONV hConv = 0; //会话句柄
HWND hWnd; //窗口句柄
DWORD idInst = 0; // DDECL 实例 ID
HANDLE hInst; // 实例句柄
HSZ hszApp = 0; // server 服务字符串句柄
HSZ hszTopic = 0; // server 目录字符串句柄
HSZ hszItem[NITEM]; // server ITEM 字符串句柄
BOOL bConnect; //建立连接标志
//由于本例是同一程序通信, 故 DDE 服务 客户参数命名相同, 但实际使用时应不同。
```

(2) 在 CDDEdemoDlg::OnInitDialog() 中添加以下代码实现初始化。

```
hWnd = m_hWnd;
if (DdeInitialize(&idInst, //实例的 ID
    (PFNCALLBACK) DdeCallback, //回调函数指针
    APPCMD_FILTERINITS | //初始化的参数
    CBF_FAIL_EXECUTES | //不响应
    XTYP_EXECUTE
    CBF_SKIP_CONNECT_CONFIRMES | //无需连接确认
    CBF_FAIL_SELFCONNECTIONS | //禁止连到实例
    CBF_FAIL_POKES, //禁止客户传送数据
```

```
0); //保留(置 0)
{
    MessageBox("DDE SERVER 初始化失败!");
    return(FALSE);
}
hInst = AfxGetApp() -> m_hInstance;
//创建 DDE string
hszApp = DdeCreateStringHandle(idInst, szApp, 0);
hszTopic = DdeCreateStringHandle(idInst, szTopic, 0);
for(int i = 0; i < NITEM; i++)
    hszItem[i] = DdeCreateStringHandle(idInst, pszItem[i], 0);
//注册服务
DdeNameService(idInst, hszApp, 0, DNS_REGISTER);
bConnect = FALSE;
SetTimer(1, 1000, NULL); //开定时
(3) 在 CDDEdemoDlg::OnDestroy() 添加以下代码完成结束会话的处理。
```

```
KillTimer(1); //销毁定时
DdeNameService(idInst, 0, 0, DNS_UNREGISTER); //注销服务
```

```
//释放 DDE strings
DdeFreeStringHandle(idInst, hszApp);
DdeFreeStringHandle(idInst, hszTopic);
for(int i = 0; i < NITEM; i++)
    DdeFreeStringHandle(idInst, hszItem[i]);
DdeUninitialize(idInst);
```

(4) 在 CDDEdemoDlg::OnChangeEdit() 中实现编辑数据的动态更新。

```
UpdateData();
ServerData[0] = m_edit;
DdePostAdvise(idInst, hszTopic, hszItem[0]); //通知 DDE 更新该数据项
```

(5) 在 CDDEdemoDlg::OnTimer(UINT nIDEvent) 中完成计数和自动连接。

```
count++;
ServerData[1].Format("%d", count);
SetDlgItemText(IDC_STATIC1, ServerData[1]);
DdePostAdvise(idInst, hszTopic, hszItem[1]); //通知更新
if(!bConnect) //如果连接没有建立
{
    hConv = DdeConnect(idInst, hszApp, hszTopic, NULL);
    //连接服务端
    if(hConv) //如果建立成功
    {
        DWORD dwResult;
        bConnect = TRUE;
        //生成热连接型会话
        for(int i = 0; i < NITEM; i++)
            DdeClientTransaction(NULL, 0, hConv, hszItem[i],
                CF_TEXT, XTYP_ADVSTART, TIMEOUT_ASYNC, &dwResult);
    }
}
```

(6) 输入关键的 DDE 回调函数。

```
HDDATA CALLBACK DdeCallback( //回调函数
```



```
UINT wType, UINT wFmt, HCONV hConv, HSZ Topic, HSZ
Item, HDDEDATA hData, DWORD IData1, DWORD IData2)
{
    int i;
    char tmp[255];
    switch (wType)
    {
        case XTYP_ADVSTART:
        case XTYP_CONNECT: //连接请求
            return((HDDEDATA)TRUE); //允许
        case XTYP_ADVDATA: //有数据到来
            for(i = 0; i < NITEM; i++)
                if(Item == hszItem[i]) //确定数据
                    项
                    {
                        DdeGetData(hData, (PBYTE) tmp, 255, 0); //取得数据

                        switch(i) //更新界面
                        {
                            case 0:
                                SetDlgItemText(hWnd, IDC_STATIC2,
tmp);

                                break;
                            case 1:
                                SetDlgItemText(hWnd, IDC_STATIC3,
tmp);

                                break;
                        }
                    }
            return((HDDEDATA)DDE_FACK); //回执
        case XTYP_ADVREQ:
        case XTYP_REQUEST: //数据请求
            for(i = 0; i < NITEM; i++)
```

```
if(Item == hszItem[i])
    return(DdeCreateDataHandle(idInst, //返回相
应数据项内容
(PBYTE)(LPCTSTR)ServerData[i], ServerData[i].GetLength()
+ 1, 0, Item, wFmt, 0));
}
return(0);
}
```

本程序在 Visual C++ 6.0、Windows2000 Professional 环境下调试运行正常。

四、调试工具

VC 的自带工具 DDESPY 是个功能强大的 DDE 程序调试工具，在运行 DDE 程序前先运行它就可以监视、记录各种 DDE 的消息事件、对象内容、函数调用，非常方便。

五、小结

DDE 是个比较完备的进程间通信协议，本文只使用了其中一小部分常用功能，还有很多可以利用的功能，例如客户端还可以发命令和数据到服务器端；可以创建主题及数据项的列表，方便使用和管理；不止限于点对点的通信，可以实现多对多的数据共享；还可以封装成 C++ 类，利于重用（网上已有）。笔者相信通过 VC 能更充分利用 DDE，可以更灵活地实现许多更强的功能。

参考文献

1. MSDN
2. John E. Swanke 著，Visual C++ MFC 编程实例，前导工作室译，北京：机械工业出版社，2000

6.2 借助内嵌资源实现 VC 对 Flash 动画的播放

郎 锐

一、引言

Flash 动画由于可以很方便地把用户的想象通过动画显现出来，使原本只属于专业制作人员的动画制作变得异乎寻常的快捷、方便。由于 Flash 制作的动画在层次、内容、表现形式等诸多方面均比较出色，因此在网络上得到迅猛的发展，更有不少厂商用 Flash 在互联网上做起了广告和产品演示，效果丝毫不比视频的差，而体积则要小的多。Flash 不仅在网络上有广泛的应用，在普通的应用程序中也可以借助 Flash 实现一些 VC、Delphi 等编程语言所难以实现的特效，例如在一些演示版的程序中完全可以将

程序运行前的闪屏用 Flash 来制作。本文将通过对内嵌资源的动态释放来实现 VC 对 Flash 动画的播放，并给出部分实现代码。

二、内嵌资源的动态释放

Flash 动画在此是作为程序的一个模块，虽然也可以以文件的形式作为一个外部资源来使用，但为了避免因外部模块遗失而造成程序的非正常运行，可将由 Flash 5.0 预先制作好的 swf 格式的文件以资源的形式打包到应用程序中，而在程序运行时再将其从资源恢复到文件，使用完毕再通过程序将其从磁盘删除。

在导入资源时由于 swf 格式文件并非 VC 的标准资源,所以在导入时需要在“Resource type”栏指定资源类型“SWF”,特别需要注意的是在此必须要包含引号。加入到资源后可以通过资源视图看到导入的 SWF 资源是以二进制形式保存的,一旦加入就不能再通过资源视图对其进行编辑了。

在使用 SWF 资源前首先要将其动态从应用程序中释放到文件中才可对资源做进一步的使用。可先通过宏 MAKEINTRESOURCE()将资源标识号 IDR_SWF 转换成字符串 Name,再分别通过 FindResource()、LoadResource()函数查找、装载该资源到内存:

```
CString Type = "swf";
HRSRC res = FindResource (NULL, Name, Type);
HGLOBAL gl = LoadResource (NULL, res);
```

当资源加载到内存后,还要通过对资源内存的锁定来返回指向资源内存的地址的指针,并藉此实现资源从内存到磁盘的保存,至于存盘的操作则由文件函数 CreateFile()、和 WriteFile()来完成:

```
LPVOID lp = LockResource(gl); //返回指向资源内存的地址的指针
CString filename = "Temp. swf"; //保存的临时文件名
// CREATE_ALWAYS 为不管文件存不存在都产生新文件
fp = CreateFile(filename, GENERIC_WRITE, 0, NULL, CREATE_ALWAYS, 0, NULL);
DWORD a;
//sizeofResource 得到资源文件的大小
if (!WriteFile (fp, lp, SizeofResource (NULL, res), &a, NULL))
    return false;
CloseHandle (fp); //关闭句柄
FreeResource (gl); //释放内存
```

通过上述代码,可将 SWF 资源从应用程序中提取并释放到临时文件 Temp. swf 中,在此后只对此临时文件操作,与程序内嵌资源无关。

三、Flash 动画的播放

swf 格式的 Flash 动画通常主要应用在网页上,也就是说 IE 浏览器本身可以支持 Flash 动画的播放。这样就不必再单独编写用于播放 swf 文件的代码,从而大大减少编程的工作量。在 VC++ 6.0 中新增了一个从 CView 派生的、用于处理网页的视类 CHtmlView,由于该类是以 Internet Explorer 为后台支持,因此在创建工程时只需在最后一步指定视类从 CHtmlView 派生就可以使程序不编一行代码而具备 IE 浏览器的网页显示能力。

程序刚生成时默认的连接主页是为微软公司的主页,需要对此修改,使程序在执行时立即显示刚才提取出来的 Flash 临时文件 Temp. swf。显示默认主页的代码是在视类的初始化函数中进行的:

```
void CEmbedModuleView::OnInitialUpdate()
{
```

```
    CHtmlView::OnInitialUpdate();
    Navigate2(_T("http://www.microsoft.com"), NULL,
    NULL);
}
```

显然要将 Navigate2() 函数的第一个参数改成 Temp. swf 的存放路径。刚才在释放资源到文件时并没有指定绝对路径,因此释放出来的资源文件应当和应用程序处于同一目录。但是在此处如果不写明绝对路径是无法显示该临时文件的。获取该临时文件的绝对路径可用如下方法实现:先获取应用程序本身的绝对路径,然后去除应用程序全名(程序名和扩展名),此时得到的是应用程序和临时文件所处文件夹的路径,最后只需在此基础上加上临时文件的文件名 Temp. swf 即可得到临时文件的全路径。下面是实现的主要代码:

```
//获取应用程序的全路径
char exeFullPath[MAX_PATH];
GetModuleFileName(NULL, exeFullPath, MAX_PATH);
//将其格式化为字符串
m_TempFile.Format("%s", exeFullPath);
//去掉应用程序的全名(15 为应用程序文件全名的长度)
exeFullPath[m_TempFile.GetLength() - 15] = '\0';
//得到应用程序所在路径
m_TempFile.Format("%s", exeFullPath);
//得到临时文件的全路径
m_TempFile += "Temp. swf";
```

最后将得到的临时文件的全路径 m_TempFile 作为参数传递给 Navigate2() 即可在程序运行时把 Flash 动画作为主页而显示,如图 1 所示。

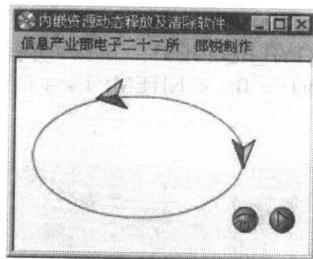


图 1

由于临时文件 Temp. swf 是在程序运行过程中从应用程序的资源中提取出来的,因此在程序退出之前需要将其删除。一般是在消息 WM_DESTROY 的响应函数里通过 DeleteFile() 函数来加以实现的。

四、小结

本文通过对 CHtmlView 和内嵌资源的动态释放实现了 Flash 动画在 VC 程序中的播放,并对资源的动态释放作了较为清晰的描述。通过类似的方法,可以将动态链接库、HTML 文件等程序模块作为资源嵌入其中,在使用时再动态释放到临时文件,这样可有效避免文件模块过多时的杂乱以及程序模块丢失导致程序非正常运行等情况的发生。本文所述程序在 Windows 98 下,由 Microsoft Visual C++ 6.0 编译通过。Flash 动画由 Macromedia Flash 5.0 制作,所需浏览器支持为 Internet Explorer 6.0。

6.3 VC 对多显示器系统的编程实现

张华敏 黎 莉

摘 要 本文主要介绍 VC++ 对多显示器系统编程的有关设置, 同时阐述多显示器系统在模拟设备设计研制中的重要意义。

关键词 多显示器系统, 分辨率, 模拟设备

Windows98 和 Windows2000 操作系统支持多显示器, 即能利用多块显卡和多个显示器, 有效扩展显示区域, 使各显示器拼成一个大显示器, 提高显示分辨率。

在操作系统中, 做好对多个显示器的设置后, 就可以享受多显示器带来的种种便利了, 例如可以把程序框从一个显示器拖到另一个显示器上, 可以同时在一个显示器上打字在其他显示器上看 VCD 等。但在用编程语言对多显示器进行编程控制时, 检测到的最大垂直和水平分辨率仅为主显示器的分辨率, 只能在主显示器上输出内容, 而在其他副显示器上得不到任何输出。经分析发现, 要使其其他显示器正确显示必须编程进行相应的设置, 下面以 VC 对双显示器编程为例, 介绍其实现的过程。

在双显示器系统正常工作的情况下, 在控制面板的显示属性中对两个显示器的位置随意安排, 共有 4 种情况: 副显示器分别在主显示器的上、下、左、右位置。

假设各显示器的分辨率都是 $800 \times 600\text{dpi}$, 对于副显示器分别在主显示器的左、右位置时的两种情况, 都可以当作 $1600 \times 600\text{dpi}$ 分辨率的一个大显示器来编程, 在上、下位置时可以看作是 $800 \times 1200\text{dpi}$ 分辨率的显示器。屏幕坐标 (0, 0) 应在组合显示器的左上角, 右下角应为组合分辨率的最大值, 但在用 VC 6.0 编程时发现, 对于以上各种情况的有效分辨率仅限于主显示器的 $800 \times 600\text{dpi}$, 副显示器的区域都是无效的, 只要程序的运行结果显示在主显示器以外, 无论怎么改变程序运行窗口, 即使完全覆盖两个显示器区域, 还是看不到任何显示。后来发现用 `SetWindowPlacement` 函数设置后即可正常使用了。该函数原型为:

`SetWindowPlacement(const WINDOWPLACEMENT * lpwndpl)` 其中最关键的是要对参数 `lpwndpl` 的 `rcNormalPosition` 分量进行设置, 该分量是一个 `CRect` 类型的变量, 为这个分量赋值后, 再调用函数 `SetWindowPlacement` 进行设置, 就可以把多显示器作为一个显示器用了。对于下图的四种情况 (假设显示器都是 $800 \times 600\text{dpi}$ 的分辨率), 当主显示器分别在副显示器的左、右、上、下位置时, 参数 `lpwndpl` 的 `rcNormalPosition` 分量的值分别为: (0, 0, 1600, 600)、(-800, 0, 800, 600)、(0, 0, 800, 1200)、(0, -600, 800,

600), 对于多于两个显示器的情况依次类推即可。

下面以主显示器在副显示器左边的情况为例, 说明双显示器满屏显示的实现过程 (首先, 必须在操作系统中确保双显示器系统已可正确显示, 并且设置主显示器为第一显示器, 副显示器为第二显示器, 主显示器在左, 副显示器在右, 设置两个显示器分辨率都是 $800 \times 600\text{dpi}$ 。)

(1) 在 VC++6.0 环境下, 建立一个名为 abc 的单文档 MFC AppWizard(exe) 程序, 其他选项保持默认; 向 `CMainFrame` 类中添加 `CRect` 类型变量 `m_FullScreenRect`。

(2) `CMainFrame` 类中添加成员函数 `OnFullScreen()`, 并定义如下:

```
void CMainFrame::OnFullScreen()
{
    CRect WindowRect;
    GetWindowRect(& WindowRect);
    CRect ClientRect;
    RepositionBars(0, 0xffff, AFX_IDW_PANE_FIRST, repos-
    Query, & ClientRect);
    ClientToScreen(& ClientRect);
    /* 以下是得到满屏参数 */
    int nFullWidth = GetSystemMetrics(SM_CXSCREEN);
    int nFullHeight = GetSystemMetrics(SM_CYSCREEN);
    m_FullScreenRect.left = WindowRect.left - ClientRect.left;
    m_FullScreenRect.top = WindowRect.top - ClientRect.top;
    m_FullScreenRect.right = (WindowRect.right - ClientRect.
    right + nFullWidth) * 2;
    m_FullScreenRect.bottom = WindowRect.bottom + nFull-
    Height;
    WINDOWPLACEMENT wndpl;
    wndpl.length = sizeof(WINDOWPLACEMENT);
    wndpl.flags = 0;
    wndpl.showCmd = SW_SHOWNORMAL;
    wndpl.rcNormalPosition = m_FullScreenRect;
    /* 按满屏区域设置窗口, 在这里实际值为 1600 x 600 */
    SetWindowPlacement(& wndpl);
}
```

(3) 按 `Ctrl+W` 键, 在对话框中选中 `CMainFrame` 类, 在 `Messages` 栏中找到 `WM_GETMINMAXINFO` 消息并双击, 产生 `OnGetMinMaxInfo` 函数, 定义如下:

```
void CMainFrame::OnGetMinMaxInfo(MINMAXINFO FAR *
```

```
lpMMI)
{
// TODO: Add your message handler code here and/or call
default
/* 以下代码保证显示器都处于全屏状态 */
lpMMI ->ptMaxSize.x = m_FullScreenRect.Width();
lpMMI ->ptMaxSize.y = m_FullScreenRect.Height();
lpMMI ->ptMaxPosition.x = m_FullScreenRect.Width();
lpMMI ->ptMaxPosition.y = m_FullScreenRect.Height();
lpMMI ->ptMaxTrackSize.x = m_FullScreenRect.Width();
lpMMI ->ptMaxTrackSize.y = m_FullScreenRect.Height();
CFrameWnd::OnGetMinMaxInfo(lpMMI);
}
```

(4) 找到 CmainFrame 类的 OnCreate 函数的定义, 在其最后一条语句“return 0;”的前面添加语句“OnFullScreen();”。

(5) 在类 CAbcView 的函数 OnDraw(CDC * pDC)中, 添加以下语句:

```
pDC ->TextOut(300, 300, “这是第一显示器, 本测试程序的副显示器应设在主显示器的右边”);
```

```
pDC ->TextOut(1100, 300, “这是第二显示器, 对双显示器的控制已经成功!”);
```

(6) 编译运行程序, 可以看到副显示器正常显示了。按 Alt + F4 键可关闭程序对于双显示器系统, 利用双头显卡也可控制两个显示器, 对于多于两个显示器的系统, 就只能采用多块显卡进行控制了。

通过编程控制, 多显示器系统在模拟设备的开发研制中有重要应用, 下面以一例来说明它的重要应用。

有某种武器, 价格昂贵, 需要两种独立的组件(两种组件间有信号交连)同时对它进行测试, 但随着通电次数的增多, 该武器的寿命将大大减少, 为完成教学, 节约成本, 决定用工控机开发该武器及测试设备的模拟系统, 以完成教学。其方框图如图 1 所示。

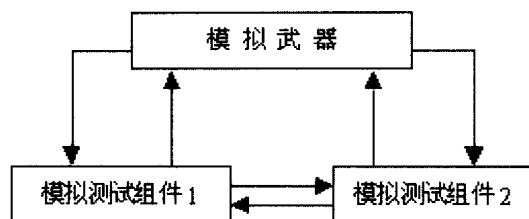


图 1 某武器模拟系统方框图

其中, 模拟武器框图从外形及信号上模拟真实武器, 模拟测试组件 1、模拟测试组件 2 分别模拟测试组件, 由微机系统组成, 做在相应的壳体里面。各测试组件(包括按钮、开关、指示灯、仪表等)的动作及显示结果由相应的触摸屏显示器完成, 为了增加模拟效果, 显示器采用上面介绍的满屏显示。在未使用多显示器技术时, 模拟测试组件 1、模拟测试组件 2 都必须为一套独立的微机系统, 都均包括主机、显示器、I/O 卡、信号调理电路, 而且编程时, 各组件都要有独立的程序, 此外还要考虑各微机的通信问题。随着测试组件的增多, 微机间的通信将变得极为复杂, 编程的难度也将显著增加。

当采用多显示器系统时, 可以只保留一套模拟测试组件的微机系统, 其他模拟测试组件只简化为一个显示器, 这样本系统就省去了一台微机及其对应的 I/O 卡、信号调理电路。测试组件越多, 节约的硬件成本就越多。而且由于是一个程序控制多个显示器, 没有微机间的通信问题, 这样测试组件间传输的信号, 就可用程序的变量直接代替, 用一套键盘、鼠标就可完成操作, 大大降低了程序的复杂性, 显著提高了程序的可靠性和稳定性。随着测试组件的增多, 编程只会在工作量上略有增加。

可见, 采用多显示器系统开发模拟设备, 能成倍地节约成本, 大大降低程序的复杂性, 提高程序的可靠性、可维护性及可操作性。

6.4 用 VC 6.0 实现汉字到区位码的批量转化

王文江 周国瑞

摘要 在考试报名填写机读卡的过程中, 查汉字区位码是经常遇到的事情。如果用手工去查, 不仅浪费时间, 而且容易出错。本文用 VC 6.0 实现汉字到区位码的批量转化, 做到快速准确的查询, 从而使填涂机读卡人员达到了事半功倍的效果。

关键词 MFC, 汉字区位码, ASCII 码, 批量转化