



LabWindows/CVI 虚拟仪器丛书

LabWindows/CVI



虚拟仪器高级应用

王建新 隋美丽 / 著



文本、文件类程序设计
菜单类程序设计



自定义控件设计
窗口操作程序设计



系统操作程序设计
高级控件使用技术



信息安全技术
场景与游戏程序设计



化学工业出版社

013038083

TP311.56
1140



LabWindows/CVI 虚拟仪器丛书

LabWindows/CVI



虚拟仪器高级应用

王建新 隋美丽 / 著



TP311.56
1140
P



北航

C1643977



化学工业出版社

· 北京 ·

080880810

图书在版编目 (CIP) 数据

LabWindows/CVI 虚拟仪器高级应用 / 王建新, 隋美丽
著. —北京: 化学工业出版社, 2013.3
ISBN 978-7-122-16256-4

I. ①L… II. ①王… ②隋… III. ①软件工具-程
序设计 IV. ①TP311.56

中国版本图书馆 CIP 数据核字 (2013) 第 003704 号

责任编辑: 宋 辉
责任校对: 徐贞珍

文字编辑: 孙 科
装帧设计: 韩 飞

出版发行: 化学工业出版社 (北京市东城区青年湖南街 13 号 邮政编码 100011)
印 刷: 北京永鑫印刷有限责任公司
装 订: 三河市万龙印装有限责任公司
787mm×1092mm 1/16 印张 25¼ 字数 706 千字 2013 年 5 月北京第 1 版第 1 次印刷

购书咨询: 010-64518888 (传真: 010-64519686) 售后服务: 010-64518899
网 址: <http://www.cip.com.cn>
凡购买本书, 如有缺损质量问题, 本社销售中心负责调换。

定 价: 88.00 元

版权所有 违者必究

➔ 前 言

LabWindows/CVI 是美国国家仪器公司推出的基于 ANSI C 的虚拟仪器开发平台, 包含了集成式开发环境、交互式编程方法、简单直观的图形用户界面设计、完善的兼容性、灵活的调试手段以及功能强大的函数库, 适用于测试测量、自动控制、数据通信、信号分析等领域。虚拟仪器技术就其实质而言, 是利用高性能的模块化硬件, 结合高效灵活的软件来完成各种测试、测量和自动化的应用。LabWindows/CVI 作为一种通用的开发工具能进行各种程序开发, 如仪器驱动开发、简单游戏开发、图形图像开发等。

本书主要以 LabWindows/CVI 2009 集成开发环境为基础展开论述, 包括了文本文件类程序设计技术、菜单类程序设计技术、自定义控件设计、窗口操作技术、系统操作技术、高级控件使用技术、信息安全技术以及其他程序设计技术等多个方面, 从各种角度、不同层次系统阐述了集成开发环境、程序设计思路、语言使用技巧, 并且提供了大量实例, 使读者更加深入地理解程序设计的一般思路与方法, 提高实际应用能力。本书的例程在 Windows XP 系统 LabWindows/CVI 2009 集成开发环境下全部调试通过, 也可以运行在 LabWindows/CVI 2010/2012 最新开发环境中。

本书由王建新、隋美丽撰写, 高献伟、李莉、靳济方、赵成、张磊、李秀滢、陈汉林、段晓毅、武玉华、董秀则、史国振、李雪梅、冼立勤、耿彦明、曾基石、周玉坤、宿淑春、王丽丰、周长春、张克军、姚栋、路而红、霍刚、王爱华、王永会、杨世凤、郭雅萌、张焱伟、童官军、赵继民、刘英等为本书编写提供了帮助, 在此一并感谢。

由于本书涉及面比较宽, 加上作者水平有限, 时间仓促, 书中不妥之处, 希望广大读者批评指正。

著者

化工版同类书推荐

ISBN	书名	定价
9787122103321	LabVIEW 虚拟仪器程序设计与应用	48
9787122113702	LabWindows/CVI 虚拟仪器测试技术及工程应用(附光盘)	85
9787122137463	ZigBee 无线传感器网络设计与实现	48
9787122116000	变频技术	29
9787122102515	变频器的使用与节能改造	28
9787122154224	变频器及软启动器应用计算	38
9787122152763	变频器实用技术问答	48
9787122103338	变频器实用手册	68
9787122093271	变频器应用技术丛书——电气传动与变频技术	30
9787122117175	传感器与测控电路	35
9787122117342	从新手到高手——过程控制系统基础与实践	48
9787122144287	电厂实用技术读本系列——热工仪表及自动控制系统	48
9787122152534	电子电工技术入门与提高丛书——图解西门子 S7-300/400PLC 技术快速入门与提高	48
9787122113931	多媒体与网络电视监控系统原理及应用	39
9787122105110	工业电视监控系统培训教程	35
9787122110343	集散控制系统应用技术	48
9787122115836	可编程片上系统 PSoC 设计指南(附光盘)	48
9787122134363	流量测量技术全书(下册)	108
9787122130075	三菱 FX 系列 PLC 完全精通教程(附光盘)	48
9787502526962	石油化工自动控制设计手册(三版)	138
9787122151124	图解电气控制入门	49
9787122156693	图解西门子 S7-200PLC 编程快速入门	46
9787122134158	图解易学变频技术(双色版)	48
9787122099624	无线传感器网络原理与实践	48
9787122108968	西门子 PLC S7-200/300/400/1200应用案例精讲(附光盘)	56
9787122138361	西门子 S7-200PLC 完全精通教程(附光盘)	49
9787122072313	西门子 S7-200系列 PLC 应用实例详解	36
9787122138545	西门子 S7-300/400PLC 快速入门手册	58
9787122083708	西门子 S7系列 PLC 电气控制精解	46
9787122156297	仪表维修工实用技能详解	29
9787122138705	英汉·汉英自动控制与仪表词汇	88
9787122024398	在线分析仪器手册	148
9787122126832	自动抄表系统原理与应用	29

以上图书由化学工业出版社 电气分社出版。如要以上图书的内容简介和详细目录,或者更多的专业图书信息,请登录 www.cip.com.cn。如要出版新著,请与编辑联系。

地址:北京市东城区青年湖南街13号(100011)

图书销售:010-64518800, 图书编辑:010-64519262

1 文本、文件类程序设计技术

1

1.1 英汉汉英词典.....	1
1.1.1 英汉汉英词典设计方法.....	1
1.1.2 英汉汉英词典程序设计.....	1
1.2 十六进制文本文件转换.....	8
1.2.1 十六进制文本文件转换设计方法.....	8
1.2.2 十六进制文本文件转换程序设计.....	8
1.3 英汉汉英词典合成版.....	16
1.4 霓虹灯效果文字.....	22
1.4.1 霓虹灯效果文字设计方法.....	22
1.4.2 霓虹灯效果文字程序设计.....	23
1.5 旋转文字.....	27
1.5.1 旋转文字设计方法.....	27
1.5.2 旋转文字程序设计.....	27
1.6 自动选中文本.....	29
1.6.1 自动选中文本设计方法.....	29
1.6.2 自动选中文本程序设计.....	29
1.7 文本自动感知.....	32
1.7.1 文本自动感知设计方法.....	32
1.7.2 文本自动感知程序设计.....	32
1.8 文件拖放.....	36
1.8.1 文件拖放设计方法.....	36
1.8.2 文件拖放程序设计.....	36
1.9 记事本.....	41
1.9.1 记事本设计方法.....	41
1.9.2 记事本程序设计.....	42
1.10 提取 MP3 信息.....	53
1.10.1 提取 MP3 信息设计方法.....	53
1.10.2 提取 MP3 信息程序设计.....	55

2.1 菜单感知	60
2.1.1 菜单感知设计方法	60
2.1.2 菜单感知程序设计	61
2.2 动态装载菜单项	65
2.2.1 动态装载菜单项设计方法	65
2.2.2 动态装载菜单项程序设计	66
2.3 系统菜单控制	69
2.3.1 系统菜单控制设计方法	69
2.3.2 系统菜单控制程序设计	70
2.4 系统菜单设计	72
2.4.1 系统菜单设计方法	72
2.4.2 系统菜单程序设计	73
2.5 系统“?”按钮	77
2.5.1 系统“?”按钮设计方法	77
2.5.2 系统“?”按钮程序设计	79

3.1 进度条移植	86
3.1.1 进度条移植设计方法	86
3.1.2 进度条移植程序设计	86
3.2 LabVIEW 风格 Button	92
3.2.1 LabVIEW 风格 Button 设计方法	92
3.2.2 LabVIEW 风格 Button 程序设计	97
3.3 LabVIEW 风格 Numeric	100
3.3.1 LabVIEW 风格 Numeric 设计方法	100
3.3.2 LabVIEW 风格 Numeric 程序设计	101
3.4 颜色拾取器	106
3.4.1 颜色拾取器设计方法	106
3.4.2 颜色拾取器程序设计	107
3.5 Slide 控件设计	112
3.5.1 Slide 控件设计方法	112
3.5.2 Slide 控件程序设计	113
3.6 Ring 控件设计	117
3.6.1 Ring 控件设计方法	117

3.6.2 Ring 控件程序设计	117
3.7 秒表制作	119
3.7.1 秒表设计方法	119
3.7.2 秒表程序设计	120
3.8 弹钢琴	123
3.8.1 弹钢琴设计方法	123
3.8.2 弹钢琴程序设计	124
3.9 用户界面设计	127
3.9.1 设计原则	127
3.9.2 可用性设计	129

4 窗口操作技术

132

4.1 欢迎界面	132
4.1.1 欢迎界面设计方法	132
4.1.2 欢迎界面程序设计	132
4.2 窗口动画效果	138
4.2.1 窗口动画效果设计方法	138
4.2.2 窗口动画效果程序设计	139
4.3 透明窗口	145
4.3.1 透明窗口设计方法	145
4.3.2 透明窗口程序设计	146
4.4 椭圆形窗口	153
4.4.1 椭圆形窗口设计方法	153
4.4.2 椭圆形窗口程序设计	154
4.5 多边形窗口	158
4.5.1 多边形窗口设计方法	158
4.5.2 多边形窗口程序设计	160
4.6 组合窗口	163
4.6.1 组合窗口设计方法	163
4.6.2 组合窗口程序设计	163
4.7 异形窗口播放器	167
4.7.1 异形窗口播放器设计方法	167
4.7.2 异形窗口播放器程序设计	169
4.8 字形窗口	177
4.8.1 字形窗口设计方法	177

4.8.2 字形窗口程序设计	178
4.9 桌面地震效果	183
4.9.1 桌面地震效果设计方法	183
4.9.2 桌面地震效果程序设计	184
4.10 截取屏幕	188
4.10.1 截取屏幕设计方法	188
4.10.2 截取屏幕程序设计	188

5 系统操作技术

202

5.1 程序只能打开一次	202
5.1.1 程序只能打开一次设计方法	202
5.1.2 程序只能打开一次程序设计	202
5.2 显示器操作	205
5.2.1 显示器操作设计方法	205
5.2.2 显示器操作程序设计	206
5.3 关机操作	213
5.3.1 关机操作设计方法	213
5.3.2 关机操作程序设计	214
5.4 Windows 操作	220
5.4.1 Windows 操作设计方法	220
5.4.2 Windows 操作程序设计	221
5.5 屏幕放大镜	227
5.5.1 屏幕放大镜设计方法	227
5.5.2 屏幕放大镜程序设计	227
5.6 提取图标	230
5.6.1 提取图标设计方法	230
5.6.2 提取图标程序设计	230
5.7 应用程序子窗口	234
5.7.1 应用程序子窗口设计方法	234
5.7.2 应用程序子窗口程序设计	236
5.8 文件关联	240
5.8.1 文件关联设计方法	240
5.8.2 文件关联程序设计	240
5.9 精确定时	244
5.9.1 Windows 常规定时器设计	244

5.9.2	多媒体定时器设计	248
5.9.3	高精度定时器设计	252
5.9.4	时间戳定时器（嵌入汇编语言）设计	256
5.10	设置墙纸	260
5.10.1	设置墙纸设计方法	260
5.10.2	设置墙纸程序设计	261

6 高级控件使用技术

202

6.1	语音朗读	265
6.1.1	语音朗读设计方法	265
6.1.2	语音朗读程序设计	268
6.2	文字识别	271
6.2.1	文字识别设计方法	271
6.2.2	文字识别程序设计	272
6.3	报表制作	278
6.3.1	报表制作设计方法	278
6.3.2	报表制作程序设计	279
6.4	PDF 阅读器	290
6.4.1	PDF 阅读器设计方法	290
6.4.2	PDF 阅读器程序设计	290
6.5	网络搜索	295
6.5.1	网络搜索设计方法	295
6.5.2	网络搜索程序设计	296
6.6	多媒体播放器	299
6.6.1	多媒体播放器设计方法	299
6.6.2	多媒体播放器程序设计	301

7 信息安全技术

306

7.1	图像信息隐藏	306
7.1.1	图像信息隐藏设计方法	306
7.1.2	图像信息隐藏程序设计	307
7.2	病毒制作	314
7.2.1	病毒制作设计方法	314
7.2.2	病毒制作程序设计	314
7.3	窗口属性	318
7.3.1	窗口属性设计方法	318

7.3.2 窗口属性程序设计	318
7.4 灰色按钮克星	325
7.4.1 灰色按钮克星设计方法	325
7.4.2 灰色按钮克星程序设计	326
7.5 控制计算器	333
7.5.1 控制计算器设计方法	333
7.5.2 控制计算器程序设计	333
7.6 获取密码	338
7.6.1 获取密码设计方法	338
7.6.2 获取密码程序设计	339
7.7 数据加密	342
7.7.1 数据加密设计方法	342
7.7.2 数据加密程序设计	343
8 其他程序设计技术	346
8.1 RPG 场景设计	346
8.1.1 RPG 场景设计设计方法	346
8.1.2 RPG 场景设计程序设计	347
8.2 钢琴演奏	354
8.2.1 钢琴演奏设计方法	354
8.2.2 钢琴演奏程序设计	355
8.3 贪吃蛇	362
8.3.1 贪吃蛇设计方法	362
8.3.2 贪吃蛇程序设计	362
8.4 扫雷	368
8.4.1 扫雷设计方法	368
8.4.2 扫雷程序设计	369
附录	375
A 数独游戏	375
B 俄罗斯方块	381
参考文献	391

1

文本、文件类程序设计技术

1.1 英汉汉英词典

1.1.1 英汉汉英词典设计方法

在 LabWindows/CVI 程序设计中，经常要参考和查询一些函数，阅读英文说明文档，英汉词典是不可或缺的工具。常用的电子词典主要是有道、金山等，一般的词典安装程序至少也在几兆大小，安装之后可能还会带有插件，能不能设计一个更简洁的词典程序，只要满足日常需要就可以呢？对于简洁的词典程序，其词库最好为文本类型，不使用任何数据库产品，减少冗余。在本例程中，利用字符串查找与匹配算法，制作了一个简单的英汉汉英电子词典，实现单词的查找释义、汉字的英文对照。

1.1.2 英汉汉英词典程序设计

(1) 面板设计

编写一个英汉汉英词典查询程序，在文本框内输入英文单词或中文，程序会打开当前目录下的“英汉汉英词典.txt”词典文件，查找与输入相匹配项，并将匹配度最高的结果及其后续的相关结果显示出来。面板设计如图 1-1 所示，面板中主要控件属性设置如表 1-1 所示。

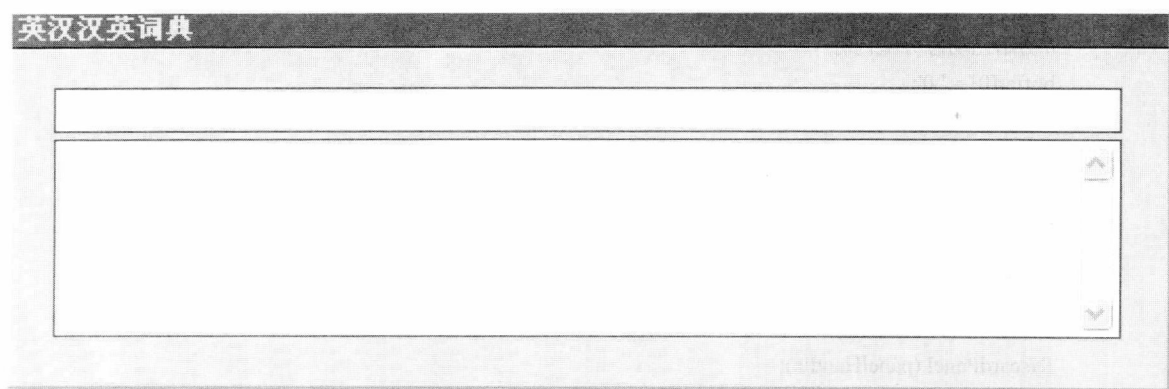


图 1-1 英汉汉英词典面板设计

表 1-1 控件属性设置表

常 量 名	控 件 类 型	控件的主要属性
PANEL	Panel	标题：英汉汉英词典 回调函数：PanelCB
STRING_FIND	String	回调函数：Find
TEXTBOX	Text Box	(显示查询结果)

(2) 程序源代码

```

//头文件声明
#include <ansi_c.h>
#include <formatio.h>
#include <cvirte.h>
#include <userint.h>
#include "英汉汉英词典.h"
//全局静态变量
static int BytesRead;
static int FileHandle;
static char *buffer;
static int panelHandle;
//主函数
int main (int argc, char *argv[])
{
    long FileSize;
    if (InitCVIRTE (0, argv, 0) == 0)
        return -1; /* out of memory */
    if ((panelHandle = LoadPanel (0, "英汉汉英词典.uir", PANEL)) < 0)
        return -1;
    //打开字典文件
    FileHandle = OpenFile ("英汉汉英词典.txt", VAL_READ_ONLY, VAL_APPEND, VAL_ASCII);
    //获得字典文件信息
    GetFileInfo ("英汉汉英词典.txt", &FileSize);
    //动态分配内存
    buffer = malloc (FileSize * sizeof(char) + 1);
    //初始化动态分配内存
    buffer[0] = '\0';
    //读取字典文件内容
    BytesRead = ReadFile (FileHandle, buffer, FileSize * sizeof(char));
    DisplayPanel (panelHandle);
    RunUserInterface ();
    //释放字典文件占用的内存
    free (buffer);
    CloseFile (FileHandle);
    DiscardPanel (panelHandle);
    return 0;
}
//面板回调函数

```

```

int CVICALLBACK PanelCB (int panel, int event, void *callbackData,
    int eventData1, int eventData2)
{
    switch (event)
    {
        case EVENT_CLOSE:
            QuitUserInterface (0);
            break;
    }
    return 0;
}

//查找文本框回调函数
int CVICALLBACK Find (int panel, int control, int event,
    void *callbackData, int eventData1, int eventData2)
{
    int Index;
    int len;
    char *str;
    int i;
    int j;
    static int flag = 0;
    int num;
    char *temp;
    switch (event)
    {
        case EVENT_VAL_CHANGED:
            //获得输入文本长度
            GetCtrlAttribute (panelHandle, PANEL_STRING_FIND, ATTR_STRING_TEXT_LENGTH,
&len);

            //将获得查找字符串
            str = malloc (len * sizeof(char) + 1);
            str[0] = '\0';
            GetCtrlVal (panelHandle, PANEL_STRING_FIND, str);
            flag = 0;
            //字符串匹配
            for (i = 0; i < BytesRead; i++)
            {
                //标志清零
                flag = 0;
                //判断字符串是否匹配
                for (j = 0; j < len; j++)
                {
                    //字符串匹配并且遇到换行，标志自增，否则查询下一个，标志清零
                    if ((str[j] == buffer[i+j]) && ((i == 0) || (buffer[i-1<=0?0:i-1] == 0x0a)))
                    {

```



```

        flag++;
    }
    else
    {
        flag = 0;
        continue;
    }
}
//如果字符串匹配，显示四组左右连续查询结果
if (flag == len)
{
    //清空以前查询内容
    ResetTextBox (panelHandle, PANEL_TEXTBOX, "");
    num = 0;
    j = 0;
    //记录四组数据的长度，每组以换行符结束
    while (j <= 4)
    {
        if (*(buffer + num) == 0x0a)
        {
            j++;
        }
        num++;
    }
    temp = malloc (num * sizeof(char) + 1);
    temp[0] = '\0';
    memcpy (temp, buffer + i, num);
    temp[num<0?0:num] = '\0';
    //查找字符串中是否存在换行符
    Index = FindPattern (temp, 0, -1, "\n", 0, 1);
    //如果不存在匹配字符串，清空 temp，否则，将换行符改为结束符
    if (Index == -1)
    {
        temp[0] = '\0';
    }
    else
    {
        temp[Index] = '\0';
    }
    //显示查询结果
    SetCtrlVal (panelHandle, PANEL_TEXTBOX, temp);
    free (temp);
    break;
}
}
//清空查询内容

```

```

        if (len == 0)
        {
            ResetTextBox (panelHandle, PANEL_TEXTBOX, "");
        }
        //释放资源
        free (str);
        break;
    }
    return 0;
}

```

(3) 程序注释

① **OpenFile 函数** 打开文件，设置文件的读写操作，并返回文件句柄以备后续操作。注意，在使用完成之后，应当用 **CloseFile** 函数释放资源。由于 Windows API 中也包含有同名函数，需要把 windows.h 头文件放在 formatio.h 头文件之前，以避免产生编译错误。函数原型为：

int OpenFile (char File_Name[], int ReadWrite_Mode, int Action, int File_Type);

File_Name[]：指定打开的文件名。

ReadWrite：指定文件读写方式，默认为只读方式。如果为写或读写方式，当指定的文件不存在时，函数会先创建该文件。可以用 **GetFileInfo** 函数判断文件是否存在。读写方式参数说明如表 1-2 所示。

表 1-2 读写方式参数说明表

常 量 名	常量值	说 明
VAL_READ_WRITE	0	以读写方式打开文件
VAL_READ_ONLY	1	以只读方式打开文件
VAL_WRITE_ONLY	2	以只写方式打开文件

Action：指定是否要删除以前的文件内容以及写操作时文件指针的位置，只有当写或读写方式时有效。操作方式说明如表 1-3 所示。

表 1-3 操作方式说明表

常 量 名	常量值	说 明
VAL_TRUNCATE	0	文件指针定位在开始位置，删除以前文件内容
VAL_APPEND	1	将新文件内容添加到原有文件的末尾
VAL_OPEN_AS_IS	2	文件指针定位在开始位置，不影响以前文件内容

File_Type：指定文件类型为 ASCII 码或二进制类型。如果以 ASCII 码类型打开文件，当为读方式时，回车/换行（CR/LF）被转换为换行，当为写方式时，换行被转换为回车/换行。文件类型如表 1-4 所示。

表 1-4 文件类型表

常 量 名	常量值	说 明
VAL_BINARY	0	以二进制模式打开文件
VAL_ASCII	1	以 ASCII 码模式打开文件

返回值：文件句柄。如果返回值为-1，表示有错误产生，函数调用失败，可能是文件无法打开或参数传递错误。

② **ReadFile 函数** 将数据从文件或标准输入端口读到缓冲区。读指针位于文件指针开始位

置,当读完一个字节后,文件指针指向下一个未读字节,读到 NULL 结束符时,文件读操作结束。

```
int ReadFile (int fileHandle, char buffer[], int count);
```

fileHandle: 文件句柄。

buffer[]: 读取文件缓冲区。

Count: 最大读取字节数,不应大于缓冲区大小。

返回值: 实际读取字节数。-1 表示有错误产生,0 表示没有读到任何字节。

③ CloseFile 函数 关闭文件。函数原型为:

```
int CloseFile (int File_Handle);
```

File_Handle: 要关闭的文件句柄。

④ FindPattern 函数 在指定的字符串缓冲区中查找匹配字符串,并返回匹配字符串在字符串缓冲区中出现的位置信息。函数原型为:

```
int FindPattern (char *buffer, int startingIndex, int numberOfBytes, char *pattern, int caseSensitive, int startFromRight);
```

*buffer: 字符串缓冲区。

startingIndex: 开始查找的位置索引,从 0 开始计数,类似于字符串数组的下标。

numberOfBytes: 如果输入-1,函数从 startingIndex 位置索引开始查找,直到字符串缓冲区结束为止。如果输入值超过 UINT_MAX (0xffffffff),函数返回一个错误号。

*pattern: 指定查找的匹配字符串。

caseSensitive: 是否大小写敏感。如果为 0,则大小写不敏感,非 0 值表示大小写敏感。

startFromRight: 指定是从左向右匹配或是从右向左匹配。如果为 0,则从最左边开始查找匹配字符串,如果为非 0 值,则从最右边开始查找匹配字符串。

返回值: 返回在缓冲区中找到的第一个匹配字符串索引。如果没有匹配字符串,则返回-1。

举例说明:

```
index = FindPattern ("1ab2ab3ab4", 3, -1, "AB", 0, 0);
```

函数返回值 index 为 4,不返回 1 的原因是,虽然从最左边开始查找,但计数初值从 3 开始,且不区分大小写。

```
index = FindPattern ("1ab2ab3ab4", 3, -1, "AB", 0, 1);
```

函数返回值 index 为 7,从最右边开始查找,且不区分大小写。

⑤ memcpy 函数 从源缓冲区复制指定数目的字符串到目标缓冲区。如果源缓冲区与目标缓冲区存在重叠,需要使用 memmove 函数。函数原型为:

```
void *memcpy (void *targetBuffer, const void *sourceBuffer, size_t number_ofBytes);
```

*targetBuffer: 目标缓冲区。

*sourceBuffer: 源缓冲区。

number_ofBytes: 复制字节数目。

返回值: 目标缓冲区指针。

⑥ EVENT_VAL_CHANGED 事件常量 当用户改变控件中的值时将持续触发该事件。在本例程中,当输入到 String 控件中的字母或中文字符发生改变时,其查询结果也会随之发生变化。若采用 EVENT_COMMIT 事件,只有当输入完成并按下回车键时,String 控件的回调事件才能获得响应,较 EVENT_VAL_CHANGED 事件,反应速度慢,有显示滞后的感觉,与常用的英汉词典程序比较,用户体验差距较大。

⑦ 可能会出现的编译错误问题 当在查询文本框内输入中文之后,点击标题栏的退出按钮,可能会弹出致命运行时错误对话框,如图 1-2 所示。