```c
// 文件: winGUI.c
// 内容: 人机界面操作实现版本2——基于 MS Windows 的图形界面
// 作者: WXQ#2018

/*  本文代码基于 Windows API 实现，也定义了
   "Win32 Application" 型应用的主函数 WinMain()
 */

#ifndef _PARSER_WITH_SEMANTICS
#error "你必须定义宏 _PARSER_WITH_SEMANTICS."
#endif

#include <stdio.h> // fprintf(), stdout
#include <stdlib.h>

#include <windows.h>
#include <wingdi.h>
//#include <winuser.h>


#include "ui.h"
#include "errlog.h"


HDC hDC;                // 窗口句柄，全局变量
char SrcFilePath[2018+1];       // 用于存放源程序文件路径
static char* Name = "XDDL Interpreter(V-C, by:WXQ)";    // 窗口名

void Action()
{
    // 语法分析器接口
    extern void Parser(const char * file_name);  // in parser.c

    InitError(); // in errlog.c
    Parser(SrcFilePath);
    CloseError();
}

// ------------------------- 检查被分析的源程序文件是否合法函数
int CheckSrcFile(LPSTR lpszCmdParam)
{
    FILE * file = NULL;

    if(strlen(lpszCmdParam) == 0)
    {
        ShowMessage(1, "未指定源程序文件 !");
        return 0;
    }
    if((file = fopen(lpszCmdParam, "r")) == NULL)
    {
        ShowMessage(1, "打开源程序文件失败 !");
        MessageBox(NULL,lpszCmdParam, "文件名", MB_OK);
        return 0;
    }
    else fclose(file);
    return 1;
}


// ------------------------- 窗口处理函数
LRESULT CALLBACK WndProc(HWND hWnd,UINT Message,WPARAM wParam,LPARAM lParam)
{    switch(Message)
```

```
63      {    case WM_DESTROY:
64              ReleaseDC(hWnd,hDC);
65              PostQuitMessage(0);
66              return 0;
67          case WM_PAINT:
68          {
69              PAINTSTRUCT pt;
70              BeginPaint(hWnd,&pt);
71              // ----------------------------------------
72              //        调用绘图语言解释器，调用方法视源程序而定
73              Action();
74              // ----------------------------------------
75              EndPaint(hWnd,&pt);
76          }
77          default:
78              return DefWindowProc(hWnd,Message,wParam,lParam);
79      }
80  }
81
82  // ------------------------ 初始化窗口函数
83  int PrepareWindow(HINSTANCE hInst,
84                    HINSTANCE hPrevInstance,
85                    int nCmdShow)
86  {
87      HWND        hWnd;
88      WNDCLASS    W;
89
90      int x, y;           // 窗口左上角的位置
91      int width, height;  // 窗口的宽度、高度
92
93      width = 740;  height = 490;
94      x = (GetSystemMetrics(SM_CXSCREEN) - width)/2; // 窗口位于屏幕中间吧
95      y = (GetSystemMetrics(SM_CYSCREEN) - height)/2;
96
97      memset(&W,0,sizeof(WNDCLASS));
98      W.style = CS_HREDRAW | CS_VREDRAW;
99      W.lpfnWndProc = WndProc;
100     W.hInstance = hInst;
101     W.hCursor = LoadCursor(NULL,IDC_ARROW);
102     W.hbrBackground = (HBRUSH) (COLOR_WINDOW + 1);
103     W.lpszClassName =Name;
104     RegisterClass(&W);
105
106     hWnd = CreateWindow(Name, Name, WS_OVERLAPPEDWINDOW,
107                         x, y, width, height,
108                         NULL,NULL,hInst,NULL);
109     if(hWnd == NULL) return 0;
110
111     hDC = GetDC(hWnd);
112     ShowWindow(hWnd,nCmdShow);
113     UpdateWindow(hWnd);
114     SetCursor(LoadCursor(hInst,IDC_ARROW));
115
116     return 1;
117 }
118
119
120 // ------------------------ window GUI 程序的主函数，不能是 main()
121 int APIENTRY WinMain(HINSTANCE hInstance,
122                      HINSTANCE hPrevInstance,
123                      LPSTR     lpCmdLine,
124                      int       nCmdShow)
```

```
{
    // 保存源文件路径
    strcpy(SrcFilePath, lpCmdLine);

    // 初始化窗口.
    if ( PrepareWindow(hInstance,hPrevInstance,nCmdShow) == 0 )
    {
        ShowMessage(1, "窗口初始化失败 !");
        return 1;
    }

    // 检查要分析的源程序文件
    if ( CheckSrcFile(lpCmdLine) == 0 ) return 1;

    // ------------------------------------------
    //        调用绘图语言解释器，调用方法视源程序而定
    Action();
    // ------------------------------------------

    // 进入window消息循环
    MSG Msg;
    while(GetMessage(&Msg,NULL,0,0))
    {
        TranslateMessage(&Msg);
        DispatchMessage(&Msg);
    }
    return Msg.wParam;
}


// 下面是两个对外提供的公共操作实现, See ui.h


void ShowMessage(int flag,  // 常规类信息=0,  错误类信息=1
            const char* msg)
{
    if(flag == 0)
        MessageBox(NULL, msg, "提示", MB_OK | MB_ICONINFORMATION);
    else
        MessageBox(NULL, msg, "ERROR", MB_OK | MB_ICONERROR);
}

void DrawPixel( unsigned long x, unsigned long y,   // 点在窗口中的坐标
                unsigned int color_val    // 点的颜色, 0 为默认色，保留参数
              )
{
    COLORREF color = RGB(255, 0, 0);
    // 一个物理像素实在太小了，因此用四个物理像素表示一个逻辑像素
    SetPixel(hDC, x,   y,   color);
    SetPixel(hDC, x+1, y,   color);
    SetPixel(hDC, x,   y+1, color);
    SetPixel(hDC, x+1, y+1, color);
}

```