1模态对话框创建

CModal dlg;//创建对话框实例

Int result = dlg.DoModal()//生成模态对话框

2非模态对话框

通过CDialog类的Create和CWnd类的ShowWindow()成员函数

3获得空间指针

CEdit\* pEdit = (CEdit \*)GetDlgItem(IDC\_EDIT1)

4获得位置和像素点像素点

方法一

使用CWindowDC获得屏幕任意一点point的像素值：

CWindowDC dc(GetDesktopWindow());

COLORREF color = dc.GetPixel(point); //获得点point的颜色值

int Red=GetRValue(color); //获得红色值

int Green=GetGValue(color); //获得绿色值

int Blue=GetBValue(color); //获得蓝色值

方法二

CPoint point;

**COLORREF** RGBcolor;

GetCursorPos(&point);

**HDC** hDC = ::GetDC(NULL);

RGBcolor=::GetPixel(hDC,point.x,point.y);

CString TmpRgb;

TmpRgb.Format("%d",GetRValue(RGBcolor));

TmpRgb.Format("%d",GetGValue(RGBcolor));

TmpRgb.Format("%d",GetBValue(RGBcolor));

如果需要取得鼠标的位置，有3种方法，一种是使用定时器，不停的GetCursorPos(&point)获取鼠标位置.另外一种是使用钩子，获取鼠标的按下，放开，移动消息。还有一种是在鼠标按下时使用SetCaputrue，然后在MouseMove中得到点，在鼠标放开时使用ReleaseCaputure。

---------------------

原文：<https://blog.csdn.net/zengweicdut/article/details/8793586>

5控件变量的分类

控件变量分为值变量和控件类的变量；

①值变量：与控件关联的数据变量，可以为任何MFC基本数据类型，在控件和变量之间有两种数据的传递方式：控件传递给变量（表征为向控件输入数值改变与之关联的变量的值），和变量传递给控件（表征为刷新后控件中的数值发生变化）

②控件类的变量：控件类的对象，方便对控件本身进行操作，比如设置焦点，显示隐藏控件本身等,代替GetDlgItem()函数；

二.控件与其相关联的变量之间的值传递

1.传递方式

①控件传递给变量：

通过修改控件框中的数值，来改变与控件关联变量的数值；

②变量传递给控件：

通过修改变量的值来改变控件框的显示；

2.数据交换和检验

现在我们来思考当控件框值和与之关联的变量有一方发生改变时，是谁监测检验并执行数据的交换呢？在MFC中有这样两个函数

DoDataExchange()； //执行数据交换

UpdateData()； //监测检验

DoDataExchange()来执行数据的交换，这个函数是MFC自动生成的函数，只要有关联变量，MFC就会在次函数中自动添加关联语句；

void CAdditionDlg::DoDataExchange(CDataExchange\* pDX)

{

// 处理MFC默认的数据交换

CDialogEx::DoDataExchange(pDX);

// 处理控件IDC\_SUMMAND\_EDIT和变量m\_editSummand之间的数据交换

DDX\_Text(pDX, IDC\_SUMMAND\_EDIT, m\_editSummand);

// 处理控件IDC\_ADDEND\_EDIT和变量m\_editAddend之间的数据交换

DDX\_Text(pDX, IDC\_ADDEND\_EDIT, m\_editAddend);

// 处理控件IDC\_SUM\_EDIT和变量m\_editSum之间的数据交换

DDX\_Text(pDX, IDC\_SUM\_EDIT, m\_editSum);

}

但是这个函数并不会自动响应，需要人为的调用，就是说何时进行数据的交换，即使用UpdateData()函数进行调用DoDataExchange()函数的执行，

UpdateData()函数的原型为：BOOL UpdateData(BOOL bSaveAndValidate = TRUE);

参数：bSaveAndValidate用于指示数据传输的方向，TRUE表示从控件传给变量，FALSE表示从变量传给控件。默认值是TRUE，即从控件传给变量。

例如：

void CAdditionDlg::OnBnClickedAddButton()

{

// 将各控件中的数据保存到相应的变量

UpdateData(TRUE);

// 将被加数和加数的加和赋值给m\_editSum

m\_editSum = m\_editSummand + m\_editAddend;

// 根据各变量的值更新相应的控件。和的编辑框会显示m\_editSum的值

UpdateData(FALSE);

}

三.关联变量生成的位置

将会在三处生成代码

1. 头文件 访问所选位置(这里是public) int iNumber;

2. 类初始化此变量

3.DoDataExchange 出生成相关代码

DDX\_Text(pDX, IDC\_EDIT4, iNumber);

DDV\_MinMaxInt(pDX, iNumber, 0, 50);

---------------------

作者：凉凉猫

来源：CSDN

原文：https://blog.csdn.net/pipinuan/article/details/53810894

版权声明：本文为博主原创文章，转载请附上博文链接！

6数字转换为字符串并输出

d.Format(\_T("%d"),point1.x);

m\_Cdc->TextOutW(5 + 25, 6, d);

方法一：

int i = atoi((LPCTSTR)CString);

方法二：

CString str="100";

int n=atoi(str.GetBuffer(str.GetLength()));

7通过ID获得控件

GetDlgItem()

8获得CDC和HDC

CDC\* m\_Cdc = GetDC();

CDC\* pORIDC = GetDlgItem(IDC\_PIC) -> GetDC() ;

HDC hdc = pORIDC->GetSafeHdc();

9 Mfc消息格式

头文件内部

DECLARE\_MESSAGE\_MAP()

插入消息处理函数

CPP文件

开头部分的

BEGIN\_MESSAGE\_MAP(CGetRGBTestApp, CWinApp)

ON\_COMMAND(ID\_HELP, &CWinApp::OnHelp)

插入消息

END\_MESSAGE\_MAP()

解决fopen、fscanf 在VS中要求替换为fopen\_s、fscanf\_s的最全解决办法 -------------- ps：在使用MFC中遇到上述问题，可以通过方法三解决方法一：在程序最前面加#define \_CRT\_SECURE\_NO\_DEPRECATE；

方法二：在程序最前面加#define \_CRT\_SECURE\_NO\_WARNINGS；

方法三：在程序最前面加#pragma warning(disable:4996)；

方法四：把scanf、scanf改为scanf\_s、fopen\_s，具体方法请百度；

方法五：无需在程序最前面加那行代码，只需在新建项目时取消勾选“SDL检查”即可；

方法六：若项目已建立好，在项目属性里关闭SDL也行；

方法七：在工程项目设置一下就行；将报错那个宏定义放到 项目属性 -- C/C++-- 预处理器 -- 预处理器定义；

方法八：在 项目属性 -- c/c++ -- 命令行 添加：/D \_CRT\_SECURE\_NO\_WARNINGS 就行了。

---------------------

作者：C赵江松

来源：CSDN

原文：https://blog.csdn.net/qq\_38149046/article/details/80359743

版权声明：本文为博主原创文章，转载请附上博文链接！

文件打开

CFileDialog dlgFile(TRUE, NULL, m\_recordingFileName.GetBuffer(), OFN\_HIDEREADONLY, \_T("PCM Files (\*.pcm)|\*.pcm|All Files (\*.\*)|\*.\*||"), NULL);

if (dlgFile.DoModal())

{

m\_recordingFileName = dlgFile.GetPathName();

}

加载图标

(HICON)::LoadImage(AfxGetInstanceHandle(), strIco, IMAGE\_ICON, nW, nW, LR\_DEFAULTCOLOR | LR\_CREATEDIBSECTION | LR\_LOADFROMFILE);

1关于ModifyStyle ModifyStyleEx修改自定义控件的问题

继承于MFC控件，比如CStatic CEdit等等，在自定义代码中许多人反映无法更改控件的外观属相，这是因为大部分人没有通知主窗体自定义控件的外观更改了，可以使用下列代码进行更改：

ModifyStyle(WS\_BORDER,0,SWP\_DRAWFRAME); ModifyStyleEx(WS\_EX\_STATICEDGE|WS\_EX\_WINDOWEDGE|WS\_EX\_DLGMODALFRAME,

WS\_EX\_CLIENTEDGE,SWP\_DRAWFRAME);

注意必须添加SWP\_DRAWFRAME

矩形框设置文字函数

int DrawTextEx(HDC, hdc, LPTSTR lpchText, int cchText, LPRECT lprc, UINT dwDTFormat, LPDRAWTEXTPARAMS lpDTParams)；

参数：

hdc：设备环境句柄。

lpchText：指向将被绘制的字符串的指针，如果参数cchText设置为C1，则此字符串必须以\0结束。如果dwDTFormat包含DT\_MODIFYSTRING，此函数可以给字符串增加4个字符，存放字符串的缓冲区必须能足以容纳增加字符。

CchText：为lpchText指定的字符串指定字符数。如果字符串是以\0结束的，则此参数可设为C1。

lprc：指向结构RECT的指针，其中含有正文将格式化于其中的矩形的逻辑坐标信息。

DwDTFormat：指定格式选项，此参数取值参见DrawText\uformat。

dwDTParams：指向DRAWTEXTPARAMS结构的指针，该结构指定附加的格式选项，此参数可以为NULL。

返回值：如果函数调用成功，返回值是正文的高度；如果函数调用失败，返回值是0。

ON\_MESSAGE响应的是自定义消息,有关自定义消息的处理请看如下步骤:

1.定义消息:

在CCDlg类的头文件中加入如下代码：

#define WM\_CUSTOMIZE WM\_USER+1

头文件中加入Customize的声明：

afx\_msg LRESULT Customize(WPARAM wParam, LPARAM lParam);

2.cpp文件中加入消息的注册

ON\_MESSAGE(WM\_CUSTOMIZE, Customize)

3．加入消息响应函数实现

在CCDlg类的实现文件中加入消息响应函数的实现部分，代码如下：

void CCDlg::Customize(WPARAM wParam, LPARAM lParam)

{

CString strTittle;

strTittle.Format("自定义消息参数：x=%d,y=%d",wParam,lParam);

SetWindowText(strTittle);

}

4.显式地发送消息:

在主框架的“帮助”主菜单中加入“自定义消息”子菜单，其ID设为ID\_CUSTOMIZE，通过类向导响应该命令，函数名默认。其函数体如下：

void CMainFrame::OnCustomize()

{

POINT pos;

//将消息参数定义为屏幕坐标值

GetCursorPos(&pos);

pdlg->SendMessage(WM\_CUSTOMIZE,(UINT)pos.x,pos.y);

}

显示窗口

m\_pDlgGetWndMsg = new CGetWndMessageDlg(this);

m\_pDlgGetWndMsg->Create(IDD\_XXXXXXXX\_DIALOG);

m\_pDlgGetWndMsg->MiShow();

ShowWindow(m\_hWnd(窗口句柄), SW\_SHOWNOACTIVATE);

 设置窗口位置

CRect reDesk;

::SystemParametersInfo(SPI\_GETWORKAREA, 0, &reDesk, SPIF\_SENDCHANGE); //获取桌面的大小

int nX = (reDesk.right - m\_nWidth) / 2;

int nY = (reDesk.bottom - m\_nHeight) / 2;

::SetWindowPos(m\_hWnd, HWND\_TOPMOST, nX, nY, m\_nWidth, m\_nHeight, g\_nWndStyle);

::SetWindowTextW(m\_hWnd, \_T("获取窗口信息-许成亮"));

通过点获得窗口句柄

CPoint pt;

::GetCursorPos(&pt);

s\_hWndCopy = ::WindowFromPoint(pt);