# 学习大纲

|  |
| --- |
|  |

## 我们上一节学习的方法传递的数据非常有限仅仅是整数，这一节学习的这个消息，能够传递很多数据叫做WM\_COPYDATA,它用一个结构体作为参数。

## COPYDATASTRUCT结构体

|  |  |  |
| --- | --- | --- |
| 包含要通过 [WM\_COPYDATA](https://learn.microsoft.com/zh-cn/windows/desktop/dataxchg/wm-copydata) 消息传递给另一个应用程序的数据。 语法 C++复制  typedef struct tagCOPYDATASTRUCT {  ULONG\_PTR dwData;  DWORD cbData;  PVOID lpData;  } COPYDATASTRUCT, \*PCOPYDATASTRUCT; 成员 dwData  类型： **ULONG\_PTR**  要传递给接收应用程序的数据类型。 接收应用程序定义有效类型。  cbData  类型：DWORD  **lpData** 成员指向的数据的大小（以字节为单位）。  lpData  类型： **PVOID**  要传递给接收应用程序的数据。 此成员可以为 **NULL**。 要求  |  |  | | --- | --- | | **标头** | winuser.h (包括 Windows.h) |  另请参阅 [WM\_COPYDATA](https://learn.microsoft.com/zh-cn/windows/desktop/dataxchg/wm-copydata) |

### WM\_COPYDATA消息和其他windows消息都不一样，因为它可以用一个结构体发送消息。

# 演练

## 1.新建一个文件夹Lesson67-copydata-msg，然后在里面先新建一个MFC基于对话框的应用程序AppSend

|  |
| --- |
|  |

## 2.然后在这个解决方案里面新建一个项目，也是MFC对话框应用程序，取名：AppRecv

|  |
| --- |
|  |

## 3.把这两个对话框的默认控件删除，然后在这两个对话框类中重写PreTranslateMessage函数

|  |
| --- |
|  |
|  |

## 4.我们先来做发送端，给发送对话框添加一个编辑框控件，可以把它拉大一点，并且允许多行文本和有垂直滚动条，id为IDC\_EDIT\_DATA

|  |
| --- |
|  |
|  |
|  |
|  |

## 5.然后添加一个发送按钮，id：IDC\_BTN\_SEND

|  |
| --- |
|  |

## 6接收程序比较简单，只需要一个编辑框控件，它在程序一启动，就去接受消息，把获取到的内容显示出来就完了

|  |
| --- |
|  |
|  |
|  |

## 7.回到发送程序，给编辑框绑定一个CEdit成员变量，方便编程

|  |
| --- |
|  |
|  |

## 8.双击按钮进入点击事件函数，添加发送代码，需要先判断一下编辑框是否为空，不为空才发送

|  |
| --- |
|  |

### 运行程序，效果如下：

|  |
| --- |
|  |
|  |

## 9.为了方便编程，我们给发送程序添加一个加载按钮，点击按钮可以打开文件打开对话框，选择一个文件后可以读取文件内容显示编辑框里面这样子比较方便测试

|  |
| --- |
|  |

## 10.给加载按钮添加点击事件函数

|  |
| --- |
| void CAppSendDlg::OnBnClickedBtnLoad()  {  // TODO: 在此添加控件通知处理程序代码  CStdioFile file;  CString filename,line,content;  CFileDialog dlg(TRUE);  if(IDOK==dlg.DoModal())  {  filename = dlg.GetPathName();  file.Open(filename,CFile::modeRead);  file.ReadString(line);//这个方法一次只能读取一行  do  {  content += line;  content +="\r\n";  }  while(file.ReadString(line));  SetDlgItemText(IDC\_EDIT\_DATA,content);  file.Close();  }  } |

### 效果

|  |
| --- |
|  |
|  |
|  |

## 11.我们也想给接受程序添加按钮和另外一个编辑框，点击按钮就把接受到的内容以编辑框指定的文件名保存下来

|  |
| --- |
|  |
|  |

## 12.给大编辑框绑定一个成员变量m\_editdata

|  |
| --- |
|  |
|  |

## 13.然后给保存按钮添加点击事件函数代码

|  |
| --- |
| void CAppRecvDlg::OnBnClickedBtnSave()  {  // TODO: 在此添加控件通知处理程序代码  CStdioFile file;  CString content,title;  GetDlgItemText(IDC\_EDIT\_DATA,content);  if(content.GetLength()==0)  {  MessageBox(\_T("请输入内容"));  m\_editdata.SetFocus();  return;  }  GetDlgItemText(IDC\_EDIT\_NAME,title);  if(title.GetLength()==0)  {  MessageBox(\_T("请输入文件名"));  m\_edittitle.SetFocus();  return;  }  file.Open(title,CFile::modeCreate|CFile::modeWrite|CFile::typeText);  file.WriteString(content);  file.Close();  } |

## 14.现在回到发送程序，继续来完成按钮代码

|  |
| --- |
| void CAppSendDlg::OnBnClickedButtonSend()  {  // TODO: 在此添加控件通知处理程序代码  COPYDATASTRUCT cds;//保存发送数据的结构体  CString str;  GetDlgItemText(IDC\_EDIT\_DATA,str);  if(str.GetLength()==0)  {  MessageBox(\_T("发送的内容不能为空"));  m\_edit.SetFocus();  return;  }  // MessageBox(str);  //查找接收数据的窗口  CWnd\* pwnd = CWnd::FindWindow(NULL,\_T("AppRecv"));  if(pwnd == NULL)  {  MessageBox(\_T("查找目标窗口失败"));  return;  }  //发送数据  cds.dwData = 0;  cds.cbData = str.GetLength()\* sizeof(TCHAR);//结构体的这个成员要求的是字节数，而str.GetLength()返回的是字符个数需要乘以每一个字符的大小  cds.lpData = str.GetBuffer(cds.cbData);    pwnd->SendMessage(WM\_COPYDATA,NULL,(LPARAM)&cds);  } |

## 15.这个WM\_COPYDATA是window标准消息我们可以在这个类的属性-消息选项卡里面找到这个消息并且添加响应

|  |
| --- |
|  |

## 16.我们来完成这个OnCopyData函数，其实也比较简单

|  |
| --- |
| BOOL CAppRecvDlg::OnCopyData(CWnd\* pWnd, COPYDATASTRUCT\* pcds)  {  // TODO: 在此添加消息处理程序代码和/或调用默认值  CString str = (LPTSTR)pcds->lpData;  str = str.Left(pcds->cbData/sizeof(TCHAR));  //MessageBox(str);  SetDlgItemText(IDC\_EDIT\_DATA,str);  return CDialogEx::OnCopyData(pWnd, pcds);  } |

## 17.注意，接收应用程序在响应WM\_COPYDATA期间，按钮事件没有任何作用，应用程序处于假死状态，所以我们是按钮事件没有任何作用，我们还是把按钮个额外的控件删除，然后在WM\_COPYDATA的响应代码里面保存文件

|  |
| --- |
|  |

### 运行程序，测试一下，注意看文件夹里面是否有新文件生成，先点击发送程序加载数据按钮

|  |
| --- |
|  |

### 然后点击发送

|  |
| --- |
|  |

### 发现接收程序能够个接收并且保存到文件

## 这一节的学习到此为止，完整代码如下，主要是对话框类的源文件

## AppRecvDlg.h

|  |
| --- |
| // AppRecvDlg.h : 头文件  //  #pragma once  #include "afxwin.h"  // CAppRecvDlg 对话框  class CAppRecvDlg : public CDialogEx  {  // 构造  public:  CAppRecvDlg(CWnd\* pParent = NULL); // 标准构造函数  // 对话框数据  enum { IDD = IDD\_APPRECV\_DIALOG };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  HICON m\_hIcon;  // 生成的消息映射函数  virtual BOOL OnInitDialog();  afx\_msg void OnSysCommand(UINT nID, LPARAM lParam);  afx\_msg void OnPaint();  afx\_msg HCURSOR OnQueryDragIcon();  DECLARE\_MESSAGE\_MAP()  public:  virtual BOOL PreTranslateMessage(MSG\* pMsg);  // 绑定编辑框控件的成员变量  CEdit m\_editdata;    afx\_msg BOOL OnCopyData(CWnd\* pWnd, COPYDATASTRUCT\* pCopyDataStruct);  }; |

## AppRecvDlg.cpp

|  |
| --- |
| // AppRecvDlg.cpp : 实现文件  //  #include "stdafx.h"  #include "AppRecv.h"  #include "AppRecvDlg.h"  #include "afxdialogex.h"  #ifdef \_DEBUG  #define new DEBUG\_NEW  #endif  // 用于应用程序“关于”菜单项的 CAboutDlg 对话框  class CAboutDlg : public CDialogEx  {  public:  CAboutDlg();  // 对话框数据  enum { IDD = IDD\_ABOUTBOX };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  DECLARE\_MESSAGE\_MAP()  };  CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)  {  }  void CAboutDlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  }  BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)  END\_MESSAGE\_MAP()  // CAppRecvDlg 对话框  CAppRecvDlg::CAppRecvDlg(CWnd\* pParent /\*=NULL\*/)  : CDialogEx(CAppRecvDlg::IDD, pParent)  {  m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);  }  void CAppRecvDlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  DDX\_Control(pDX, IDC\_EDIT\_DATA, m\_editdata);    }  BEGIN\_MESSAGE\_MAP(CAppRecvDlg, CDialogEx)  ON\_WM\_SYSCOMMAND()  ON\_WM\_PAINT()  ON\_WM\_QUERYDRAGICON()  ON\_WM\_COPYDATA()  END\_MESSAGE\_MAP()  // CAppRecvDlg 消息处理程序  BOOL CAppRecvDlg::OnInitDialog()  {  CDialogEx::OnInitDialog();  // 将“关于...”菜单项添加到系统菜单中。  // IDM\_ABOUTBOX 必须在系统命令范围内。  ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);  ASSERT(IDM\_ABOUTBOX < 0xF000);  CMenu\* pSysMenu = GetSystemMenu(FALSE);  if (pSysMenu != NULL)  {  BOOL bNameValid;  CString strAboutMenu;  bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);  ASSERT(bNameValid);  if (!strAboutMenu.IsEmpty())  {  pSysMenu->AppendMenu(MF\_SEPARATOR);  pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);  }  }  // 设置此对话框的图标。当应用程序主窗口不是对话框时，框架将自动  // 执行此操作  SetIcon(m\_hIcon, TRUE); // 设置大图标  SetIcon(m\_hIcon, FALSE); // 设置小图标  // TODO: 在此添加额外的初始化代码  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  }  void CAppRecvDlg::OnSysCommand(UINT nID, LPARAM lParam)  {  if ((nID & 0xFFF0) == IDM\_ABOUTBOX)  {  CAboutDlg dlgAbout;  dlgAbout.DoModal();  }  else  {  CDialogEx::OnSysCommand(nID, lParam);  }  }  // 如果向对话框添加最小化按钮，则需要下面的代码  // 来绘制该图标。对于使用文档/视图模型的 MFC 应用程序，  // 这将由框架自动完成。  void CAppRecvDlg::OnPaint()  {  if (IsIconic())  {  CPaintDC dc(this); // 用于绘制的设备上下文  SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);  // 使图标在工作区矩形中居中  int cxIcon = GetSystemMetrics(SM\_CXICON);  int cyIcon = GetSystemMetrics(SM\_CYICON);  CRect rect;  GetClientRect(&rect);  int x = (rect.Width() - cxIcon + 1) / 2;  int y = (rect.Height() - cyIcon + 1) / 2;  // 绘制图标  dc.DrawIcon(x, y, m\_hIcon);  }  else  {  CDialogEx::OnPaint();  }  }  //当用户拖动最小化窗口时系统调用此函数取得光标  //显示。  HCURSOR CAppRecvDlg::OnQueryDragIcon()  {  return static\_cast<HCURSOR>(m\_hIcon);  }  BOOL CAppRecvDlg::PreTranslateMessage(MSG\* pMsg)  {  // TODO: 在此添加专用代码和/或调用基类  if(pMsg->message == WM\_KEYDOWN)  {  switch(pMsg->wParam)  {  case VK\_RETURN:  return TRUE;  case VK\_ESCAPE:  return TRUE;  }  }  return CDialogEx::PreTranslateMessage(pMsg);  }  BOOL CAppRecvDlg::OnCopyData(CWnd\* pWnd, COPYDATASTRUCT\* pcds)  {  // TODO: 在此添加消息处理程序代码和/或调用默认值  CString str = (LPTSTR)pcds->lpData;  str = str.Left(pcds->cbData/sizeof(TCHAR));  //MessageBox(str);  SetDlgItemText(IDC\_EDIT\_DATA,str);  CStdioFile file;  file.Open(\_T("data.txt"),CFile::modeCreate|CFile::modeWrite|CFile::typeText);  file.WriteString(str);  file.Flush();  file.Close();  return CDialogEx::OnCopyData(pWnd, pcds);  } |

## AppSendDlg.h

|  |
| --- |
| // AppSendDlg.h : 头文件  //  #pragma once  #include "afxwin.h"  // CAppSendDlg 对话框  class CAppSendDlg : public CDialogEx  {  // 构造  public:  CAppSendDlg(CWnd\* pParent = NULL); // 标准构造函数  // 对话框数据  enum { IDD = IDD\_APPSEND\_DIALOG };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  HICON m\_hIcon;  // 生成的消息映射函数  virtual BOOL OnInitDialog();  afx\_msg void OnSysCommand(UINT nID, LPARAM lParam);  afx\_msg void OnPaint();  afx\_msg HCURSOR OnQueryDragIcon();  DECLARE\_MESSAGE\_MAP()  public:  virtual BOOL PreTranslateMessage(MSG\* pMsg);  afx\_msg void OnBnClickedButtonSend();  // 绑定编辑框控件的成员变量  CEdit m\_edit;  afx\_msg void OnBnClickedBtnLoad();  }; |

## AppSendDlg.cpp

|  |
| --- |
| // AppSendDlg.cpp : 实现文件  //  #include "stdafx.h"  #include "AppSend.h"  #include "AppSendDlg.h"  #include "afxdialogex.h"  #ifdef \_DEBUG  #define new DEBUG\_NEW  #endif  // 用于应用程序“关于”菜单项的 CAboutDlg 对话框  class CAboutDlg : public CDialogEx  {  public:  CAboutDlg();  // 对话框数据  enum { IDD = IDD\_ABOUTBOX };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  DECLARE\_MESSAGE\_MAP()  };  CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)  {  }  void CAboutDlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  }  BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)  END\_MESSAGE\_MAP()  // CAppSendDlg 对话框  CAppSendDlg::CAppSendDlg(CWnd\* pParent /\*=NULL\*/)  : CDialogEx(CAppSendDlg::IDD, pParent)  {  m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);  }  void CAppSendDlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  DDX\_Control(pDX, IDC\_EDIT\_DATA, m\_edit);  }  BEGIN\_MESSAGE\_MAP(CAppSendDlg, CDialogEx)  ON\_WM\_SYSCOMMAND()  ON\_WM\_PAINT()  ON\_WM\_QUERYDRAGICON()  ON\_BN\_CLICKED(IDC\_BUTTON\_SEND, &CAppSendDlg::OnBnClickedButtonSend)  ON\_BN\_CLICKED(IDC\_BTN\_LOAD, &CAppSendDlg::OnBnClickedBtnLoad)  END\_MESSAGE\_MAP()  // CAppSendDlg 消息处理程序  BOOL CAppSendDlg::OnInitDialog()  {  CDialogEx::OnInitDialog();  // 将“关于...”菜单项添加到系统菜单中。  // IDM\_ABOUTBOX 必须在系统命令范围内。  ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);  ASSERT(IDM\_ABOUTBOX < 0xF000);  CMenu\* pSysMenu = GetSystemMenu(FALSE);  if (pSysMenu != NULL)  {  BOOL bNameValid;  CString strAboutMenu;  bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);  ASSERT(bNameValid);  if (!strAboutMenu.IsEmpty())  {  pSysMenu->AppendMenu(MF\_SEPARATOR);  pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);  }  }  // 设置此对话框的图标。当应用程序主窗口不是对话框时，框架将自动  // 执行此操作  SetIcon(m\_hIcon, TRUE); // 设置大图标  SetIcon(m\_hIcon, FALSE); // 设置小图标  // TODO: 在此添加额外的初始化代码  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  }  void CAppSendDlg::OnSysCommand(UINT nID, LPARAM lParam)  {  if ((nID & 0xFFF0) == IDM\_ABOUTBOX)  {  CAboutDlg dlgAbout;  dlgAbout.DoModal();  }  else  {  CDialogEx::OnSysCommand(nID, lParam);  }  }  // 如果向对话框添加最小化按钮，则需要下面的代码  // 来绘制该图标。对于使用文档/视图模型的 MFC 应用程序，  // 这将由框架自动完成。  void CAppSendDlg::OnPaint()  {  if (IsIconic())  {  CPaintDC dc(this); // 用于绘制的设备上下文  SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);  // 使图标在工作区矩形中居中  int cxIcon = GetSystemMetrics(SM\_CXICON);  int cyIcon = GetSystemMetrics(SM\_CYICON);  CRect rect;  GetClientRect(&rect);  int x = (rect.Width() - cxIcon + 1) / 2;  int y = (rect.Height() - cyIcon + 1) / 2;  // 绘制图标  dc.DrawIcon(x, y, m\_hIcon);  }  else  {  CDialogEx::OnPaint();  }  }  //当用户拖动最小化窗口时系统调用此函数取得光标  //显示。  HCURSOR CAppSendDlg::OnQueryDragIcon()  {  return static\_cast<HCURSOR>(m\_hIcon);  }  BOOL CAppSendDlg::PreTranslateMessage(MSG\* pMsg)  {  // TODO: 在此添加专用代码和/或调用基类  if(pMsg->message == WM\_KEYDOWN)  {  switch(pMsg->wParam)  {  case VK\_RETURN:  return TRUE;  case VK\_ESCAPE:  return TRUE;  }  }  return CDialogEx::PreTranslateMessage(pMsg);  }  void CAppSendDlg::OnBnClickedButtonSend()  {  // TODO: 在此添加控件通知处理程序代码  COPYDATASTRUCT cds;//保存发送数据的结构体  CString str;  GetDlgItemText(IDC\_EDIT\_DATA,str);  if(str.GetLength()==0)  {  MessageBox(\_T("发送的内容不能为空"));  m\_edit.SetFocus();  return;  }  // MessageBox(str);  //查找接收数据的窗口  CWnd\* pwnd = CWnd::FindWindow(NULL,\_T("AppRecv"));  if(pwnd == NULL)  {  MessageBox(\_T("查找目标窗口失败"));  return;  }  //发送数据  cds.dwData = 0;  cds.cbData = str.GetLength()\* sizeof(TCHAR);//结构体的这个成员要求的是字节数，而str.GetLength()返回的是字符个数需要乘以每一个字符的大小  cds.lpData = str.GetBuffer(cds.cbData);    pwnd->SendMessage(WM\_COPYDATA,NULL,(LPARAM)&cds);  }  void CAppSendDlg::OnBnClickedBtnLoad()  {  // TODO: 在此添加控件通知处理程序代码  CStdioFile file;  CString filename,line,content;  CFileDialog dlg(TRUE);  if(IDOK==dlg.DoModal())  {  filename = dlg.GetPathName();  file.Open(filename,CFile::modeRead);  file.ReadString(line);//这个方法一次只能读取一行  do  {  content += line;  content +="\r\n";  }  while(file.ReadString(line));  SetDlgItemText(IDC\_EDIT\_DATA,content);  file.Close();  }  } |