# 学习大纲

|  |
| --- |
|  |

## 这是一种重要的进程间通信方式。

# 演练

## 1.新建一个文件夹Lesson69-proc-comm-mem-maping,然后在这里新建一个MFC对话框应用程序取名App1，然后在这个解决方案上面点击右键-》添加-》新建项目，也是一个MFC对话框项目取名：App2，然后先把这两个对话框的默认控件删除，然后重写PreTranslateMessage函数，用来屏蔽回车键和esc键

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

## 2.把App1的布局设置如下

|  |
| --- |
|  |
|  |

## 3.给App1的编辑框控件关联一个成员变量

|  |
| --- |
|  |

## 4.双击按钮进入点击事件函数，先添加一些空处理代码

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |
|  |

## 5.我们给CApp1Dlg类添加2给成员变量

|  |
| --- |
|  |

## 6.我们在App1的对话框初始化函数代码里面添加一些初始化代码

|  |
| --- |
| BOOL CApp1Dlg::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  //创建内存映射文件，这里不是通过读取文件的方式，注意：发送方的内存的名称和接收方的内存名称需要一致  m\_hmapObj = CreateFileMapping((HANDLE)0xffffffff,NULL,PAGE\_READWRITE,0,0x1000,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("创建内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_WRITE,0,0,0);  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  } |

### 测试：编译运行程序，没有报错，说明创建内存映射文件成功

|  |
| --- |
|  |

## 7.回到按钮的点击事件代码里面，我们在这里把编辑框的内容映射到内存映射文件中

|  |
| --- |
| void CApp1Dlg::OnBnClickedBtnSend()  {  // TODO: 在此添加控件通知处理程序代码  CString str;  GetDlgItemText(IDC\_EDIT\_CTN,str);  if(str.GetLength() == 0)  {  MessageBox(\_T("请输入需要发送的内容"));  m\_editSend.SetFocus();  return;  }  //MessageBox(str);  //把获取到的数据拷贝到共享内存中  lstrcpy(m\_szmapView,str);  } |

## 8. 然后我们给App2应用程序的对话框添加一个按钮和一个编辑框

|  |
| --- |
|  |
|  |

## 9.我们先给App2对话框类添加两个成员变量

|  |
| --- |
|  |

## 10.然后我们在App2应用程序的OnInitDialog函数里面初始化这两个成员变量，注意内存的名称必须和App1里面的一致

|  |
| --- |
| BOOL CApp2Dlg::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  //创建内存映射文件，这里不是通过读取文件的方式，注意：发送方的内存的名称和接收方的内存名称需要一致  m\_hmapObj = CreateFileMapping((HANDLE)0xffffffff,NULL,PAGE\_READWRITE,0,0x1000,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("创建内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_WRITE,0,0,0);  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  } |

## 11.然后我们给App2应用程序的按钮添加点击事件代码，把内存的内容拷贝到一个CString变量里面然后再设置到编辑框

|  |
| --- |
|  |

### 重新生成所有项目，把2给项目都运行起来，效果如下

|  |
| --- |
|  |

## 12.App2在初始化函数的代码也可以这么写

|  |
| --- |
| BOOL CApp2Dlg::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  //创建内存映射文件，这里不是通过读取文件的方式，注意：发送方的内存的名称和接收方的内存名称需要一致  //m\_hmapObj = CreateFileMapping((HANDLE)0xffffffff,NULL,PAGE\_READWRITE,0,0x1000,\_T("shared\_mem"));  //这里可以不需要创建，因为App1已经创建了。这里只需要打开就可以了  m\_hmapObj = OpenFileMapping(FILE\_MAP\_READ,FALSE,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("打开内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_READ,0,0,0);//接收程序里面是读操作  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  } |

### 需要注意：一定要先运行发送程序，然后再运行接收程序，否则接收不到数据

# 其实我们也可以使用定时器，在定时器的消息响应函数里面定时获取共享内存中的内容

## 13，在解决方案文件上面点击右键-》添加-》新建项目，创建一个MFC对话框程序取名App3，也是用来作为接收程序，我们把它的默认控件删除，然后重写PreTranslateMessage函数

|  |
| --- |
|  |
|  |

## 14.给App3对话框添加2个静态文本，一个是提示信息，另外一个是显示用的

|  |
| --- |
|  |

## 15.给这个类添加2个成员变量

|  |
| --- |
|  |

## 16.在App3的OnInitDialog函数里面添加如下内容

|  |
| --- |
| BOOL CApp3Dlg::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  //这里可以不需要创建，因为App1已经创建了。这里只需要打开就可以了  m\_hmapObj = OpenFileMapping(FILE\_MAP\_READ,FALSE,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("打开内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_READ,0,0,0);//接收程序里面是读操作  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  //创建定时器  SetTimer(1,500,NULL);  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  } |

## 17.给CApp3Dialog类添加对WM\_TIMER消息的响应函数

|  |
| --- |
|  |
|  |

## 18.然后我们在OnTimer函数里面添加代码

|  |
| --- |
|  |

### 重新生成所有项目先启动App1，再启动App3，效果如下

|  |
| --- |
|  |

# 这一节的学习到此为止，完整代码如下，主要是对话框类的代码

## App1Dlg.h

|  |
| --- |
| // App1Dlg.h : 头文件  //  #pragma once  #include "afxwin.h"  // CApp1Dlg 对话框  class CApp1Dlg : public CDialogEx  {  // 构造  public:  CApp1Dlg(CWnd\* pParent = NULL); // 标准构造函数  // 对话框数据  enum { IDD = IDD\_APP1\_DIALOG };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  HICON m\_hIcon;  //添加成员变量  HANDLE m\_hmapObj;  LPTSTR m\_szmapView;  // 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 OnBnClickedBtnSend();  // 关联编辑框控件的成员变量  CEdit m\_editSend;  }; |

## App1Dlg.cpp

|  |
| --- |
| // App1Dlg.cpp : 实现文件  //  #include "stdafx.h"  #include "App1.h"  #include "App1Dlg.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()  // CApp1Dlg 对话框  CApp1Dlg::CApp1Dlg(CWnd\* pParent /\*=NULL\*/)  : CDialogEx(CApp1Dlg::IDD, pParent)  {  m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);  }  void CApp1Dlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  DDX\_Control(pDX, IDC\_EDIT\_CTN, m\_editSend);  }  BEGIN\_MESSAGE\_MAP(CApp1Dlg, CDialogEx)  ON\_WM\_SYSCOMMAND()  ON\_WM\_PAINT()  ON\_WM\_QUERYDRAGICON()  ON\_BN\_CLICKED(IDC\_BTN\_SEND, &CApp1Dlg::OnBnClickedBtnSend)  END\_MESSAGE\_MAP()  // CApp1Dlg 消息处理程序  BOOL CApp1Dlg::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  //创建内存映射文件，这里不是通过读取文件的方式，注意：发送方的内存的名称和接收方的内存名称需要一致  m\_hmapObj = CreateFileMapping((HANDLE)0xffffffff,NULL,PAGE\_READWRITE,0,0x1000,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("创建内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_WRITE,0,0,0);  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  }  void CApp1Dlg::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 CApp1Dlg::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 CApp1Dlg::OnQueryDragIcon()  {  return static\_cast<HCURSOR>(m\_hIcon);  }  BOOL CApp1Dlg::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 CApp1Dlg::OnBnClickedBtnSend()  {  // TODO: 在此添加控件通知处理程序代码  CString str;  GetDlgItemText(IDC\_EDIT\_CTN,str);  if(str.GetLength() == 0)  {  MessageBox(\_T("请输入需要发送的内容"));  m\_editSend.SetFocus();  return;  }  //MessageBox(str);  //把获取到的数据拷贝到共享内存中  lstrcpy(m\_szmapView,str);  } |

## App2Dlg.h

|  |
| --- |
| // App2Dlg.h : 头文件  //  #pragma once  // CApp2Dlg 对话框  class CApp2Dlg : public CDialogEx  {  // 构造  public:  CApp2Dlg(CWnd\* pParent = NULL); // 标准构造函数  // 对话框数据  enum { IDD = IDD\_APP2\_DIALOG };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  HICON m\_hIcon;  //添加成员变量  HANDLE m\_hmapObj;  LPTSTR m\_szmapView;  // 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 OnBnClickedBtnRecv();  }; |

## App2Dlg.cpp

|  |
| --- |
| // App2Dlg.cpp : 实现文件  //  #include "stdafx.h"  #include "App2.h"  #include "App2Dlg.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()  // CApp2Dlg 对话框  CApp2Dlg::CApp2Dlg(CWnd\* pParent /\*=NULL\*/)  : CDialogEx(CApp2Dlg::IDD, pParent)  {  m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);  }  void CApp2Dlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  }  BEGIN\_MESSAGE\_MAP(CApp2Dlg, CDialogEx)  ON\_WM\_SYSCOMMAND()  ON\_WM\_PAINT()  ON\_WM\_QUERYDRAGICON()  ON\_BN\_CLICKED(IDC\_BTN\_RECV, &CApp2Dlg::OnBnClickedBtnRecv)  END\_MESSAGE\_MAP()  // CApp2Dlg 消息处理程序  BOOL CApp2Dlg::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  //创建内存映射文件，这里不是通过读取文件的方式，注意：发送方的内存的名称和接收方的内存名称需要一致  //m\_hmapObj = CreateFileMapping((HANDLE)0xffffffff,NULL,PAGE\_READWRITE,0,0x1000,\_T("shared\_mem"));  //这里可以不需要创建，因为App1已经创建了。这里只需要打开就可以了  m\_hmapObj = OpenFileMapping(FILE\_MAP\_READ,FALSE,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("打开内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_READ,0,0,0);//接收程序里面是读操作  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  }  void CApp2Dlg::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 CApp2Dlg::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 CApp2Dlg::OnQueryDragIcon()  {  return static\_cast<HCURSOR>(m\_hIcon);  }  BOOL CApp2Dlg::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 CApp2Dlg::OnBnClickedBtnRecv()  {  // TODO: 在此添加控件通知处理程序代码  CString str;  str = m\_szmapView;  SetDlgItemText(IDC\_EDIT\_DATA,str);  } |

## App3Dlg.h

|  |
| --- |
| // App3Dlg.h : 头文件  //  #pragma once  // CApp3Dlg 对话框  class CApp3Dlg : public CDialogEx  {  // 构造  public:  CApp3Dlg(CWnd\* pParent = NULL); // 标准构造函数  // 对话框数据  enum { IDD = IDD\_APP3\_DIALOG };  protected:  virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持  // 实现  protected:  HICON m\_hIcon;  //添加成员变量  HANDLE m\_hmapObj;  LPTSTR m\_szmapView;  // 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 OnTimer(UINT\_PTR nIDEvent);  }; |

## App3Dlg.cpp

|  |
| --- |
| // App3Dlg.cpp : 实现文件  //  #include "stdafx.h"  #include "App3.h"  #include "App3Dlg.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()  // CApp3Dlg 对话框  CApp3Dlg::CApp3Dlg(CWnd\* pParent /\*=NULL\*/)  : CDialogEx(CApp3Dlg::IDD, pParent)  {  m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);  }  void CApp3Dlg::DoDataExchange(CDataExchange\* pDX)  {  CDialogEx::DoDataExchange(pDX);  }  BEGIN\_MESSAGE\_MAP(CApp3Dlg, CDialogEx)  ON\_WM\_SYSCOMMAND()  ON\_WM\_PAINT()  ON\_WM\_QUERYDRAGICON()  ON\_WM\_TIMER()  END\_MESSAGE\_MAP()  // CApp3Dlg 消息处理程序  BOOL CApp3Dlg::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  //这里可以不需要创建，因为App1已经创建了。这里只需要打开就可以了  m\_hmapObj = OpenFileMapping(FILE\_MAP\_READ,FALSE,\_T("shared\_mem"));  if(!m\_hmapObj)  {  MessageBox(\_T("打开内存映射失败"));  return FALSE;  }  //把共享内存映射到进程内存空间  m\_szmapView = (LPTSTR)MapViewOfFile(m\_hmapObj,FILE\_MAP\_READ,0,0,0);//接收程序里面是读操作  if(!m\_szmapView)  {  MessageBox(\_T("映射内存文件失败"));  return FALSE;  }  //创建定时器  SetTimer(1,500,NULL);  return TRUE; // 除非将焦点设置到控件，否则返回 TRUE  }  void CApp3Dlg::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 CApp3Dlg::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 CApp3Dlg::OnQueryDragIcon()  {  return static\_cast<HCURSOR>(m\_hIcon);  }  BOOL CApp3Dlg::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 CApp3Dlg::OnTimer(UINT\_PTR nIDEvent) //定时器函数  {  // TODO: 在此添加消息处理程序代码和/或调用默认值  CString str(m\_szmapView);  SetDlgItemText(IDC\_STATIC\_SHOW,str);  CDialogEx::OnTimer(nIDEvent);  } |