Lecture Notes on C++ Multi-Paradigm Programming

Bachelor of Software Engineering, Spring 2013

Wan Hai

whwanhai@163.com 13512768378

Software School, Sun Yat-sen University, GZ



Windows and MFC

命令行程序

```
int main(int argc ,char*argv[])
{
    FILE* file = fopen(argv[1],"r");
    .....
}
```

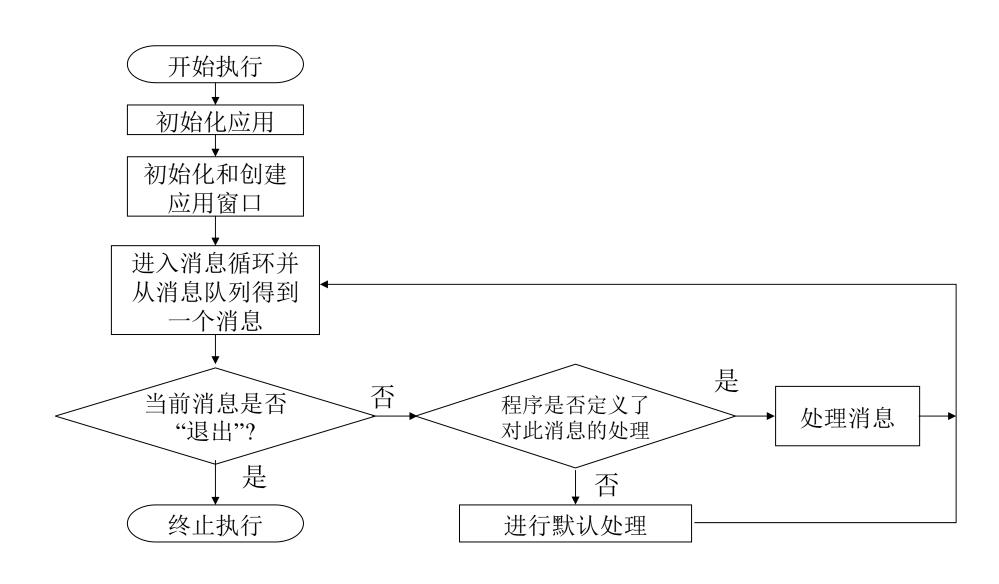
```
int WINAPI WinMain(
HINSTANCE hInstance,
//程序当前实例的句柄,以后随时可以用GetModuleHandle(0)来获得
HINSTANCE hPrevInstance,
//这个参数在Win32环境下总是0,已经废弃不用了
char * IpCmdLine,
//指向以/0结尾的命令行,不包括EXE本身的文件名,
//以后随时可以用GetCommandLine()来获取完整的命令行
int nCmdShow
//指明应该以什么方式显示主窗口
);
```



// The one and only CHelloworldApp object

CHelloworldApp theApp;

Windows程序的基本结构



WinMain()函数

- 初始化应用
- 初始化和创建应用窗口
- 进入应用程序的消息循环



窗口消息回调函数

```
LRESULT CALLBACK onMainWndMessage(HWND wnd, UINT msg, W
PARAM wParam, LPARAM IParam) {
       switch (msg) {
       case WM_DESTROY:
          PostQuitMessage(0); //如果是"窗口销毁"事件,则应该在消息
队列中投递
          break;
                      //一个WM_QUIT消息,使GetMessage()返回
FALSE
       default:
          return DefWindowProc(wnd, msg, wParam, IParam);
       return 0;
```

登记窗口类

```
bool registerMyClass() {
  WNDCLASSEX wce = \{0\};
  wce.cbSize
                 = sizeof(wce);
                = CS_VREDRAW | CS_HREDRAW;
  wce.style
                   = &onMainWndMessage; //指明回调函数
  wce.lpfnWndProc
  wce.hInstance
                  = GetModuleHandle(0);
  wce.hlcon
                 = LoadIcon(0, MAKEINTRESOURCE(IDI_WINLOGO));
  wce.hCursor
                 = LoadCursor(0, MAKEINTRESOURCE(IDC_ARROW));
  wce.hbrBackground = reinterpret_cast<HBRUSH>(COLOR_BTNFACE+1);
  wce.lpszClassName = CLASS_NAME; //独一无二的类名
  wce.hlconSm
                  = wce.hlcon:
  return 0!=RegisterClassEx(&wce);
```

创建主窗口

```
bool createMyWindow(int cmdShow) {
         HWND mainWnd = CreateWindowEx(0, CLASS_NAME, "Demo",
WS_OVERLAPPEDWINDOW,
            CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
            0, 0, GetModuleHandle(0), 0);
         if (0!=mainWnd) {
            ShowWindow(mainWnd, cmdShow);
            UpdateWindow(mainWnd);
            return true;
          } else {
            return false;
```

消息循环

```
int messageLoop() {
    MSG msg;
    while (GetMessage(&msg, 0, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return static_cast<int>(msg.wParam);
}
```



MFC简单例子



2013/6/5

```
//hello.cpp
#include <afxwin.h>
// 说明应用程序类
class CHelloApp : public CWinApp
        public:
                virtual BOOL InitInstance();
};
// 建立应用程序类的实例
CHelloApp HelloApp;
// 说明主窗口类
class CHelloWindow: public CFrameWnd
        CStatic* cs;
        public:
                CHelloWindow();
};
```



```
# 每当应用程序首次执行时都要调用的初始化函数
BOOL CHelloApp::InitInstance()
   m_pMainWnd = new CHelloWindow();
   m_pMainWnd->ShowWindow(m_nCmdShow); m_pMainWnd-
   >UpdateWindow();
   return TRUE;
// 窗口类的构造函数
CHelloWindow::CHelloWindow()
  #建立窗口本身
   Create(NULL, "Hello World!", WS_OVERLAPPEDWINDOW,
   CRect(0,0,200,200));
  #建立静态标签
   cs = new CStatic();
   cs->Create("hello world", WS_CHILD|WS_VISIBLE|SS_CENTER,
       CRect(50,80,150,150), this);
```



第一,它建立了一个应用程序对象。你所编写的每个 MFC 程序都有一个单一的程序对象,它是处理 MFC 和 Windows 的初始细节的。

第二,应用程序建立了一个窗口来作为应用程序的主窗口。最后,在应用程序的窗口中建立了一个静态文本标签,它包含有"hello world"几个字。

在第二部分中我们会仔细研究这段程序,以理解其结构。

2013/6/5

启动 VC++, 如果你是刚刚安装好,则你会在 屏幕上看到一个带有工具栏的空窗口。如果 VC++ 已经在该机器上使用过了,则所显示的窗口可能有 些不同,因为 VC++ 会记忆和自动重新打开上次使 用后退出时的项目和文件。我们需要的是它没有装 如任何项目和代码。如果程序启动后弹出对话框指 示不能打开某些文件, 你只要单击"No"即可。在 "Window"菜单中选取"Close All"选项关闭所有窗口 。在"File"菜单中选取"Close"选项来关闭其它窗口 。现在, 你就处于开始状态了。

2013/6/5

MFC库

- 类库是一个可以在应用程序中使用的相互关联的类的集合。
- MFC库——Microsoft 基本类库是一个Windows应用程序框架,它定义了应用程序的结构,并实现了标准的用户接口:
 - 管理窗口、菜单、对话框,实现基本的输入/输出和数据 存储。

MFC 4.21 类别组织框架图 (Class Hierarchy) Classes Not Derived From CObject Run-time Object Model Support CArchive Graphical Drawing Objects Arrays Application Architecture Graphical Drawing File Services CGdiOblect CArray (template) CDumpContext User objects CFile CDC CBitmap CByteArray CRuntimeClass Exceptions CMemFile CDocument CClientDC Simple Value Types CException CBrush CDWordArray COleDocument CSharedFile CMetaFileDC **CPoint** CObArray CArchiveException CFont COleLinkingDoc COleStreamFile CPaintDC CRect **CPalette CPtrArray** CMonikerFile COleServerDoc User application CWindowDC CSize CDBException CPen CStringArray CDocTemplate CAsyncMonikerFile Control Support CString CFileException CUIntArray CRgn User documents **CDataPathProperty** CDockState CTime CWordArray CinternetException CCachedDataPathProperty CMultiDocTemplate CDocitem Menus ClmageList CTimeSpan Arrays of user types CMemoryException COleObjectFactory COleCilen CSocketFile Structures **ODBC** Database Support COleTemplateServer CNotSupportedException CStdioFile Lists CCommandLineInfo CDatabase COleException COleDataSource User client items CInternetFile CList (template) **CCreateContext** CRecordset COleServerItem COleDropSource CGopherFile **CPtrList** COleSafeArray User recordsets COleDropTarget CDocObjectServerHen **CHttpFile** CObList CMemoryState CLongBinary CUserException COleMessageFilter User server items CRecentFileList CStringList CPrintInfo **DAO Database Support** CDocObjectServer Lists of user types Support Classes Window Support CCmdUl Maps CMap (template) COleCmdUI Frame Windows Controis Dialog Boxes Views CAnimateCtrl CMapWordToPtr CView CDataExchange CButton CMapPtrToWord User MDI windows CBitmapButton CMapPtrToPtr **CDBVariant** CEditView Synchronization CMapWordToOb CFieldExchange CComboBox User MDI workspaces CEdit CMapStringToPtr COleDataObject CMiniFrameWnd CHeaderCtrl CMapStringToOb COleDispatchDriver CHotKeyCtrl CMapStringToString User SDI windows CScrollView COMPFrameWnd CListBox CRectTracker Maps of user types COleBusyDialog User scroll views Control Bars CCheckListBox CWaitCursor CFormView COleChangelconDialo Internet Services Windows Sockets CDragListBox Typed Template Collections User form views CAsyncSocket CInternetSession CDialogBar CListCtrl CTypedPtrArray CInternetConnection COleConvertDialog COleResizeBar CTypedPtrList COleinsertDialog CFtpConnection CTypedPtrMap | CProgressCtrl COleLinksDialog User record views CGopherConnection Internet Server API (ISAPI) CToolBar **OLE Type Wrappers** CRichEditCtrl COleUpdateDialog CHttpConnection CHtmlStream CScrollBar CFileFind CHttpFilter Property Sheets CSliderCtrl CFtpFileFind New Class introduced with MFC 4.2 **OLE Automation Types** CHttpFilterContext CSpinButtonCtrl CGophorFileFind New class, introduced with MFC 4.0 CHttpServer Rewritten common control class CPrintDialog CGophorLocator **CHttpServerContext** Former CDK class now in MFC 4.0 CStatusBarCtrl Pre-Existing Classes **CTabCtrl** CPropertyPage User Classes CToolBarCtrl Synchronization User dialog boxes CTpolTipCtrl **CTreeCtrl**

应用程序框架

- 应用程序框架是一种类库的超集
- 在程序运行时,流程的控制多数是由框架实现的。
- 应用MFC框架来构造应用程序时,程序员的角色就是提供应用程序专用的代码,并指定这些代码是用来响应哪些消息和命令的,以使框架能够在消息和代码间建立联系。

"文档一视图"结构

- 应用程序框架的核心是"文档一视图"结构。MFC通过"文档一视图"结构为应用程序提供一种将数据与视图相分离的存储方式。
 - 文档类的作用是将应程序的数据保存在文档类对象中,以 及从磁盘文件中读或向磁盘文件中写数据。
 - 视图类的作用是显示数据和编辑数据。