Windows编程基础

第二章 MFC简介及第一个MFC程序

Programming Your Future

本课程旨在向学员介绍：

1）了解MFC类库

2）理解第一个MFC程序执行过程

编写Windows程序

- 编写Windows应用程序主要有三种方法
  - 方法1：调用Windows环境提供的Win32 API函数(C语言方法)
    - 大量程序代码由用户自己编写
  - 方法2：使用MFC(微软基础类库)直接编写
    - 提供大量预先编好的类和支持代码
  - 方法3：使用MFC和向导(Wizards)编写
    - 用AppWizard来生成Windows应用程序框架

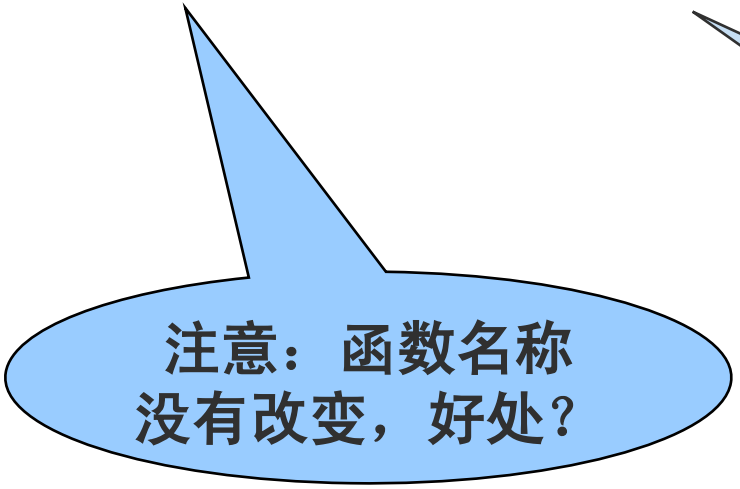
本章内容

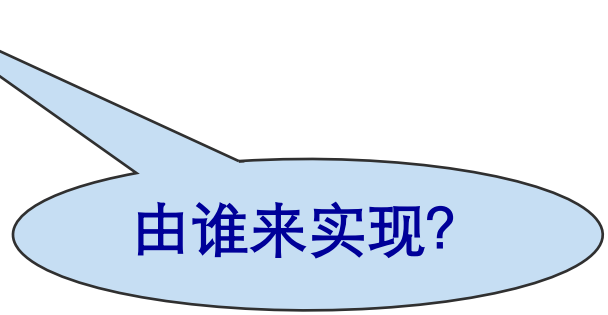
MFC类库产生原因

- 用Windows API方式编程的碰到的问题:
  - 熟悉Windows内部原理
  - 手工编写代码（数以千计的API函数）
- 回忆开发环境自动产生的代码
- 如果界面再丰富些，菜单再多一些,会怎么样?
  - 如何解决？
  - 利用面向对象的思想解决

关于面向对象的思想

- 面向对象程序设计语言可以将一些变量和函数封装到类中
  - 当变量被类封装后，称之为属性或数据成员。
  - 当函数被封装后称之为方法或成员函数
  - 对Windows API函数进行封装，如Windows API的LoadIcon函数被转化为CWinApp的成员函数。





MFC简介-1/2

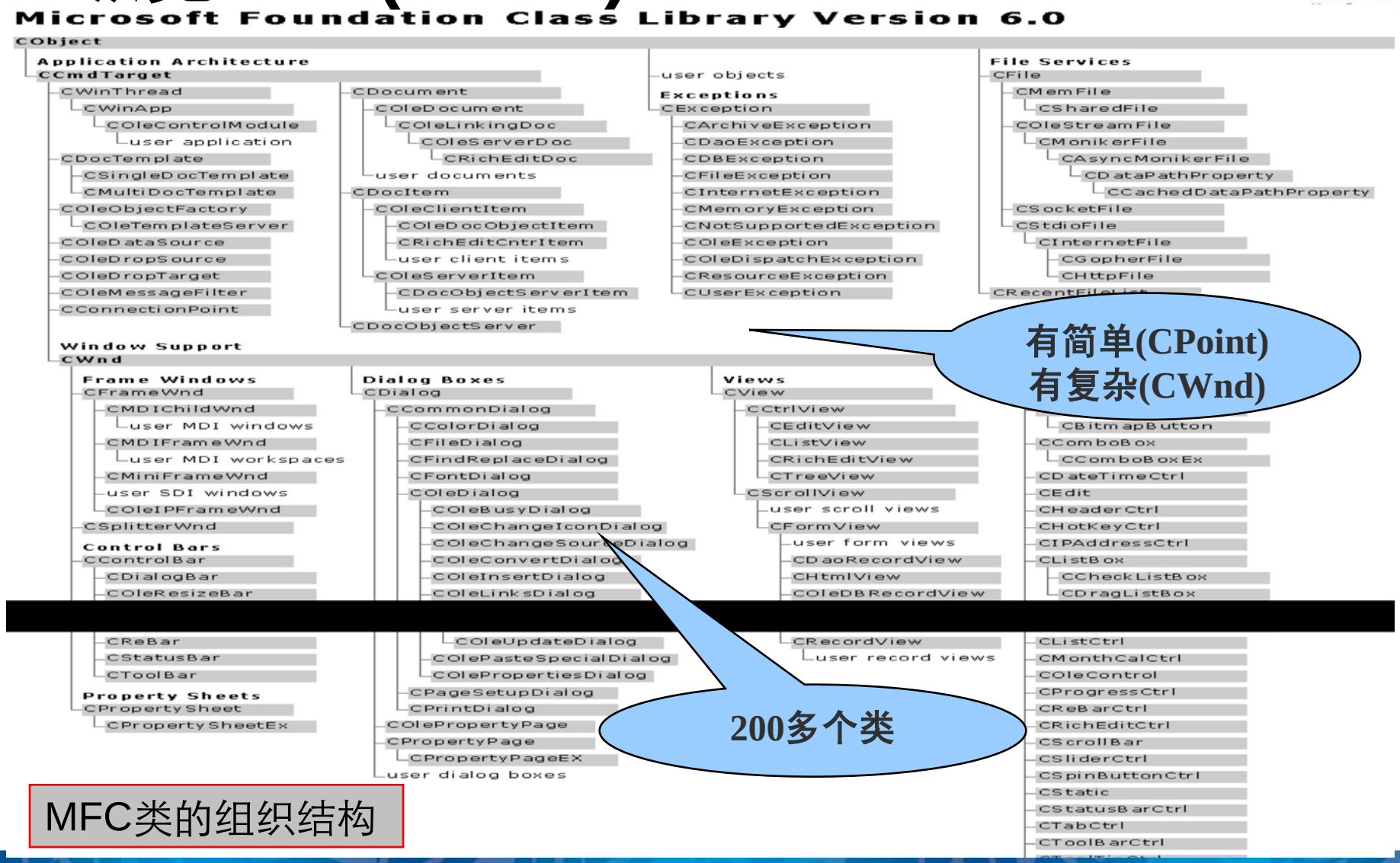
- MFC全称：Microsoft Foundation Classes，是微软把Window API进行封装的类库，该类库以层次结构组织起来，其中封装了大部分Windows API函数和Windows控件。1989 年微软公司成立 Application Framework 技术团队，名为 AFX 小组，用以开发 C++面向对象工具给 Windows 应用程序开发人员使用。AFX 的 “X” 其实没有什么意义，只是为了凑成一个响亮好念的名字。
- 应用MFC编程的好处：使Windows程序员能够利用C++面向对象的思想进行编程。
- 出发点：**有好的类库做出发点、减少代码编写量**
  - 添加工具条：有CToolBar.
  - 动态数组：有CList,CArray

MFC简介-2/2

Programming Your Future

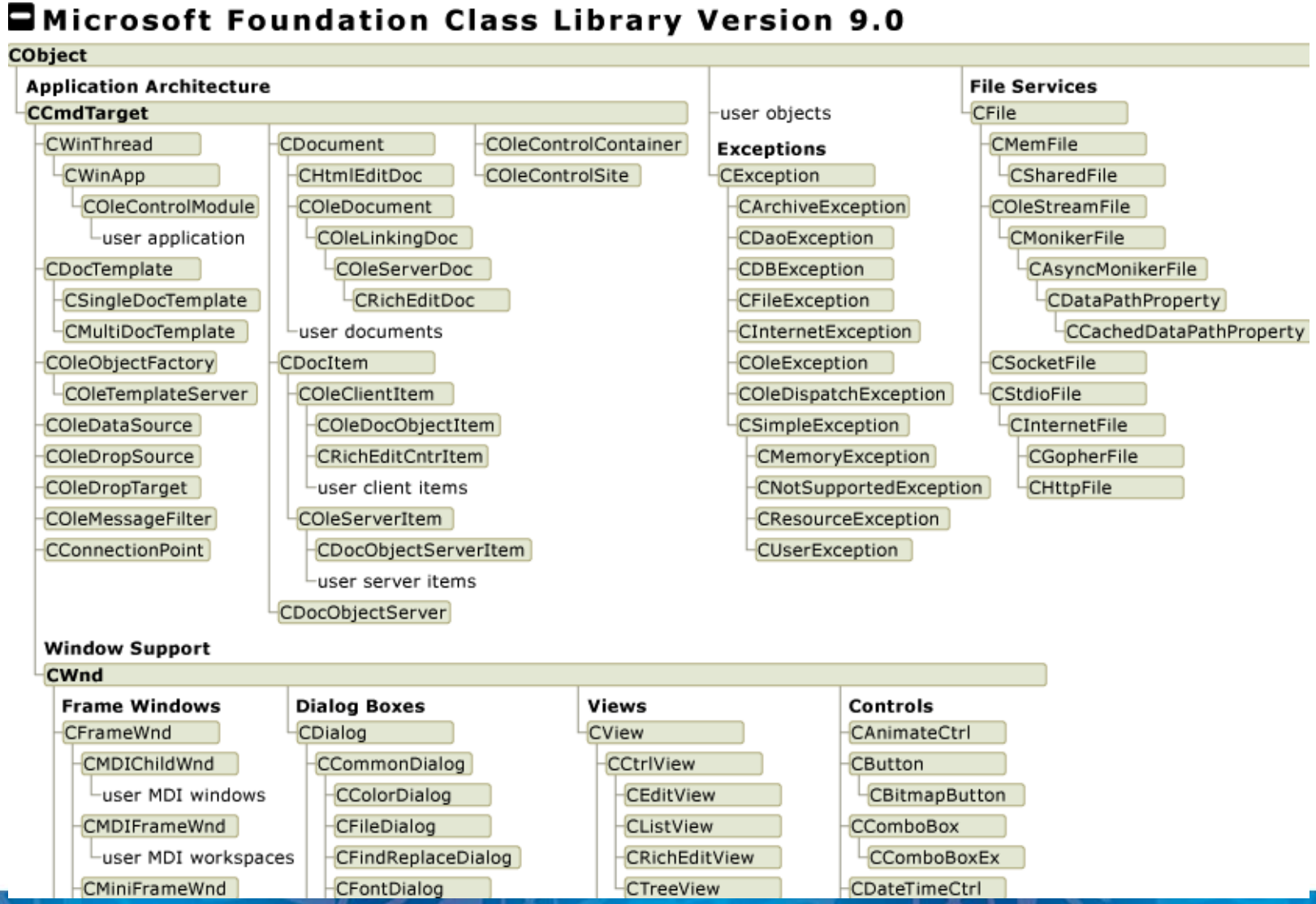
- 两个层次：
  - 首先，我们先会用这些类
  - 其次，思考AFX小组究竟怎么抽象出来的这些类。

纵览MFC(VC6.0)



MFC类的组织结构

纵览MFC(.net)


Microsoft Foundation Class Library Version 9.0
CObject
Application Architecture
CCmdTarget
CWinThread
CWinApp
COleControlModule
user application
CDocTemplate
CSingleDocTemplate
CMultiDocTemplate
COleObjectFactory
COleTemplateServer
COleDataSource
COleDropSource
COleDropTarget
COleMessageFilter
CConnectionPoint
CDocument
CHtmlEditDoc
COleDocument
COleLinkingDoc
COleServerDoc
CRichEditDoc
user documents
CDocItem
COleClientItem
COleDocObjectItem
CRichEditCntrItem
user client items
COleServerItem
CDocObjectServerItem
user server items
CDocObjectServer
COleControlContainer
COleControlSite
user objects
Exceptions
CException
CArchiveException
CDaoException
CDBException
CFileException
CInternetException
COleException
COleDispatchException
CSimpleException
CMemoryException
CNotSupportedException
CResourceException
CUserException
File Services
CFile
CMemFile
CSharedFile
COleStreamFile
CMonikerFile
CAsyncMonikerFile
CDataPathProperty
CCachedDataPathProperty
CSocketFile
CStdioFile
CInternetFile
CGopherFile
CHttpFile
Window Support
CWnd
Frame Windows
CFrameWnd
CMDIChildWnd
user MDI windows
CMDIFrameWnd
user MDI workspaces
CMiniFrameWnd
Dialog Boxes
CDialog
CCommonDialog
CColorDialog
CFileDialog
CFindReplaceDialog
CFontDialog
Views
CView
CCtrlView
CEditView
CListView
CRichEditView
CTreeView
Controls
CAnimateCtrl
CButton
CBitmapButton
CComboBox
CComboBoxEx
CDateTimeCtrl

MFC类定义头文件

- MFC类定义头文件:“afxwin.h”

```
class CSize;
class CPoint;
class CRect;

//CObject
  //CException
   //CSimpleException
          class CResourceException
          class CUserException;

  class CGdiObject;     // CDC drawing tool
  class CPen;             // a pen / HPEN wrapper
  class CBrush;          // a brush / HBRUSH wrapper
  class CFont;            // a font / HFONT wrapper
  class CBitmap;        // a bitmap / HBITMAP wrapper
  class CPalette;        // a palette / HPALLETE wrapper
  class CRgn;             // a region / HRGN wrapper
…
```

全局函数

- 不属于任何类，以afx开头

AfxWinInit	被winMain调用的函数，初始化
AfxBeginThread	开始一个新线程
AfxEndThread	结束一个旧线程
AfxMessageBox	Api函数messagebox
AfxGetApp	获得程序对象的指针
AfxGetMainWnd	获得程序主窗口指针
AfxGetInstanceHandle	获得程序实例句柄
AfxRegisterClass	注册窗口类

CSDN一篇文章-1/2

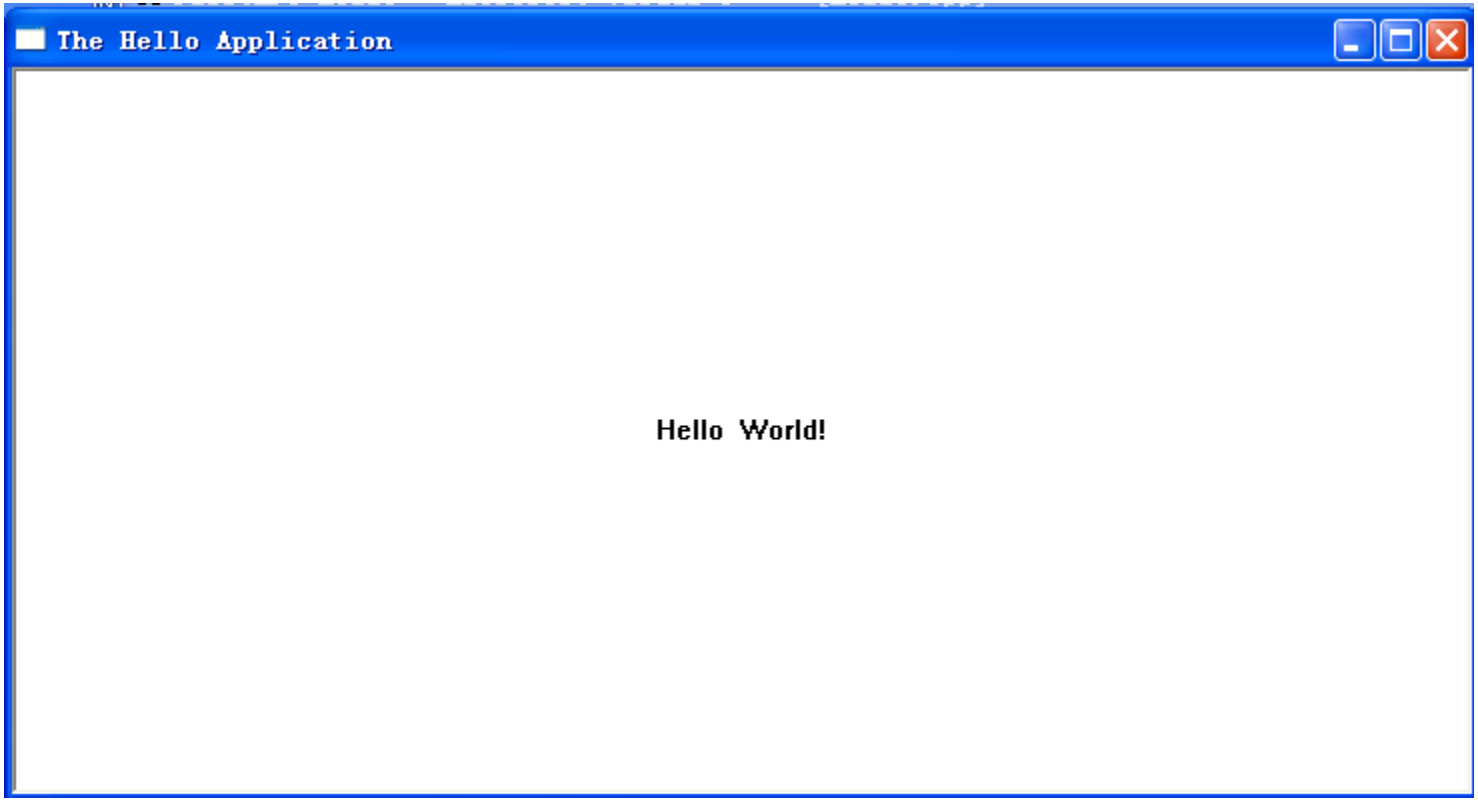
- Hello World文化一脉相承、多种语言百花齐放
- C语言　　C++

CSDN一篇文章-2/2

- JAVA MFC

简化的MFC程序运行效果

- Hello应用程序



简化的MFC程序文件说明

Programming Your Future

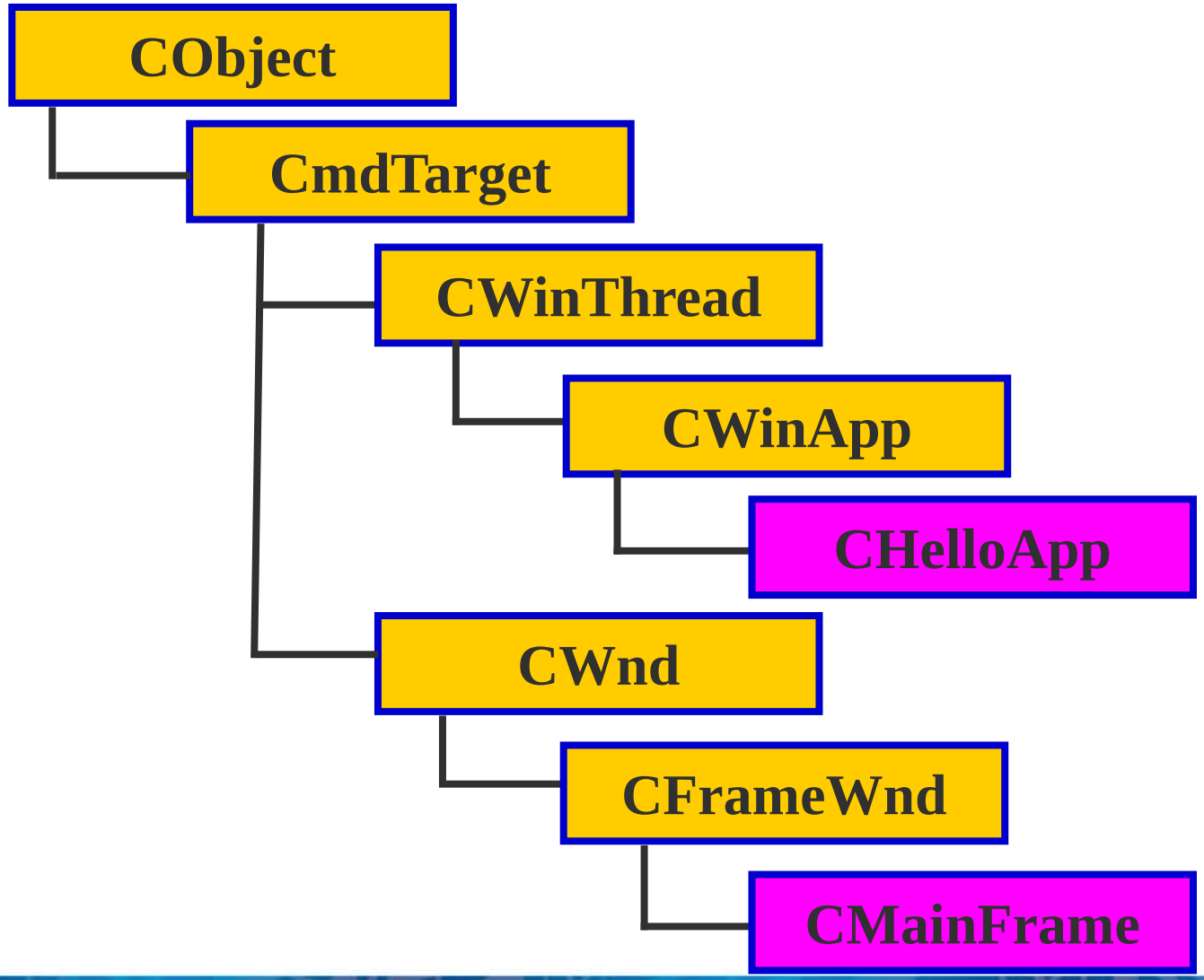
- 习惯：
  - 把类的定义放在.h文件中，把源程序放在.cpp文件中，随着代码量增加，这种存放方法的优势将更加明显。
  - 窗口拥有哪些功能？
    - 可以移动、调整大小、最小、最大化及正常关闭等

简化的MFC程序如何运行？

Programming Your Future

- 使用Windows API进行Windows应用程序的开发的好处是可以清楚地看见整个程序的来龙去脉，但MFC程序把这些内容都包起来了。
- 一系列问题：
  - WinMain在哪里？
  - RegisterClass,CreateWindow在哪？
  - 消息循环在哪？
  - 窗口过程函数如何执行？

简化的MFC程序类图


CObject
CmdTarget
CWinThread
CWinApp
CHelloApp
CWnd
CFrameWnd
CMainFrame

基本类-1/4

- 接下来我们开始学习几个非常重要的类：
  - CObject,
  - CCmdTarget,
  - CWinThread,
  - CWinApp
  - CWnd
  - CFrameWnd
- CObject类
  - Microsoft基本类库中的大多数类都是由CObject类派生而来的。CObject对所有由它派生出的类提供了有用的基本服务,包括动态类信息、动态创建(Dynamic)、对象序列化(Serialize)等等

基本类-2/4

- CCmdTarget类
  - CCmdTarget 是 MFC 消息映射结构的基类。消息映射将命令或消息发送给应用程序编写的处理命令或消息的响应函数。具有消息处理能力的类均应为该类的派生类。
- CWinThread 类
  - Microsoft 的类库支持多个线程的并行执行。每个应用程序至少有一个线程（被称作主线程）。CWinThread 封装了操作系统的调度功能。

基本类-3/4

- CWinApp类：CWinApp 封装了 Windows 应用程序初始化、运行、终止应用程序的代码。基于框架建立的应用程序必须有一个且只有一个从CWinApp派生的类对象。
  - 三个重要的函数
  - virtual BOOL InitApplication();
  - virtual BOOL InitInstance(); //Cre
  - virtual int Run(); //消息循环
  - 注意：CWinApp是从CWinThread派生而来，因此在构造该对象时会调用beginthreadex和endthreadex来创建和结束线程。

该类是核心

完成WinMain函数工作

基本类-4/4

- CWnd类
  - CWnd是各种窗口、对话框和控制框的通用基类，提供窗口处理中公共的窗口类注册、窗口创建与撤消等处理操作。
- CFrameWnd类
  - 提供了一个 Windows 单文档界面，该界面具有重叠或弹出功能，并且可以通过成员函数实现对窗口的某些控制操作。

简化的MFC程序启动过程-1/2

- 1. 构造全局对象—CWinApp派生类对象；
- 2. 运行由应用程序框架提供的WinMain函数； WinMain是MFC程序早已准备好,并由连接器直接加入到代码中了.
- 3. WinMain调用AfxWinMain函数，通过AfxGetApp获得全局对象的指针pApp，调用全局函数AfxWinInit，为CWinApp的成员变量m_hInstance, m_hPrevInstance, m_lpCmdLine,m_nCmdShow赋初值；
- 4. 调用pApp->InitApplication，这是CWinApp的虚函数，一般不需要改写；

简化的MFC程序启动过程-2/2

- 5. 调用 pApp->InitInstance，每个程序都必需改写这个函数，进行应用程序初始化；在InitInstance函数中，先用new构造一个CFrameWnd派生类对象，其构造函数又调用Create，创建主窗口，MFC依次自动为应用程序注册窗口类；

注意看定义

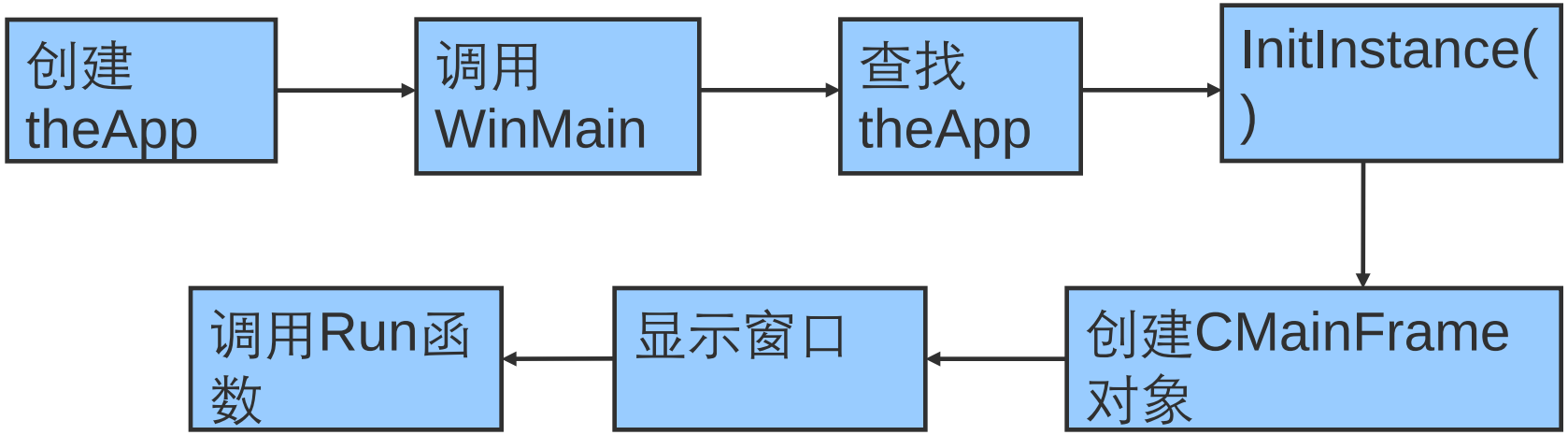
- 6. 然后调用ShowWindow显示窗口，调用UpdateWindow，发出WM_PAINT消息；
- 7. 回到AfxWinMain中，调用pApp->Run，进入消息循环；

简化的MFC程序结束过程

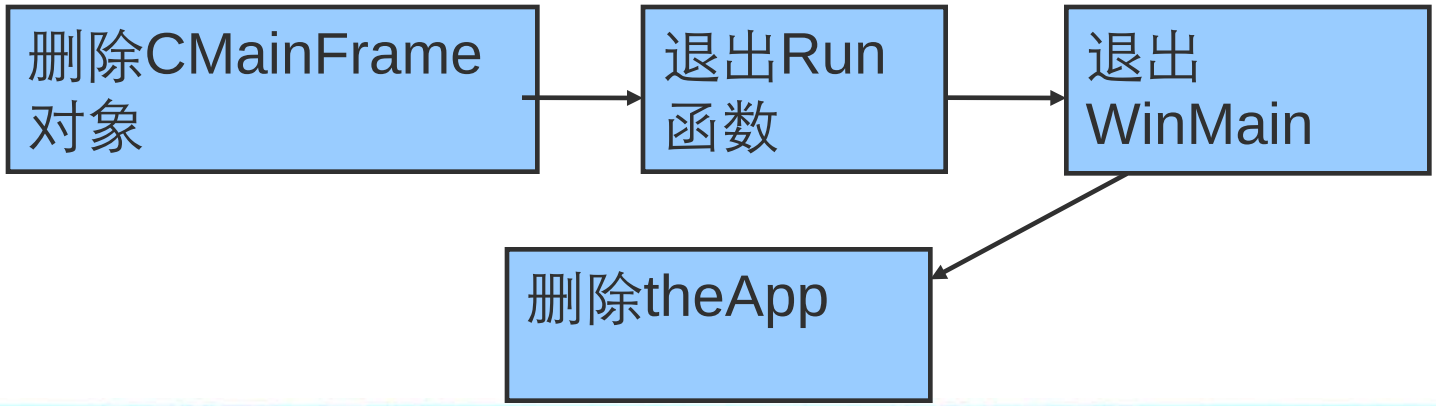
- 1. 若用户选择菜单File/Close，则程序收到WM_CLOSE消息，调用::DestroyWindow发出WM_DESTROY消息.
- 2. 调PostQuitMessage，发出WM_QUIT消息，此时Run会结束其内部消息循环.
- 3. 调用ExitInstance；最后，返回AfxWinMain，执行WinTerm，结束程序运行。

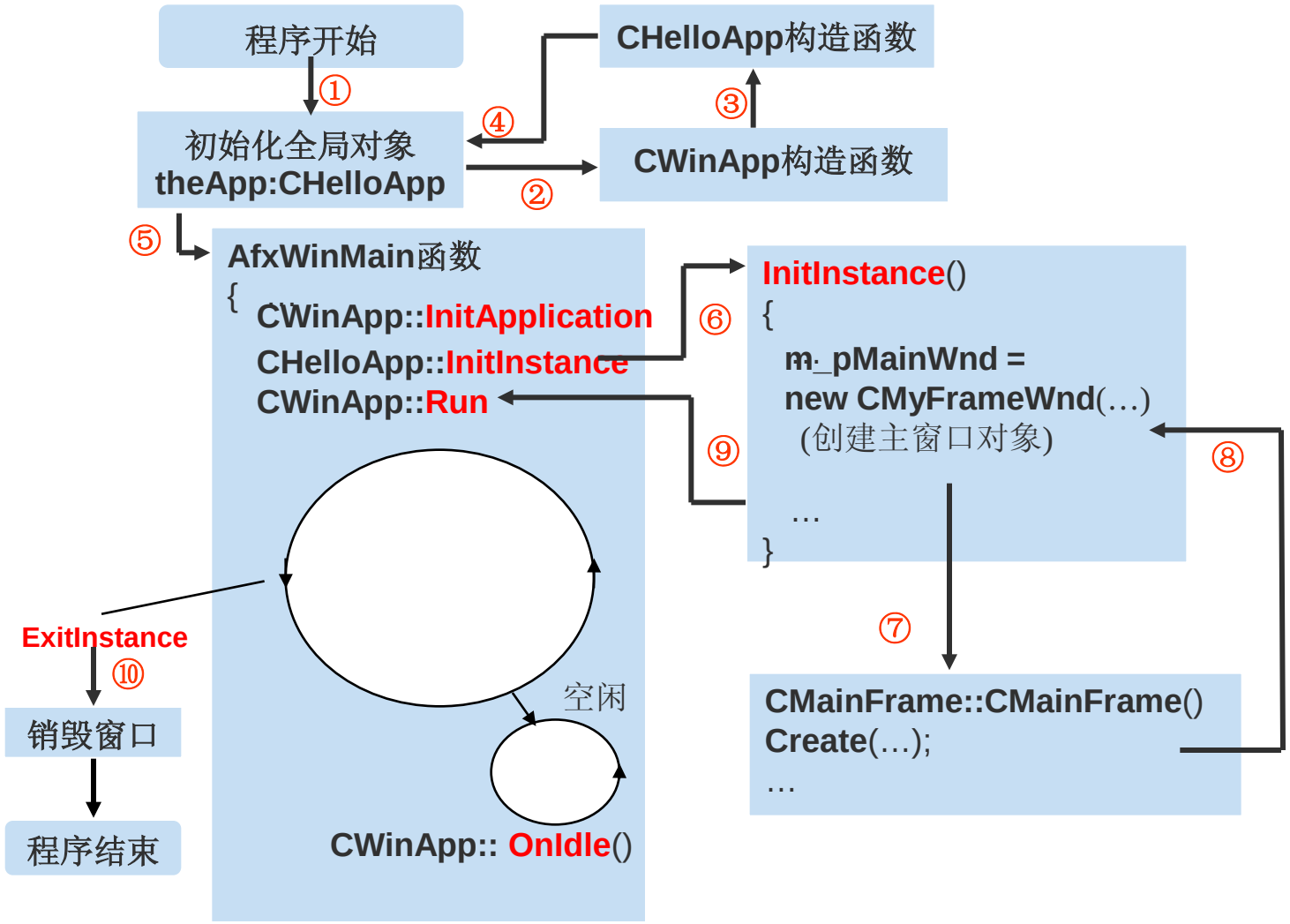
简化的MFC程序启动和结束过程

- 开始



- 结束




程序开始
①
初始化全局对象
theApp:CHelloApp
④
CHelloApp构造函数
③
CWinApp构造函数
②
⑤
AfxWinMain函数
{
CWinApp::InitApplication
CHelloApp::InitInstance
CWinApp::Run
⑥
InitInstance()
{
m_pMainWnd =
new CMyFrameWnd(…)
(创建主窗口对象)
…
}
⑨
⑧
⑦
CMainFrame::CMainFrame()
Create(…);
…
空闲
CWinApp:: OnIdle()
ExitInstance
⑩
销毁窗口
程序结束
…
}

应用程序类-1/3

- MFC应用程序的核心就是基于CWinApp类的应用程序对象。CWinApp提供了消息循环来检索消息并将消息调度给应用程序窗口。它还包括可以被覆盖的、用来自定义应用程序的主要虚函数。
- 实现方法：
  - 包含AfxWin.h
  - 在Hello.cpp中使用如下语句：
    - CHelloApp theApp;

应用程序类-2/3

- Hello.h中包含了类的声明代码

```
class CHelloApp : public CWinApp
{
public:
    virtual BOOL InitInstance ();
};
```

- Initinstance函数调用比较早，在应用程序开始运行之后而窗口创建之前被调用。
- 注意：除非InitInstance创建一个窗口，否则应用程序不会有窗口。这正是为什么即使最小的MFC程序也必须重载Initinstance函数的原因。

应用程序类-3/3

```
BOOL CHelloApp::InitInstance ()
{   m_pMainWnd = new CMainFrame;
    m_pMainWnd->ShowWindow (m_nCmdShow);
    m_pMainWnd->UpdateWindow ();
    return TRUE;
}
```

- 构造了一个CMainFrame对象，并将其地址复制到了应用程序对象m_pMainWnd数据成员中。
- ShowWindow和UpdateWindow是CWnd类的成员函数它是对Windows API函数的封装
- m_nCmdShow是CWinApp类的成员

关于框架窗口-1/3

Programming Your Future

- 注意： CWinThread有一个重要的成员
  - CWnd* m_pMainWnd
  - 我们需要定义一个此类型或其派生类型的对象。
  - 如何实现？
    - m_pMainWnd = new CMainFrame;
  - 为什么定义为指针
    - 首先，因为我们要创建窗口，而这个窗口在InitInstance函数运行之后依然存在，所以不能是局部对象。
    - 如果是全局对象，因为创建窗口需要得到当前窗口的实例句柄，而定义全局对象时实例句柄还不存在，没有办法，只好在堆上申请空间。

关于框架窗口-2/3

- 在API开发程序时我们使用的是CreateWindow函数，MFC使用从CWnd继承下来的Create函数来完成。

```
BOOL Create(LPCTSTR lpszClassName,
        LPCTSTR lpszWindowName,
        DWORD dwStyle  = WS_OVERLAPPEDWINDOW,
        const RECT &rect = rectDefault, //窗口矩形区域
        CWnd * pParentWnd = NULL,
        LPCTSTR lpszMenuName = NULL,
        DWORD dwExStyle = 0,
        CCreateContext * pContext = NULL)
```

关于框架窗口-3/3

- Create接收的8个参数中的6个默认值定义。我们可以只指定前两个参数。
  - 参数1：lpszClassName指定了窗口基于WNDCLASS类的名称。如果设置为NULL，则创建一个默认风格的框架窗口
  - 参数2：窗口的标题
- 其它风格举例
- Create(NULL, _T(“Hello”),WS_OVERLAPPEDWINDOW|WS_VSCROLL);
- 就创建窗口来说，MFC比Windows API编程容易多了。

窗口的绘制

Programming Your Future

- 相关消息？
  - WM_PAINT
- 处理函数
  - CMainWindow::OnPaint来处理
  - CPaintDC dc(this);
- MFC的CPaintDC类是从MFC的更为一般的CDC类派生的,为WM_PAINT消息服务。
- 调用CWnd::GetClientRect来使用窗口客户区的坐标来初始化这个矩形
- CDC::DrawText完成文本的输出，有四个参数

关于消息映射-1/2

Programming Your Future

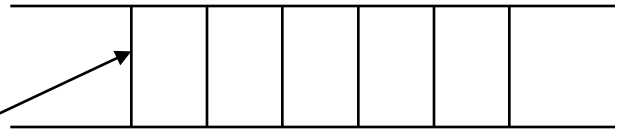
- 三个重要的宏
- 在头文件添加如下消息宏
  - DECLARE_MESSAGE_MAP()
- 在源文件中添加如下消息宏
  - BEGIN_MESSAGE_MAP()
  - END_MESSAGE_MAP()

关于消息映射-2/2

```
//WinMain函数
int AFXAPI AfxWinMain(...)
{   ...
    int nReturnCode = -1;
    CWinThread* pThread = AfxGetThread();
    CWinApp* pApp = AfxGetApp();
    if (!AfxWinInit(...))
    ...
    if (!pThread->InitInstance())
    {...
    }
    nReturnCode = pThread->Run();
```

CWinApp::Run()

```
int CWinThread::Run()
{   ...
    for (;;)
    {
        while (bIdle &&!::PeekMessage(...))
        {
            if (!OnIdle(lIdleCount++))
                bIdle = FALSE;
        }
        do
        {
            if (!PumpMessage())
                return ExitInstance();
        } while (::PeekMessage(...));
    }
```



WM_PAINT

消息映射表

WM_LBUTTONDOWN	OnLButtonDown
WM_PAINT	OnPaint

自己手动编写这个程序-1/2

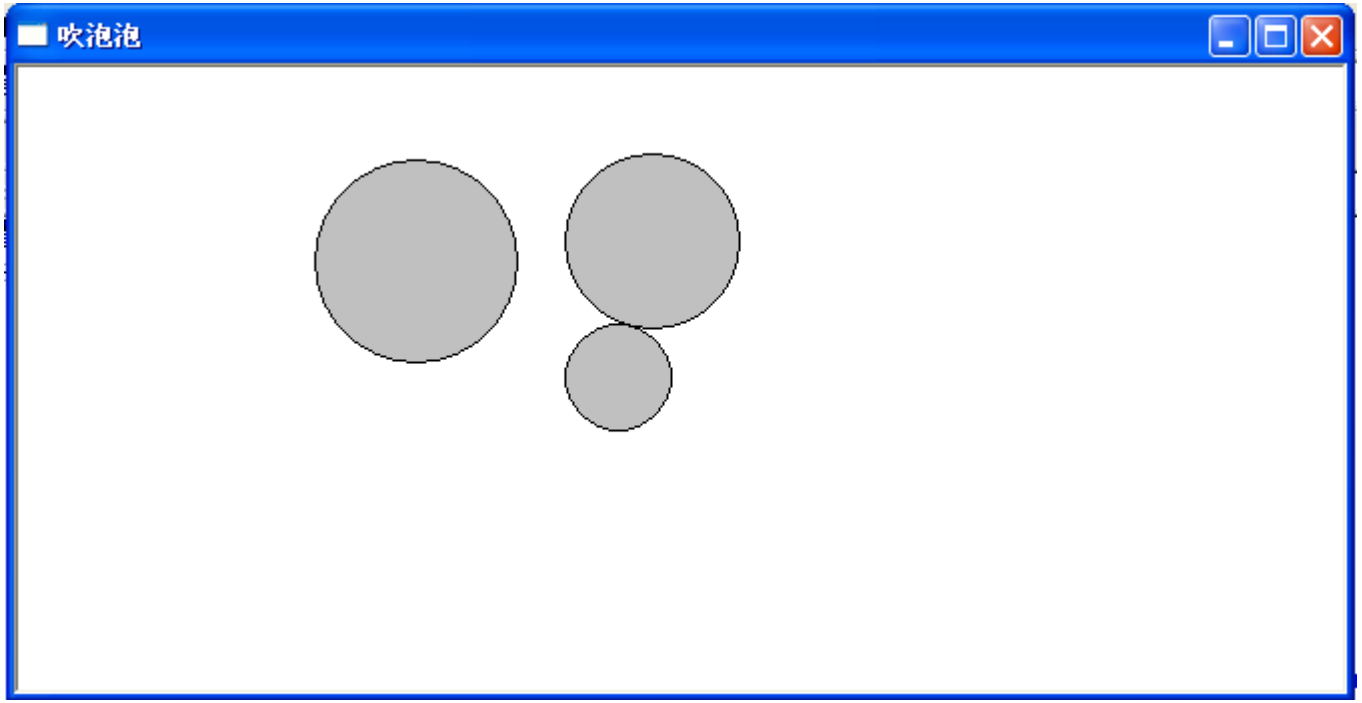
- 换个角度回忆一下我们这个需求
  - 想在一个界面上显示文本信息，如何实现？
  - 针对我们这个需求，大家的思考模式是怎样的呢？
  - 我们说所有的MFC程序都要包含一个有CWinApp派生的类，所以这个是其中一个待实现的类。
  - 另外还需要一个显示文档的界面，从哪个类派生，MFC给我们提供了一个现成的类就是CFrameWnd，所以我们的实现要从这个类派生。

自己手动编写这个程序-2/2

- 实现步骤，首先，大家要对程序执行过程有个了解，回忆一下我们上一章讲的流程。
- 注意：我们选择的工程类型是CWin32类型。
- 关于派生自CWinApp类的类，我们需要重载InitInstance()这个函数来做初始化的工作。
- 关于派生自CFrameWnd的类，需要两个函数，一个是构造函数，用于创建窗口，另一个是响应WM_PAINT消息的函数。

实现Bubble(吹泡泡)程序-1/2

- 巩固大家所学习的知识
- 实例：吹泡泡程序，每当用户在窗口客户区按下鼠标左键时即可产生一个泡泡



实现Bubble(吹泡泡)程序-2/2

Programming Your Future

- 设计思想：显示一个泡泡所需的数据包括其位置和大小，在MFC中可用其包含矩形表示。可设置一数组，每当用户按下鼠标左键时，就产生一个泡泡的数据存入数组中，再由框架窗口类的OnDraw() 函数显示所有的泡泡。
- 注意：需要哪些数据？
- 一共有哪几个类？
- 代码量不大，但要注意理解
- 问题：第一个圆的大小是否是随机的，如何产生一个真正的随机大小的程序