第四部分　附　　录

附录A　消息和重载顺序

在第一章中，对MFC基础特征及其封装Windows API机制进行了回顾，并重点讨论了窗口如何创建、窗口间如何进行消息发送，　MFC应用程序由什么类组成，以及如何使用ClassWizard来重载并以此增强程序的处理能力。在该章中未深入涉及错误重载的有关内容，这也是一个导致编程失败的重要因素和来源，即使对一个熟练的　MFC程序员而言也是如此。换句话说，错误重载包括为错误的消息添加了窗口消息处理函数或者重载了错误的　MFC类函数等等。为避免出现错误重载，在此需要了解消息发送、函数调用以及应用程序中各种行为的发生顺序，包括：

- 创建、移动、显示和关闭窗口。
- 创建和关闭有模式对话框或者无模式对话框。
- 创建和关闭对话框应用程序。
- 创建和关闭MDI应用程序以及装载和保存MDI文档。
- 创建和关闭MDI应用程序以及装载和保存MDI文档。

在本附录中将把以上行为所涉及到的消息和重载归纳成文档。如果附录中的下列实例都不适用于读者自己的应用程序，可以自行使用 ClassWizard添加并重载每个可能的消息处理函数，然后将它们加入到受其影响的类中，并设置断点，通过这种方式来创建自己的事件发生序列表。

注意　这些事件序列极少归档。可能的一个原因在于它们经常随着不同MFC版本而经常修改。一般情况下本附录所述的序列表很少被修改。

A.1　窗口

这一节中将列出在创建、显示和关闭任何类型的窗口 (包括重叠窗口、弹出窗口和子窗口) 的情况下用到的消息和所调用的函数。

注意　如果窗口正在由对话框创建，例如一个控件，则消息以WM_CTLCOLOR开始。

A.1.1　创建窗口

一旦调用了CWnd的CreateEx()函数，便开始消息发送和函数的调用过程。可以看到，在Windows API被调用之前，还可以用PreCreateWindow()来把握最后的机会以改变创建过程。

以上便是创建窗口的过程，注意窗口还没有绘制在屏幕上，系统将随后发送　WM_PAINT和WM_NCPAINT消息。

表A.1 创建窗口

处理函数	类 型	描 述	参见注释
PreCreateWindow()	重载函数	允许在窗口被创建之前改变创建参数	第1条
PreSubclassWindow()	重载函数	允许用户可以首先对窗口子类化	第2条
OnGetMinMaxInfo()	消息处理函数	允许设置对窗口大小的限制	第3条
OnNcCreate()	消息处理函数	通知窗口的非客户区将要创建	第4条
OnNcCalcSize()	消息处理函数	允许改变窗口客户区的大小	第5条
OnCreate()	消息处理函数	通知窗口已经被创建	第6条
OnSize()	消息处理函数	通知窗口大小正在改变	第7条
OnMove()	消息处理函数	通知窗口正在移动	第7条
OnChildNotify()	重载函数	其调用作为由消息反射过程的一部分，以通知父窗口其子窗口已经创建。该函数仅能被子窗口调用	第8条

注释

1) 应该重载CWnd::PreCreateWindow()函数来改变用于创建窗口的创建参数。为什么不只用正确的参数来调用CreateEx()呢？这是因为，有时用户无权访问它。例如，当应用程序打开主窗口的时候便是这种情况。应用程序启动的时候采用缺省的窗口风格和窗口类，通过调用该函数就获得了改变它们的机会。由于应用程序自动打开窗口的这种情况不是很多，因此该成员函数通常都在CMainFrame而不是在其他类中重载。

2) 应当重载CWnd::PreSubclassWindow()成员函数，使其首先对窗口进行子类化。有时这是很重要的。因为最后子类化窗口的窗口过程有可能接收不到它所需要的所有消息。例如， MFC为CDragListBox类重载该函数，并在用户子类化该控件之前使该函数首先对它进行子类化。

3) WM_GETMINMAXINFO消息由系统发出，它可以防止窗口变得太大或者太小。例如，用户拖动一个可重置大小的窗口一角时便发出该消息。

4) WM_NCCREATE消息由系统发出，用于通知窗口非客户区将要创建。这里不要为了意图改变缺省窗口过程而修改其所传递的参数。 MFC保存这个原始消息并在调用缺省窗口过程时使用它。但是，可以调用自己的缺省窗口过程，这样可以安全地绕过 MFC的窗口过程。

5) WM_NCCALCSIZE消息由系统发出，允许改变窗口客户区的大小。客户区通常是除窗口边界和滚动条等占用空间(即已经绘制的非客户区)之外的剩余空间。

6) WM_CREATE消息由系统在其创建窗口之后发出。此时适于创建窗口自己的所属窗口(例如控件窗口)。

7) 发送WM_SIZE和WM_MOVE消息是为了定位窗口位置和重置窗口大小。其依赖的参数由::CreateWindowEx()函数设定。此时适合于重新改变所属窗口的大小，特别是这些所属窗口的大小依赖于窗口尺寸时更是如此。重叠窗口不能获取这些消息。但它们在用ShowWindow()显示窗口的时候便可以获取这些消息。

8) WM_PARENTNOTIFY消息由控制该窗口的父窗口MFC类反射。WM_ PARENTNOTIFY消息此时被发送到父窗口以通知它子窗口已经被创建。但是，因为该窗口首先从 WM_CREATE消息得知了这一情况，所以该消息实际上并没有太多的用处，只有子窗口才能得到该消息。

A.1.2 移动窗口

以下调用序列在用MoveWindow()函数移动和/或者重置窗口大小的时候发生。

表A-2 移动窗口

处 理 函 数	类 型	描 述	参见注释
OnWindowPosChanging()	消息处理函数	通知窗口的X,Y甚至Z坐标将要改变	第9条
OnGetMinMaxInfo()	消息处理函数	允许设置窗口大小的限制	第3条
OnNcCalcSize()	消息处理函数	允许改变窗口客户区的大小	第5条
OnWindowPosChanged()	消息处理函数	通知隐藏的窗口被显示并且WM_SIZE和WM_MOVE消息将要发送	第10条
OnSize()	消息处理函数	通知窗口消息正在改变	第7条
OnMove()	消息处理函数	通知窗口正在移动	第7条

注释

9) WM_WINDOWPOSCHANGING消息由系统发出以响应改变窗口位置的函数，或者以x-y坐标或者其z坐标的形式表明窗口将要移动。

10) WM_WINDOWPOSCHANGED消息由系统在它显示了隐藏窗口之后，但在发送 WM_SIZE和WM_MOVE消息之前发出。显然此时重载该消息处理函数并发送 WM_SIZE和WM_MOVE消息是很有效率的。

A.1.3 显示子窗口

以下调用序列在使用ShowWindow()函数以显示子窗口的发生时刻。

表A-3 显示子窗口

处理函数	类 型	描 述	参见注释
OnShowWindow()	消息处理函数	由ShowWindow()函数发出	第11条
OnWindowPosChanging()	消息处理函数	通知窗口的x-y甚至z坐标将要改变	第9条
OnWindowPosChanged()	消息处理函数	通知隐藏的窗口已经显示，将要发送WM_SIZE和WM_MOVE消息	第10条

注释

11) WM_SHOWWINDOW消息由ShowWindow()函数发出，这里所要完成的工作最好在OnCreate或者OnSize()函数中完成。

A.1.4 显示重叠窗口或弹出窗口

对于重叠窗口和弹出窗口。其调用序列：

表A-4 显示重叠或者弹出窗口

处理函数	类 型	描 述	参见注释
OnShowWindow()	消息处理函数	由ShowWindow()函数发出	第11条
OnWindowPosChanging()	消息处理函数	通知窗口的X,Y坐标甚至Z坐标将要改变	第9条
OnQueryNewPalette()	消息处理函数	允许改变系统调色板	第12条
OnWindowPosChanging()	消息处理函数	通知窗口的X,Y坐标甚至Z坐标将要改变	第9条
OnNcPaint()	消息处理函数	允许自行绘制窗口的非客户区	第13条
OnEraseBkgnd()	消息处理函数	允许选取绘制窗口背景的颜色	第14条

(续)

处理函数	类 型	描 述	参见注释
OnWindowPosChanged()	消息处理函数	通知隐藏窗口已经显示，将要发送WM_SIZE和WM_MOVE消息	第10条
OnNcActivate()	消息处理函数	通知使用当前颜色绘制非客户区	第15条
OnActivate()	消息处理函数	通知将被成为当前活动的窗口	第16条
OnSetFocus()	消息处理函数	通知将要接收键盘输入的窗口	第17条
OnNcPaint()	消息处理函数	允许绘制窗口的非客户区	第13条
OnWindowPosChanged()	消息处理函数	通知隐藏窗口已经显示，将要发送WM_SIZE和WM_MOVE消息	第10条
OnSize()	消息处理函数	通知窗口的大小正在改变，仅能由重叠窗口发出	第7条
OnMove()	消息处理函数	通知窗口正在移动。仅能由重叠窗口发出	第7条

注释

12) WM_QUERYNEWPALETTE消息由系统发出，以通知某个窗口它将成为活动窗口并可以拥有系统调色板。对没有足够显示内存以显示真彩色的系统，将由系统调色板决定获得输入焦点的窗口在屏幕上显示什么颜色，有关该主题可以参阅第 1章。

13) WM_NCPAINT消息由系统发出，以允许绘制窗口的非客户区。

14) WM_ERASEBKGND消息由系统发出，允许设置窗口的背景色。

15) WM_NCACTIVATE 消息由系统发出，以通知窗口用当前颜色绘制其非客户区。

16) WM_ACTIVATE消息发送到窗口，表明它将成为活动窗口并允许其在客户区进行任何颜色改变操作。

17) WM_SETFOCUS消息发送到窗口，以通知它现在具有输入焦点，这就意味着键盘输入将被发送到该窗口。

A.1.5 关闭窗口

关闭任何类型窗口的时发生下列调用序列：

表A-5 关闭窗口

处理函数	类 型	描 述	参见注释
OnClose()	消息处理函数	单击关闭按钮时发送	第18条
OnDestroy()	消息处理函数	窗口将要被销毁时发送	第19条
OnNcDestroy()	消息处理函数	窗口被销毁后发送	第20条
PostNcDestroy()	重载函数	CWnd处理OnNcDetroy()的最后调用函数	第21条

注意

18) 按下窗口非客户区的关闭按钮时将发送 WM_CLOSE消息，如果窗口没有关闭按钮，可以自己发送该消息。这时可以询问用户是否真的要关闭窗口，如果回答不，则不要调用向窗口发送WM_DESTROY消息的基类成员函数。

19) WM_DESTROY消息由任何希望销毁窗口的事件发出。这时可以发送自己窗口的销毁消息，但要记住系统将自动销毁子窗口。

20) WM_NCDESTROY消息在窗口已经销毁后由系统发出。它执行 MFC的清除工作。

21) 该CWnd成员函数由CWnd的WM_NCDESTROY消息的缺省实现所调用。这是销毁所有与该窗口相关的所有东西的最后机会。如果控制该窗口的 CWnd对象不再有用但并不希望销毁它，则可以在此使用下列语句来析构该对象：

```
delete this;
```

A.2 对话框

本节将列出创建和关闭对话框时调用的消息处理函数和函数。对话框可以通过两种方式创建：有模式和无模式。

- 有模式对话框直到其关闭才放弃控制。它有自己的消息循环，这就允许它执行其子窗口的特定处理(例如：编辑框之间的跳格(tabbing))。
- 无模式对话框使用应用程序的消息循环，这就允许任何窗口接收输入焦点，但失去了有模式消息循环功能。

为了给对话框增加自己的功能，必须首先从 CDialog类派生新类并使用ClassWizard来包含以下的重载函数和消息处理函数。

A.2.1 创建有模式对话框

以下调用序列在创建有模式对话框的时候发生：

```
CMyDialog dlg;
dlg.DoModal();
```

表A-6 打开有模式对话框

处理函数	类 型	描 述	参见注释
DoModal()	重载函数	重载DoModal()成员函数	
PreSubclassWindow()	重载函数	允许首先子类化该窗口	第2条
OnCreate()	消息处理函数	通知窗口已经被创建	第2、27条
OnSize()	消息处理函数	通知窗口的大小正在改变	第7条
OnMove()	消息处理函数	通知窗口正在移动	第7条
OnSetFont()	消息处理函数	允许改变对话框内控件所用的字体	第22条
OnInitDialog()	消息处理函数	允许初始化对话框内的控件或者创建新的控件	第23条
OnMove()	消息处理函数	通知窗口正在移动	第7条
OnShowWindow()	消息处理函数	由ShowWindow()函数发出	第11条
OnCtlColor()	消息处理函数	由父窗口发出，允许改变对话框颜色	第24条
OnChildNotify()	重载函数	作为WM_CTLCOLOR消息的结果发出	第8、25条

注释

22) WM_SETFONT消息由系统在创建对话框的时候发出，允许重载用于对话框资源文件中指定的字体。字体用于对话框为其控件绘制文本。

23) WM_INITDIALOG消息由系统发出，以允许初始化或者创建任何额外的控件窗口。

24) WM_CTLCOLOR消息由系统发出，以允许确定用什么颜色绘制对话框。

25) 该成员函数由处理WM_CTLCOLOR消息的缺省消息处理函数调用。以便于将全部子窗口通知消息处理过程放进OnChildNotify内。

A.2.2 关闭模式对话框

以下关闭模式对话框的调用序列与关闭窗口过程类似：

表A-7 关闭模式对话框

处理函数	类 型	描 述	参见注释
OnClose	消息处理函数	单击关闭按钮时发送	第18条
OnKillFocus()	消息处理函数	窗口失去键盘输入焦点前发送	第26条
OnDestroy()	消息处理函数	窗口将要销毁时发送	第19条
OnNcDestroy()	消息处理函数	窗口销毁后发送	第20条
PostNcDestroy()	消息处理函数	由CWnd调用的处理OnNcDestroy()的最后过程	第21条

注释

26) 通知窗口它将不再从键盘接收击键操作。

A.2.3 创建无模式对话框

创建无模式对话框时发生以下调用序列：

```
CMyDialog dlg;
dlg.Create(…);
```

表A-8 打开无模式对话框

处理函数	类 型	描 述	参见注释
PreSubclassWindow()	重载函数	允许首先子类化该窗口	第2条
OnCreate()	消息处理函数	通知已经创建窗口	第6、27条
OnSize()	消息处理函数	通知窗口大小正在改变	第7条
OnMove()	消息处理函数	通知窗口正在移动	第7条
OnSetFont()	消息处理函数	允许改变用于对话框控件的字体	第22条

正如以上提到的，无模式对话框并不使用特定的 DoModalLoop()消息循环，这意味着在有模式对话框中提供的某些方便功能在这里不得不手工实现。当然，无模式对话框的优点是应用程序的其他部分并不“冻结”，其他窗口在无模式对话框启动时仍然可用。

注释

27) WM_CREATE消息在对话框创建之后，但在任何控件窗口创建之前发送，因为无模式对话框并不发送WM_INITDIALOG消息，所以在调用InitDialog()函数之前需要等待Create()函数返回。

A.3 对话框应用程序

本节列出创建和关闭对话框应用程序时调用的消息处理函数和函数。回顾第 1章的内容，对话框应用程序实际上就是简单地将一个有模式对话框作为自己的界面，对话框应用程序没有文档或者视。

A.3.1 创建对话框应用程序

除了创建应用程序所需要的特定步骤以外，创建对话框应用程序的序列和以上创建有模式对话框的调用序列是一样的。

表A-9 创建对话框应用程序

处理函数	类 型	描 述	参见注释
InitInstance()	重载函数	强制性的 —— 初始化并确定应用程序的类型	第28条
参见表A-6 打开有模式对话框			第29条
DoModalLoop()	重载函数	执行对话框内控件窗口的特定处理过程	

注释

28) 该成员函数由CWinApp调用以启动应用程序。实际上，对话框应用程序在该函数的派生函数上花费了几乎所有的时间。该函数是初始化应用程序 (例如：打开数据库，注册窗口类) 的最佳场合。然后为创建对话框应用程序。 AppWizard将加进下列3行：

```
CXxxDlg   dlg;
m_pMainWnd = &d
int nResponse = dlg.DoModal();
```

这三行代码负责创建有模式对话框 (代码中的“ Xxx ”是工程名字)。

29) 在此时调用序列与创建有模式对话框完全一样，除了两点不同：

- AppWizard自动为WM_INITDIALOG增加一个消息处理函数，它在系统菜单中添加了About...菜单项。当单击重叠窗口左上角的图标时就会出现系统菜单。 OnInitDialog此时也设置该图标。 MDI和MDI应用程序在其装载框架窗口 (下面将讨论)的时候自动地设置该图标。
- AppWizard还在其中增加某些功能，以在其缩小时在任务栏上绘制应用程序的图标。在SDI或 MDI应用程序中，这些工作对用户来说是不可见的。

A.3.2 关闭对话框应用程序

关闭对话框应用程序的调用序列与关闭窗口或者关闭有模式对话框是一样的，只是增加了关闭应用程序自身的一步：

表A-10 关闭对话框应用程序

处理函数	类 型	描 述	参见注释
OnClose()	消息处理函数	单击关闭按钮时发送	第18、30条
OnDestroy()	消息处理函数	窗口将要销毁时发送	第19条
OnNcDestroy()	消息处理函数	窗口销毁后发送	第20条
PostNcDestroy()	重载函数	由CWnd调用作为OnNcDestroy()的最后处理过程	第21条
ExitInstance()	重载函数	调用来终止应用程序	第31条

注释

30) 这是唯一可以询问用户它们是否确定终止应用程序的位置。如果不终止应用程序，将不会调用发送WM_DESTROY消息给对话框的OnClose()基类函数。

31) 由CWinApp调用以终止应用程序，此处适于关闭数据库和释放应用程序资源。

A.4 SDI应用程序

本部分将列出创建和关闭SDI应用程序时所调用的消息处理函数和函数，同时查看SDI应用程序在以下情况所使用的调用序列：

- 创建新文档。
- 打开已有文档。
- 保存文档。

A.4.1 创建SDI应用程序

创建SDI应用程序时发生下列调用序列。

表A-11 打开SDI应用程序

处理函数	类 型	描 述	参见注释
InitInstance()	重载函数	强制性的 —— 初始化并决定应用程序的类型	第32条
创建了新文档或者加载已有文档 —— 参见表A-12和A-13			
Run()	重载函数	CWinApp的后台处理函数	第33条
OnIdle()	重载函数	允许进行自己的后台处理	第34条

注释

32) 由CWinApp调用的成员函数，它启动应用程序并允许定制自己应用程序的初始化 (如：打开数据库，注册窗口类)。当AppWizard创建工程时它重载InitInstance()函数并填写一些初始化代码。对SDI应用程序还增加了下列3个步骤：

- 创建文档模板。
- 解析并处理命令行。
- 显示主窗口。

文档模板由下列代码创建：

```
CSingleDocTemplate* pDocTemplate;
pDocTemplate = new CSingleDocTemplate(
    IDR_MAINFRAME,
    RUNTIME_CLASS( CExamplesDoc ),
    RUNTIME_CLASS( CMainFrame ),        // main SDI frame window
    RUNTIME_CLASS( CExamplesView )
    );
AddDocTemplate( pDocTemplate );
```

文档模板将框架、文档和视类关联起来，用于装载特定的文档。每个应用程序可能有一个或者一个以上的文档模板允许处理多种类型的文档，但这种情况并不多见，而且是为大型MDI应用程序所保留的功能，如Developer Studio。

具有多个文档模板的SDI应用程序

如果为SDI应用程序添加其他的文档模板：

- 只要用户创建或者打开一个文档， MFC就会用一个对话框提示并询问它们使用哪个模板。MFC使用在资源文件中对应资源标识符下找到的模板名来填充该对话框，此时的资源标识符是IDR_MAINFRAME。

- 如果用户创建或者装载了已经被装载的文档类型， MFC并不创建其他框架、文档和视类对象，而是使用已有的对象。但是，如果打开一个新文档类型就会创建第二组对象。

下列代码负责解析和处理命令行：

```
CCommandLineInfo cmdInfo;
ParseCommandLine( cmdInfo );
if ( !ProcessShellCommand( cmdInfo ) )
   return FALSE;
```

命令行在ParseCommandLine()中进行解析，并在ProcessShellCommand()中接受处理。一个空命令行将使 ProcessSellCommand()创建新文档。命令行上的文件名导致 ProcessShell-Command ()试图将其作为文档打开。

ProcessShellCommand()函数通过调用CWinApp的OnCmdMsg()成员函数来创建并打开一个文档。其中用到两个预定义的命令： ID_FILE_NEW，用于创建一个新文档，以及 ID_FILE_OPEN，用于打开一个已有文档。

InitInstance()的最后一行代码显示并绘制主窗口：

```
m_pMainWnd->ShowWindow(SW_SHOW);
m_pMainWnd->UpdateWindow();
```

m_pMainWnd包含了指向主窗口的指针，该主窗口由 ProcessShellCommand()创建。

33) 这个CWinApp成员函数在应用程序运行后调用。该函数花费其大量时间在空闲循环(idle loop)中进行后台处理工作，并查看应用程序消息队列中寄送来的消息—特别是，WM_CLOSE消息将导致该循环和应用程序的终止。

34) 该CWinApp成员函数允许重载 Run()函数的空闲循环，以便在应用程序中执行后台处理过程。

A.4.2 创建新的SDI文档

下列调用序列创建一个新 SDI文档。如果文档、框架和视类都还不存在，例如在程序初始化的时候，则该文档仍然被创建。 SDI文档按照下列步骤创建：

- 如果没有文档类对象，则创建一个。
- 如果没有框架类对象并且窗口退出，则创建一个。
- 如果框架类对象已经创建了，则创建视类对象和窗口。
- 初始化文档。
- 激活框架和视。

表A-12 创建一个新SDI文档

处理函数	类 型	描 述	参见注释
CWinApp::OnFileNew()	消息处理函数	单击主窗口的New命令时发送	第35条
CSingleDocTemplate::OpenDocumentFile()	重载函数	由缺省的OnFileNew()调用。如果文件名参数是NULL则创建一个空的文档。如果需要，则调用下列函数创建文档、框架和视类对象	第36条

如果不存在文档类对象则创建它...

(续)

处理函数	类 型	描 述	参见注释
CDocTemplate:: CreateNewDocument()	重载函数	用文档模板的CRuntimeClass和CreateObject()函数创建一个新文档类对象	第37条
如果不存在框架类对象则创建它...			
CDocTemplate:: CreateNewFrame()	重载函数	用文档模板的CRuntimeClass和CreateObject()函数创建一个新框架类对象	
CMainFrame:: LoadFrame()	重载函数	导致框架窗口被创建并装载资源	
CFrameWnd::Create()	重载函数	创建框架窗口	
CMainFrame:: PreCreateWindow()	重载函数	允许框架窗口在从模板创建之前重载其风格和类	第1、38条
CMainFrame:: PreCreateWindow()	重载函数	不是打印错误 —— 第2次调用，参看注释	第1、38条
CMainFrame:: PreSubclassWindow()	重载函数	允许框架窗口预子类化	第2条
CMainFrame:: OnGetMinMaxInfo()	消息处理函数	允许框架窗口有其最大或者最小尺寸	第3条
CMainFrame:: OnCreate()	消息处理函数	通知框架窗口已经创建	第6、39条
CMainFrame:: OnCreateClient()	重载函数	由LoadFrame()调用来创建视类对象和窗口，OnCreateClient()调用接下来的函数	
如果正在创建新的框架，则创建新的视类对象和视窗口...			
CFrameWnd:: CreateView()	重载函数	用文档模板的CRuntimeClass和CreateObject()函数创建一个新视类对象	
CView::Create()	重载函数	创建视窗口	
CView:: PreCreateWindow()	重载函数	允许视窗口在从模板创建之前重载其风格、类	第1条
CView:: PreSubclassWindow()	重载函数	允许视窗口预子类化	第2条
CView::OnCreate()	消息处理函数	通知视窗口已经创建	第6、40条
CDocument:: OnChangedViewList()	重载函数	被调用以告诉文档类已经为其添加了一个视	
CView::OnSize()	消息处理函数	重置视窗口大小	第7条
CView::OnMove	消息处理函数	移动视窗口	第7条
CView:: OnChildNotify()	重载函数	通知视窗口已经创建	第8条
CView::OnShowWindow()	消息处理函数	显示视窗口	第11条
如果框架和视已经创建，设置它们各自的大小...			
CMainFrame::OnMove()	消息处理函数	沿视窗口移动框架窗口	第7条
CMainFrame::OnSize()	消息处理函数	沿视窗口改变框架窗口大小	第7条
CMainFrame:: RecalcLayout()	重载函数	重新定位框架窗口内的视和控制条	
CView:: CalcWindowRect()	重载函数	在设计的客户区大小基础之上确定整个窗口的大小	
CView::OnMove()	消息处理函数	在框架窗口之内移动视	第7条

(续)

处理函数	类 型	描 述	参见注释
CView::OnSize()	消息处理函数	在框架窗口之内改变视的大小	第7条
CMainFrame::RecalcLayout()	重载函数	再一次……	
CView::CalcWindowRect()	重载函数	……同样的	
完成文档创建…			
CSingleDocTemplate::SetDefaultTitle()	重载函数	设置文档的缺省标题	
CDocument::SetTitle()	重载函数	设置实际的标题	
CDocument::OnNewDocument()	重载函数	初始化，通常留给下面的函数来完成	第41条
CDocument::DeleteContents()	重载函数	初始化CDocumnet的成员变量。 用于已经装载了文档的时候	第41条
第1次更新视…			
CDocTemplate::InitialUpdateFrame()	重载函数	通知视第1次初始化	
CMainFrame::InitialUpdateFrame()	重载函数	通知视第1次初始化	
CView::OnInitialUpdate()	重载函数	通知视第1次初始化	第42条
CView::OnUpdate()	重载函数	只要文档变化就调用以更新视	第43条
激活框架窗口…			
CMainFrame::ActivateFrame()	重载函数	激活框架	
CMainFrame::OnQueryNewPalette()	消息处理函数	允许用自己的颜色装载系统调色板	第12条
CMainFrame::OnActivateApp()	消息处理函数	通知应用程序正被激活	
CView::OnActivateView()	消息处理函数	通知视正被激活	
CMainFrame::OnActivate()	消息处理函数	通知框架正被激活	
CMainFrame::OnShowWindow()	消息处理函数	显示框架	第11条
CMainFrame::OnEraseBkgnd()	消息处理函数	允许改变框架背景颜色	
CMainFrame::OnPaint()	消息处理函数	绘制框架窗口的客户区	
CView::OnPaint()	消息处理函数	绘制视，但使用OnDraw()函数	
CView::OnDraw()	重载函数	被CView的OnPaint()函数调用	第44条

35) ID_FILE_NEW命令消息表明用户希望装载空文档。该消息或者在应用程序初始化的时候由ProcessShellCommand()函数发送，或者在用户单击了 File菜单下的New命令后发送。ID_FILE_NEW消息是几个预定义MFC消息中的一个，可以自己处理该消息以创建自定义文档，

或者让MFC通过创建新MFC风格文档来处理。接下来的调用序列说明 MFC如何创建一个新文档。

36) OpenDocumentFile()函数是文档模板类CDocTemplate的成员函数。可以在应用程序的任何地方调用OpendocumentFile()来创建一个新文档，只需选取要创建的文档模板并调用以下代码即可：

```
pTemplate->OpenDocumentFile(NULL);
```

37) CreateNewDocument()函数是文档模板类 CDocTemplate的另一个成员函数。可以用CreateObject()函数从InitInstance()函数中的模板创建一个文档对象。但实际上不必用该函数创建文档对象。可以用以前的方法来完成：

```
Doc = new CDocument;
```

38) 第1次调用了PreCreateWindow()函数之后，MFC试图确定用什么窗口类来创建框架窗口。如果这里没有指定窗口类，MFC将自动地创建AfxFrameOrView作为模板，而AfxRegisterWndClass()则实际创建它并为其命名。注意框架窗口用 AfxWndProc作为其处理函数。第 2次调用该函数则是在实际窗口正在被创建之时。

39) 通常这是在CMainFrame中创建任何控制条(对话条、状态栏和工具栏等)的场合。

40) 此处适于创建视中的任何控件。

41) 可以按照某特定方式重载该函数以初始化文档的成员变量。但是，如果使用该函数的缺省实现并且重载了 DeleteContents()而不是初始化变量，就必须再次进行该项工作以打开已有文件。这两种情况下都要调用 DeleteContents()函数。

42) 此处适于初始化视中的任何控件。

43) 此处适于在文档发生变化时更新视中的任何控件或者图像。

44) 该函数通常由OnPaint()调用。应当重载该函数而不是 OnPaint()函数，这是因为如果这样做，则将屏幕打印到打印机的功能是内置的。但只要使用了设备环境绘制到屏幕，则视内的任何控件不能自动被打印。

A.4.3 打开已有的SDI文档

下列序列从文件中装载已有的 SDI文档，如果文档、框架和视类对象还不存在，例如在程序初始化阶段，它们仍然被创建，但是因为以上已经回顾了这种情况，下面仅讨论这些对象已经存在的情况：

表A-13 打开已有SDI文档

处理函数	类 型	描 述	参见注释
CWinApp::OnFileOpen()	消息处理函数	单击主菜单中的“ Open ”命令时发出	第45条
CSingleDocTemplate:: OpenDocumentFile()	重载函数	由缺省的OnFileOpen()函数调用。将指定文件装载到文档，如果需要，可随意创建文档、框架和视类对象	
CDocument:: SaveModified()	重载函数	保证当前文档不被修改 —— 如果被修改了，用户可以将其保存	
CDocument:: OnOpenDocument()	重载函数	调用下列函数来装载已有文档的缺省实现	
CDocument:: DeleteContents()	重载函数	初始化CDocument的成员变量	第41条

(续)

处理函数	类 型	描 述	参见注释
CDocument::Serialize()	重载函数	使用MFC的串行化特征来装载文件到成员变量	
CDocument::SetPathName()	重载函数	设置被装载文件的路径名	
CDocument::SetTitle()	重载函数	设置文档标题	
CWinApp::AddToRecentFileList()	重载函数	在File菜单中的最近打开文件的路径列表添加文件的路径名	
CView::OnInitialUpdate()	重载函数	通知视第1次初始化	第42条
CView::OnUpdate()	重载函数	只要文档发生改变即被调用并更新视	第43条

注释

45) OnFileOpen()处理另一个预定义命令消息：ID_FILE_OPEN。

A.4.4 保存SDI文档

以下调用序列将已经修改的文档保存到文件。

表A-14 保存SDI文档

处理函数	类 型	描 述	参见注释
CDocument::OnFileSave()	消息处理函数	单击主菜单中的Save命令后发出	第46条
CDocument::OnSaveDocument()	重载函数	调用以下函数保存文档到文件的缺省实现	第47条
CDocument::Serialize()	重载函数	使用MFC的串行化特征将成员变量串行化到文件	第48条
CDocument::SetPathName()	重载函数	设置保存文件的路径名	第49条
CDocument::SetTitle()	重载函数	设置保存文档的文档名	第49条
CWinApp::AddToRecentFileList()	重载函数	在File菜单中的最近打开文件的路径列表添加文件的路径名	第49条

注释

46) OnFileSave()处理另一个预定义的命令消息：ID_FILE_SAVE。这样做的另一个方式是由CDocument::OnFileSaveAs()函数处理ID_FILE_SAVE_AS。

47) 该函数打开传递给它的文件名，然后打开一个具有该文件名的档案文件并调用Serialize()将文件存盘。

48) Serialize()允许自动地将文档存入磁盘。如果要将其存入系统注册表或者数据库，可以重载OnFileSave()函数并在其中保存数据。

49) 这些函数只有在使用Save As命令期间路径名发生改变的情况下才被调用。

A.4.5 关闭SDI应用程序

以下调用序列关闭SDI应用程序。该序列假设文档没有进行修改。参见表 A-14了解用于保

存被修改过文件时的调用序列。

表A-15 关闭SDI文档

处理函数	类 型	描 述	参见注释
CMainFrame::OnClose()	消息处理函数	单击了关闭按钮或者File菜单下的Exit命令发送该消息	第50条
CDocument::CanCloseFrame()	重载函数	由OnClose调用，它是调用下面的函数以查看文档是否被修改以及用户是否打算保存的缺省实现	
CDocument::SaveModified()	重载函数	询问用户保存已修改的文档	
CMainFrame::OnShowWindow()	消息处理函数	隐藏主框架窗口	第11条
CView::OnActivateView()	消息处理函数	使视当前不被激活	
CMainFrame::OnActivate()	消息处理函数	使框架当前不被激活	
CMainFrame::OnActivateApp()	消息处理函数	使应用程序当前不被激活	
CView::OnKillFocus()	消息处理函数	从应用程序中去除键盘输入焦点	第26条
CDocument::OnCloseDocument()	重载函数	关闭文档，缺省实现调用下列函数	
CDocument::DeleteContents()	重载函数	释放内存以初始化文档类	第41条
CMainFrame::DestroyWindow()	重载函数	销毁主框架窗口	
CMainFrame::OnDestroy()	消息处理函数	发送该消息以销毁主框架窗口	第19条
CView::OnDestroy()	消息处理函数	发送该消息以销毁视窗口	第19条
CView::OnActivateView	消息处理函数	使视当前不被激活	
CView::PostNcDestroy()	重载函数	在视窗口销毁后调用	第21条
CDocument::OnChangedViewList()	重载函数	通知文档它已经失去视	
CMainFrame::PostNcDestroy()	重载函数	在框架窗口销毁之后调用	第21条
CWinApp::ExitInstance()	重载函数	允许释放分配给应用程序的所有资源	第30条

注释

50) 此处是工具栏Are you sure?提示消息的地方。如果回答是no，则不调用其基类函数发送WM_DESTROY消息给主框架窗口。否则将等待框架窗口使用 SaveModified()函数检查文档是否已经被修改。

A.5 MDI应用程序

本节将列出创建和关闭 MDI应用程序的消息和所调用的函数。除了 MDI应用程序拥有一

个由一个或一个以上子框架窗口所填充的主框架窗口以外，　MDI应用程序与SDI应用程序类似。

另外，还将看到 MDI应用程序在以下情况所使用的调用序列：

- 创建新文档。
- 打开已有文档。
- 保存MDI文档。
- 关闭MDI子框架。

表A-16　打开MDI应用程序

处理函数	类　型	描　　述	参见注释
InitInstance()	重载函数	强制性的—初始化并决定应用程序的类型	
	主窗口创建 —— 参见表A-17		
	创建了新文档或者装载了已有文档 —— 参见表A-18和A-19		
Run()	重载函数	CWinApp的后台处理函数	
OnIdle()	重载函数	允许进行自己的后台处理	

51) 由CWinApp调用的成员函数，它启动应用程序并允许定制自己应用程序的初始化 (如：打开数据库、注册窗口类)。当AppWizard创建工程时它重载InitInstance()函数并填写一些初始化代码。对MDI应用程序还增加了下列4个步骤：

- 创建文档模板。
- 创建主框架窗口。
- 解析并处理命令行。
- 显示主窗口。

文档模板由下列代码创建：

```
CMultiDocTemplate* pDocTemplate;
pDocTemplate = new CMultiDocTemplate(
   IDR_EXAMPLTYPE,
   RUNTIME_CLASS( CExamplemdiDoc ),
   RUNTIME_CLASS( CChildFrame ),        // custom MDI child frame
   RUNTIME_CLASS( CExamplemdiView )
   );
AddDocTemplate( pDocTemplate );
```

文档模板将框架、文档和视类关联起来用于装载特定的文档。每个应用程序可能有一个或者一个以上的文档模板允许处理多种类型的文档，但这种情况并不多见，而且是为大型MDI应用程序所保留的功能，如Developer Studio。

注意这里使用的框架窗口是子框架窗口。现在只要装载一个新文档或者已有文档，就会创建一个新的子框架、视和文档类对象并驻留在主框架窗口。那么，主框架窗口是什么时候创建的呢？

主框架窗口由下列几行代码创建：

```
// create main MDI Frame window
CMainFrame* pMainFrame = new CMainFrame;
if ( !pMainFrame -> LoadFrame( IDR_MAINFRAME ) )
```

```
    return FALSE;
m_pMainWnd = pMainFrame;
```

LoadFrame()函数和SDI应用程序一样，在此再次使用并导致下列函数调用消息序列：

表A-17 创建主框架

处理函数	类 型	描 述	参见注释
CMainFrame::LoadFrame()	重载函数	调用以下函数以装载框架	
CMainFrame::PreCreateWindow()	重载函数	为框架窗口指定窗口类的时候调用	第1、38条
CMainFrame::PreCreateWindow()	重载函数	改变框架窗口的窗口风格或者大小的时候调用	第1、38条
CMainFrame::PreSubclassWindow()	重载函数	允许预子类化窗口	第2条
CMainFrame::OnGetMinMaxInfo()	消息处理函数	允许限制主框架窗口的大小	第3条
CMainFrame::OnCreate()	消息处理函数	主框架窗口被创建了——可以创建工具栏、对话条和状态栏	第6条
CMainFrame::OnCreateClient()	重载函数	使用“MDICLIENT”窗口类的窗口填充MDI应用程序的客户区	
CMainFrame::RecalcLayout()	重载函数	在框架窗口内重新定位视和控制条的时候调用	

在主框架窗口创建后，下列代码负责解析和处理命令行：

```
CCommandLineInfo cmdInfo;
ParseCommandLine( cmdInfo );
if ( !ProcessShellCommand( cmdInfo ) )
    return FALSE;
```

命令行在ParseCommandLine()中进行解析，并在ProcessShellCommand()中接受处理。一个空命令行将使ProcessSellCommand()创建新文档。命令行上的文件名导致ProcessShell-Command ()试图将其作为文档打开。

ProcessShellCommand()函数通过调用CWinApp的OnCmdMsg()成员函数来创建并打开一个文档。其中用到两个预定义的MFC命令：ID_FILE_NEW，用于创建一个新文档，以及ID_FILE_OPEN，用于打开一个已有文档。这与SDI应用程序一样。

InitInstance()的最后一行代码显示并绘制主窗口：

```
m_pMainWnd -> ShowWindow( SW_SHOW );
m_pMainWnd -> UpdateWindow();
```

m_pMainwnd包含了指向以上所创建的主窗口的指针。

A.5.1 创建新的MDI文档

下列调用序列创建一个新MDI文档。文档、子框架和视类对象现在都已经创建了。除此以外，该调用序列几乎同以上创建新的SDI文档完全一样。

表A-18　创建一个新MDI文档

处理函数	类　型	描　　述	参见注释
CWinApp::OnFileNew()	消息处理函数	单击主窗口的New命令时发送	第35条
CMultiDocTemplate::OpenDocumentFile()	重载函数	由OnFileNew()的缺省实现调用。如果文件名参数是NULL则创建一个空的文档。调用下列函数创建文档、子框架和视类对象	第36条
创建文档类对象...			
CDocTemplate::CreateNewDocument()	重载函数	用文档模板的CRuntimeClass和CreateObject()函数创建一个新文档类对象	第37条
创建子框架类对象和子框架窗口...			
CDocTemplate::CreateNewFrame()	重载函数	创建一个新框架类对象	
CChildFrame::LoadFrame()	重载函数	导致框架窗口被创建并装载资源	
CChildFrame::PreCreateWindow()	重载函数	在框架窗口类从模板中创建之前允许重载风格和类	第1、38条
CChildFrame::Create()	重载函数	创建框架窗口	
CMainFrame::RecalcLayout()	重载函数	在框架窗口内重新定位视和控制条	
CChildFrame::PreCreateWindow()	重载函数	不是打印错误 —— 第2次调用该函数，参看注释	第1、38条
CChildFrame::PreSubclassWindow()	重载函数	允许框架窗口预子类化	第2条
CChildFrame::OnGetMinMaxInfo()	消息处理函数	允许框架窗口有其最大或者最小尺寸	第3条
CChildFrame::PreCreateWindow()	重载函数	允许框架窗口在从模板创建之前重载其风格、类	第1、38条
CChildFrame::OnCreate()	消息处理函数	通知框架窗口已经创建	第6、39条
CChildFrame::OnCreateClient()	重载函数	由LoadFrame()调用来创建视类对象和窗口。LoadFrame()调用接下来的函数	
创建新的视类对象和视窗口...			
CFrameWnd::CreateView()	重载函数	用文档模板的CRuntimeClass和CreateObject()函数创建一个新视类对象	
CView::Create()	重载函数	创建视窗口	
CView::PreCreateWindow()	重载函数	允许视窗口在从模板创建之前重载其风格、类	第1条
CView::PreSubclassWindow()	重载函数	允许视窗口预子类化	第2条
CView::OnCreate()	消息处理函数	通知视窗口已经创建	第6、40条
CDocument::OnChangedViewList()	重载函数	被调用以通知文档类已经为其添加了一个视	

(续)

处理函数	类 型	描 述	参见注释
CView::OnSize()	消息处理函数	重置视窗口大小	第7条
CView::OnMove	消息处理函数	移动视窗口	第7条
CView::OnChildNotify()	重载函数	通知视窗口已经创建	第8条
CView::OnShowWindow()	消息处理函数	显示视窗口	第11条
如果子框架和视已经创建，则设置各自的大小...			
CMainFrame::OnMove()	消息处理函数	沿视窗口移动框架窗口	第7条
CMainFrame::OnSize()	消息处理函数	沿视窗口改变框架窗口大小	第7条
CMainFrame::RecalcLayout()	重载函数	重新定位框架窗口内的视和控制条	
CView::CalcWindowRect()	重载函数	在预期的客户区大小基础上确定整个窗口的大小	
CView::OnMove()	消息处理函数	在框架窗口之内移动视	第7条
CView::OnSize()	消息处理函数	在框架窗口之内改变视的大小	第7条
CMainFrame::RecalcLayout()	重载函数	再一次……	
CView::CalcWindowRect()	重载函数	……同样的	
CChildFrame::OnMDIActivate()	重载函数	激活子框架	
完成文档创建...			
CDocument::SetTitle()	重载函数	设置实际标题	
CDocument::OnNewDocument()	重载函数	初始化，通常留给下面的函数来完成	第41条
CDocument::DeleteContents()	重载函数	初始化CDocumnet的成员变量。用于已经装载了文档的时候	第41条
第1次更新视...			
CDocTemplate::InitialUpdateFrame()	重载函数	通知视第1次初始化	
CMainFrame::InitialUpdateFrame()	重载函数	通知视第1次初始化	
CView::OnInitialUpdate()	重载函数	通知视第1次初始化	第42条
CView::OnUpdate()	重载函数	只要文档发生变化便调用以更新视	第43条
激活主框架窗口和子框架...			
CChildFrame::ActivateFrame()	重载函数	激活主框架窗口	
CChildFrame::OnMDIActivate()	重载函数	激活子框架窗口	
CChildFrame::OnShowWindow()	消息处理函数	显示框架	

(续)

处理函数	类　型	描　　述	参见注释
CView::OnActivateView()	消息处理函数	激活视	
CMainFrame::OnShowWindow()	消息处理函数	显示主框架窗口	第11条
CMainFrame::OnQueryNewPalette()	消息处理函数	允许主框架窗口用它自己的颜色填充系统调色板	第12条
CMainFrame::OnEraseBkgnd()	消息处理函数	允许主框架窗口改变绘制背景的颜色	
CMainFrame::OnActivateApp()	消息处理函数	激活应用程序	
CView::OnActivateView()	消息处理函数	激活视	
CMainFrame::OnActivate()	消息处理函数	激活框架窗口	
CView::OnPaint()	消息处理函数	绘制视，但使用的是OnDraw()函数	
CView::OnDraw()	重载函数	被CView的OnPaint()函数调用	

A.5.2 打开已有的MDI文档

下列序列从文件中装载已有的MDI文档，这也是与SDI文档很相似的，只是文档、框架和视类对象都已经创建了。

表A-19　打开已有MDI文档

处理函数	类　型	描　　述	参见注释
CWinApp::OnFileOpen()	消息处理函数	单击主菜单中的Open命令时发出	第35条
CMultiDocTemplate::OpenDocumentFile()	重载函数	由缺省的OnFileOpen()函数调用。将指定文件装载到文档,调用以下函数以创建文档、子框架和视类对象	第36条
创建文档类对象...			
CDocTemplate::CreateNewDocument()	重载函数	创建一个新的文档类对象	第37条
创建子框架类对象和窗口...			
CChildFrame::LoadFrame()	重载函数	调用下列函数来装载子框架	
CChildFrame::PreCreateWindow()	重载函数	用自己的窗口类来创建子框架窗口	第1、38条
CMainFrame::RecalcLayout()	重载函数	重新定位框架窗口内的视和控制条	
CChildFrame::PreCreateWindow()	重载函数	改变子框架窗口的其他属性	第1、38条
CChildFrame::PreSubclassWindow()	重载函数	允许预子类化窗口	第2条
CChildFrame::OnGetMinMaxInfo()	重载函数	允许设置子框架窗口大小限制	第3条
CChildFrame::OnCreate()	消息处理函数	子框架窗口已经创建，此时可以为该子框架创建工具栏、对话条、状态栏	第6条

(续)

处理函数	类 型	描 述	参见注释
CChildFrame::OnCreateClient()	重载函数	用下面的函数为该子框架创建视	
创建视类对象和窗口...			
CView::PreCreateWindow()	重载函数	允许改变所创建视类窗口的属性	第1条
CView::PreSubclassWindow()	重载函数	允许预子类化视窗口	第2条
CView::OnCreate()	重载函数	视窗口已经创建，为该视窗口创建控件	第6条
CDocument::OnChangedViewList()	重载函数	通知文档已经添加视	
CView::OnSize()	消息处理函数	设置视的大小，此时适于可以设置该视的任何子窗口的大小	第7条
CView::OnMove()	消息处理函数	移动视窗口	第7条
CView::OnChildNotify()	重载函数	通知视已经创建了窗口	第8条
CView::OnShowWindow()	消息处理函数	显示视窗口	第11条
为子框架设置视...			
CChildFrame::OnMove()	消息处理函数	移动子框架窗口	第7条
CChildFrame::OnSize()	消息处理函数	设置子框架窗口大小	第7条
CChildFrame::RecalcLayout()	重载函数	重新定位框架窗口内的视和控制条	
CView::CalcWindowRect()	重载函数	在所预期的客户区大小基础之上确定整个窗口的大小	
完成打开文档...			
CDocument::OnOpenDocument()	重载函数	调用下列函数来装载已有文档的缺省实现	
CDocument::DeleteContents()	重载函数	初始化CDocument的成员变量	第41条
CDocument::Serialize()	重载函数	使用MFC的串行化特征装载文件到成员变量	
CDocument::SetPathName()	重载函数	设置被装载文件的路径名	
CDocument::SetTitle()	重载函数	设置文档标题	
CWinApp::AddToRecentFileList()	重载函数	在File菜单中的最近打开文件的路径列表添加文件的路径名	
CView::OnInitialUpdate()	重载函数	通知视第1次初始化	第42条
CView::OnUpdate()	重载函数	只要文档发生改变即被调用，以更新视	第43条
激活子框架和视...			

(续)

处理函数	类　型	描　　述	参见注释
CChildFrame::ActivateFrame()	重载函数	用于激活主框架窗口	
CChildFrame::OnMDIActivate()	重载函数	用于激活子框架窗口	
CView::OnActivateView()	消息处理函数	发送该消息用于激活视	
CView::OnKillFocus()	消息处理函数	从视中去除键盘焦点...	第26条
CChildFrame::OnSetFocus()	消息处理函数	...添加到子框架窗口	
CChildFrame::OnShowWindow()	消息处理函数	显示子框架窗口	第11条
CView::OnActivateView()	消息处理函数	发送该消息用于激活视	

A.5.3　保存MDI文档

以下序列将已经修改的文档保存到文件。

表A-20　保存MDI文档

处理函数	类　型	描　　述	参见注释
CDocument::OnFileSave()	消息处理函数	单击主菜单中的Save命令后发出	第46条
CDocument::OnSaveDocument()	重载函数	调用以下函数保存文档到文件的缺省实现	第47条
CDocument::Serialize()	重载函数	用MFC的串行化特色将成员变量串行化到文件	第48条
CDocument::SetPathName()	重载函数	设置保存文件的路径名	第49条
CDocument::SetTitle()	重载函数	设置保存文档的文档名	第49条
CWinApp::AddToRecentFileList()	重载函数	在File菜单中的最近打开文件的路径列表增加文件的路径名	第49条

A.5.4　关闭MDI子框架

下列序列发生在关闭 MDI子框架的情况下。该序列假设该子框架并没有修改过的文档。参见表A-19的序列用于保存MDI文档。

表A-21　关闭MDI子框架

处理函数	类　型	描　　述	参见注释
CChildFrame::OnClose()	消息处理函数	单击了子框架窗口的关闭按钮之后发出	第50条
CDocument::CanCloseFrame()	重载函数	调用下面的函数以确定是否可以关闭子框架	

(续)

处理函数	类 型	描 述	参见注释
CDocument::SaveModified()	重载函数	检查文档是否被修改过，如果是将保存它	第52条
CDocument::OnCloseDocument()	重载函数	调用以下函数的缺省实现	
CDocument::DeleteContents()	重载函数	释放文档内存	第41条
CChildFrame::DestroyWindow()	重载函数	销毁子框架窗口	
CChildFrame::OnShowWindow()	消息处理函数	隐藏子框架窗口	第11条
CChildFrame::OnMDIActivate()	消息处理函数	使子框架窗口当前不被激活	
CView::OnActivateView()	消息处理函数	使子框架内的视当前不被激活	
CView::OnKillFocus()	消息处理函数	从视中去除键盘焦点	第26条
CChildFrame::OnDestroy()	消息处理函数	用于销毁子框架窗口	第19条
CView::OnDestroy()	消息处理函数	用于销毁视窗口	第19条
CView::PostNcDestroy()	重载函数	在视窗口销毁后调用	第21条
CView::OnChangedViewList()	重载函数	告诉文档它的一个视已经删除了	
CChildFrame::PostNcDestroy()	重载函数	在子框架销毁后调用	第21条

52) 该函数的缺省实现将打开一个文件对话框，要求命名保存文件的路径。如果不保存文件，而是保存到数据库或者系统注册表，就必须重载该函数而不能使用缺省实现。

A.5.5 关闭MDI应用程序

以下序列发生于关闭 MDI应用程序的时候。该序列假设所有的 MDI文档都没有修改。参见表A-20了解子框架被关闭时的序列。

表A-22 关闭MDI文档

处理函数	类 型	描 述	参见注释
CMainFrame::OnClose()	消息处理函数	单击了关闭按钮或者File菜单下的Exit命令发送该消息	第50条
CWinApp::SaveAllModified()	重载函数	为每个打开的文档调用SaveModified()函数	
CMainFrame::OnShowWindow()	消息处理函数	隐藏主框架窗口	第11条
CMainFrame::OnActivate()	消息处理函数	使主框架当前不被激活	
CMainFrame::OnActivateApp()	消息处理函数	使应用程序当前不被激活	

(续)

处理函数	类 型	描 述	参见注释
CMainFrame:: DestroyWindow()	重载函数	主框架窗口被告之已经销毁	
CMainFrame:: OnDestroy()	消息处理函数	发送该消息以销毁主框架窗口	第19条
CMainFrame:: PostNcDestroy()	重载函数	在主框架窗口销毁后调用	第21条
CWinApp:: ExitInstance()	重载函数	允许释放分配给应用程序的所有资源	

A.6 定制

以上描述的各个序列是针对3种标准应用程序的：对话框、SDI和MDI，使用Serialize()函数进行标准的文档装载和保存。显然，还有许多种改变序列的情况。例如，使用数据库而不是使用Serialize()函数，在框架窗口内装载对话条——但要将各种情况全部列出来将覆盖全书的内容。在本附录中列出以上内容是希望使读者认识到 MFC实际所进行的工作，而且还希望读者意识到，不仅可以改变其缺省行为，而且还能创建自己的序列。特别是：

- 可以创建混合了对话框、SDI和MDI的应用程序。
- 可以用文档模板来创建想要的任何文档，不仅仅是单击 New或者Open的结果。

A.6.1 混合应用程序

打算创建什么样的应用程序经常会使人焦虑。但实际上可以干脆创建对话框、SDI和MDI的混合应用程序。例如，是否打算要应用程序在某些场合作为对话框应用程序，而在另一场合就成为MDI应用程序？为此，必须修改CWinApp的InitInstance()函数。

例如，如果有一个MDI应用程序，并希望它偶尔能够以对话框应用程序的面目出现，来完成一些自动创建处理工作，那么可以如下修改 InitInstance()函数：

如果命令行标记设置为 -D，运行以下代码将创建对话框应用程序。

```
CExampledlgDlg dlg;
m_pMainWnd = &dlg;
int nResponse = dlg.DoModal();
return FALSE;:
```

否则执行下列代码启动一个MDI应用程序。

```
CMultiDocTemplate* pDocTemplate;
pDocTemplate = new CMultiDocTemplate(
   IDR_EXAMPLTYPE,
   RUNTIME_CLASS( CExamplemdiDoc ),
   RUNTIME_CLASS( CChildFrame ),            // custom MDI child frame
   RUNTIME_CLASS( CExamplemdiView )
   );
AddDocTemplate( pDocTemplate );
CMainFrame* pMainFrame = new CMainFrame;
if ( !pMainFrame -> LoadFrame( IDR_MAINFRAME ) )
return FALSE;
```

```
m_pMainWnd = pMainFrame;
pMainFrame -> ShowWindow( m_nCmdShow );
pMainFrame -> UpdateWindow();
```

具体请参考实例4。

A.6.2 使用文档的乐趣

MFC类是用预定义的命令消息ID_FILE_NEW、ID_FILE_OPEN和ID_FILE_SAVE，它们提供了自动装载以及保存文档的方法。但是，可以只通过调用 CDocTemplate的OpenDocumentFile ()成员函数，在任何时候装载一个文档/框架/视组合。

例如，当要打开MDI应用程序中的一个文档的时候，可能打算同时打开另一个文档，只需保存以后想打开的模板即可，如下所示：

```
m_pSaveDocTemplate = new CMultiDocTemplate(
   IDR_EXAMPLTYPE,
   RUNTIME_CLASS( CExamplemdiDoc ),
   RUNTIME_CLASS( CChildFrame ),              // custom MDI child frame
   RUNTIME_CLASS( CExamplemdiView)
   );
```

然后重载OpenDocumentFile()成员函数并添加下列代码：

```
m_pSaveDocTemplate -> OpenDocumentFile( xxx );
```