访问 HTML 文件的任何人都可以自定义应用程序的用户界面,所要用到的工具就像 Notepad 中的一样简单。相反,修改编译后的可执行文件的用户界面就需要更精心的处理了。

由于 DHTML 是语言无关的,所以 DHTML 脚本能够用任何可用在脚本引擎中的脚本编写语言编写。大多数 DHTML 脚本是用 JavaScript 或 Microsoft Visual Basic 的脚本版本(VBScript)(来自于 Visual Basic)编写的。下列 HTML 文件基于 MSDN 提供的示例程序。它使用了 DHTML 和内置的 JavaScript 来显示走动的数字时钟:

```
<HTML>
<HEAD><TITLE>DHTML Clock Demo</TITLE></HEAD>
<BODY BGCOLOR="#FF0000">
<H1 STYLE="font family:comic sans ms" ALIGN=center>DHTML Clock</H1>
<DIV ID=Clock ALIGN=center
STYLE="font-family:arial; font-size:64; color:#FFFFFF">
 </DIV>

<SCRIPT LANGUAGE="JavaScript">
<!--
function tick() {
   var hours, minutes, seconds, ampm;
   var today = new Date();
   var h = today.getHours();
   var m = today.getMinutes();
   var s = today.getSeconds();

   if (h < 12) {
      hours = h + ":";
      ampm = "A.M.";
   }
   else if (h == 12) {
       hours = "12:";
       ampm = "P.M.";
   }
   else {
       h = h - 12;
       hours = h + ":";
       ampm = "P.M.";
   }

   if (m < 10)
      minutes = "0" + m + ":";
   else
       minutes = m + ":";

   if (s < 10)
      seconds = "0" + s + " ";
   else
```

```
        seconds = s + " ";

    Clock.innerHTML = hours + minutes + seconds + ampm;
    window.setTimeout("tick();", 100);
}

window.onload = tick;
-->
</SCRIPT>
</BODY>
</HTML>
```

图 10-4 显示了基于 CHtmlView 的应用程序 HtmlClock，它使用此 HTML 脚本作为时钟程序的基础。HTML 保存在名为 Clock.htm 的文件中。当 HtmlClock 启动时，视图的 OnInitialUpdate 函数会把到 Clock.htm 的路径传递给 Navigate 函数。（由于指定路径名的方式的限制，Clock.htm 必须放在与 HtmlClock.exe 相同的路径下。）实际上，Navigate 将路径名传递给了 WebBrowser 控件，WebBrowser 控件再加载文件，分析 HTML，以及执行脚本。

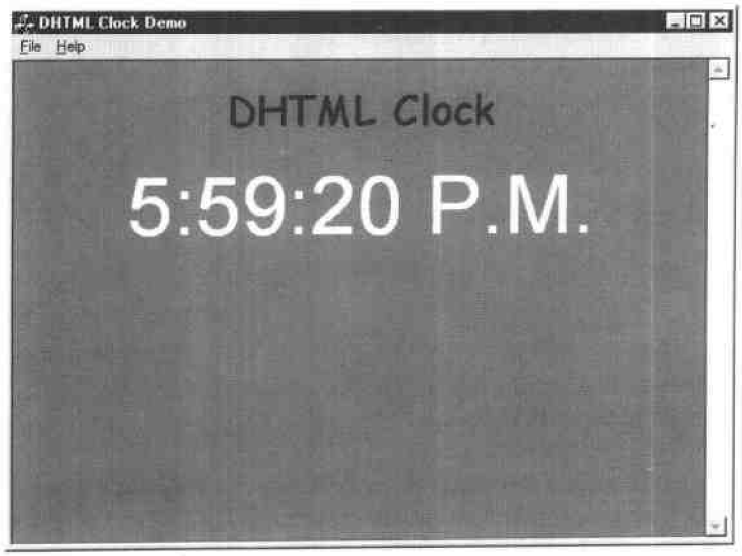

图 10-4　HtmlClock 窗口

在图 10-5 中给出了 HtmlClock 视图类的源程序。为创建 HtmlClock，我用 AppWizard 创建了一个基于 CHtmlView 视图的 SDI 文档/视图程序。修改 AppWizard 提供的 OnInitialUpdate 函数来加载 Clock.htm，添加一个 OnTitleChange 函数用来在框架窗口的标题栏中显示页标题（"DHTML Clock Demo"），并去掉了大多数 AppWizard 生成的应用程序菜单选项。

HtmlClock 只是简要地说明了 HTML 视图的用途。例如，可以在 HTML 视图中运行 Java 小应用程序，可以编写 C++ 程序与 DHTML 对象交互。CHtmlView 还是用来创建基于 HTML

的帮助系统的绝妙工具。如果 HTML 仍旧是软件工业中的宠儿,那么 CHtmlView 就是您进入成功殿堂的门票。

HtmlClockView.h

```
// HtmlClockView.h : interface of the CHtmlClockView class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(
//    AFX_HTMLCLOCKVIEW_H__D39825ED_99C0_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_HTMLCLOCKVIEW_H__D39825ED_99C0_11D2_8E53_006008A82731__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif //_MSC_VER > 1000

class CHtmlClockView : public CHtmlView
{
protected: // create from serialization only
    CHtmlClockView();
    DECLARE_DYNCREATE(CHtmlClockView)

// Attributes
public:
    CHtmlClockDoc* GetDocument();

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CHtmlClockView)
    public:
    virtual void OnDraw(CDC* pDC); // overridden to draw this view
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    virtual void OnTitleChange(LPCTSTR lpszText);
    protected:
    virtual void OnInitialUpdate(); // called first time after construct
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CHtmlClockView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
```

```
protected:

// Generated message map functions
protected:
    //{{AFX_MSG(CHtmlClockView)
        // NOTE - the ClassWizard will add and remove member functions here.
        //    DO NOT EDIT what you see in these blocks of generated code !
        //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG // debug version in HtmlClockView.cpp
inline CHtmlClockDoc* CHtmlClockView::GetDocument()
    { return (CHtmlClockDoc*)m_pDocument; }
#endif

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//    AFX_HTMLCLOCKVIEW_H__D3982BED_99C0_11D2_8B53_006008A82731__INCLUDED_)
```

HtmlClockView.cpp

```
// HtmlClockView.cpp : implementation of the CHtmlClockView class
//

#include "stdafx.h"
#include "HtmlClock.h"

#include "HtmlClockDoc.h"
#include "HtmlClockView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CHtmlClockView

IMPLEMENT_DYNCREATE(CHtmlClockView, CHtmlView)

BEGIN_MESSAGE_MAP(CHtmlClockView, CHtmlView)
    //{{AFX_MSG_MAP(CHtmlClockView)
        // NOTE - the ClassWizard will add and remove mapping macros here.
```

```
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
////////////////////////////////////////////////////////////////////////////
// CHtmlClockView construction/destruction

CHtmlClockView::CHtmlClockView()
{
}

CHtmlClockView::~CHtmlClockView()
{
}

BOOL CHtmlClockView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CHtmlView::PreCreateWindow(cs);
}

////////////////////////////////////////////////////////////////////////////
// CHtmlClockView drawing

void CHtmlClockView::OnDraw(CDC* pDC)
{
    CHtmlClockDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
}

void CHtmlClockView::OnInitialUpdate()
{
    CHtmlView::OnInitialUpdate();

    TCHAR szPath[MAX_PATH];
    ::GetModuleFileName(NULL, szPath, sizeof(szPath)/sizeof(TCHAR));

    CString string = szPath;
    int nIndex = string.ReverseFind(_T('\\'));
    ASSERT(nIndex != -1);
    string = string.Left(nIndex + 1) + _T("Clock.htm");
    Navigate(string);
}

////////////////////////////////////////////////////////////////////////////
// CHtmlClockView diagnostics

#ifdef _DEBUG
void CHtmlClockView::AssertValid() const
{
    CHtmlView::AssertValid();
}
```

```
void CHtmlClockView::Dump(CDumpContext& dc) const
{
    CHtmlView::Dump(dc);
}

CHtmlClockDoc* CHtmlClockView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CHtmlClockDoc)));
    return (CHtmlClockDoc*)m_pDocument;
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CHtmlClockView message handlers

void CHtmlClockView::OnTitleChange(LPCTSTR lpszText)
{
    CHtmlView::OnTitleChange(lpszText);
    AfxGetMainWnd()->SetWindowText(lpszText);
}
```

图 10-5　HtmlClock 应用程序

10.2.4　树形视图

使用 MFC 的 CTreeView 类，程序员可以创建类似于 Windows 资源管理器的左窗格中的视图。树形视图显示一种树形结构，包含的项目有文本和图形。项目还包含子项目和子项目的集合，或“子树”，可以展开和折叠来显示和隐藏其中包含的信息。树形视图用来描述固有分层结构的数据最为理想，如描述硬盘的路径结构。如果您做过一些 Windows 程序设计的话，就可能发现许多对树形视图的应用。

CTreeView 相当简单，这是由于它从树形视图控件中派生了大多数功能，该控件是 Microsoft Windows 95引入的公用控件库中的一员。在 MFC 中，CTreeCtrl 给树形视图控件提供了程序接口。一个树形视图是在树形视图控件的基础上调用 CTreeCtrl 函数编程实现的。CTreeView 函数 GetTreeCtrl 返回该控件的 CTreeCtrl 引用。这样，要想确定树形视图中包含的项目的数量，就不应使用 CTreeView 函数，而要调用 CTreeCtrl::GetCount，如下所示：

```
UINT nCount = GetTreeCtrl().GetCount();
```

此范例调用视图成员函数来获取相应控件的引用，是所有 MFC 的 CCtrlView 派生类共同使用的一种方法。

10.2.5 初始化树形视图

树形视图支持几个影响其外观和操作的窗口样式。其中 6 个样式可以用在所有运行 Windows 95 及其更高版本或运行 Microsoft Windows NT 3.51 及其更高版本的系统中,其他样式可以用在安装了 Internet Explorer 3.0 的系统中,甚至有的样式可用在配有 Internet Explorer 4.0 或更高版本的系统中。(公用控件和 Internet Explorer 之间相关性的介绍请参见第 16 章。)可以对树形视图应用任何系统支持的样式,只要在传递给 PreCreateWindow 的 CREATESTRUCT 结构中的 style 字段对它们进行或运算即可。表 10-4 中列出了对所有树形视图都有效的 6 个样式。

表 10-4 树形视图样式

样式	说 明
TVS_HASLINES	添加线段,将子项目和其父项目连接起来
TVS_LINESATROOT	添加线段,将分层结构的顶层或称为根的项目连接起来。只有指定了 TVS_HASLINES,此样式才有效
TVS_HASBUTTONS	给具有了项目的项目添加带有加号或减号的按钮。单击该按钮可以展开或折叠相关子树
TVS_EDITLABELS	使置换式标签编辑通知有效
TVS_DISABLEDRAGDROP	使拖放通知无效
TVS_SHOWSELALWAYS	指定当前选中的项目总被加亮显示。默认状态下,控件失去输入焦点时加亮显示将被消除

树形视图控件中的每个项目都是由文本字符串(也称为"标签")和可选的图形列表中的图形组成的。图形列表是 Windows 95 中引入的另一个控件类型。在 MFC 中,图形列表由类 CImageList 的实例来表示。可以把图形列表当作具有相似尺寸位图的集合,其中每个位图都由一个从 0 开始的索引号标识。语句

```
CImageList il;
il.Create (IDB_IMAGES, 16, 1, RGB (255, 0, 255));
```

从包含多个图形的位图资源 (ID = IDB_IMAGES) 中创建了一个图形列表。Create 的第二个参数说明每个图形都具有 16 像素宽。最后一个参数中的 COLORREF 值指定品红为图形列表的透明颜色。当图形列表中的图形在树形视图中被显示时,只有非品红像素才出现。

如果想在树形视图中包含文本和图形,就必须创建和初始化一个图形列表,并使用 CTreeCtrl::SetImageList 把它分配给树形视图。假设 il 是一个 CImageList 对象,语句

```
GetTreeCtrl ().SetImageList (&il, TVSIL_NORMAL);
```

将图形列表和控件联系在了一起。TVSIL_NORMAL 告诉树形视图,图形列表中的图形将被

用来表示选中和未选中的项目。可以将单独的 TVSIL_STATE 图形列表分配给树形视图，来表示具有由应用程序定义的状态的项目。注意在树形视图被销毁之前，不能销毁图形列表；否则，图形将会从控件中消失。

CTreeCtrl::InsertItem 给树形视图控件添加一个项目。项目由 HTREEITEM 句柄来标识，并且输入给 InsertItem 的参数之一是父项目的 HTREEITEM 句柄。创建子项目就是将一个项目添加到树形视图中并指定一个项目作为父亲。根项目（树的最顶层项目）是通过指定 TVI_ROOT 作为父亲而创建的。下列示例程序用 2 个 70 年代的摇滚演唱组和他们的唱片初始化了树形视图和其中的子树：

```
// Root items first, with automatic sorting.
HTREEITEM hEagles = GetTreeCtrl().InsertItem(_T("Eagles"),
    TVI_ROOT, TVI_SORT);
HTREEITEM hDoobies = GetTreeCtrl().InsertItem(_T("Doobie Brothers"),
    TVI_ROOT, TVI_SORT);

// Eagles subitems second (no sorting).
GetTreeCtrl().InsertItem(_T("Eagles"), hEagles);
GetTreeCtrl().InsertItem(_T("On the Border"), hEagles);
GetTreeCtrl().InsertItem(_T("Hotel California"), hEagles);
GetTreeCtrl().InsertItem(_T("The Long Run"), hEagles);

// Doobie subitems third (no sorting).
GetTreeCtrl().InsertItem(_T("Toulouse Street"), hDoobies);
GetTreeCtrl().InsertItem(_T("The Captain and Me"), hDoobies);
GetTreeCtrl().InsertItem(_T("Stampede"), hDoobies);
```

给 InsertItem 传递一个 TVI_SORT 标记可以自动地根据同一子树中的其他项目将添加到树中的项目排序。默认值是 TVI_LAST，它只是简单地将新项目加到列表最后。您还可以指定使用 TVI_FIRST 将项目添加到列表头部。

这只是给树形视图控件添加项目的一种方法。由于 CTreeCtrl 提供了四种不同的 InsertItem 版本，所以还可以选择其他方式来添加。让我们把上段中的例子进一步处理一下，假设您想在树形视图项目中包含文本和图形。假定您已经创建了包含两个图形的图形列表。图形 0 描绘的是吉他，图形 1 描绘的是唱片封面。我们希望吉他和摇滚演唱组的名字一起出现，而唱片图形和唱片标题一同出现。以下给出的程序初始化该控件：

```
// Add the image list to the control.
GetTreeCtrl().SetImageList(pImageList, TVSIL_NORMAL);

// Root items first, with automatic sorting
HTREEITEM hEagles = GetTreeCtrl().InsertItem(_T("Eagles"),
```

```
        0, 0, TVI_ROOT, TVI_SORT);
    HTREEITEM hDoobies = GetTreeCtrl().InsertItem(_T("Doobie Brothers"),
        0, 0, TVI_ROOT, TVI_SORT);

    // Eagles subitems second (no sorting)
    GetTreeCtrl().InsertItem(_T("Eagles"), 1, 1, hEagles);
    GetTreeCtrl().InsertItem(_T("On the Border"), 1, 1, hEagles);
    GetTreeCtrl().InsertItem(_T("Hotel California"), 1, 1, hEagles);
    GetTreeCtrl().InsertItem(_T("The Long Run"), 1, 1, hEagles);

    // Doobie subitems third (no sorting)
    GetTreeCtrl().InsertItem(_T("Toulouse Street"), 1, 1, hDoobies);
    GetTreeCtrl().InsertItem(_T("The Captain and Me"), 1, 1, hDoobies);
    GetTreeCtrl().InsertItem(_T("Stampede"), 1, 1, hDoobies);
```

在这种形式的 InsertItem 中，第 2 和第 3 个参数传递的是图形索引号。第一个索引号指定了在项目未选中时树形视图将显示的图形，第二个指定了项目被选定后显示的图形。给两个参数指定相同的索引号说明将用相同的图形表示项目的两种状态。在 Windows 资源管理器的左窗格中，树形视图控件使用了闭合的文件夹来表示未被选中的文件夹项目，用打开的文件夹来表示选中的文件夹项目。因此，当您用箭头键上下移动加亮条时，文件夹会随着加亮条的到达和离开而打开和关闭。

10.2.6 树形视图的成员函数和通知

CTreeCtrl 提供了大量的成员函数来操纵作为基础的树形视图控件，以及用来获取有关项目的信息。例如，DeleteItem 用来从控件中删除项目，DeleteAllItems 用来删除所有的项目，Expand 用来展开或折叠一个子树，SetItemText 用来修改一个项目的标签；而 GetItemText 用来检索标签。SortChildren 可以用来排列子树中的项目。随便您指定什么，可能都会有相应的 CTreeCtrl 函数存在。

对于几乎所有这些函数，最重要的是 HTREEITEM 句柄，它标识了操作的目标项目。如果愿意，可以将 InsertItem 返回的句柄保存在数组、链表或其他结构中，以便以后参考。可以使用 CTreeCtrl::GetSelectedItem 来检索所选项目的句柄。如果必要，可以使用 GetParentItem、GetChildItem、GetNextItem、GetNextSiblingItem 以及其他 CTreeCtrl 函数来从树形视图控件的第一个项目开始逐个枚举项目。

一旦项目添加进去，树形视图就能依靠自己的功能处理大多数用户输入了。用户可以展开和折叠目录分支来浏览项目，可以通过指向和单击来作出选择。通过处理表 10-5 中给出的通知，可以给树形视图添加更多的功能(或自定义对常规输入的默认响应)。通知以 WM_NOTIFY 消息的形式出现，在大多数情况下，lParam 指向一个 NM_TREEVIEW 结构，其中包含有关产生消息的事件的附加信息。以下给出树形视图通知的一些用途：

- 允许原位编辑标签，以便用户可以在树形视图中编辑文本。
- 通过将 LPSTR_TEXTCALLBACK 和 I_IMAGECALLBACK 参数传递给 InsertItem 并处理 TVN_GETDISPINFO 通知来更新项目文本和图形。
- 处理 TVN_KEYDOWN 通知来自定义控件对键盘输入的响应。
- 支持拖放操作。

当然还有很多用途，表 10-5 只是让您对树形视图的广泛适用性有个概念上的认识。

表 10-5　树形视图通知

通知	发送条件
TVN_BEGINDRAG	用鼠标左键开始拖放操作　如果控件具有 TVS　DISABLEDRAG-DROP 样式则不发送
TVN_BEGINRDRAG	用鼠标右键开始拖放操作。如果控件具有 TVS　DISABLEDRAG-DROP 样式则不发送
TVN_BEGINLABELEDIT	开始标签编辑操作。只有在控件具有 TVS　EDITLABELS 样式时才发送
TVN　ENDLABELEDIT	完成标签编辑操作。只有在控件具有 TVS_EDITLABELS 样式时才发送
TVN_GETDISPINFO	控件需要别的信息来显示项目。如果项目文本是 LPSTR_TEXTCALLBACK 或图形索引是 I_IMAGECALLBACK 时才发送
TVN_DELETEITEM	一个项目被删除
TVN_ITEMEXPANDED	子树已经展开或折叠
TVN　ITEMEXPANDING	子树即将展开或折叠
TVN_KEYDOWN	控件具有输入焦点时一个键被按下
TVN_SELCHANGED	选中项目已经更改
TVN　SELCHANGING	选中项目即将更改
TVN_SETDISPINFO	TV　DISPINFO 结构中的信息需要更新

10.2.7　DriveTree 应用程序

图 10-6 所示的 DriveTree 应用程序使用了 CTreeView 派生类 CDriveView，提供了宿主 PC 机的驱动器和目录结构的交互视图。CDriveView::OnInitialUpdate 使用 SetImageList 来引入包含表示不同驱动器类型的图标的图形列表，然后调用辅助函数 AddDrives 来初始化驱动器列表。AddDrives 使用 Win32::GetLogicalDrives 函数来标识系统中的逻辑驱动器。对每个驱动器，它都使用 CDriveView::AddDriveItem 将“驱动器项目”（代表驱动器的树形视图项目）添加到树的最高层。::GetLogicalDrives 返回一个 DWORD 值，其中各二进制位标识驱动器的有效性：位 0 与驱动器 A 相应，位 1 与驱动器 B 相应，等等。AddDrives 只需要几行代码就可以枚

举系统中的驱动器并为它们创建驱动器项目(参见图 10-7)。

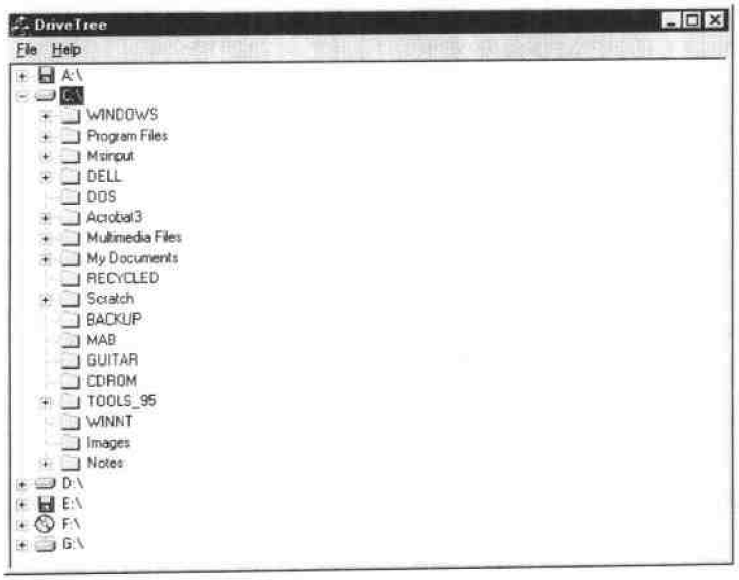

图 10-6 DriveTree 窗口

AddDriveItem 使用 CTreeCtrl::InsertItem 给树添加驱动器项目。对每一个添加的驱动器项目,都要添加一个"虚设"的子项目,使加号出现在驱动器项目的旁边。要确定驱动器的类型(软驱、硬盘等等),以便给驱动器分配图形列表中的图形,AddDriveItem 将使用::GetDriveType API 函数。提供一个指向到驱动器根目录的路径的字符串,::GetDriveType 就会返回一个标识驱动器类型的 UINT 值。表 10-6 中列出了可能的返回值。

表 10-6 ::GetDriveType 可能的返回值

返回值	涵 义
DRIVE_UNKNOWN	驱动器类型未知
DRIVE_NO_ROOT_DIR	驱动器没有根目录
DRIVE_REMOVABLE	驱动器是可移动的(对于软驱和其他可移动媒体驱动器,如 Zip 驱动器,返回此值)
DRIVE_FIXED	驱动器是固定的(对硬盘返回此值)
DRIVE_REMOTE	驱动器是远程驱动器(对网络驱动器返回此值)
DRIVE_CDROM	驱动器是 CD-ROM 驱动器
DRIVE_RAMDISK	驱动器是 RAM 驱动器

AddDriveItem 使用了 switch-case 来处理每个可能的返回值。在 DriveView.cpp 开头处定义的一系列 ILI 值与驱动器类型和图形索引号相关。

在 DriveView.cpp 中剩下的许多代码是为处理 TVN_ITEMEXPANDING 通知编写的。出于操作上的原因，CDriveView 并不用所有驱动器上代表每个目录的项目来初始化自己。相反，它在子树被显示之前才将目录项目添加给子树，在折叠子树的时候删除它们。如果折叠后的子树至少包含一个目录，那么一个子项目就会插入，使得加号出现在子树旁边。子项目永远不会被看到，因为在子树展开之前它就会被删除而由代表实际目录的项目取代。在消息映射表中的 ON_NOTIFY_REFLECT 输入项反射 TVN_ITEMEXPANDING 通知，以便 CDriveView 自己可以处理它们。通知处理程序 OnItemExpanding 既可以给子树添加项目也可以删除项目，但这取决于 NM_TREEVIEW 结构的 action 字段表明是要展开还是要折叠子树。OnItemExpanding 使用辅助函数 AddDirectories 用项目来添加树分支的内容。AddDirectories 接下来使用第 6 章讨论过的::FindFirstFile 和::FindNextFile 函数来枚举目录。

从标题栏删除文档名称

DriveTree 根本就没有使用文档对象。它的 File 菜单中没有包含文档处理命令。由于应用程序不支持加载和保存文档，所以在标题栏显示文档名称就没有意义了。但是 MFC 会自动将文档名称加到标题栏中，因此您必须采取一些特殊措施以防止 MFC 将文档名称插入标题栏。

在框架窗口类中您会找到用于删除文档名称的代码。CMainFrame::PreCreateWindow 包含语句

```
cs.style &= ~FWS_ADDTOTITLE;
```

FWS_ADDTOTITLE 是 MFC 特有的一种特殊的窗口样式，默认状态下包含在框架窗口中。只有具有此样式的窗口才会把文档名称添加给窗口标题栏。在 PreCreateWindow 内通过从窗口标题栏撤消 FWS_ADDTOTITLE 位就阻止了框架结构修改其窗口标题。您可以使用此技术从任何文档/视图应用程序的标题栏中删除文档名称。

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_MAINFRM_H__090B3829_959D_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__090B3829_959D_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
```

```
class CMainFrame : public CFrameWnd
{

protected: // create from serialization only
    CMainFrame();
    DECLARE_DYNCREATE(CMainFrame)

// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMainFrame)
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    //}}AFX_VIRTUAL
// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

// Generated message map functions
protected:
    //{{AFX_MSG(CMainFrame)
        // NOTE - the ClassWizard will add and remove member functions here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//    AFX_MAINFRM_H__090B3829_959D_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//
```

```
#include "stdafx.h"
#include "DriveTree.h"

#include "MainFrm.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMainFrame
IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
}

CMainFrame::~CMainFrame()
{
}

BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CFrameWnd::PreCreateWindow(cs) )
        return FALSE;

    cs.style &= ~FWS_ADDTOTITLE;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
    CFrameWnd::AssertValid();
}
```

```
void CMainFrame::Dump(CDumpContext& dc) const
{
    CFrameWnd::Dump(dc);
}

#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers
```

DriveView.h

```
// DriveTreeView.h : interface of the CDriveView class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_DRIVETREEVIEW_H__090B382D_959D_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_DRIVETREEVIEW_H__090B382D_959D_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif //_MSC_VER > 1000

class CDriveView : public CTreeView
{
protected: // create from serialization only
    CDriveView();
    DECLARE_DYNCREATE(CDriveView)

// Attributes
public:
    CDriveTreeDoc* GetDocument();

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CDriveView)
    public:
    virtual void OnDraw(CDC* pDC); // overridden to draw this view
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    virtual void OnInitialUpdate(); // called first time after construct
    //}}AFX_VIRTUAL

// Implementation
public:
```

```
    virtual ~CDriveView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
protected:

// Generated message map functions
protected:
    BOOL AddDriveItem (LPCTSTR pszDrive);
    int AddDirectories (HTREEITEM hItem, LPCTSTR pszPath);
    void DeleteAllChildren (HTREEITEM hItem);
    void DeleteFirstChild (HTREEITEM hItem);
    CString GetPathFromItem (HTREEITEM hItem);
    BOOL SetButtonState (HTREEITEM hItem, LPCTSTR pszPath);
    int AddDrives ();
    CImageList m_ilDrives;
    //{{AFX_MSG(CDriveView)
    afx_msg void OnItemExpanding(NMHDR* pNMHDR, LRESULT* pResult);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

#ifndef _DEBUG // debug version in DriveTreeView.cpp
inline CDriveTreeDoc* CDriveView::GetDocument()
   { return (CDriveTreeDoc*)m_pDocument; }
#endif

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//    AFX_DRIVETREEVIEW_H__090B382D_959D_11D2_8E53_006008A82731__IN-
CLUDED_)
```

DriveView.cpp

```
// DriveTreeView.cpp : implementation of the CDriveView class
//

#include "stdafx.h"
#include "DriveTree.h"

#include "DriveTreeDoc.h"
#include "DriveView.h"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

// Image indexes
#define ILI_HARD_DISK          0
#define ILI_FLOPPY             1
#define ILI_CD_ROM             2
#define ILI_NET_DRIVE          3
#define ILI_CLOSED_FOLDER      4
#define ILI_OPEN_FOLDER        5

/////////////////////////////////////////////////////////////////////////////
// CDriveView

IMPLEMENT_DYNCREATE(CDriveView, CTreeView)

BEGIN_MESSAGE_MAP(CDriveView, CTreeView)
    //{{AFX_MSG_MAP(CDriveView)
    ON_NOTIFY_REFLECT(TVN_ITEMEXPANDING, OnItemExpanding)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CDriveView construction/destruction

CDriveView::CDriveView()
{
}

CDriveView::~CDriveView()
{
}

BOOL CDriveView::PreCreateWindow(CREATESTRUCT& cs)
{
    if (! CTreeView::PreCreateWindow (cs))
        return FALSE;

    cs.style|= TVS_HASLINES|TVS_LINESATROOT| TVS_HASBUTTONS |
         TVS_SHOWSELALWAYS;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CDriveView drawing
void CDriveView::OnDraw(CDC* pDC)
{
```

```
    CDriveTreeDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    // TODO: add draw code for native data here
}

void CDriveView::OnInitialUpdate()
{
    CTreeView::OnInitialUpdate();

    //
    // Initialize the image list.
    //
    m_ilDrives.Create (IDB_DRIVEIMAGES, 16, 1, RGB (255, 0, 255));
    GetTreeCtrl ().SetImageList (&m_ilDrives, TVSIL_NORMAL);

    //
    // Populate the tree view with drive items.
    //
    AddDrives ();

    //
    // Show the folders on the current drive.
    //
    TCHAR szPath[MAX_PATH];
    ::GetCurrentDirectory (sizeof (szPath) / sizeof (TCHAR), szPath);
    CString strPath = szPath;
    strPath = strPath.Left (3);

    HTREEITEM hItem = GetTreeCtrl ().GetNextItem (NULL, TVGN_ROOT);
    while (hItem != NULL) {
        if (GetTreeCtrl ().GetItemText (hItem) == strPath)
            break;
        hItem = GetTreeCtrl ().GetNextSiblingItem (hItem);
    }

    if (hItem != NULL) {
        GetTreeCtrl ().Expand (hItem, TVE_EXPAND);
        GetTreeCtrl ().Select (hItem, TVGN_CARET);
    }
}

/////////////////////////////////////////////////////////////////////////////
// CDriveView diagnostics
#ifdef _DEBUG
void CDriveView::AssertValid() const
{
    CTreeView::AssertValid();
}
```

```
void CDriveView::Dump(CDumpContext& dc) const
{
    CTreeView::Dump(dc);
}

CDriveTreeDoc* CDriveView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CDriveTreeDoc)));
    return (CDriveTreeDoc*)m_pDocument;
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CDriveView message handlers

int CDriveView::AddDrives()
{
    int nPos = 0;
    int nDrivesAdded = 0;
    CString string = _T("?:\\");

    DWORD dwDriveList = ::GetLogicalDrives();

    while (dwDriveList) {
        if (dwDriveList & 1) {
            string.SetAt(0, _T('A') + nPos);
            if (AddDriveItem(string))
                nDrivesAdded++;
        }
        dwDriveList >>= 1;
        nPos++;
    }
    return nDrivesAdded;
}

BOOL CDriveView::AddDriveItem(LPCTSTR pszDrive)
{
    CString string;
    HTREEITEM hItem;

    UINT nType = ::GetDriveType(pszDrive);
    switch (nType) {

    case DRIVE_REMOVABLE:
        hItem = GetTreeCtrl().InsertItem(pszDrive, ILI_FLOPPY,
            ILI_FLOPPY);
        GetTreeCtrl().InsertItem(_T(""), ILI_CLOSED_FOLDER,
            ILI_CLOSED_FOLDER, hItem);
        break;
```

```
    case DRIVE_FIXED:
    case DRIVE_RAMDISK:
        hItem = GetTreeCtrl().InsertItem(pszDrive, ILI_HARD_DISK,
            ILI_HARD_DISK);
        SetButtonState(hItem, pszDrive);
        break;

    case DRIVE_REMOTE:
        hItem = GetTreeCtrl().InsertItem(pszDrive, ILI_NET_DRIVE,
            ILI_NET_DRIVE);
        SetButtonState(hItem, pszDrive);
        break;

    case DRIVE_CDROM:
        hItem = GetTreeCtrl().InsertItem(pszDrive, ILI_CD_ROM,
            ILI_CD_ROM);
        GetTreeCtrl().InsertItem(_T(""), ILI_CLOSED_FOLDER,
            ILI_CLOSED_FOLDER, hItem);
        break;

    default:
        return FALSE;
    }
    return TRUE;
}

BOOL CDriveView::SetButtonState(HTREEITEM hItem, LPCTSTR pszPath)
{
    HANDLE hFind;
    WIN32_FIND_DATA fd;
    BOOL bResult = FALSE;

    CString strPath = pszPath;
    if (strPath.Right(1) != _T("\\"))
        strPath += _T("\\");
    strPath += _T("*.*");

    if ((hFind = ::FindFirstFile(strPath, &fd)) == INVALID_HANDLE_VALUE)
        return bResult;
    do {
        if (fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY) {
            CString strComp = (LPCTSTR)&fd.cFileName;
            if ((strComp != _T(".")) && (strComp != _T(".."))) {
                GetTreeCtrl().InsertItem(_T(""), ILI_CLOSED_FOLDER,
                    ILI_CLOSED_FOLDER, hItem);
                bResult = TRUE;
```

```
                break;
            }
        }
    } while (::FindNextFile (hFind, &fd));

    ::FindClose (hFind);
    return bResult;
}

void CDriveView::OnItemExpanding(NMHDR * pNMHDR, LRESULT * pResult)
{
    NM_TREEVIEW * pNMTreeView = (NM_TREEVIEW *)pNMHDR;
    HTREEITEM hItem = pNMTreeView->itemNew.hItem;
    CString string = GetPathFromItem (hItem);

    *pResult = FALSE;

    if (pNMTreeView->action == TVE_EXPAND) {
        DeleteFirstChild (hItem);
        if (AddDirectories (hItem, string) == 0)
            *pResult = TRUE;
    }
    else { // pNMTreeView->action == TVE_COLLAPSE
        DeleteAllChildren (hItem);
        if (GetTreeCtrl ().GetParentItem (hItem) == NULL)
            GetTreeCtrl ().InsertItem (_T (""), ILI_CLOSED_FOLDER,
                ILI_CLOSED_FOLDER, hItem);
            else
                SetButtonState (hItem, string);
        }
    }

    CString CDriveView::GetPathFromItem(HTREEITEM hItem)
    {
        CString strResult = GetTreeCtrl ().GetItemText (hItem);

        HTREEITEM hParent;
        while ((hParent = GetTreeCtrl ().GetParentItem (hItem)) != NULL) {
            CString string = GetTreeCtrl ().GetItemText (hParent);
            if (string.Right (1) != _T ("\\"))
                string += _T ("\\");
            strResult = string + strResult;
            hItem = hParent;
        }
        return strResult;
    }
```

```
void CDriveView::DeleteFirstChild(HTREEITEM hItem)
{
    HTREEITEM hChildItem;
    if ((hChildItem = GetTreeCtrl().GetChildItem(hItem)) != NULL)
        GetTreeCtrl().DeleteItem(hChildItem);
}

void CDriveView::DeleteAllChildren(HTREEITEM hItem)
{
    HTREEITEM hChildItem;
    if ((hChildItem = GetTreeCtrl().GetChildItem(hItem)) == NULL)
        return;

    do {
        HTREEITEM hNextItem =
            GetTreeCtrl().GetNextSiblingItem(hChildItem);
        GetTreeCtrl().DeleteItem(hChildItem);
        hChildItem = hNextItem;
    } while (hChildItem != NULL);
}

int CDriveView::AddDirectories(HTREEITEM hItem, LPCTSTR pszPath)
{
    HANDLE hFind;
    WIN32_FIND_DATA fd;
    HTREEITEM hNewItem;

    int nCount = 0;

    CString strPath = pszPath;
    if (strPath.Right(1) != _T("\\"))
        strPath += _T("\\");
    strPath += _T("*.*");

    if ((hFind = ::FindFirstFile(strPath, &fd)) == INVALID_HANDLE_VALUE) {
        if (GetTreeCtrl().GetParentItem(hItem) == NULL)
            GetTreeCtrl().InsertItem(_T(""), ILI_CLOSED_FOLDER,
                ILI_CLOSED_FOLDER, hItem);
        return 0;
    }
    do {
        if (fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY) {
            CString strComp = (LPCTSTR) &fd.cFileName;
            if ((strComp != _T(".")) && (strComp != _T(".."))) {
                hNewItem =
                    GetTreeCtrl().InsertItem((LPCTSTR) &fd.cFileName,
```

```
                    ILI_CLOSED_FOLDER, ILI_OPEN_FOLDER, hItem);

                CString strNewPath = pszPath;
                if (strNewPath.Right (1) != _T ("\\"))
                    strNewPath += _T ("\\");

                strNewPath += (LPCTSTR) &fd.cFileName;
                SetButtonState (hNewItem, strNewPath);
                nCount ++ ;
            }
        }
    } while (::FindNextFile (hFind, &fd));

    ::FindClose (hFind);
    return nCount;
}
```

图 10-7 DriveTree 应用程序

10.3 列表视图

列表视图与树形视图的相似之处在于,它们都为向用户呈现复杂的数据集合提供了功能强大的基础结构。但是如果树形视图是描述分层关系的理想视图的话,那么可以说列表视图最适合描述"扁平"数据集合,如文件名列表。

像树形视图中的项目一样,列表视图中的项目也可以包含文本和图形。另外,这些项目可以具有只能是文本的"子项目",其中包含有关相关项目的其他信息。当控件处于"报表"模式的时候子项目可见,该模式是列表视图支持的4种表现样式中的一种。其他表现样式是:大图标模式、小图标模式、列表模式。要看4种表现样式的例子,可以启动 Windows 资源管理器并使用【查看】菜单来修改右窗格。其中【大图标】命令相应于大图标模式,【小图标】命令相应于小图标模式,【列表】相应于列表模式,【详细资料】相应于报表模式。

10.3.1 初始化列表视图

MFC 的 CListView 类是列表视图的基类。CListView 从列表视图控件中派生了大部分的功能,列表视图控件像树形视图控件一样,是公用控件库中的一部分。MFC 在 CListCtrl 类中封装了列表视图控件。要编制一个列表视图,可以调用 CListView::GetListCtrl 来获得出现在列表视图内部的控件的 CListCtrl 引用,然后调用 CListCtrl 函数使用返回的引用。

在您从 CListView 派生类时,总要在派生类中覆盖 PreCreateWindow 并给视图应用一个以上的默认样式。表 10-7 中列出了所有列表视图都支持的样式。其他列表视图样式在运行 Internet Explorer 3.0 或更高版本的系统中有效。

表 10-7　列表视图样式

样式	说　明
LVS_ICON	选择大图标模式
LVS_SMALLICON	选择小图标模式
LVS_LIST	选择列表模式
LVS_REPORT	选择报表模式
LVS_NOCOLUMNHEADER	删除通常出现在报表模式中的头标控件
LVS_NOSORTHEADER	使 LVN_COLUMNCLICK 通知失效，默认状态下该通知在报表模式中的列头标被单击时发送
LVS_ALIGNLEFT	在大图标和小图标模式下沿左边框对齐项目
LVS_ALIGNTOP	在大图标和小图标模式下沿顶边对齐项目
LVS_AUTOARRANGE	在大图标和小图标模式下自动按行列排列项目
LVS_EDITLABELS	使原位编辑标签有效
LVS_NOLABELWRAP	在大图标模式下限制标签为单行显示
LVS_NOSCROLL	使滚动失效。在默认时有效
LVS_OWNERDRAWFIXED	指定响应 WM_DRAWITEM 消息时控件所有者将绘制项目
LVS_SHAREIMAGELISTS	防止列表视图在自身被删除时去自动删除与它关联的图形列表
LVS_SINGLESEL	使多重选择支持无效
LVS_SHOWSELALWAYS	指定被选中的项目始终加亮显示。默认情况下，加亮效果在视图失去输入焦点时消失
LVS_SORTASCENDING	指定项目按字母升序排列(例如从 A 到 Z)
LVS_SORTDESCENDING	指定项目按字母降序排列(例如从 Z 到 A)

像树形视图一样，列表视图在刚创建时也是空的，初始化可分为 5 步进行：

1. 创建一对图形列表用来保存列表视图项目的图形。一个图形列表保存“大”图形用于大图标模式；另一个保存“小”图形用于小图标、列表以及报表模式。
2. 使用 CListCtrl::SetImageList 将图形列表与列表视图联系起来。给 SetImageList 传递 LVSIL_NORMAL 标志说明图形列表包含大图形，传递 LVSIL_SMALL 标志说明图形列表包含小图形。
3. 用 CListCtrl::InsertColumn 给列表视图控件添加列。最左边的列显示控件中加入的项目。右边的列显示子项目，但只有在报表模式下才可见。
4. 用 CListCtrl::InsertItem 给控件添加项目。
5. 用 CListCtrl::SetItemText 给项目的子项目分配文本字符串。

实际处理过程并不像听上去这么难。下列程序代码用代表美国 8 个州的项目初始化了一个列表视图。每个项目都包含一个标签和图形。标签是州的名字，图形大致显示了滑块大小的轮廓图。每个项目还包含一对子项目：一个文本字符串命名首府、一个文本字符串

描述州的土地面积。在报表模式下,子项目出现在头标为"Capital"和"Area(sq. miles)"的列中。

```
static CString text[8][3] = {
    _T("Tennessee"),        _T("Nashville"),      _T("41,154"),
    _T("Alabama"),          _T("Montgomery"),     _T("50,766"),
    _T("Mississippi"),      _T("Jackson"),        _T("47,234"),
    _T("Florida"),          _T("Tallahassee"),    _T("54,157"),
    _T("Georgia"),          _T("Atlanta"),        _T("58,060"),
    _T("Kentucky"),         _T("Frankfort"),      _T("39,674"),
    _T("North Carolina"),   _T("Raleigh"),        _T("48,843"),
    _T("South Carolina"),   _T("Columbia"),       _T("30,207")
};

// Assign image lists.
GetListCtrl().SetImageList(&ilLarge, LVSIL_NORMAL);
GetListCtrl().SetImageList(&ilSmall, LVSIL_SMALL);

// Add columns.
GetListCtrl().InsertColumn(0,_T("State"), LVCFMT_LEFT, 96);
GetListCtrl().InsertColumn(1,_T("Capital"), LVCFMT_LEFT, 96);
GetListCtrl().InsertColumn(2,_T("Area(sq. miles)"),
    LVCFMT_RIGHT, 96);

// Add items and subitems.
for(int i = 0; i < 8; i++) {
    GetListCtrl().InsertItem(i, (LPCTSTR) text[i][0], i);
    GetListCtrl().SetItemText(i, 1, (LPCTSTR) text[i][1]);
    GetListCtrl().SetItemText(i, 2, (LPCTSTR) text[i][2]);
}
```

传递给 InsertColumn 的参数按顺序指定了列的索引号(从 0 开始编号)、出现在列顶部的标签、列的对齐方式(在列中显示的数据是左对齐、右对齐,还是居中)以及以像素为单位的列宽度。可以使用 CListCtrl::GetStringWidth 来获得由字符宽度所决定的列宽度,这里的字符是控件字体下的字符,并以此列宽度为基准将文本字符串用像素来记数。传递给 InsertItem 的参数指定了项目的索引号(从 0 开始编号)、项目标签以及图形列表中相应图形的索引号。传递给 SetItemText 的参数按顺序指定了项目数、子项目数以及子项目文本。

10.3.2　修改表现样式

在列表视图被创建时,它的表现样式 LVS_ICON、LVS_SMALLICON、LVS_LIST 或 LVS_REPORT 分别决定了它的启动模式是大图标模式、小图标模式、列表模式还是报表模式。在 PreCreateWindow 中使用的是默认表现样式。但是通过修改表现模式您可以自由地选择显示模式。下列语句给列表视图选择了小图标模式:

```
ModifyStyle (LVS_TYPEMASK, LVS_SMALLICON);
```

与此相似，下列语句给视图选择了报表模式：

```
ModifyStyle (LVS_TYPEMASK, LVS_REPORT);
```

ModifyStyle 是一个 CWnd 函数，它是由 CListView 继承而来的。传递给 ModifyStyle 的第一个参数指定某样式位关闭，第 2 个参数指定某样式位打开。LVS_TYPEMASK 可以屏蔽所有 4 个表现样式。

LVS_ICON、LVS_SMALLICON、LVS_LIST 以及 LVS_REPORT 不是真正的位标志，因此可以很方便地用 LVS_TYPEMASK 来查询列表视图当前的表现样式。以下程序是不正确的：

```
// Wrong!
DWORD dwStyle = GetStyle();
if (dwStyle & LVS_ICON)
    // Large icon mode.
else if (dwStyle & LVS_SMALLICON)
    // Small icon mode.
else if (dwStyle & LVS_LIST)
    // List mode.
else if (dwStyle & LVS_REPORT)
    // Report mode.
```

而下列程序是正确的：

```
DWORD dwStyle = GetStyle() & LVS_TYPEMASK;
if (dwStyle == LVS_ICON)
    // Large icon mode.
else if (dwStyle == LVS_SMALLICON)
    // Small icon mode.
else if (dwStyle == LVS_LIST)
    // List mode.
else if (dwStyle == LVS_REPORT)
    // Report mode.
```

这是一种在更新菜单项或其他用户界面对象(它们依赖于列表视图的表现样式)之前确定视图类型的很好方法。

10.3.3　在列表视图中排序

在一个不带 LVS_NOCOLUMNHEADER 样式的列表视图中选择了报表模式时，它会自动显示一个头标控件，即带有形似按钮而作为每列标题的“头标项”。用户可以拖动分割头标项的垂直分隔线来改变列的宽度。(为进一步理解，您可以用 CListCtrl::GetColumnWidth 在列表视图销毁之前检索列宽度，并把结果保存在寄存器中。在下次列表视图创建时可以恢复列宽度，这样用户列宽度参数就可以保留下去。)除非列表视图具有样式 LVS_

NOSORTHEADER,否则单击头标项会给列表视图的父亲发送一个 LVN _COLUMNCLICK 通知。消息的 lParam 指向 NM _LISTVIEW 结构,该结构的 iSubItem 字段保存从 0 开始编号的索引号,用来标识被单击的列。

应用程序通常对 LVN _COLUMNCLICK 通知的响应是调用 CListCtrl::SortItems 来给列表视图项目排序。您看,有多么棒! 现在我可以创建一个能够排序的列表视图了,并且不需要为排序操作编写代码。可是您确实需要提供一个回调函数,控件的内置排序例程可以调用它来比较一对任意选中的项目,而编写比较函数实质上要比编写完整的冒泡排序或快速排序例程简单。而且比较函数是由应用程序定义的,因此您可以对列表视图控件中的项目按词典顺序排序的方法享有全部控制权。

不利的方面是比较程序仅接收 3 个参数:两个被比较项目的 32 位 lParam 值和一个应用程序定义的 lParam 值,该值等于传递给 SortItems 的第 2 个参数。可以在 InsertItem 的调用中或在单独对 CListCtrl::SetItemData 的调用中赋给项目 lParam 值。除非应用程序保存了每个项目数据的私有副本,并在 lParam 中存放了一个值允许项目的数据被检索,否则比较函数得不到执行。虽然应用程序给每个项目分配内存并将指针填入项目的 lParams 中并不困难,但是由于事后必须释放内存,所以把事情搞复杂了。而且如果一个应用程序将文本字符串赋给列表视图的项目和子项目,而又要保存自己的项目数据,那么在内存使用方面它的效率就不高了,因为数据最终会被保存两次。要避免这种浪费,可以对项目和子项目文本指定 LPSTR _TEXTCALLBACK,并且在响应 LVN _GETDISPINFO 通知时给列表视图控件提供文本。但这样也给编程增加了复杂性,要求支持 CListCtrl::SortItems 的基础结构就不像起先那么简单了。稍后,我们将开发一个在列表视图中应用了可排序列的应用程序,您就可以直接看到处理方法了。

10.3.4 列表视图中的命中测试

通过处理 NM _CLICK、NM _DBLCLK、NM _RCLICK 以及 NM _RDBLCLK 通知可以在列表视图中响应鼠标单击事件。通常,响应这些事件的方法依赖于单击(或双击)发生时鼠标指针下面的情况。可以使用 CListCtrl::HitTest 对列表视图中的项目进行命中测试。给定一个点的坐标,HitTest 返回该点处项目的索引号,或者该点与项目不相对应时返回 -1。

下列程序说明了如何在列表视图中处理双击事件。消息映射表中的 ON _NOTIFY _REFLECT输入项将 NM _DBLCLK 通知反射回列表视图。NM _DBLCLK 处理程序使用了 MFC 的 TRACE 宏将被双击项目的名字返回到调试输出窗口:

```
// In CMyListView's message map
ON_NOTIFY_REFLECT (NM_DBLCLK, OnDoubleClick)
   .
   .
   .
void CMyListView::OnDoubleClick (NMHDR* pnmh, LRESULT* pResult)
```

```
{
    DWORD dwPos = ::GetMessagePos ();
    CPoint point ((int) LOWORD (dwPos), (int) HIWORD (dwPos));
    GetListCtrl ().ScreenToClient (&point);

    int nIndex;
    if ((nIndex = GetListCtrl ().HitTest (point)) != -1) {
        CString string = GetListCtrl ().GetItemText (nIndex, 0);
        TRACE (_T ("%s was double-clicked\n"), string);
    }
    *pResult = 0;
}
```

在 NM_DBLCLK 通知中并不包含鼠标指针的坐标，因此可以用::GetMessagePos 来检索指针位置。由::GetMessagePos 返回的屏幕坐标被转换为列表视图中的客户区坐标并传递给了 CListCtrl::HitTest。如果 HitTest 返回项目索引号，索引号就用来检索项目文本。

10.3.5　WinDir 应用程序

图 10-8 所示的 WinDir 应用程序之所以如此命名是因为它的输出让我们回忆起了 MS-DOS 中的 DIR 命令，尽管它是图形形式。它使用 CListView 派生类 CFileView 来显示指定目录下的所有文件的一个列表。可以通过选中 File 菜单中的 New Directory 命令并输入路径名来选择目录。在检索了输入的路径之后，WinDir 将路径名传递给 CFileView::Refresh 来显示目录中的内容。您可以在 FileView.cpp 中找到这些程序，另外在图 10-9 中还给出了 WinDir 的其他源程序代码。

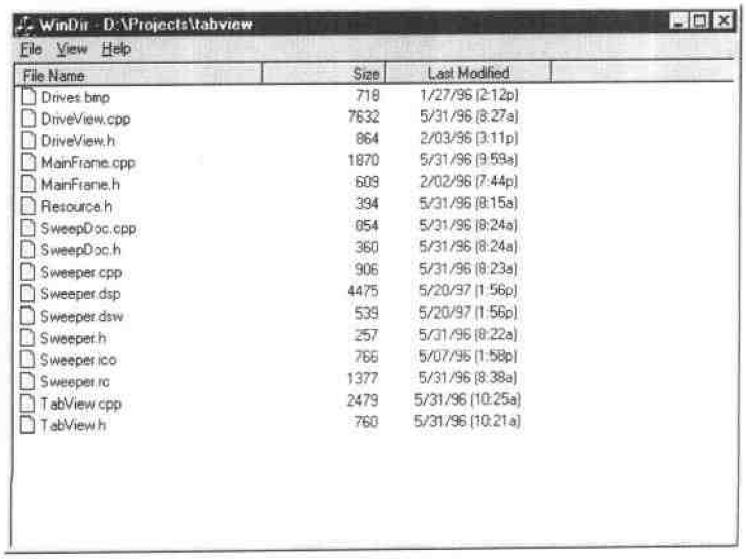

图 10-8　WinDir 窗口

下面概括性地介绍一下 CFileView 的工作过程。首先,CFileView::Refresh 使用 ::FindFirstFile 和 ::FindNextFile 生成一个文件名列表。对于它标识的每个文件,Refresh 都要调用 CFileView::AddItem 给列表视图添加一个项目。接下来,AddItem 为 ITEMINFO 数据结构(在 FileView.h 中定义)分配内存;用文件的名称、大小以及日期和时间标记来初始化该结构;并给其 lParam 是结构地址的列表视图添加项目。下面是删除了错误检查代码后的程序段:

```
ITEMINFO * pItem;
pItem = new ITEMINFO;

pItem -> strFileName = pfd -> cFileName;
pItem -> nFileSizeLow = pfd -> nFileSizeLow;
pItem -> ftLastWriteTime = pfd -> ftLastWriteTime;

LV_ITEM lvi;
lvi.mask = LVIF_TEXT|LVIF_IMAGE|LVIF_PARAM;
lvi.iItem = nIndex;
lvi.iSubItem = 0;
lvi.iImage = 0;
lvi.pszText = LPSTR_TEXTCALLBACK;
lvi.lParam = (LPARAM) pItem;

GetListCtrl().InsertItem(&lvi);
```

请注意 LV_ITEM 结构的 pszText 字段中指定的 LPSTR_TEXTCALLBACK 值。AddItem 并没有赋给项目一个文本字符串,而是告诉列表视图"项目需要标签时再调用我"。由于 LPSTR_TEXTCALLBACK 是子项目的默认值,所以没必要初始化子项目了。

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_MAINFRM_H__18BD7B7C_95C6_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__18BD7B7C_95C6_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMainFrame : public CFrameWnd
{

protected: // create from serialization only
    CMainFrame();
    DECLARE_DYNCREATE(CMainFrame)
```

```
// Attributes
public:

// Operations
public:
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMainFrame)
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

// Generated message map functions
protected:
    //{{AFX_MSG(CMainFrame)
      // NOTE - the ClassWizard will add and remove member functions here.
      //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(AFX_MAINFRM_H__18BD757C_95C6_11D2_8253_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//

#include "stdafx.h"
#include "WinDir.h"
#include "MainFrm.h"

#ifdef _DEBUG
#define new DEBUG_NEW
```

```
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMainFrame

IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code !
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
}

CMainFrame::~CMainFrame()
{
}

BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CFrameWnd::PreCreateWindow(cs) )
        return FALSE;

    cs.style &= ~FWS_ADDTOTITLE;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
    CFrameWnd::AssertValid();
}

void CMainFrame::Dump(CDumpContext& dc) const
{
    CFrameWnd::Dump(dc);
}

#endif // _DEBUG
```

```
/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers
```

FileView.h

```
// FileView.h : interface of the CFileView class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_FILEVIEW_H__18BD7B80_95C6_11D2_8B53_006008A82731__INCLUDED_)
#define AFX_FILEVIEW_H__18BD7B80_95C6_11D2_8B53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

typedef struct tagITEMINFO {
    CString     strFileName;
    DWORD       nFileSizeLow;
    FILETIME    ftLastWriteTime;
} ITEMINFO;

class CFileView : public CListView
{
protected: // create from serialization only
    CFileView();
    DECLARE_DYNCREATE(CFileView)

// Attributes
public:
    CWinDirDoc* GetDocument();
// Operations
public:
    static int CALLBACK CompareFunc (LPARAM lParam1, LPARAM lParam2,
        LPARAM lParamSort);

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CFileView)
    public:
    virtual void OnDraw(CDC* pDC);  // overridden to draw this view
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    virtual void OnInitialUpdate(); // called first time after construct
    //}}AFX_VIRTUAL

// Implementation
public:
    int Refresh (LPCTSTR pszPath);
```

```
    virtual ~CFileView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    CString m_strPath;
    void FreeItemMemory();
    BOOL AddItem(int nIndex, WIN32_FIND_DATA* pfd);
    CImageList m_ilSmall;
    CImageList m_ilLarge;
    //{{AFX_MSG(CFileView)
    afx_msg void OnDestroy();
    afx_msg void OnGetDispInfo(NMHDR* pNMHDR, LRESULT* pResult);
    afx_msg void OnColumnClick(NMHDR* pNMHDR, LRESULT* pResult);
    afx_msg void OnViewLargeIcons();
    afx_msg void OnViewSmallIcons();
    afx_msg void OnViewList();
    afx_msg void OnViewDetails();
    afx_msg void OnUpdateViewLargeIcons(CCmdUI* pCmdUI);
    afx_msg void OnUpdateViewSmallIcons(CCmdUI* pCmdUI);
    afx_msg void OnUpdateViewList(CCmdUI* pCmdUI);
    afx_msg void OnUpdateViewDetails(CCmdUI* pCmdUI);
    afx_msg void OnFileNewDirectory();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in FileView.cpp
inline CWinDirDoc* CFileView::GetDocument()
   { return (CWinDirDoc*)m_pDocument; }
#endif

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//     AFX_FILEVIEW_H__13BD7B80_95C6_11D2_8B53_006008A82731__INCLUDED_)
```

FileView.cpp

```
// FileView.cpp : implementation of the CFileView class
//

#include "stdafx.h"
#include "WinDir.h"
#include "PathDialog.h"
#include "WinDirDoc.h"
#include "FileView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CFileView

IMPLEMENT_DYNCREATE(CFileView, CListView)

BEGIN_MESSAGE_MAP(CFileView, CListView)
    //{{AFX_MSG_MAP(CFileView)
    ON_WM_DESTROY()
    ON_NOTIFY_REFLECT(LVN_GETDISPINFO, OnGetDispInfo)
    ON_NOTIFY_REFLECT(LVN_COLUMNCLICK, OnColumnClick)
    ON_COMMAND(ID_VIEW_LARGE_ICONS, OnViewLargeIcons)
    ON_COMMAND(ID_VIEW_SMALL_ICONS, OnViewSmallIcons)
    ON_COMMAND(ID_VIEW_LIST, OnViewList)
    ON_COMMAND(ID_VIEW_DETAILS, OnViewDetails)
    ON_UPDATE_COMMAND_UI(ID_VIEW_LARGE_ICONS, OnUpdateViewLargeIcons)
    ON_UPDATE_COMMAND_UI(ID_VIEW_SMALL_ICONS, OnUpdateViewSmallIcons)
    ON_UPDATE_COMMAND_UI(ID_VIEW_LIST, OnUpdateViewList)
    ON_UPDATE_COMMAND_UI(ID_VIEW_DETAILS, OnUpdateViewDetails)
    ON_COMMAND(ID_FILE_NEW_DIR, OnFileNewDirectory)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CFileView construction/destruction

CFileView::CFileView()
{
}

CFileView::~CFileView()
{
}
```

```
BOOL CFileView::PreCreateWindow(CREATESTRUCT& cs)
{
    if (! CListView::PreCreateWindow (cs))
        return FALSE;

    cs.style &= ~LVS_TYPEMASK;
    cs.style|= LVS_REPORT;
    return TRUE;
}

////////////////////////////////////////////////////////////////////////////
// CFileView drawing

void CFileView::OnDraw(CDC * pDC)
{
    CWinDirDoc * pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    // TODO: add draw code for native data here
}

void CFileView::OnInitialUpdate()
{
    CListView::OnInitialUpdate();

    //
    // Initialize the image list.
    //
    m_ilLarge.Create (IDB_LARGEDOC, 32, 1, RGB (255, 0, 255));
    m_ilSmall.Create (IDB_SMALLDOC, 16, 1, RGB (255, 0, 255));

    GetListCtrl ().SetImageList (&m_ilLarge, LVSIL_NORMAL);
    GetListCtrl ().SetImageList (&m_ilSmall, LVSIL_SMALL);

    //
    // Add columns to the list view.
    //
    GetListCtrl ().InsertColumn (0,_T ("File Name"), LVCFMT_LEFT, 192);
    GetListCtrl ().InsertColumn (1,_T ("Size"), LVCFMT_RIGHT, 96);
    GetListCtrl ().InsertColumn (2,_T ("Last Modified"), LVCFMT_CENTER, 128);

    //
    // Populate the list view with items.
    //
    TCHAR szPath[MAX_PATH];
    ::GetCurrentDirectory (sizeof (szPath) / sizeof (TCHAR), szPath);
    Refresh (szPath);
}

////////////////////////////////////////////////////////////////////////////
```

```
// CFileView diagnostics

#ifdef _DEBUG
void CFileView::AssertValid() const

    CListView::AssertValid();
}

void CFileView::Dump(CDumpContext& dc) const
{
    CListView::Dump(dc);
}

CWinDirDoc* CFileView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CWinDirDoc)));
    return (CWinDirDoc*)m_pDocument;
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CFileView message handlers

int CFileView::Refresh(LPCTSTR pszPath)
{
    CString strPath = pszPath;
    if (strPath.Right(1) != _T("\\"))
        strPath += _T("\\");
    strPath += _T("*.*");

    HANDLE hFind;
    WIN32_FIND_DATA fd;
    int nCount = 0;

    if ((hFind = ::FindFirstFile(strPath, &fd)) != INVALID_HANDLE_VALUE) {
        //
        // Delete existing items (if any).
        //
        GetListCtrl().DeleteAllItems();

        //
        // Show the path name in the frame window's title bar.
        //
        TCHAR szFullPath[MAX_PATH];
        ::GetFullPathName(pszPath, sizeof(szFullPath) / sizeof(TCHAR),
            szFullPath, NULL);
        m_strPath = szFullPath;

        CString strTitle = _T("WinDir - ");
```

```
        strTitle += szFullPath;
        AfxGetMainWnd()->SetWindowText(strTitle);

        //
        // Add items representing files to the list view.
        //
        if (!(fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY))
            AddItem(nCount++, &fd);

        while (::FindNextFile(hFind, &fd)) {
            if (!(fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY))
                if (!AddItem(nCount++, &fd))
                    break;
        }
        ::FindClose(hFind);
    }
    return nCount;
}

BOOL CFileView::AddItem(int nIndex, WIN32_FIND_DATA *pfd)
{
    //
    // Allocate a new ITEMINFO structure and initialize it with information
    // about the item.
    //
    ITEMINFO* pItem;
    try {
        pItem = new ITEMINFO;
    }
    catch (CMemoryException* e) {
        e->Delete();
        return FALSE;
    }

    pItem->strFileName = pfd->cFileName;
    pItem->nFileSizeLow = pfd->nFileSizeLow;
    pItem->ftLastWriteTime = pfd->ftLastWriteTime;

    //
    // Add the item to the list view.
    //
    LV_ITEM lvi;
    lvi.mask = LVIF_TEXT|LVIF_IMAGE|LVIF_PARAM;
    lvi.iItem = nIndex;
    lvi.iSubItem = 0;
    lvi.iImage = 0;
    lvi.pszText = LPSTR_TEXTCALLBACK;
    lvi.lParam = (LPARAM) pItem;
```

```
    if (GetListCtrl().InsertItem(&lvi) == -1)
        return FALSE;

    return TRUE;
}

void CFileView::FreeItemMemory()
{
    int nCount = GetListCtrl().GetItemCount();
    if (nCount) {
        for (int i = 0; i < nCount; i++)
            delete (ITEMINFO*) GetListCtrl().GetItemData(i);
    }
}

void CFileView::OnDestroy()
{
    FreeItemMemory();
    CListView::OnDestroy();
}

void CFileView::OnGetDispInfo(NMHDR* pNMHDR, LRESULT* pResult)
{
    CString string;
    LV_DISPINFO* pDispInfo = (LV_DISPINFO*) pNMHDR;

    if (pDispInfo->item.mask & LVIF_TEXT) {
        ITEMINFO* pItem = (ITEMINFO*) pDispInfo->item.lParam;

        switch (pDispInfo->item.iSubItem) {

        case 0: // File name.
            ::lstrcpy(pDispInfo->item.pszText, pItem->strFileName);
            break;

        case 1: // File size.
            string.Format(_T("%u"), pItem->nFileSizeLow);
            ::lstrcpy(pDispInfo->item.pszText, string);
            break;

        case 2: // Date and time.
            CTime time(pItem->ftLastWriteTime);

            BOOL pm = FALSE;
            int nHour = time.GetHour();
            if (nHour == 0)
                nHour = 12;
            else if (nHour == 12)
                pm = TRUE;
            else if (nHour > 12) {
```

```
                nHour -= 12;
                pm = TRUE;
            }

            string.Format (_T("%d/%0.2d/%0.2d (%d:%0.2d%c)"),
                time.GetMonth (), time.GetDay (), time.GetYear () % 100,
                nHour, time.GetMinute (), pm ? _T('p') : _T('a'));
            ::lstrcpy (pDispInfo->item.pszText, string);
            break;
        }
    }
    *pResult = 0;
}

void CFileView::OnColumnClick(NMHDR* pNMHDR, LRESULT* pResult)
{
    NM_LISTVIEW* pNMListView = (NM_LISTVIEW*)pNMHDR;
    GetListCtrl().SortItems(CompareFunc, pNMListView->iSubItem);
    *pResult = 0;
}

int CALLBACK CFileView::CompareFunc(LPARAM lParam1, LPARAM lParam2,
    LPARAM lParamSort)
{
    ITEMINFO* pItem1 = (ITEMINFO*)lParam1;
    ITEMINFO* pItem2 = (ITEMINFO*)lParam2;
    int nResult;

    switch(lParamSort) {

    case 0: // File name.
        nResult = pItem1->strFileName.CompareNoCase(pItem2->strFileName);
        break;

    case 1: // File size.
        nResult = pItem1->nFileSizeLow - pItem2->nFileSizeLow;
        break;

    case 2: // Date and time.
        nResult = ::CompareFileTime(&pItem1->ftLastWriteTime,
            &pItem2->ftLastWriteTime);
        break;
    }
    return nResult;
}

void CFileView::OnViewLargeIcons()
{
    ModifyStyle(LVS_TYPEMASK, LVS_ICON);
}

void CFileView::OnViewSmallIcons()
```

```
    ModifyStyle(LVS_TYPEMASK, LVS_SMALLICON);
}

void CFileView::OnViewList()
{
    ModifyStyle(LVS_TYPEMASK, LVS_LIST);
}

void CFileView::OnViewDetails()
{
    ModifyStyle(LVS_TYPEMASK, LVS_REPORT);
}
void CFileView::OnUpdateViewLargeIcons(CCmdUI* pCmdUI)
{
    DWORD dwCurrentStyle = GetStyle() & LVS_TYPEMASK;
    pCmdUI->SetRadio(dwCurrentStyle == LVS_ICON);
}

void CFileView::OnUpdateViewSmallIcons(CCmdUI* pCmdUI)
{
    DWORD dwCurrentStyle = GetStyle() & LVS_TYPEMASK;
    pCmdUI->SetRadio(dwCurrentStyle == LVS_SMALLICON);
}

void CFileView::OnUpdateViewList(CCmdUI* pCmdUI)
{
    DWORD dwCurrentStyle = GetStyle() & LVS_TYPEMASK;
    pCmdUI->SetRadio(dwCurrentStyle == LVS_LIST);
}

void CFileView::OnUpdateViewDetails(CCmdUI* pCmdUI)
{
    DWORD dwCurrentStyle = GetStyle() & LVS_TYPEMASK;
    pCmdUI->SetRadio(dwCurrentStyle == LVS_REPORT);
}

void CFileView::OnFileNewDirectory()
{
    CPathDialog dlg;
    dlg.m_strPath = m_strPath;
    if(dlg.DoModal() == IDOK)
        Refresh(dlg.m_strPath);
}
```

图 10-9　WinDir 应用程序

对于项目和子项目文本，CFileView 使用了回调函数，以便自己可以保存项目的数据而不必强迫控件也保存数据的副本。回调以 LVN_GETDISPINFO 通知的形式出现，该通知由

CFileView 通过 ON_NOTIFY_REFLECT 消息映射表输入项反射给自己的 OnGetDispInfo 处理程序。在 OnGetDispInfo 被调用时,pNMHDR 指向一个 LV_DISPINFO 结构。结构的 item.lParam 字段保存着被查询项目的 ITEMINFO 结构的地址,并且 item.iSubItem 字段保存着所请求的子项目的索引号。CFileView::OnGetDispInfo 将保存在 ITEMINFO 结构的 strFileName、nFileSizeLow 或 ftLastWriteTime 字段中的数据组合为文本字符串,并将结果复制到保存在 LV_DISPINFO 结构的 *item*.pszText 字段内的地址中。列表视图然后就在屏幕上显示出文本了。

CFileView 保存自己的项目数据,以便可以调用 CListCtrl::SortItems,而且 CFileView::CompareFunc 可以通过间接引用保存在项目的 lParam 中的指针来检索任何或全部项目的数据。在列表视图处于报表模式下时,如果用户单击了列头标,那么消息映射表中的 ON_NOTIFY_REFLECT 输入项就会激活 CFileView::OnColumnClick,接下来 OnColumnClick 将调用列表视图的 SortItems 函数,并传递给它被单击的列的索引号:

```
GetListCtrl().SortItems(CompareFunc, pNMListView->iSubItem);
```

CompareFunc 是应用程序定义的排序例程,用来比较项目对。它是回调函数,所以声明为静态。CompareFunc 使用在 lParam1 和 lParam2 中传递的指针来检索有关待比较项目的数据,并使用 lParamSort 中的列索引号来确定使用项目的哪个子项目作为比较的基准。整个函数少于 20 行代码:

```
int CALLBACK CFileView::CompareFunc(LPARAM lParam1, LPARAM lParam2,
    LPARAM lParamSort)
{
    ITEMINFO* pItem1 = (ITEMINFO*) lParam1;
    ITEMINFO* pItem2 = (ITEMINFO*) lParam2;
    int nResult;

    switch(lParamSort) {

    case 0: // File name.
        nResult =
            pItem1->strFileName.CompareNoCase(pItem2->strFileName);
        break;

    case 1: // File size.
        nResult = pItem1->nFileSizeLow - pItem2->nFileSizeLow;
        break;

    case 2: // Date and time.
        nResult = ::CompareFileTime(&pItem1->ftLastWriteTime,
            &pItem2->ftLastWriteTime);
        break;
    }
    return nResult;
}
```

CompareFunc 返回负值表明项目 1 小于项目 2(应该放在前面),返回 0 意味着它们相等,正值说明项目 1 大于项目 2。利用 ::CompareFileTime API 函数可以很容易地比较封装在 FILETIME 值中的日期和时间。还可以根据 FILETIME 值创建 CTime 对象,并使用 <、> 以及其他运算符来比较日期和时间。

可能您还不很明白,但确实您已经看到了带有可排序列的列表视图必须保存自己的数据的原因。CompareFunc 接收到的有关所请求比较项目的唯一信息是项目的 lParam 值。所以,lParam 必须对所有的项目数据提供完整的访问。一种实现的方法就是将项目数据保存在应用程序分配的内存中(以 WinDir 为例,在用 new 分配的 ITEMINFO 结构中),并在每个项目自己的 lParam 中保存指向数据的指针。由自己亲自保存项目数据与将它转换为文本并递交给列表视图相比具有更大的灵活性,因为这样数据可以用二进制形式保存。那么如何给出现在 CFileView 的 Last Modified 列中的信息排序呢?这里字符串排序用处就不大了,"1/1/96"会被排在"9/30/85"的前面,即使前者代表了更靠后的日期。但由于 CFileView 是以 FILETIME 格式保存日期和时间的,所以使得排序非常容易。

对于 CFileView 最后要注意的是删除由 AddItem 分配的 ITEMINFO 结构时所采用的方法。CFileView::FreeItemMemory 通过迭代处理列表视图中的项目来释放为每个项目保留的内存,并利用保存在项目的 lParams 中的指针调用 delete。视图的 WM_DESTROY 处理程序会在应用程序关闭之前调用 FreeItemMemory 来释放 ITEMINFO 结构。

10.4　自制控件视图

CTreeView 和 CListView 是"控件视图"的两个例子,"控件视图"是基本功能从 Windows 控件中得到的视图。这两个都是从 CCtrlView 派生来的,CCtrlView 同时也是 CEditView 和 CRichEditView 的基类。CCtrlView 提供了所有控件视图公用的基本功能。用它作为基类,您可以创建封装了其他 Windows 控件的控件视图。

为便于理解,下列 CCtrlView 派生类定义了一个标签视图,它是一种封装了 Win32 标签控件的简单视图。该视图被显示时,看上去与普通视图相似,就是在顶部有类似于属性页上的标签:

```
class CTabView : public CCtrlView
{
   DECLARE_DYNCREATE (CTabView)
public:
    CTabView () :
        CCtrlView (_T ("SysTabControl32"), AFX_WS_DEFAULT_VIEW) {}
    CTabCtrl& GetTabCtrl () const { return *(CTabCtrl*) this; }
    virtual BOOL PreCreateWindow (CREATESTRUCT& cs);
    virtual void OnInitialUpdate ();
};
```

```
IMPLEMENT_DYNCREATE (CTabView, CCtrlView)

BOOL CTabView::PreCreateWindow (CREATESTRUCT& cs)
{
    ::InitCommonControls ();
    if (! CCtrlView::PreCreateWindow (cs))
        return FALSE;
    cs.style|= TCS_FIXEDWIDTH; // Fixed-width tabs.
    return TRUE;
}

void CTabView::OnInitialUpdate ()
{
    static CString strLabel[] = {
        _T ("Tab No. 1"),
        _T ("Tab No. 2"),
        _T ("Tab No. 3")
    };

    // Set the tab width to 96 pixels.
    GetTabCtrl ().SetItemSize (CSize (96, 0));

    // Add three tabs.
    TC_ITEM item;
    item.mask = TCIF_TEXT;
    for (int i = 0; i < 3; i++) {
        item.pszText = (LPTSTR) (LPCTSTR) strLabel[i];
        item.cchTextMax = strLabel[i].GetLength ();
        GetTabCtrl ().InsertItem (i, &item);
    }
}
```

这个类的主要功能有：默认的构造函数，它给基类的构造函数传递了标签控件的 WNDCLASS 的名字(SysTabControl32)；GetTabCtrl 函数，它返回作为基础的标签控件的引用；以及 OnInitialUpdate，它给控件添加了三个标签。PreCreateWindow 也起了主要作用，它初始化了公用控件类并对控件应用了默认样式。

第 11 章　多文档和多视图

文档/视图应用程序并不局限于一个文档或文档数据的一个视图。使用 MFC 提供的分隔窗口，单文档界面(SDI)应用程序可以在分割了框架窗口客户区的可缩放"窗格"中显示两个以上相同文档的视图。文档/视图体系结构还扩展到了多文档界面(MDI)应用程序，该应用程序支持一个文档的多个视图、多个打开的文档、甚至多个文档类型。虽然 Microsoft 不鼓励使用多文档界面，但是依赖 MDI 模型的应用程序仍旧普遍存在，并且在将来的一段时间内还会出现，不断成功的 Microsoft Word 和其他主要的 Microsoft Windows 应用程序就是证明。

在第 9 章中，学习了如何编写 SDI 文档/视图应用程序。您会发现将该范例程序扩展使其包含多文档和多视图是件很简单的事情。在本章中，首先我们讨论一下 MFC 对 MDI 的支持，会看到很容易就能创建一个 MDI 应用程序。然后，我们将研究在 SDI 应用程序中使用分隔窗口给打开的文档提供多个视图的方法。

11.1　MFC 和多文档界面

从用户的角度出发，有 5 个特性可以用来区分 MDI 应用程序和 SDI 应用程序：

- MDI 应用程序允许用户同时打开两个以上的文档来进行编辑。相反，SDI 应用程序会要求用户在打开另一个文档之前关闭当前文档。
- MDI 应用程序有时支持多个文档类型。例如：把字处理、电子表格和制图程序合在一起的应用程序可能就是 MDI 应用程序，它支持三种文档类型：包含文本的字处理文档、包含电子表格的电子表格文档以及包含图表的制图文档。
- MDI 应用程序具有一个 Window 菜单，其中 New Window 命令用来打开文档的第 2 个视图，菜单中还有排列用来显示视图的窗口的命令。Window 菜单中还包含一个打开视图的列表。从菜单中选中一个视图就可以使该视图成为活动视图，与该视图关联的文档成为活动文档。
- SDI 应用程序通常只有一个菜单。而 MDI 应用程序最少具有两个：其中一个在没有文档打开时显示，另一个在至少一个文档打开时显示。一些 MDI 应用程序具有多于两个的菜单。支持多文档类型的 MDI 应用程序通常对每个文档类型都实现一个菜单。
- SDI 应用程序仅仅使用一个框架窗口(顶层框架窗口)，作为应用程序的主窗口并包含打开文档的视图。MDI 应用程序具有两个：一个顶层框架窗口以及"子框架"或称"文档框架"，它在顶层框架窗口中浮动，用来包含打开文档的视图。

如果没有像 MFC 这样的主结构提供帮助，创建 MDI 应用程序就要比创建 SDI 应用程序

需要更多的投入。例如：在文档被打开、关闭以及在两个状态间切换时更新出现在顶层框架窗口中的菜单就成为了程序开发者的责任。实现 Window 菜单也是程序开发者的责任。创建和管理在顶层框架窗口浮动的文档框架也成了开发者的责任。实际上，所有这些 MDI 用户界面模型的特性功能就变成了许多必须由人完成的讨厌的琐碎工作。

这真是坏消息。但幸运的是 MFC 的文档/视图体系结构抽象了用户界面模型，使得编写 MDI 应用程序与编写 SDI 应用程序只是稍微有些不同。与相应的 SDI 一样，MDI 文档/视图应用程序在基于 CDocument 的文档类中保存数据，在基于 CView 或其派生类的视图对象中呈现数据的视图。用 MFC 创建的 MDI 和 SDI 应用程序结构上的主要区别在于：

- MDI 应用程序从 CMDIFrameWnd 而不是 CFrameWnd 派生它们的顶层框架窗口类。
- MDI 应用程序使用基于 CMDIChildWnd 的类来代表包含文档视图的子框架窗口。
- MDI 应用程序使用 CMultiDocTemplate 而不是 CSingleDocTemplate 来创建文档模板。在 CMultiDocTemplate 的构造函数中引用的框架窗口类是子框架窗口类而不是顶层框架窗口类。
- MDI 应用程序至少具有两个菜单资源，而 SDI 只有一个。其中一个在没有文档打开时显示，另一个在至少有一个文档打开时显示。

这里确实看到了一些不同。在内部，MFC 为特殊的 MDI 杂务投入了大量的程序代码，这些杂务有：动态切换菜单、给打开的文档创建新视图等。简言之，主结构管理了 MDI 应用程序用户界面的几乎所有方面内容，使得您可以不必亲自处理这些杂务了。在很大程度上，那些 MFC 没有为您自动处理的细小工作都由 AppWizard 处理了。如果在 AppWizard 的 Step 1 对话框中选择了 Multiple Documents 而不是 Single Documents(如图 11-1 所示)，AppWizard 将生成一个 MDI 应用程序的结构骨架。从此以后，编写 MDI 应用程序就和编写 SDI 应用程序很像了。只需要编写文档/视图应用程序；剩下的由 MFC 来处理。

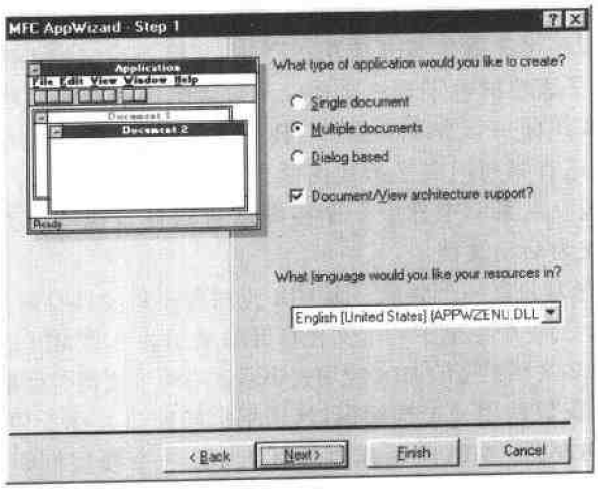

图 11-1 使用 AppWizard 创建 MDI 应用程序

MFC 处理了几乎所有剩余的工作，但是您还不能忘记必须完成一个重要的细节工作。这个"细节工作"就是下一节的标题。

11.1.1　同步文档的多个视图

在选择使用 MDI 用户界面模型时，就隐含地给用户提供了同时显示文档的多个视图的自由。编辑 100 页文档的用户可能会利用应用程序提供的这一功能将第 1 到第 100 页并排起来进行比较。

当从 Window 菜单选择了 New Windows 命令后，MFC 提供的命令处理程序会调出文档模板，提取标识视图类和框架窗口类的 CRuntimeClass 指针，并用它实例化一个新视图和新的框架窗口（是子框架，并非顶层框架）。实际上，第二个视图的地址被添加在了由文档对象维护的视图链接列表中，以便文档可以明了有它的两个独立的视图出现在了屏幕上。如果任一个视图要求重绘，都要调用 GetDocument 来获取指向文档对象的指针，向文档查询需要的数据然后重绘。由于两个视图都连结在同一个文档上（也就是说，对每个视图 GetDocument 都返回相同的指针），所以每一个都享有访问文档数据相同部分的权利。另外，体系结构是可以扩充的：对于成百上千个打开的视图，它也会像对于两个视图一样工作。

到此为止，一切都很好。但让我们考虑一下，如果用户在其中的一个视图编辑了文档，情况会怎样？如果修改在其他视图中是可见的（具有可见的结果），其他视图就应该更新来反映所做的修改。问题是更新操作不会自动执行，它是您的责任，要确保在一个视图中编辑了文档，其他视图（如果存在的话）也要被更新。上结构以 CDocument::UpdateAllViews 和 CView::OnUpdate 的形式提供了实现此任务的机制，在第 9 章曾经提到过。现在到了详细介绍它的时候了。

假设您编写一个程序编辑器，它利用 MDI 体系结构允许用户显示源程序文件的多个视图。如果在一个视图中对文件所做的修改在其他视图中可见，那么这个文件的所有视图都要被更新来反映修改内容。这就是 UpdateAllViews 的功能所在。在多视图应用程序中当文档数据被修改时，或者是产生修改的对象（通常是视图对象），或者是文档对象就要调用 UpdateAllViews来更新视图。UpdateAllViews 迭代处理与文档关联的视图列表，调用每个视图的虚拟 OnUpdate 函数。

CView 提供了一个很小的 OnUpdate 实现，它使视图无效并促使调用 OnDraw。如果您希望整体重绘，那就没必要覆盖 OnUpdate，但是如果希望使更新操作尽可能提高效率，只重绘视图中修改过的部分，那就要在视图类中覆盖 OnUpdate 并且要使用传递给 UpdateAllViews 的提示信息。UpdateAllViews 的原型如下：

```
void UpdateAllViews (CView * pSender, LPARAM lHint = 0L,
    CObject * pHint = NULL);
```

与 OnUpdate 函数原型很相似：

```
virtual void OnUpdate (CView * pSender, LPARAM lHint,
    CObject * pHint)
```

lHint 和 pHint 将提示信息从 UpdateAllViews 带到 OnUpdate。如何使用这些参数取决于应用程序。提示信息的一个简单用途是传递 RECT 结构的地址或是指定视图中需要更新部分的 CRect 对象的地址。OnUpdate 可以在调用 InvalidateRect 中使用此信息,如下:

```
// In the document class
UpdateAllViews (NULL, 1, (CObject * ) pRect);
   .
   .
   .
// In the view class
void CMyView::OnUpdate (CView * pSender, LPARAM lHint, CObject * pHint)
{
    if (lHint == 1) {
        CRect * pRect = (CRect * ) pHint;
        InvalidateRect (pRect);
        return;
    }
    CView::OnUpdate (pSender, lHint, pHint);
}
```

如果文档数据由 CObjects 数组组成,并且由于新的 CObject 添加到了文档中所以要调用 UpdateAllViews,那么可能会用 pHint 来传递新 CObject 的地址。下例中假定 pLine 包含一个指针,该指针指向名为 CLine 的 CObject 派生类实例,还假定 CLine 包含一个公用成员函数 Draw,调用它可以在屏幕上绘制输出 CLine:

```
// In the document class
UpdateAllViews (NULL, 1, pLine);
   .
   .
   .
// In the view class
void CMyView::OnUpdate (CView * pSender, LPARAM lHint, CObject * pHint)
{
    if (lHint == 1) {
        CLine * pLine = (CLine * ) pHint;
        CClientDC dc (this);
        pLine -> Draw (&dc);
        return;
    }
    CView::OnUpdate (pSender, lHint, pHint);
}
```

在这两个例子中，只要 lHint 不是传递给 UpdateAllViews 的特定应用程序的值时，OnUpdate 就把调用提交给基类。这一点很重要，因为 MFC 有时自己会用 lHint 等于 0 来调用 OnUpdate。可以对 lHint 使用任何随意的非零值。甚至可以定义多个“提示集”，给 pHint 赋予不同的含义并使用 lHint 来标识提示类型。

可以使用 UpdateAllViews 的第一个参数 pSender 从更新循环操作中忽略一个视图。如果 pSender 为 NULL，UpdateAllViews 将调用每个视图的 OnUpdate 函数。如果 pSender 是非 NULL 值，UpdateAllViews 将调用除了 pSender 标识的视图以外其他所有视图的 OnUpdate。当文档类中的函数调用 UpdateAllViews 时，通常会把 pSender 设置为 NULL 以便更新所有视图。但是如果视图调用 UpdateAllViews，它可以将 pSender 设置为 this 来防止自己的 OnUpdate 函数被调用。如果视图在响应用户输入时已经更新了自身，就可以不必调用它的 OnUpdate 函数了。但是如果视图没有更新自身，又因为在 OnUpdate 中它执行所有更新工作，就可以给 UpdateAllViews 的第一个参数传递 NULL 值。

下节中的示例程序几乎没怎么使用 UpdateAllViews，没有用提示参数调用它。第二个视图是通过 OnUpdate 的默认实现更新的。在本章稍后，我们将开发一个更实用的多视图应用程序，将提示信息传递给 UpdateAllViews 并在 OnUpdate 中使用该信息。

11.1.2　MdiSquares 应用程序

图 11-2 所示的 MdiSquares 应用程序是第 9 章中 SdiSquares 的 MDI 版本。它使用的文档和视图类与 SdiSquares 中使用的相同，只不过 MdiSquares 的视图类为了节省屏幕空间，绘制的方格要稍微小一些。

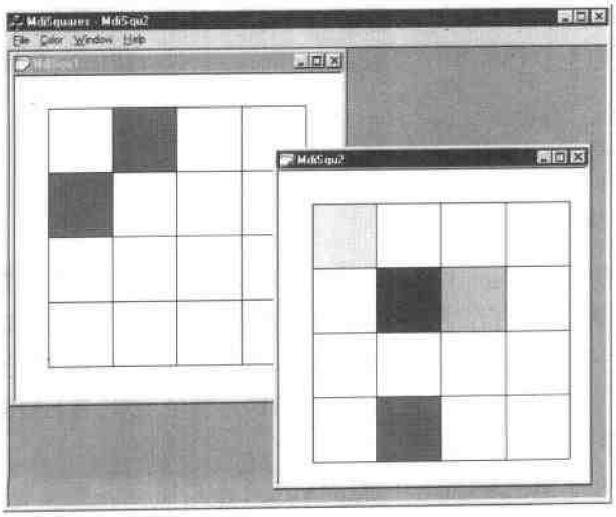

图 11-2　打开了两个文档的 MdiSquares

在运行 MdiSquares 时,第一个文档被自动打开。可以从 File 菜单选择 New 命令来打开其他文档。要打开一个文档的另一个视图,可以从 Windows 菜单选择 New Windows 菜单。如果显示了一个文档的两个视图,观察一下在一个视图中单击方格,方格的颜色会在两个视图中改变。这是因为文档的 SetSquare 函数(视图调用来给方格添加颜色)在将方格的颜色记录到 m_clrGrid 之后会调用 UpdateAllViews。下面给出 SquaresDoc.cpp 中有关的语句:

```
UpdateAllViews (NULL);
```

因为在调用中没有传递提示信息,并且由于 CSquaresView 没有覆盖 OnUpdate,所以每个视图在调用 SetSquare 时都要整个进行重绘。如果您仔细地看,就会发现每次单击方格时视图都会闪烁。闪烁是由于每次调用 UpdateAllViews 时整个视图都被删除和重绘而造成的。

图 11-3 给出了 SquaresDoc.cpp 和其他 MdiSquares 源程序文件。主框架窗口类 CMainFrame,代表应用程序的顶层窗口。视图显示在子框架窗口类 CChildFrame 的实例中。注意在 InitInstance 中,是 CChildFrame 而不是 CMainFrame 在初始化文档模板时被标识为了框架窗口类:

```
CMultiDocTemplate* pDocTemplate;
pDocTemplate = new CMultiDocTemplate(
    IDR_MDISQUTYPE,
    RUNTIME_CLASS(CSquaresDoc),
    RUNTIME_CLASS(CChildFrame), // custom MDI child frame
    RUNTIME_CLASS(CSquaresView));
```

因此,在 MDI 应用程序中调用 ProcessShellCommand 将创建一个新的子框架窗口而不是顶层框架窗口。所以 MDI 应用程序必须在调用 ProcessShellCommand 之前亲自创建顶层框架窗口。创建 MdiSquares 的主窗口的程序可以在 InitInstance 中的其他地方找到:

```
CMainFrame* pMainFrame = new CMainFrame;
if (!pMainFrame->LoadFrame(IDR_MAINFRAME))
    return FALSE;
m_pMainWnd = pMainFrame;
```

此程序以及其他 CMdiSquaresApp、CMainFrame 和 CChildFrame 中的程序都是由 AppWizard 生成的。除非手工编写 MDI 应用程序,否则您的主要工作都集中在文档和视图类中。

如果在 Visual C++ 中打开 MdiSquares,浏览一下它的资源列表,您会发现它包含两个图标、两个菜单以及两个文档字符串。它们的资源 ID 分别是 IDR_MAINFRAME 和 IDR_MDISQUTYPE。下面将介绍这些资源的用途:

- 在顶层窗口的标题栏中显示 IDR_MAINFRAME 图标。在子框架的标题栏中显示 IDR_MDISQUTYPE 图标。如果愿意,可以使用相同的图标,但大多数 MDI 应用程序对文档窗口使用不同的图标。

- 在没有文档打开时显示 IDR_MAINFRAME 菜单。至少一个文档打开时显示 IDR_MDISQUTYPE 菜单。IDR_MAINFRAME 菜单是最小的一种菜单，具有 File 菜单，其中有 New、Open 和 Exit 命令以及最近使用过的文件列表，除此几乎没有其他菜单。另一方面，IDR_MDISQUTYPE 则是一个完整的菜单，包含所有属于 MdiSquares 文档的命令。
- IDR_MAINFRAME 文档字符串只包含了显示在主窗口标题栏上的标题。IDR_MDISQUTYPE 文档字符串包含有关文档类型的所有相关信息，包括默认文件扩展名。

除了本节中讨论的相当小的差别外，MdiSquares 和 SdiSquares 本质上是相同的。这是使用 MFC 的文档/视图体系结构的一个长处：一旦学会了编写 SDI 应用程序，那么也就学会编写 MDI 应用程序了。

MdiSquares.h

```
// MdiSquares.h : main header file for the MDISQUARES application
//

#if !defined(AFX_MDISQUARES_H__36D513DB_9CA0_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MDISQUARES_H__36D513DB_9CA0_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#ifndef __AFXWIN_H__
    #error include 'stdafx.h' before including this file for PCH
#endif

#include "resource.h"       // main symbols

/////////////////////////////////////////////////////////////////////////////
// CMdiSquaresApp:
// See MdiSquares.cpp for the implementation of this class
//

class CMdiSquaresApp : public CWinApp
{
public:
    CMdiSquaresApp();

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMdiSquaresApp)
    public:
```

```
    virtual BOOL InitInstance();
    //{{AFX_VIRTUAL
// Implementation
    //{{AFX_MSG(CMdiSquaresApp)
    afx_msg void OnAppAbout();
      // NOTE - the ClassWizard will add and remove member functions here.
      //    DO NOT EDIT what you see in these blocks of generated code !
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//     AFX_MDISQUARES_H__36D513DB_9CA0_11D2_8E53_006008A82731__INCLUDED_)
```

MdiSquares.cpp

```
// MdiSquares.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "MdiSquares.h"

#include "MainFrm.h"
#include "ChildFrm.h"
#include "SquaresDoc.h"
#include "SquaresView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMdiSquaresApp

BEGIN_MESSAGE_MAP(CMdiSquaresApp, CWinApp)
    //{{AFX_MSG_MAP(CMdiSquaresApp)
    ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
```

```
    //}}AFX_MSG_MAP
    // Standard file based document commands
    ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
    ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMdiSquaresApp construction

CMdiSquaresApp::CMdiSquaresApp()
{
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CMdiSquaresApp object

CMdiSquaresApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CMdiSquaresApp initialization

BOOL CMdiSquaresApp::InitInstance()
{
    SetRegistryKey(_T("Local AppWizard-Generated Applications"));

    LoadStdProfileSettings();  // Load standard INI file
                               // options (including MRU)

    CMultiDocTemplate* pDocTemplate;
    pDocTemplate = new CMultiDocTemplate(
        IDR_MDISQUTYPE,
        RUNTIME_CLASS(CSquaresDoc),
        RUNTIME_CLASS(CChildFrame), // custom MDI child frame
        RUNTIME_CLASS(CSquaresView));
    AddDocTemplate(pDocTemplate);

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

    // Enable drag/drop open
    m_pMainWnd->DragAcceptFiles();
    // Enable DDE Execute open
    EnableShellOpen();
    RegisterShellFileTypes(TRUE);

    // Parse command line for standard shell commands, DDE, file open
    CCommandLineInfo cmdInfo;
```

```
    ParseCommandLine(cmdInfo);

    // Dispatch commands specified on the command line
    if (!ProcessShellCommand(cmdInfo))
        return FALSE;

    // The main window has been initialized, so show and update it.
    pMainFrame->ShowWindow(m_nCmdShow);
    pMainFrame->UpdateWindow();

    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
    CAboutDlg();

// Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    //{{AFX_MSG(CAboutDlg)
        // No message handlers
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
```

```
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
        // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

// App command to run the dialog
void CMdiSquaresApp::OnAppAbout()
{
    CAboutDlg aboutDlg;
    aboutDlg.DoModal();
}

/////////////////////////////////////////////////////////////////////////////
// CMdiSquaresApp message handlers
```

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_MAINFRM_H__36D513DF_9CA0_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__36D513DF_9CA0_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMainFrame : public CMDIFrameWnd
{
    DECLARE_DYNAMIC(CMainFrame)
public:
    CMainFrame();

// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMainFrame)
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
```

```
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

// Generated message map functions
protected:
    //{{AFX_MSG(CMainFrame)
        // NOTE - the ClassWizard will add and remove member functions here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
// AFX_MAINFRM_H__36D513DF_9CA0_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//

#include "stdafx.h"
#include "MdiSquares.h"

#include "MainFrm.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMainFrame

IMPLEMENT_DYNAMIC(CMainFrame, CMDIFrameWnd)
```

```
BEGIN_MESSAGE_MAP(CMainFrame, CMDIFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code !
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
}

CMainFrame::~CMainFrame()
{
}

BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CMDIFrameWnd::PreCreateWindow(cs) )
        return FALSE;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
    CMDIFrameWnd::AssertValid();
}

void CMainFrame::Dump(CDumpContext& dc) const
{
    CMDIFrameWnd::Dump(dc);
}

#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers
```

ChildFrm.h

```
// ChildFrm.h : interface of the CChildFrame class
//
/////////////////////////////////////////////////////////////////////////////
```

```
#if !defined(AFX_CHLDFRM_H__36D513E1_9CA0_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_CHILDFRM_H__36D513E1_9CA0_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif //_MSC_VER > 1000

class CChildFrame : public CMDIChildWnd
{
    DECLARE_DYNCREATE(CChildFrame)
public:
    CChildFrame();

// Attributes
public:

// Operations
public:
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CChildFrame)
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CChildFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

// Generated message map functions
protected:
    //{{AFX_MSG(CChildFrame)
      // NOTE - the ClassWizard will add and remove member functions here.
      //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
```

```
// !defined(
// AFX_CHILDFRM_H__36D513E1_9CA0_11D2_8E53_006008A82731__INCLUDED_)
```

ChildFrm.cpp

```
// ChildFrm.cpp : implementation of the CChildFrame class
//

#include "stdafx.h"
#include "MdiSquares.h"

#include "ChildFrm.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CChildFrame

IMPLEMENT_DYNCREATE(CChildFrame, CMDIChildWnd)

BEGIN_MESSAGE_MAP(CChildFrame, CMDIChildWnd)
    //{{AFX_MSG_MAP(CChildFrame)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CChildFrame construction/destruction

CChildFrame::CChildFrame()
{
}

CChildFrame::~CChildFrame()
{
}

BOOL CChildFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CMDIChildWnd::PreCreateWindow(cs) )
        return FALSE;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
```

```
// CChildFrame diagnostics

#ifdef _DEBUG
void CChildFrame::AssertValid() const
{
    CMDIChildWnd::AssertValid();
}
void CChildFrame::Dump(CDumpContext& dc) const
{
    CMDIChildWnd::Dump(dc);
}

#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CChildFrame message handlers
```

SquaresDoc.h

```
// SquaresDoc.h : interface of the CSquaresDoc class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_SQUARESDOC_H__36D513E3_9CA0_11D2_8E53_006008A82731__IN-
CLUDED_)
#define AFX_SQUARESDOC_H__36D513E3_9CA0_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000


class CSquaresDoc : public CDocument
{
protected: // create from serialization only
    CSquaresDoc();
    DECLARE_DYNCREATE(CSquaresDoc)

// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CSquaresDoc)
    public:
    virtual BOOL OnNewDocument();
```

```
    virtual void Serialize(CArchive& ar);
    //}}AFX_VIRTUAL
// Implementation
public:
    void SetSquare (int i, int j, COLORREF color);
    COLORREF GetSquare (int i, int j);
    COLORREF GetCurrentColor();
    virtual ~CSquaresDoc();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    COLORREF m_clrCurrentColor;
    COLORREF m_clrGrid[4][4];
    //{{AFX_MSG(CSquaresDoc)
    afx_msg void OnColorRed();
    afx_msg void OnColorYellow();
    afx_msg void OnColorGreen();
    afx_msg void OnColorCyan();
    afx_msg void OnColorBlue();
    afx_msg void OnColorWhite();
    afx_msg void OnUpdateColorRed(CCmdUI* pCmdUI);
    afx_msg void OnUpdateColorYellow(CCmdUI* pCmdUI);
    afx_msg void OnUpdateColorGreen(CCmdUI* pCmdUI);
    afx_msg void OnUpdateColorCyan(CCmdUI* pCmdUI);
    afx_msg void OnUpdateColorBlue(CCmdUI* pCmdUI);
    afx_msg void OnUpdateColorWhite(CCmdUI* pCmdUI);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//     AFX_SQUARESDOC_H__36D513E3_9CA0_11D2_8E53_006008A82731__INCLUDED_)
```

SquaresDoc.cpp

```
// SquaresDoc.cpp : implementation of the CSquaresDoc class
//

#include "stdafx.h"
#include "MdiSquares.h"

#include "SquaresDoc.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSquaresDoc

IMPLEMENT_DYNCREATE(CSquaresDoc, CDocument)

BEGIN_MESSAGE_MAP(CSquaresDoc, CDocument)
    //{{AFX_MSG_MAP(CSquaresDoc)
    ON_COMMAND(ID_COLOR_RED, OnColorRed)
    ON_COMMAND(ID_COLOR_YELLOW, OnColorYellow)
    ON_COMMAND(ID_COLOR_GREEN, OnColorGreen)
    ON_COMMAND(ID_COLOR_CYAN, OnColorCyan)
    ON_COMMAND(ID_COLOR_BLUE, OnColorBlue)
    ON_COMMAND(ID_COLOR_WHITE, OnColorWhite)
    ON_UPDATE_COMMAND_UI(ID_COLOR_RED, OnUpdateColorRed)
    ON_UPDATE_COMMAND_UI(ID_COLOR_YELLOW, OnUpdateColorYellow)
    ON_UPDATE_COMMAND_UI(ID_COLOR_GREEN, OnUpdateColorGreen)
    ON_UPDATE_COMMAND_UI(ID_COLOR_CYAN, OnUpdateColorCyan)
    ON_UPDATE_COMMAND_UI(ID_COLOR_BLUE, OnUpdateColorBlue)
    ON_UPDATE_COMMAND_UI(ID_COLOR_WHITE, OnUpdateColorWhite)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSquaresDoc construction/destruction

CSquaresDoc::CSquaresDoc()
{
}
CSquaresDoc::~CSquaresDoc()
{
}

BOOL CSquaresDoc::OnNewDocument()
{
```

```
    if (!CDocument::OnNewDocument())
        return FALSE;

    for (int i = 0; i < 4; i++)
        for (int j = 0; j < 4; j++)
            m_clrGrid[i][j] = RGB(255, 255, 255);

    m_clrCurrentColor = RGB(255, 0, 0);
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CSquaresDoc serialization

void CSquaresDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        for (int i = 0; i < 4; i++)
            for (int j = 0; j < 4; j++)
                ar << m_clrGrid[i][j];
        ar << m_clrCurrentColor;
    }
    else
    {
        for (int i = 0; i < 4; i++)
            for (int j = 0; j < 4; j++)
                ar >> m_clrGrid[i][j];
        ar >> m_clrCurrentColor;
    }
}

/////////////////////////////////////////////////////////////////////////////
// CSquaresDoc diagnostics

#ifdef _DEBUG
void CSquaresDoc::AssertValid() const
{
    CDocument::AssertValid();
}

void CSquaresDoc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CSquaresDoc commands
```

```
COLORREF CSquaresDoc::GetCurrentColor()
{
    return m_clrCurrentColor;
}

COLORREF CSquaresDoc::GetSquare(int i, int j)
{
    ASSERT(i >= 0 && i <= 3 && j >= 0 && j <= 3);
    return m_clrGrid[i][j];
}

void CSquaresDoc::SetSquare(int i, int j, COLORREF color)
{
    ASSERT(i >= 0 && i <= 3 && j >= 0 && j <= 3);
    m_clrGrid[i][j] = color;
    SetModifiedFlag(TRUE);
    UpdateAllViews(NULL);
}

void CSquaresDoc::OnColorRed()
{
    m_clrCurrentColor = RGB(255, 0, 0);
}

void CSquaresDoc::OnColorYellow()
{
    m_clrCurrentColor = RGB(255, 255, 0);
}
void CSquaresDoc::OnColorGreen()
{
    m_clrCurrentColor = RGB(0, 255, 0);
}

void CSquaresDoc::OnColorCyan()
{
    m_clrCurrentColor = RGB(0, 255, 255);
}

void CSquaresDoc::OnColorBlue()
{
    m_clrCurrentColor = RGB(0, 0, 255);
}

void CSquaresDoc::OnColorWhite()
{
    m_clrCurrentColor = RGB(255, 255, 255);
}

void CSquaresDoc::OnUpdateColorRed(CCmdUI * pCmdUI)
```

```
}
    pCmdUI -> SetRadio (m_clrCurrentColor == RGB (255, 0, 0));
}

void CSquaresDoc::OnUpdateColorYellow(CCmdUI * pCmdUI)
{
    pCmdUI -> SetRadio (m_clrCurrentColor == RGB (255, 255, 0));
}

void CSquaresDoc::OnUpdateColorGreen(CCmdUI * pCmdUI)
{
    pCmdUI -> SetRadio (m_clrCurrentColor == RGB (0, 255, 0));
}

void CSquaresDoc::OnUpdateColorCyan(CCmdUI * pCmdUI)
{
    pCmdUI -> SetRadio (m_clrCurrentColor == RGB (0, 255, 255));


void CSquaresDoc::OnUpdateColorBlue(CCmdUI * pCmdUI)

    pCmdUI -> SetRadio (m_clrCurrentColor == RGB (0, 0, 255));
}
void CSquaresDoc::OnUpdateColorWhite(CCmdUI * pCmdUI)
{
    pCmdUI -> SetRadio (m_clrCurrentColor == RGB (255, 255, 255));
}
```

SquaresView.h

```
// SquaresView.h : interface of the CSquaresView class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_SQUARESVIEW_H__36D513E5_9CA0_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_SQUARESVIEW_H__36D513E5_9CA0_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000


class CSquaresView : public CView
{
protected: // create from serialization only
```

```
    CSquaresView();
    DECLARE_DYNCREATE(CSquaresView)

// Attributes
public:
    CSquaresDoc* GetDocument();

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CSquaresView)
    public:
    virtual void OnDraw(CDC* pDC);  // overridden to draw this view
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    //}}AFX_VIRTUAL
// Implementation
public:
    virtual ~CSquaresView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    //{{AFX_MSG(CSquaresView)
    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

#ifndef _DEBUG  // debug version in SquaresView.cpp
inline CSquaresDoc* CSquaresView::GetDocument()
   { return (CSquaresDoc*)m_pDocument; }
#endif

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
```

```
// !defined(
//     AFX_SQUARESVIEW_H__36D513E5_9CA0_11D2_8E53_006008A82731__INCLUDED_)
```

SquaresView.cpp

```
// SquaresView.cpp : implementation of the CSquaresView class
//

#include "stdafx.h"
#include "MdiSquares.h"

#include "SquaresDoc.h"
#include "SquaresView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSquaresView

IMPLEMENT_DYNCREATE(CSquaresView, CView)

BEGIN_MESSAGE_MAP(CSquaresView, CView)
    //{{AFX_MSG_MAP(CSquaresView)
    ON_WM_LBUTTONDOWN()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSquaresView construction/destruction

CSquaresView::CSquaresView()
{
}

CSquaresView::~CSquaresView()
{
}

BOOL CSquaresView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CView::PreCreateWindow(cs);
}

/////////////////////////////////////////////////////////////////////////////
// CSquaresView drawing

void CSquaresView::OnDraw(CDC* pDC)
```

```
{
    CSquaresDoc * pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    //
    // Set the mapping mode to MM_LOENGLISH.
    //
    pDC -> SetMapMode (MM_LOENGLISH);
    //
    // Draw the 16 squares.
    //
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            COLORREF color = pDoc -> GetSquare (i, j);
            CBrush brush (color);
            int x1 = (j * 70) + 35;
            int y1 = (i * 70) - 35;
            int x2 = x1 + 70;
            int y2 = y1 - 70;
            CRect rect (x1, y1, x2, y2);
            pDC -> FillRect (rect, &brush);
        }
    }

    //
    // Then draw the grid lines surrounding them.
    //
    for (int x = 35; x <= 315; x += 70) {
        pDC -> MoveTo (x, -35);
        pDC -> LineTo (x, -315);
    }

    for (int y = -35; y >= -315; y -= 70) {
        pDC -> MoveTo (35, y);
        pDC -> LineTo (315, y);
    }
}

////////////////////////////////////////////////////////////////////////////
// CSquaresView diagnostics

#ifdef _DEBUG
void CSquaresView::AssertValid() const
{
    CView::AssertValid();
}
```

```
void CSquaresView::Dump(CDumpContext& dc) const
{
    CView::Dump(dc);
}

CSquaresDoc* CSquaresView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CSquaresDoc)));
    return (CSquaresDoc*)m_pDocument;
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CSquaresView message handlers

void CSquaresView::OnLButtonDown(UINT nFlags, CPoint point)
{
    CView::OnLButtonDown(nFlags, point);

    //
    // Convert click coordinates to MM_LOENGLISH units.
    //
    CClientDC dc(this);
    dc.SetMapMode(MM_LOENGLISH);
    CPoint pos = point;
    dc.DPtoLP(&pos);

    //
    // If a square was clicked, set its color to the current color.
    //
    if (pos.x >= 35 && pos.x <= 315 && pos.y <= -35 && pos.y >= -315) {
        int i = (-pos.y - 35) / 70;
        int j = (pos.x - 35) / 70;
        CSquaresDoc* pDoc = GetDocument();
        COLORREF clrCurrentColor = pDoc->GetCurrentColor();
        pDoc->SetSquare(i, j, clrCurrentColor);
    }
}
```

图 11-3　MdiSquares 应用程序

11.1.3　支持多个文档类型

用 MFC 编写的 MDI 应用程序在默认状态下支持多个文档实例。每次在用户执行 File/

New 命令时都会创建一个新的文档实例。MDI 应用程序还可以支持多个文档“类型”,每一个的特性都由文档模板来定义。

假设要给 MdiSquares 添加第 2 个文档类型,如圆环文档,当 File/New 被选中时,会给用户提供一个选项来确定是创建方格文档还是创建圆环文档。以下给出具体实现方法。

1. 派生一个新文档类和一个新视图类作为新文档类型。对于本示例程序,假定类名为 CCirclesDoc 和 CCirclesView。像 AppWizard 生成的文档和视图类那样,使它们可以被动态创建。
2. 为圆环文档在项目中添加 4 个新资源:图标、菜单、加速键(可选)和文档字符串。赋给 4 个资源相同的资源 ID,例如:IDR_CIRCLETYPE。
3. 修改 InitInstance 来创建新的文档模板,其中包含资源 ID 和为文档、视图以及框架窗口类设置的 CRuntimeClass 指针。然后调用 AddDocTemplate 并传递文档模板对象的地址。

下列是 InitInstance 函数中的一个片段,修改后用来注册两个文档模板:

```
// AppWizard-generated code
CMultiDocTemplate* pDocTemplate;
pDocTemplate = new CMultiDocTemplate(
    IDR_MDISQUTYPE,
    RUNTIME_CLASS(CSquaresDoc),
    RUNTIME_CLASS(CChildFrame), // custom MDI child frame
    RUNTIME_CLASS(CSquaresView));
AddDocTemplate(pDocTemplate);

// Your code
pDocTemplate = new CMultiDocTemplate(
    IDR_CIRCLETYPE,
    RUNTIME_CLASS(CCirclesDoc),
    RUNTIME_CLASS(CChildFrame),
    RUNTIME_CLASS(CCirclesView));
AddDocTemplate(pDocTemplate);
```

这就是所有基本内容。本例使用了 CChildFrame 作为两个文档类型的子框架类,但是如果愿意可以派生独立的子框架类。

在用此方法注册了多文档类型之后,MFC 的 File/New 命令处理程序会显示一个对话框,其中给用户提供了文档类型的选项。在对话框中标识每个文档类型的字符串来自文档字符串,具体地讲,是来自文档字符串中七个可能存在的子字符串中的第三个。有了这样的基础结构,编写允许用户创建和编辑不同种类文档的多功能 MDI 应用程序就相当简单了。也可以编写支持两个以上文档类型的 SDI 应用程序,但是多文档范例很少用在单文档应用程序中。

11.1.4 MDI之外的其他选择

如果想给用户在应用程序实例中提供同时编辑几个文档的能力，那么多文档界面并不是您唯一的选择。The Windows Interface Guidelines for Software Design列出了除MDI程序设计模型以外的3种可选模型：

- 基于工作区的模型，将相关的文档组合在称为"工作区"的对象中，并允许包含在工作区中的文档可以在类似MDI文档窗口（是顶层框架窗口的子窗口）中被查看和编辑。Visual C++就是这样的应用程序的一个例子，使用了工作区包含模型。
- 工作手册模型，每个视图都占据顶层框架窗口的整个客户区，但任何时候只有一个视图可见。外观与MDI应用程序中的最大化文档框架相似。每个视图都做了一个标签，用户可以通过单击按钮而在视图间切换，视图好像是属性页中的页面一样。
- 项目模型，将相关的文档组合在一个项目中，但只允许各个文档在类似SDI的框架窗口中编辑。项目模型、MDI以及工作区模型三者之间主要的区别在于，在项目模型中没有提供顶层框架窗口来包含文档框架。

MFC并不直接支持任何一种选项，但您总是可以编程来实现。可选用户界面模型正处于Microsoft的MFC工作组监视之下，很有可能在未来的MFC版本中支持用户界面模型而不是SDI和MDI。

11.2 拆分窗口

MDI应用程序本身就支持一个文档多个视图；而SDI应用程序并不这样。对于SDI应用程序，呈现一个文档具有两个以上同步视图的最好方法是使用基于MFC CSplitterWnd类的拆分窗口。拆分窗口是这样的窗口，能够使用可移动的拆分条把它沿水平方向、垂直方向，或在两个方向上分成两个以上的窗格。每个窗格都包含文档数据的一个视图。视图是拆分窗口的子窗口，通常拆分窗口又是框架窗口的子窗口。在SDI应用程序中，拆分窗口是顶层框架窗口的子窗口。而在MDI应用程序中，拆分窗口是MDI文档框架的子窗口。位于拆分窗口内的视图可以使用CView::GetParentFrame来获得指向其父框架窗口的指针。

MFC支持两种类型的拆分窗口：静态的和动态的。静态拆分窗口的行列数在拆分窗口被创建时就设置好了，用户不能更改。但是用户可以缩放各行各列。一个静态拆分窗口最多可以包含16行16列。要找一个使用了静态拆分窗口的应用程序，只要看一下Windows资源管理器即可，资源管理器的主窗口在垂直方向上被静态拆分窗口分成了两半。

动态拆分窗口最多可以有两行两列，但它们可以相互拆分或合并。显示在动态拆分窗口中的视图并不是彼此完全独立的：动态拆分窗口在水平方向上被拆分后，两行窗口具有各自独立的垂直滚动条但却公用一个水平滚动条。与此类似，垂直方向拆分动态拆分窗口

得到的两列窗口也具有各自的水平滚动条而共享同一个垂直滚动条。动态拆分窗口的最大行列数是在创建拆分窗口时指定的。因此,可以很容易地创建一个可以在水平或垂直方向而不能同时在两个方向拆分的动态拆分窗口。Visual C++ 就使用了动态拆分窗口使得可以同时编辑源程序文件的两个以上不同部分(参见图 11-4)。

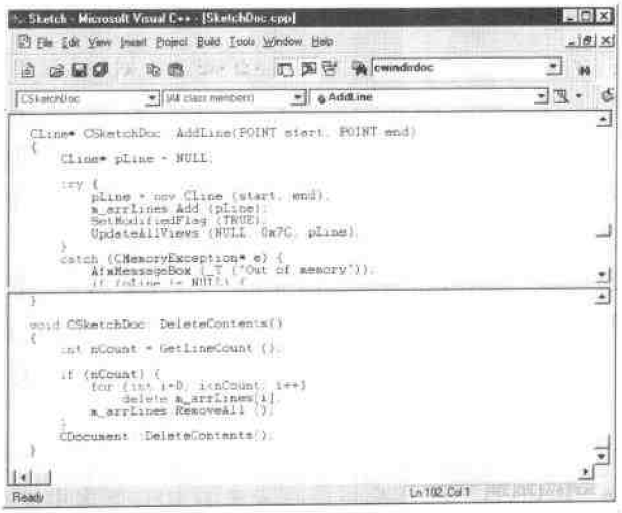

图 11-4 Visual C++ 中显示文档两个视图的动态拆分窗口

选择静态或动态拆分窗口的一个准则是是否希望用户能够交互式地修改拆分窗口的行列配置。如果希望就选用动态拆分窗口。另一个决定因素是计划在拆分窗口中使用的视图种类。在静态拆分窗口中很容易使用两个以上不同种类的视图,因为您可以在每个窗格中指定所用的视图类型。但是在动态拆分窗口中,MFC 管理着视图,除非从 CSplitterWnd 派生一个新类并修改拆分窗口的默认操作功能,否则拆分窗口中的所有视图使用的都是相同的视图类。

11.2.1 动态拆分窗口

动态拆分窗口是用 MFC 的 CSplitterWnd::Create 函数创建的。创建和初始化动态拆分窗口只有简单的两步过程:

1. 给框架窗口类添加一个 CSplitterWnd 数据成员。
2. 覆盖框架窗口的虚拟 OnCreateClient 函数,并调用 CSplitterWnd::Create 在框架窗口的客户区创建动态拆分窗口。

假设 m_wndSplitter 是一个 CSplitterWnd 对象,该对象是框架窗口类 CMainFrame 的一个成员,下列 OnCreateClient 覆盖函数将在框架窗口内部创建动态拆分窗口:

```
BOOL CMainFrame::OnCreateClient(LPCREATESTRUCT pcs,
    CCreateContext* pContext)
{
    return m_wndSplitter.Create(this, 2, 2, CSize(2, 2), pContext);
}
```

CSplitterWnd::Create 中的第一个参数标识了拆分窗口的父窗口，它是框架窗口。第二和第三个参数指定了窗口能够拆分的最大行列数。由于动态拆分窗口所支持的最大值是两行两列，所以这两个参数值不是 1 就是 2。第四个参数以像素为单位指定了每个窗格的最小宽度和最小高度。主结构使用这些值来确定在拆分条移动过程中何时创建或销毁窗格。CSize 的值为(1,1)指定了窗格可以小到 1 像素宽 1 像素高。第五个参数是指向主结构提供的 CCreateContext 结构的指针。结构中 m_pNewViewClass 成员标识了用来创建拆分窗格中视图的视图类。主结构创建一个最初的视图并把它应用到第一个窗格中。其他相同类的视图在别的窗格被生成时会自动地创建。

CSplitterWnd::Create 支持可选的第六和第七个参数，分别指定拆分窗口的样式和其子窗口 ID。在大多数实例中，默认值就很好。默认的子窗口 ID AFX_IDW_PANE_FIRST 是一个很妙的数值，用它可以使框架窗口标识与其关联的拆分窗口。只有在已经包含拆分窗口的框架窗口中创建第二个拆分窗口时才需要修改此 ID。

一旦创建了动态拆分窗口，主结构就提供逻辑途径来使它工作。例如：窗口最初没有拆分，用户将垂直拆分条拖至窗口中间之后，MFC 就会在垂直方向上拆分窗口并在新窗格中生成新视图。由于新视图是在运行时创建的，所以视图类必须支持动态创建机制。如果以后用户将垂直拆分条拖至窗口的左边或右边(或者足够地接近一条边使得任一个窗格的宽度小于创建拆分窗口时指定的最小宽度)，MFC 就会销毁第二个窗格以及其中的视图。

CSplitterWnd 类包含了许多有用的成员函数，可以调用来向拆分窗口查询一些相关信息。首先，可以查询当前显示的窗口的行列数，查询一行或一列的高度或宽度，或者查询指向特定行列窗格中视图的 CView 指针。如果想在应用程序的菜单中添加 Split 命令，就要一个 ID 值为 ID_WINDOW_SPLIT 的菜单项。此 ID 已经预先设置在了 CView 消息映射表中的命令处理程序 CView::OnSplitCmd 和更新处理程序 CView::OnUpdateSplitCmd 中了。在内部，CView::OnSplitCmd 会调用 CSplitterWnd::DoKeyboardSplit 开始跟踪处理，允许使用箭头键上下移动幻影拆分条。在 Enter 键按下时跟踪结束、接受新的拆分位置，或在 Esc 键被按下时取消操作。

11.2.2　Sketch 应用程序

图 11-5 所示的应用程序是一个画草图的应用程序，用它可以绘制简单的线段图形。要画一条线段，按下鼠标左键并在按着的同时拖动鼠标。释放鼠标左键时就会有实线替换跟随光标移动的橡皮筋线。View 菜单中的 Grid 命令用来切换捕获状态。在捕获功能有效时，

线段端点会自动捕获最近的栅格点。

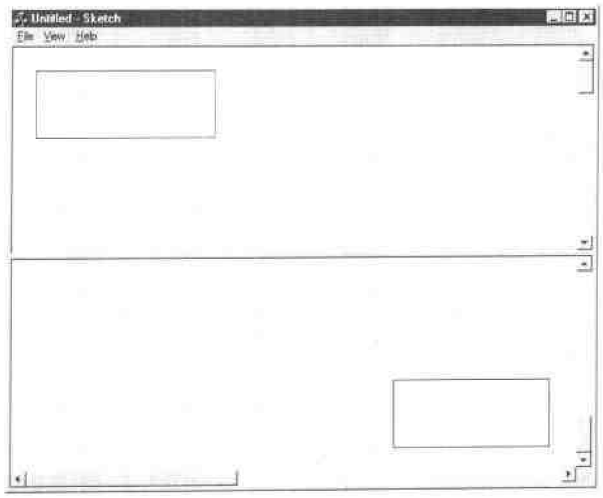

图 11-5 由动态拆分窗口拆分的 Sketch 窗口

在图 11-6 中给出了 Sketch 的源程序代码。值得注意的是,Sketch 是一个标准的 SDI 文档/视图应用程序。用户所画的线段由 CLine 的实例代表,其中包含保存线段端点的 CPoint 成员变量和在屏幕上绘制线段的 Draw 函数。文档对象在基于 MFC CTypedPtrArray 类的动态数组中保存指向 CLine 对象的指针。每次在屏幕上绘制一条线段时,视图会调用文档的 AddLine函数并给其传递线段的端点,视图为确保每个 WM_LBUTTONDOWN 消息都伴有 WM_LBUTTONUP消息而采用了鼠标捕获功能。

AddLine 接下去将根据这些端点来创建新的 CLine 对象并在数组中记录 CLine 的地址:

```
// In SketchDoc.h
typedef CTypedPtrArray < CObArray, CLine * > CLineArray;
  .
  .
  .
CLineArray m_arrLines;

// In SketchDoc.cpp
CLine * CSketchDoc::AddLine(POINT from, POINT to)
{
    CLine * pLine = NULL;

    try {
        pLine = new CLine (from, to);
        m_arrLines.Add (pLine);
        SetModifiedFlag (TRUE);
        UpdateAllViews (NULL, 0x7C, pLine);
```

```
    }
    catch (CMemoryException* e)
    {
        AfxMessageBox(_T("Out of memory"));
        if (pLine != NULL)
        {
            delete pLine;
            pLine = NULL;
        }
        e->Delete();
    }
    return pLine;
}
```

由于 CLine 是可串行化的类，并且由于 CTypedPtrArray 能够通过简单的函数调用串行化所有的可串行化的元素，所以 CSketchDoc::Serialize 中的一条语句就保存或加载了用户所绘的每一条线段：

```
m_arrLines.Serialize(ar);
```

CSketchDoc 还覆盖了 DeleteContents 并使用它在当前文档释放之前删除所有用 AddLine 创建的 CLine 对象。如果没有按此方式处理，CLines 将会在每次关闭文档时导致内存泄漏。

将 Sketch 与那些乏味的 SDI 文档/视图应用程序分开的一个原因是它使用了动态拆分窗口。在 CMainFrame::OnCreateClient 中拆分窗口是用以下语句创建的：

```
return m_wndSplitter.Create(this, 2, 2, CSize(8, 8), pContext);
```

值得注意的是，这是 Sketch 中唯一一处明确地为拆分窗口服务的代码；MFC 处理拆分操作的所有其他方面。

显示在拆分窗口中文档的并发视图必须像 MDI 应用程序中并发视图那样取得同步。如果窗口被拆分了，在绘制线段时，在 CSketchDoc::AddLine 中调用 UpdateAllViews 确保两个视图都被更新。没有依赖 OnUpdate 的默认实现，CSketchView 覆盖了 OnUpdate 并依靠传递给 UpdateAllViews 的提示信息执行“智能更新”操作。每当一条线添加给文档时，AddLine 都会明确地调用 UpdateAllViews 并在 pHint 中传递一个代表新线段的 CLine 指针：

```
UpdateAllViews(NULL, 0x7C, pLine);
```

视图的 OnUpdate 函数把 pHint 强制转换回 CLine，并要求 CLine 在屏幕上绘制自身：

```
void CSketchView::OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)
{
    if (lHint == 0x7C)
    {
        CLine* pLine = (CLine*) pHint;
        ASSERT(pLine->IsKindOf(RUNTIME_CLASS(CLine)));
        CClientDC dc(this);
        OnPrepareDC(&dc);
```

```
        pLine -> Draw (&dc);
        return;
    }
    CScroll View::OnUpdate (pSender, lHint, pHint);
}
```

由于无论文档中保存了多少线段,更新视图只涉及绘制一条新的线段,所以此方法比用 OnDraw 重绘整个视图效率要高得多。结果是 Sketch 中也不存在影响 MdiSquares 的闪烁效果了。

Sketch.h

```
// Sketch.h : main header file for the SKETCH application
//

#if !defined(AFX_SKETCH_H__1260AFC5_9CAC_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_SKETCH_H__1260AFC5_9CAC_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#ifndef __AFXWIN_H__
    #error include 'stdafx.h' before including this file for PCH
#endif

#include "resource.h"       // main symbols

/////////////////////////////////////////////////////////////////////////////
// CSketchApp:
// See Sketch.cpp for the implementation of this class
//

class CSketchApp : public CWinApp
{
public:
    CSketchApp();

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CSketchApp)
    public:
    virtual BOOL InitInstance();
    //}}AFX_VIRTUAL

// Implementation
    //{{AFX_MSG(CSketchApp)
    afx_msg void OnAppAbout();
```

```
        // NOTE - the ClassWizard will add and remove member functions here.
        //    DO NOT EDIT what you see in these blocks of generated code !
    //{{AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(AFX_SKETCH_H__1260AFC5_9CAC_11D2_8E53_006008A82731__INCLUDED_)
```

Sketch.cpp

```
// Sketch.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "Line.h"
#include "Sketch.h"
#include "MainFrm.h"
#include "SketchDoc.h"
#include "SketchView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSketchApp

BEGIN_MESSAGE_MAP(CSketchApp, CWinApp)
    //{{AFX_MSG_MAP(CSketchApp)
    ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
//}}AFX_MSG_MAP
    // Standard file based document commands
    ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
    ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
```

```
// CSketchApp construction

CSketchApp::CSketchApp()
{
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CSketchApp object

CSketchApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CSketchApp initialization

BOOL CSketchApp::InitInstance()
{
    SetRegistryKey(_T("Local AppWizard-Generated Applications"));

    LoadStdProfileSettings();  // Load standard INI file
                               // options (including MRU)

    CSingleDocTemplate* pDocTemplate;
    pDocTemplate = new CSingleDocTemplate(
        IDR_MAINFRAME,
        RUNTIME_CLASS(CSketchDoc),
        RUNTIME_CLASS(CMainFrame),       // main SDI frame window
        RUNTIME_CLASS(CSketchView));
    AddDocTemplate(pDocTemplate);

    // Enable DDE Execute open
    EnableShellOpen();
    RegisterShellFileTypes(TRUE);

    // Parse command line for standard shell commands, DDE, file open
    CCommandLineInfo cmdInfo;
    ParseCommandLine(cmdInfo);
    // Dispatch commands specified on the command line
    if (! ProcessShellCommand(cmdInfo))
        return FALSE;

    // The one and only window has been initialized, so show and update it.
    m_pMainWnd->ShowWindow(SW_SHOW);
    m_pMainWnd->UpdateWindow();

    // Enable drag/drop open
    m_pMainWnd->DragAcceptFiles();

    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
```

```
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
    CAboutDlg();

// Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    //{{AFX_MSG(CAboutDlg)
        // No message handlers
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
        // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

// App command to run the dialog
void CSketchApp::OnAppAbout()
{
    CAboutDlg aboutDlg;
```

```
    aboutDlg.DoModal();
}

/////////////////////////////////////////////////////////////////////////////
// CSketchApp message handlers
```

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_MAINFRM_H__1260AFC9_9CAC_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__1260AFC9_9CAC_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMainFrame : public CFrameWnd
{

protected: // create from serialization only
    CMainFrame();
    DECLARE_DYNCREATE(CMainFrame)
// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMainFrame)
    public:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    virtual BOOL OnCreateClient(LPCREATESTRUCT lpcs,
        CCreateContext* pContext);
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
```

```
#endif

// Generated message map functions
protected:
    CSplitterWnd m_wndSplitter;
    //{{AFX_MSG(CMainFrame)
      // NOTE - the ClassWizard will add and remove member functions here.
      //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(AFX_MAINFRM_H__1260AFC9_9CAC_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//

#include "stdafx.h"
#include "Sketch.h"

#include "MainFrm.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMainFrame

IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
```

```
/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
}

CMainFrame::~CMainFrame()
{
}

BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CFrameWnd::PreCreateWindow(cs) )
        return FALSE;
    return TRUE;
}
/////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
    CFrameWnd::AssertValid();
}

void CMainFrame::Dump(CDumpContext& dc) const
{
    CFrameWnd::Dump(dc);
}

#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers


BOOL CMainFrame::OnCreateClient(LPCREATESTRUCT lpcs, CCreateContext* pContext)
{
    return m_wndSplitter.Create(this, 2, 1, CSize(8, 8), pContext);
}
```

SketchDoc.h

```
// SketchDoc.h : interface of the CSketchDoc class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_SKETCHDOC_H__1260AFCB_9CAC_11D2_8E53_006008A82731__IN-
```

```
CLUDED )
#define AFX_SKETCHDOC_H__1260AFCB_9CAC_11D2_8E53_006008A82731__INCLUDED

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

typedef CTypedPtrArray < CObArray, CLine * > CLineArray;
class CSketchDoc : public CDocument
{
protected: // create from serialization only
    CSketchDoc();
    DECLARE_DYNCREATE(CSketchDoc)

// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CSketchDoc)
    public:
    virtual BOOL OnNewDocument();
    virtual void Serialize(CArchive& ar);
    virtual void DeleteContents();
    //}}AFX_VIRTUAL

// Implementation
public:
    CLine * GetLine (int nIndex);
    int GetLineCount ();
    CLine * AddLine (POINT from, POINT to);
    BOOL IsGridVisible ();
    virtual ~CSketchDoc();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    CLineArray m_arrLines;
    BOOL m_bShowGrid;
    //{{AFX_MSG(CSketchDoc)
    afx_msg void OnViewGrid();
```

```
    afx_msg void OnUpdateViewGrid(CCmdUI * pCmdUI);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//     AFX_SKETCHDOC_H__1260AFCB_9CAC_11D2_8E53_006008A82731__INCLUDED_)
```

SketchDoc.cpp

```
// SketchDoc.cpp : implementation of the CSketchDoc class
//

#include "stdafx.h"
#include "Line.h"
#include "Sketch.h"
#include "SketchDoc.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSketchDoc

IMPLEMENT_DYNCREATE(CSketchDoc, CDocument)

BEGIN_MESSAGE_MAP(CSketchDoc, CDocument)
    //{{AFX_MSG_MAP(CSketchDoc)
    ON_COMMAND(ID_VIEW_GRID, OnViewGrid)
    ON_UPDATE_COMMAND_UI(ID_VIEW_GRID, OnUpdateViewGrid)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSketchDoc construction/destruction

CSketchDoc::CSketchDoc()

{
}

CSketchDoc::~CSketchDoc()
```

```
}

BOOL CSketchDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;

    m_bShowGrid = TRUE;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CSketchDoc serialization

void CSketchDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        ar << m_bShowGrid;
    }
    else
    {
        ar >> m_bShowGrid;
    }
    m_arrLines.Serialize(ar);
}

/////////////////////////////////////////////////////////////////////////////
// CSketchDoc diagnostics

#ifdef _DEBUG
void CSketchDoc::AssertValid() const
{
    CDocument::AssertValid();
}

void CSketchDoc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CSketchDoc commands

BOOL CSketchDoc::IsGridVisible()
{
    return m_bShowGrid;
}
```

```
void CSketchDoc::OnViewGrid()
{
    if (m_bShowGrid)
        m_bShowGrid = FALSE;
    else
        m_bShowGrid = TRUE;

    SetModifiedFlag (TRUE);
    UpdateAllViews (NULL);
}

void CSketchDoc::OnUpdateViewGrid(CCmdUI * pCmdUI)
{
    pCmdUI -> SetCheck (m_bShowGrid);
}

CLine * CSketchDoc::AddLine(POINT from, POINT to)
{
    CLine * pLine = NULL;

    try {
        pLine = new CLine (from, to);
        m_arrLines.Add (pLine);
        SetModifiedFlag (TRUE);
        UpdateAllViews (NULL, 0xFC, pLine);
    }
    catch (CMemoryException * e)
        AfxMessageBox (_T ("Out of memory"));
        if (pLine != NULL) {
            delete pLine;
            pLine = NULL;
        }
        e -> Delete ();
    }
    return pLine;
}
int CSketchDoc::GetLineCount()
{
    return m_arrLines.GetSize ();
}

CLine * CSketchDoc::GetLine(int nIndex)
{
    ASSERT (nIndex < GetLineCount ());
    return m_arrLines[nIndex];
}

void CSketchDoc::DeleteContents()
```

```
{
    int nCount = GetLineCount();
    if (nCount) {
        for (int i = 0; i < nCount; i++)
            delete m_arrLines[i];
        m_arrLines.RemoveAll();
    }
    CDocument::DeleteContents();
}
```

SketchView.h

```
// SketchView.h : interface of the CSketchView class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_SKETCHVIEW_H__1260AFCD_9CAC_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_SKETCHVIEW_H__1260AFCD_9CAC_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000


class CSketchView : public CScrollView
{
protected: // create from serialization only
    CSketchView();
    DECLARE_DYNCREATE(CSketchView)
// Attributes
public:
    CSketchDoc* GetDocument();

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CSketchView)
    public:
    virtual void OnDraw(CDC* pDC);  // overridden to draw this view
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    virtual void OnInitialUpdate(); // called first time after construct
    virtual void OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint);
```

```
    //{{AFX_VIRTUAL

// Implementation
public:
    virtual ~CSketchView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    void InvertLine (CDC* pDC, POINT from, POINT to);
    CPoint m_ptFrom;
    CPoint m_ptTo;
    HCURSOR m_hCursor;
    //{{AFX_MSG(CSketchView)
    afx_msg BOOL OnSetCursor(CWnd* pWnd, UINT nHitTest, UINT message);
    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
    afx_msg void OnMouseMove(UINT nFlags, CPoint point);
    afx_msg void OnLButtonUp(UINT nFlags, CPoint point);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in SketchView.cpp
inline CSketchDoc* CSketchView::GetDocument()
   { return (CSketchDoc*)m_pDocument; }
#endif

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//     AFX_SKETCHVIEW_H__1260AFCD_9CAC_11D2_8E53_006008A82731__INCLUDED_)
```

SketchView.cpp

```
// SketchView.cpp : implementation of the CSketchView class
//

#include "stdafx.h"
#include "Line.h"
```

```
#include "Sketch.h"
#include "SketchDoc.h"
#include "SketchView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSketchView

IMPLEMENT_DYNCREATE(CSketchView, CScrollView)

BEGIN_MESSAGE_MAP(CSketchView, CScrollView)
    //{{AFX_MSG_MAP(CSketchView)
    ON_WM_SETCURSOR()
    ON_WM_LBUTTONDOWN()
    ON_WM_MOUSEMOVE()
    ON_WM_LBUTTONUP()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSketchView construction/destruction

CSketchView::CSketchView()
{
    m_hCursor = AfxGetApp()->LoadStandardCursor(IDC_CROSS);
}

CSketchView::~CSketchView()
{
}

BOOL CSketchView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CScrollView::PreCreateWindow(cs);
}

/////////////////////////////////////////////////////////////////////////////
// CSketchView drawing

void CSketchView::OnDraw(CDC* pDC)
{
    CSketchDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    //
```

```
    // Draw the snap grid.
    //
    if (pDoc -> IsGridVisible ()) {
        for (int x = 25; x < 1600; x += 25)
            for (int y = - 25; y > - 1200; y - = 25)
                pDC -> SetPixel (x, y, RGB (128, 128, 128));
    }

    //
    // Draw the lines.
    //
    int nCount = pDoc -> GetLineCount ();
    if (nCount) {
        for (int i = 0; i < nCount; i ++ )
            pDoc -> GetLine (i) -> Draw (pDC);
    }
}

void CSketchView::OnInitialUpdate()
{
    CScrollView::OnInitialUpdate();
    SetScrollSizes(MM_LOENGLISH, CSize (1600, 1200));
}

////////////////////////////////////////////////////////////////////////////
// CSketchView diagnostics

#ifdef _DEBUG
void CSketchView::AssertValid() const
{
    CScrollView::AssertValid();
}

void CSketchView::Dump(CDumpContext& dc) const
{
    CScrollView::Dump(dc);
}

CSketchDoc * CSketchView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument -> IsKindOf(RUNTIME_CLASS(CSketchDoc)));
    return (CSketchDoc * )m_pDocument;
}
#endif //_DEBUG

////////////////////////////////////////////////////////////////////////////
// CSketchView message handlers

BOOL CSketchView::OnSetCursor(CWnd * pWnd, UINT nHitTest, UINT message)
```

```
}
    ::SetCursor (m_hCursor);
    return TRUE;
}

void CSketchView::OnLButtonDown(UINT nFlags, CPoint point)
{
    CScrollView::OnLButtonDown(nFlags, point);

    CPoint pos = point;
    CClientDC dc(this);
    OnPrepareDC(&dc);
    dc.DPtoLP(&pos);

    if (GetDocument()->IsGridVisible()) {
        pos.x = ((pos.x + 12) / 25) * 25;
        pos.y = ((pos.y + 12) / 25) * 25;
    }

    m_ptFrom = pos;
    m_ptTo = pos;
    SetCapture();
}

void CSketchView::OnMouseMove(UINT nFlags, CPoint point)
{
    CScrollView::OnMouseMove(nFlags, point);

    if (GetCapture() == this) {
        CPoint pos = point;
        CClientDC dc(this);
        OnPrepareDC(&dc);
        dc.DPtoLP(&pos);

        if (GetDocument()->IsGridVisible()) {
            pos.x = ((pos.x + 12) / 25) * 25;
            pos.y = ((pos.y + 12) / 25) * 25;
        }

        if (m_ptTo != pos)
            InvertLine(&dc, m_ptFrom, m_ptTo);
            InvertLine(&dc, m_ptFrom, pos);
            m_ptTo = pos;
        }
    }


void CSketchView::OnLButtonUp(UINT nFlags, CPoint point)
{
```

```
    CScrollView::OnLButtonUp(nFlags, point);

    if (GetCapture() == this)
        ::ReleaseCapture();

        CPoint pos = point;
        CClientDC dc(this);
        OnPrepareDC(&dc);
        dc.DPtoLP(&pos);

        if (GetDocument()->IsGridVisible())
            pos.x = ((pos.x - 12) / 25) * 25;
            pos.y = ((pos.y - 12) / 25) * 25;

        InvertLine(&dc, m_ptFrom, m_ptTo);

        CSketchDoc* pDoc = GetDocument();
        CLine* pLine = pDoc->AddLine(m_ptFrom, m_ptTo);
    }
}

void CSketchView::InvertLine(CDC* pDC, POINT from, POINT to)
{
    int nOldMode = pDC->SetROP2(R2_NOT);
    pDC->MoveTo(from);
    pDC->LineTo(to);
    pDC->SetROP2(nOldMode);
}

void CSketchView::OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)
{
    if (lHint == 0x7C)
        CLine* pLine = (CLine*) pHint;
        ASSERT(pLine->IsKindOf(RUNTIME_CLASS(CLine)));
        CClientDC dc(this);
        OnPrepareDC(&dc);
        pLine->Draw(&dc);
        return;
    }
    CScrollView::OnUpdate(pSender, lHint, pHint);
}
```

图 11-6 Sketch 应用程序

11.2.3 静态拆分窗口

对静态拆分窗口的处理与对动态拆分窗口的处理很相似,只是在创建它时要另外多进

行一步操作。静态拆分窗口是用 CSplitterWnd::CreateStatic 而不是用 CSplitterWnd::Create 创建的，并且由于 MFC 不会自动创建静态拆分窗口中显示的视图，所以您要亲自在 CreateStatic 返回之后创建视图。CSplitterWnd 为此提供了名为 CreateView 的函数。给框架窗口添加静态拆分窗口的过程如下：

1. 给框架窗口类添加一个 CSplitterWnd 数据成员。
2. 覆盖框架窗口的 OnCreateClient 函数，并调用 CSplitterWnd::CreateStatic 来创建静态拆分窗口。
3. 使用 CSplitterWnd::CreateView 在每个静态拆分窗口的窗格中创建视图。

使用静态拆分窗口的一个优点是由于您自己给窗格添加视图，所以可以控制放入视图的种类。下例中创建的静态拆分窗口包含了两种不同的视图：

```
BOOL CMainFrame::OnCreateClient (LPCREATESTRUCT lpcs,
    CCreateContext * pContext)
{
    if (!m_wndSplitter.CreateStatic (this, 1, 2)||
        !m_wndSplitter.CreateView (0, 0, RUNTIME_CLASS (CTextView),
            CSize (128, 0), pContext)||
        !m_wndSplitter.CreateView (0, 1, RUNTIME_CLASS (CPictureView),
            CSize (0, 0), pContext))
        return FALSE;

    return TRUE;
}
```

传递给 CreateStatic 的参数指定了拆分窗口的父亲以及拆分窗口包含的行列数。对每个窗格调用一次 CreateView。用从 0 开始的行列编号来标识窗格。在此例中，第一次调用 CreateView在左窗格(0 行 0 列)中加入类型为 CTextView 的视图，第二次调用在右窗格(0 行 1 列)加入类型为 CPictureView 的视图。视图并不是直接实例化得到的，而是由 MFC 创建的。所以，要将指向 CreateView 的指针而不是指向已存在的 CView 对象的指针传递给 CRuntimeClass。和动态拆分窗口一样，静态拆分窗口中的视图也必须进行动态创建，否则主结构不能使用它们。

传递给 CreateView 的 CSize 对象指定了窗格的初始尺寸。在本例中，CTextView 窗格的初始宽度为 128 像素，CPictureView 窗格将占据剩余的窗口宽度。指定右窗格宽度的值和指定两个窗格高度的值都是 0，这是因为主结构会忽略它们。如果拆分窗口仅包含一行，则无论指定的 CSize 值是多少，这一行都会具有父窗口的整个客户区高度。与此相似，如果拆分窗口包含 n 列，最右一列将占据第 n－1 列右边和其窗口边界之间的空间。

11.2.4　Wanderer 应用程序

图 11-7 所示的 Wanderer 应用程序使用了静态拆分窗口来模拟 Windows 资源管理器的

外观和功能。拆分窗口将框架窗口分为两个窗格。左窗格包含 CDriveView,它是一个 CTreeView类,被自定义来显示宿主 PC 机的目录结构。右窗格包含 CFileView,它是一个 CListView 类,用来列出在 CDriveView 中所选目录下的文件。

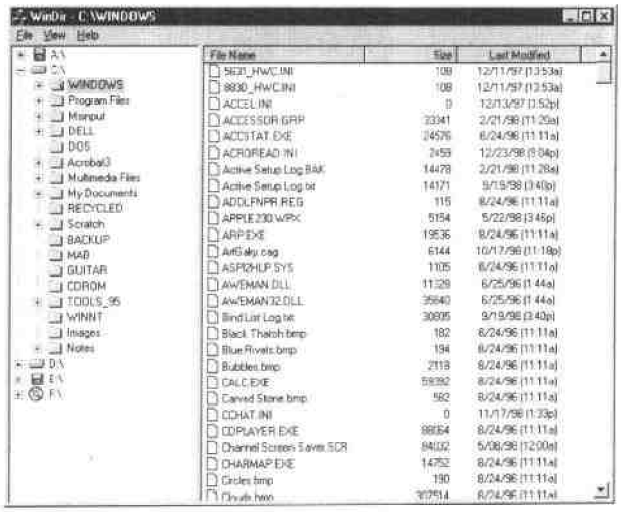

图 11-7 由静态拆分窗口分开的 Wanderer 窗口

Wanderer 使用的 CDriveView 和 CFileView 类与第 10 章介绍的同名类几乎完全相同。我对 CDriveView 稍微作了一些修改,对被反射的 TVN_SELCHANGED 通知添加了一个处理程序,该通知表明树形视图选项被修改。处理程序将选中的项目转换为路径名,并使用 UpdateAllViews的 pHint 参数将路径名传送给 CFileView。

```
void CDriveView::OnSelectionChanged(NMHDR * pNMHDR, LRESULT * pResult)
{
    NM_TREEVIEW * pNMTreeView = (NM_TREEVIEW * ) pNMHDR;
    CString strPath = GetPathFromItem(pNMTreeView->itemNew.hItem);
    GetDocument()->UpdateAllViews(this, 0x5A,
    (CObject * ) (LPCTSTR) strPath);
    * pResult = 0;
}
```

我还修改了 CFileView,如果 lHint 等于 0x5A 就显示 pHint 所指定目录中的内容从而响应对 OnUpdate 的调用:

```
void CFileView::OnUpdate(CView * pSender, LPARAM lHint, CObject * pHint)
{
    if (lHint == 0x5A) {
        FreeItemMemory();
```

```
        GetListCtrl().DeleteAllItems();
        Refresh((LPCTSTR) pHint);
        return;
    }
    CListView::OnUpdate(pSender, lHint, pHint);
}
```

这两个地方的修改一同将左右窗格结合在了一起，只要左边窗格中所选目录项有变动，右边视图就会被更新。

在 CMainFrame::OnCreateClient 中创建并初始化了静态拆分窗口。在拆分窗口创建之后，OnCreateClient 用 CreateView 将 CDriveView 放置在左窗格中，将 CFileView 放在右窗格中（参见图 11-8）。Wanderer 中对 OnCreateClient 的实现只有一点不同寻常之处，它首先创建的是右边的视图然后才是左边的视图。原因很简单。CDriveView 的 OnInitialUpdate 函数调用 UpdateAllViews 来告诉 CFileView 选中了哪个目录，接着，CFileView 的 OnUpdate 函数显示该目录下的文件。但是如果先创建 CDriveView，那么在调用 CDriveView::OnInitialUpdate 时 CFileView 还不存在。首先创建 CFileView 是解决此问题的一种方法。

Wanderer.h

```
// Wanderer.h : main header file for the WANDERER application
//

#if !defined(AFX_WANDERER_H__AEC6FFA_9B0F_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_WANDERER_H__AEC6FFA_9B0F_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#ifndef __AFXWIN_H__
    #error include 'stdafx.h' before including this file for PCH
#endif
#include "resource.h"       // main symbols

/////////////////////////////////////////////////////////////////////////////
// CWandererApp:
// See Wanderer.cpp for the implementation of this class
//

class CWandererApp : public CWinApp
{
public:
    CWandererApp();
```

```
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CWandererApp)
    public:
    virtual BOOL InitInstance();
    //}}AFX_VIRTUAL

// Implementation
    //{{AFX_MSG(CWandererApp)
    afx_msg void OnAppAbout();
      // NOTE - the ClassWizard will add and remove member functions here.
      //    DO NOT EDIT what you see in these blocks of generated code !
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(
//     AFX_WANDERER_H__AE0A6FFA_9B0F_11D2_8E53_006008A82731__INCLUDED_)
```

Wanderer.cpp

```
// Wanderer.cpp : Defines the class behaviors for the application.
//
#include "stdafx.h"
#include "Wanderer.h"

#include "MainFrm.h"
#include "WandererDoc.h"
#include "DriveView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CWandererApp

BEGIN_MESSAGE_MAP(CWandererApp, CWinApp)
```

```
    //{{AFX_MSG_MAP(CWandererApp)
    ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
    // Standard file based document commands
    ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
    ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CWandererApp construction

CWandererApp::CWandererApp()
{
    // TODO: add construction code here,
    // Place all significant initialization in InitInstance


/////////////////////////////////////////////////////////////////////////////
// The one and only CWandererApp object

CWandererApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CWandererApp initialization

BOOL CWandererApp::InitInstance()

    // Standard initialization
    // If you are not using these features and wish to reduce the size
    //  of your final executable, you should remove from the following
    //  the specific initialization routines you do not need.

    // Change the registry key under which our settings are stored.
    // TODO: You should modify this string to be something appropriate
    // such as the name of your company or organization.
    SetRegistryKey(_T("Local AppWizard-Generated Applications"));

    LoadStdProfileSettings();  // Load standard INI file
                               // options (including MRU)

    // Register the application's document templates.  Document templates
    //  serve as the connection between documents, frame windows and views.

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

```
        RUNTIME_CLASS(CDriveView));
    AddDocTemplate(pDocTemplate);

    // Parse command line for standard shell commands, DDE, file open
    CCommandLineInfo cmdInfo;
    ParseCommandLine(cmdInfo);

    // Dispatch commands specified on the command line
    if (!ProcessShellCommand(cmdInfo))
        return FALSE;

    // The one and only window has been initialized, so show and update it.
    m_pMainWnd->ShowWindow(SW_SHOW);
    m_pMainWnd->UpdateWindow();

    return TRUE;
}


/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
    CAboutDlg();
// Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    //{{AFX_MSG(CAboutDlg)
        // No message handlers
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
```

```
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
        // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

// App command to run the dialog
void CWandererApp::OnAppAbout()
{
    CAboutDlg aboutDlg;
    aboutDlg.DoModal();
}

/////////////////////////////////////////////////////////////////////////////
// CWandererApp message handlers
```

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_MAINFRM_H__AF0A6FF5_9B0F_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__AF0A6FF5_9B0F_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMainFrame : public CFrameWnd
{

protected: // create from serialization only
    CMainFrame();
    DECLARE_DYNCREATE(CMainFrame)

// Attributes
public:
```

```
// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMainFrame)
    public:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    virtual BOOL OnCmdMsg(UINT nID, int nCode, void* pExtra,
        AFX_CMDHANDLERINFO* pHandlerInfo);
    protected:
    virtual BOOL OnCreateClient(LPCREATESTRUCT lpcs,
        CCreateContext* pContext);
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
// Generated message map functions
protected:
    CSplitterWnd m_wndSplitter;
    //{{AFX_MSG(CMainFrame)
      // NOTE - the ClassWizard will add and remove member functions here.
      //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.

#endif
// !defined(AFX_MAINFRM_H__AF0A6FFE_980F_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//

#include "stdafx.h"
```

```
#include "Wanderer.h"
#include "WandererDoc.h"
#include "DriveView.h"
#include "FileView.h"
#include "MainFrm.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMainFrame

IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
}

CMainFrame::~CMainFrame()
{
}

BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CFrameWnd::PreCreateWindow(cs) )
        return FALSE;

    cs.style &= ~FWS_ADDTOTITLE;
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
```