Trường Đại học Kỹ Thuật Công nghệ
Khoa Công nghệ Thông tin

GIÁO TRÌNH MÔN HỌC

LẬP TRÌNH WINDOWS

VỚI VC/MFC

Biên soạn: Nguyễn Chánh Thành

Tháng 03 năm 2006

TÀI LIỆU THAM KHẢO

- Sách:
  - Các sách tiếng Việt về Visual C++ /lập trình Windows (của SAMIS, của nhóm tác giả ELICOM, hay của các tác giả khác)
  - Sách tiếng Anh:
    - Beginning Visual C++ 6
    - Professional Visual C++ 6 (của nhà xuất bản WROX)
  - Các eBook tiếng Anh về Visual C++ hay lập trình Windows như:
    - Programming Microsoft C++, 5th Edition eBook (của Microsoft Press)
    - Programming Windows with MFC, 2nd Edition eBook (của Microsoft Press)
- Chương trình tham khảo:
  - MSDN (bộ đĩa CD tài liệu tham khảo của Mircosoft)
  - Source code mẫu ở website:
    - http://www.wrox.com
  - Các ví dụ đặc biệt ở website:
    - http://www.codeguru.com
    - http://www.codeproject.com

CHƯƠNG 0. ÔN TẬP LÝ THUYẾT C/C++

0.1 Ôn tập C

0.1.1 Kiểu dữ liệu, biến và chuyển đổi kiểu

0.2 Hàm và lời gọi hàm

0.2.1 Phát biểu điều khiển

0.2.2 Array

0.2.3 Pointer

0.2.4 File

0.2.5 Debug – bẫy lỗi

0.3 Ôn tập C++

0.3.1 Class

0.3.2 Cấu trúc thừa kế

0.3.3 Tầm vực truy xuất

0.3.4 Object

CHƯƠNG 1. CÁC VẤN ĐỀ CƠ BẢN CỦA ỨNG DỤNG WINDOWS VÀ MFC

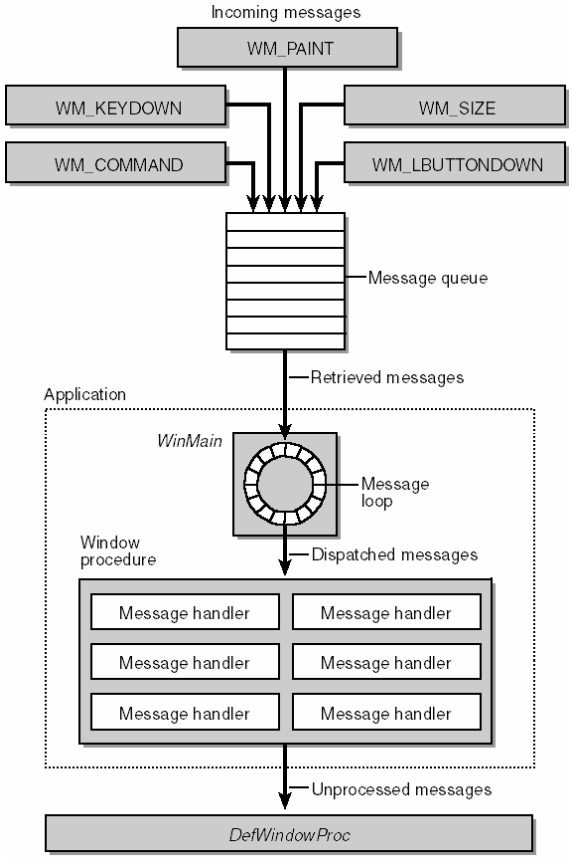
1.1 GIỚI THIỆU KHUNG ỨNG DỤNG WINDOWS (WINDOWS APPLICATION) VÀ XÂY DỰNG CHƯƠNG TRÌNH MẪU VỚI MFC APP FRAMEWORK

1.1.1 Lập trình Windows

Lập trình Windows là kỹ thuật lập trình sử dụng các hàm Windows API để xây dựng các trình ứng dụng trong Windows (*Window App*) và các dạng ứng dụng khác như DLL, ActiveX, …Tuy là kỹ thuật lập trình mạnh mẽ nhưng đòi hỏi tính chuyên nghiệp cao của lập trình viên, giải quyết kế thừa kém, khó phát triển nhanh một ứng dụng.

1.1.2 Mô hình lập trình Windows

Kỹ thuật lập trình sử dụng các hàm Windows API còn gọi là lập trình Windows SDK. Một ứng dụng xây dựng theo kỹ thuật này chứa đựng hàm WinMain (xử lý các thông báo (*message*) nhận được/gửi đi nhằm đáp ứng yêu cầu tương tác của người dùng và của hệ thống cũng như của ứng dụng khác) và hàm DefWinProc (điều phối hoạt động tương ứng với các thông báo nhận được). Tổ chức hệ thống của ứng dụng Windows dạng SDK như sau:



Ví dụ:

```
#include <windows.h>

LONG WINAPI WndProc(HWND, UINT, WPARAM, LPARAM);
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
    LPSTR lpszCmdLine, int nCmdShow)
{
    WNDCLASS wc;
    HWND hwnd;
    MSG msg;
```

```
    wc.style = 0;                           // Class style
    wc.lpfnWndProc = (WNDPROC) WndProc;     // Window procedure address
    wc.cbClsExtra = 0;                      // Class extra bytes
    wc.cbWndExtra = 0;                      // Window extra bytes
    wc.hInstance = hInstance;               // Instance handle
    wc.hIcon = LoadIcon(NULL, IDI_WINLOGO);         // Icon handle
    wc.hCursor = LoadCursor(NULL, IDC_ARROW);       // Cursor handle
    wc.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1); // Background color
    wc.lpszMenuName = NULL;                          // Menu name
    wc.lpszClassName = "MyWndClass";                 // WNDCLASS name
    RegisterClass(&wc);

    hwnd = CreateWindow(
        "MyWndClass",               // WNDCLASS name
        "SDK Application",          // Window title
        WS_OVERLAPPEDWINDOW,        // Window style
        CW_USEDEFAULT,              // Horizontal position
        CW_USEDEFAULT,              // Vertical position
        CW_USEDEFAULT,              // Initial width
        CW_USEDEFAULT,              // Initial height
        HWND_DESKTOP,               // Handle of parent window
        NULL,                       // Menu handle
        hInstance,                  // Application's instance handle
        NULL                        // Window-creation data
    );

    ShowWindow(hwnd, nCmdShow);
    UpdateWindow(hwnd);

    while(GetMessage(&msg, NULL, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return msg.wParam;
}
LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam,
    LPARAM lParam)
{
    PAINTSTRUCT ps;
    HDC hdc;
    switch(message) {
    case WM_PAINT:
        hdc = BeginPaint(hwnd, &ps);
        Ellipse(hdc, 0, 0, 200, 100);
        EndPaint(hwnd, &ps);
        return 0;

    case WM_DESTROY:
        PostQuitMessage(0);
        return 0;
    }
    return DefWindowProc(hwnd, message, wParam, lParam);
}
```

1.1.3 Lập trình Windows với MFC

Lập trình Windows với MFC là kỹ thuật lập trình sử dụng bộ thư viện MFC của Microsoft để xây dựng các trình ứng dụng trong Windows (*Window App*) và các dạng ứng dụng khác như DLL, COM, ActiveX …

MFC (*Microsoft Foundation Classes*) là thư viện cơ sở chứa các lớp (*class*) C++ do Microsoft cung cấp nhằm đặt một trình bao bọc cho Windows API tạo sự thuận lợi cao cho người dùng trong việc phát triển ứng dụng. Ngoài ra, MFC còn cung cấp kiến trúc View/Document giúp định nghĩa cấu trúc chương trình và cấu trúc tài liệu cho trình ứng dụng đơn giản, uyển chuyển và dễ phát triển hơn. Do đó MFC còn được xem là một khung ứng dụng (*application framework*)

Việc hỗ trợ lớp thừa kế và các hàm AFX cũng như các lớp tiện ích của MFC giúp người dùng thuận tiện hơn việc phát triển ứng dụng tạo nhanh các điều khiển (*control*) trong Windows và truy xuất chúng nhanh chóng và dễ dàng.

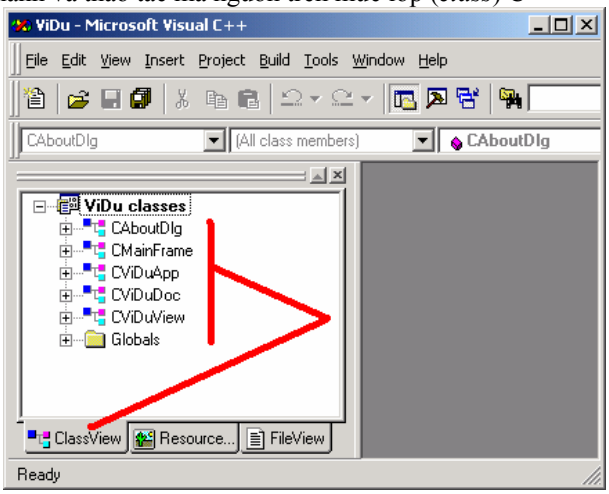
1.1.4 Môi trường lập trình MS Visual C++

Môi trường lập trình Visual C++ bao gồm:

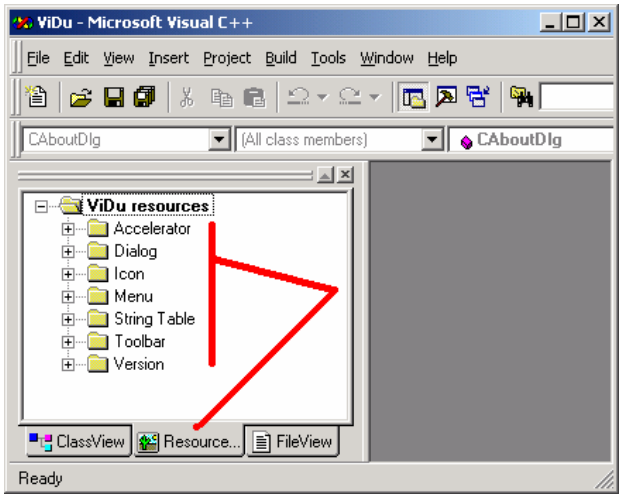
1.1.4.1 Miền làm việc

Khi khởi động lần đầu tiên, vùng bên trái Developer Studion được gọi là miền làm việc, đây chính là vùng để điều hành các phần khác nhau của các dự án pháp triển (*project*). Miền làm việc này cho phép xem các phần của ứng dụng theo ba các khác nhau (như các hình dưới đây):

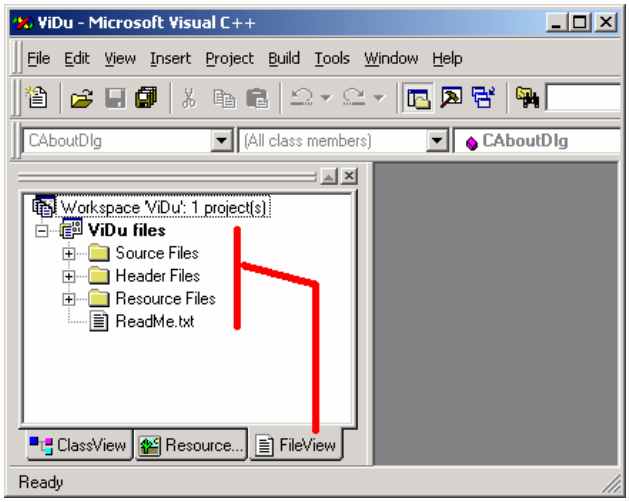
Class View: cho phép điều hành và thao tác mã nguồn trên mức lớp (*class*) C++



Resource View: cho phép tìm và chọn lọc các tài nguyên khác nhau trong ứng dụng như thiết kế cửa sổ hội thoại, biểu tượng, menu, toolbar...

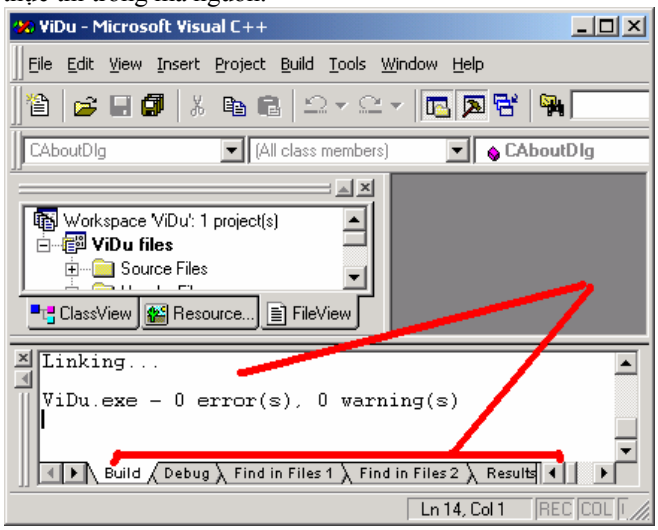


File View: cho phép xem và điều hành tất cả các file trong ứng dụng.



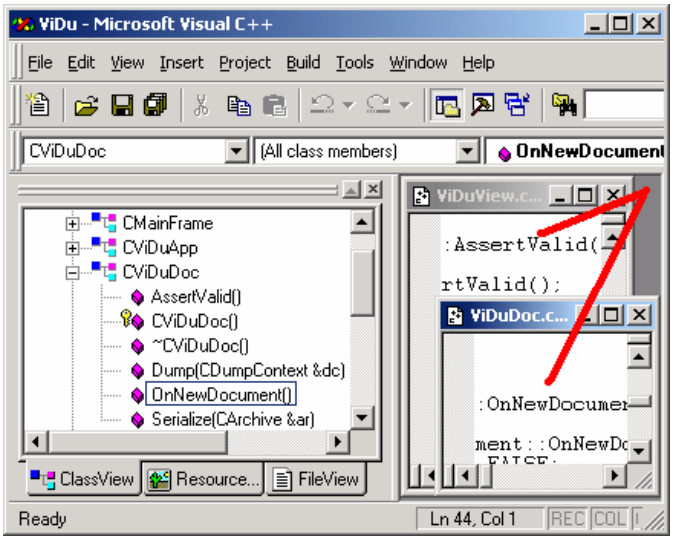
1.1.4.2 Cửa sổ xuất (output pane)

Cửa sổ này nằm ở phần dưới cùng trong cửa sổ ứng Visual C++, thường có thể không hiện trên màn hình khi khởi động ứng dụng Visual C++ lần đầu tiên mà sẽ xuất hiện sau khi thực hiện biên dịch ứng dụng lần đầu tiên. Phần cửa sổ này là nơi cung cấp tất cả thông tin cần thiết cho người dùng như: các câu lệnh, lời cảnh báo và thông báo lỗi của trình biên dịch, đồng thời là nơi chương trình gỡ rối hiển thị tất cả các iến với những giá trị hiện hành trong thời gian thực thi trong mã nguồn.



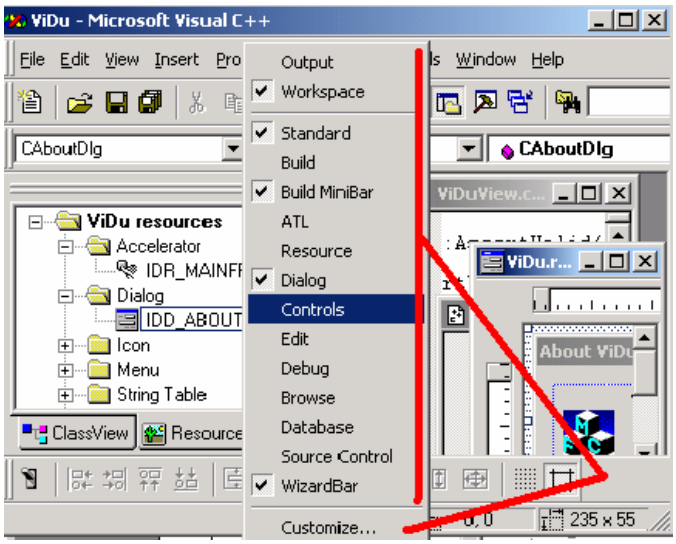
1.1.4.3 Vùng soạn thảo

Đây là vùng bên phải của môi trường để người dùng thực hiện tất cả thao tác soạn thảo chương trình khi sử dụng Visual C++, nơi các cửa sổ soạn thảo chương trình hiển thị, đồng thời là nơi cửa sổ vẽ hiển thị khi người dùng thiết kế hộp thoại.

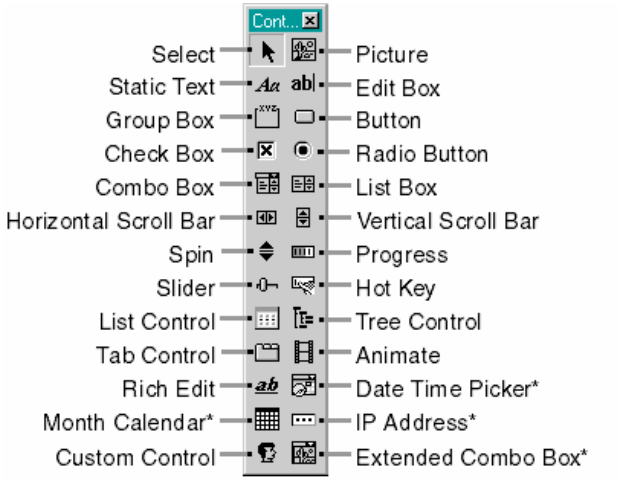


1.1.4.4 Thanh thực đơn (menu)

Lần đầu tiên chạy Visual C++, có ba thanh công cụ hiển thị ngay dưới thanh menu (*menu bar*). Trong Visual C++ có sẵn nhiều thanh công cụ khác nhau, người dùng có thể tùy biến tạo các thanh công cụ phù hợp nhất cho riêng mình.



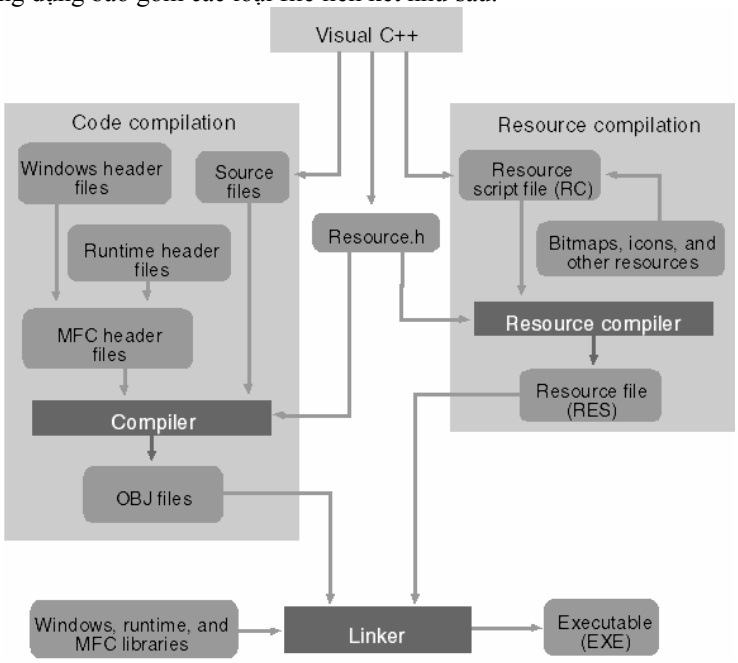
1.1.4.5 Thanh công cụ



*Indicates a new Internet Explorer 4 common control introduced in Visual C++ 6.0. These are covered in Chapter 9.

1.1.5 Các thành phần của ứng dụng phát triển với MS Visual C++

Các thành phần của ứng dụng bao gồm các loại file liên kết như sau:



Các loại file liên quan đến ứng dụng VC++:

Phần mở rộng	Diễn giải
APS	Supports ResourceView
BSC	Browser information file
CLW	Supports ClassWizard
DEP	Dependency file
DSP	Project file*
DSW	Workspace file*
MAK	External makefile
NCB	Supports ClassView
OPT	Holds workspace configuration
PLG	Builds log file

Ví dụ tổng hợp:

Hello.h

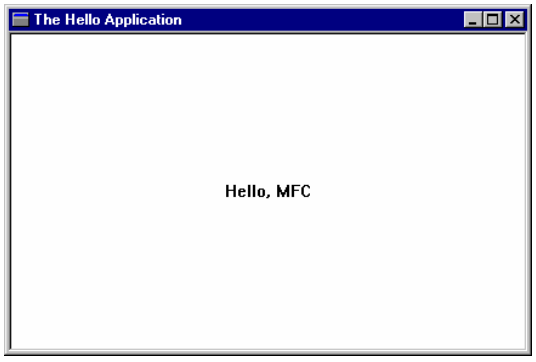
```
class CMyApp : public CWinApp
{
public:
    virtual BOOL InitInstance();
};
class CMainWindow : public CFrameWnd
{
public:
    CMainWindow();

protected:
    afx_msg void OnPaint();
    DECLARE_MESSAGE_MAP()
};
```

Hello.cpp

```
#include <afxwin.h>
#include "Hello.h"

CMyApp myApp;
```

```
//////////////////////////////////////////////////////////////////////////
// CMyApp member functions
BOOL CMyApp::InitInstance()
{
    m_pMainWnd = new CMainWindow;

    m_pMainWnd->ShowWindow(m_nCmdShow);
    m_pMainWnd->UpdateWindow();
    return TRUE;
}
//////////////////////////////////////////////////////////////////////////
// CMainWindow message map and member functions
BEGIN_MESSAGE_MAP(CMainWindow, CFrameWnd)
    ON_WM_PAINT()
END_MESSAGE_MAP()
CMainWindow::CMainWindow()
{
    Create(NULL, _T("The Hello Application"));
}
void CMainWindow::OnPaint()
{
    CPaintDC dc(this);

    CRect rect;
    GetClientRect(&rect);

    dc.DrawText(_T("Hello, MFC"), -1, &rect,
        DT_SINGLELINE | DT_CENTER | DT_VCENTER);
}
```

Màn hình kết quả như sau:



Một ứng dụng phát triển dựa trên tập thư viện cơ sở MFC bao gồm một số đối tượng và quá trình xử lý như sau::

1.1.5.1 Đối tượng ứng dụng (Application)

Trung tâm của một ứng dụng MFC là đối tượng (*application*) dựa trên lớp CWinApp. CWinApp cung cấp một vòng lặp thông báo để nhận các thông báo và phân phối chúng cho cửa sổ ứng dụng. Nó cũng bao gồm các hàm ảo chính yếu (nó mà có thể bị khai báo và điều chỉnh lại các hành vi của ứng dụng). CWinApp và các lớp MFC khác được đưa vào trong ứng dụng khi chúng ta gắn kết (*include*) file tiêu đề Afxwin.h. Ứng dụng MFC có thể có duy nhất một đối tượng ứng dụng và đối tượng ứng dụng này cần được khai báo với phạm vi toàn cục được khởi tạo trong bộ nhớ từ lúc khởi điểm của chương trình.

CMyApp (trong ví dụ trên) khai báo không có biến thành viên và khai báo lại (*overrides*) một hàm kế thừa từ lớp cha CWinApp. Hàm InitInstance được gọi từ rất sớm trong thời gian sống của ứng dụng, ngay sau khi ứng dụng bắt đầu thực thi nhưng trước khi cửa sổ ứng dụng được tạo ra. Thực tế, ngoại trừ việc hàm InitInstance tạo một cửa sổ cho ứng dụng thì ứng dụng không hề có một cửa sổ. Đây chính là lý do thậm chí một ứng dụng MFC ở mức tối thiểu củng cần kế thừa một lớp từ CWinApp và khai báo lại hàm CWinApp::InitInstance.

1.1.5.2 Đối tượng Khung Cửa sổ (Frame Window)

Lớp CWnd và các phát sinh của nó cung cấp một giao diện hướng đối tượng cho một hay nhiều cửa sổ do ứng dụng tạo ra. Lớp cửa sổ chính của ứng dụng, CMainWindow, được kế thừa từ lớp CFrameWnd (cùng được kế thừa từ CWnd). Lớp CFrameWnd mô hình hoá các hành vi của khung cửa sổ, đồng thời nó là cửa sổ mức cao nhất phục vụ như một giao diện chủ yếu của ứng dụng với thế giới bên ngoài. Trong ngữ cảnh lý tưởng của kiến trúc document/view, cửa sổ khung đóng vai trò như là một lớp chứa thông minh cho các views, toolbars, status bars, và các đối tượng giao diện người sử dụng (*user-interface, UI*) khác.

Một ứng dụng MFC tạo một cửa sổ thông qua việc tạo một đối tượng cửa sổ và gọi hàm Create hay CreateEx của nó có dạng như sau:

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

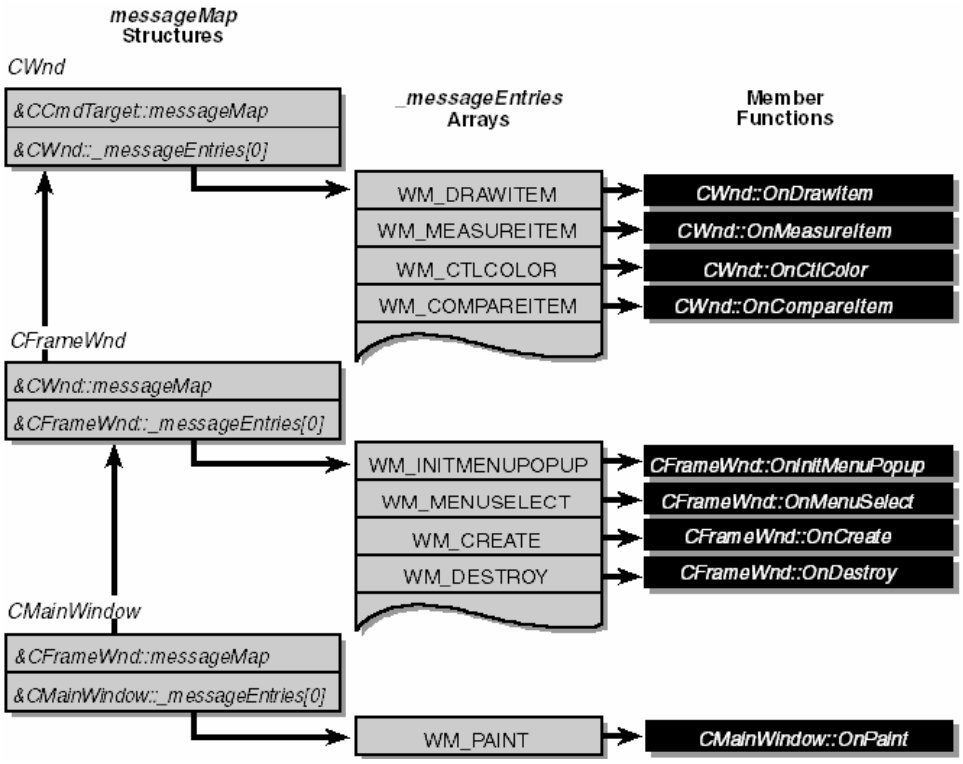
1.1.5.3 Quá trình làm việc của các ánh xạ thông báo (Message Map)

Quá trình làm việc này khảo sát các macro DECLARE_MESSAGE_MAP, BEGIN_MESSAGE_MAP, và END_MESSAGE_MAP trong Afxwin.h và mã lệnh cho hàm CWnd::WindowProc trong Wincore.cpp

```
// In the class declaration
DECLARE_MESSAGE_MAP()


// In the class implementation
BEGIN_MESSAGE_MAP(CMainWindow, CFrameWnd)
   ON_WM_PAINT()
END_MESSAGE_MAP()
```

Để phân phối thông báo, khung ứng dụng gọi hàm ảo WindowProc (mà lớp CMainWindow kế thừa từ CWnd). Hàm WindowProc gọi OnWndMsg mà trong đó gọi tiếp hàm GetMessageMap để lấy con trỏ chỉ đến CMainWindow::messageMap và tìm kiếm CMainWindow::_messageEntries cho những thông báo có ID trùng với ID của các thông báo đang chờ xử lý. Nếu kết quả tìm thấy, lớp CMainWindow tương ứng (của những địa chỉ được lưu trong dãy _messageEntries với ID của thông báo) được gọi. Ngược lại, OnWndMsg tham khảo CMainWindow::messageMap cho một con trỏ chỉ tới CFrameWnd::messageMap và lặp lại quá trình cho lớp cơ sở. Nếu lớp cơ sở không có một điều khiển (*handler*) cho thông báo, khung ứng dụng the framework phát triển lên mức khác và tham khảo đến lớp cơ sở cha.

Các ánh xạ thông báo CMainWindow thể hiện dạng sơ đồ như hình dưới đây và thể hiện các nhánh truy xuất/tìm kiếm cho một điều khiển trùng với ID của thông báo đã cho, bắt đầu với các ánh xạ thông báo cho CMainWindow.



Hình. Quá trình xử lý thông báo

1.1.5.4 Windows, Character Sets, và _T Macro

Windows 98 và Windows NT sử dụng hai tập ký tự khác nhau từ các dạng ký tự và chuỗi. Windows 98 và các phiên bản trước đó sử dụng tập ký tự ANSI 8 bit tương tự với tập ký tự ASCII thân thiện với các lập trình viên. Windows NT và Windows 2000 sử dụng tập ký tự Unicode 16 bit bao trùm cả tập ký tự ANSI nhằm phục vụ cho các ứng dụng quốc tế (có thể không sử dụng bảng mẫu tự tiếng Anh).

Các chương trình được biên dịch với ký tự ANSI sẽ hoạt động được trên Windows NT and Windows 2000, nhưng các chương trình dùng Unicode sẽ có thể thực thi nhanh hơn vì Windows NT và Windows 2000 không hỗ trợ việc chuyển đổi từ ANSI sang Unicode cho tất cả ký tự. Ngược lại, ứng dụng dùng Unicode không thực thi trên Windows 98 ngoại trừ khi thực hiện việc chuyển đổi mọi chuỗi ký tự từ Unicode sang dạng ANSI.

Nếu một chuỗi như: "Hello" thì trình biên dịch sẽ thể hiện dạng chuỗi ký tự ANSI.

Nếu khai báo chuỗi trên theo dạng L"Hello" thì trình biên dịch sẽ thể hiện dạng chuỗi ký tự Unicode.

Nhưng nếu dùng macro _T (của MFC) cho chuỗi trên theo dạng _T ("Hello") thì kết quả sẽ được thể hiện dạng Unicode nếu ký hiệu tiền xử lý _UNICODE được định nghĩa, và mặc định là dạng ANSI.

1.1.5.5 Hàm UpdateData

Hàm có dạng **UpdateData**(*tham_số*) với:

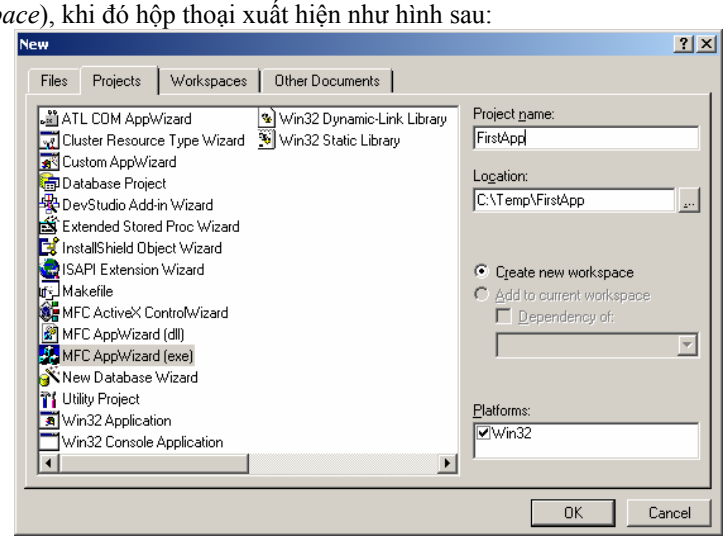
- *tham_số* là TRUE: hàm sẽ thực hiện việc cập nhật dữ liệu trong các điều khiển vào các biến liên kết tương ứng.
- *tham_số* là FALSE: hàm sẽ thực hiện việc cập nhật dữ liệu từ các biến liên kết vào trong các điều khiển tương ứng và hiển thị trên giao diện chương trình.

Một số lưu ý:

- Nên khai báo ký tự kiểu TCHAR thay cho kiểu char. Nếu ký hiệu _UNICODE được định nghĩa, TCHAR xác định kiểu wchar_t (ký tự Unicode 16 bit). Nếu _UNICODE không được định nghĩa thì TCHAR trở thành ký hiệu thông thường.
- Không nên dùng char* hay wchar_t* để khai báo con trỏ kiểu chuỗi TCHAR. Nên dùng TCHAR* hay LPTSTR (con trỏ chỉ tới chuỗi TCHAR) và LPCTSTR (con trỏ chỉ tới chuỗi hằng TCHAR).
- Không nên giả định là ký tự chỉ có độ rộng 8 bit. Để chuyển một độ dài của bộ đệm nhanh ở dạng byte sang dạng ký tự, nên dùng sizeof(TCHAR).
- Thay các việc gọi hàm chuỗi trong thư việc C-trong thời gian thực thi (ví dụ strcpy) với các macros tương ứng trong file tiêu đề Tchar.h (ví dụ, _tcscpy).

1.1.6 Tạo ứng dụng với MS Visual C++

Từ menu File, người dùng chọn lệnh New... để tạo mới một dự án (project), một tập tin (*file*) hay một không gian làm việc (*workspace*), khi đó hộp thoại xuất hiện như hình sau:



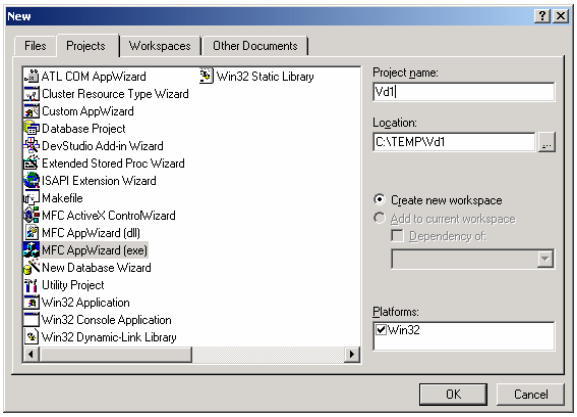
Trong hộp thoại này, người dùng có nhiều loại trình ứng dụng có thể tạo với MS Visual C++:

- Tạo ứng dụng thực thi trong Windows (dạng EXE file) với MFC có hỗ trợ tư vấn với **MFC AppWizard (exe)**
- Tạo thư viện trong Windows (dạng DLL file) với MFC có hỗ trợ tư vấn với **MFC AppWizard (dll)**
- Tạo ứng dụng thực thi trong Windows (dạng EXE file) dạng thông thường sử dụng API với **Win32 Application**
- Tạo ứng dụng thực thi trong DOS (dạng EXE file) dạng thông thường với **Win32 Console Application...**

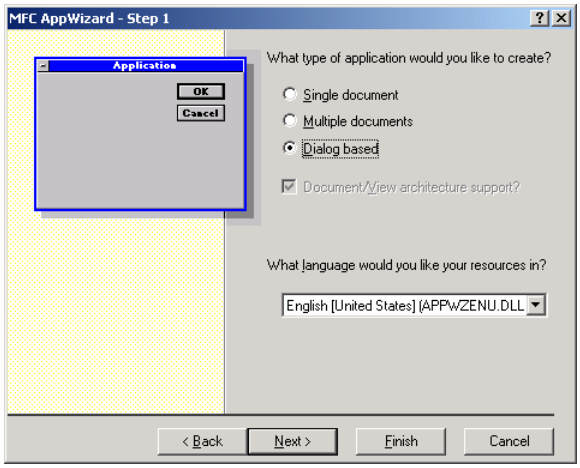
Đặc biệt, khi người dùng chọn cách tạo ứng dụng dạng cửa sổ với **MFC AppWizard (exe)**, người dùng có thể tạo trình ứng dụng dạng hộp thoại (*dialog*), ứng dụng đơn tài liệu (*Single Document Interface - SDI*), ứng dụng đa tài liệu (*Multi Document Interface - MDI*)

Nếu tạo ứng dụng dạng hộp thoại (dialog), người dùng cần làm như sau:

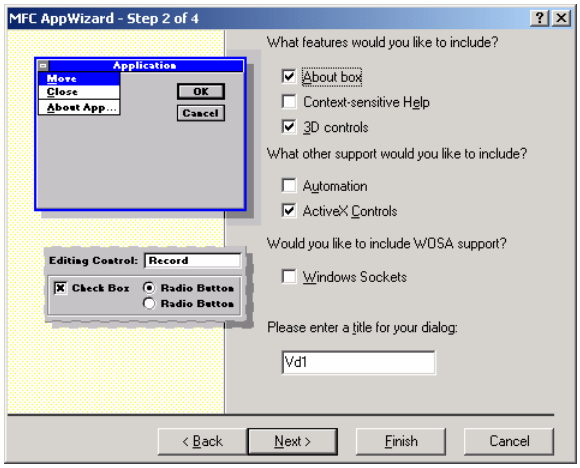
- Bước khởi đầu:



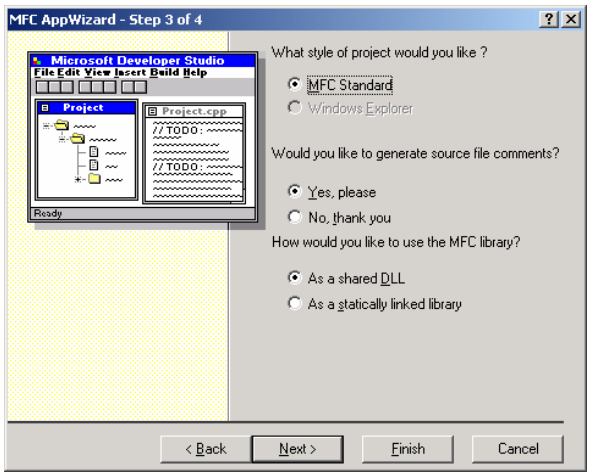
- Bước 1:



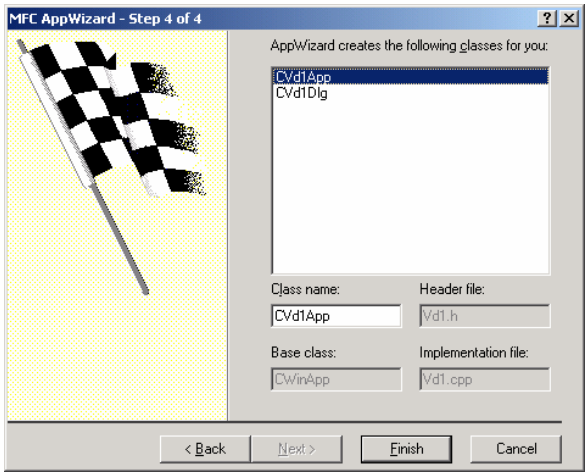
- Bước 2:



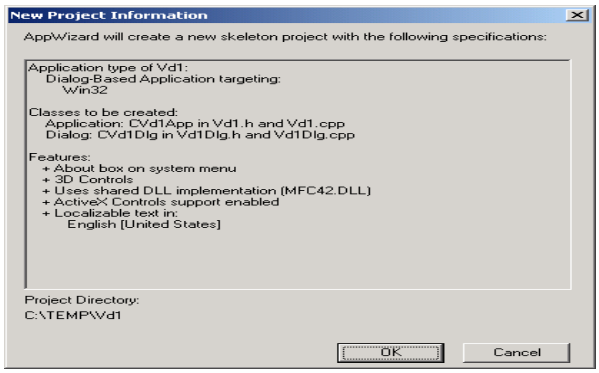
- Bước 3:



- Bước 4:

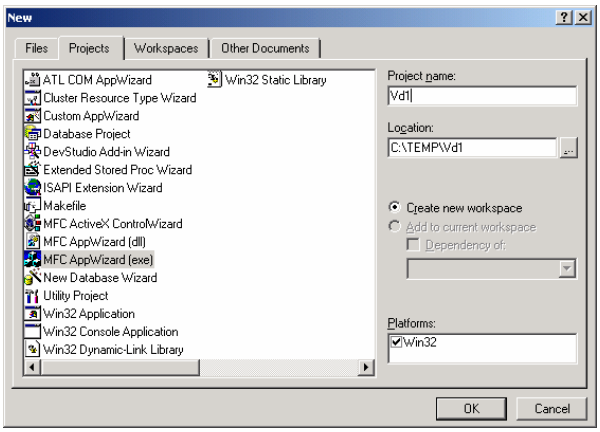


- Bước kết thúc:

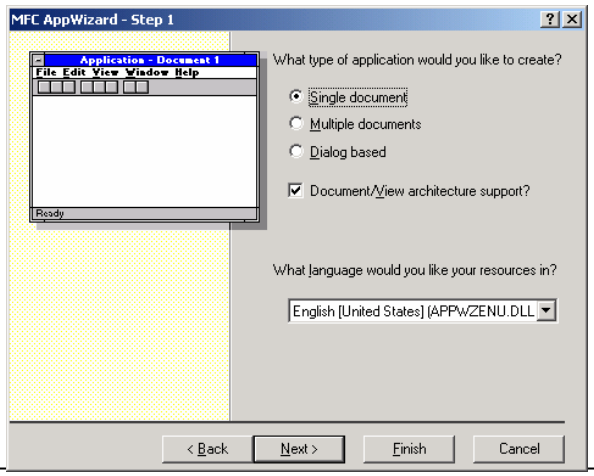


Nếu tạo ứng dụng dạng đơn tài liệu hay đa tài liệu, người dùng cần làm như sau:

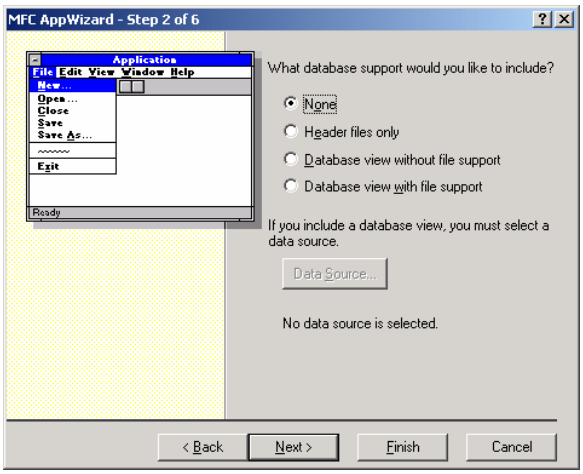
- Bước khởi đầu:



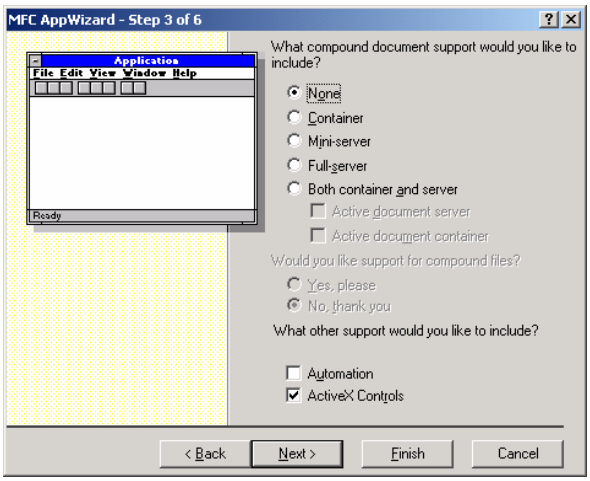
- Bước 1:



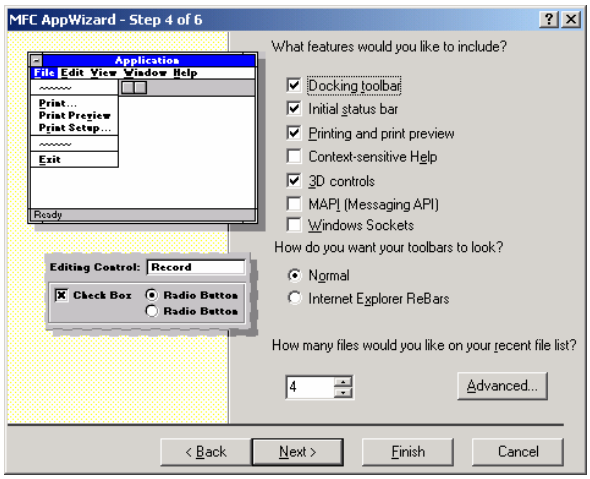
- Bước 2:



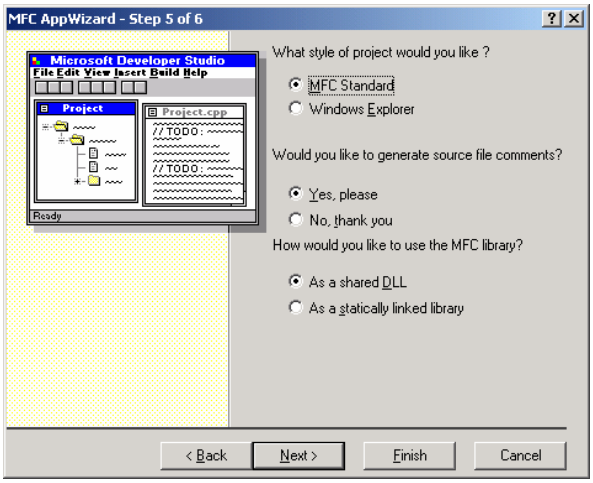
- Bước 3:



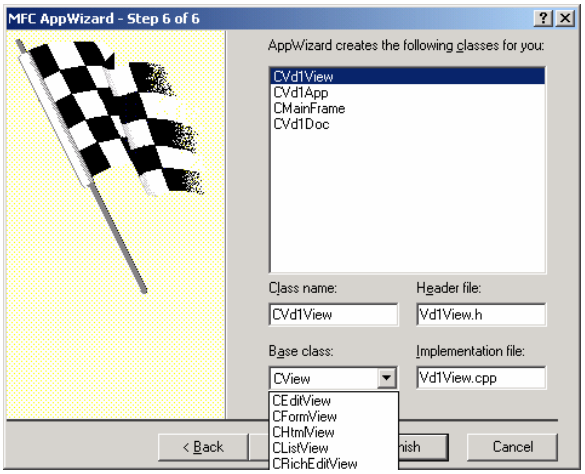
- Bước 4:



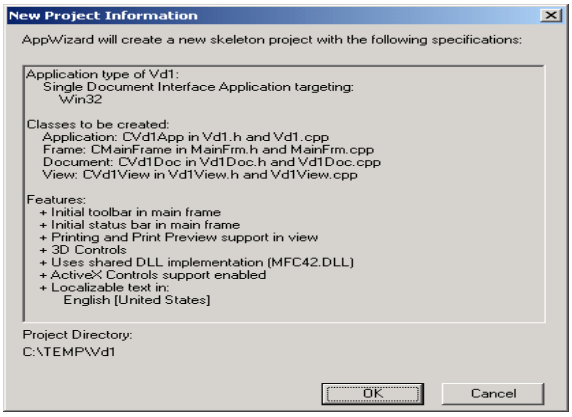
- Bước 5:



- Bước 6:



- Bước kết thúc:



1.2 XỬ LÝ VẼ HÌNH TRONG ỨNG DỤNG WINDOWS

1.2.1 Vấn đề quan tâm

- Tìm hiểu về ngữ cảnh thiết bị và giao diện thiết bị đồ hoạ.
- Sự hỗ trợ của MFC về các lớp công cụ vẽ (CPen, CBrush…)

1.2.2 Giới thiệu

Hệ điều hành Windows cung cấp thiết bị đồ hoạ ảo (*Graphics Device Interface - GDI*) để giúp người dùng thực hiện các thao tác vẽ đồ hoạ dễ dàng hơn vì thiết bị này không phụ thuộc vào phần cứng đồ hoạ của hệ thống.

Một chương trình ứng dụng (*WindowsApp*) không vẽ trực tiếp ra màn hình, máy in… mà chỉ vẽ trên "bề mặt luận lý" thể hiện bởi ngữ cảnh thiết bị (*Device Context – DC*). Ngữ cảnh thiết bị chứa các thông tin về hệ thống, ứng dụng và cửa sổ của WindowsApp cũng như các đối tượng đồ hoạ đang được vẽ trong WindowApp đó. Thực chất ngữ cảnh thiết bị dùng hiển thị đồ hoạ là ngữ cảnh ảo của cửa sổ.

1.2.3 Truy xuất ngữ cảnh thiết bị

MFC cung cấp một số lớp ngữ cảnh thiết bị (kế thừa từ lớp CDC) như:

Class	Mô tả
CPaintDC	Sử dụng cho việc vẽ trong vùng ứng dụng của cửa sổ (chỉ thao tác với sự kiện OnPaint)
CClientDC	Sử dụng cho việc vẽ trong vùng ứng dụng của cửa sổ (vẽ bất cứ nơi nào nhưng chỉ thao tác với sự kiện OnPaint)
CWindowDC	Sử dụng cho việc vẽ trong cửa sổ, bao gồm cả vùng không là vùng ứng dụng của cửa sổ
CMetaFileDC	Sử dụng cho việc vẽ một GDI metafile

Để lấy ngữ cảnh thiết bị, dùng:

```
CDC dc(this);
```

hay

```
CDC* pDC = GetDC();
```

Để giải phóng ngữ cảnh thiết bị, dùng:

```
ReleaseDC(pDC);
delete pDC;
```

Ngữ cảnh thiết bị dùng "bút vẽ" (*pen*) để vẽ các đường thẳng/hình dáng, và "cọ vẽ" (*brush*) để tô đầy các vùng của hình vẽ trên màn hình.

Ví dụ:

```
CRect rect;
GetClientRect(&rect);
CClientDC dc(this);
dc.MoveTo(rect.left, rect.top);
dc.LineTo(rect.right, rect.bottom);
dc.Ellipse(0, 0, 100, 100);
```

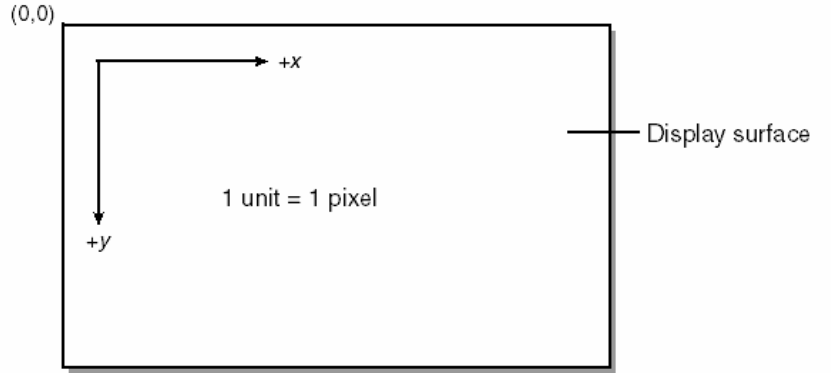
hay:

```
CRect rect;
GetClientRect(&rect);
CDC* pDC = GetDC();
pDC->MoveTo(rect.left, rect.top);
pDC->LineTo(rect.right, rect.bottom);
pDC->Ellipse(0, 0, 100, 100);
```

1.2.3.1 Xác định chế độ đo lường

Chế độ liên kết	Khoảng cách tương ứng với một đơn vị luận lý	Hướng của trục x và y
MM_TEXT	1 pixel	
MM_LOMETRIC	0.1 mm	
MM_HIMETRIC	0.01 mm	

MM_LOENGLISH	0.01 in.	
MM_HIENGLISH	0.001 in.	
MM_TWIPS	1/1440 in. (0.0007 in.)	
MM_ISOTROPIC	Người dùng định nghĩa (x và y có tỉ lệ xác định)	Người dùng định nghĩa
MM_ANISOTROPIC	Người dùng định nghĩa (x và y có tỉ lệ xác định)	Người dùng định nghĩa



1.2.4 Thao tác vẽ với bút vẽ

Trong MFC, lớp bút vẽ là CPen, được dùng để vẽ kiểu đường bất kỳ với màu/độ rộng xác định, ví dụ như sau tạo bút vẽ mới và chọn làm bút vẽ hiện thời, dùng:

```
CDC *pDC = GetDC();
CPen cp(PS_SOLID, 1, RGB(0,0,0));
pDC->SelectObject(&cp);
...
```

tạo bút vẽ mới đồng thời lưu lại bút vẽ đã sử dụng trước đó:

```
CDC *pDC = GetDC();
CPen pen (PS_SOLID, 10, RGB (255, 0, 0));
CPen* pOldPen = pDC->SelectObject (&pen);
...
```

trong đó hàm ***RGB(r, g, b)*** tạo màu chỉ định dựa trên 3 màu cơ bản là R, G, B với các tham số r, g và b $\in$ [0, 255] và các kiểu nét như PS_SOLID, PS_DASH, PS_DOT ... như sau:

PS_SOLID

PS_DASH

PS_DOT

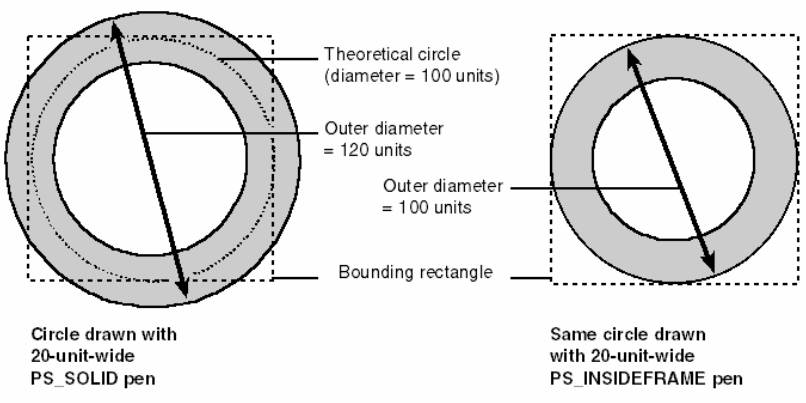
PS_DASHDOT

PS_DASHDOTDOT

PS_NULL

PS_INSIDEFRAME

trong đó



Một số phương thức vẽ đường:

Phương thức	Mô tả
MoveTo	Di chuyển bút vẽ đến 1 điểm xác định
LineTo	Vẽ 1 đoạn thẳng từ điểm hiện hành của bút vẽ đến 1 điểm xác định và di chuyển vị trí hiện hành đến điểm mới này
Polyline	Vẽ đường gấp khúc (tập hợp các đoạn gấp khúc)
PolylineTo	Vẽ đường gấp khúc và di chuyển vị trí hiện hành đến đỉnh cuối cùng của đường này.
Arc	Vẽ cung
ArcTo	Vẽ cung và di chuyển vị trí hiện hành đến đỉnh cuối cùng của cung này.
PolyBezier	Vẽ đường Bezier
PolyBezierTo	Vẽ đường Bezier và di chuyển vị trí hiện hành đến đỉnh cuối cùng của đường này.

Một số phương thức vẽ hình khối:

Phương thức	Mô tả
Chord	Vẽ hình dạng bán cầu
Ellipse	Vẽ hình dạng ellipse
Pie	Vẽ hình dạng bánh
Polygon	Nối một tập các điểm của một đa giác
Rectangle	Vẽ hình dạng chữ nhật
RoundRect	Vẽ hình dạng chữ nhật tròn góc

Một số mã màu thông dụng:

Tên màu	R	G	B	Tên màu	R	G	B
Black	0	0	0	Light gray	192	192	192
Blue	0	0	192	Bright blue	0	0	255
Green	0	192	0	Bright green	0	255	0
Cyan	0	192	192	Bright cyan	0	255	255
Red	192	0	0	Bright red	255	0	0
Magenta	192	0	192	Bright magenta	255	0	255
Yellow	192	192	0	Bright yellow	255	255	0
Dark gray	128	128	128	White	255	255	255

Chú ý:

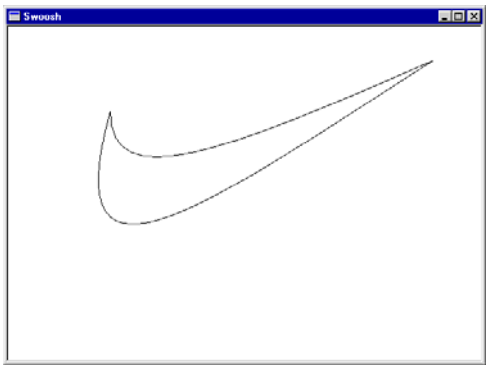
Việc xử lý vẽ có thể đặt trong sự kiện OnPaint, OnDraw hay các sự kiện liên quan đến thao tác chuột và bàn phím.

Ví dụ:

Với đoạn chương trình sau đây:

```
POINT aPoint1[4] = { 120, 100, 120, 200, 250, 150, 500, 40 };
POINT aPoint2[4] = { 120, 100,  50, 350, 250, 200, 500, 40 };
dc.PolyBezier (aPoint1, 4);
dc.PolyBezier (aPoint2, 4);
```

Màn hình kết quả là :

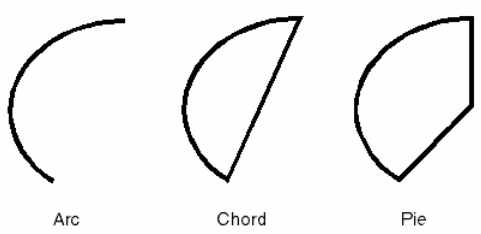


Ví dụ:

Với đoạn chương trình sau đây:

```
#include <math.h>
#define PI 3.1415926

void CMainWindow::OnPaint()
{
    CPaintDC dc (this);
    int nRevenues[4] = { 125, 376, 252, 184 };

    CRect rect;
    GetClientRect(&rect);
    dc.SetViewportOrg(rect.Width() / 2, rect.Height() / 2);

    int nTotal = 0;
    for (int i=0; i<4; i++)
        nTotal += nRevenues[i];
    int x1 = 0;
    int y1 = -1000;
    int nSum = 0;
    for (i=0; i<4; i++) {
        nSum += nRevenues[i];
        double rad = ((double)(nSum * 2 * PI)/(double) nTotal) + PI;
        int x2 = (int) (sin (rad) * 1000);
        int y2 = (int) (cos (rad) * 1000 * 3) / 4;
        dc.Pie (-200, -150, 200, 150, x1, y1, x2, y2);
        x1 = x2;
        y1 = y2;
    }
}
```

Màn hình kết quả là



1.2.5 Thao tác tô màu với cọ vẽ

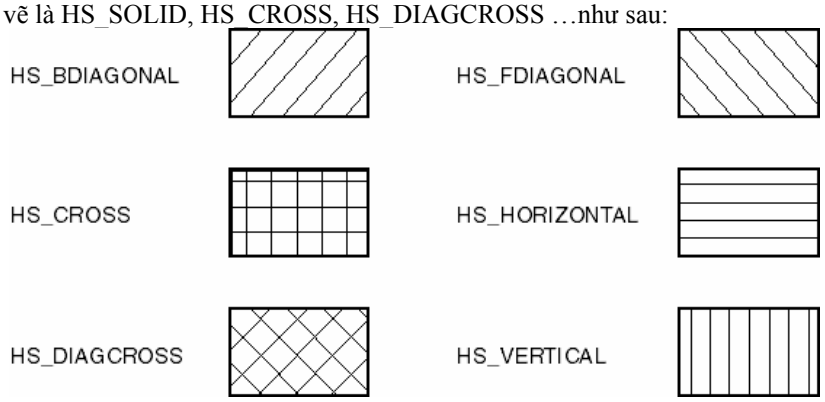
Lớp cọ vẽ là CBrush, được dùng để tạo cọ vẽ với màu/mẫu vẽ xác định:

```
CDC *pDC=GetDC();
CBrush brush (HS_DIAGCROSS, RGB (255, 255, 255));
pDC->SelectObject(&brush);
...
```

tạo cọ vẽ mới đồng thời lưu lại cọ vẽ đã sử dụng trước đó:

```
CDC *pDC=GetDC();
CBrush brush (HS_DIAGCROSS, RGB (255, 255, 255));
CBrush* pOldBrush = pDC->SelectObject(&brush);
...
```

trong đó các mẫu vẽ là HS_SOLID, HS_CROSS, HS_DIAGCROSS …như sau:



Ví dụ:

```
CBrush brush (HS_DIAGCROSS, RGB (0, 0, 0));
pDC->SelectObject (&brush);
pDC->SetBkMode (TRANSPARENT);
pDC->Rectangle (0, 0, 100, 100);
```

hay:

```
CBrush brush (HS_DIAGCROSS, RGB (255, 255, 255));
pDC->SelectObject (&brush);
pDC->SetBkColor (RGB (192, 192, 192));
pDC->Rectangle (0, 0, 100, 100);
```

1.2.6 Hiển thị văn bản trong môi trường đồ hoạ

Ngữ cảnh thiết bị cung cấp một số hàm thực hiện việc hỗ trợ hiển thị văn bản trong môi trường đồ hoạ như sau:

Hàm	Mô tả
DrawText	Vẽ một văn bản trong một khung chữ nhật định dạng trước
TextOut	Xuất một hành văn bản tại vị trí hiện hành hay tại điểm xác định
TabbedTextOut	Xuất một hành văn bản chứa đựng các ký hiệu tab
ExtTextOut	Xuất một hàng văn bản trong một khung chữ nhật có màu tùy chọn hay các ký tự xen giữa khác nhau
GetTextExtent	Lấy độ rộng chuỗi với font sử dụng hiện hành
GetTabbedTextExtent	Lấy độ rộng chuỗi (có chứa đựng ký hiệu tab) với font sử dụng hiện hành
GetTextMetrics	Lấy font metrics (chiều cao ký tự, độ rộng trung bình) của font hiện hành

SetTextAlign	Canh lề của văn bản cho hàm TextOut và các hàm xuất nội dung văn bản khác
SetTextJustification	Canh đều văn bản
SetTextColor	Đặt màu cho văn bản xuất ra
SetBkColor	Đặt màu nền

Ví dụ:

```
CString string = _T ("Now is the time");
CSize size = dc.GetTextExtent (string);
dc.SetTextJustification (nWidth - size.cx, 3);
dc.TextOut (0, y, string);
```

hay:

```
dc.DrawText (_T ("Hello, MFC"), -1, &rect,
    DT_SINGLELINE ¦ DT_CENTER ¦ DT_VCENTER);
```

1.2.7 GDI Fonts và lớp CFont

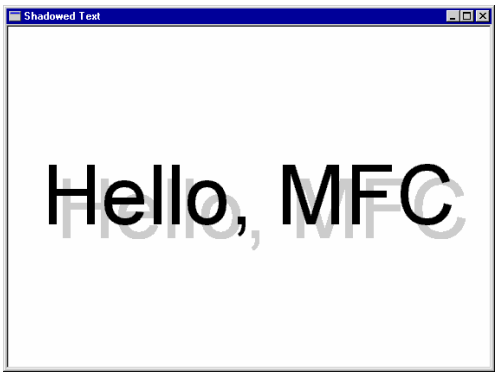
Ngoài việc sử dụng font chữ mặc định, người dùng còn có thể tạo font chữ trong chế độ đồ hoạ tùy chọn.

Ví dụ:

Với đoạn chương trình sau đây:

```
void CMainWindow::OnPaint ()
{
    CRect rect;
    GetClientRect (&rect);

    CFont font;
    font.CreatePointFont (720, _T ("Arial"));

    CPaintDC dc (this);
    dc.SelectObject (&font);
    dc.SetBkMode (TRANSPARENT);

    CString string = _T ("Hello, MFC");

    rect.OffsetRect (16, 16);
    dc.SetTextColor (RGB (192, 192, 192));
    dc.DrawText (string, &rect, DT_SINGLELINE ¦
        DT_CENTER ¦ DT_VCENTER);

    rect.OffsetRect (-16, -16);
    dc.SetTextColor (RGB (0, 0, 0));
    dc.DrawText (string, &rect, DT_SINGLELINE ¦
        DT_CENTER ¦ DT_VCENTER);
}
```

Màn hình kết quả như sau:

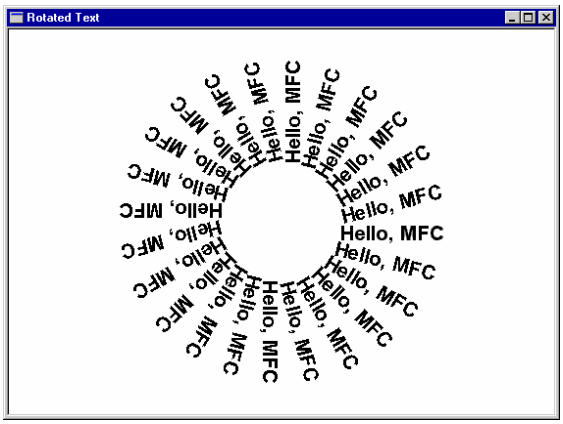


Ví dụ:

Với đoạn chương trình sau đây:

```
void CMainWindow::OnPaint()
{
    CRect rect;
    GetClientRect(&rect);

    CPaintDC dc(this);
    dc.SetViewportOrg(rect.Width () / 2, rect.Height () / 2);
    dc.SetBkMode(TRANSPARENT);

    for (int i=0; i<3600; i+=150) {
        LOGFONT lf;
        ::ZeroMemory(&lf, sizeof (lf));
        lf.lfHeight = 160;
        lf.lfWeight = FW_BOLD;
        lf.lfEscapement = i;
        lf.lfOrientation = i;
        ::lstrcpy(lf.lfFaceName, _T ("Arial"));

        CFont font;
        font.CreatePointFontIndirect(&lf);

        CFont* pOldFont = dc.SelectObject(&font);
        dc.TextOut(0, 0, CString (_T ("          Hello, MFC")));
        dc.SelectObject(pOldFont);
    }
}
```

Màn hình kết quả như sau:



1.2.8 Ví dụ tổng hợp

1.2.8.1 Chương trình 1

Accel.h

```
#define LINESIZE 8

class CMyApp : public CWinApp
{
public:
    virtual BOOL InitInstance();
};
class CMainWindow : public CFrameWnd
{
protected:
    int m_nCellWidth;   // Cell width in pixels
    int m_nCellHeight;  // Cell height in pixels
```

```
    int m_nRibbonWidth; // Ribbon width in pixels
    int m_nViewWidth;   // Workspace width in pixels
    int m_nViewHeight;  // Workspace height in pixels
    int m_nHScrollPos;  // Horizontal scroll position
    int m_nVScrollPos;  // Vertical scroll position
    int m_nHPageSize;   // Horizontal page size
    int m_nVPageSize;   // Vertical page size

public:
    CMainWindow();
protected:
    afx_msg void OnPaint();
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    afx_msg void OnSize(UINT nType, int cx, int cy);
    afx_msg void OnHScroll(UINT nCode, UINT nPos,
        CScrollBar* pScrollBar);
    afx_msg void OnVScroll(UINT nCode, UINT nPos,
        CScrollBar* pScrollBar);

    DECLARE_MESSAGE_MAP()
};
```

Accel.cpp

```
#include <afxwin.h>
#include "Accel.h"

CMyApp myApp;

//////////////////////////////////////////////////////////////////////
// CMyApp member functions
BOOL CMyApp::InitInstance()
{
    m_pMainWnd = new CMainWindow;
    m_pMainWnd->ShowWindow (m_nCmdShow);
    m_pMainWnd->UpdateWindow();
    return TRUE;
}
//////////////////////////////////////////////////////////////////////
// CMainWindow message map and member functions
BEGIN_MESSAGE_MAP(CMainWindow, CFrameWnd)
    ON_WM_CREATE()
    ON_WM_SIZE()
    ON_WM_PAINT()
    ON_WM_HSCROLL()
    ON_WM_VSCROLL()
END_MESSAGE_MAP()
CMainWindow::CMainWindow()
{
    Create(NULL, _T ("Accel"),
        WS_OVERLAPPEDWINDOW ¦ WS_HSCROLL ¦ WS_VSCROLL);
}
int CMainWindow::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
        return -1;
```

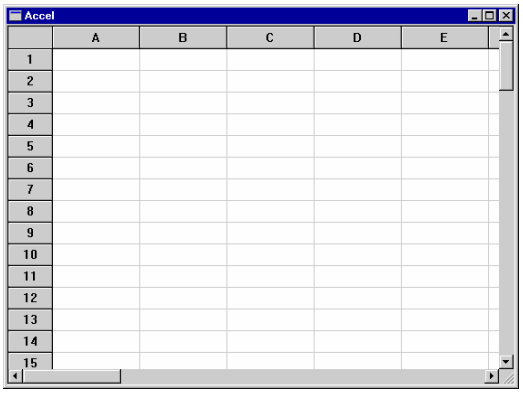
```
    CClientDC dc(this);
    m_nCellWidth = dc.GetDeviceCaps(LOGPIXELSX);
    m_nCellHeight = dc.GetDeviceCaps(LOGPIXELSY) / 4;
    m_nRibbonWidth = m_nCellWidth / 2;
    m_nViewWidth = (26 * m_nCellWidth) + m_nRibbonWidth;
    m_nViewHeight = m_nCellHeight * 100;
    return 0;
}
void CMainWindow::OnSize(UINT nType, int cx, int cy)
{
    CFrameWnd::OnSize(nType, cx, cy);
    //
    // Set the horizontal scrolling parameters.
    //
    int nHScrollMax = 0;
    m_nHScrollPos = m_nHPageSize = 0;

    if (cx < m_nViewWidth) {
        nHScrollMax = m_nViewWidth - 1;
        m_nHPageSize = cx;
        m_nHScrollPos = min(m_nHScrollPos, m_nViewWidth -
            m_nHPageSize - 1);
    }

    SCROLLINFO si;
    si.fMask = SIF_PAGE | SIF_RANGE | SIF_POS;
    si.nMin = 0;
    si.nMax = nHScrollMax;
    si.nPos = m_nHScrollPos;
    si.nPage = m_nHPageSize;

    SetScrollInfo(SB_HORZ, &si, TRUE);
    //
    // Set the vertical scrolling parameters.
    //
    int nVScrollMax = 0;
    m_nVScrollPos = m_nVPageSize = 0;

    if (cy < m_nViewHeight) {
        nVScrollMax = m_nViewHeight - 1;
        m_nVPageSize = cy;
        m_nVScrollPos = min(m_nVScrollPos, m_nViewHeight -
            m_nVPageSize - 1);
    }

    si.fMask = SIF_PAGE | SIF_RANGE | SIF_POS;
    si.nMin = 0;
    si.nMax = nVScrollMax;
    si.nPos = m_nVScrollPos;
    si.nPage = m_nVPageSize;

    SetScrollInfo(SB_VERT, &si, TRUE);
}
void CMainWindow::OnPaint ()
{
```

```
CPaintDC dc (this);
//
// Set the window origin to reflect the current scroll positions.
//
dc.SetWindowOrg(m_nHScrollPos, m_nVScrollPos);
//
// Draw the grid lines.
//
CPen pen(PS_SOLID, 0, RGB (192, 192, 192));
CPen* pOldPen = dc.SelectObject (&pen);

for (int i=0; i<99; i++) {
    int y = (i * m_nCellHeight) + m_nCellHeight;
    dc.MoveTo(0, y);
    dc.LineTo(m_nViewWidth, y);
}

for (int j=0; j<26; j++) {
    int x = (j * m_nCellWidth) + m_nRibbonWidth;
    dc.MoveTo(x, 0);
    dc.LineTo(x, m_nViewHeight);
}
dc.SelectObject (pOldPen);
//
// Draw the bodies of the rows and the column headers.
//
CBrush brush;
brush.CreateStockObject(LTGRAY_BRUSH);

CRect rcTop(0, 0, m_nViewWidth, m_nCellHeight);
dc.FillRect(rcTop, &brush);
CRect rcLeft(0, 0, m_nRibbonWidth, m_nViewHeight);
dc.FillRect(rcLeft, &brush);

dc.MoveTo(0, m_nCellHeight);
dc.LineTo(m_nViewWidth, m_nCellHeight);
dc.MoveTo(m_nRibbonWidth, 0);
dc.LineTo(m_nRibbonWidth, m_nViewHeight);

dc.SetBkMode(TRANSPARENT);
//
// Add numbers and button outlines to the row headers.
//
for (i=0; i<99; i++) {
    int y = (i * m_nCellHeight) + m_nCellHeight;
    dc.MoveTo(0, y);
    dc.LineTo(m_nRibbonWidth, y);

    CString string;
    string.Format(_T ("%d"), i + 1);

    CRect rect(0, y, m_nRibbonWidth, y + m_nCellHeight);
    dc.DrawText(string, &rect, DT_SINGLELINE ¦
        DT_CENTER ¦ DT_VCENTER);
```

```
        rect.top++;
        dc.Draw3dRect(rect, RGB (255, 255, 255),
            RGB (128, 128, 128));
    }
    //
    // Add letters and button outlines to the column headers.
    //
    for (j=0; j<26; j++) {
        int x = (j * m_nCellWidth) + m_nRibbonWidth;
        dc.MoveTo(x, 0);
        dc.LineTo(x, m_nCellHeight);

        CString string;
        string.Format(_T ("%c"), j + `A');

        CRect rect(x, 0, x + m_nCellWidth, m_nCellHeight);
        dc.DrawText(string, &rect, DT_SINGLELINE ¦
            DT_CENTER ¦ DT_VCENTER);

        rect.left++;
        dc.Draw3dRect(rect, RGB (255, 255, 255),
            RGB (128, 128, 128));
    }
}
void CMainWindow::OnHScroll(UINT nCode, UINT nPos, CScrollBar* pScrollBar)
{
    int nDelta;

    switch (nCode) {
    case SB_LINELEFT:
        nDelta = -LINESIZE;
        break;

    case SB_PAGELEFT:
        nDelta = -m_nHPageSize;
        break;
    case SB_THUMBTRACK:
        nDelta = (int) nPos - m_nHScrollPos;
        break;
    case SB_PAGERIGHT:
        nDelta = m_nHPageSize;
        break;
    case SB_LINERIGHT:
        nDelta = LINESIZE;
        break;
    default: // Ignore other scroll bar messages
        return;
    }

    int nScrollPos = m_nHScrollPos + nDelta;
    int nMaxPos = m_nViewWidth - m_nHPageSize;

    if (nScrollPos < 0)
        nDelta = -m_nHScrollPos;
    else if (nScrollPos > nMaxPos)
```

```
        nDelta = nMaxPos - m_nHScrollPos;

    if (nDelta != 0) {
        m_nHScrollPos += nDelta;
        SetScrollPos (SB_HORZ, m_nHScrollPos, TRUE);
        ScrollWindow (-nDelta, 0);
    }
}
void CMainWindow::OnVScroll(UINT nCode, UINT nPos, CScrollBar* pScrollBar)
{
    int nDelta;

    switch (nCode) {
    case SB_LINEUP:
        nDelta = -LINESIZE;
        break;
    case SB_PAGEUP:
        nDelta = -m_nVPageSize;
        break;
    case SB_THUMBTRACK:
        nDelta = (int) nPos - m_nVScrollPos;
        break;
    case SB_PAGEDOWN:
        nDelta = m_nVPageSize;
        break;
    case SB_LINEDOWN:
        nDelta = LINESIZE;
        break;
    default: // Ignore other scroll bar messages
        return;
    }

    int nScrollPos = m_nVScrollPos + nDelta;
    int nMaxPos = m_nViewHeight - m_nVPageSize;

    if (nScrollPos < 0)
        nDelta = -m_nVScrollPos;
    else if (nScrollPos > nMaxPos)
        nDelta = nMaxPos - m_nVScrollPos;

    if (nDelta != 0) {
        m_nVScrollPos += nDelta;
        SetScrollPos(SB_VERT, m_nVScrollPos, TRUE);
        ScrollWindow(0, -nDelta);
    }
}
```

Màn hình kết quả như sau



1.2.8.2 Chương trình 2

Ruler.h

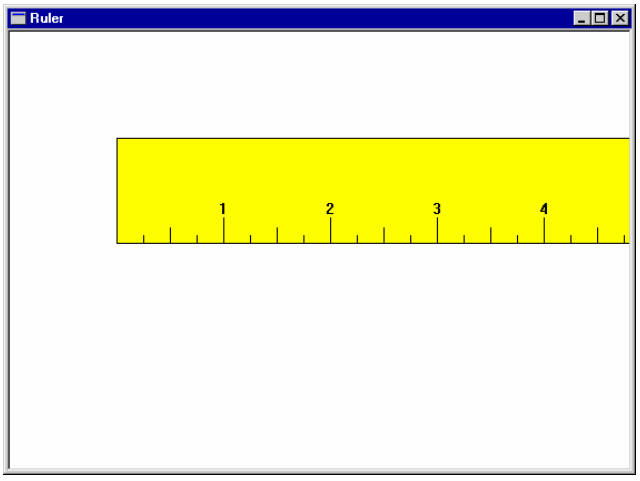
```
class CMyApp : public CWinApp
{
public:
    virtual BOOL InitInstance();
};

class CMainWindow : public CFrameWnd
{
public:
    CMainWindow();

protected:
    afx_msg void OnPaint();
    DECLARE_MESSAGE_MAP()
};
```

Ruler.cpp

```
#include <afxwin.h>
#include "Ruler.h"

CMyApp myApp;
//////////////////////////////////////////////////////////////////////////
// CMyApp member functions
BOOL CMyApp::InitInstance()
{
    m_pMainWnd = new CMainWindow;
    m_pMainWnd->ShowWindow(m_nCmdShow);
    m_pMainWnd->UpdateWindow();
    return TRUE;
}
//////////////////////////////////////////////////////////////////////////
// CMainWindow message map and member functions
BEGIN_MESSAGE_MAP(CMainWindow, CFrameWnd)
    ON_WM_PAINT()
END_MESSAGE_MAP()
CMainWindow::CMainWindow()
{
    Create(NULL, _T ("Ruler"));
}
void CMainWindow::OnPaint()
{
    CPaintDC dc(this);
```

```
    //
    // Initialize the device context.
    //
    dc.SetMapMode(MM_LOENGLISH);
    dc.SetTextAlign(TA_CENTER | TA_BOTTOM);
    dc.SetBkMode(TRANSPARENT);
    //
    // Draw the body of the ruler.
    //
    CBrush brush (RGB (255, 255, 0));
    CBrush* pOldBrush = dc.SelectObject(&brush);
    dc.Rectangle(100, -100, 1300, -200);
    dc.SelectObject(pOldBrush);
    //
    // Draw the tick marks and labels.
    //
    for (int i=125; i<1300; i+=25) {
        dc.MoveTo(i, -192);
        dc.LineTo(i, -200);
    }

    for (i=150; i<1300; i+=50) {
        dc.MoveTo(i, -184);
        dc.LineTo(i, -200);
    }

    for (i=200; i<1300; i+=100) {
        dc.MoveTo(i, -175);
        dc.LineTo(i, -200);

        CString string;
        string.Format(_T ("%d"), (i / 100) - 1);
        dc.TextOut(i, -175, string);
    }
}
```

Màn hình kết quả như sau:



1.3 XỬ LÝ BÀN PHÍM/CHUỘT TRONG ỨNG DỤNG WINDOWS

1.3.1 Vấn đề quan tâm

- Các thông điệp (*message*) và sự kiện của chuột, bàn phím

- Nhận biết và xử lý các thao tác liên quan đến các sự kiện của chuột, bàn phím.

1.3.2 Các sự kiện của chuột

MFC hỗ trợ thao tác với chuột bằng việc liên hệ các thông điệp và các sự kiện như:

Thông điệp	Macro kết hợp	Sự kiện	Hàm điều khiển
WM_LBUTTONDOWN	*ON_WM_LBUTTONDOWN*	Nút trái chuột được nhấn xuống.	OnLButtonDown
WM_LBUTTONUP	*ON_WM_LBUTTONUP*	Nút trái chuột được nhả ra.	OnLButtonUp
WM_LBUTTONDBLCLK	*ON_WM_LBUTTONDBLCLK*	Nút trái chuột được nhấn kép(double)	OnLButtonDblClk
WM_MBUTTONDOWN	*ON_WM_MBUTTONDOWN*	Nút giữa chuột được nhấn xuống.	OnMButtonDown
WM_MBUTTONUP	*ON_WM_MBUTTONUP*	Nút giữa chuột được nhả ra.	OnMButtonUp
WM_MBUTTONDBLCLK	*ON_WM_MBUTTONDBLCLK*	Nút giữa chuột được nhấn kép(double)	OnMButtonDblClk
WM_RBUTTONDOWN	*ON_WM_RBUTTONDOWN*	Nút phải chuột được nhấn xuống.	OnRButtonDown
WM_RBUTTONUP	*ON_WM_RBUTTONUP*	Nút phải chuột được nhả ra.	OnRButtonUp
WM_RBUTTONDBLCLK	*ON_WM_RBUTTONDBLCLK*	Nút phải chuột được nhấn kép(double)	OnRButtonDblClk
WM_MOUSEMOVE	*ON_WM_MOUSEMOVE*	Con trỏ chuột di chuyển trong vùng hiển thị của cửa sổ.	OnMouseMove

Các hàm điều khiển có dạng:

```
afx_msg void OnMsgName(UINT nFlags, CPoint point)
```

Trong đó nFlags(dùng để nhận biết nút bấm chuột và các phím Ctrl/Shift bởi phép toán &) có giá trị:

Mặt nạ (Mask)	Nhận biết khi
MK_LBUTTON	Nút trái chuột được nhấn
MK_MBUTTON	Nút giữa chuột được nhấn
MK_RBUTTON	Nút phải chuột được nhấn
MK_CONTROL	Phím Ctrl được nhấn
MK_SHIFT	Phím Shift được nhấn

Ví dụ:

Để nhận biết phím Ctrl có được nhấn kèm với chuột dùng:

```
if ((nFlags & MK_CONTROL) == MK_CONTROL) {…}
```

Ví dụ:

Kết hợp xử lý chuột và vẽ hình:

```
void CVd3aView::OnMouseMove(UINT nFlags, CPoint point)
{
```

```
	// TODO: Add your message handler code here and/or call default
	CDC *pDC = GetDC();
	CString szTextOut;
	szTextOut.Format("Current point(%d, %d)", point.x, point.y);
	pDC->TextOut(0, 0, szTextOut);
	if((nFlags & MK_CONTROL)==MK_CONTROL)
		MyDrawFunction(point);
	CView::OnMouseMove(nFlags, point);
}
void CVd3aView::OnLButtonDown(UINT nFlags, CPoint point)
{
	// TODO: Add your message handler code here and/or call default
	MyDrawFunction(point);
	CView::OnLButtonDown(nFlags, point);
}
void CVd3aView::MyDrawFunction(CPoint point)
{
	CDC *pDC = GetDC();
	if((m_PrevPoint.x == -1)&&(m_PrevPoint.y == -1))
		pDC->MoveTo(point);
	else
	{
		pDC->MoveTo(m_PrevPoint);
		pDC->LineTo(point);
	}
	m_PrevPoint = point;
}
```

1.3.3 Các sự kiện của bàn phím

MFC hỗ trợ thao tác với bàn phím bằng việc liên hệ các thông điệp và các sự kiện như sau:

Thông điệp	Macro kết hợp	Sự kiện	Hàm điều khiển
WM_KEYDOWN	*ON_WM_KEYDOWN*	Phím được nhấn xuống.	OnKeyDown
WM_KEYUP	*ON_WM_KEYUP*	Phím được nhả ra.	OnKeyUp
WM_SYSKEYDOWN	*ON_WM_SYSKEYDOWN*	Nút chức năng được nhấn xuống.	OnSysKeyDown
WM_SYSKEYUP	*ON_WM_SYSKEYUP*	Phím chức năng được nhả ra.	OnSysKeyUp

Các hàm điều khiển có dạng:

```
afx_msg void OnMsgName(UINT nChar, UINT nRepCnt, UINT nFlags)
```

Trong đó, ngoài các ký tự thông thường, nChar(là Virtual Key Code) còn có giá trị như:

Mã phím ảo	Phím tương ứng
VK_F1_VK_F12	Function keys F1_F12
VK_NUMPAD0_VK_NUMPAD9	Numeric keypad 0_9 with Num Lock on
VK_CANCEL	Ctrl-Break
VK_RETURN	Enter
VK_BACK	Backspace
VK_TAB	Tab
VK_CLEAR	Numeric keypad 5 with Num Lock off
VK_SHIFT	Shift
VK_CONTROL	Ctrl
VK_MENU	Alt

VK_PAUSE	Pause
VK_ESCAPE	Esc
VK_SPACE	Spacebar
VK_PRIOR	Page Up and PgUp
VK_NEXT	Page Down and PgDn
VK_END	End
VK_HOME	Home
VK_LEFT	Left arrow
VK_UP	Up arrow
VK_RIGHT	Right arrow
VK_DOWN	Down arrow
VK_SNAPSHOT	Print Screen
VK_INSERT	Insert and Ins
VK_DELETE	Delete and Del
VK_MULTIPLY	Numeric keypad *
VK_ADD	Numeric keypad +
VK_SUBTRACT	Numeric keypad -
VK_DECIMAL	Numeric keypad .
VK_DIVIDE	Numeric keypad /
VK_CAPITAL	Caps Lock
VK_NUMLOCK	Num Lock
VK_SCROLL	Scroll Lock
VK_LWIN	Left Windows key
VK_RWIN	Right Windows key
VK_APPS	Menu key

Và nFlags được dùng để nhận biết phím Alt có được nhấn kèm hay không.

Ngoài ra, có thể dùng hàm:

void OnChar(UINT nChar, UINT nRepCnt, UINT nFlags)

Ví dụ:

```
void CVd3bView::OnChar(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	// TODO: Add your message handler code here and/or call default
	if(((nChar >= _T('A')) &&(nChar <= _T('Z'))) ||
		((nChar >= _T('a')) &&(nChar <= _T('z')))) {
		// Display the character
		m_szInput += _T(nChar);
		MyDrawFunction(m_szInput);
	}
	else if((nChar >= _T('0')) &&(nChar <= _T('9'))) {
		// Display the character
		m_szInput += _T(nChar);
		MyDrawFunction(m_szInput);
	}
	else if(nChar == VK_SPACE) {
		// Display the character
		m_szInput += _T(' ');
		MyDrawFunction(m_szInput);
	}
	else if(nChar == VK_RETURN) {
		// Process the Enter key
		m_nLine++;
	}
	else if(nChar == VK_BACK) {
```

```
            // Process the Backspace key
            m_szInput = m_szInput.Left(m_szInput.GetLength()-1);
            MyDrawFunction(m_szInput);
      }
      CView::OnChar(nChar, nRepCnt, nFlags);
}
void CVd3bView::MyDrawFunction(CString szText)
{
      CDC *pDC = GetDC();
      CBrush *pCurrentBrush = pDC->GetCurrentBrush();
      CBrush *pNewBrush = new CBrush();
      pNewBrush->CreateSolidBrush(RGB(255, 255, 255));
      CRect rect;
      GetClientRect(&rect);
      pDC->SelectObject(pNewBrush);
      pDC->FillRect(rect, pNewBrush);
      pDC->TextOut(0, 20*m_nLine, szText);
}
```

1.4 CÁC LỚP MFC COLLECTION: ARRAY, LIST, MAP*, TYPE POINTER MAP*

1.4.1 Vấn đề quan tâm

- Tìm hiểu cách sử dụng các MFC collection classes

1.4.2 Array collection

MFC cung cấp một số lớp hỗ trợ việc sử dụng array thuận tiện hơn như:

Tên lớp	Kiểu dữ liệu
CArray	Lớp mảng tổng quát nhất
CByteArray	Lớp mảng kiểu số nguyên 8-bit -bytes(BYTEs)
CWordArray	Lớp mảng kiểu số nguyên 16-bit -words(WORDs)
CDWordArray	Lớp mảng kiểu số nguyên 32-bit -double words(DWORDs)
CUIntArray	Lớp mảng kiểu số nguyên không âm -unsigned integers(UINTs)
CStringArray	Lớp mảng kiểu chuỗi CCtrings
CPtrArray	Lớp mảng kiểu con trỏ không kiểu -void pointers
CObArray	Lớp mảng kiểu con trỏ đối tượng -CObject pointers

Chú ý:

Người dùng phải khai báo **#include "Afxtempl.h"** khi sử dụng các lớp này.

Ví dụ 1:

```
// Add 10 items.
CUIntArray array;
array.SetSize(10);
for(int i=0; i<10; i++)
array[i] = i + 1;
// Remove the item at index 0.
array.RemoveAt(0);
TRACE(_T("Count = %d\n"), array.GetSize()); // 9 left.
// Remove items 0, 1, and 2.
array.RemoveAt(0, 3);
TRACE(_T("Count = %d\n"), array.GetSize()); // 6 left.
// Empty the array.
array.RemoveAll();
TRACE(_T("Count = %d\n"), array.GetSize()); // 0 left.
```

Ví dụ 2:

```
CArray<CPoint, CPoint&> array;
```

```
// Populate the array, growing it as needed.
for(int i=0; i<10; i++)
  array.SetAtGrow(i, CPoint(i*10, 0));
// Enumerate the items in the array.
int nCount = array.GetSize();
for(i=0; i<nCount; i++) {
  CPoint point = array[i];
  TRACE(_T("x=%d, y=%d\n"), point.x, point.y);
}
```

Ví dụ 3:

```
class CProduct
{
public :
CString m_szProductName;
CString m_szCategoryName;
}
```

....

```
CArrar <CProduct, CProduct&> arrayProductList ;
CProduct oProduct ;
oProduct.m_szProductName = 'product 1';
oProduct.m_ szCategoryName = 'category 1';
arrayProductList.Add(oProduct) ;
oProduct.m_szProductName = 'product 2';
oProduct.m_ szCategoryName = 'category 2';
arrayProductList.Add(oProduct) ;
```

...

```
oProduct = arrayProductList.GetAt(0);
AfxMessageBox(oProduct.m_szProductName);
```

Ngoài ra, có thể khai báo một kiểu dữ liệu mới từ CArray như sau:

Ví dụ 4:

```
typedef CArray<UINT, UINT> CUIntArray;
```

Chú ý:

Định nghĩa mảng kiểu đối tượng cần phải có ký hiệu & trong thành phần thứ hai trong định nghĩa.

```
CArray <kiểu_đối_tượng, kiểu_đối_tượng&> array;
```

Định nghĩa mảng kiểu cơ sở không có ký hiệu & trong thành phần thứ hai trong định nghĩa.

```
CArray <kiểu_cơ_sở, kiểu_cơ_sở> array;
```

1.4.3 List collection

MFC cung cấp các lớp hỗ trợ truy xuất danh sách liên kết như sau:

Tên lớp	Kiểu dữ liệu
CList	Lớp danh sách liên kết tổng quát nhất
CObList	Lớp danh sách liên kết kiểu con trỏ đối tượng -CObject pointers
CPtrList	Lớp danh sách liên kết kiểu con trỏ không kiểu-void pointers
CStringList	Lớp danh sách liên kết kiểu CString

Chú ý:

Người dùng phải khai báo **#include "Afxtempl.h"** khi sử dụng các lớp này.

Ví dụ 1:

```
// Schools of the Southeastern Conference
TCHAR szSchools[10][20];
szSchools[0] = _T("Alabama"); szSchools[1] = _T("Arkansas");
szSchools[2] = _T("Florida"); szSchools[3] = _T("Georgia");
```

```
szSchools[4] = _T("Kentucky"); szSchools[5] = _T("Mississippi");
szSchools[6] = _T("Mississippi State"); szSchools[7] = _T("South Carolina");
szSchools[8] = _T("Tennessee"); szSchools[0] = _T("Vanderbilt") ;
CStringList list;
for(int i=0; i<10; i++)
  list.AddTail(szSchools[i]);
POSITION pos = list.GetHeadPosition();
while(pos != NULL) {
  CString string = list.GetNext(pos);
  TRACE(_T("%s\n"), string);
}
```

Ví dụ 2:

```
CList <CPoint, CPoint&> list;
// Populate the list.
for(int i=0; i<10; i++)
  list.AddTail(CPoint(i*10, 0));
// Enumerate the items in the list.
POSITION pos = list.GetHeadPosition();
While (pos != NULL) {
  CPoint point = list.GetNext(pos);
  TRACE(_T("x=%d, y=%d\n"), point.x, point.y);
}
```

1.4.4 Map

Một map, được xem như một dictionary, là một bảng gồm nhiều phần tử được xác định bởi khoá. MFC cung cấp tập các lớp hỗ trợ sử dụng map như:

Tên lớp	Mô tả
CMap	Lớp từ điển tổng quát nhất.
CMapWordToPtr	Lớp từ điển kiểu ánh xạ WORD với con trỏ
CMapPtrToWord	Lớp từ điển kiểu ánh xạ con trỏ với WORD
CMapPtrToPtr	Lớp từ điển kiểu ánh xạ con trỏ với con trỏ khác
CMapWordToOb	Lớp từ điển kiểu ánh xạ WORD với con trỏ đối tượng
CMapStringToOb	Lớp từ điển kiểu ánh xạ CString với con trỏ đối tượng
CMapStringToPtr	Lớp từ điển kiểu ánh xạ CString với con trỏ
CMapStringToString	Lớp từ điển kiểu ánh xạ CString với String

Chú ý:

Người dùng phải khai báo **#include "Afxtempl.h"** khi sử dụng các lớp này.

Ví dụ:

```
CMapStringToString map;
map[_T("Sunday")]   = _T("Dimanche");
map[_T("Monday")]   = _T("Lundi");
map[_T("Tuesday")]  = _T("Mardi");
map[_T("Wednesday")] = _T("Mercredi");
map[_T("Thursday")] = _T("Jeudi");
map[_T("Friday")]   = _T("Vendredi");
map[_T("Saturday")] = _T("Samedi");
POSITION pos = map.GetStartPosition();
while(pos != NULL) {
  CString strKey, strItem;
  map.GetNextAssoc(pos, strKey, strItem);
  TRACE(_T("Key=%s, Item=%s\n"), strKey, strItem);
}
```

1.4.5 Type pointer map

Tên lớp	Mô tả
CTypedPtrArray	Lớp quản lý mảng con trỏ

CTypedPtrList	Lớp quản lý danh sách liên kết con trỏ
CTypedPtrMap	Lớp quản lý từ điển con trỏ

Chú ý:

Người dùng phải khai báo **#include "Afxtempl.h"** khi sử dụng các lớp này.

Ví dụ:

```
CTypedPtrList <CObList, CLine*> list;
// Populate the list.
for(int i=0; i<10; i++) {
  int x = i * 10;
  CLine* pLine = new CLine(x, 0, x, 100);
  list.AddTail(pLine);
}
// Enumerate the items in the list.
POSITION pos = list.GetHeadPosition();
while(pos != NULL)
  CLine* pLine = list.GetNext(pos); // No casting!
```

1.5 TRUY XUẤT FILE (I/O) VÀ SERIALIZATION

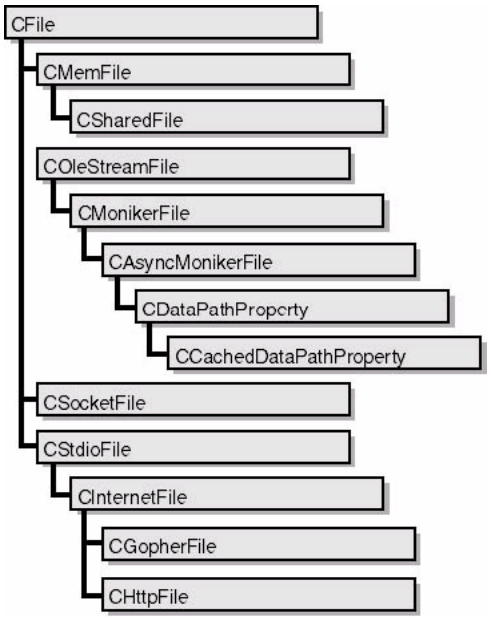
1.5.1 Vấn đề quan tâm

- Lớp CFile và thao tác truy xuất file
- Lớp CArchive các thao tác chuỗi hoá đối tượng lưu trữ.

1.5.2 Lớp CFile

MFC cung cấp lớp CFile hỗ trợ các thao tác truy xuất file như: tạo mới file, đọc/ghi file…

Tổ chức thừa kế lớp CFile và các lớp con như sau:



Việc truy xuất file liên quan đến chọn lựa "chia sẻ" quyền truy cập như:

Kiểu chia sẻ	Mô tả
CFile::shareDenyNone	Mở một file không ngăn cấm việc loại trừ
CFile::shareDenyRead	Cấm chương trình khác truy xuất đọc
CFile::shareDenyWrite	Cấm chương trình khác truy xuất ghi
CFile::shareExclusive	Cấm chương trình khác truy xuất đọc lẫn ghi

Và chọn lựa kiểu truy xuất như:

Kiểu truy xuất	Mô tả
CFile::modeReadWrite	Yêu cầu truy xuất đọc và ghi
CFile::modeRead	Yêu cầu truy xuất chỉ đọc
CFile::modeWrite	Yêu cầu truy xuất chỉ ghi

Ví dụ 1:

```
BYTE buffer[0x1000];
try
{
  CFile file(_T("C:\\MyDocument.txt"), CFile::modeReadWrite);
  DWORD dwBytesRemaining = file.GetLength();
  while(dwBytesRemaining)
  {
    DWORD dwPosition = file.GetPosition();
    UINT nBytesRead = file.Read(buffer, sizeof(buffer));
    ::CharLowerBuff((LPTSTR)buffer, nBytesRead);// chuyển sang chữ thường
    file.Seek(dwPosition, CFile::begin);
    file.Write(buffer, nBytesRead);
    dwBytesRemaining -= nBytesRead;
  }
}
catch(CFileException* e)
{
  e->ReportError();
  e->Delete();
}
file.Close();
```

Ví dụ 2:

Liệt kê tất cả các file trong thư mục hiện hành:

```
WIN32_FIND_DATA fd;
HANDLE hFind = ::FindFirstFile (_T ("*.*"), &fd);

if (hFind != INVALID_HANDLE_VALUE) {
    do {
        if (!(fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY))
            TRACE (_T ("%s\n"), fd.cFileName);
    } while (::FindNextFile (hFind, &fd));
    ::FindClose (hFind);
}
```

và liệt kê tất cả thư mục con trong thư mục hiện hành:

```
WIN32_FIND_DATA fd;
HANDLE hFind = ::FindFirstFile (_T ("*.*"), &fd);

if (hFind != INVALID_HANDLE_VALUE) {
    do {
        if (fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY)
            TRACE (_T ("%s\n"), fd.cFileName);
    } while (::FindNextFile (hFind, &fd));
    ::FindClose (hFind);
}
```

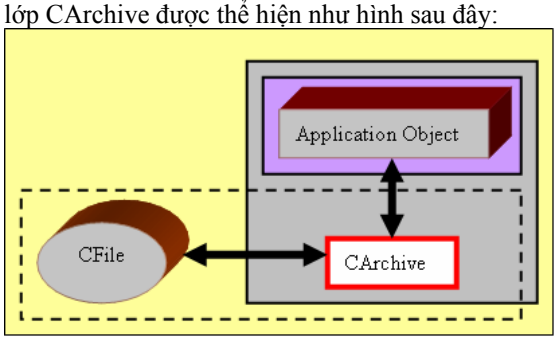
1.5.3 Chuỗi hoá và CArchive

Dù lớp CFile khá hữu ích trong quá trình xây dựng WinApp, nhưng MFC cung cấp lớp CArchive nhằm đơn giản hoá việc truy xuất với việc dùng toán tử >> và <<

Chuỗi hoá là một phần khá quan trọng trong lập trình MFC vì nó giúp chuyển đổi các đối tượng sang dạng dòng (*stream*) dữ liệu thuận lợi hơn trong quá trình lưu trữ/truy cập. Lớp CArchive được sử dụng trong hàm Serialize trên các đối tượng dữ liệu và tài liệu trong ứng dụng.

Trong hàm Serialize, để xác định đối tượng cần lưu trữ hiện tại đang được lưu xuống đĩa hay đang được đọc lên từ đĩa, cần sử dụng hàm thành phần **IsStoring()** và **IsLoading()**.

Sơ đồ tương tác và vai trò của lớp CArchive được thể hiện như hình sau đây:



Ví dụ 3:

```
void CVd3bDoc::Serialize(CArchive& ar)
{
	if(ar.IsStoring())
	{
		// TODO: add storing code here
		ar << m_MyVar;
	}
	else
	{
		// TODO: add loading code here
		ar >> m_MyVar;
	}
}
```

Hàm Serialize có thể được đặt trong bất cứ class nào nhằm xử lý thao tác lưu/đọc cho đối tượng đó. Để xử lý cho trường hợp này cho class (giả sử có tên CMyClass), cần thực thi theo trình tự sau:

- Cho lớp CMyClass này *kế thừa lớp CObject*.
- *Thêm macro DECLARE_SERIAL trong phần khai báo* của lớp CMyClass theo dạng sau: DECLARE_SERIAL(tên class thừa kế này). Ví dụ: DECLARE_SERIAL(CMyClass)
- *Thêm hàm Serialize* vào class CMyClass và khai báo lại hàm Serialize này để xử lý chuỗi hoá dữ liệu của lớp CMyClass
- *Thêm hàm sinh (constructor)* cho lớp CMyClass nếu nó chưa có.
- *Thêm macro IMPLEMENT_SERIAL trong phần thân* của lớp CMyClass theo dạng sau: IMPLEMENT_SERIAL(tên class thừa kế, tên class cha, số hiệu phiên bản). Ví dụ: IMPLEMENT_SERIAL(CMyClass, CObject, 1)

Ví dụ 4:

Bước 0:

```
class CLine
{
protected:
  CPoint m_ptFrom;
  CPoint m_ptTo;

public:
  CLine(CPoint from, CPoint to) { m_ptFrom = from; m_ptTo = to; }
};
```

Bước 1-2:

```
class CLine : public CObject
{
DECLARE_SERIAL(CLine)

protected:
  CPoint m_ptFrom;
  CPoint m_ptTo;
```

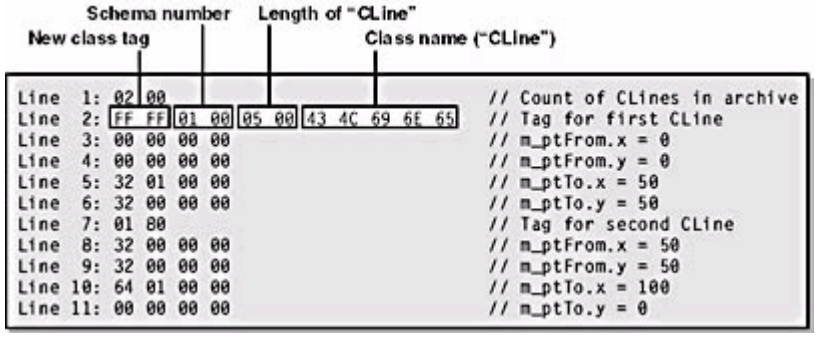
```
public:
  CLine() {} // Required!
  CLine(CPoint from, CPoint to) { m_ptFrom = from; m_ptTo = to; }
  void Serialize(CArchive& ar);
};
```

Bước 3:

```
void CLine::Serialize(CArchive& ar)
{
  CObject::Serialize(ar);
  if(ar.IsStoring())
    ar << m_ptFrom << m_ptTo;
  else // Loading, not storing
    ar >> m_ptFrom >> m_ptTo;
}
```

Bước 5a:

```
_AFX_INLINE CArchive& AFXAPI operator<<(CArchive& ar,
  const CObject* pOb)
  { ar.WriteObject(pOb); return ar; }
Bước 5b:
_AFX_INLINE CArchive& AFXAPI operator>>(CArchive& ar, CObject*& pOb)
  { pOb = ar.ReadObject(NULL); return ar; }

_AFX_INLINE CArchive& AFXAPI operator>>(CArchive& ar,
  const CObject*& pOb)
  { pOb = ar.ReadObject(NULL); return ar; }
```



Và có thể sử dụng để lưu trữ lớp CLine như sau:

Lưu xuống:

```
// Create two CLines and initialize an array of pointers.
CLine line1(CPoint(0, 0), CPoint(50, 50));
CLine line2(CPoint(50, 50), CPoint(100, 0));
CLine* pLines[2] = { &line1, &line2 };
int nCount = 2;

// Serialize the CLines and the CLine count.
for(int i=0; i<nCount; i++)
  ar << pLines[i];
// Lưu thêm số lượng đối tượng
ar << nCount;
```

Đọc lên:

```
int nCount;
ar >> nCount;
CLine* pLines = new CLine[nCount];
for(int i=0; i<nCount; i++)
```

```
  ar >> pLines[i];
// đọc lên số phần tử thực có
ar >> nCount;
```

Và cách thực thi khác:

```
// Serialize.
CLine line(CPoint(0, 0), CPoint(100, 50));
line.Serialize(ar);

// Deserialize.
CLine line;
line.Serialize(ar);
```

Ví dụ 5:

- Ghi vào file thông qua bộ xử lý chuỗi hoá:

```
try
{
	CFile myFile("C:\\myfile.dat", CFile::modeCreate  | CFile::modeWrite);
	// Create a storing archive.
	CArchive arStore(&myFile, CArchive::store);
	////////////////////////////////////
	CString szSize;
	szSize.Format("%d", (m_arrDSSV.GetSize()+1));
	arStore.Write(szSize, 10);
	////////////////////////////////////
	int nCount;
	////////////////////////////////////
	for (nCount=0; nCount < m_arrDSSV.GetSize(); nCount++)
	{
		// Write the object to the archive
		arStore.WriteObject( m_arrDSSV[nCount] );
	}
	// Close the storing archive
	arStore.Close();
	return TRUE;
}
catch(CException *e)
{
	return FALSE;
}
```

- đọc từ file lên thông qua bộ xử lý phi chuỗi hoá

```
try
{
	////////////////////////////////////////////////
	XoaNoiDung();
	////////////////////////////////////////////////
	CFile myFile("C:\\myfile.dat", CFile::modeRead);
	// Create a loading archive.
	myFile.SeekToBegin();
	CArchive arLoad(&myFile, CArchive::load);
	CSinhVien* pSV;
	////////////////////////////////
	char szSize[10];
	arLoad.Read(szSize, 10);
	int nSize = (int) atoi(szSize);
	int nCount;
```

```
		////////////////////////////////
		for (nCount=0; nCount < nSize; nCount++)
		{
			// Verify the object is in the archive.
			pSV = (CSinhVien*)arLoad.ReadObject(RUNTIME_CLASS(CSinhVien));
			if (pSV)
				m_arrDSSV.Add(pSV);
		}
		////////////////////////////////
		arLoad.Close();
		return TRUE;
	}
	catch(CException *e)
	{
		return FALSE;
	}
```

1.6 CÁC LỚP MFC CỦA CÁC ĐIỀU KHIỂN WINDOWS.

1.6.1 Vấn đề quan tâm

- Tính chất/hoạt động và xử lý/thao tác với các điều khiển Windows (*Windows control*)

1.6.2 Các loại điều khiển

MFC cung cấp các lớp đại diện cho các Windows control như:

Kiểu điều khiển	WNDCLASS	MFC Class
Buttons	"BUTTON"	CButton
List boxes	"LISTBOX"	CListBox
Edit controls	"EDIT"	CEdit
Combo boxes	"COMBOBOX"	CComboBox
Scroll bars	"SCROLLBAR"	CScrollBar
Static controls	"STATIC"	CStatic

Đồng thời với việc tạo các điều khiển này thông qua thanh công cụ trong chế độ Resource View, người dùng cũng có thể tạo các điều khiển này bằng hàm thành viên **Create**.

Ví dụ 1:

```
CButton myButton1, myButton2, myButton3, myButton4;

// Create a push button.
myButton1.Create(_T("My button"), WS_CHILD|WS_VISIBLE|BS_PUSHBUTTON,
  CRect(10,10,100,30), pParentWnd, 1);

// Create a radio button.
myButton2.Create(_T("My button"), WS_CHILD|WS_VISIBLE|BS_RADIOBUTTON,
  CRect(10,40,100,70), pParentWnd, 2);

// Create an auto 3-state button.
myButton3.Create(_T("My button"), WS_CHILD|WS_VISIBLE|BS_AUTO3STATE,
  CRect(10,70,100,100), pParentWnd, 3);

// Create an auto check box.
myButton4.Create(_T("My button"), WS_CHILD|WS_VISIBLE|BS_AUTOCHECKBOX,
  CRect(10,100,100,130), pParentWnd, 4);
```

Hay:

```
extern CWnd* pParentWnd;
// The pointer to my list box.
extern CListBox* pmyListBox;
```

```
pmyListBox->Create(WS_CHILD|WS_VISIBLE|LBS_STANDARD|WS_HSCROLL,
  CRect(10,10,200,200), pParentWnd, 1);
Và:
CEdit* pEdit = new CEdit;
  pEdit->Create(ES_MULTILINE | WS_CHILD | WS_VISIBLE | WS_TABSTOP | WS_BORDER,
   CRect(10, 10, 100, 100), this, 1);
```

Chú ý:

- Nên sử dụng các biến con trỏ để đại diện cho các điều hiển để tạo các điều khiển trong thời gian thực thi.
- Có thể xử lý việc tạo các điều khiển này trong thời gian thực thi bằng cách đặt chúng vào trong các hàm thành viên ::OnInitialUpdate() và hàm khởi tạo ::PreCreateWindow()

1.6.3 Loại CButton

Kiểu	Mô tả
BS_PUSHBUTTON	Tạo một điều khiển nút nhấn
BS_DEFPUSHBUTTON	Tạo một điều khiển nút nhấn và sử dụng trong hộp thoại với trạng thái mặc định khi nhấn phím Enter
BS_CHECKBOX	Tạo một điều khiển check box
BS_AUTOCHECKBOX	Tạo một điều khiển check box tự động chọn và bỏ chọn khi được bấm vào
BS_3STATE	Tạo một điều khiển check box 3 trạng thái.
BS_AUTO3STATE	Tạo một điều khiển check box 3 trạng thái tự động chọn và bỏ chọn khi được bấm vào
BS_RADIOBUTTON	Tạo một điều khiển radio button
BS_AUTORADIOBUTTON	Tạo một điều khiển radion button tự động chọn và bỏ chọn khi được bấm vào
BS_GROUPBOX	Tạo một nhóm các điều khiển

Và các kiểu riêng phần:

Kiểu	Mô tả
BS_LEFTTEXT	Đặt văn bản của các điều khiển radio button hay check box nằm bên trái của điều khiển
BS_RIGHTBUTTON	Tương tự như BS_LEFTTEXT nhưng về phía phải
BS_LEFT	Canh lề trái văn bản trong điều khiển
BS_CENTER	Canh lề giữa văn bản trong điều khiển
BS_RIGHT	Canh lề phải văn bản trong điều khiển
BS_TOP	Canh lề phía đỉnh trên văn bản trong điều khiển
BS_VCENTER	Canh lề giữa văn bản trong điều khiển theo chiều dọc
BS_BOTTOM	Canh lề phía dưới văn bản trong điều khiển
BS_MULTILINE	Xác định chế độ nhiều dòng

1.7 DIALOG BOX, COMMON DIALOG VÀ PROPERTY SHEET.

1.7.1 Vấn đề quan tâm

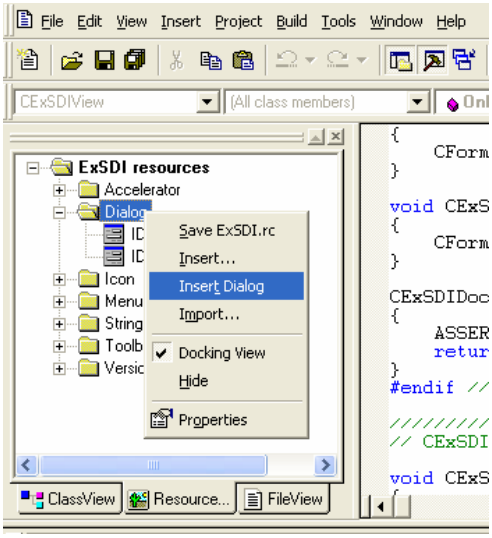
- Tính chất/hoạt động và xử lý/thao tác với các Dialog
- Cách tạo PropertySheet/PropertyPage

1.7.2 Hộp thoại (dialog)

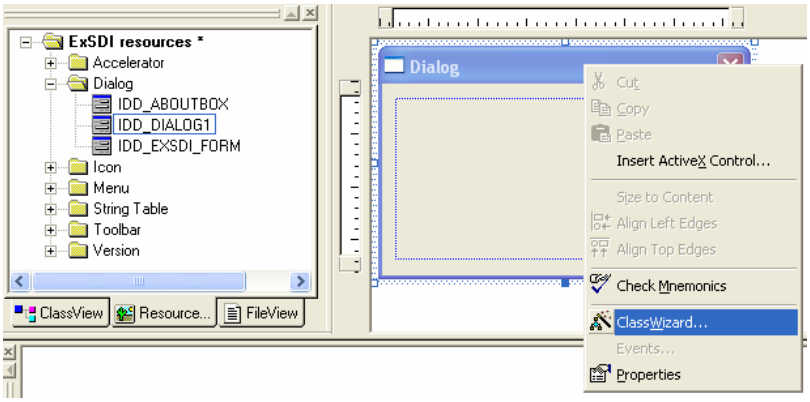
Có thể tạo các hộp thoại tùy ý bằng cách kết hợp giữa việc tạo Dialog form (trong ResourceView) và 1 class liên kết (tạo bởi ClassWizard)

Việc tạo này theo trình tự sau đây:

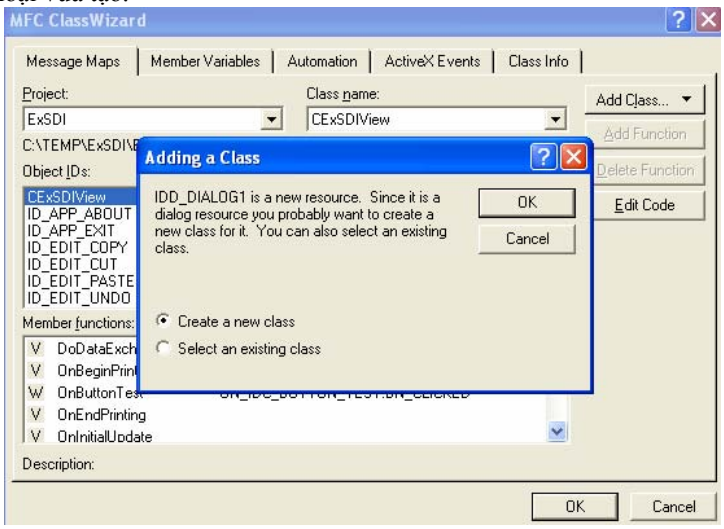
- Bước 1:



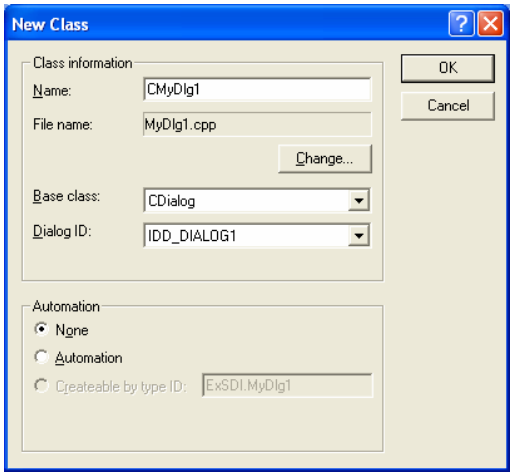
- Bước 2:



- Bước 3: Có thể cập nhật ID của hộp thoại mới thêm này, và chọn Create New Class để tạo một lớp liên kết với hộp thoại vừa tạo.



nhập thông tin cần thiết:



- Bước 4:

Tại file cần sử dụng hộp thoại này, thêm 1 hàng

```
#include "tên_file.h"
```

- Bước 5:

Tạo biến đại diện cho hộp thoại này và gọi hàm **DoModal()** của hộp thoại để kích hoạt hộp thoại hiển thị lên.

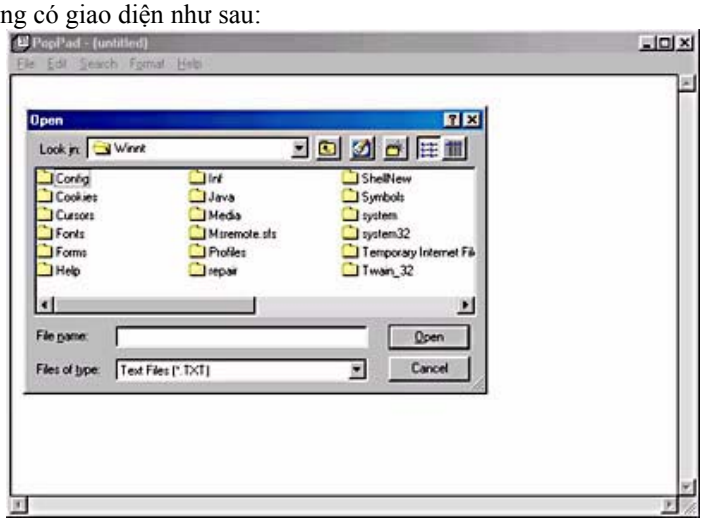
1.7.3 Các hộp thoại thông dụng (Common Dialog Classes)

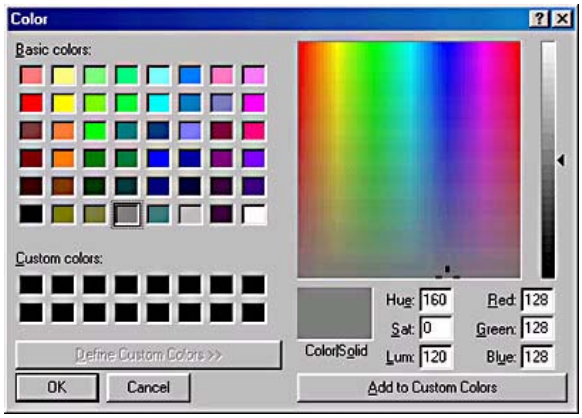
Các hộp thoại thông dụng như FontDialog, ColorDialog, FileDialog… được MFC cung cấp các class đại diện giúp việc sử dụng dễ dàng hơn.

MFC hỗ trợ các lớp đại diện cho các hộp thoại này như sau:

Tên lớp	Kiểu hộp thoại
CFileDialog	Hộp thoại mở và lưu trữ (Open and Save As dialog boxes)
CPrintDialog	Hộp thoại in ấn và cài đặt in ấn (Print and Print Setup dialog boxes)
CPageSetupDialog	Hộp thoại cài đặt trang (Page Setup dialog boxes)
CFindReplaceDialog	Hộp thoại tìm kiếm và thay thế (Find and Replace dialog boxes)
CColorDialog	Hộp thoại chọn màu (Color dialog boxes)
CFontDialog	Hộp thoại chọn phông chữ (Font dialog boxes)

Các hộp thoại thông dụng có giao diện như sau:





Ví dụ 1:

```
TCHAR szFilters[] = _T("Text files(*.txt)¦*.txt¦All files *.*)¦*.*¦¦");

CFileDialog
    dlg(TRUE,_T("txt"),_T("*.txt"),OFN_FILEMUSTEXIST¦OFN_HIDEREADONLY,zFilters);

if(dlg.DoModal() == IDOK) {
  filename = dlg.GetPathName();
  // Open the file and read it.
}
```

Ví dụ 2:

```
void CChildView::OnFileSaveAs()
{
CFileDialog
    dlg(FALSE,_T("phn"),m_strPathName,
          OFN_OVERWRITEPROMPT¦OFN_PATHMUSTEXIST¦OFN_HIDEREADONLY,
          m_szFilters);

if(dlg.DoModal() == IDOK)
{
       if(SaveFile(dlg.GetPathName()))
           m_strPathName = dlg.GetPathName();
}
```

Ví dụ 3:

```
CColorDialog dlg(RGB(255, 0, 0), CC_FULLOPEN);
if(dlg.DoModal() == IDOK)
{
       ////////////////////////////////////
       AfxMessageBox("Chon mau", 0, 0);
       m_oColor = dlg.GetColor();
}
```

1.7.4 Property Sheet/Property Page

Property Sheet còn gọi là tab, Property Page còn gọi là page.

PropertySheet được sử dụng nhằm cung cấp nhiều sự chọn lựa cho người dùng bằng cách tích hợp nhiều chọn lựa – mỗi chọn lựa là một hộp thoại (*property page*).

Để tạo Property Sheet/Property Page, cần làm theo các bước sau:

- Xác định số page (property page) cần có trong một property sheet
- Tạo (chọn) các class dự định dùng làm các page hiển thị trong property page – **kế thừa từ class CPropertyPage**.
- Thêm các control và các biến liên kết vào trong từng page và class liên kết ở bước trên.
- Tạo class mới cho property sheet bởi việc **kế thừa từ CPropertySheet class**

- Kích hoạt hàmDoModal()

Ví dụ 4:

- Tạo:

```
class CFirstPage : public CPropertyPage
{
public:
  CFirstPage() : CPropertyPage(IDD_FIRSTPAGE) {};
  // Declare CFirstPage's data members here.

protected:
  virtual void DoDataExchange(CDataExchange*);
};
class CSecondPage : public CPropertyPage
{
public:
  CSecondPage() : CPropertyPage(IDD_SECONDPAGE) {};
  // Declare CSecondPage's data members here.

protected:
  virtual void DoDataExchange(CDataExchange*);
};
class CMyPropertySheet : public CPropertySheet
{
public:
  CFirstPage m_firstPage;     // First page
  CSecondPage m_secondPage;    // Second page

  // Constructor adds the pages automatically.
  CMyPropertySheet(LPCTSTR pszCaption,
    CWnd* pParentWnd = NULL) :
    CPropertySheet(pszCaption, pParentWnd, 0)
  {
    AddPage(&m_firstPage);
    AddPage(&m_secondPage);
  }
};
```

- Sử dụng:

```
CMyPropertySheet ps(_T("Properties"));
ps.DoModal();
```

- Nếu muốn nhúng property sheet vào của sổ chương trình hiện hành thì cập nhập bởi các lệnh

```
CMyPropertySheet *ps = new CMyPropertySheet(_T("Properties"));
ps->Create(this, WS_CHILD);
ps->ShowWindow(SW_SHOW);
ps->MoveWindow(0,0,800,600);
```

Ví dụ tổng hợp:

MainFrm.h

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

#if !defined(AFX_MAINFRM_H__9CE2B4A8_9067_11D2_8E53_006008A82731__INCLUDED_)
```

```
#define AFX_MAINFRM_H__9CE2B4A8_9067_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "ChildView.h"

class CMainFrame : public CFrameWnd
{
public:
    CMainFrame();
protected:
    DECLARE_DYNAMIC(CMainFrame)
// Attributes
public:
// Operations
public:
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMainFrame)
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    virtual BOOL OnCmdMsg(UINT nID, int nCode, void* pExtra,
        AFX_CMDHANDLERINFO* pHandlerInfo);
    //}}AFX_VIRTUAL
// Implementation
public:
    virtual ~CMainFrame();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
    CChildView    m_wndView;
// Generated message map functions
protected:
    //{{AFX_MSG(CMainFrame)
    afx_msg void OnSetFocus(CWnd *pOldWnd);
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    //}}AFX_MSG
    afx_msg LRESULT OnApply (WPARAM wParam, LPARAM lParam);
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
// !defined(AFX_MAINFRM_H__9CE2B4A8_9067_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//
#include "stdafx.h"
#include "PropDemo.h"
#include "MainFrm.h"
#ifdef _DEBUG
#define new DEBUG_NEW
```

```
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CMainFrame
IMPLEMENT_DYNAMIC(CMainFrame, CFrameWnd)
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
    ON_WM_SETFOCUS()
    ON_WM_CREATE()
    //}}AFX_MSG_MAP
    ON_MESSAGE (WM_USER_APPLY, OnApply)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction
CMainFrame::CMainFrame()
{
}
CMainFrame::~CMainFrame()
{
}
BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CFrameWnd::PreCreateWindow(cs) )
        return FALSE;
    cs.dwExStyle &= ~WS_EX_CLIENTEDGE;
    cs.lpszClass = AfxRegisterWndClass(0);
    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
void CMainFrame::OnSetFocus(CWnd* pOldWnd)
{
    // forward focus to the view window
    m_wndView.SetFocus();
}
BOOL CMainFrame::OnCmdMsg(UINT nID, int nCode, void* pExtra,
    AFX_CMDHANDLERINFO* pHandlerInfo)
{
    // let the view have first crack at the command
    if (m_wndView.OnCmdMsg(nID, nCode, pExtra, pHandlerInfo))
        return TRUE;
    // otherwise, do default handling
```

```
    return CFrameWnd::OnCmdMsg(nID, nCode, pExtra, pHandlerInfo);
}
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
        return -1;

    if (!m_wndView.Create(NULL, NULL, AFX_WS_DEFAULT_VIEW,
        CRect(0, 0, 0, 0), this, AFX_IDW_PANE_FIRST, NULL))
        return -1;
    return 0;
}
LRESULT CMainFrame::OnApply (WPARAM wParam, LPARAM lParam)
{
    m_wndView.SendMessage (WM_USER_APPLY, wParam, lParam);
    return 0;
}
```

ChildView.h

```
// ChildView.h : interface of the CChildView class
//
/////////////////////////////////////////////////////////////////////////////

#if !defined(AFX_CHILDVIEW_H__9CE2B4AA_9067_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_CHILDVIEW_H__9CE2B4AA_9067_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
/////////////////////////////////////////////////////////////////////////////
// CChildView window
class CChildView : public CWnd
{
// Construction
public:
    CChildView();
// Attributes
public:
// Operations
public:
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CChildView)
    protected:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    //}}AFX_VIRTUAL
// Implementation
public:
    virtual ~CChildView();
    // Generated message map functions
protected:
    int m_nUnits;
    int m_nHeight;
    int m_nWidth;
    int m_nColor;
    //{{AFX_MSG(CChildView)
```

```
    afx_msg void OnPaint();
    afx_msg void OnFileProperties();
    //}}AFX_MSG
    afx_msg LRESULT OnApply (WPARAM wParam, LPARAM lParam);
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
//!defined(AFX_CHILDVIEW_H__9CE2B4AA_9067_11D2_8E53_006008A82731__INCLUDED_)
```

ChildView.cpp

```
// ChildView.cpp : implementation of the CChildView class
//
#include "stdafx.h"
#include "PropDemo.h"
#include "ChildView.h"
#include "MyPropertySheet.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CChildView
CChildView::CChildView()
{
    m_nWidth = 4;
    m_nHeight = 2;
    m_nUnits = 0;
    m_nColor = 0;
}
CChildView::~CChildView()
{
}
BEGIN_MESSAGE_MAP(CChildView,CWnd )
    //{{AFX_MSG_MAP(CChildView)
    ON_WM_PAINT()
    ON_COMMAND(ID_FILE_PROPERTIES, OnFileProperties)
    //}}AFX_MSG_MAP
    ON_MESSAGE (WM_USER_APPLY, OnApply)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CChildView message handlers
BOOL CChildView::PreCreateWindow(CREATESTRUCT& cs)
{
    if (!CWnd::PreCreateWindow(cs))
        return FALSE;

    cs.dwExStyle |= WS_EX_CLIENTEDGE;
    cs.style &= ~WS_BORDER;
    cs.lpszClass = AfxRegisterWndClass(CS_HREDRAW|CS_VREDRAW|CS_DBLCLKS,
        ::LoadCursor(NULL, IDC_ARROW), HBRUSH(COLOR_WINDOW+1), NULL);
```

```
    return TRUE;
}
void CChildView::OnPaint()
{
    CPaintDC dc(this); // Device context for painting.

    CBrush brush (CColorPage::m_clrColors[m_nColor]);
    CBrush* pOldBrush = dc.SelectObject (&brush);

    switch (m_nUnits) {
    case 0: // Inches.
        dc.SetMapMode (MM_LOENGLISH);
        dc.Ellipse (0, 0, m_nWidth * 100, -m_nHeight * 100);
        break;
    case 1: // Centimeters.
        dc.SetMapMode (MM_LOMETRIC);
        dc.Ellipse (0, 0, m_nWidth * 100, -m_nHeight * 100);
        break;
    case 2: // Pixels.
        dc.SetMapMode (MM_TEXT);
        dc.Ellipse (0, 0, m_nWidth, m_nHeight);
        break;
    }
    dc.SelectObject (pOldBrush);
}
void CChildView::OnFileProperties()
{
    CMyPropertySheet ps (_T ("Properties"));
    ps.m_sizePage.m_nWidth = m_nWidth;
    ps.m_sizePage.m_nHeight = m_nHeight;
    ps.m_sizePage.m_nUnits = m_nUnits;
    ps.m_colorPage.m_nColor = m_nColor;
    if (ps.DoModal () == IDOK) {
        m_nWidth = ps.m_sizePage.m_nWidth;
        m_nHeight = ps.m_sizePage.m_nHeight;
        m_nUnits = ps.m_sizePage.m_nUnits;
        m_nColor = ps.m_colorPage.m_nColor;
        Invalidate ();
    }
}
LRESULT CChildView::OnApply (WPARAM wParam, LPARAM lParam)
{
    ELLPROP* pep = (ELLPROP*) lParam;
    m_nWidth = pep->nWidth;
    m_nHeight = pep->nHeight;
    m_nUnits = pep->nUnits;
    m_nColor = pep->nColor;
    Invalidate ();
    return 0;
}
```

MyPropertySheet.h

```
#if
!defined(AFX_MYPROPERTYSHEET_H__418271A3_90D4_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MYPROPERTYSHEET_H__418271A3_90D4_11D2_8E53_006008A82731__INCLUDED_
```

```
#include "SizePage.h"    // Added by ClassView
#include "ColorPage.h"    // Added by ClassView
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// MyPropertySheet.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CmyPropertySheet
class CMyPropertySheet : public CPropertySheet
{
    DECLARE_DYNAMIC(CMyPropertySheet)

// Construction
public:
    CMyPropertySheet(UINT nIDCaption, CWnd* pParentWnd = NULL,
        UINT iSelectPage = 0);
    CMyPropertySheet(LPCTSTR pszCaption, CWnd* pParentWnd = NULL,
        UINT iSelectPage = 0);
// Attributes
public:
    CColorPage m_colorPage;
    CSizePage m_sizePage;
// Operations
public:
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMyPropertySheet)
    //}}AFX_VIRTUAL
// Implementation
public:
    virtual ~CMyPropertySheet();
    // Generated message map functions
protected:
    //{{AFX_MSG(CMyPropertySheet)
        // NOTE - the ClassWizard will add and remove
        // member functions here.
    //}}AFX_MSG
    afx_msg void OnApply ();
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
// !defined(
//  AFX_MYPROPERTYSHEET_H__418271A3_90D4_11D2_8E53_006008A82731__INCLUDED_)
```

MyPropertySheet.cpp

```
// MyPropertySheet.cpp : implementation file
//
#include "stdafx.h"
#include "PropDemo.h"
#include "MyPropertySheet.h"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CMyPropertySheet
IMPLEMENT_DYNAMIC(CMyPropertySheet, CPropertySheet)

CMyPropertySheet::CMyPropertySheet(UINT nIDCaption, CWnd* pParentWnd,
    UINT iSelectPage) : CPropertySheet(nIDCaption, pParentWnd, iSelectPage)
{
    AddPage (&m_sizePage);
    AddPage (&m_colorPage);
}
CMyPropertySheet::CMyPropertySheet(LPCTSTR pszCaption, CWnd* pParentWnd,
    UINT iSelectPage) : CPropertySheet(pszCaption, pParentWnd, iSelectPage)
{
    AddPage (&m_sizePage);
    AddPage (&m_colorPage);
}
CMyPropertySheet::~CMyPropertySheet()
{
}
BEGIN_MESSAGE_MAP(CMyPropertySheet, CPropertySheet)
    //{{AFX_MSG_MAP(CMyPropertySheet)
        // NOTE - the ClassWizard will add and remove mapping macros here.
    //}}AFX_MSG_MAP
    ON_BN_CLICKED (ID_APPLY_NOW, OnApply)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CMyPropertySheet message handlers
void CMyPropertySheet::OnApply ()
{
    GetActivePage ()->UpdateData (TRUE);

    ELLPROP ep;
    ep.nWidth = m_sizePage.m_nWidth;
    ep.nHeight = m_sizePage.m_nHeight;
    ep.nUnits = m_sizePage.m_nUnits;
    ep.nColor = m_colorPage.m_nColor;

    GetParent ()->SendMessage (WM_USER_APPLY, 0, (LPARAM) &ep);

    m_sizePage.SetModified (FALSE);
    m_colorPage.SetModified (FALSE);
}
```

SizePage.h

```
#if !defined(AFX_SIZEPAGE_H__418271A1_90D4_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_SIZEPAGE_H__418271A1_90D4_11D2_8E53_006008A82731__INCLUDED_

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

```
// SizePage.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CSizePage dialog
class CSizePage : public CPropertyPage
{
    DECLARE_DYNCREATE(CSizePage)

// Construction
public:
    CSizePage();
    ~CSizePage();

// Dialog Data
    //{{AFX_DATA(CSizePage)
    enum { IDD = IDD_SIZE_PAGE };
    int        m_nWidth;
    int        m_nHeight;
    int        m_nUnits;
    //}}AFX_DATA
// Overrides
    // ClassWizard generate virtual function overrides
    //{{AFX_VIRTUAL(CSizePage)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL
// Implementation
protected:
    // Generated message map functions
    //{{AFX_MSG(CSizePage)
        // NOTE: the ClassWizard will add member functions here
    //}}AFX_MSG
    afx_msg void OnChange ();
   DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
//!defined(AFX_SIZEPAGE_H__418271A1_90D4_11D2_8E53_006008A82731__INCLUDED_)
```

SizePage.cpp

```
// SizePage.cpp : implementation file
//
#include "stdafx.h"
#include "PropDemo.h"
#include "SizePage.h"

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

/////////////////////////////////////////////////////////////////////////////
// CSizePage property page
```

```
IMPLEMENT_DYNCREATE(CSizePage, CPropertyPage)

CSizePage::CSizePage() : CPropertyPage(CSizePage::IDD)
{
    //{{AFX_DATA_INIT(CSizePage)
    m_nWidth = 0;
    m_nHeight = 0;
    m_nUnits = -1;
    //}}AFX_DATA_INIT
}
CSizePage::~CSizePage()
{
}
void CSizePage::DoDataExchange(CDataExchange* pDX)
{
    CPropertyPage::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CSizePage)
    DDX_Text(pDX, IDC_WIDTH, m_nWidth);
    DDV_MinMaxInt(pDX, m_nWidth, 1, 128);
    DDX_Text(pDX, IDC_HEIGHT, m_nHeight);
    DDV_MinMaxInt(pDX, m_nHeight, 1, 128);
    DDX_Radio(pDX, IDC_INCHES, m_nUnits);
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CSizePage, CPropertyPage)
    //{{AFX_MSG_MAP(CSizePage)
        // NOTE: the ClassWizard will add message map macros here
    //}}AFX_MSG_MAP
    ON_EN_CHANGE (IDC_WIDTH, OnChange)
    ON_EN_CHANGE (IDC_HEIGHT, OnChange)
    ON_BN_CLICKED (IDC_INCHES, OnChange)
    ON_BN_CLICKED (IDC_CENTIMETERS, OnChange)
    ON_BN_CLICKED (IDC_PIXELS, OnChange)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CSizePage message handlers
void CSizePage::OnChange ()
{
    SetModified (TRUE);
}
```

ColorPage.h

```
#if !defined(AFX_COLORPAGE_H__418271A2_90D4_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_COLORPAGE_H__418271A2_90D4_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// ColorPage.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CColorPage dialog
class CColorPage : public CPropertyPage
{
    DECLARE_DYNCREATE(CColorPage)
```

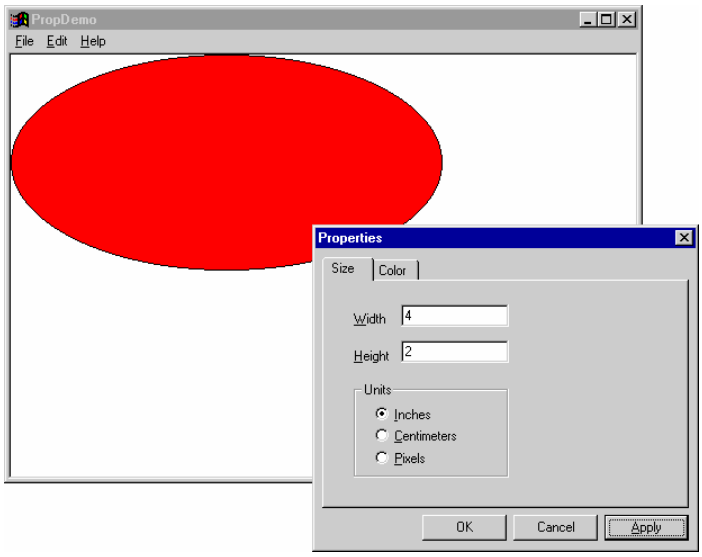
```
// Construction
public:
    CColorPage();
    ~CColorPage();
    static const COLORREF m_clrColors[3];
// Dialog Data
    //{{AFX_DATA(CColorPage)
    enum { IDD = IDD_COLOR_PAGE };
    int        m_nColor;
    //}}AFX_DATA
// Overrides
    // ClassWizard generate virtual function overrides
    //{{AFX_VIRTUAL(CColorPage)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL
// Implementation
protected:
    // Generated message map functions
    //{{AFX_MSG(CColorPage)
        // NOTE: the ClassWizard will add member functions here
    //}}AFX_MSG
    afx_msg void OnChange ();
    DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
//defined(AFX_COLORPAGE_H__418271A2_90D4_11D2_8E53_006008A82731__INCLUDED_)
```

ColorPage.cpp

```
// ColorPage.cpp : implementation file
//
#include "stdafx.h"#include "PropDemo.h"
#include "ColorPage.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CColorPage property page
IMPLEMENT_DYNCREATE(CColorPage, CPropertyPage)

const COLORREF CColorPage::m_clrColors[3] = {
    RGB (255,   0,   0),    // Red
    RGB (  0, 255,   0),    // Green
    RGB (  0,   0, 255)     // Blue
};
CColorPage::CColorPage() : CPropertyPage(CColorPage::IDD)
{
    //{{AFX_DATA_INIT(CColorPage)
    m_nColor = -1;
```

```
    //}}AFX_DATA_INIT
}
CColorPage::~CColorPage()
{
}
void CColorPage::DoDataExchange(CDataExchange* pDX)
{
    CPropertyPage::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CColorPage)
    DDX_Radio(pDX, IDC_RED, m_nColor);
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CColorPage, CPropertyPage)
    //{{AFX_MSG_MAP(CColorPage)
        // NOTE: the ClassWizard will add message map macros here
    //}}AFX_MSG_MAP
    ON_BN_CLICKED (IDC_RED, OnChange)
    ON_BN_CLICKED (IDC_GREEN, OnChange)
    ON_BN_CLICKED (IDC_BLUE, OnChange)
END_MESSAGE_MAP()
////////////////////////////////////////////////////////////////////////////
// CColorPage message handlers
void CColorPage::OnChange ()
{
    SetModified (TRUE);
}
```

Màn hình kết quả như sau:



1.8 Một số điều khiển trong Windows 9.x*

1.8.1 Các loại điều khiển

Các điều khiển được phát sinh thêm trong hệ điều hành Windows 95 (hay các hệ điều hành Windows mới hơn) có các lớp tương ứng trong MFC như sau:

Control Type	WNDCLASS	WNDCLASS Alias	MFC Class
Animation	"SysAnimate32"	ANIMATE_CLASS	*CAnimateCtrl*
ComboBoxEx*	"ComboBoxEx32"	WC_COMBOBOXEX	*CComboBoxEx*
Date-Time*	"SysDateTimePick32"	DATETIMEPICK_CLASS	*CDateTimeCtrl*

Header	"SysHeader32"	WC_HEADER	*CHeaderCtrl*
Hotkey	"msctls_hotkey32"	HOTKEY_CLASS	*CHotKeyCtrl*
Image list	N/A	N/A	*CImageList*
IP address**	"SysIPAddress32"	WC_IPADDRESS	*CIPAddressCtrl*
List view	"SysListView32"	WC_LISTVIEW	*CListCtrl*
Month calendar*	"SysMonthCal32"	MONTHCAL_CLASS	*CMonthCalCtrl*
Progress	"msctls_progress32"	PROGRESS_CLASS	*CProgressCtrl*
Property sheet	N/A	N/A	*CPropertySheet*
Rebar*	"ReBarWindow32"	REBARCLASSNAME	*CReBarCtrl*
Rich edit	"RichEdit20A" (ANSI) or "RichEdit20W" (Unicode)	RICHEDIT_CLASS	*CRichEditCtrl*
Slider	"msctls_trackbar32"	TRACKBAR_CLASS	*CSliderCtrl*
Spin button	"msctls_updown32"	UPDOWN_CLASS	*CSpinButtonCtrl*
Status bar	"msctls_statusbar32"	STATUSCLASSNAME	*CStatusBarCtrl*
Tab	"SysTabControl32"	WC_TABCONTROL	*CTabCtrl*
Toolbar	"ToolbarWindow32"	TOOLBARCLASSNAME	*CToolBarCtrl*
ToolTip	"tooltips_class32"	TOOLTIPS_CLASS	*CToolTipCtrl*
Tree view	"SysTreeView32"	WC_TREEVIEW	*CTreeCtrl*

1.8.2 Ví dụ tổng hợp:

1.8.2.1 Chương trình 1:

ChildView.h

```
// ChildView.h : interface of the CChildView class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_CHILDVIEW_H__A4559BAA_ABE5_11D2_8E53_006008A82731__INCLUDED_)

#define AFX_CHILDVIEW_H__A4559BAA_ABE5_11D2_8E53_006008A82731__INCLUDED_

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

/////////////////////////////////////////////////////////////////////////////
// CChildView window
class CChildView : public CWnd
{
// Construction
public:
    CChildView();

// Attributes
public:

// Operations
```

```
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CChildView)
    protected:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    //}}AFX_VIRTUAL

// Implementation
public:
    virtual ~CChildView();

    // Generated message map functions
protected:
    int m_nWeight;
    int m_cy;
    int m_cx;
    //{{AFX_MSG(CChildView)
    afx_msg void OnPaint();
    afx_msg void OnOptionsGridSettings();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//     AFX_CHILDVIEW_H__A4559BAA_ABE5_11D2_8E53_006008A82731__INCLUDED_)
```

ChildView.cpp

```
// ChildView.cpp : implementation of the CChildView class
//
#include "stdafx.h"
#include "GridDemo.h"
#include "ChildView.h"
#include "SettingsDialog.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CChildView
CChildView::CChildView()
{
    m_cx = 8;
    m_cy = 8;
    m_nWeight = 4;
}
CChildView::~CChildView()
{
}
```

```
BEGIN_MESSAGE_MAP(CChildView,CWnd )
    //{{AFX_MSG_MAP(CChildView)
    ON_WM_PAINT()
    ON_COMMAND(ID_OPTIONS_GRID_SETTINGS, OnOptionsGridSettings)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CChildView message handlers
BOOL CChildView::PreCreateWindow(CREATESTRUCT& cs)
{
    if (!CWnd::PreCreateWindow(cs))
        return FALSE;

    cs.dwExStyle |= WS_EX_CLIENTEDGE;
    cs.style &= ~WS_BORDER;
    cs.lpszClass = AfxRegisterWndClass(CS_HREDRAW|CS_VREDRAW|CS_DBLCLKS,
        ::LoadCursor(NULL, IDC_ARROW), HBRUSH(COLOR_WINDOW+1), NULL);

    return TRUE;
}
void CChildView::OnPaint()
{
    CRect rect;
    GetClientRect (&rect);

    int nShade = m_nWeight * 32;
    if (nShade != 0)
        nShade- -;

    CPaintDC dc (this);
    CPen pen (PS_SOLID, 1, RGB (nShade, nShade, nShade));
    CPen* pOldPen = dc.SelectObject (&pen);

    int x;
    for (int i=1; i<m_cx; i++) {
        x = (rect.Width () * i) / m_cx;
        dc.MoveTo (x, 0);
        dc.LineTo (x, rect.Height ());
    }

    int y;
    for (i=1; i<m_cy; i++) {
        y = (rect.Height () * i) / m_cy;
        dc.MoveTo (0, y);
        dc.LineTo (rect.Width (), y);
    }

    dc.SelectObject (pOldPen);
}
void CChildView::OnOptionsGridSettings()
{
    CSettingsDialog dlg;

    dlg.m_cx = m_cx;
    dlg.m_cy = m_cy;
```

```
    dlg.m_nWeight = m_nWeight;

    if (dlg.DoModal () == IDOK) {
        m_cx = dlg.m_cx;
        m_cy = dlg.m_cy;
        m_nWeight = dlg.m_nWeight;
        Invalidate ();
    }
}
```

SettingsDialog.h

```
#if !defined(
    AFX_SETTINGSDIALOG_H__A4559BB0_ABE5_11D2_8E53_006008A82731__INCLUDED_)
#define
    AFX_SETTINGSDIALOG_H__A4559BB0_ABE5_11D2_8E53_006008A82731__INCLUDED_

#include "MyToolTipCtrl.h"    // Added by ClassView
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// SettingsDialog.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CSettingsDialog dialog
class CSettingsDialog : public CDialog
{
// Construction
public:
    int m_nWeight;
    CSettingsDialog(CWnd* pParent = NULL);   // standard constructor

// Dialog Data
    //{{AFX_DATA(CSettingsDialog)
    enum { IDD = IDD_SETTINGDLG };
    CSpinButtonCtrl    m_wndSpinVert;
    CSpinButtonCtrl    m_wndSpinHorz;
    CSliderCtrl    m_wndSlider;
    int        m_cx;
    int        m_cy;
    //}}AFX_DATA

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

// Implementation
protected:
    CMyToolTipCtrl m_ctlTT;
    // Generated message map functions
    //{{AFX_MSG(CSettingsDialog)
    virtual BOOL OnInitDialog();
    virtual void OnOK();
    //}}AFX_MSG
```

```
    DECLARE_MESSAGE_MAP()
};

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

#endif
// !defined(
//   AFX_SETTINGSDIALOG_H__A4559BB0_ABE5_11D2_8E53_006008A82731__INCLUDED_)
```

SettingsDialog.cpp

```
// SettingsDialog.cpp : implementation file
//
#include "stdafx.h"
#include "GridDemo.h"
#include "MyToolTipCtrl.h"
#include "SettingsDialog.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CSettingsDialog dialog
CSettingsDialog::CSettingsDialog(CWnd* pParent /*=NULL*/)
    : CDialog(CSettingsDialog::IDD, pParent)
{
    //{{AFX_DATA_INIT(CSettingsDialog)
    m_cx = 0;
    m_cy = 0;
    //}}AFX_DATA_INIT
}
void CSettingsDialog::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CSettingsDialog)
    DDX_Control(pDX, IDC_SPINVERT, m_wndSpinVert);
    DDX_Control(pDX, IDC_SPINHORZ, m_wndSpinHorz);
    DDX_Control(pDX, IDC_SLIDER, m_wndSlider);
    DDX_Text(pDX, IDC_EDITHORZ, m_cx);
    DDX_Text(pDX, IDC_EDITVERT, m_cy);
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CSettingsDialog, CDialog)
    //{{AFX_MSG_MAP(CSettingsDialog)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CSettingsDialog message handlers
BOOL CSettingsDialog::OnInitDialog()
{
    CDialog::OnInitDialog();
    //
    // Initialize the slider control.
    //
```

```
    m_wndSlider.SetRange (0, 8);
    m_wndSlider.SetPos (m_nWeight);
    //
    // Initialize the spin button controls.
    //
    m_wndSpinHorz.SetRange (2, 64);
    m_wndSpinVert.SetRange (2, 64);
    //
    // Create and initialize a tooltip control.
    //
    m_ctlTT.Create (this);
    m_ctlTT.AddWindowTool (GetDlgItem (IDC_SLIDER),
        MAKEINTRESOURCE (IDS_SLIDER));
    m_ctlTT.AddWindowTool (GetDlgItem (IDC_EDITHORZ),
        MAKEINTRESOURCE (IDS_EDITHORZ));
    m_ctlTT.AddWindowTool (GetDlgItem (IDC_EDITVERT),
        MAKEINTRESOURCE (IDS_EDITVERT));
    return TRUE;
}
void CSettingsDialog::OnOK()
{
    //
    // Read the slider control's thumb position
    // before dismissing the dialog.
    //
    m_nWeight = m_wndSlider.GetPos ();
    CDialog::OnOK();
}
```

MyToolTipCtrl.h

```
#if !defined(
    AFX_MYTOOLTIPCTRL_H__A4559BB1_ABE5_11D2_8E53_006008A82731__INCLUDED_)
#define
    AFX_MYTOOLTIPCTRL_H__A4559BB1_ABE5_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// MyToolTipCtrl.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CMyToolTipCtrl window
class CMyToolTipCtrl : public CToolTipCtrl
{
// Construction
public:
    CMyToolTipCtrl();

// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
```

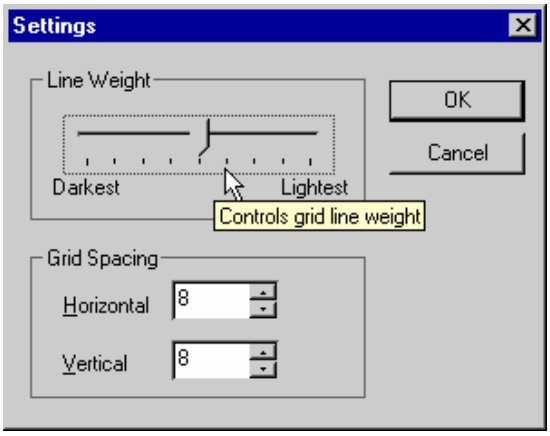
```
    //{{AFX_VIRTUAL(CMyToolTipCtrl)
    //}}AFX_VIRTUAL

// Implementation
public:
    BOOL AddRectTool (CWnd* pWnd, LPCTSTR pszText, LPCRECT pRect,
        UINT nIDTool);
    BOOL AddWindowTool (CWnd* pWnd, LPCTSTR pszText);
    virtual ~CMyToolTipCtrl();

    // Generated message map functions
protected:
    //{{AFX_MSG(CMyToolTipCtrl)
       // NOTE - the ClassWizard will add and remove member functions here.
    //}}AFX_MSG

    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//    AFX_MYTOOLTIPCTRL_H__A4559BB1_ABE5_11D2_8E53_006008A82731__INCLUDED_)
```

MyToolTipCtrl.cpp

```
// MyToolTipCtrl.cpp : implementation file
//
#include "stdafx.h"
#include "GridDemo.h"
#include "MyToolTipCtrl.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CMyToolTipCtrl
CMyToolTipCtrl::CMyToolTipCtrl()
{
}
CMyToolTipCtrl::~CMyToolTipCtrl()
{
}
BEGIN_MESSAGE_MAP(CMyToolTipCtrl, CToolTipCtrl)
    //{{AFX_MSG_MAP(CMyToolTipCtrl)
        // NOTE - the ClassWizard will add and remove mapping macros here.
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CMyToolTipCtrl message handlers
BOOL CMyToolTipCtrl::AddWindowTool(CWnd *pWnd, LPCTSTR pszText)
{
    TOOLINFO ti;
```

```
    ti.cbSize = sizeof (TOOLINFO);
    ti.uFlags = TTF_IDISHWND | TTF_SUBCLASS;
    ti.hwnd = pWnd->GetParent ()->GetSafeHwnd ();
    ti.uId = (UINT) pWnd->GetSafeHwnd ();
    ti.hinst = AfxGetInstanceHandle ();
    ti.lpszText = (LPTSTR) pszText;
    return (BOOL) SendMessage (TTM_ADDTOOL, 0, (LPARAM) &ti);
}
BOOL CMyToolTipCtrl::AddRectTool(CWnd *pWnd, LPCTSTR pszText,
    LPCRECT pRect, UINT nIDTool)
{
    TOOLINFO ti;
    ti.cbSize = sizeof (TOOLINFO);
    ti.uFlags = TTF_SUBCLASS;
    ti.hwnd = pWnd->GetSafeHwnd ();
    ti.uId = nIDTool;
    ti.hinst = AfxGetInstanceHandle ();
    ti.lpszText = (LPTSTR) pszText;
    ::CopyRect (&ti.rect, pRect);

    return (BOOL) SendMessage (TTM_ADDTOOL, 0, (LPARAM) &ti);
}
```

Màn hình kết quả như sau:



1.8.2.2 Chương trình 2:

PathListDlg.h

```
// PathListDlg.h : header file
//
#if !defined(
    AFX_PATHLISTDLG_H__710413E6_AC66_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_PATHLISTDLG_H__710413E6_AC66_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
/////////////////////////////////////////////////////////////////////////////
// CPathListDlg dialog
class CPathListDlg : public CDialog
{
// Construction
public:
    CPathListDlg(CWnd* pParent = NULL);    // standard constructor
// Dialog Data
```

```
    //{{AFX_DATA(CPathListDlg)
    enum { IDD = IDD_PATHLIST_DIALOG };
    CPathComboBox    m_wndCBEx;
    //}}AFX_DATA

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

// Implementation
protected:
    HICON m_hIcon;

    // Generated message map functions
    //{{AFX_MSG(CPathListDlg)
    virtual BOOL OnInitDialog();
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    afx_msg void OnSelEndOK();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//     AFX_PATHLISTDLG_H__710413E6_AC66_11D2_8E53_006008A82731__INCLUDED_)
```

PathListDlg.cpp

```
// PathListDlg.cpp : implementation file
//
#include "stdafx.h"
#include "PathList.h"
#include "PathComboBox.h"
#include "PathListDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CPathListDlg dialog
CPathListDlg::CPathListDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CPathListDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CPathListDlg)
    //}}AFX_DATA_INIT
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
void CPathListDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CPathListDlg)
```

```
    DDX_Control(pDX, IDC_CBEX, m_wndCBEx);
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CPathListDlg, CDialog)
    //{{AFX_MSG_MAP(CPathListDlg)
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_CBN_SELENDOK(IDC_CBEX, OnSelEndOK)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CPathListDlg message handlers
BOOL CPathListDlg::OnInitDialog()
{
    CDialog::OnInitDialog();

    SetIcon(m_hIcon, TRUE);
    SetIcon(m_hIcon, FALSE);
    //
    // Initialize the ComboBoxEx control.
    //
    TCHAR szPath[MAX_PATH];
    ::GetCurrentDirectory (sizeof (szPath) / sizeof (TCHAR), szPath);
    m_wndCBEx.SetPath (szPath);
    return TRUE;
}
void CPathListDlg::OnPaint()
{
    if (IsIconic())
    {
        CPaintDC dc(this); // device context for painting

        SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

        // Center icon in client rectangle
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;

        // Draw the icon
        dc.DrawIcon(x, y, m_hIcon);
    }
    else
    {
        CDialog::OnPaint();
    }
}
HCURSOR CPathListDlg::OnQueryDragIcon()
{
    return (HCURSOR) m_hIcon;
}
void CPathListDlg::OnSelEndOK()
```

```
{
    //
    // Display the path just selected from the ComboBoxEx control.
    //
    MessageBox (m_wndCBEx.GetPath ());
}
```

PathComboBox.h

```
#if !defined(
    AFX_PATHCOMBOBOX_H__710413F1_AC66_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_PATHCOMBOBOX_H__710413F1_AC66_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// PathComboBox.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CPathComboBox window
class CPathComboBox : public CComboBoxEx
{
// Construction
public:
    CPathComboBox();

// Attributes
public:
// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CPathComboBox)
    //}}AFX_VIRTUAL

// Implementation
public:
    CString GetPath();
    BOOL SetPath (LPCTSTR pszPath);
    virtual ~CPathComboBox();

    // Generated message map functions
protected:
    void GetSubstring (int& nStart, CString& string, CString& result);
    int m_nIndexEnd;
    int m_nIndexStart;
    BOOL m_bFirstCall;
    CImageList m_il;
    //{{AFX_MSG(CPathComboBox)
    //}}AFX_MSG

    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
```

```
// immediately before the previous line.
#endif
// !defined(
//     AFX_PATHCOMBOBOX_H__710413F1_AC66_11D2_8E53_006008A82731__INCLUDED_)
```

PathComboBox.cpp

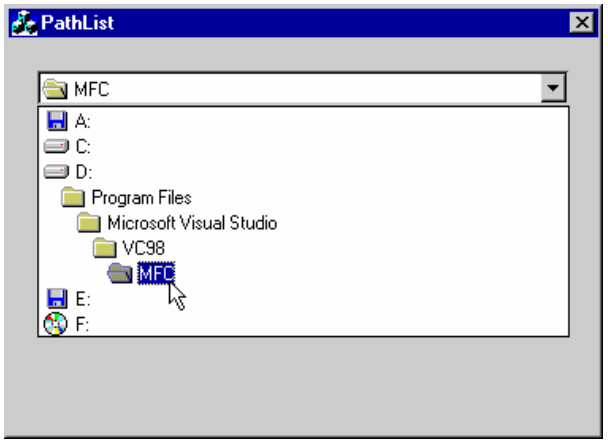
```
// PathComboBox.cpp : implementation file
//
#include "stdafx.h"
#include "PathList.h"
#include "PathComboBox.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CPathComboBox
CPathComboBox::CPathComboBox()
{
    m_bFirstCall = TRUE;
    m_nIndexStart = -1;
    m_nIndexEnd = -1;
}
CPathComboBox::~CPathComboBox()
{
}
BEGIN_MESSAGE_MAP(CPathComboBox, CComboBoxEx)
    //{{AFX_MSG_MAP(CPathComboBox)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CPathComboBox message handlers
BOOL CPathComboBox::SetPath(LPCTSTR pszPath)
{
    if (m_bFirstCall) {
        m_bFirstCall = FALSE;
        //
        // Add an image list containing drive and folder images.
        //
        m_il.Create (IDB_IMAGES, 16, 1, RGB (255, 0, 255));
        SetImageList (&m_il);
        //
        // Add icons representing the drives on the host system.
        //
        int nPos = 0;
        int nCount = 0;
        CString string = _T ("?:\\");

        DWORD dwDriveList = ::GetLogicalDrives ();

        while (dwDriveList) {
            if (dwDriveList & 1) {
                string.SetAt (0, _T (`A') + nPos);
                CString strDrive = string.Left (2);
```

```
                UINT nType = ::GetDriveType (string);

                int nImage = 0;
                switch (nType) {
                case DRIVE_FIXED:
                    nImage = 0;
                    break;
                case DRIVE_REMOVABLE:
                    nImage = 1;
                    break;
                case DRIVE_CDROM:
                    nImage = 2;
                    break;
                case DRIVE_REMOTE:
                    nImage = 3;
                    break;
                }

                COMBOBOXEXITEM cbei;
                cbei.mask = CBEIF_TEXT | CBEIF_IMAGE | CBEIF_SELECTEDIMAGE;
                cbei.iItem = nCount++;
                cbei.pszText = (LPTSTR) (LPCTSTR) strDrive;
                cbei.iImage = nImage;
                cbei.iSelectedImage = nImage;
                InsertItem (&cbei);
            }
            dwDriveList >>= 1;
            nPos++;
        }
    }

    //
    // Find the item that corresponds to the drive specifier in pszPath.
    //
    CString strPath = pszPath;
    CString strDrive = strPath.Left (2);

    int nDriveIndex = FindStringExact (-1, strDrive);
    if (nDriveIndex == CB_ERR)
        return FALSE;
    //
    // Delete previously added folder items (if any).
    //
    if (m_nIndexStart != -1 && m_nIndexEnd != -1) {
        ASSERT (m_nIndexEnd >= m_nIndexStart);
        int nCount = m_nIndexEnd - m_nIndexStart + 1;
        for (int i=0; i<nCount; i++)
            DeleteItem (m_nIndexStart);
        if (m_nIndexStart < nDriveIndex)
            nDriveIndex -= nCount;
        m_nIndexStart = -1;
        m_nIndexEnd = -1;
    }
    //
    // Add items representing the directories in pszPath.
```

```
    //
    int nCount = 0;
    int nStringIndex = strPath.Find (_T (`\\'), 0);

    if (nStringIndex++ != -1) {
        CString strItem;
        GetSubstring (nStringIndex, strPath, strItem);

        while (!strItem.IsEmpty ()) {
            COMBOBOXEXITEM cbei;
            cbei.mask = CBEIF_TEXT | CBEIF_IMAGE | CBEIF_SELECTEDIMAGE |
                CBEIF_INDENT;
            cbei.iItem = nDriveIndex + ++nCount;
            cbei.pszText = (LPTSTR) (LPCTSTR) strItem;
            cbei.iImage = 4;
            cbei.iSelectedImage = 5;
            cbei.iIndent = nCount;
            InsertItem (&cbei);

            GetSubstring (nStringIndex, strPath, strItem);
        }
    }
    //
    // Record the indexes of the items that were added, too.
    //
    if (nCount) {
        m_nIndexStart = nDriveIndex + 1;
        m_nIndexEnd = nDriveIndex + nCount;
    }
    //
    // Finish up by selecting the final item.
    //
    int nResult = SetCurSel (nDriveIndex + nCount);
    return TRUE;
}
void CPathComboBox::GetSubstring(int& nStart, CString &string,
    CString &result)
{
    result = _T ("");
    int nLen = string.GetLength ();
    if (nStart >= nLen)
        return;

    int nEnd = string.Find (_T (`\\'), nStart);
    if (nEnd == -1) {
        result = string.Right (nLen - nStart);
        nStart = nLen;
    }
    else {
        result = string.Mid (nStart, nEnd - nStart);
        nStart = nEnd + 1;
    }
}
CString CPathComboBox::GetPath()
{
```

```
    //
    // Get the index of the selected item.
    //
    CString strResult;
    int nEnd = GetCurSel ();
    int nStart = nEnd + 1;
    //
    // Find the index of the "root" item.
    //
    COMBOBOXEXITEM cbei;
    do {
        cbei.mask = CBEIF_INDENT;
        cbei.iItem = –nStart;
        GetItem (&cbei);
    } while (cbei.iIndent != 0);
    //
    // Build a path name by combining all the items from the root item to
    // the selected item.
    //
    for (int i=nStart; i<=nEnd; i++) {
        TCHAR szItem[MAX_PATH];
        COMBOBOXEXITEM cbei;
        cbei.mask = CBEIF_TEXT;
        cbei.iItem = i;
        cbei.pszText = szItem;
        cbei.cchTextMax = sizeof (szItem) / sizeof (TCHAR);
        GetItem (&cbei);
        strResult += szItem;
        strResult += _T ("\\");
    }
    //
    // Strip the trailing backslash.
    //
    int nLen = strResult.GetLength ();
    strResult = strResult.Left (nLen - 1);
    return strResult;
}
```

Màn hình kết quả như sau:



CHƯƠNG 2. CẤU TRÚC DOCUMENT-VIEW CỦA MFC WINDOWS APP

2.1 GIỚI THIỆU DOCUMENT-VIEW VÀ SDI (SINGLE DOCUMENT INTERFACE)

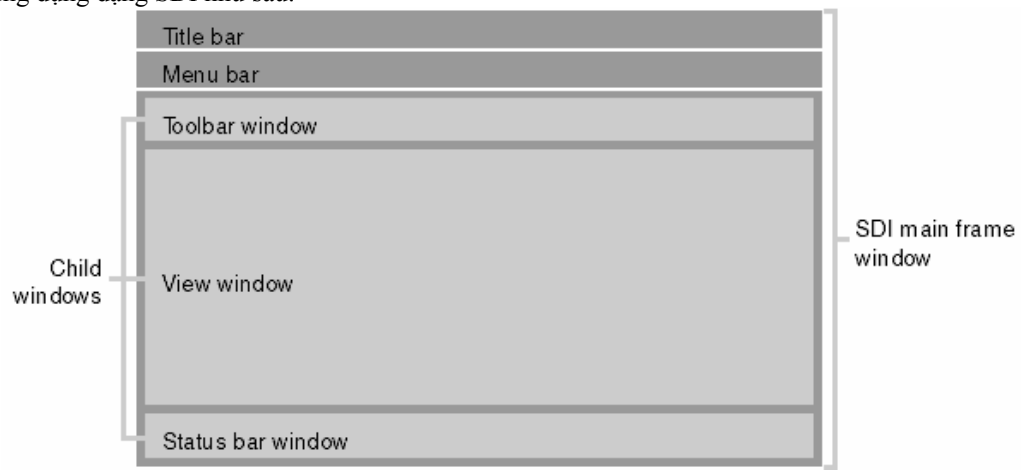
2.1.1 Vấn đề quan tâm

- Hiểu về các class thành phần trong 1 project.
- Biết và sử dụng đúng chức năng của các class về View, Document.

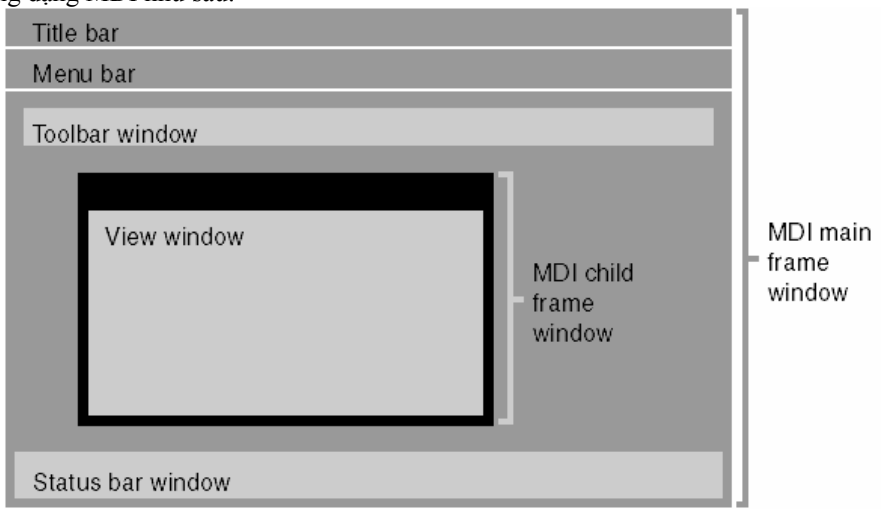
2.1.2 Giới thiệu

Kết cấu của ứng dụng dạng SDI và MDI được hợp thành bởi thành phần CView và CDocument, chính là cấu trúc Document/View.

Khung ứng dụng dạng SDI như sau:



Khung ứng dụng dạng MDI như sau:



Trong quá trình tạo ứng dụng dạng SDI và MDI, các class được tạo ra hầu hết được kế thừa (dẫn xuất) từ các class như:

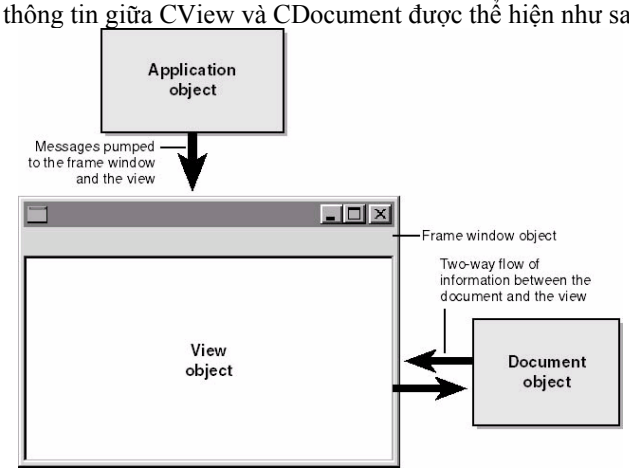
- Với dạng SDI, các class này là CWinApp, CFrameWnd, CDocument, CView
- Với dạng MDI, các class này là: CWinApp, CMDIFrameWnd, CMDIChildWnd, CDocument, CView

Vai trò của mỗi class được mô tả như sau:

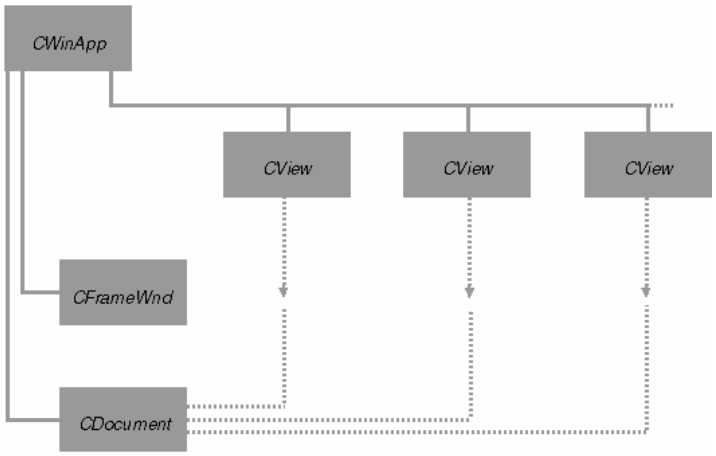
Lớp	Vai trò
CWinApp	Tạo tất cả thành phần khác trong ứng dụng. Đây là lớp nhận thông báo sự kiện và truyền tới lớp CView và CDocument
CFrameWnd	- Tạo và quản lý khung cửa sổ, chứa menu, toolbar, scrollbar và các đối tượng nhìn thấy được gắn vào cửa sổ.

	- Xác định số tài liệu nhìn thấy được trong một thời điểm bất kỳ
CMDIFrameWnd	- (thể hiện bởi CMainFrame trong project) là khung chính của ứng dụng - Cung cấp không gian bao bọc trên màn hình nền diễn ra toàn bộ tương tác ứng dụng. - Cửa sổ này là khung được gắn menu và thanh công cụ
CMDIChildFrameWnd	- (thể hiện bởi CChildFrame trong project) là khung chứa lớp CView - Là lớp truyền thông điệp (message) và sự kiện (event) tới các lớp hiển thị để xử lý hay hiển thị.
CDocument	- Chứa tài liệu của ứng dụng. Đây là nơi sẽ lưu trữ cấu trúc dữ liệu cần thiết để chứa và thao tác dữ liệu tạo nên tài liệu. - Lớp này nhận dữ liệu từ lớp CView và truyền thông tin sang lớp CView. - Lưu giữ và lấy dữ liệu về tài liệu từ các file
CView	- Quản lý phần hiển thị phần trình bày nhìn thấy của tài liệu cho người dùng xem. - Truyền thông tin vào lớp CDocument và nhận thông tin từ lớp CDocument.

Việc tương tác và trao đổi thông tin giữa CView và CDocument được thể hiện như sau:



hay:

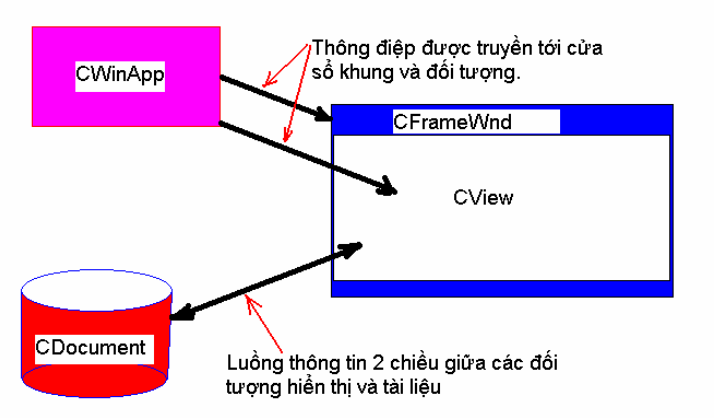


Đặc biệt, lớp CView cung cấp một số dạng giao diện đồ hoạ thông qua các lớp con như:

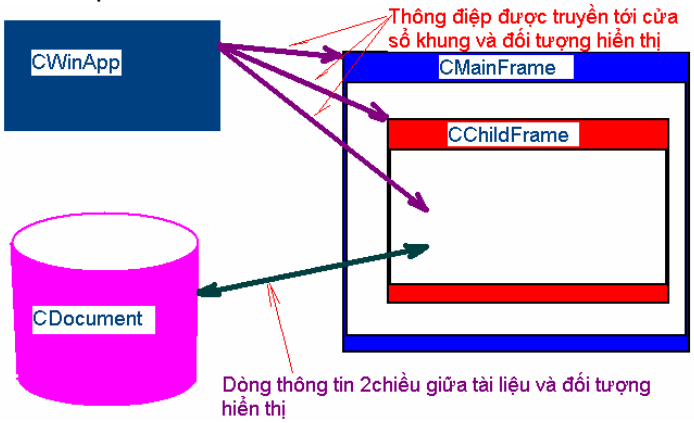
- ***CEditView***: cung cấp giao diện dạng editor soạn thảo
- ***CFormView***: cung cấp giao diện dạng form để gắn thêm các điều khiển
- ***CHtmlView***: cung cấp giao diện dạng trình duyệt web
- ***CListView***: cung cấp giao diện dạng danh sách
- ***CRichEditView***: cung cấp giao diện dạng editor soạn thảo có nhiều hỗ trợ
- ***CScrollView***: cung cấp giao diện có thanh cuộn
- ***CTreeView***: cung cấp giao diện dạng cây

2.1.3 Cấu trúc Document/View

Với dạng SDI, sự liên kết thể hiện như sau:



Với dạng MDI, sự liên kết thể hiện như sau:



2.1.4 Sự tương tác giữa phần Document và phần View

Với cả 2 dạng SDI và MDI, sự liên kết thể hiện thông qua 1 biến pointer kiểu CDocument (*biến này thường có tên là pDoc*). Thông qua biến này, chúng ta có thể điều khiển thao tác chuyển thông tin (*từ lớp CView sang CDocument*) hay lấy thông tin về (*từ lớp CDocument sang lớp CView*).

Thông thường, với cả ứng dụng dạng SDI hay MDI thì trong lớp CView luôn có hàm **GetDocument** nhằm hỗ trợ việc lấy biến con trỏ này.

Ví dụ 1:

```
CVd5bDoc* CVd5bView::GetDocument() // non-debug version is inline
{
	ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CVd5bDoc)));
	return(CVd5bDoc*)m_pDocument;
}
```

và việc sử dụng có thể gọi hàm như sau:

```
void CVd5bView::OnButtonThem()
{
	CVd5bDoc* pDoc = GetDocument();

	// TODO: Add your control notification handler code here
	UpdateData(TRUE);
	pDoc->Them(m_nSTT, m_szMSSV, m_szHoTen, m_bPhai,
m_cboChuyenNganh.GetCurSel(), m_chkTheThao.GetCheck(), m_chkVanNghe.GetCheck());
	OnButtonReset();
	HienThiSoRecord();
}
```

2.2 CÁC KIỂU VIEW

2.2.1 Vấn đề quan tâm

- Hiểu về vai trò của các lớp con của CView.

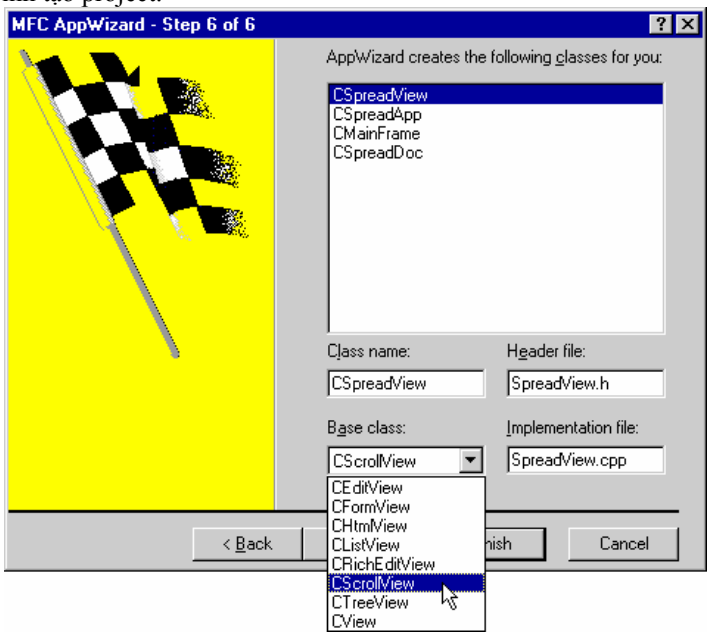
- Biết và sử dụng đúng chức năng của các class về View.

2.2.2 Giới thiệu

MFC cung cấp lớp CView và các lớp con để hỗ trợ vấn đề xây dựng giao diện như:

Tên lớp	Mô tả
CView	Lớp cơ sở
CCtrlView	Dùng làm lớp cơ sở cho các dạng view khác
CEditView	Cung cấp giao diện dạng editor
CRichEditView	Cung cấp giao diện dạng editor có nhiều hỗ trợ
CListView	Cung cấp giao diện dạng danh sách
CTreeView	Cung cấp giao diện dạng cây
CHtmlView	Cung cấp giao diện dạng trình duyệt web
CScrollView	Cung cấp giao diện có thanh cuộn
CFormView	Cung cấp giao diện dạng form để gắn các điều khiển
CRecordView	Cung cấp giao diện có tương tác điều khiển dữ liệu
CDaoRecordView	Cung cấp giao diện có tương tác điều khiển dữ liệu (dạng DAO)
COleDBRecordView	Cung cấp giao diện có tương tác điều khiển dữ liệu (dạng OLE)

Để có thể sử dụng các lớp này trong việc thể hiện giao diện của ứng dụng, chúng ta sẽ chọn chúng là lớp cơ sở trong bước 6 của quá trình tạo project.



2.2.3 Lớp CScrollView và ứng dụng

CScrollView được sử dụng trong trường hợp cần có giao diện hỗ trợ scrollbar.

Để xác định kích thước của vùng kích thước "luận lý" của cửa sổ, sử dụng hàm ảo (*virtual function*) **OnInitialUpdate** và hàm **SetScrollSizes**.

Ví dụ 1:

```
void CMyView::OnInitialUpdate()
{
  CScrollView::OnInitialUpdate();
  SetScrollSizes(MM_TEXT, CSize(1280, 1024));
}
```

hay

```
void CMyView::OnInitialUpdate()
{
  CScrollView::OnInitialUpdate();
  SetScrollSizes(MM_LOENGLISH, CSize(850, 1100));
}
```

Ngoài ra, có thể nhận biết được vị trí của scrollbar hiện hành hay ấn định vị trí của scrollbar bởi hàm **GetScrollPosition()** và **ScrollToPosition()**

Ví dụ 2:

```
CPoint pos = GetScrollPosition(); // lấy vị trí scrollbar hiện tại

ScrollToPosition(CPoint(100, 100));     // xác định vị trí scrollbar

CSize size = GetTotalSize();      // lấy kích thước cửa sổ hiện hành
int nWidth = size.cx;
int nHeight = size.cy;

SetScaleToFitSize(GetTotalSize());      // thay đổi tỉ lệ & kích thước
SetScrollSizes();          // khôi phục kích thước cửa sổ
```

Để chuyển đổi từ dạng CView sang dạng CScrollView, cần thực thi theo 2 bước sau:

- Tìm trong file .h và .cpp và thay thế CView bởi CScrollView
- Trong hàm OnInitialUpdate gọi SetScrollSize để xác định kích thước cửa sổ.

Ví dụ tổng hợp:

ScrollDemoView.h

```
// ScrollDemoView.h : interface of the CScrollDemoView class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(AFX_SCROLLDEMOVIEW_H__DCCF4E0D_9735_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_SCROLLDEMOVIEW_H__DCCF4E0D_9735_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CScrollDemoView : public CScrollView
{
protected: // create from serialization only
    CScrollDemoView();
    DECLARE_DYNCREATE(CScrollDemoView)
// Attributes
public:
    CScrollDemoDoc* GetDocument();
// Operations
public:
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CScrollDemoView)
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 ~CScrollDemoView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Generated message map functions
protected:
```

```
    BOOL m_bSmooth;
    void GetCellRect (int row, int col, LPRECT pRect);
    void DrawAddress (CDC* pDC, int row, int col);
    void DrawPointer (CDC* pDC, int row, int col, BOOL bHighlight);
    CFont m_font;
    int m_nCurrentCol;
    int m_nCurrentRow;
    int m_nRibbonWidth;
    int m_nCellHeight;
    int m_nCellWidth;
    //{{AFX_MSG(CScrollDemoView)
    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in ScrollDemoView.cpp
inline CScrollDemoDoc* CScrollDemoView::GetDocument()
   { return (CScrollDemoDoc*)m_pDocument; }
#endif
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
// !defined(
//     AFX_SCROLLDEMOVIEW_H__DCCF4E0D_9735_11D2_8E53_006008A82731__INCLUDED_)
```

ScrollDemoView.cpp

```
// ScrollDemoView.cpp : implementation of the CScrollDemoView class
//
#include "stdafx.h"
#include "ScrollDemo.h"
#include "ScrollDemoDoc.h"
#include "ScrollDemoView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CScrollDemoView
IMPLEMENT_DYNCREATE(CScrollDemoView, CScrollView)

BEGIN_MESSAGE_MAP(CScrollDemoView, CScrollView)
    //{{AFX_MSG_MAP(CScrollDemoView)
    ON_WM_LBUTTONDOWN()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CScrollDemoView construction/destruction
CScrollDemoView::CScrollDemoView()
{
    m_font.CreatePointFont (80, _T ("MS Sans Serif"));
}
CScrollDemoView::~CScrollDemoView()
```

```
{
}
BOOL CScrollDemoView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CScrollView::PreCreateWindow(cs);
}
/////////////////////////////////////////////////////////////////////////////
// CScrollDemoView drawing
void CScrollDemoView::OnDraw(CDC* pDC)
{
    CScrollDemoDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    //
    // Draw the grid lines.
    //
    CSize size = GetTotalSize ();

    CPen pen (PS_SOLID, 0, RGB (192, 192, 192));
    CPen* pOldPen = pDC->SelectObject (&pen);
    for (int i=0; i<99; i++) {
        int y = (i * m_nCellHeight) + m_nCellHeight;
        pDC->MoveTo (0, y);
        pDC->LineTo (size.cx, y);
    }
    for (int j=0; j<26; j++) {
        int x = (j * m_nCellWidth) + m_nRibbonWidth;
        pDC->MoveTo (x, 0);
        pDC->LineTo (x, size.cy);
    }
    pDC->SelectObject (pOldPen);
    //
    // Draw the bodies of the rows and column headers.
    //
    CBrush brush;
    brush.CreateStockObject (LTGRAY_BRUSH);

    CRect rcTop (0, 0, size.cx, m_nCellHeight);
    pDC->FillRect (rcTop, &brush);
    CRect rcLeft (0, 0, m_nRibbonWidth, size.cy);
    pDC->FillRect (rcLeft, &brush);

    pDC->MoveTo (0, m_nCellHeight);
    pDC->LineTo (size.cx, m_nCellHeight);
    pDC->MoveTo (m_nRibbonWidth, 0);
    pDC->LineTo (m_nRibbonWidth, size.cy);

    pDC->SetBkMode (TRANSPARENT);
    //
    // Add numbers and button outlines to the row headers.
    //
    for (i=0; i<99; i++) {
        int y = (i * m_nCellHeight) + m_nCellHeight;
        pDC->MoveTo (0, y);
        pDC->LineTo (m_nRibbonWidth, y);
```

```
        CString string;
        string.Format (_T ("%d"), i + 1);

        CRect rect (0, y, m_nRibbonWidth, y + m_nCellHeight);
        pDC->DrawText (string, &rect, DT_SINGLELINE ¦
            DT_CENTER ¦ DT_VCENTER);

        rect.top++;
        pDC->Draw3dRect (rect, RGB (255, 255, 255),
            RGB (128, 128, 128));
    }
    //
    // Add letters and button outlines to the column headers.
    //
    for (j=0; j<26; j++) {
        int x = (j * m_nCellWidth) + m_nRibbonWidth;
        pDC->MoveTo (x, 0);
        pDC->LineTo (x, m_nCellHeight);

        CString string;
        string.Format (_T ("%c"), j + `A');

        CRect rect (x, 0, x + m_nCellWidth, m_nCellHeight);
        pDC->DrawText (string, &rect, DT_SINGLELINE ¦
            DT_CENTER ¦ DT_VCENTER);

        rect.left++;
        pDC->Draw3dRect (rect, RGB (255, 255, 255),
            RGB (128, 128, 128));
    }
    //
    // Draw address labels into the individual cells.
    //
    CRect rect;
    pDC->GetClipBox (&rect);
    int nStartRow = max (0, (rect.top - m_nCellHeight) / m_nCellHeight);
    int nEndRow = min (98, (rect.bottom - 1) / m_nCellHeight);
    int nStartCol = max (0, (rect.left - m_nRibbonWidth) / m_nCellWidth);
    int nEndCol = min (25, ((rect.right + m_nCellWidth - 1) -
        m_nRibbonWidth) / m_nCellWidth);

    for (i=nStartRow; i<=nEndRow; i++)
        for (j=nStartCol; j<=nEndCol; j++)
            DrawAddress (pDC, i, j);
    //
    // Draw the cell pointer.
    //
    DrawPointer (pDC, m_nCurrentRow, m_nCurrentCol, TRUE);
}
void CScrollDemoView::OnInitialUpdate()
{
    CScrollView::OnInitialUpdate();

    m_nCurrentRow = 0;
    m_nCurrentCol = 0;
```

```
    m_bSmooth = FALSE;

    CClientDC dc (this);
    m_nCellWidth = dc.GetDeviceCaps (LOGPIXELSX);
    m_nCellHeight = dc.GetDeviceCaps (LOGPIXELSY) / 4;
    m_nRibbonWidth = m_nCellWidth / 2;

    int nWidth = (26 * m_nCellWidth) + m_nRibbonWidth;
    int nHeight = m_nCellHeight * 100;
    SetScrollSizes (MM_TEXT, CSize (nWidth, nHeight));
}
/////////////////////////////////////////////////////////////////////////////
// CScrollDemoView diagnostics
#ifdef _DEBUG
void CScrollDemoView::AssertValid() const
{
    CScrollView::AssertValid();
}
void CScrollDemoView::Dump(CDumpContext& dc) const
{
    CScrollView::Dump(dc);
}
CScrollDemoDoc* CScrollDemoView::GetDocument() // non-debug version is
                                               inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CScrollDemoDoc)));
    return (CScrollDemoDoc*)m_pDocument;
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CScrollDemoView message handlers
void CScrollDemoView::OnLButtonDown(UINT nFlags, CPoint point)
{
    CScrollView::OnLButtonDown(nFlags, point);
    //
    // Convert the click point to logical coordinates.
    //
    CPoint pos = point;
    CClientDC dc (this);
    OnPrepareDC (&dc);
    dc.DPtoLP (&pos);
    //
    // If a cell was clicked, move the cell pointer.
    //
    CSize size = GetTotalSize ();
    if (pos.x > m_nRibbonWidth && pos.x < size.cx &&
        pos.y > m_nCellHeight && pos.y < size.cy) {

        int row = (pos.y - m_nCellHeight) / m_nCellHeight;
        int col = (pos.x - m_nRibbonWidth) / m_nCellWidth;
        ASSERT (row >= 0 && row <= 98 && col >= 0 && col <= 25);

        DrawPointer (&dc, m_nCurrentRow, m_nCurrentCol, FALSE);
        m_nCurrentRow = row;
        m_nCurrentCol = col;
```

```
        DrawPointer (&dc, m_nCurrentRow, m_nCurrentCol, TRUE);
    }
}
void CScrollDemoView::DrawPointer(CDC *pDC, int row, int col,
    BOOL bHighlight)
{
    CRect rect;
    GetCellRect (row, col, &rect);
    CBrush brush (bHighlight ? RGB (0, 255, 255) :
        ::GetSysColor (COLOR_WINDOW));
    pDC->FillRect (rect, &brush);
    DrawAddress (pDC, row, col);
}
void CScrollDemoView::DrawAddress(CDC *pDC, int row, int col)
{
    CRect rect;
    GetCellRect (row, col, &rect);

    CString string;
    string.Format (_T ("%c%d"), col + _T (`A'), row + 1);

    pDC->SetBkMode (TRANSPARENT);
    CFont* pOldFont = pDC->SelectObject (&m_font);
    pDC->DrawText (string, rect, DT_SINGLELINE ¦ DT_CENTER ¦ DT_VCENTER);
    pDC->SelectObject (pOldFont);
}
void CScrollDemoView::GetCellRect(int row, int col, LPRECT pRect)
{
    pRect->left = m_nRibbonWidth + (col * m_nCellWidth) + 1;
    pRect->top = m_nCellHeight + (row * m_nCellHeight) + 1;
    pRect->right = pRect->left + m_nCellWidth - 1;
    pRect->bottom = pRect->top + m_nCellHeight - 1;
}
```

2.2.4 Lớp CHtmlView và ứng dụng

CHtmlView được sử dụng trong trường hợp cần có giao diện hỗ trợ hiển thị trang web.

CHtmlView được xây dựng từ một số hàm tương tác với WebBrower control

Hàm	Mô tả
GetBusy	Trả về trạng thái chương trình có download nào đang hoạt động không
GetLocationName	Trả về tựa đề nếu là trang web; trả về tên file/thư mục nếu là đường dẫn vật lý
GetLocationURL	Trả về địa chỉ URL dạng http://.... hay file://...
GoBack	Quay về trang trước
GoForward	Đi tới trang kế
Navigate	Hiển thị nội dung của địa chỉ URL xác định
Refresh	Tải và hiển thị lại nội dung của địa chỉ URL hiện hành
Stop	Ngừng việc tải dữ liệu

Ví dụ 1:

```
// In CMyView's message map
ON_COMMAND(ID_BACK, OnBack)
ON_COMMAND(ID_FORWARD, OnForward)
ON_COMMAND(ID_REFRESH, OnRefresh)
ON_COMMAND(ID_STOP, OnStop)

void CMyView::OnBack()
{
```

```
  GoBack();
}
void CMyView::OnForward()
{
  GoForward();
}
void CMyView::OnRefresh()
{
  Refresh();
}
void CMyView::OnStop()
{
  Stop();
}
```

và:

```
Navigate(_T("http://www.microsoft.com"));
Navigate(_T("file://c:/my documents/budget.xls"));
```

Hàm	Mô tả
OnNavigateComplete2	Được gọi sau khi truy xuất một file
OnBeforeNavigate2	Được gọi sau khi truy xuất một file
OnProgressChange	Được gọi khi cung cấp trạng thái của quá trình tải dữ liệu
OnDownloadBegin	Được gọi khi quá trình tải dữ liệu bắt đầu
OnDownloadComplete	Được gọi khi quá trình tải dữ liệu hoàn tất
OnTitleChange	Được gọi khi tựa đề thay đổi
OnDocumentComplete	Được gọi khi tài liệu được tải hoàn tất

Ví dụ tổng hợp:

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__D39825ED_99C0_11D2_8E53_006008A82731__INCLUDED_)
```

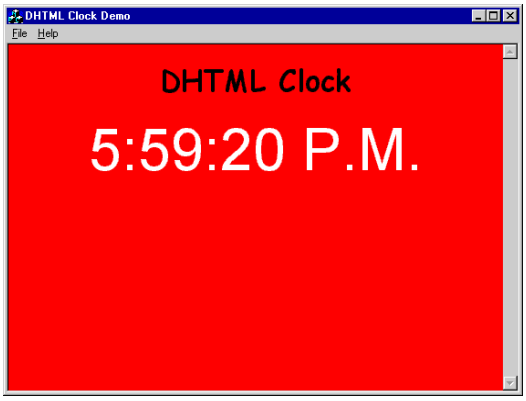
HtmlClockView.cpp

```
// 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);
}
```

Màn hình kết quả như sau:



2.2.5 Lớp CTreeView và ứng dụng

CTreeView được sử dụng trong trường hợp cần có giao diện hỗ trợ hiển thị cây.

CTreeView được xây dựng liên quan đến TreeView control

Để điều khiển được cây trong giao diện chương trình, chúng ta sử dụng hàm **GetTreeCtrl** để lấy quyền điều khiển/truy xuất cây.

Ví dụ 1:

```
UINT nCount = GetTreeCtrl().GetCount(); // lấy số nút trên cây
```

Để tạo cây, có thể dùng các kiểu sau:

Kiểu	Mô tả
TVS_HASLINES	Adds lines connecting subitems to their parents.
TVS_LINESATROOT	Adds lines connecting items at the top level, or root, of the hierarchy. This style is valid only if TVS_HASLINES is also specified.
TVS_HASBUTTONS	Adds buttons containing plus or minus signs to items that have subitems. Clicking a button expands or collapses the associated subtree.
TVS_EDITLABELS	Enables in-place label editing notifications.
TVS_DISABLEDRAGDROP	Disables drag-and-drop notifications.
TVS_SHOWSELALWAYS	Specifies that the item that's currently selected should always be highlighted. By default, the highlight is removed when the control loses the input focus.

Và sử dụng các khai báo này trong hàm **PreCreateWindow**

Ví dụ 1:

```
BOOL CVd8cView::PreCreateWindow(CREATESTRUCT& cs) {
      // TODO: Modify the Window class or styles here by modifying       // the
CREATESTRUCT cs
cs.style |= TVS_HASLINES | TVS_LINESATROOT | TVS_HASBUTTONS |    TVS_SHOWSELALWAYS;
      return CTreeView::PreCreateWindow(cs);
}
```

Hàm **InsertItem** được dùng để thêm phần tử

Ví dụ 2:

```
// 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);
```

Ví dụ tổng hợp:

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__INCLUDED_)
```

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);
```

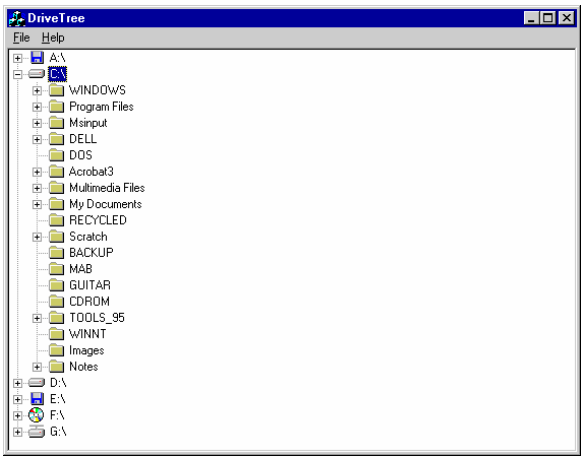
```
}
CDriveTreeDoc* CDriveView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CDriveTreeDoc)));
    return (CDriveTreeDoc*)m_pDocument;
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////
// CDriveView message handlers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

Màn hình kết quả như sau:



2.2.6 Lớp CListView và ứng dụng

Tương tự các dạng view nêu trên, CListView giúp tạo giao diện dạng list

Việc điều khiển, tương tác với danh sách (*list*) được thực hiện thông qua việc gọi hàm **GetListCtrl** nhằm lấy quyền điều khiển

Các dạng list được chọn lựa theo các tùy chọn sau:

Kiểu	Mô tả
LVS_ICON	Selects large icon mode.
LVS_SMALLICON	Selects small icon mode.
LVS_LIST	Selects list mode.
LVS_REPORT	Selects report mode.
LVS_NOCOLUMNHEADER	Removes the header control that's normally displayed in report mode.
LVS_NOSORTHEADER	Disables the LVN_COLUMNCLICK notifications that are sent by default when a column header is clicked in report mode.
LVS_ALIGNLEFT	Aligns items along the left border in large and small icon mode.

LVS_ALIGNTOP	Aligns items along the top border in large and small icon mode.
LVS_AUTOARRANGE	Automatically arranges items in rows and columns in large and small icon mode.
LVS_EDITLABELS	Enables in-place label editing notifications.
LVS_NOLABELWRAP	Restricts labels to single lines in large icon mode.
LVS_NOSCROLL	Disables scrolling. Scrolling is enabled by default.
LVS_OWNERDRAWFIXED	Specifies that the control's owner will draw the items in response to WM_DRAWITEM messages.
LVS_SHAREIMAGELISTS	Prevents a list view from automatically deleting the image lists associated with it when the view itself is deleted.
LVS_SINGLESEL	Disables multiple-selection support.
LVS_SHOWSELALWAYS	Specifies that the selected items should always be highlighted. By default, the highlight is removed when the view loses the input focus.
LVS_SORTASCENDING	Specifies that items should be sorted in ascending order(for example, A through Z).
LVS_SORTDESCENDING	Specifies that items should be sorted in descending order(for example, Z through A).

Việc khởi tạo cho 1 list gồm 5 bước sau đây:

- Tạo 1 cặp hình bitmap bao gồm 1 hình cho chế độ xem lớn, và 1 hình cho các chế độ xem còn lại.
- Sử dụng SetImageList để liên kết danh sách hình với list view control
- Dùng **InsertColumn()** để thêm các cột cho list (tương ứng với chế độ xem chi tiết)
- Thêm các phần tử với lệnh **InsertItem()**
- Gán tựa cho nút con với **SetItemText()**

Ví dụ tổng hợp:

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__18BD7B7C_95C6_11D2_8E53_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_8E53_006008A82731__INCLUDED_)
#define AFX_FILEVIEW_H__18BD7B80_95C6_11D2_8E53_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__18BD7B80_95C6_11D2_8E53_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).
        //
```

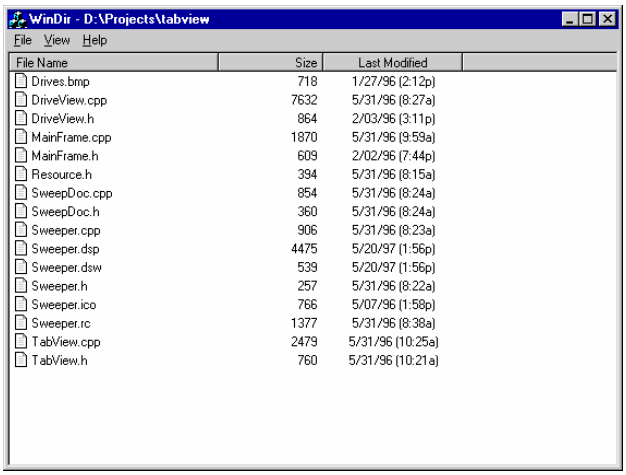
```
        GetListCtrl ().DeleteAllItems ();
        //
        // Show the path name in the frame window's title bar.
        //
        TCHAR szFullPath[MAX_PATH];
        ::GetFullPathName (pszPath, sizeof (szFullPath) / sizeof (TCHAR),
            szFullPath, NULL);
        m_strPath = szFullPath;

        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);
}
```

Màn hình kết quả như sau:



2.3 MULTI-DOCUMENT, MULTI-VIEW VÀ MDI (MULTIPLE DOCUMENT INTERFACE)

2.3.1 Vấn đề quan tâm

- Hiểu về cấu trúc của dạng MDI
- Hiểu về vai trò của các lớp trong cấu trúc MDI.

2.3.2 Các hoạt động trong dạng MDI

Trong ứng dụng MDI, 1 document có thể được thể hiện trong nhiều view (**windows**), để có thể đồng bộ các view này cần sử dụng phương thức UpdateAllViews (trong lớp CDocument)

Ví dụ 1:

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

Hay

```
UpdateAllViews(NULL);
```

Khi gọi hàm này, chương trình sẽ gọi hàm **OnUpdate** của tất cả các view hiện hành và thực hiện thao tác cập nhật tương ứng.

Ví dụ 2:

```
// Thực hiện trong lớp CDocument
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);
}
```

Ví dụ tổng hợp:

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_CHILDFRM_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__INCLUDED_)
```

```
#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_)
```

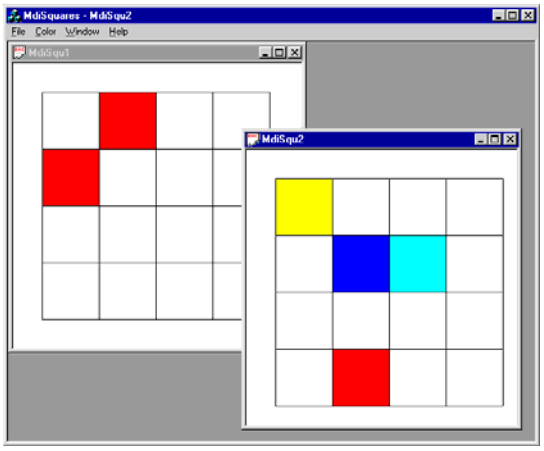
SquaresView.cpp

```
// 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);
    }
}
```

Màn hình kết quả như sau:



2.3.3 Dạng splitter

Để phân chia cửa sổ thành các vùng riêng biệt, cần thực hiện việc phân chia cửa sổ, thao tác như sau:

2.3.3.1 Tạo vùng phân chia động:

- Thêm biến thành viên (thuộc class CSplitterWnd) vào class CMainFrame cửa sổ khung chương trình.
- Khai báo lại hàm **OnCreateClient** và gọi hàm CSplitterWnd::**Create** để tạo khung cửa sổ phân chia được trong cửa sổ khung.

Ví dụ:

```
class CMainFrame : public CFrameWnd
{
…
protected:
  CSplitterWnd m_wndSplitter;
…
};
```

và

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

2.3.3.2 Tạo vùng phân chia tĩnh:

- Thêm biến thành viên (thuộc class CSplitterWnd) vào class CMainFrame cửa sổ khung chương trình.
- Khai báo lại hàm **OnCreateClient** và gọi hàm CSplitterWnd::**CreateStatic** để tạo khung cửa sổ phân chia được trong cửa sổ khung
- Dùng CSplitterWnd::**CreateView** để tạo các view cho mỗi thành phần.

Ví dụ:

```
class CMainFrame : public CFrameWnd
{
…
protected:
  CSplitterWnd m_wndSplitter;
…
};
```

và

```
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;
}
```

2.3.4 Ví dụ tổng hợp

2.3.4.1 Chương trình 1:

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__INCLUDED_)
#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, 0x7C, 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

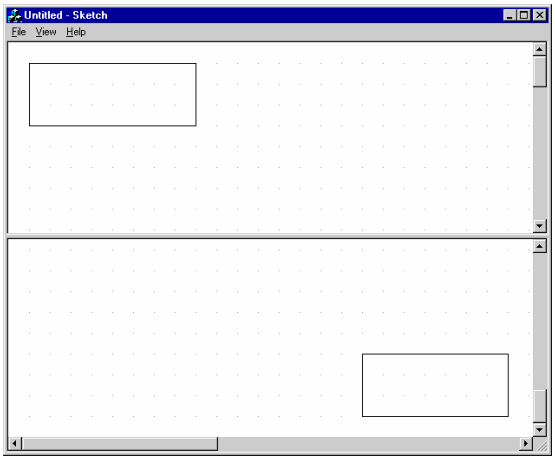
```
// SketchView.cpp : implementation of the CSketchView class
//
#include "stdafx.h"
#include "Line.h"
#include "Sketch.h"
#include "SketchDoc.h"
#include "SketchView.h"

#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);
}
```

Màn hình kết quả như sau:



2.3.4.2 Chương trình 2:

Wanderer.h

```
// Wanderer.h : main header file for the WANDERER application
//
#if !defined(AFX_WANDERER_H__AE0A6FFA_9B0F_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_WANDERER_H__AE0A6FFA_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__AE0A6FFE_9B0F_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__AE0A6FFE_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__AE0A6FFE_9B0F_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
{
    CFrameWnd::AssertValid();
}
void CMainFrame::Dump(CDumpContext& dc) const
{
    CFrameWnd::Dump(dc);
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers
BOOL CMainFrame::OnCreateClient(LPCREATESTRUCT lpcs,
    CCreateContext* pContext)
{
    //
    // Note: Create the CFileView first so the CDriveView's OnInitialUpdate
    // function can call OnUpdate on the CFileView.
    //
    if (!m_wndSplitter.CreateStatic (this, 1, 2) ||
        !m_wndSplitter.CreateView (0, 1, RUNTIME_CLASS
            (CFileView), CSize (0, 0), pContext) ||
        !m_wndSplitter.CreateView (0, 0, RUNTIME_CLASS (CDriveView),
            CSize (192, 0), pContext))
        return FALSE;

    return TRUE;
}
BOOL CMainFrame::OnCmdMsg(UINT nID, int nCode, void* pExtra,
```

```
    AFX_CMDHANDLERINFO* pHandlerInfo)
{
    //
    // Route to standard command targets first.
    //
    if (CFrameWnd::OnCmdMsg (nID, nCode, pExtra, pHandlerInfo))
        return TRUE;

    //
    // Route to inactive views second.
    //
    CWandererDoc* pDoc = (CWandererDoc*) GetActiveDocument ();
    if (pDoc != NULL) { // Important!
        return pDoc->RouteCmdToAllViews (GetActiveView (),
            nID, nCode, pExtra, pHandlerInfo);
    }
    return FALSE;
}
```

WandererDoc.h

```
// WandererDoc.h : interface of the CWandererDoc class
//
/////////////////////////////////////////////////////////////////////////////
//
#if !defined(AFX_WANDERERDOC_H__AE0A7000_9B0F_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_WANDERERDOC_H__AE0A7000_9B0F_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CWandererDoc : public CDocument
{
protected: // create from serialization only
    CWandererDoc();
    DECLARE_DYNCREATE(CWandererDoc)

// Attributes
public:

// Operations
public:

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

// Implementation
public:
    BOOL RouteCmdToAllViews (CView* pView, UINT nID, int nCode,
        void* pExtra, AFX_CMDHANDLERINFO* pHandlerInfo);
    virtual ~CWandererDoc();
#ifdef _DEBUG
```

```
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif

protected:

// Generated message map functions
protected:
    //{{AFX_MSG(CWandererDoc)
       // 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_WANDERERDOC_H__AE0A7000_9B0F_11D2_8E53_006008A82731__INCLUDED_)
```

WandererDoc.cpp

```
// WandererDoc.cpp : implementation of the CWandererDoc class
//
#include "stdafx.h"
#include "Wanderer.h"

#include "WandererDoc.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CWandererDoc
IMPLEMENT_DYNCREATE(CWandererDoc, CDocument)

BEGIN_MESSAGE_MAP(CWandererDoc, CDocument)
    //{{AFX_MSG_MAP(CWandererDoc)
        // 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()
/////////////////////////////////////////////////////////////////////////////
// CWandererDoc construction/destruction
CWandererDoc::CWandererDoc()
{
}
CWandererDoc::~CWandererDoc()
{
}
BOOL CWandererDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
```

```
        return FALSE;
    return TRUE;
}
/////////////////////////////////////////////////////////////////////////////
// CWandererDoc serialization
void CWandererDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        // TODO: add storing code here
    }
    else
    {
        // TODO: add loading code here
    }
}
/////////////////////////////////////////////////////////////////////////////
// CWandererDoc diagnostics
#ifdef _DEBUG
void CWandererDoc::AssertValid() const
{
    CDocument::AssertValid();
}
void CWandererDoc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CWandererDoc commands
BOOL CWandererDoc::RouteCmdToAllViews(CView *pView, UINT nID, int nCode,
    void *pExtra, AFX_CMDHANDLERINFO *pHandlerInfo)
{
    POSITION pos = GetFirstViewPosition ();

    while (pos != NULL) {
        CView* pNextView = GetNextView (pos);
        if (pNextView != pView) {
            if (pNextView->OnCmdMsg (nID, nCode, pExtra, pHandlerInfo))
                return TRUE;
        }
    }
    return FALSE;
}
```

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:
    CWandererDoc* 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 void OnSelectionChanged(NMHDR* pNMHDR, LRESULT* pResult);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

#ifndef _DEBUG  // debug version in DriveTreeView.cpp
inline CWandererDoc* CDriveView::GetDocument()
    { return (CWandererDoc*)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__INCLUDED_)
```

DriveView.cpp

```
// DriveTreeView.cpp : implementation of the CDriveView class
//
#include "stdafx.h"
#include "Wanderer.h"

#include "WandererDoc.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)
    ON_NOTIFY_REFLECT(TVN_SELCHANGED, OnSelectionChanged)
    //}}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)
{
    CWandererDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
}
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);
    }
    //
    // Initialize the list view.
    //
    strPath = GetPathFromItem (GetTreeCtrl ().GetSelectedItem ());
    GetDocument ()->UpdateAllViews (this, 0x5A,
        (CObject*) (LPCTSTR) strPath);
}
/////////////////////////////////////////////////////////////////////////////
// CDriveView diagnostics
#ifdef _DEBUG
void CDriveView::AssertValid() const
{
    CTreeView::AssertValid();
}
void CDriveView::Dump(CDumpContext& dc) const
{
    CTreeView::Dump(dc);
}
CWandererDoc* CDriveView::GetDocument() // non-debug version is inline
{
```

```
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CWandererDoc)));
    return (CWandererDoc*)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;
}
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;
}
```

FileView.h

```
// FileView.h : interface of the CFileView class
//
/////////////////////////////////////////////////////////////////////////////
//
#if !defined(AFX_FILEVIEW_H__18BD7B80_95C6_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_FILEVIEW_H__18BD7B80_95C6_11D2_8E53_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:
    CWandererDoc* 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
    virtual void OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint);
    //}}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
    DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in FileView.cpp
inline CWandererDoc* CFileView::GetDocument()
```

```
    { return (CWandererDoc*)m_pDocument; }
#endif
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately
// before the previous line.
#endif
// !defined(AFX_FILEVIEW_H__18BD7B80_95C6_11D2_8E53_006008A82731__INCLUDED_)
```

FileView.cpp

```
// FileView.cpp : implementation of the CFileView class
//
#include "stdafx.h"
#include "Wanderer.h"
#include "WandererDoc.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)
    //}}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)
{
    CWandererDoc* 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);
}
CWandererDoc* CFileView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CWandererDoc)));
    return (CWandererDoc*)m_pDocument;
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CFileView message handlers
int CFileView::Refresh(LPCTSTR pszPath)
{
```

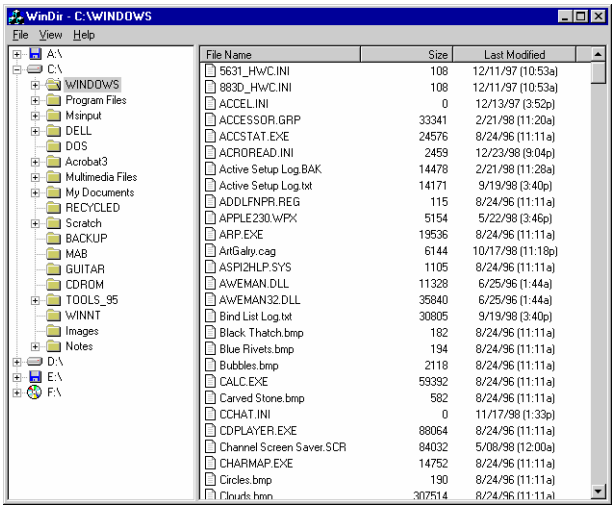
```
    CString strPath = pszPath;
    if (strPath.Right (1) != _T ("\\"))
        strPath += _T ("\\");
    strPath += _T ("*.*");

    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::OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)
{
    if (lHint == 0x5A) {
        FreeItemMemory ();
        GetListCtrl ().DeleteAllItems ();
        Refresh ((LPCTSTR) pHint);
        return;
    }
    CListView::OnUpdate (pSender, lHint, pHint);
}
```

Màn hình kết quả như sau:



2.4 TOOLBAR VÀ STATUSBAR

2.4.1 Vấn đề quan tâm

- Hiểu và sử dụng được các class về Toolbar, StatusBar.

2.4.2 ToolBar

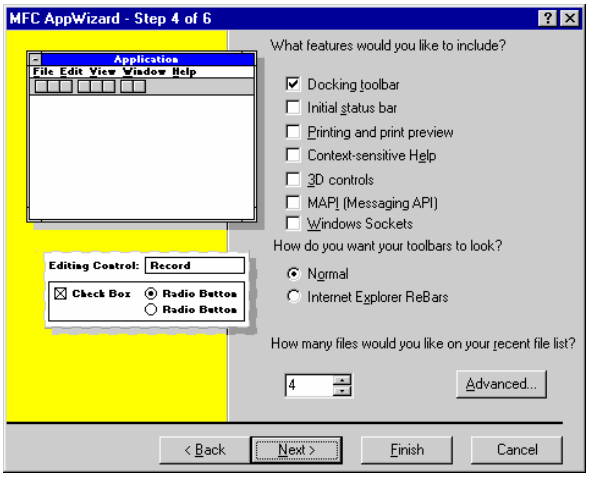
ToolBar là công cụ liên kết giữa các hình bitmap với các hàm chức năng tương ứng để thể hiện vai trò tương tự như 1 button

MFC cung cấp lớp CToolBar để giúp thao tác với toolbar thuận lợi hơn. Tuy nhiên class CToolBarCtrl còn giúp mở rộng khả năng thao tác với toolbar control (trong comctl32.dll)

Để tạo và khai báo 1 toolbar, cần thực hiện theo trình tự sau đây:

- Vào Resources View, thêm 1 toolbal vào và thêm/bớt các hình bitmap tương ứng.
- Có thể tạo 1 class liên kết với toolbar này hay sử dụng class CToolBar cơ sở
- Vào class CMainFrame
  - Thêm biến liên kết với toolbar(có kiểu là CToolBar hay class tự tạo)
  - Thêm lệnh khởi tạo cho toolbar này trong hàm OnCreate và thiết lập các đặc tính tương ứng.
  - Liên kết với document bởi việc gọi hàm EnableDocking và DockControlBar
  - Thêm các hàm chức năng tương ứng với các bitmap trong toolbar để đáp ứng các sự kiện tương tác với chúng.

Sử dụng sự hỗ trợ từ AppWizard như sau:



Ví dụ 1:

Trong Resources View

```
// Tạo 1 toolbar
IDR_MYTOOLBAR BITMAP MyToolbar.bmp
// Các thành phần của toolbar này
IDR_MYTOOLBAR TOOLBAR 16, 15
BEGIN
  BUTTON ID_CHAR_BOLD
  BUTTON ID_CHAR_ITALIC
  BUTTON ID_CHAR_UNDERLINE
  SEPARATOR
  BUTTON ID_PARA_LEFT
  BUTTON ID_PARA_CENTER
  BUTTON ID_PARA_RIGHT
END
```

// Trong file H

```
class CMainFrame : public CFrameWnd
{
…
protected:
CToolBar m_wndMyToolBar;
…
}
```

Trong file CPP

```
BOOL CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
m_wndMyToolBar.Create(this, TBSTYLE_FLAT ¦ WS_CHILD ¦ WS_VISIBLE ¦ CBRS_BOTTOM ¦
CBRS_TOOLTIPS | CBRS_FLYBY);
m_wndMyToolBar.LoadToolBar(IDR_MYTOOLBAR);

m_wndMyToolBar.SetButtonText(0, _T("My Bold"));
m_wndMyToolBar.SetButtonText(1, _T("My Italic"));
m_wndMyToolBar.SetButtonText(2, _T("My Underline"));
m_wndMyToolBar.SetButtonText(4, _T("My Left"));
m_wndMyToolBar.SetButtonText(5, _T("My Center"));
m_wndMyToolBar.SetButtonText(6, _T("My Right"));
m_wndMyToolBar.SetSizes(CSize(48, 42), CSize(40, 19));

m_wndMyToolBar.SetButtonStyle(0, TBBS_CHECKBOX);
m_wndMyToolBar.SetButtonStyle(1, TBBS_CHECKBOX);
m_wndMyToolBar.SetButtonStyle(2, TBBS_CHECKBOX);
m_wndMyToolBar.SetButtonStyle(4, TBBS_CHECKGROUP);
m_wndMyToolBar.SetButtonStyle(5, TBBS_CHECKGROUP);
m_wndMyToolBar.SetButtonStyle(6, TBBS_CHECKGROUP);

m_wndMyToolBar.EnableDocking(CBRS_ALIGN_TOP ¦ CBRS_ALIGN_BOTTOM);
EnableDocking(CBRS_ALIGN_ANY);
DockControlBar(&m_wndMyToolBar);

//FloatControlBar(&m_wndMyToolBar, CPoint(50, 100));
}
```

Ví dụ 2:

Trong file RC

```
IDR_TOOLBAR BITMAP Toolbar.bmp

IDR_TOOLBAR TOOLBAR 16, 15
BEGIN
    BUTTON ID_FILE_NEW
    BUTTON ID_FILE_OPEN
    BUTTON ID_FILE_SAVE
    SEPARATOR
    BUTTON ID_EDIT_CUT
    BUTTON ID_EDIT_COPY
    BUTTON ID_EDIT_PASTE
    BUTTON ID_EDIT_UNDO
    SEPARATOR
    BUTTON ID_FILE_PRINT
END
```

Trong file Cpp

```
m_wndToolBar.Create (this);
m_wndToolBar.LoadToolBar (IDR_TOOLBAR);
```

Trong file RC

```
IDR_TOOLBAR BITMAP Toolbar.bmp
```

```
IDR_TOOLBAR TOOLBAR 40, 19
```

Trong file Cpp

```
m_wndToolBar.Create (this);
m_wndToolBar.LoadToolBar (IDR_TOOLBAR);

m_wndToolBar.SetButtonText (0, _T ("New"));
m_wndToolBar.SetButtonText (1, _T ("Open"));
m_wndToolBar.SetButtonText (2, _T ("Save"));
m_wndToolBar.SetButtonText (4, _T ("Cut"));
m_wndToolBar.SetButtonText (5, _T ("Copy"));
m_wndToolBar.SetButtonText (6, _T ("Paste"));
m_wndToolBar.SetButtonText (7, _T ("Undo"));
m_wndToolBar.SetButtonText (9, _T ("Print"));

m_wndToolBar.SetSizes (CSize (48, 42), CSize (40, 19));
```

2.4.3 StatusBar

StatusBar là công cụ thể hiện thông tin trong cửa sổ ứng dụng.

MFC cung cấp class CStatusBar để giúp thao tác với statusbar thuận lợi hơn.

Để tạo và khai báo 1 statusbar, cần thực hiện theo trình tự sau đây:

- Xác định danh sách các phần(pane) trong statusbar.
- Vào class CMainFrame
  - Thêm biến liên kết với statusbar(có kiểu là CStatusBar)
  - Thêm lệnh khởi tạo cho toolbar này trong hàm OnCreate và thiết lập các đặc tính tương ứng.
  - Gọi hàm SetIndicators để khai báo số lượng pane trong statusbar
  - Để tương tác(điều khiển, hiển thị thông tin…) , truy xuất thông qua biến liên kết các hàm như SetPaneText

Ví dụ 1:

Trong Resources View

```
Khai báo các thành phần trong statusbar
static UINT nIndicators[] = {
  ID_SEPARATOR,
  ID_SEPARATOR,
  ID_SEPARATOR
};

STRINGTABLE DISCARDABLE DISCARDABLE
BEGIN
  ID_INDICATOR_EXT    "EXT"
  ID_INDICATOR_CAPS    "CAP"
  ID_INDICATOR_NUM    "NUM"
  ID_INDICATOR_SCRL    "SCRL"
  ID_INDICATOR_OVR    "OVR"
  ID_INDICATOR_REC    "REC"
END
```

Trong file H

```
class CMainFrame : public CFrameWnd
{
…
protected:
CStatusBar m_wndMyStatusBar;
…
```

```
}
```

Trong file CPP

```
// In the message map
ON_UPDATE_COMMAND_UI(ID_INDICATOR_TIME, OnUpdateTime)

BOOL CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
       …
m_wndMyStatusBar.Create(this);
m_wndSMytatusBar.SetIndicators(nIndicators, 3);

m_wndMyStatusBar.SetPaneInfo(0, ID_SEPARATOR, SBPS_NOBORDERS, 64);
m_wndMyStatusBar.SetPaneInfo(1, ID_SEPARATOR, SBPS_POPOUT, 64);
m_wndMyStatusBar.SetPaneInfo(2, ID_SEPARATOR, SBPS_NORMAL ¦
       SBPS_STRETCH, 0);
…
SetTimer(ID_TIMER, 200, NULL);
       …
}

void CMainFrame::OnTimer(UINT nTimerID)
{
  CTime time = CTime::GetCurrentTime();
  int nSecond = time.GetSecond();
  int nMinute = time.GetMinute();
  int nHour = time.GetHour() % 12;

  CString string;
  string.Format(_T("%0.2d:%0.2d:%0.2d"), nHour, nMinute, nSecond);
  m_wndMyStatusBar.SetPaneText(2, string);
}

void CMainFrame::OnUpdateTime(CCmdUI* pCmdUI)
{
  CTime time = CTime::GetCurrentTime();
  int nSecond = time.GetSecond();
  int nMinute = time.GetMinute();
  int nHour = time.GetHour() % 12;

  CString string;
  string.Format(_T("%0.2d:%0.2d:%0.2d"), nHour, nMinute, nSecond);
  pCmdUI->SetText(string);
}
```

Ví dụ tổng hợp

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_MAINFRM_H__C85C9089_A154_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__C85C9089_A154_11D2_8E53_006008A82731__INCLUDED_

#include "StyleBar.h"    // Added by ClassView
#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

protected:  // control bar embedded members
    CStyleBar   m_wndStyleBar;
    CStatusBar  m_wndStatusBar;
    CToolBar    m_wndToolBar;
// Generated message map functions
protected:
    BOOL CreateToolBar ();
    BOOL CreateStyleBar ();
    BOOL CreateStatusBar ();
    //{{AFX_MSG(CMainFrame)
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    afx_msg void OnClose();
    //}}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__C85C9089_A154_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//
#include "stdafx.h"
#include "MyWord.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)
    ON_WM_CREATE()
    ON_WM_CLOSE()
    //}}AFX_MSG_MAP

ON_COMMAND_EX (IDW_STYLE_BAR, OnBarCheck)
    ON_UPDATE_COMMAND_UI (IDW_STYLE_BAR, OnUpdateControlBarMenu)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction
CMainFrame::CMainFrame()
{
}
CMainFrame::~CMainFrame()
{
}
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
        return -1;
    //
    // Tell the frame window to permit docking.
    //
    EnableDocking (CBRS_ALIGN_ANY);
    //
    // Create the toolbar, style bar, and status bar.
    //
    if (!CreateToolBar () ||
        !CreateStyleBar () ||
        !CreateStatusBar ())
        return -1;
    //
    // Load the saved bar state (if any).
    //
    LoadBarState (_T ("MainBarState"));
    return 0;
}
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
void CMainFrame::OnClose()
{
    SaveBarState (_T ("MainBarState"));
    CFrameWnd::OnClose();
}
BOOL CMainFrame::CreateToolBar()
{
    if (!m_wndToolBar.Create (this) ||
        !m_wndToolBar.LoadToolBar (IDR_MAINFRAME))
        return FALSE;

    m_wndToolBar.SetBarStyle (m_wndToolBar.GetBarStyle () |
        CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);

    m_wndToolBar.SetWindowText (_T ("Main"));
    m_wndToolBar.EnableDocking (CBRS_ALIGN_ANY);
    DockControlBar (&m_wndToolBar);
    return TRUE;
}
BOOL CMainFrame::CreateStyleBar()
{
    if (!m_wndStyleBar.Create (this, WS_CHILD | WS_VISIBLE | CBRS_TOP |
        CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC, IDW_STYLE_BAR))
        return FALSE;

    m_wndStyleBar.SetWindowText (_T ("Styles"));
    m_wndStyleBar.EnableDocking (CBRS_ALIGN_TOP | CBRS_ALIGN_BOTTOM);
    DockControlBar (&m_wndStyleBar);
    return TRUE;
}
BOOL CMainFrame::CreateStatusBar()
{
    static UINT nIndicators[] = {
        ID_SEPARATOR,
        ID_INDICATOR_LINE,
        ID_INDICATOR_CAPS,
        ID_INDICATOR_NUM
    };

    if (!m_wndStatusBar.Create (this))
        return FALSE;

    m_wndStatusBar.SetIndicators (nIndicators, 4);
```

```
    return TRUE;
}
```

MyWordDoc.h

```
// MyWordDoc.h : interface of the CMyWordDoc class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_MYWORDDOC_H__C85C908B_A154_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MYWORDDOC_H__C85C908B_A154_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CMyWordDoc : public CRichEditDoc
{
protected: // create from serialization only
    CMyWordDoc();
    DECLARE_DYNCREATE(CMyWordDoc)

// Attributes
public:

// Operations
public:

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

    virtual BOOL OnNewDocument();
    virtual void Serialize(CArchive& ar);
    //}}AFX_VIRTUAL
    virtual CRichEditCntrItem* CreateClientItem(REOBJECT* preo) const;

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

protected:

// Generated message map functions
protected:
    //{{AFX_MSG(CMyWordDoc)
       // 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_MYWORDDOC_H__C85C908B_A154_11D2_8E53_006008A82731__INCLUDED_)
```

MyWordDoc.cpp

```
// MyWordDoc.cpp : implementation of the CMyWordDoc class
//
#include "stdafx.h"
#include "MyWord.h"

#include "MyWordDoc.h"
#include "CntrItem.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CMyWordDoc
IMPLEMENT_DYNCREATE(CMyWordDoc, CRichEditDoc)

BEGIN_MESSAGE_MAP(CMyWordDoc, CRichEditDoc)
    //{{AFX_MSG_MAP(CMyWordDoc)
        // 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
    // Enable default OLE container implementation
    ON_UPDATE_COMMAND_UI(ID_OLE_EDIT_LINKS,
        CRichEditDoc::OnUpdateEditLinksMenu)
    ON_COMMAND(ID_OLE_EDIT_LINKS, CRichEditDoc::OnEditLinks)
    ON_UPDATE_COMMAND_UI_RANGE(ID_OLE_VERB_FIRST,
        ID_OLE_VERB_LAST, CRichEditDoc::OnUpdateObjectVerbMenu)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CMyWordDoc construction/destruction
CMyWordDoc::CMyWordDoc()
{
}
CMyWordDoc::~CMyWordDoc()
{
}
BOOL CMyWordDoc::OnNewDocument()
{
    if (!CRichEditDoc::OnNewDocument())
        return FALSE;
    return TRUE;
}
CRichEditCntrItem* CMyWordDoc::CreateClientItem(REOBJECT* preo) const
{
    return new CMyWordCntrItem(preo, (CMyWordDoc*) this);
}
/////////////////////////////////////////////////////////////////////////////
// CMyWordDoc serialization
```

```
void CMyWordDoc::Serialize(CArchive& ar)
{
    CRichEditDoc::Serialize(ar);
}
/////////////////////////////////////////////////////////////////////////////
// CMyWordDoc diagnostics
#ifdef _DEBUG
void CMyWordDoc::AssertValid() const
{
    CRichEditDoc::AssertValid();
}
void CMyWordDoc::Dump(CDumpContext& dc) const
{
    CRichEditDoc::Dump(dc);
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CMyWordDoc commands
```

MyWordView.h

```
// MyWordView.h : interface of the CMyWordView class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_MYWORDVIEW_H__C85C908D_A154_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MYWORDVIEW_H__C85C908D_A154_11D2_8E53_006008A82731__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMyWordCntrItem;
class CMyWordView : public CRichEditView
{
protected: // create from serialization only
    CMyWordView();
    DECLARE_DYNCREATE(CMyWordView)
// Attributes
public:
    CMyWordDoc* GetDocument();

// Operations
public:
    void GetFontInfo (LPTSTR pszFaceName, int& nSize);
    void ChangeFont (LPCTSTR pszFaceName);
    void ChangeFontSize (int nSize);

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMyWordView)
    public:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    protected:
    virtual void OnInitialUpdate(); // called first time after construct
    //}}AFX_VIRTUAL

// Implementation
```

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

protected:

// Generated message map functions
protected:
    //{{AFX_MSG(CMyWordView)
    afx_msg void OnDestroy();
    afx_msg void OnCharBold();
    afx_msg void OnCharItalic();
    afx_msg void OnCharUnderline();
    afx_msg void OnParaLeft();
    afx_msg void OnParaCenter();
    afx_msg void OnParaRight();
    afx_msg void OnUpdateCharBold(CCmdUI* pCmdUI);
    afx_msg void OnUpdateCharItalic(CCmdUI* pCmdUI);
    afx_msg void OnUpdateCharUnderline(CCmdUI* pCmdUI);
    afx_msg void OnUpdateParaLeft(CCmdUI* pCmdUI);
    afx_msg void OnUpdateParaCenter(CCmdUI* pCmdUI);
    afx_msg void OnUpdateParaRight(CCmdUI* pCmdUI);
    //}}AFX_MSG
    afx_msg void OnUpdateLineNumber (CCmdUI* pCmdUI);
    DECLARE_MESSAGE_MAP()
};

#ifndef _DEBUG  // debug version in MyWordView.cpp
inline CMyWordDoc* CMyWordView::GetDocument()
    { return (CMyWordDoc*)m_pDocument; }
#endif
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//     AFX_MYWORDVIEW_H__C85C908D_A154_11D2_8E53_006008A82731__INCLUDED_)
```

MyWordView.cpp

```
// MyWordView.cpp : implementation of the CMyWordView class
//
#include "stdafx.h"
#include "MyWord.h"
#include "MyWordDoc.h"
#include "CntrItem.h"
#include "MyWordView.h"

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

```
/////////////////////////////////////////////////////////////////////////////
// CMyWordView
IMPLEMENT_DYNCREATE(CMyWordView, CRichEditView)
BEGIN_MESSAGE_MAP(CMyWordView, CRichEditView)
    //{{AFX_MSG_MAP(CMyWordView)
    ON_WM_DESTROY()
    ON_COMMAND(ID_CHAR_BOLD, OnCharBold)
    ON_COMMAND(ID_CHAR_ITALIC, OnCharItalic)
    ON_COMMAND(ID_CHAR_UNDERLINE, OnCharUnderline)
    ON_COMMAND(ID_PARA_LEFT, OnParaLeft)
    ON_COMMAND(ID_PARA_CENTER, OnParaCenter)
    ON_COMMAND(ID_PARA_RIGHT, OnParaRight)
    ON_UPDATE_COMMAND_UI(ID_CHAR_BOLD, OnUpdateCharBold)
    ON_UPDATE_COMMAND_UI(ID_CHAR_ITALIC, OnUpdateCharItalic)
    ON_UPDATE_COMMAND_UI(ID_CHAR_UNDERLINE, OnUpdateCharUnderline)
    ON_UPDATE_COMMAND_UI(ID_PARA_LEFT, OnUpdateParaLeft)
    ON_UPDATE_COMMAND_UI(ID_PARA_CENTER, OnUpdateParaCenter)
    ON_UPDATE_COMMAND_UI(ID_PARA_RIGHT, OnUpdateParaRight)
    //}}AFX_MSG_MAP
    ON_UPDATE_COMMAND_UI(ID_INDICATOR_LINE, OnUpdateLineNumber)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CMyWordView construction/destruction
CMyWordView::CMyWordView()
{
}
CMyWordView::~CMyWordView()
{
}
BOOL CMyWordView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CRichEditView::PreCreateWindow(cs);
}
void CMyWordView::OnInitialUpdate()
{
    CRichEditView::OnInitialUpdate();

    CHARFORMAT cf;
    cf.cbSize = sizeof (CHARFORMAT);
    cf.dwMask = CFM_BOLD | CFM_ITALIC | CFM_UNDERLINE |
        CFM_PROTECTED | CFM_STRIKEOUT | CFM_FACE | CFM_SIZE;
    cf.dwEffects = 0;
    cf.yHeight = 240; // 240 twips == 12 points
    ::lstrcpy (cf.szFaceName, _T ("Times New Roman"));
    SetCharFormat (cf);
}
void CMyWordView::OnDestroy()
{
    // Deactivate the item on destruction; this is important
    // when a splitter view is being used.
   CRichEditView::OnDestroy();
   COleClientItem* pActiveItem = GetDocument()->GetInPlaceActiveItem(this);
   if (pActiveItem != NULL && pActiveItem->GetActiveView() == this)
   {
       pActiveItem->Deactivate();
```

```
       ASSERT(GetDocument()->GetInPlaceActiveItem(this) == NULL);
   }
}
/////////////////////////////////////////////////////////////////////////////
// CMyWordView diagnostics
#ifdef _DEBUG
void CMyWordView::AssertValid() const
{
    CRichEditView::AssertValid();
}
void CMyWordView::Dump(CDumpContext& dc) const
{
    CRichEditView::Dump(dc);
}
CMyWordDoc* CMyWordView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CMyWordDoc)));
    return (CMyWordDoc*)m_pDocument;
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CMyWordView message handlers
void CMyWordView::OnCharBold()
{
    CHARFORMAT cf;
    cf = GetCharFormatSelection ();

    if (!(cf.dwMask & CFM_BOLD) || !(cf.dwEffects & CFE_BOLD))
        cf.dwEffects = CFE_BOLD;
    else
        cf.dwEffects = 0;

    cf.dwMask = CFM_BOLD;
    SetCharFormat (cf);
}
void CMyWordView::OnCharItalic()
{
    CHARFORMAT cf;
    cf = GetCharFormatSelection ();

    if (!(cf.dwMask & CFM_ITALIC) || !(cf.dwEffects & CFE_ITALIC))
        cf.dwEffects = CFE_ITALIC;
    else
        cf.dwEffects = 0;

    cf.dwMask = CFM_ITALIC;
    SetCharFormat (cf);
}
void CMyWordView::OnCharUnderline()
{
    CHARFORMAT cf;
    cf = GetCharFormatSelection ();

    if (!(cf.dwMask & CFM_UNDERLINE) || !(cf.dwEffects & CFE_UNDERLINE))
        cf.dwEffects = CFE_UNDERLINE;
```

```
    else
        cf.dwEffects = 0;

    cf.dwMask = CFM_UNDERLINE;
    SetCharFormat (cf);
}
void CMyWordView::OnParaLeft()
{
    OnParaAlign (PFA_LEFT);
}
void CMyWordView::OnParaCenter()
{
    OnParaAlign (PFA_CENTER);
}
void CMyWordView::OnParaRight()
{
    OnParaAlign (PFA_RIGHT);
}
void CMyWordView::OnUpdateCharBold(CCmdUI* pCmdUI)
{
    OnUpdateCharEffect (pCmdUI, CFM_BOLD, CFE_BOLD);
}
void CMyWordView::OnUpdateCharItalic(CCmdUI* pCmdUI)
{
    OnUpdateCharEffect (pCmdUI, CFM_ITALIC, CFE_ITALIC);
}
void CMyWordView::OnUpdateCharUnderline(CCmdUI* pCmdUI)
{
    OnUpdateCharEffect (pCmdUI, CFM_UNDERLINE, CFE_UNDERLINE);
}
void CMyWordView::OnUpdateParaLeft(CCmdUI* pCmdUI)
{
    OnUpdateParaAlign (pCmdUI, PFA_LEFT);
}
void CMyWordView::OnUpdateParaCenter(CCmdUI* pCmdUI)
{
    OnUpdateParaAlign (pCmdUI, PFA_CENTER);
}
void CMyWordView::OnUpdateParaRight(CCmdUI* pCmdUI)
{
    OnUpdateParaAlign (pCmdUI, PFA_RIGHT);
}
void CMyWordView::OnUpdateLineNumber(CCmdUI* pCmdUI)
{
    int nLine = GetRichEditCtrl ().LineFromChar (-1) + 1;

    CString string;
    string.Format (_T ("Line %d"), nLine);
    pCmdUI->Enable (TRUE);
    pCmdUI->SetText (string);
}
void CMyWordView::ChangeFont(LPCTSTR pszFaceName)
{
    CHARFORMAT cf;
    cf.cbSize = sizeof (CHARFORMAT);
```

```
    cf.dwMask = CFM_FACE;
    ::lstrcpy (cf.szFaceName, pszFaceName);
    SetCharFormat (cf);
}
void CMyWordView::ChangeFontSize(int nSize)
{
    CHARFORMAT cf;
    cf.cbSize = sizeof (CHARFORMAT);
    cf.dwMask = CFM_SIZE;
    cf.yHeight = nSize;
    SetCharFormat (cf);
}
void CMyWordView::GetFontInfo(LPTSTR pszFaceName, int& nSize)
{
    CHARFORMAT cf = GetCharFormatSelection ();
    ::lstrcpy (pszFaceName,
        cf.dwMask & CFM_FACE ? cf.szFaceName : _T (""));
    nSize = cf.dwMask & CFM_SIZE ? cf.yHeight : -1;
}
```

StyleBar.h

```
#if !defined(
    AFX_STYLEBAR_H__C85C9099_A154_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_STYLEBAR_H__C85C9099_A154_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// StyleBar.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CStyleBar command target
class CStyleBar : public CToolBar
{
// Attributes
public:

// Operations
public:
    static int CALLBACK EnumFontNameProc (ENUMLOGFONT* lpelf,
    NEWTEXTMETRIC* lpntm, int nFontType, LPARAM lParam);

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CStyleBar)
    //}}AFX_VIRTUAL
    virtual void OnUpdateCmdUI (CFrameWnd* pTarget,
        BOOL bDisableIfNoHndler);
// Implementation
protected:
    void InitTypefaceList (CDC* pDC);
    CFont m_font;
    CComboBox m_wndFontNames;
    CComboBox m_wndFontSizes;
    // Generated message map functions
    //{{AFX_MSG(CStyleBar)
```

```
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    //}}AFX_MSG
    afx_msg void OnSelectFont ();
    afx_msg void OnSelectSize ();
    afx_msg void OnCloseUp ();
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//     AFX_STYLEBAR_H__C85C9099_A154_11D2_8E53_006008A82731__INCLUDED_)
```

StyleBar.cpp

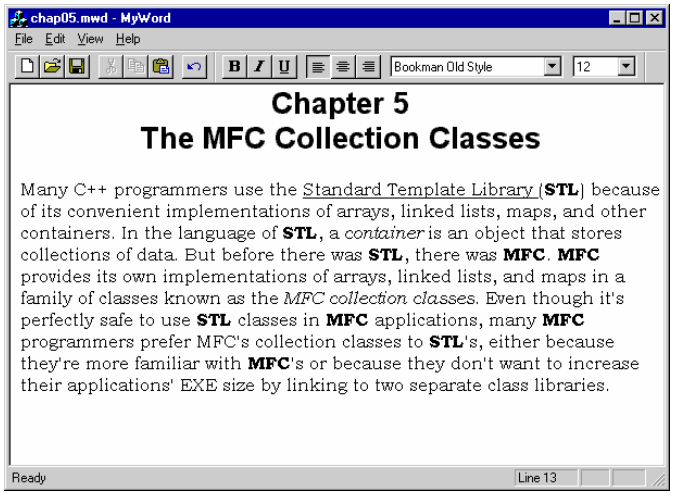
```
// StyleBar.cpp : implementation file
//
#include "stdafx.h"
#include "MyWord.h"
#include "MyWordDoc.h"
#include "MyWordView.h"
#include "StyleBar.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CStyleBar
BEGIN_MESSAGE_MAP(CStyleBar, CToolBar)
    //{{AFX_MSG_MAP(CStyleBar)
    ON_WM_CREATE()
    //}}AFX_MSG_MAP
    ON_CBN_SELENDOK (IDC_FONTNAMES, OnSelectFont)
    ON_CBN_SELENDOK (IDC_FONTSIZES, OnSelectSize)
    ON_CBN_CLOSEUP (IDC_FONTNAMES, OnCloseUp)
    ON_CBN_CLOSEUP (IDC_FONTSIZES, OnCloseUp)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CStyleBar message handlers
int CStyleBar::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    static int nFontSizes[] = {
        8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 32, 36, 48, 72
    };
    if (CToolBar::OnCreate(lpCreateStruct) == -1)
        return -1;
    //
    // Load the toolbar.
    //
    if (!LoadToolBar (IDR_STYLE_BAR))
        return -1;
    //
    // Create an 8-point MS Sans Serif font for the combo boxes.
```

```
    //
    CClientDC dc (this);
    m_font.CreatePointFont (80, _T ("MS Sans Serif"));
    CFont* pOldFont = dc.SelectObject (&m_font);

    TEXTMETRIC tm;
    dc.GetTextMetrics (&tm);
    int cxChar = tm.tmAveCharWidth;
    int cyChar = tm.tmHeight + tm.tmExternalLeading;

    dc.SelectObject (pOldFont);
    //
    // Add the font name combo box to the toolbar.
    //
    SetButtonInfo (8, IDC_FONTNAMES, TBBS_SEPARATOR, cxChar * 32);

    CRect rect;
    GetItemRect (8, &rect);
    rect.bottom = rect.top + (cyChar * 16);

    if (!m_wndFontNames.Create (WS_CHILD | WS_VISIBLE | WS_VSCROLL |
        CBS_DROPDOWNLIST | CBS_SORT, rect, this, IDC_FONTNAMES))
        return -1;

    m_wndFontNames.SetFont (&m_font);
    InitTypefaceList (&dc);
    //
    // Add the font size combo box to the toolbar.
    //
    SetButtonInfo (10, IDC_FONTSIZES, TBBS_SEPARATOR, cxChar * 12);

    GetItemRect (10, &rect);
    rect.bottom = rect.top + (cyChar * 14);

    if (!m_wndFontSizes.Create (WS_CHILD | WS_VISIBLE | WS_VSCROLL |
        CBS_DROPDOWNLIST, rect, this, IDC_FONTSIZES))
        return -1;

    m_wndFontSizes.SetFont (&m_font);
    CString string;
    int nCount = sizeof (nFontSizes) / sizeof (int);
    for (int i=0; i<nCount; i++) {
        string.Format (_T ("%d"), nFontSizes[i]);
        m_wndFontSizes.AddString (string);
    }
    return 0;
}
void CStyleBar::OnSelectFont ()
{
    TCHAR szFaceName[LF_FACESIZE];
    int nIndex = m_wndFontNames.GetCurSel ();
    m_wndFontNames.GetLBText (nIndex, szFaceName);

    CMyWordView* pView =
        (CMyWordView*) ((CFrameWnd*) AfxGetMainWnd ())->GetActiveView ();
```

```
    pView->ChangeFont (szFaceName);
}
void CStyleBar::OnSelectSize ()
{
    TCHAR szSize[8];
    int nIndex = m_wndFontSizes.GetCurSel ();
    m_wndFontSizes.GetLBText (nIndex, szSize);

    int nSize = atoi (szSize) * 20; // Need twips

    CMyWordView* pView =
        (CMyWordView*) ((CFrameWnd*) AfxGetMainWnd ())->GetActiveView ();
    pView->ChangeFontSize (nSize);
}
void CStyleBar::OnCloseUp ()
{
    ((CFrameWnd*) AfxGetMainWnd ())->GetActiveView ()->SetFocus ();
}
void CStyleBar::InitTypefaceList (CDC* pDC)
{
    ::EnumFontFamilies (pDC->m_hDC, NULL,
        (FONTENUMPROC) EnumFontNameProc, (LPARAM) this);
}
int CALLBACK CStyleBar::EnumFontNameProc (ENUMLOGFONT* lpelf,
    NEWTEXTMETRIC* lpntm, int nFontType, LPARAM lParam)
{
    CStyleBar* pWnd = (CStyleBar*) lParam;
    if (nFontType & TRUETYPE_FONTTYPE)
        pWnd->m_wndFontNames.AddString (lpelf->elfLogFont.lfFaceName);
    return 1;
}
void CStyleBar::OnUpdateCmdUI (CFrameWnd* pTarget, BOOL bDisableIfNoHndler)
{
    CToolBar::OnUpdateCmdUI (pTarget, bDisableIfNoHndler);

    CWnd* pWnd = GetFocus ();
    if ((pWnd == &m_wndFontNames) || (pWnd == &m_wndFontSizes))
        return;
    //
    // Get the font name and size.
    //
    int nTwips;
    TCHAR szFaceName[LF_FACESIZE];

    CMyWordView* pView =
        (CMyWordView*) ((CFrameWnd*) AfxGetMainWnd ())->GetActiveView ();
    pView->GetFontInfo (szFaceName, nTwips);
    //
    // Update the font name combo box.
    //
    TCHAR szSelection[LF_FACESIZE];
    m_wndFontNames.GetWindowText (szSelection,
        sizeof (szSelection) / sizeof (TCHAR));

    if (::lstrcmp (szFaceName, szSelection) != 0) {
```

```
        if (szFaceName[0] == 0)
            m_wndFontNames.SetCurSel (-1);
        else {
            if (m_wndFontNames.SelectString (-1, szFaceName) == CB_ERR)
                m_wndFontNames.SetCurSel (-1);
        }
    }
    //
    // Update the font size combo box.
    //
    TCHAR szSize[4];
    m_wndFontSizes.GetWindowText (szSize,
        sizeof (szSize) / sizeof (TCHAR));
    int nSizeFromComboBox = atoi (szSize);
    int nSizeFromView = nTwips / 20;

    if (nSizeFromComboBox != nSizeFromView) {
        if (nTwips == -1)
            m_wndFontSizes.SetCurSel (-1);
        else {
            CString string;
            string.Format (_T ("%d"), nSizeFromView);
            if (m_wndFontSizes.SelectString (-1, string) == CB_ERR)
                m_wndFontSizes.SetCurSel (-1);
        }
    }
}
```

Màn hình kết quả như sau:



CHƯƠNG 3. XỬ LÝ HỆ THỐNG

3.1 TIMER/IDLE

3.1.1 Vấn đề quan tâm

- Hiểu và sử dụng được các class về Timer/Idles.

3.1.2 Timer

Việc tạo sự kiện thực thi lặp đi lặp lại theo chu kỳ nhất định được giải quyết bởi việc dùng WM_TIMER

Để tạo sự kiện dạng Timer, cần thực hiện theo trình tự sau:

- Vào *Add Windows Message Handler* thêm message **WM_TIMER** vào ứng dụng, hàm OnTimer xuất hiện sau khi thêm.
- Trong hàm OnCreate của class, thêm hàm **SetTimer** để khai báo thời gian(chu kỳ) lặp lại(thời gian tối đa là 2^32-1 milliseconds)
- Hiện thực các thao tác trong hàm **OnTimer** để đáp ứng sự kiện WM_TIMER.

Ví dụ 1:

```
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
…
  ON_WM_CREATE()
  ON_WM_TIMER()
…
END_MESSAGE_MAP()

int CMainFrame::OnCreate(LPCREATESTRUCT lpcs)
{
  if(CFrameWnd::OnCreate(lpcs) == -1)
    return -1;

      // ID_TIMER_ELLIPSE là timerID của sự kiện Timer này
  if(!SetTimer(ID_TIMER_ELLIPSE, 100, NULL)) {
    MessageBox(_T("Error: SetTimer failed"));
    return -1;
  }
  return 0;
}
```

hay:

```
void CMainFrame::OnTimer(UINT nTimerID)
{
  CRect rect;
  GetClientRect(&rect);

  int x1 = rand() % rect.right;
  int x2 = rand() % rect.right;
  int y1 = rand() % rect.bottom;
  int y2 = rand() % rect.bottom;

  CClientDC dc(this);
  CBrush brush(RGB(rand() % 255, rand() % 255,
    rand() % 255));
  CBrush* pOldBrush = dc.SelectObject(&brush);
  dc.Ellipse(min(x1, x2), min(y1, y2), max(x1, x2),
    max(y1, y2));
  dc.SelectObject(pOldBrush);
}
```

Để ngừng sự kiện dạng Timer, dung hàm **KillTimer(timerID)**

Ví dụ 2:

```
KillTimer(ID_TIMER_ELLIPSE);
```

3.1.3 Idles

Việc làm xử lý gián đoạn trong hệ thống được giải quyết bởi hàm Idle hay sự kiện OnIdle

Ví dụ 1:

```
BOOL CMyApp::OnIdle(LONG lCount)
{
  BOOL bMFCContinue = CWinApp::OnIdle(lCount);
  BOOL bAppContinue = TRUE;
  if(lCount >= 2)
    bAppContinue = DoIdleWork(); // Do custom idle processing.
  return(bMFCContinue œœ bAppContinue);
}
```

Xử lý Idle là trái ngược với xử lý đa luồng (*multithreads*)

Ví dụ tổng hợp:

Resource.h

```
#define IDM_SYSMENU_FULL_WINDOW      16
#define IDM_SYSMENU_STAY_ON_TOP      32
#define IDI_APPICON                 100
```

Clock.rc

```
#include <afxres.h>
#include "Resource.h"

IDI_APPICON ICON Clock.ico
```

Clock.h

```
class CMyApp : public CWinApp
{
public:
    virtual BOOL InitInstance ();
};
class CMainWindow : public CFrameWnd
{
protected:
    BOOL m_bFullWindow;
    BOOL m_bStayOnTop;

    int m_nPrevSecond;
    int m_nPrevMinute;
    int m_nPrevHour;

    void DrawClockFace (CDC* pDC);
    void DrawSecondHand (CDC* pDC, int nLength, int nScale, int nDegrees,
        COLORREF clrColor);
    void DrawHand (CDC* pDC, int nLength, int nScale, int nDegrees,
        COLORREF clrColor);

    void SetTitleBarState ();
    void SetTopMostState ();
    void SaveWindowState ();
    void UpdateSystemMenu (CMenu* pMenu);

public:
    CMainWindow ();
```

```
    virtual BOOL PreCreateWindow (CREATESTRUCT& cs);
    BOOL RestoreWindowState ();

protected:
    afx_msg int OnCreate (LPCREATESTRUCT lpcs);
    afx_msg void OnGetMinMaxInfo (MINMAXINFO* pMMI);
    afx_msg void OnTimer (UINT nTimerID);
    afx_msg void OnPaint ();
    afx_msg UINT OnNcHitTest (CPoint point);
    afx_msg void OnSysCommand (UINT nID, LPARAM lParam);
    afx_msg void OnContextMenu (CWnd* pWnd, CPoint point);
    afx_msg void OnEndSession (BOOL bEnding);
    afx_msg void OnClose ();

    DECLARE_MESSAGE_MAP ()
};
```

Clock.cpp

```
#include <afxwin.h>
#include <math.h>
#include "Clock.h"
#include "Resource.h"
#define SQUARESIZE 20
#define ID_TIMER_CLOCK 1

CMyApp myApp;
////////////////////////////////////////////////////////////////////////
// CMyApp member functions

BOOL CMyApp::InitInstance ()
{
    SetRegistryKey (_T ("Programming Windows with MFC"));
    m_pMainWnd = new CMainWindow;
    if (!((CMainWindow*) m_pMainWnd)->RestoreWindowState ())
        m_pMainWnd->ShowWindow (m_nCmdShow);
    m_pMainWnd->UpdateWindow ();
    return TRUE;
}
////////////////////////////////////////////////////////////////////////
// CMainWindow message map and member functions
BEGIN_MESSAGE_MAP (CMainWindow, CFrameWnd)
    ON_WM_CREATE ()
    ON_WM_PAINT ()
    ON_WM_TIMER ()
    ON_WM_GETMINMAXINFO ()
    ON_WM_NCHITTEST ()
    ON_WM_SYSCOMMAND ()
    ON_WM_CONTEXTMENU ()
    ON_WM_ENDSESSION ()
    ON_WM_CLOSE ()
END_MESSAGE_MAP ()

CMainWindow::CMainWindow ()
{
    m_bAutoMenuEnable = FALSE;
```

```
    CTime time = CTime::GetCurrentTime ();
    m_nPrevSecond = time.GetSecond ();
    m_nPrevMinute = time.GetMinute ();
    m_nPrevHour = time.GetHour () % 12;

    CString strWndClass = AfxRegisterWndClass (
        CS_HREDRAW œ CS_VREDRAW,
        myApp.LoadStandardCursor (IDC_ARROW),

        (HBRUSH) (COLOR_3DFACE + 1),
        myApp.LoadIcon (IDI_APPICON)    );

    Create (strWndClass, _T ("Clock"));
}
BOOL CMainWindow::PreCreateWindow (CREATESTRUCT& cs)
{
    if (!CFrameWnd::PreCreateWindow (cs))
        return FALSE;

    cs.dwExStyle &= ~WS_EX_CLIENTEDGE;
    return TRUE;
}
int CMainWindow::OnCreate (LPCREATESTRUCT lpcs)
{
    if (CFrameWnd::OnCreate (lpcs) == -1)
        return -1;
    //
    // Set a timer to fire at 1-second intervals.
    //
    if (!SetTimer (ID_TIMER_CLOCK, 1000, NULL)) {
        MessageBox (_T ("SetTimer failed"), _T ("Error"),
            MB_ICONSTOP œ MB_OK);
        return -1;
    }
    //
    // Customize the system menu.
    //
    CMenu* pMenu = GetSystemMenu (FALSE);
    pMenu->AppendMenu (MF_SEPARATOR);
    pMenu->AppendMenu (MF_STRING, IDM_SYSMENU_FULL_WINDOW,
        _T ("Remove &Title"));
    pMenu->AppendMenu (MF_STRING, IDM_SYSMENU_STAY_ON_TOP,
        _T ("Stay on To&p"));
    return 0;
}
void CMainWindow::OnClose ()
{
    SaveWindowState ();
    KillTimer (ID_TIMER_CLOCK);
    CFrameWnd::OnClose ();
}
void CMainWindow::OnEndSession (BOOL bEnding)
{
    if (bEnding)
        SaveWindowState ();
```

```
    CFrameWnd::OnEndSession (bEnding);
}
void CMainWindow::OnGetMinMaxInfo (MINMAXINFO* pMMI)
{
    pMMI->ptMinTrackSize.x = 120;
    pMMI->ptMinTrackSize.y = 120;
}
UINT CMainWindow::OnNcHitTest (CPoint point)
{
    UINT nHitTest = CFrameWnd::OnNcHitTest (point);
    if ((nHitTest == HTCLIENT) && (::GetAsyncKeyState (MK_LBUTTON) < 0))
        nHitTest = HTCAPTION;
    return nHitTest;
}
void CMainWindow::OnSysCommand (UINT nID, LPARAM lParam)
{
    UINT nMaskedID = nID & 0xFFF0;

    if (nMaskedID == IDM_SYSMENU_FULL_WINDOW) {
        m_bFullWindow = m_bFullWindow ? 0 : 1;
        SetTitleBarState ();
        return;
    }
    else if (nMaskedID == IDM_SYSMENU_STAY_ON_TOP) {
        m_bStayOnTop = m_bStayOnTop ? 0 : 1;
        SetTopMostState ();
        return;
    }
    CFrameWnd::OnSysCommand (nID, lParam);
}
void CMainWindow::OnContextMenu (CWnd* pWnd, CPoint point)
{
    CRect rect;
    GetClientRect (&rect);
    ClientToScreen (&rect);

    if (rect.PtInRect (point)) {
        CMenu* pMenu = GetSystemMenu (FALSE);
        UpdateSystemMenu (pMenu);

        int nID = (int) pMenu->TrackPopupMenu (TPM_LEFTALIGN œ
            TPM_LEFTBUTTON œ TPM_RIGHTBUTTON œ TPM_RETURNCMD, point.x,
            point.y, this);

        if (nID > 0)
            SendMessage (WM_SYSCOMMAND, nID, 0);
        return;
    }
    CFrameWnd::OnContextMenu (pWnd, point);
}
void CMainWindow::OnTimer (UINT nTimerID)
{
    //
    // Do nothing if the window is minimized.
    //
```

```
    if (IsIconic ())
        return;
    //
    // Get the current time and do nothing if it hasn't changed.
    //
    CTime time = CTime::GetCurrentTime ();
    int nSecond = time.GetSecond ();
    int nMinute = time.GetMinute ();
    int nHour = time.GetHour () % 12;

    if ((nSecond == m_nPrevSecond) &&
        (nMinute == m_nPrevMinute) &&
        (nHour == m_nPrevHour))
        return;
    //
    // Center the origin and switch to the MM_ISOTROPIC mapping mode.
    //
    CRect rect;
    GetClientRect (&rect);

    CClientDC dc (this);
    dc.SetMapMode (MM_ISOTROPIC);
    dc.SetWindowExt (1000, 1000);
    dc.SetViewportExt (rect.Width (), -rect.Height ());
    dc.SetViewportOrg (rect.Width () / 2, rect.Height () / 2);
    //
    // If minutes have changed, erase the hour and minute hands.
    //
    COLORREF clrColor = ::GetSysColor (COLOR_3DFACE);

    if (nMinute != m_nPrevMinute) {
        DrawHand (&dc, 200, 4, (m_nPrevHour * 30) + (m_nPrevMinute / 2),
            clrColor);
        DrawHand (&dc, 400, 8, m_nPrevMinute * 6, clrColor);
        m_nPrevMinute = nMinute;
        m_nPrevHour = nHour;
    }
    //
    // If seconds have changed, erase the second hand and redraw all hands.
    //
    if (nSecond != m_nPrevSecond) {
        DrawSecondHand (&dc, 400, 8, m_nPrevSecond * 6, clrColor);
        DrawSecondHand (&dc, 400, 8, nSecond * 6, RGB (0, 0, 0));
        DrawHand (&dc, 200, 4, (nHour * 30) + (nMinute / 2),
            RGB (0, 0, 0));
        DrawHand (&dc, 400, 8, nMinute * 6, RGB (0, 0, 0));
        m_nPrevSecond = nSecond;
    }
}
void CMainWindow::OnPaint ()
{
    CRect rect;
    GetClientRect (&rect);

    CPaintDC dc (this);
```

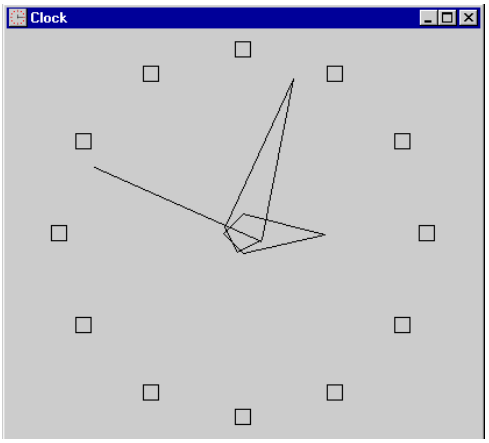
```
    dc.SetMapMode (MM_ISOTROPIC);
    dc.SetWindowExt (1000, 1000);
    dc.SetViewportExt (rect.Width (), -rect.Height ());
    dc.SetViewportOrg (rect.Width () / 2, rect.Height () / 2);

    DrawClockFace (&dc);
    DrawHand (&dc, 200, 4, (m_nPrevHour * 30) +
        (m_nPrevMinute / 2), RGB (0, 0, 0));
    DrawHand (&dc, 400, 8, m_nPrevMinute * 6, RGB (0, 0, 0));
    DrawSecondHand (&dc, 400, 8, m_nPrevSecond * 6, RGB (0, 0, 0));
}
void CMainWindow::DrawClockFace (CDC* pDC)
{
    static CPoint point[12] = {
        CPoint (   0,  450),    // 12 o'clock
        CPoint ( 225,  390),    //  1 o'clock
        CPoint ( 390,  225),    //  2 o'clock
        CPoint ( 450,    0),    //  3 o'clock
        CPoint ( 390, -225),    //  4 o'clock
        CPoint ( 225, -390),    //  5 o'clock
        CPoint (   0, -450),    //  6 o'clock
        CPoint (-225, -390),    //  7 o'clock
        CPoint (-390, -225),    //  8 o'clock
        CPoint (-450,    0),    //  9 o'clock
        CPoint (-390,  225),    // 10 o'clock
        CPoint (-225,  390),    // 11 o'clock
    };

    pDC->SelectStockObject (NULL_BRUSH);

    for (int i=0; i<12; i++)
        pDC->Rectangle (point[i].x - SQUARESIZE,
            point[i].y + SQUARESIZE, point[i].x + SQUARESIZE,
            point[i].y - SQUARESIZE);
}
void CMainWindow::DrawHand (CDC* pDC, int nLength, int nScale,
    int nDegrees, COLORREF clrColor)
{
    CPoint point[4];
    double nRadians = (double) nDegrees * 0.017453292;

    point[0].x = (int) (nLength * sin (nRadians));
    point[0].y = (int) (nLength * cos (nRadians));

    point[2].x = -point[0].x / nScale;
    point[2].y = -point[0].y / nScale;

    point[1].x = -point[2].y;
    point[1].y = point[2].x;

    point[3].x = -point[1].x;
    point[3].y = -point[1].y;

    CPen pen (PS_SOLID, 0, clrColor);
    CPen* pOldPen = pDC->SelectObject (&pen);
```

```
    pDC->MoveTo (point[0]);
    pDC->LineTo (point[1]);
    pDC->LineTo (point[2]);
    pDC->LineTo (point[3]);
    pDC->LineTo (point[0]);

    pDC->SelectObject (pOldPen);
}
void CMainWindow::DrawSecondHand (CDC* pDC, int nLength, int nScale,
    int nDegrees, COLORREF clrColor)
{
    CPoint point[2];
    double nRadians = (double) nDegrees * 0.017453292;

    point[0].x = (int) (nLength * sin (nRadians));
    point[0].y = (int) (nLength * cos (nRadians));

    point[1].x = -point[0].x / nScale;
    point[1].y = -point[0].y / nScale;

    CPen pen (PS_SOLID, 0, clrColor);
    CPen* pOldPen = pDC->SelectObject (&pen);

    pDC->MoveTo (point[0]);
    pDC->LineTo (point[1]);

    pDC->SelectObject (pOldPen);
}
void CMainWindow::SetTitleBarState ()
{
    CMenu* pMenu = GetSystemMenu (FALSE);

    if (m_bFullWindow ) {
        ModifyStyle (WS_CAPTION, 0);
        pMenu->ModifyMenu (IDM_SYSMENU_FULL_WINDOW, MF_STRING,
            IDM_SYSMENU_FULL_WINDOW, _T ("Restore &Title"));
    }
    else {
        ModifyStyle (0, WS_CAPTION);
        pMenu->ModifyMenu (IDM_SYSMENU_FULL_WINDOW, MF_STRING,
            IDM_SYSMENU_FULL_WINDOW, _T ("Remove &Title"));
    }
    SetWindowPos (NULL, 0, 0, 0, 0, SWP_NOMOVE œ SWP_NOSIZE œ
        SWP_NOZORDER œ SWP_DRAWFRAME);
}
void CMainWindow::SetTopMostState ()
{
    CMenu* pMenu = GetSystemMenu (FALSE);

    if (m_bStayOnTop) {
        SetWindowPos (&wndTopMost, 0, 0, 0, 0, SWP_NOMOVE œ SWP_NOSIZE);
        pMenu->CheckMenuItem (IDM_SYSMENU_STAY_ON_TOP, MF_CHECKED);
    }
    else {
```

```
        SetWindowPos (&wndNoTopMost, 0, 0, 0, 0, SWP_NOMOVE œ SWP_NOSIZE);
        pMenu->CheckMenuItem (IDM_SYSMENU_STAY_ON_TOP, MF_UNCHECKED);
    }
}
BOOL CMainWindow::RestoreWindowState ()
{
    CString version = _T ("Version 1.0");
    m_bFullWindow = myApp.GetProfileInt (version, _T ("FullWindow"), 0);
    SetTitleBarState ();
    m_bStayOnTop = myApp.GetProfileInt (version, _T ("StayOnTop"), 0);
    SetTopMostState ();

    WINDOWPLACEMENT wp;
    wp.length = sizeof (WINDOWPLACEMENT);
    GetWindowPlacement (&wp);

    if (((wp.flags =
            myApp.GetProfileInt (version, _T ("flags"), -1)) != -1) &&
        ((wp.showCmd =
            myApp.GetProfileInt (version, _T ("showCmd"), -1)) != -1) &&
        ((wp.rcNormalPosition.left =
            myApp.GetProfileInt (version, _T ("x1"), -1)) != -1) &&
        ((wp.rcNormalPosition.top =
            myApp.GetProfileInt (version, _T ("y1"), -1)) != -1) &&
        ((wp.rcNormalPosition.right =
            myApp.GetProfileInt (version, _T ("x2"), -1)) != -1) &&
        ((wp.rcNormalPosition.bottom =
            myApp.GetProfileInt (version, _T ("y2"), -1)) != -1)) {

        wp.rcNormalPosition.left = min (wp.rcNormalPosition.left,
            ::GetSystemMetrics (SM_CXSCREEN) -
            ::GetSystemMetrics (SM_CXICON));
        wp.rcNormalPosition.top = min (wp.rcNormalPosition.top,
            ::GetSystemMetrics (SM_CYSCREEN) -
            ::GetSystemMetrics (SM_CYICON));
        SetWindowPlacement (&wp);
        return TRUE;
    }
    return FALSE;
}
void CMainWindow::SaveWindowState ()
{
    CString version = _T ("Version 1.0");
    myApp.WriteProfileInt (version, _T ("FullWindow"), m_bFullWindow);
    myApp.WriteProfileInt (version, _T ("StayOnTop"), m_bStayOnTop);

    WINDOWPLACEMENT wp;
    wp.length = sizeof (WINDOWPLACEMENT);
    GetWindowPlacement (&wp);

    myApp.WriteProfileInt (version, _T ("flags"), wp.flags);
    myApp.WriteProfileInt (version, _T ("showCmd"), wp.showCmd);
    myApp.WriteProfileInt (version, _T ("x1"), wp.rcNormalPosition.left);
    myApp.WriteProfileInt (version, _T ("y1"), wp.rcNormalPosition.top);
    myApp.WriteProfileInt (version, _T ("x2"), wp.rcNormalPosition.right);
```

```
    myApp.WriteProfileInt (version, _T ("y2"), wp.rcNormalPosition.bottom);
}
void CMainWindow::UpdateSystemMenu (CMenu* pMenu)
{
    static UINT nState[2][5] = {
        { MFS_GRAYED,  MFS_ENABLED, MFS_ENABLED,
          MFS_ENABLED, MFS_DEFAULT },
        { MFS_DEFAULT, MFS_GRAYED,  MFS_GRAYED,
          MFS_ENABLED, MFS_GRAYED  }
    };
    if (IsIconic ()) // Shouldn't happen, but let's be safe
        return;
    int i = 0;
    if (IsZoomed ())
        i = 1;
    CString strMenuText;
    pMenu->GetMenuString (SC_RESTORE, strMenuText, MF_BYCOMMAND);
    pMenu->ModifyMenu (SC_RESTORE, MF_STRING œ nState[i][0], SC_RESTORE,
        strMenuText);

    pMenu->GetMenuString (SC_MOVE, strMenuText, MF_BYCOMMAND);
    pMenu->ModifyMenu (SC_MOVE, MF_STRING œ nState[i][1], SC_MOVE,
        strMenuText);

    pMenu->GetMenuString (SC_SIZE, strMenuText, MF_BYCOMMAND);
    pMenu->ModifyMenu (SC_SIZE, MF_STRING œ nState[i][2], SC_SIZE,
        strMenuText);

    pMenu->GetMenuString (SC_MINIMIZE, strMenuText, MF_BYCOMMAND);
    pMenu->ModifyMenu (SC_MINIMIZE, MF_STRING œ nState[i][3], SC_MINIMIZE,
        strMenuText);

    pMenu->GetMenuString (SC_MAXIMIZE, strMenuText, MF_BYCOMMAND);
    pMenu->ModifyMenu (SC_MAXIMIZE, MF_STRING œ nState[i][4], SC_MAXIMIZE,
        strMenuText);

    SetMenuDefaultItem (pMenu->m_hMenu, i ? SC_RESTORE :
        SC_MAXIMIZE, FALSE);
}
```

Màn hình kết quả như sau:



3.2 THREADS

3.2.1 Vấn đề quan tâm

- Hiểu và sử dụng được kỹ thuật lập trình về thread.

3.2.2 Giới thiệu

Thread (*tiểu trình/luồng*) là một phần lệnh của chương trình ứng dụng thực thi với mục đích độc lập và không phân chia nhỏ hơn nữa.

Một quá trình trong ứng dụng dạng Win-32 bit được bắt đầu như đơn luồng (single thread) nhưng cũng có thể thêm nhiều luồng khác để trở thành đa luồng (multi threads)

Việc xây dựng ứng dụng dạng multi threads không đơn giản và không dành cho tất cả các dạng ứng dụng mà chỉ dàng cho các ứng dụng dạng lập trình/xử lý song song (*paralell processing*)

MFC hỗ trợ việc lập trình với thread bởi lớp **CWinThread**

3.2.3 Threads

Để tạo một thread, gọi hàm **AfxBeginThread**

Ví dụ 1:

```
CWinThread* pThread = AfxBeginThread(ThreadFunc, &threadInfo);
Và
UINT ThreadFunc(LPVOID pParam)
{
  UINT nIterations =(UINT) pParam;
  for(UINT i=0; i<nIterations; i++);
  return 0;
}
```

Dạng tổng quát của **AfxBeginThread** là:

```
CWinThread* AfxBeginThread(AFX_THREADPROC pfnThreadProc,
 LPVOID pParam, int nPriority = THREAD_PRIORITY_NORMAL,
 UINT nStackSize = 0, DWORD dwCreateFlags = 0,
 LPSECURITY_ATTRIBUTES lpSecurityAttrs = NULL)
```

Có thể dùng hàm **SetThreadPriority** để đặt độ ưu tiên cho mỗi thread

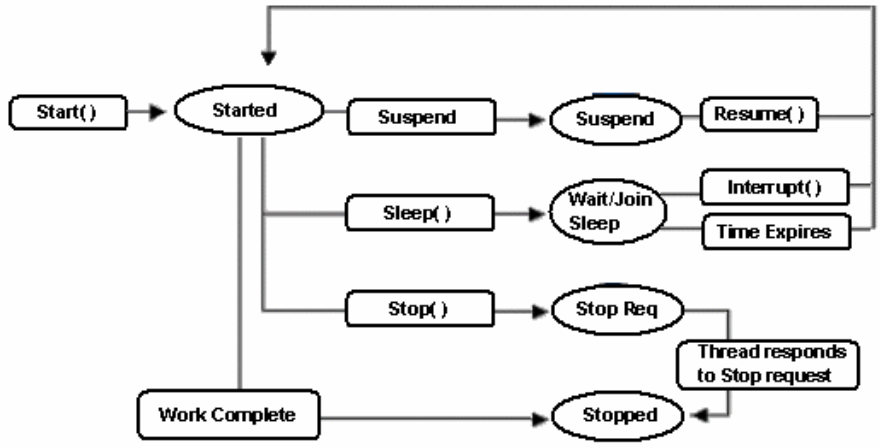
Có thể tạo 1 class liên kết với mỗi thread

Ví dụ 2:

```
// The CUIThread class
class CUIThread : public CWinThread
{
  DECLARE_DYNCREATE(CUIThread)

public:
  virtual BOOL InitInstance();
};
IMPLEMENT_DYNCREATE(CUIThread, CWinThread)

BOOL CUIThread::InitInstance()
{
  m_pMainWnd = new CMainWindow;
  m_pMainWnd->ShowWindow(SW_SHOW);
  m_pMainWnd->UpdateWindow();
  return TRUE;
}
// The CMainWindow class
class CMainWindow : public CFrameWnd
{
public:
```

```
  CMainWindow();

protected:
  afx_msg void OnLButtonDown(UINT, CPoint);
  DECLARE_MESSAGE_MAP()
};
BEGIN_MESSAGE_MAP(CMainWindow, CFrameWnd)
  ON_WM_LBUTTONDOWN()
END_MESSAGE_MAP()

CMainWindow::CMainWindow()
{
  Create(NULL, _T("UI Thread Window"));
}
void CMainWindow::OnLButtonDown(UINT nFlags, CPoint point)
{
  PostMessage(WM_CLOSE, 0, 0);
}
```

Mỗi thread có thể chuyển đổi qua nhiều trạng thái theo như sơ đồ sau:



Có thể tạm ngừng (*Suspend*) và khôi phục (*Resume*) việc hoạt động của thread bởi các lệnh **SuspendThread** và **ResumeThread**

Có thể đặt thread vào trạng thái tạm ngừng trong môt thời gian hạn định bằng việc dùng lệnh **Sleep**

Kết thúc một thread bằng việc dùng lệnh **AfxEndThread**

Đặt độ ưu tiên cho thread bằng SetPriorityClass, với các giá trị có thể chọn như sau:

Độ ưu tiên	Mô tả
IDLE_PRIORITY_CLASS	The process runs only when the system is idle—for example, when no other thread is waiting for a given CPU.
NORMAL_PRIORITY_CLASS	The default process priority class. The process has no special scheduling needs.
HIGH_PRIORITY_CLASS	The process receives priority over IDLE_PRIORITY_CLASS and NORMAL_PRIORITY_CLASS processes.
REALTIME_PRIORITY_CLASS	The process must have the highest possible priority, and its threads should preempt even threads belonging to HIGH_PRIORITY_CLASS processes.

Và:

Mức ưu tiên	Mô tả
THREAD_PRIORITY_IDLE	The thread's base priority level is 1 if the process's priority class is HIGH_PRIORITY_CLASS or lower, or 16 if the process's priority class is REALTIME_PRIORITY_CLASS.
THREAD_PRIORITY_LOWEST	The thread's base priority level is equal to the

	process's priority class minus 2.
THREAD_PRIORITY_BELOW_NORMAL	The thread's base priority level is equal to the process's priority class minus 1.
THREAD_PRIORITY_NORMAL	The default thread priority value. The thread's base priority level is equal to the process's priority class.
THREAD_PRIORITY_ABOVE_NORMAL	The thread's base priority level is equal to the process's priority class plus 1.
THREAD_PRIORITY_HIGHEST	The thread's base priority level is equal to the process's priority class plus 2.
THREAD_PRIORITY_TIME_CRITICAL	The thread's base priority level is 15 if the process's priority class is HIGH_PRIORITY_CLASS or lower, or 31 if the process's priority class is REALTIME_PRIORITY_CLASS.

Ví dụ 3:

Trong file MyDlg.h

```
#define WM_USER_THREAD_FINISHED WM_USER+0x100

UINT ThreadFunc(LPVOID pParam);
int MyFunc(int nMax);

typedef struct tagTHREADPARMS {
  int nMax;
  HWND hWnd;
} THREADPARMS;

class CMyDlg : public CDialog
{
…
// Implementation
protected:
  HICON m_hIcon;

  // Generated message map functions
  //{{AFX_MSG(CMyDlg)
  virtual BOOL OnInitDialog();
  afx_msg void OnPaint();
  afx_msg HCURSOR OnQueryDragIcon();
  afx_msg void OnStart();
  //}}AFX_MSG
  afx_msg LONG OnThreadFinished(WPARAM wParam, LPARAM lParam);
  DECLARE_MESSAGE_MAP()
…
};
```

Trong file MyDlg.cpp

```
BEGIN_MESSAGE_MAP(CMyDlg, CDialog)
  //{{AFX_MSG_MAP(CMyDlg)
  ON_BN_CLICKED(IDC_START, OnStart)
  //}}AFX_MSG_MAP
  ON_MESSAGE(WM_USER_THREAD_FINISHED, OnThreadFinished)
END_MESSAGE_MAP()
…
void CMyDlg::OnStart()
{
  int nMax = GetDlgItemInt(IDC_MAX);
```

```
  if(nMax < 10) {
    MessageBox(_T("The number you enter must be 10 or higher"));
    GetDlgItem(IDC_MAX)->SetFocus();
    return;
  }

  SetDlgItemText(IDC_RESULT, _T(""));
  GetDlgItem(IDC_START)->EnableWindow(FALSE);

  THREADPARMS* ptp = new THREADPARMS;
  ptp->nMax = nMax;
  ptp->hWnd = m_hWnd;
  AfxBeginThread(ThreadFunc, ptp);
}
LONG CMyDlg::OnThreadFinished(WPARAM wParam, LPARAM lParam)
{
  SetDlgItemInt(IDC_RESULT,(int) wParam);
  GetDlgItem(IDC_START)->EnableWindow(TRUE);
  return 0;
}
////////////////////////////////////////////////////////////////////////////
// Global functions
UINT ThreadFunc(LPVOID pParam)
{
  THREADPARMS* ptp =(THREADPARMS*) pParam;
  int nMax = ptp->nMax;
  HWND hWnd = ptp->hWnd;
  delete ptp;

  int nCount = MyFunction(nMax);
  ::PostMessage(hWnd, WM_USER_THREAD_FINISHED,(WPARAM) nCount, 0);
  return 0;
}
int MyFunction(int nMax)
{
  PBYTE pBuffer = new BYTE[nMax + 1];
  ::FillMemory(pBuffer, nMax + 1, 1);

  int nLimit = 2;
  while(nLimit * nLimit < nMax)
    nLimit++;

  for(int i=2; i<=nLimit; i++) {
    if(pBuffer[i]) {
      for(int k=i + i; k<=nMax; k+=i)
        pBuffer[k] = 0;
    }
  }

  int nCount = 0;
  for(i=2; i<=nMax; i++)
    if(pBuffer[i])
      nCount++;

  delete[] pBuffer;
```

```
  return nCount;
}
```

Ví dụ tổng hợp:

Sieve.h

```
// Sieve.h : main header file for the SIEVE application
//
#if !defined(AFX_SIEVE_H__6DF40C9B_7EA1_11D1_8E53_E4D9F9C00000__INCLUDED_)
#define AFX_SIEVE_H__6DF40C9B_7EA1_11D1_8E53_E4D9F9C00000__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
/////////////////////////////////////////////////////////////////////////////
// CSieveApp:
// See Sieve.cpp for the implementation of this class
//
class CSieveApp : public CWinApp
{
public:
    CSieveApp();

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

// Implementation

    //{{AFX_MSG(CSieveApp)
       // 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 Developer Studio will insert additional declarations
// immediately before the previous line.

#endif
// !defined(AFX_SIEVE_H__6DF40C9B_7EA1_11D1_8E53_E4D9F9C00000__INCLUDED_)
```

Sieve.cpp

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

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CSieveApp
BEGIN_MESSAGE_MAP(CSieveApp, CWinApp)
    //{{AFX_MSG_MAP(CSieveApp)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CSieveApp construction
CSieveApp::CSieveApp()
{
    // TODO: add construction code here,
    // Place all significant initialization in InitInstance
}
/////////////////////////////////////////////////////////////////////////////
// The one and only CSieveApp object
CSieveApp theApp;
/////////////////////////////////////////////////////////////////////////////
// CSieveApp initialization
BOOL CSieveApp::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.

    CSieveDlg dlg;
    m_pMainWnd = &dlg;
    int nResponse = dlg.DoModal();
    if (nResponse == IDOK)
    {
        // TODO: Place code here to handle when the dialog is
        //  dismissed with OK
    }
    else if (nResponse == IDCANCEL)
    {
        // TODO: Place code here to handle when the dialog is
        //  dismissed with Cancel
    }

    // Since the dialog has been closed, return FALSE so that we exit the
    //  application, rather than start the application's message pump.
    return FALSE;
}
```

SieveDlg.h

```
// SieveDlg.h : header file
//
```

```
#if !defined(
    AFX_SIEVEDLG_H__6DF40C9D_7EA1_11D1_8E53_E4D9F9C00000__INCLUDED_)
#define AFX_SIEVEDLG_H__6DF40C9D_7EA1_11D1_8E53_E4D9F9C00000__INCLUDED_

#if _MSC_VER >= 1000
#pragma once
#endif // _MSC_VER >= 1000
#define WM_USER_THREAD_FINISHED WM_USER+0x100
UINT ThreadFunc (LPVOID pParam);
int Sieve (int nMax);
typedef struct tagTHREADPARMS {
    int nMax;
    HWND hWnd;
} THREADPARMS;
/////////////////////////////////////////////////////////////////////////////
// CSieveDlg dialog
class CSieveDlg : public CDialog
{
// Construction
public:
    CSieveDlg(CWnd* pParent = NULL);    // standard constructor
// Dialog Data
    //{{AFX_DATA(CSieveDlg)
    enum { IDD = IDD_SIEVE_DIALOG };
        // NOTE: the ClassWizard will add data members here
    //}}AFX_DATA

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

    // Generated message map functions
    //{{AFX_MSG(CSieveDlg)
    virtual BOOL OnInitDialog();
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    afx_msg void OnStart();
    //}}AFX_MSG
    afx_msg LONG OnThreadFinished (WPARAM wParam, LPARAM lParam);
    DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Developer Studio will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//     AFX_SIEVEDLG_H__6DF40C9D_7EA1_11D1_8E53_E4D9F9C00000__INCLUDED_)
```

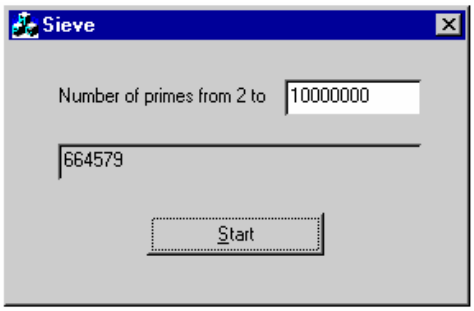
SieveDlg.cpp

```
// SieveDlg.cpp : implementation file
//
```

```
#include "stdafx.h"
#include "Sieve.h"
#include "SieveDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CSieveDlg dialog
CSieveDlg::CSieveDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CSieveDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CSieveDlg)
        // NOTE: the ClassWizard will add member initialization here
    //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent
    // DestroyIcon in Win32
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
void CSieveDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CSieveDlg)
        // NOTE: the ClassWizard will add DDX and DDV calls here
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CSieveDlg, CDialog)
    //{{AFX_MSG_MAP(CSieveDlg)
    ON_BN_CLICKED(IDC_START, OnStart)
    //}}AFX_MSG_MAP
    ON_MESSAGE (WM_USER_THREAD_FINISHED, OnThreadFinished)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CSieveDlg message handlers
BOOL CSieveDlg::OnInitDialog()
{
    CDialog::OnInitDialog();
    SetIcon(m_hIcon, TRUE);
    SetIcon(m_hIcon, FALSE);
    return TRUE;
}
void CSieveDlg::OnStart()
{
    int nMax = GetDlgItemInt (IDC_MAX);
    if (nMax < 10) {
        MessageBox (_T ("The number you enter must be 10 or higher"));
        GetDlgItem (IDC_MAX)->SetFocus ();
        return;
    }

    SetDlgItemText (IDC_RESULT, _T (""));
    GetDlgItem (IDC_START)->EnableWindow (FALSE);
```

```
    THREADPARMS* ptp = new THREADPARMS;
    ptp->nMax = nMax;
    ptp->hWnd = m_hWnd;
    AfxBeginThread (ThreadFunc, ptp);
}
LONG CSieveDlg::OnThreadFinished (WPARAM wParam, LPARAM lParam)
{
    SetDlgItemInt (IDC_RESULT, (int) wParam);
    GetDlgItem (IDC_START)->EnableWindow (TRUE);
    return 0;
}
////////////////////////////////////////////////////////////////////////
// Global functions
UINT ThreadFunc (LPVOID pParam)
{
    THREADPARMS* ptp = (THREADPARMS*) pParam;
    int nMax = ptp->nMax;
    HWND hWnd = ptp->hWnd;
    delete ptp;

    int nCount = Sieve (nMax);
    ::PostMessage (hWnd, WM_USER_THREAD_FINISHED, (WPARAM) nCount, 0);
    return 0;
}
int Sieve(int nMax)
{
    PBYTE pBuffer = new BYTE[nMax + 1];
    ::FillMemory (pBuffer, nMax + 1, 1);

    int nLimit = 2;
    while (nLimit * nLimit < nMax)
        nLimit++;

    for (int i=2; i<=nLimit; i++) {
        if (pBuffer[i]) {
            for (int k=i + i; k<=nMax; k+=i)
                pBuffer[k] = 0;
        }
    }
    int nCount = 0;
    for (i=2; i<=nMax; i++)
        if (pBuffer[i])
            nCount++;

    delete[] pBuffer;
    return nCount;
}
```

Màn hình kết quả như sau:



3.2.4 Đồng bộ các threads(Thread Synchronization)

(*Xem Chương 17, phần Thread Synchronization, sách Programming Windows with MFC, 2nd edition*)

Windows cung cấp 4 kiểu của đối tượng đồng bộ mà có thể dùng để đồng bộ các hành động của các thread hoạt động song hành như sau:

- Critical sections
- Mutexes
- Events
- Semaphores

Việc đồng bộ các thread bởi các đối tượng này thông qua việc xử lý hiện thực tác vụ **Lock** và **Unlock** để điều phối quyền truy xuất dử liệu/bộ nhớ đồng thời hay không.

Ngoài ra MFC còn cung cấp lớp ***CSingleLock*** and ***CMultiLock*** để trợ gúp việc xử lý hiện thực tác vụ Lock và Unlock của các đối tượng trên một cách đơn giản hơn.

```
CCriticalSection g_cs;

CSingleLock lock (&g_cs); // Wrap it in a CSingleLock.
lock.Lock ();            // Lock the critical section.
```

hay

```
CMutex g_mutex;
CEvent g_event[2];
CSyncObject* g_pObjects[3] = { &g_mutex, &g_event[0], &g_event[1] };

// Block until all three objects become signaled.
CMultiLock multiLock (g_pObjects, 3);
multiLock.Lock ();

// Block until one of the three objects becomes signaled.
CMultiLock multiLock (g_pObjects, 3);
multiLock.Lock (INFINITE, FALSE);
```

Ví dụ tổng hợp:

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_MAINFRM_H__9D77AEE8_AA14_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_MAINFRM_H__9D77AEE8_AA14_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

protected:  // control bar embedded members
    CSpecialStatusBar m_wndStatusBar;
// Generated message map functions protected:
    int m_nPercentDone;
    //{{AFX_MSG(CMainFrame)
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    afx_msg BOOL OnQueryNewPalette();
    afx_msg void OnPaletteChanged(CWnd* pFocusWnd);
    //}}AFX_MSG
    afx_msg LRESULT OnUpdateImageStats (WPARAM wParam, LPARAM lParam);
    afx_msg LRESULT OnThreadUpdate (WPARAM wParam, LPARAM lParam);
    afx_msg LRESULT OnThreadFinished (WPARAM wParam, LPARAM lParam);
    afx_msg LRESULT OnThreadAborted (WPARAM wParam, LPARAM lParam);
    DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(AFX_MAINFRM_H__9D77AEE8_AA14_11D2_8E53_006008A82731__INCLUDED_)
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//
#include "stdafx.h"
#include "ImageEdit.h"
#include "ImageEditDoc.h"
#include "SpecialStatusBar.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)

    ON_WM_CREATE()
    ON_WM_QUERYNEWPALETTE()
    ON_WM_PALETTECHANGED()
    //}}AFX_MSG_MAP
    ON_MESSAGE (WM_USER_UPDATE_STATS, OnUpdateImageStats)
    ON_MESSAGE (WM_USER_THREAD_UPDATE, OnThreadUpdate)
    ON_MESSAGE (WM_USER_THREAD_FINISHED, OnThreadFinished)
    ON_MESSAGE (WM_USER_THREAD_ABORTED, OnThreadAborted)
END_MESSAGE_MAP()
static UINT indicators[] =
{
    ID_SEPARATOR,
    ID_SEPARATOR,
    ID_SEPARATOR
};
/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction
CMainFrame::CMainFrame()
{
    m_nPercentDone = -1;
}
CMainFrame::~CMainFrame()
{
}
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
        return -1;

    if (!m_wndStatusBar.Create(this))
    {
        TRACE0("Failed to create status bar\n");
        return -1;      // fail to create
    }
    return 0;
}
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::OnQueryNewPalette()
{
    CDocument* pDoc = GetActiveDocument ();
    if (pDoc != NULL)
        GetActiveDocument ()->UpdateAllViews (NULL);
    return TRUE;
}
void CMainFrame::OnPaletteChanged(CWnd* pFocusWnd)
{
    if (pFocusWnd != this) {
        CDocument* pDoc = GetActiveDocument ();
        if (pDoc != NULL)
            GetActiveDocument ()->UpdateAllViews (NULL);
    }
}
LRESULT CMainFrame::OnUpdateImageStats (WPARAM wParam, LPARAM lParam)
{
    m_wndStatusBar.SetImageStats ((LPCTSTR) lParam);
    return 0;
}
LRESULT CMainFrame::OnThreadUpdate (WPARAM wParam, LPARAM lParam)
{
    int nPercentDone = ((int) wParam * 100) / (int) lParam;
    if (nPercentDone != m_nPercentDone) {
        m_wndStatusBar.SetProgress (nPercentDone);
        m_nPercentDone = nPercentDone;
    }
    return 0;
}
LRESULT CMainFrame::OnThreadFinished (WPARAM wParam, LPARAM lParam)
{
    CImageEditDoc* pDoc = (CImageEditDoc*) GetActiveDocument ();
    if (pDoc != NULL) {
        pDoc->ThreadFinished ();
        m_wndStatusBar.SetProgress (0);
        m_nPercentDone = -1;
    }
    return 0;
}
LRESULT CMainFrame::OnThreadAborted (WPARAM wParam, LPARAM lParam)
{
    CImageEditDoc* pDoc = (CImageEditDoc*) GetActiveDocument ();
    if (pDoc != NULL) {
        pDoc->ThreadAborted ();
        m_wndStatusBar.SetProgress (0);
        m_nPercentDone = -1;
    }
    return 0;
}
```

ImageEditDoc.h

```
// ImageEditDoc.h : interface of the CImageEditDoc class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_IMAGEEDITDOC_H__9D77AEEA_AA14_11D2_8E53_006008A82731__INCLUDED_)
#define AFX_IMAGEEDITDOC_H__9D77AEEA_AA14_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
UINT ThreadFunc (LPVOID pParam);
LOGPALETTE* CreateGrayScale ();
class CImageEditDoc : public CDocument
{
protected: // create from serialization only
    CImageEditDoc();
    DECLARE_DYNCREATE(CImageEditDoc)
// Attributes
public:

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CImageEditDoc)
    public:
    virtual BOOL OnNewDocument();
    virtual BOOL OnOpenDocument(LPCTSTR lpszPathName);
    virtual void DeleteContents();
    //}}AFX_VIRTUAL

// Implementation
public:
    void ThreadAborted();
    void ThreadFinished();
    CPalette* GetPalette();
    CBitmap* GetBitmap();
    virtual ~CImageEditDoc();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Generated message map functions
protected:
    CCriticalSection m_cs;
    CEvent m_event;
    HANDLE m_hThread;
    BOOL m_bWorking;
    CPalette m_palette;
    CBitmap m_bitmap;
    //{{AFX_MSG(CImageEditDoc)
    afx_msg void OnGrayScale();
    afx_msg void OnUpdateGrayScale(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_IMAGEEDITDOC_H__9D77AEEA_AA14_11D2_8E53_006008A82731__INCLUDED_)
```

ImageEditDoc.cpp

```
// ImageEditDoc.cpp : implementation of the CImageEditDoc class
//
#include "stdafx.h"
#include "ImageEdit.h"
#include "ImageEditDoc.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CImageEditDoc
IMPLEMENT_DYNCREATE(CImageEditDoc, CDocument)
BEGIN_MESSAGE_MAP(CImageEditDoc, CDocument)
    //{{AFX_MSG_MAP(CImageEditDoc)
    ON_COMMAND(ID_EFFECTS_GRAY_SCALE, OnGrayScale)
    ON_UPDATE_COMMAND_UI(ID_EFFECTS_GRAY_SCALE, OnUpdateGrayScale)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CImageEditDoc construction/destruction
CImageEditDoc::CImageEditDoc() :
    m_event (FALSE, TRUE) // Manual-reset event, initially unowned
{
    m_hThread = NULL;
    m_bWorking = FALSE;
}
CImageEditDoc::~CImageEditDoc()
{
}
BOOL CImageEditDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;
    return TRUE;
}
/////////////////////////////////////////////////////////////////////////////
// CImageEditDoc diagnostics
#ifdef _DEBUG
void CImageEditDoc::AssertValid() const
{
    CDocument::AssertValid();
}
```

```
void CImageEditDoc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG
////////////////////////////////////////////////////////////////////////
// CImageEditDoc commands
BOOL CImageEditDoc::OnOpenDocument(LPCTSTR lpszPathName)
{
    //
    // Return now if an image is being processed.
    //
    if (m_bWorking) {
        AfxMessageBox (_T ("You can't open an image while another is " \
            "being converted"));
        return FALSE;
    }
    //
    // Let the base class do its thing.
    //
    if (!CDocument::OnOpenDocument (lpszPathName))
        return FALSE;
    //
    // Open the file and create a DIB section from its contents.
    //
    HBITMAP hBitmap = (HBITMAP) ::LoadImage (NULL, lpszPathName,
        IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE œ LR_CREATEDIBSECTION);

    if (hBitmap == NULL) {
        CString string;
        string.Format (_T ("%s does not contain a DIB"), lpszPathName);
        AfxMessageBox (string);
        return FALSE;
    }
    m_bitmap.Attach (hBitmap);
    //
    // Return now if this device doesn't support palettes.
    //
    CClientDC dc (NULL);
    if ((dc.GetDeviceCaps (RASTERCAPS) & RC_PALETTE) == 0)
        return TRUE;
    //
    // Create a palette to go with the DIB section.
    //
    if ((HBITMAP) m_bitmap != NULL) {
        DIBSECTION ds;
        m_bitmap.GetObject (sizeof (DIBSECTION), &ds);

        int nColors;
        if (ds.dsBmih.biClrUsed != 0)
            nColors = ds.dsBmih.biClrUsed;
        else
            nColors = 1 << ds.dsBmih.biBitCount;
        //
        // Create a halftone palette if the DIB section contains more
```

```
        // than 256 colors.
        //
        if (nColors > 256)
            m_palette.CreateHalftonePalette (&dc);
        //
        // Create a custom palette from the DIB section's color table
        // if the number of colors is 256 or less.
        //
        else {
            RGBQUAD* pRGB = new RGBQUAD[nColors];

            CDC memDC;
            memDC.CreateCompatibleDC (&dc);
            CBitmap* pOldBitmap = memDC.SelectObject (&m_bitmap);
            ::GetDIBColorTable ((HDC) memDC, 0, nColors, pRGB);
            memDC.SelectObject (pOldBitmap);

            UINT nSize = sizeof (LOGPALETTE) +
                (sizeof (PALETTEENTRY) * (nColors - 1));
            LOGPALETTE* pLP = (LOGPALETTE*) new BYTE[nSize];

            pLP->palVersion = 0x300;
            pLP->palNumEntries = nColors;

            for (int i=0; i<nColors; i++) {
                pLP->palPalEntry[i].peRed = pRGB[i].rgbRed;
                pLP->palPalEntry[i].peGreen = pRGB[i].rgbGreen;
                pLP->palPalEntry[i].peBlue = pRGB[i].rgbBlue;
                pLP->palPalEntry[i].peFlags = 0;
            }

            m_palette.CreatePalette (pLP);
            delete[] pLP;
            delete[] pRGB;
        }
    }
    return TRUE;
}
void CImageEditDoc::DeleteContents()
{
    if ((HBITMAP) m_bitmap != NULL)
        m_bitmap.DeleteObject ();

    if ((HPALETTE) m_palette != NULL)
        m_palette.DeleteObject ();
    CDocument::DeleteContents();
}
CBitmap* CImageEditDoc::GetBitmap()
{
    return ((HBITMAP) m_bitmap == NULL) ? NULL : &m_bitmap;
}
CPalette* CImageEditDoc::GetPalette()
{
    return ((HPALETTE) m_palette == NULL) ? NULL : &m_palette;
}
```

```
void CImageEditDoc::ThreadFinished()
{
    ASSERT (m_hThread != NULL);
    ::WaitForSingleObject (m_hThread, INFINITE);
    ::CloseHandle (m_hThread);
    m_hThread = NULL;
    m_bWorking = FALSE;
    //
    // Replace the current palette with a gray scale palette.
    //
    if ((HPALETTE) m_palette != NULL) {
        m_palette.DeleteObject ();
        LOGPALETTE* pLP = CreateGrayScale ();
        m_palette.CreatePalette (pLP);
        delete[] pLP;
    }
    //
    // Tell the view to repaint.
    //
    UpdateAllViews (NULL);
}
void CImageEditDoc::ThreadAborted()
{
    ASSERT (m_hThread != NULL);
    ::WaitForSingleObject (m_hThread, INFINITE);
    ::CloseHandle (m_hThread);
    m_hThread = NULL;
    m_bWorking = FALSE;
}
void CImageEditDoc::OnGrayScale()
{
    if (!m_bWorking) {
        m_bWorking = TRUE;
        m_event.ResetEvent ();
        //
        // Package data to pass to the image processing thread.
        //
        THREADPARMS* ptp = new THREADPARMS;
        ptp->pWnd = AfxGetMainWnd ();
        ptp->pBitmap = &m_bitmap;
        ptp->pPalette = &m_palette;
        ptp->pCriticalSection = &m_cs;
        ptp->pEvent = &m_event;
        //
        // Start the image processing thread and duplicate its handle.
        //
        CWinThread* pThread = AfxBeginThread (ThreadFunc, ptp,
            THREAD_PRIORITY_NORMAL, 0, CREATE_SUSPENDED);

        ::DuplicateHandle (GetCurrentProcess (),
            pThread->m_hThread, GetCurrentProcess (), &m_hThread,
            0, FALSE, DUPLICATE_SAME_ACCESS);

        pThread->ResumeThread ();
    }
```

```
    else
        //
        // Kill the image processing thread.
        //
        m_event.SetEvent ();
}
void CImageEditDoc::OnUpdateGrayScale(CCmdUI* pCmdUI)
{
    if (m_bWorking) {
        pCmdUI->SetText (_T ("Stop &Gray Scale Conversion"));
        pCmdUI->Enable ();
    }
    else {
        pCmdUI->SetText (_T ("Convert to &Gray Scale"));
        pCmdUI->Enable ((HBITMAP) m_bitmap != NULL);
    }
}
////////////////////////////////////////////////////////////////////////
// Thread function and other globals
UINT ThreadFunc (LPVOID pParam)
{
    THREADPARMS* ptp = (THREADPARMS*) pParam;
    CWnd* pWnd = ptp->pWnd;
    CBitmap* pBitmap = ptp->pBitmap;
    CPalette* pPalette = ptp->pPalette;
    CCriticalSection* pCriticalSection = ptp->pCriticalSection;
    CEvent* pKillEvent = ptp->pEvent;
    delete ptp;

    DIBSECTION ds;
    pBitmap->GetObject (sizeof (DIBSECTION), &ds);
    int nWidth = ds.dsBm.bmWidth;
    int nHeight = ds.dsBm.bmHeight;
    //
    // Initialize one memory DC (memDC2) to hold a color copy of the
    // image and another memory DC (memDC1) to hold a gray scale copy.
    //
    CClientDC dc (pWnd);
    CBitmap bitmap1, bitmap2;
    bitmap1.CreateCompatibleBitmap (&dc, nWidth, nHeight);
    bitmap2.CreateCompatibleBitmap (&dc, nWidth, nHeight);

    CDC memDC1, memDC2;
    memDC1.CreateCompatibleDC (&dc);
    memDC2.CreateCompatibleDC (&dc);
    CBitmap* pOldBitmap1 = memDC1.SelectObject (&bitmap1);
    CBitmap* pOldBitmap2 = memDC2.SelectObject (&bitmap2);

    CPalette* pOldPalette1 = NULL;
    CPalette* pOldPalette2 = NULL;
    CPalette grayPalette;

    if (pPalette->m_hObject != NULL) {
        LOGPALETTE* pLP = CreateGrayScale ();
        grayPalette.CreatePalette (pLP);
```

```
        delete[] pLP;

        pOldPalette1 = memDC1.SelectPalette (&grayPalette, FALSE);
        pOldPalette2 = memDC2.SelectPalette (pPalette, FALSE);
        memDC1.RealizePalette ();
        memDC2.RealizePalette ();
    }
    //
    // Copy the bitmap to memDC2.
    //
    CDC memDC3;
    memDC3.CreateCompatibleDC (&dc);
    pCriticalSection->Lock ();
    CBitmap* pOldBitmap3 = memDC3.SelectObject (pBitmap);
    memDC2.BitBlt (0, 0, nWidth, nHeight, &memDC3, 0, 0, SRCCOPY);
    memDC3.SelectObject (pOldBitmap3);
    pCriticalSection->Unlock ();
    //
    // Convert the colors in memDC2 to shades of gray in memDC1.
    //
    int x, y;
    COLORREF crColor;
    BYTE grayLevel;

    for (y=0; y<nHeight; y++) {
        for (x=0; x<nWidth; x++) {
            crColor = memDC2.GetPixel (x, y);
            grayLevel = (BYTE)
                (((((UINT) GetRValue (crColor)) * 30) +
                (((UINT) GetGValue (crColor)) * 59) +
                (((UINT) GetBValue (crColor)) * 11)) / 100);
            memDC1.SetPixel (x, y,
                PALETTERGB (grayLevel, grayLevel, grayLevel));
        }
        //
        // Kill the thread if the pKillEvent event is signaled.
        //
        if (::WaitForSingleObject (pKillEvent->m_hObject, 0) ==
            WAIT_OBJECT_0) {
            memDC1.SelectObject (pOldBitmap1);
            memDC2.SelectObject (pOldBitmap2);

            if (pPalette->m_hObject != NULL) {
                memDC1.SelectPalette (pOldPalette1, FALSE);
                memDC2.SelectPalette (pOldPalette2, FALSE);
            }
            pWnd->PostMessage (WM_USER_THREAD_ABORTED, y + 1, 0);
            return (UINT) -1;
        }
        pWnd->SendMessage (WM_USER_THREAD_UPDATE, y + 1, nHeight);
    }
    //
    // Copy the gray scale image over the original bitmap.
    //
    CPalette* pOldPalette3 = NULL;
```

```
    if (pPalette->m_hObject != NULL) {
        pOldPalette3 = memDC3.SelectPalette (&grayPalette, FALSE);
        memDC3.RealizePalette ();
    }
    pCriticalSection->Lock ();
    pOldBitmap3 = memDC3.SelectObject (pBitmap);
    memDC3.BitBlt (0, 0, nWidth, nHeight, &memDC1, 0, 0, SRCCOPY);
    memDC3.SelectObject (pOldBitmap3);
    pCriticalSection->Unlock ();
    //
    // Clean up the memory DCs.
    //
    memDC1.SelectObject (pOldBitmap1);
    memDC2.SelectObject (pOldBitmap2);

    if (pPalette->m_hObject != NULL) {
        memDC1.SelectPalette (pOldPalette1, FALSE);
        memDC2.SelectPalette (pOldPalette2, FALSE);
        memDC3.SelectPalette (pOldPalette3, FALSE);
    }
    //
    // Tell the frame window we're done.
    //
    pWnd->PostMessage (WM_USER_THREAD_FINISHED, 0, 0);
    return 0;
}
LOGPALETTE* CreateGrayScale ()
{
    UINT nSize = sizeof (LOGPALETTE) + (sizeof (PALETTEENTRY) * 63);
    LOGPALETTE* pLP = (LOGPALETTE*) new BYTE[nSize];

    pLP->palVersion = 0x300;
    pLP->palNumEntries = 64;

    for (int i=0; i<64; i++) {
        pLP->palPalEntry[i].peRed = i * 4;
        pLP->palPalEntry[i].peGreen = i * 4;
        pLP->palPalEntry[i].peBlue = i * 4;
        pLP->palPalEntry[i].peFlags = 0;
    }
    return pLP;
}
```

ImageEditView.h

```
// ImageEditView.h : interface of the CImageEditView class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(
    AFX_IMAGEEDITVIEW_H__9D77AEEC_AA14_11D2_8E53_006008A82731__INCLUDED_)
#define
    AFX_IMAGEEDITVIEW_H__9D77AEEC_AA14_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CImageEditView : public CScrollView
```

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

// Attributes
public:
    CImageEditDoc* GetDocument();

// Operations
public:

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CImageEditView)
    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 ~CImageEditView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Generated message map functions
protected:
    //{{AFX_MSG(CImageEditView)
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in ImageEditView.cpp
inline CImageEditDoc* CImageEditView::GetDocument()
   { return (CImageEditDoc*)m_pDocument; }
#endif
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.

#endif
// !defined(
//    AFX_IMAGEEDITVIEW_H__9D77AEEC_AA14_11D2_8E53_006008A82731__INCLUDED_)
```

ImageEditView.cpp

```
// ImageEditView.cpp : implementation of the CImageEditView class
//
#include "stdafx.h"
#include "ImageEdit.h"
#include "ImageEditDoc.h"
#include "ImageEditView.h"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CImageEditView
IMPLEMENT_DYNCREATE(CImageEditView, CScrollView)
BEGIN_MESSAGE_MAP(CImageEditView, CScrollView)
    //{{AFX_MSG_MAP(CImageEditView)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CImageEditView construction/destruction
CImageEditView::CImageEditView()
{
}
CImageEditView::~CImageEditView()
{
}
BOOL CImageEditView::PreCreateWindow(CREATESTRUCT& cs)
{
    return CScrollView::PreCreateWindow(cs);
}
/////////////////////////////////////////////////////////////////////////////
// CImageEditView drawing
void CImageEditView::OnDraw(CDC* pDC)
{
    CImageEditDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    CBitmap* pBitmap = pDoc->GetBitmap ();

    if (pBitmap != NULL) {
        CPalette* pOldPalette;
        CPalette* pPalette = pDoc->GetPalette ();

        if (pPalette != NULL) {
            pOldPalette = pDC->SelectPalette (pPalette, FALSE);
            pDC->RealizePalette ();
       }

        DIBSECTION ds;
        pBitmap->GetObject (sizeof (DIBSECTION), &ds);

        CDC memDC;
        memDC.CreateCompatibleDC (pDC);
        CBitmap* pOldBitmap = memDC.SelectObject (pBitmap);

        pDC->BitBlt (0, 0, ds.dsBm.bmWidth, ds.dsBm.bmHeight, &memDC,
            0, 0, SRCCOPY);

        memDC.SelectObject (pOldBitmap);

        if (pPalette != NULL)
```

```
            pDC->SelectPalette (pOldPalette, FALSE);
    }
}
void CImageEditView::OnInitialUpdate()
{
    CScrollView::OnInitialUpdate ();

    CString string;
    CSize sizeTotal;
    CBitmap* pBitmap = GetDocument ()->GetBitmap ();
    //
    // If a bitmap is loaded, set the view size equal to the bitmap size.
    // Otherwise, set the view's width and height to 0.
    //
    if (pBitmap != NULL) {
        DIBSECTION ds;
        pBitmap->GetObject (sizeof (DIBSECTION), &ds);
        sizeTotal.cx = ds.dsBm.bmWidth;
        sizeTotal.cy = ds.dsBm.bmHeight;
        string.Format (_T ("\t%d x %d, %d bpp"), ds.dsBm.bmWidth,
            ds.dsBm.bmHeight, ds.dsBmih.biBitCount);
    }
    else {
        sizeTotal.cx = sizeTotal.cy = 0;
        string.Empty ();
    }

    AfxGetMainWnd ()->SendMessage (WM_USER_UPDATE_STATS, 0,
        (LPARAM) (LPCTSTR) string);
    SetScrollSizes (MM_TEXT, sizeTotal);
}
/////////////////////////////////////////////////////////////////////////////
// CImageEditView diagnostics
#ifdef _DEBUG
void CImageEditView::AssertValid() const
{
    CScrollView::AssertValid();
}
void CImageEditView::Dump(CDumpContext& dc) const
{
    CScrollView::Dump(dc);
}
CImageEditDoc* CImageEditView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CImageEditDoc)));
    return (CImageEditDoc*)m_pDocument;
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CImageEditView message handlers
```

SpecialStatusBar.h

```
// SpecialStatusBar.h: interface for the CSpecialStatusBar class.
//
//////////////////////////////////////////////////////////////////////
#if !defined(
```

```
   AFX_SPECIALSTATUSBAR_H__4BA7D301_AA24_11D2_8E53_006008A82731__INCLUDED_)
#define
   AFX_SPECIALSTATUSBAR_H__4BA7D301_AA24_11D2_8E53_006008A82731__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CSpecialStatusBar : public CStatusBar
{
public:
    void SetProgress (int nPercent);
    void SetImageStats(LPCTSTR pszStats);
    CSpecialStatusBar();
    virtual ~CSpecialStatusBar();

protected:
    CProgressCtrl m_wndProgress;
    afx_msg int OnCreate (LPCREATESTRUCT lpcs);
    afx_msg void OnSize (UINT nType, int cx, int cy);
    DECLARE_MESSAGE_MAP ()
};
#endif
// !defined(
// AFX_SPECIALSTATUSBAR_H__4BA7D301_AA24_11D2_8E53_006008A82731__INCLUDED_)
```

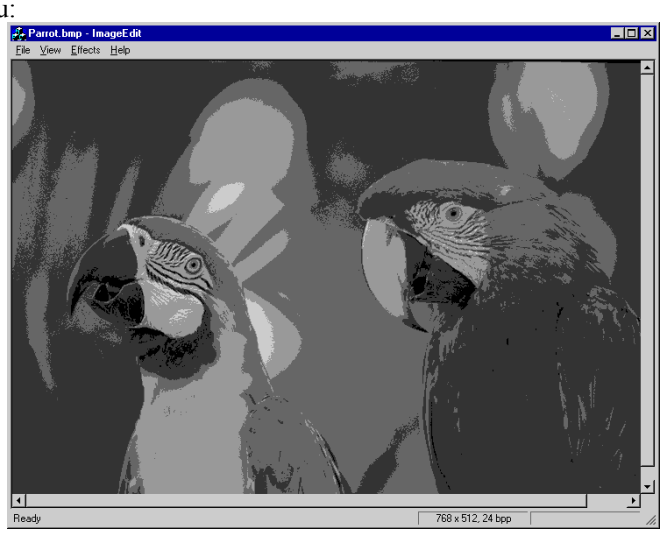
SpecialStatusBar.cpp

```
// SpecialStatusBar.cpp: implementation of the CSpecialStatusBar class.
//
//////////////////////////////////////////////////////////////////////
#include "stdafx.h"
#include "ImageEdit.h"
#include "SpecialStatusBar.h"
#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif
//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

BEGIN_MESSAGE_MAP(CSpecialStatusBar, CStatusBar)
    ON_WM_CREATE ()
    ON_WM_SIZE ()
END_MESSAGE_MAP()
CSpecialStatusBar::CSpecialStatusBar()
{
}
CSpecialStatusBar::~CSpecialStatusBar()
{
}
int CSpecialStatusBar::OnCreate (LPCREATESTRUCT lpcs)
{
    static UINT nIndicators[] =
    {
        ID_SEPARATOR,
```

```
        ID_SEPARATOR,
        ID_SEPARATOR
    };
    if (CStatusBar::OnCreate (lpcs) == -1)
        return -1;
    //
    // Add panes to the status bar.
    //
    SetIndicators (nIndicators, sizeof (nIndicators) / sizeof (UINT));
    //
    // Size the status bar panes.
    //
    TEXTMETRIC tm;
    CClientDC dc (this);
    CFont* pFont = GetFont ();

    CFont* pOldFont = dc.SelectObject (pFont);
    dc.GetTextMetrics (&tm);
    dc.SelectObject (pOldFont);

    int cxWidth;
    UINT nID, nStyle;
    GetPaneInfo (1, nID, nStyle, cxWidth);
    SetPaneInfo (1, nID, nStyle, tm.tmAveCharWidth * 24);
    GetPaneInfo (2, nID, nStyle, cxWidth);
    SetPaneInfo (2, nID, SBPS_NOBORDERS, tm.tmAveCharWidth * 24);
    //
    // Place a progress control in the rightmost pane.
    //
    CRect rect;
    GetItemRect (2, &rect);
    m_wndProgress.Create (WS_CHILD œ WS_VISIBLE œ PBS_SMOOTH,
        rect, this, -1);
    m_wndProgress.SetRange (0, 100);
    m_wndProgress.SetPos (0);
    return 0;
}
void CSpecialStatusBar::OnSize (UINT nType, int cx, int cy)
{
    CStatusBar::OnSize (nType, cx, cy);
    //
    // Resize the rightmost pane to fit the resized status bar.
    //
    CRect rect;
    GetItemRect (2, &rect);
    m_wndProgress.SetWindowPos (NULL, rect.left, rect.top,
        rect.Width (), rect.Height (), SWP_NOZORDER);
}
void CSpecialStatusBar::SetImageStats(LPCTSTR pszStats)
{
    SetPaneText (1, pszStats, TRUE);
}
void CSpecialStatusBar::SetProgress(int nPercent)
{
    ASSERT (nPercent >= 0 && nPercent <= 100);
```

```
    m_wndProgress.SetPos (nPercent);
}
```

Màn hình kết quả như sau:



CHƯƠNG 4. LẬP TRÌNH CƠ SỞ DỮ LIỆU VỚI MFC

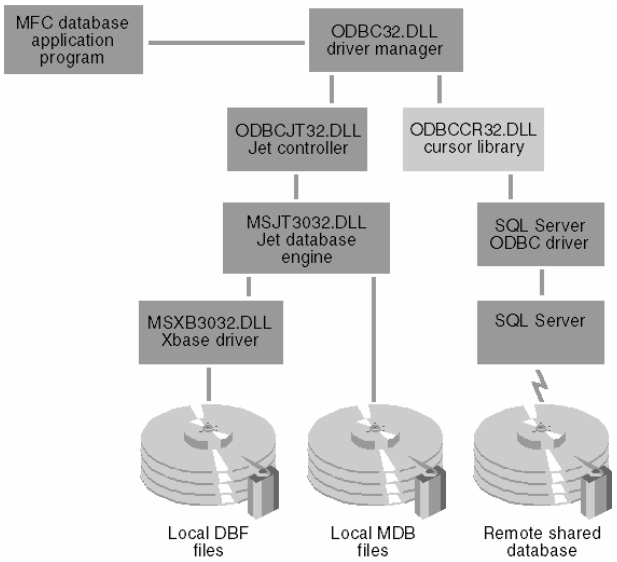
4.1 TRUY XUẤT VỚI ODBC/DAO

4.1.1 Vấn đề quan tâm

- Hiểu và sử dụng được các class về truy xuất dữ liệu mà MFC hỗ trợ.
- Hiểu và lập trình được các sự kiện tương tác dữ liệu.

4.1.2 Giới thiệu

Cấu trúc ODBC được mô tả như sau:



Tổ chức các class giúp truy xuất dữ liệu bao gồm:

Class	Use
CDaoWorkspace	Giao diện quản lý các đơn kết nối.
CDaoDatabase	Giao diện làm việc với database
CDaoRecordset	Giao diện làm việc với tập các mẩu tin (kiểu table, dynaset, snapshot)
CDaoTableDef	Giao diện thao tác định nghĩa hay truy xuất bảng dữ liệu.
CDaoQueryDef	Giao diện truy vấn dữ liệu

Trong quá trình truy xuất dữ liệu, dữ liệu lấy từ cơ sở dữ liệu về thường được lưu trữ và quản lý bởi class CRecordset. Thành phần của class CRecordset như sau:

Tên hàm	Mô tả
Open	Mở recordset
AddNew	Chuẩn bị để thêm mới mẫu dữ liệu vào bảng dữ liệu
Update	Hoàn tất thao tác AddNew hay Edit bởi thực hiện lưu vào cơ sở dữ liệu.
Delete	Xoá mẫu dữ liệu hiện hành
Edit	Chuẩn bị để thay đổi mẫu dữ liệu vào bảng dữ liệu
IsBOF	Nhận biết vị trí đứng đầu trong tập dữ liệu
IsEOF	Nhận biết vị trí đứng cuối trong tập dữ liệu
MoveNext	Di chuyển sang mẫu dữ liệu kế tiếp
MoveFirst	Di chuyển sang mẫu dữ liệu đầu tiên
MoveLast	Di chuyển sang mẫu dữ liệu cuối cùng
MovePrev	Di chuyển sang mẫu dữ liệu liền trước
GetDefaultConnect	Lấy về chuỗi kết nối cơ sở dữ liệu mặc định
GetDefaultSQL	Lấy về chuỗi truy vấn cơ sở dữ liệu mặc định
DoFieldExchange	Thực hiện trao đổi dữ liệu giữa thành phần dữ liệu trong tập dữ liệu với biến liên kết tương ứng
GetStatus	Lấy về chỉ mục của mẫu dữ liệu hiện hành và trạng thấy của nó
GetRecordCount	Lấy về số lượng mẫu dữ liệu trong tập dữ liệu
GetODBCFieldCount	Lấy về số lượng thành phần dữ liệu trong tập dữ liệu
GetODBCFieldInfo	Lấy về thông tin thành phần dữ liệu trong tập dữ liệu

Việc nhận biết trạng thái và số lượng "mẫu tin" (row/record) trong quá trình truy vấn dữ liệu từ đối tượng CRecordset được thực hiện bởi hàm **GetRecordCount**() và **GetStatus**()

Việc di chuyển giữa các record bởi hàm **MoveNext**(), **MovePrevious**(), **MoveFirst**(), **MoveLast**().

Để xử lý bẫy lỗi trong quá trình truy xuất, dùng cơ chế **try…catch** theo ví dụ sau:

Ví dụ 1:

```
try {
  m_pSet->Delete();        // thao tác cập nhật - xoá dữ liệu
}
catch(CDBException* e) {
  AfxMessageBox(e->m_strError);
  e->Delete();
  m_pSet->MoveFirst(); // lost our place!
  UpdateData(FALSE);
  return;
}
m_pSet->MoveNext();
```

Để duyệt tập hợp dữ liệu trong đối tượng CRecordset, dùng vòng lặp theo ví dụ sau:

Ví dụ 2:

Tạo 1 class CMySet như sau:

```
class CMySet: public CRecordset
{
       int          MyID;
       CString      MyName;
…
       CString GetDefaultConnect() {return _T("ODBC;DSN=MyDabase");};
       CString GetDefaultSQL() { return _T("[MyTable]");};
…
}
```

và truy cập dữ liệu này như sau:

- Mở recordset:

```
CMySet* m_pSet;
…
if(m_pSet->IsOpen()) {
   m_pSet->Close();
 }
m_pSet->Open();
```

- Kiểm tra recordset có chứa dữ liệu không

```
if(m_pSet->IsBOF())       // detects empty recordset
{
        return;
}
```

- Duyệt qua các mẫu dữ liệu trong recordset:

```
m_pSet->MoveFirst();  // fails if recordset is empty
while(!m_pSet->IsEOF())
{
        str.Format("%ld", m_pSet->m_MyID);
        pDC->TextOut(10, 20, str);
        pDC->TextOut(10, 50, m_pSet->m_MyName);
```

```
        m_pSet->MoveNext();
}
m_pSet->Close();
```

- Chuyển sang chế độ thêm mới một mẫu tin:

```
if (AfxMessageBox("Are you  want to add new record?",1,1) = IFYES ){
        m_pSet->AddNew();
}
```

- Chuyển sang chế độ cập nhật một mẫu tin:

```
if (AfxMessageBox("Are you  want to edit this record?",1,1) = IFYES ){
        m_pSet->Edit();
}
```

- Thực hiện lưu thông tin vào cơ sở dữ liệu

```
m_pSet->Update();
```

- Thực hiện xoá 1 mẫu tin

```
if (AfxMessageBox("Are you  want to delete this record?",1,1) = IFYES ){
        m_pSet->Delete();
}
```

Chú ý:

Để sử dụng các lớp truy xuất dữ liệu, cần thêm vào hàng cuối cùng trong file **StdAfx.h** hàng khai báo như sau:

```
#include <afxdb.h>
```

Trong file *.RC, thêm hàng khai báo

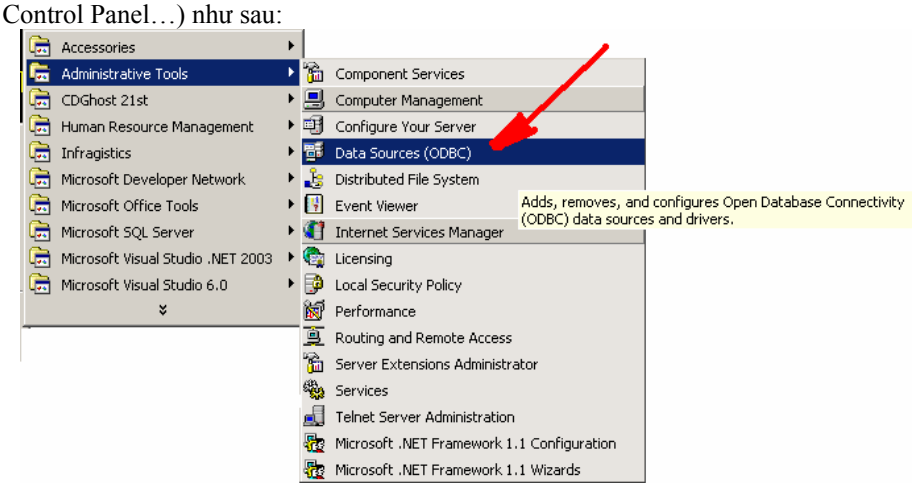
```
"#include ""afxdb.rc""  // database resources\r\n"
```

sau hàng lệnh
"#include ""afxprint.rc"" // printing print preview resources\r\n"
thêm hàng khai báo

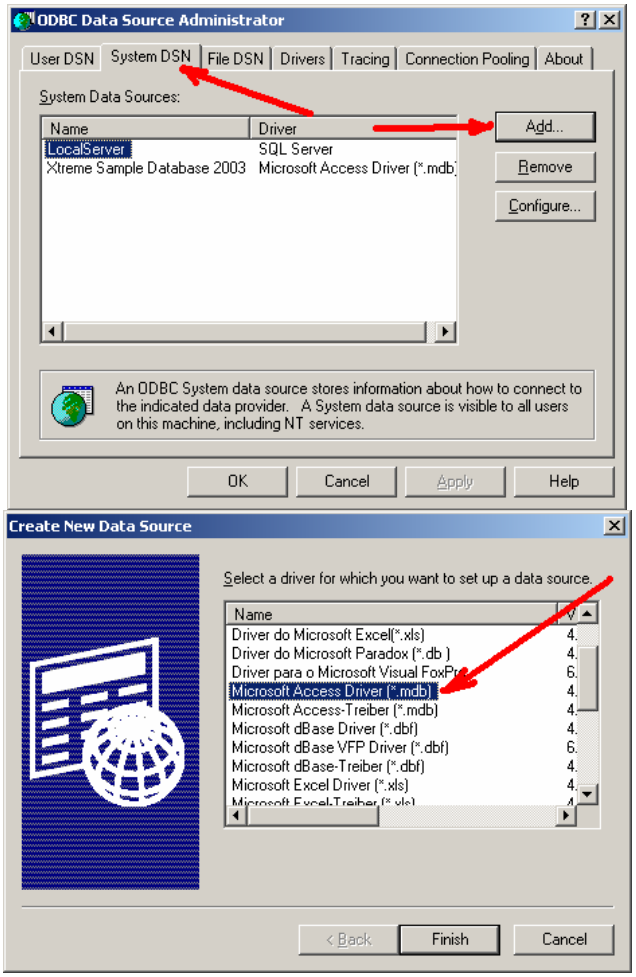
```
#include "afxdb.rc"    // database resources
```

sau hàng lệnh
#include "afxprint.rc" // printing print preview resources

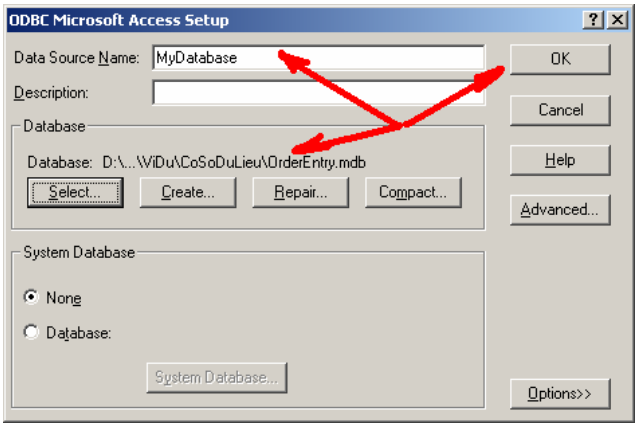
- Để khai báo nguồn dữ liệu, cần sử dụng tiện ích DataSource(ODBC) có trong chương trình Windows (tại Control Panel…) như sau:



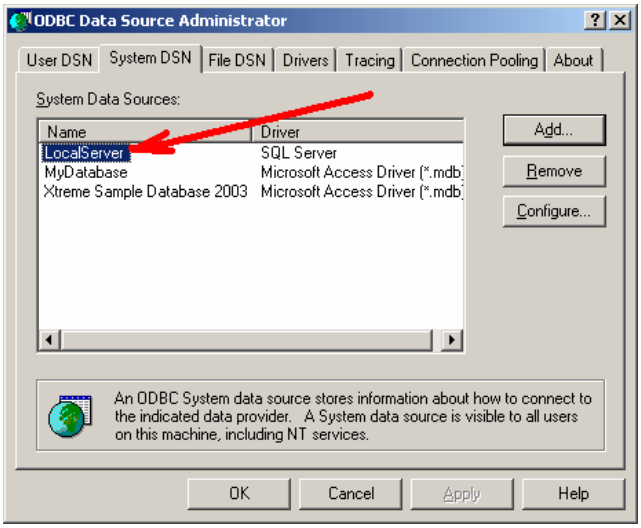
và quá trình khai báo tên nguồn dữ liệu được tiến hành trong ứng dụng này:


ODBC Data Source Administrator
User DSN
System DSN
File DSN
Drivers
Tracing
Connection Pooling
About
System Data Sources:
Name
Driver
LocalServer
SQL Server
Xtreme Sample Database 2003
Microsoft Access Driver (*.mdb)
Add...
Remove
Configure...
An ODBC System data source stores information about how to connect to the indicated data provider. A System data source is visible to all users on this machine, including NT services.
OK
Cancel
Apply
Help
Create New Data Source
Select a driver for which you want to set up a data source.
Name
Driver do Microsoft Excel(*.xls)
Driver do Microsoft Paradox (*.db)
Driver para o Microsoft Visual FoxPro
Microsoft Access Driver (*.mdb)
Microsoft Access-Treiber (*.mdb)
Microsoft dBase Driver (*.dbf)
Microsoft dBase VFP Driver (*.dbf)
Microsoft dBase-Treiber (*.dbf)
Microsoft Excel Driver (*.xls)
< Back
Finish
Cancel

và:


ODBC Microsoft Access Setup
Data Source Name:
MyDatabase
Description:
Database
Database: D:\...\ViDu\CoSoDuLieu\OrderEntry.mdb
Select...
Create...
Repair...
Compact...
System Database
None
Database:
System Database...
OK
Cancel
Help
Advanced...
Options>>

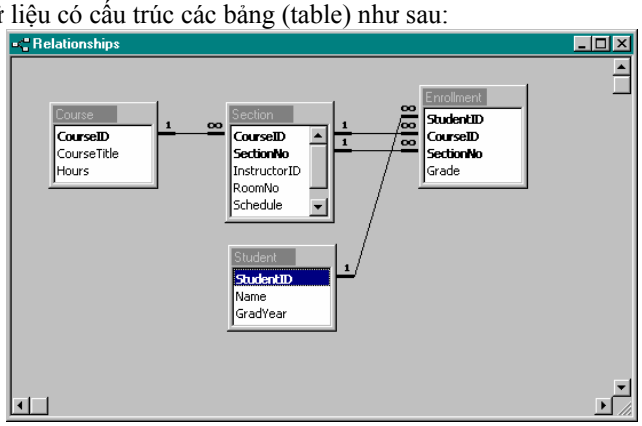
và:



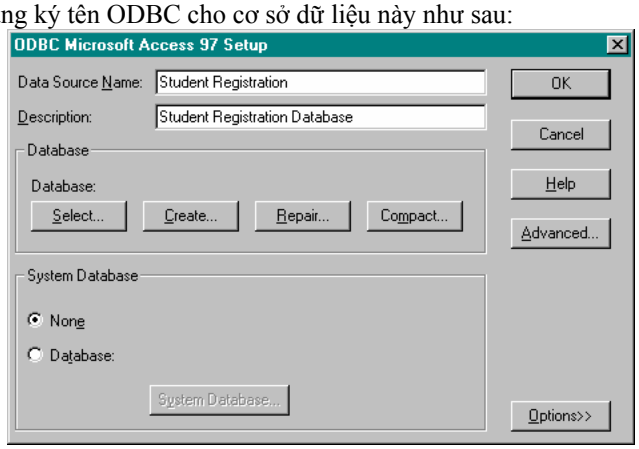
4.1.3 Ví dụ tổng hợp

4.1.3.1 Chương trình 1:

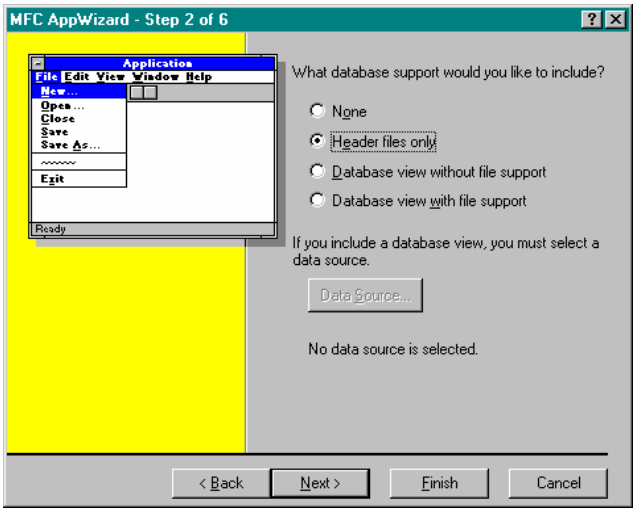
- B1: Cho cơ sở dữ liệu có cấu trúc các bảng (table) như sau:



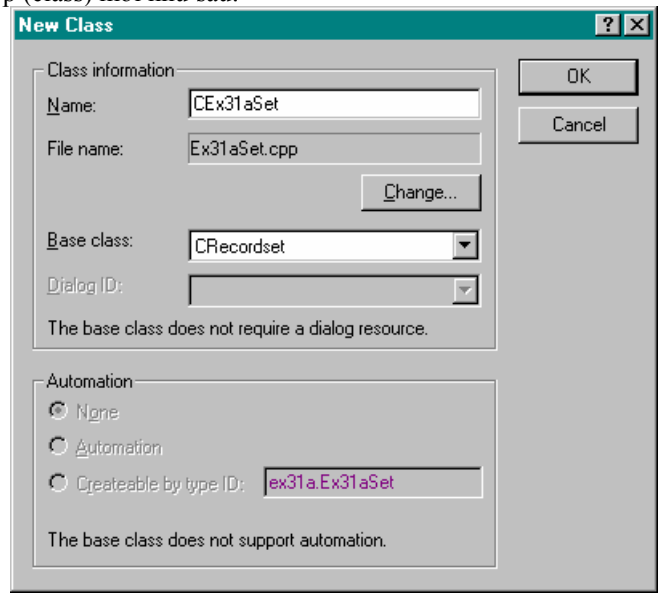
- B2: Thực hiện đăng ký tên ODBC cho cơ sở dữ liệu này như sau:



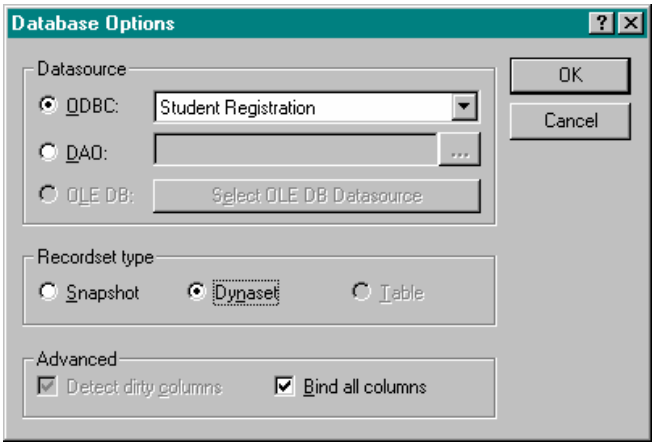
- B3: Tạo ứng dụng trong môi trường Visual C++ như sau:



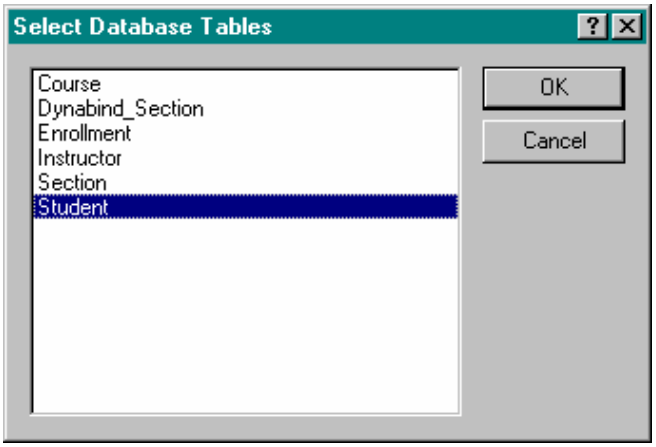
- B4: Tạo thêm lớp (class) mới như sau:



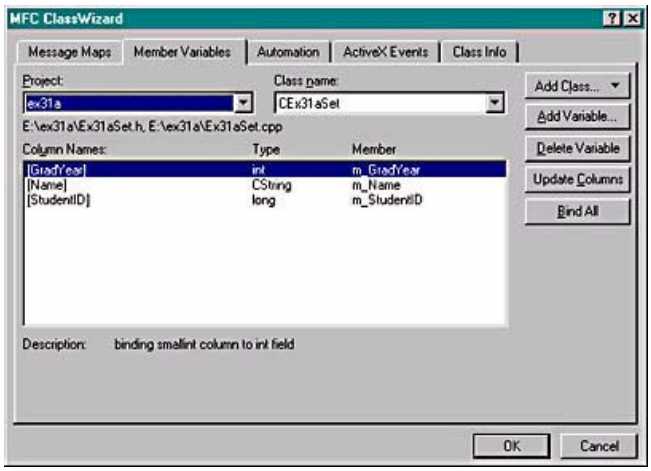
và:



và:



- B5: Tạo các biến liên kết như sau:



- B6: Cập nhật chương trình như sau:

Trong lớp (class) CEx31aDoc của file ex31aDoc.h, thêm:

```
CEx31aSet m_ex31aSet;
```

Trong file ex31aDoc.cpp, thêm:

```
#include "ex31aSet.h"
```

Trong lớp (class) CEx31aView của file ex31aView.h, thêm:

```
CEx31aSet* m_pSet;
```

Cập nhật lại các hàm trong file ex31aView.cpp như sau:

```
void CEx31aView::OnDraw(CDC* pDC)
{
    TEXTMETRIC tm;
    pDC->GetTextMetrics(&tm);
    int nLineHeight=tm.tmHeight+tm.tmExternalLeading;
    CPoint pText(0,0);

    int y = 0;
    CString str;
    if (m_pSet->IsBOF()) { // detects empty recordset
        return;
    }
    m_pSet->MoveFirst();   // fails if recordset is empty
    while (!m_pSet->IsEOF()) {
        str.Format("%ld", m_pSet->m_StudentID);
        pDC->TextOut(pText.x, pText.y, str);
        pDC->TextOut(pText.x+1000, pText.y, m_pSet->m_Name);
        str.Format("%d", m_pSet->m_GradYear);
```

```
        pDC->TextOut(pText.x+4000, pText.y, str);
        m_pSet->MoveNext();
        pText.y -= nLineHeight;
    }
}
void CEx31aView::OnInitialUpdate()
{
    CScrollView::OnInitialUpdate();
    CSize sizeTotal(8000, 10500);
    SetScrollSizes(MM_HIENGLISH, sizeTotal);

    m_pSet = &GetDocument()->m_ex31aSet;
    // Remember that documents/views are reused in SDI applications!
    if (m_pSet->IsOpen()) {
        m_pSet->Close();
    }
    m_pSet->Open();
}
```

Trong file ex31aView.cpp, thêm:

```
#include "ex31aSet.h"
```

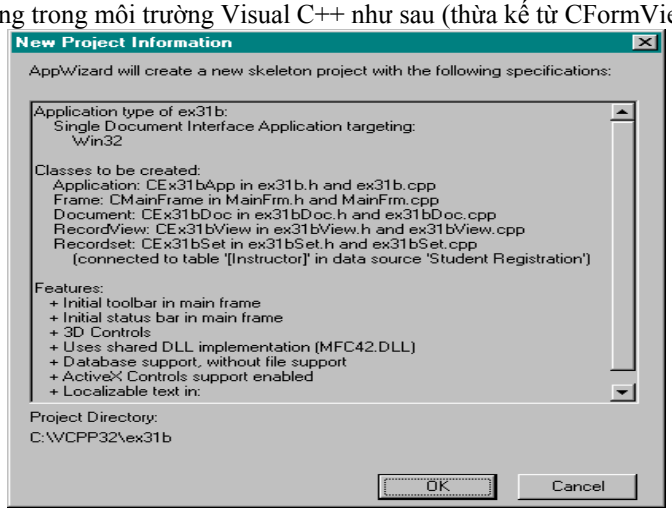
Trong file ex31a.cpp, thêm:

```
#include "ex31aSet.h"
```

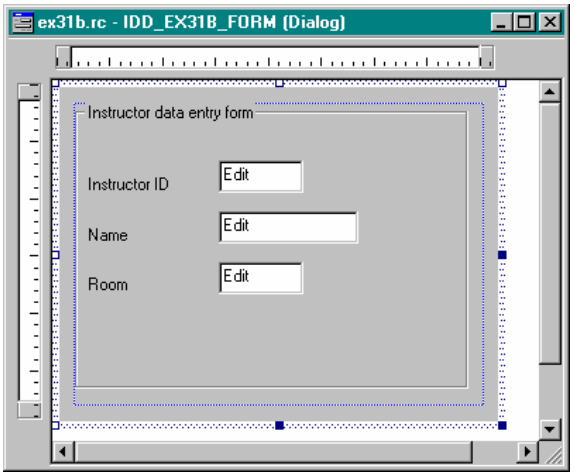
- B7: Biên dịch và thực thi chương trình, kết quả như sau:

4.1.3.2 Chương trình 2:

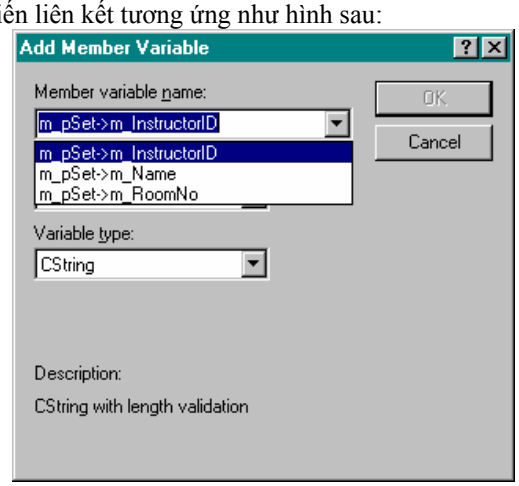
- B1: Tạo ứng dụng trong môi trường Visual C++ như sau (thừa kế từ CFormView):



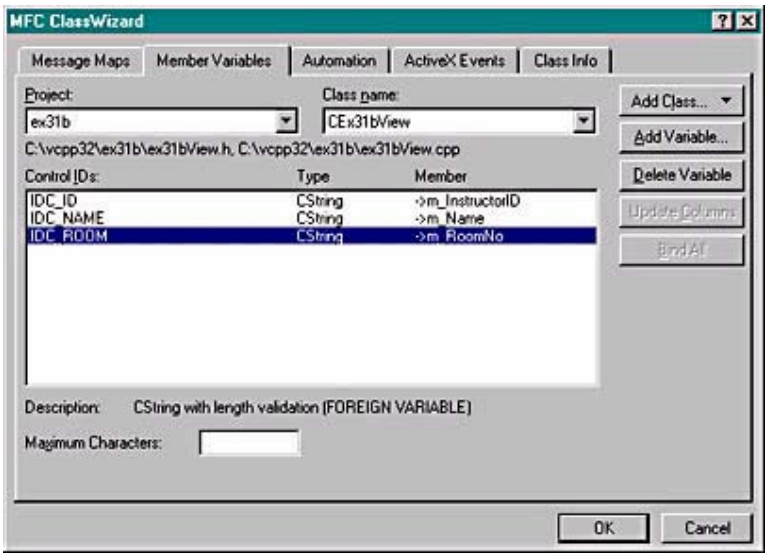
- B2: Tạo giao diện chương trình như sau (với các edit box có tên lần lượt là IDC_NAME, and IDC_ROOM):



- B3: Lần lượt tạo các biến liên kết tương ứng như hình sau:



và:



- B4: Biên dịch và thực thi chương trình
- B5: Tạo các hàm liên kết các icon (next, back, first, last) có trên toolbar nhuư sau:

Menu Command	Command ID	Command Handler	Update Command UI Handler
Add Record	ID_RECORD_ADD	OnRecordAdd	
Clear Fields	ID_RECORD_CLEARFIELDS	OnRecordClearfields	
Delete	ID_RECORD_DELETE	OnRecordDelete	OnUpdateRecordDelete

Record			
Update Record	ID_RECORD_UPDATE	OnRecordUpdate	OnUpdateRecordUpdate

- B6: Cập nhật hàm OnMove trong lớp CEx31bView như sau:

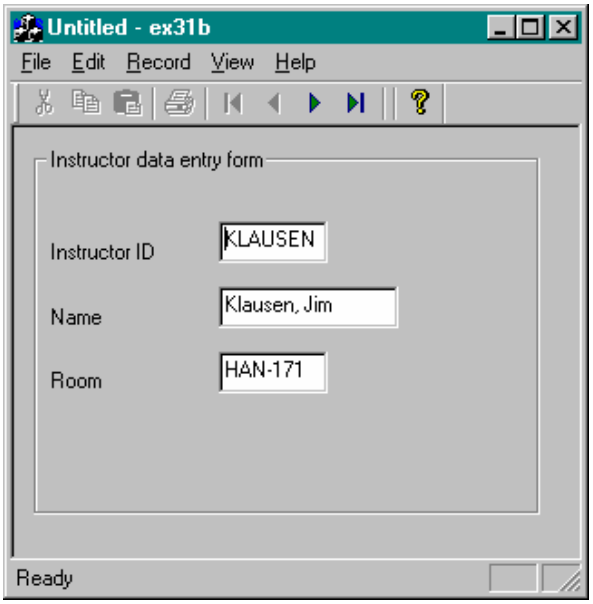
```
BOOL CEx31bView::OnMove(UINT nIDMoveCommand)
{
    switch (nIDMoveCommand)
    {
    case ID_RECORD_PREV:
        m_pSet->MovePrev();
        if (!m_pSet->IsBOF())
            break;

    case ID_RECORD_FIRST:
        m_pSet->MoveFirst();
        break;

    case ID_RECORD_NEXT:
        m_pSet->MoveNext();
        if (!m_pSet->IsEOF())
            break;
        if (!m_pSet->CanScroll()) {
            // Clear screen since we're sitting on EOF
            m_pSet->SetFieldNull(NULL);
            break;
        }

    case ID_RECORD_LAST:
        m_pSet->MoveLast();
        break;

    default:
        // unexpected case value
        ASSERT(FALSE);
    }

    // Show results of Move operation
    UpdateData(FALSE);
    return TRUE;
}
```

và:

```
void CEx31bView::OnRecordAdd()
{
    m_pSet->AddNew();
    UpdateData(TRUE);
    if (m_pSet->CanUpdate()) {
        m_pSet->Update();
    }
    if (!m_pSet->IsEOF()) {
        m_pSet->MoveLast();
    }
    m_pSet->Requery(); // for sorted sets
    UpdateData(FALSE);
}
```

```
void CEx31bView::OnRecordClearfields()
{
    m_pSet->SetFieldNull(NULL);
    UpdateData(FALSE);
}

void CEx31bView::OnRecordDelete()
{
    CRecordsetStatus status;
    try {
        m_pSet->Delete();
    }
    catch(CDBException* e) {
        AfxMessageBox(e->m_strError);
        e->Delete();
        m_pSet->MoveFirst(); // lost our place!
        UpdateData(FALSE);
        return;
    }
    m_pSet->GetStatus(status);
    if (status.m_lCurrentRecord == 0) {
        // We deleted last of 2 records
        m_pSet->MoveFirst();
    }
    else {
        m_pSet->MoveNext();
    }
    UpdateData(FALSE);
}

void CEx31bView::OnUpdateRecordDelete(CCmdUI* pCmdUI)
{
    pCmdUI->Enable(!m_pSet->IsEOF());
}

void CEx31bView::OnRecordUpdate()
{
    m_pSet->Edit();
    UpdateData(TRUE);
    if (m_pSet->CanUpdate()) {
        m_pSet->Update();
    }
// should requery if key field changed
}

void CEx31bView::OnUpdateRecordUpdate(CCmdUI* pCmdUI)
{
    pCmdUI->Enable(!m_pSet->IsEOF());
}
```

- B7: Biên dịch và thực thi chương trình, kết quả như sau:



4.2 TRUY XUẤT VỚI OLEDB

4.2.1 Vấn đề quan tâm

- Hiểu về bản chất của ADO và sử dụng được các class về truy xuất dữ liệu mà MFC hỗ trợ.

4.2.2 Giới thiệu OLEDB

OLEDB là một tập hợp các giao diện truy xuất dữ liệu thông qua COM.

Cấu trúc của OLEDB bao gồm:

- *Enumerators*: có tác vụ tìm các nguồn dữ liệu khả dụng.
- *Data source objects*: Data source objects chứa các cơ chế kết nối tới nguồn dữ liệu - một phiên làm việc (*session*) được tạo ra khi chương trình kết nối đến một nguồn dữ liệu.
- *Sessions*: Sessions thể hiện một kết nối(phiên làm việc) đến nguồn dữ liệu – có thể nguồn dữ liệu có thể tạo nhiều sessions. Mỗi Sessions có thể tạo ra transactions, commands, và rowsets.
- *Transaction*: là đối tượng quản lý các thao tác truy xuất dữ liệu và bảo đảm an toàn dữ liệu.
- *Commands*: là đối tượng cho phép thực thi các lệnh SQL. Nếu lệnh SQL là lệnh SELECT, thì đối tượng này sẽ tạo ra (nhận về) rowsets. Một seesion có thể tạo sử dụng với commands.
- *Rowsets*: là tập dữ liệu dạng bảng (tabular). Rowsets có thể được tạo từ session hay command.
- *Errors*: Errors có thể được tạo ra bởi bất kỳ giao diện của đối tượng OLE DB nào. Errors chứa các thông tin về lỗi.

4.2.3 Thực hiện tác vụ truy xuất dữ liệu với OLEDB:

MFC cung cấp một số class giúp thao tác với OLEDB như:

Class	Use
CDataSource	This class represents the data source component and manages the connection to a data source.
CEnumerator	This class provides a way to select a provider by cycling through a list of providers. Its functionality is equivalent to the SQLBrowseConnect and SQLDriverConnect functions.
CSession	This class handles transactions. You can use this class to create rowsets, commands, and many other objects. A CDataSource object creates a CSession object using the CSession::Open method
CAccessor	This class is used when a record is statically bound to a data source—it contains the pre-existing data buffer and understands the data format up front. CAccessor is used when you know the structure and the type of the database ahead of time.
CDynamicAccessor	This class is used for retrieving data from a source whose structure is not known at design time. This class uses

	IColumnsInfo::GetColumnInfo to get the database column information. CDynamicAccessor creates and manages the data buffer.
CDynamicParameterAccessor	This class is similar to CDynamicAccessor except that it's used with commands. When used to prepare commands, CDynamicParameterAccessor can get parameter information from the ICommandWithParameters interface, which is especially useful for handling unknown command types.
CManualAccessor	This class lets you access whatever data types you want as long as the provider can convert the type. CManualAccessor handles both result columns and command parameters.
CTable	The CTable class is a minimal class implementation that opens a table on a data source(which you can specify programmatically). Use this class when you need bare-bones access to a source, since CTable is designed for simple providers that do not support commands
CCommand	CCommand is used mostly for executing commands. This class has a function named Open that executes singular commands. This class also has a function named Prepare for setting up a command to execute multiple times

Một số giao diện được cung cấp để tương tác với OLEDB như:

Interface	Required?	Implemented?
IDBInitialize	Mandatory	Yes
IDBCreateSession	Mandatory	Yes
IDBProperties	Mandatory	Yes
IPersist	Mandatory	Yes
IDBDataSourceAdmin	Optional	No
IDBInfo	Optional	No
IPersistFile	Optional	No
ISupportErrorInfo	Optional	No
IDBInitialize	Mandatory	Yes
IDBCreateSession	Mandatory	Yes
IDBProperties	Mandatory	Yes
IPersist	Mandatory	Yes
IDBDataSourceAdmin	Optional	No
IDBInfo	Optional	No
IPersistFile	Optional	No
ISupportErrorInfo	Optional	No
IGetDataSource	Mandatory	Yes
IOpenRowset	Mandatory	Yes
ISessionProperties	Mandatory	Yes
IDBCreateCommand	Optional	Yes
IDBSchemaRowset	Optional	Yes
IIndexDefinition	Optional	No
ISupportErrorInfo	Optional	No
ITableDefinition	Optional	No
ITransactionJoin	Optional	No
ITransactionLocal	Optional	No
ITransactionObject	Optional	No
IAccessor	Mandatory	Yes
IColumnsInfo	Mandatory	Yes
IConvertType	Mandatory	Yes
IRowset	Mandatory	Yes
IRowsetInfo	Mandatory	Yes
IColumnsRowset	Optional	No
IConnectionPointContainer	Optional	Yes, through ATL
IRowsetChange	Optional	No
IRowsetIdentity	Required for Level 0	Yes
IRowsetLocate	Optional	No

IRowsetResynch	Optional	No
IRowsetScroll	Optional	No
IRowsetUpdate	Optional	No
ISupportErrorInfo	Optional	No

Ví dụ 1:

```
class CMainFrame : public CFrameWnd
{
…
    CDataSource m_db;
    bool m_bConnectionValid;
    CSession m_session;
…
}
…

BOOL CMainFrame::OpenConnection()
{
    HRESULT hr = m_db.Open();

    if(SUCCEEDED(hr))
        m_bConnectionValid = true;

    if(SUCCEEDED(hr) && m_session.Open(m_db) != S_OK){
        AfxMessageBox("Could not open a Session to the database");
        return FALSE;
    }

    return TRUE;
}
```

Ví dụ 2:

```
CCommand<CDynamicAccessor, CRowset> dbCommand;
try {
    Recordset20Ptr spRs;
    ADORecordsetConstructionPtr spADOsCt;

    CDBPropSet propset(DBPROPSET_ROWSET);
    propset.AddProperty(DBPROP_CLIENTCURSOR, true);
    propset.AddProperty(DBPROP_IRowsetChange, true);
    propset.AddProperty(DBPROP_UPDATABILITY, DBPROPVAL_UP_CHANGE |
        DBPROPVAL_UP_INSERT | DBPROPVAL_UP_DELETE);

    CString sCommand;
    sCommand.Format("SELECT * FROM [%s]", sTableName);

    HRESULT hr = dbCommand.Create(
        pMainFrame->m_session,(LPCTSTR)sCommand);
    if(FAILED(hr))
        _com_issue_error(hr);

    hr = dbCommand.Open(&propset, NULL, true);
    if(FAILED(hr))
        _com_issue_error(hr);

    hr = spRs.CreateInstance(__uuidof(Recordset));
    if(FAILED(hr))
```

```
        _com_issue_error(hr);

    hr = spRs->QueryInterface(
        __uuidof(ADORecordsetConstruction),(void **)&spADOsCt);
    if(FAILED(hr))
        _com_issue_error(hr);

    hr = spADOsCt->put_Rowset(dbCommand.m_spRowset);
    if(FAILED(hr))
        _com_issue_error(hr);

    //Demonstrates, how to populate DataGrid by assigning it a Recordset object.
    m_ctlDataGrid.SetCaption(sTableName);
    m_ctlDataGrid.SetRefDataSource(NULL);
    m_ctlDataGrid.SetRefDataSource((LPUNKNOWN) spRs );
          m_ctlDataGrid.Refresh();
}
catch(_com_error &e) {
    AfxMessageBox(GetErrorDescription(e));
}

UpdateData(FALSE);
```

Ví dụ 3:

```
try {
    CTables tableSet;
    HRESULT hr = tableSet.Open(pMainFrame->m_session);
    if(SUCCEEDED(hr)) {
        CString sName, sNameShort, sSchema;
        int nPos = -1;
        HRESULT hr = S_OK;
        int nIndex = 0;

        hTreeRoot = GetTreeCtrl().InsertItem("Tables", 0, 0);

        while(tableSet.MoveNext() == S_OK) {
            sName = tableSet.m_szName;
            sNameShort = sName;
            nPos = sName.Find(';');
            if(nPos != -1)
                sName = sName.Left(nPos);
            if(sName.Find(' ') != -1) // MS SQL Server scenario
                sName = "[" + sName + "]";
            // Alternatively...
            sSchema = tableSet.m_szSchema;

            HTREEITEM hTreeSPRoot = GetTreeCtrl().InsertItem(
                sName, 1, 1, hTreeRoot);
        }
    }
    else
    {
        if(FAILED(hr))
            _com_issue_error(hr);
    }
}
```

```
catch(_com_error e) {
    bRet = false;
    AfxMessageBox(GetErrorDescription(e));
}
catch(...) {
    bRet = false;
    AfxMessageBox("UnKnown Error");
}
```

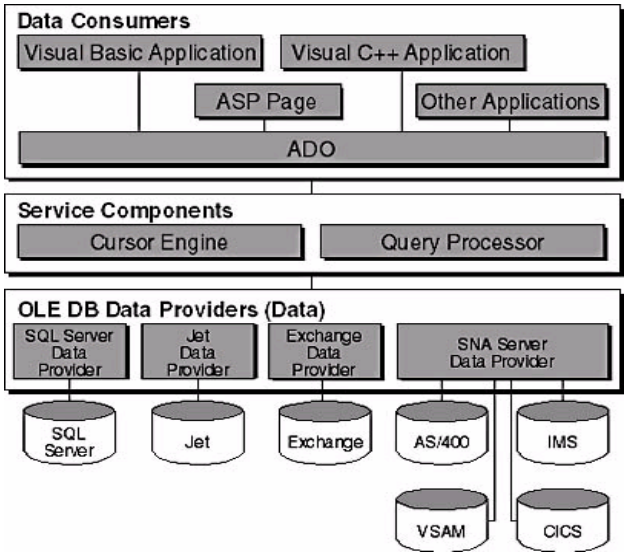
4.3 TRUY XUẤT VỚI ADO

4.3.1 Vấn đề quan tâm

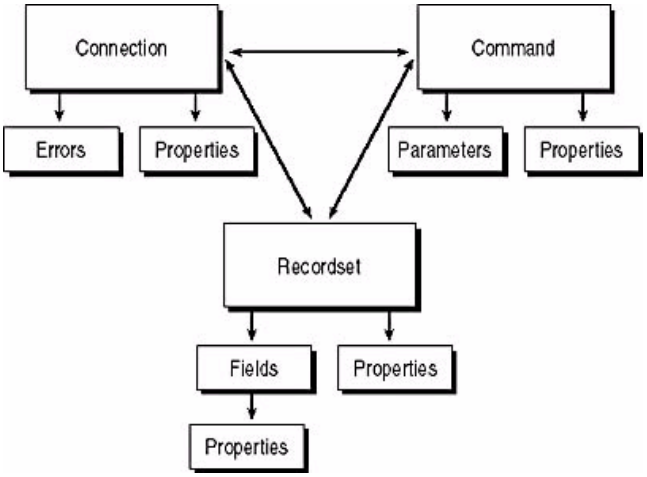
- Hiểu về bản chất của ADO và sử dụng được các class về truy xuất dữ liệu mà MFC hỗ trợ.

4.3.2 Giới thiệu ADO

ADO là một tập hợp các giao diện truy xuất dữ liệu thông qua COM (tương tự như OLEDB nhưng sử dụng đơn giản hơn). ADO đóng vai trò trung gian tương tác với chương trình ứng dụng và cơ sở dữ liệu, được thể hiện như hình sau:



Cấu trúc của ADO bao gồm:



- **Connection**: là đối tượng quản lý các tác vụ truy cập/kết nối cơ sở dữ liệu.
- **Command**: là đối tượng quản lý các tác vụ cập nhật dữ liệu
- **Recordset**: là đối tượng quản lý dữ liệu truy vấn được
- **Error**: là đối tượng quản lý lỗi xãy ra

4.3.3 Thực hiện truy xuất cơ sở dữ liệu với ADO:

Cần thêm khai báo về thư viện ADO vào hàng cuối cùng của file StdAfx.h như sau:

```
#import "c:\program files\common  files\system\ado\msado15.dll" \
no_namespace \
rename ("EOF", "adoEOF")
```

Cần thêm lệnh ::CoInitialize(NULL); trước khi tạo kết nối

Việc truy xuất dữ liệu với ADO được thực hiện thông qua các ví dụ minh hoạ sau đây:

Khai báo biến cần thiết:

```
_ConnectionPtr  m_pConnection;
_CommandPtr             m_pCommand;
```

Khai báo kết nối đến cơ sở dữ liệu:

```
// Kết nối đến Access:
m_szConnection="Provider=Microsoft.JET.OLEDB.4.0;Data source=xxx";
```

(trong đó xxx là tên file Access có đầy đủ đường dẫn)

```
// Kết nối đến SQL Server:
m_szConnection="Provider=SQLOLEDB;Data source=my_server_name;Initial
Catalog=my_database_name;User ID=my_user;Password=my_password;";
```

và:

```
// Kết nối tổng quát:
m_szConnection="File name=xxx";
```

*(trong đó xxx là tên file *.UDL có đầy đủ đường dẫn)*

Kết nối đến cơ sở dữ liệu:

```
// Khoi tao moi truong COM <-- lenh nay bat buoc phai co
::CoInitialize(NULL);
try {
      m_pConnection.CreateInstance(__uuidof(Connection));
      m_pConnection-> Open((_bstr_t)m_szConnectionString,"","",-1);
}
catch(_com_error *e)     {
      CString Error = e->ErrorMessage();
      AfxMessageBox(e->ErrorMessage());
      return FALSE;
}
catch(...)
{
      AfxMessageBox("Loi bat ky");
      return FALSE;
}
```

Đóng kết nối:

```
if (m_pConnection->GetState() == adStateOpen)
      m_pConnection->Close();
m_pConnection = NULL;
```

Truy xuất dữ liệu:

```
if (m_pConnection->GetState() == adStateOpen) {
```

```
    // Reset noi dung dang co trong listbox
    m_lstData.ResetContent();
    //
    m_pRecordset.CreateInstance(__uuidof(Recordset));
    _variant_t myValue;
    _variant_t myKey((short)4);
    int nCount = 0;
    try {
        m_pRecordset->Open(
            "SELECT * FROM Products", m_pConnection.GetInterfacePtr(),
            adOpenDynamic,
            adLockOptimistic,
            adCmdText);
        while(!m_pRecordset->adoEOF) {
            myValue = m_pRecordset->GetCollect("ProductName");
            if (myValue.vt!=VT_NULL) {
                m_lstData.AddString((char*)_bstr_t(myValue));
                myKey = m_pRecordset->GetCollect("ProductID");
                m_lstData.SetItemData(nCount++, (int)myKey.iVal);
            }
            m_pRecordset->MoveNext();
        }
        m_pRecordset->Close();
    }
    catch(_com_error *e) {
        CString Error = e->ErrorMessage();
        AfxMessageBox(e->ErrorMessage());
    }
    catch(...) {
        MessageBox("Whoa");
    }
    // Always set these pointers to null when you are done with them!
    m_pRecordset = NULL;
    UpdateData(FALSE);
}
```

Cập nhật dữ liệu:

```
if (AfxMessageBox("Delete it?", 1,0)!=IDOK)
    return;
if (m_pConnection->GetState() == adStateOpen) {
    //
    UpdateData(TRUE);
    //
    CString szSQL;
```

```
    szSQL.Format("DELETE FROM Products WHERE ProductID=%d", m_nID);
    //
    try {
        m_pConnection->Execute((_bstr_t)szSQL, NULL, 0);
    }
    catch(_com_error *e) {
        CString Error = e->ErrorMessage();
    }
    catch(...)    {
        MessageBox("Whoa");
    }
    OnButtonLoad();
}
```

Ví dụ tổng hợp:

MyADO.h

```
// MyADO.h: interface for the CMyADO class.
//
//////////////////////////////////////////////////////////////////////
#if !defined(AFX_MYADO_H__E144D98C_2388_4800_BC00_F7E963816A73__INCLUDED_)
#define AFX_MYADO_H__E144D98C_2388_4800_BC00_F7E963816A73__INCLUDED_

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

#import "C:\Program Files\Common Files\System\ADO\msado15.dll"
no_namespace
rename( "EOF", "adoEOF" )

class CMyADO
{
public:
      CMyADO();
      virtual ~CMyADO();

      HRESULT Open(_bstr_t btConnectionString,_bstr_t btUserID,
                                                  _bstr_t btPassword);
      HRESULT Close();
      HRESULT AddParameterReturnValue();
      HRESULT AddParameterInputLong(_bstr_t btParameterName, long lValue);
      HRESULT AddParameterInputText(_bstr_t btParameterName, _bstr_t btValue);
      HRESULT AddParameterInputOutputLong(_bstr_t btParameterName, long lValue);
      HRESULT AddParameterInputOutputText(_bstr_t btParameterName,
                                    _bstr_t btValue, DWORD dwMaxTextSize );
      HRESULT AddParameterOutputLong(_bstr_t btParameterName);
      HRESULT AddParameterOutputText(_bstr_t btParameterName,DWORD dwMaxTextSize);
      HRESULT Execute();
      HRESULT GetFieldLong(_bstr_t btFieldName, long* plValue);
      HRESULT GetFieldText(_bstr_t btFieldName,char* szText, DWORD dwMaxTextSize);
      HRESULT GetParameterReturnValue(long* plReturnValue);
      HRESULT GetParameterLong(_bstr_t btParameterName, long* plValue);
```

```
	HRESULT GetParameterText(_bstr_t btParameterName, char* szText,
                                                    DWORD dwMaxTextSize);
	HRESULT Initialize(_bstr_t btStoredProcedureName);
	BOOL IsEOF();
	HRESULT MoveNext();
protected:
	HRESULT GetRecordCount(long* lRecordCount);
private:
	HRESULT AddParameter( _bstr_t btParameterName, DataTypeEnum enDataType,
                              ParameterDirectionEnum enParameterDirection,
                              long lSize, _variant_t vtValue);
	HRESULT GetField(_variant_t vtFieldName, _variant_t& vtValue);
	HRESULT GetParameter(_variant_t vtParameterName, _variant_t& vtValue);
	BOOL IsConnected();
	BOOL IsInitialized();

	_ConnectionPtr m_pConnectionPtr;
	_CommandPtr m_pCommandPtr;
	_RecordsetPtr m_pRecordsetPtr;
};
#endif // !defined(AFX_MYADO_H__E144D98C_2388_4800_BC00_F7E963816A73__INCLUDED_)
```

MyADO.cpp

```
// MyADO.cpp: implementation of the CMyADO class.
//////////////////////////////////////////////////////////////////////
#include <stdio.h>
#include <string.h>
#include "MyADO.h"
//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////
//     Class Constructor
CMyADO::CMyADO()
{
	m_pCommandPtr = NULL;
}
//     Class Destructor
CMyADO::~CMyADO()
{
	if( m_pRecordsetPtr )
	{
		if( m_pRecordsetPtr->State == adStateOpen )
			m_pRecordsetPtr->Close();
		m_pRecordsetPtr = NULL;
	}
	m_pCommandPtr = NULL;
}
//Create a Parameter and Add it to the CommandPtr Object
//(Which will be used to Execute the Stored Procedure)
HRESULT CMyADO::AddParameter( _bstr_t btParameterName, DataTypeEnum enDataType,
ParameterDirectionEnum enParameterDirection, long lSize, _variant_t vtValue )
{
	HRESULT hReturn = S_FALSE;
	if( IsConnected() && IsInitialized())
	{
		try
```

```
            {
                  _ParameterPtr pParameterPtr = m_pCommandPtr->CreateParameter(
btParameterName, enDataType, enParameterDirection, lSize, vtValue );
                  m_pCommandPtr->Parameters->Append( pParameterPtr );
                  hReturn = S_OK;
            }
            catch( _com_error& eComError )
            {
                  char szErrorMsg[256];
                  _snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::Execute() - %s\n", eComError.ErrorMessage());
                  OutputDebugString( szErrorMsg );
            }
            catch( ... )
            {
            }
      }
      return hReturn;
}
//Add Parameter Heler Function for Long Type and Input Direction
HRESULT CMyADO::AddParameterInputLong( _bstr_t btParameterName, long lValue )
{
      return AddParameter( btParameterName, adInteger, adParamInput, sizeof(long),
_variant_t( lValue ));
}
//Add Parameter Helper Function for Text Type and Input Direction
HRESULT CMyADO::AddParameterInputText( _bstr_t btParameterName, _bstr_t btValue )
{
      return AddParameter( btParameterName, adVarChar, adParamInput,
btValue.length(), _variant_t( btValue ));
}
//Add Parameter Helper Function for Long Type and Input/Output Direction
HRESULT CMyADO::AddParameterInputOutputLong( _bstr_t btParameterName, long lValue )
{
      return AddParameter( btParameterName, adInteger, adParamInputOutput, sizeof(
long ), _variant_t( lValue ));
}
//Add Parameter Helper Function for Text Type and Input/Output Direction
HRESULT CMyADO::AddParameterInputOutputText( _bstr_t btParameterName, _bstr_t
btValue, DWORD dwMaxTextSize )
{
      return AddParameter( btParameterName, adVarChar, adParamInputOutput,
dwMaxTextSize, _variant_t( btValue ));
}
//Add Parameter Helper Function for Long Type and Output Direction
HRESULT CMyADO::AddParameterOutputLong( _bstr_t btParameterName )
{
      _variant_t vtNull;
      return AddParameter( btParameterName, adInteger, adParamOutput, 0, vtNull );
}
//Add Parameter Helper Function for Text Type and Output Direction
HRESULT CMyADO::AddParameterOutputText( _bstr_t btParameterName, DWORD
dwMaxTextSize )
{
      _variant_t vtNull;
```

```
	return AddParameter( btParameterName, adVarChar, adParamOutput,
dwMaxTextSize, vtNull );
}
//Add Parameter Helper Function for Return Value
HRESULT CMyADO::AddParameterReturnValue()
{
	_variant_t vtNull;
	return AddParameter( "RETURN_VALUE", adInteger, adParamReturnValue, 0,
vtNull );
}
//Close the Current ADO Connection
HRESULT CMyADO::Close()
{
	HRESULT hReturn = S_FALSE;
	if( m_pConnectionPtr )
	{
		if( m_pConnectionPtr->State == adStateOpen )
		{
			try
			{
				hReturn = m_pConnectionPtr->Close();
				m_pConnectionPtr = NULL;
			}
			catch( _com_error& eComError )
			{
				char szErrorMsg[256];
				_snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::Close() - %s\n", eComError.ErrorMessage());
				OutputDebugString( szErrorMsg );
			}
			catch( ... )
			{
			}
		}
		else
			hReturn = S_OK;
	}
	else
		hReturn = S_OK;
	return hReturn;
}
//Execute the Stored Procedure using the CommandPtr Object
HRESULT CMyADO::Execute()
{
	HRESULT hReturn = S_FALSE;
	if( IsConnected() && IsInitialized())
	{
		try
		{
			m_pRecordsetPtr = m_pCommandPtr->Execute( NULL, NULL,
adCmdStoredProc );
			hReturn = S_OK;
		}
		catch( _com_error& eComError )
		{
```

```
                    char szErrorMsg[256];
                    _snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::Execute() - %s\n", eComError.ErrorMessage());
                    OutputDebugString( szErrorMsg );
             }
             catch( ... )
             {
             }
      }
      return hReturn;
}
//Retrieve a Value from the Recordset (which was created during Stored Procedure
Execution)
HRESULT CMyADO::GetField( _variant_t vtFieldName, _variant_t& vtValue )
{
      HRESULT hReturn = S_FALSE;
      if( IsConnected() && IsInitialized())
      {
             try
             {
                    vtValue = m_pRecordsetPtr->Fields->GetItem(vtFieldName)->Value;
                    hReturn = S_OK;
             }
             catch( _com_error& eComError )
             {
                    char szErrorMsg[256];
                    _snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::GetField() - %s\n", eComError.ErrorMessage());
                    OutputDebugString( szErrorMsg );
             }
             catch( ... )
             {
             }
      }
      return hReturn;
}
//Get Field Helper Function for Long Type
HRESULT CMyADO::GetFieldLong( _bstr_t btFieldName, long* plValue )
{
      _variant_t vtValue;
      HRESULT hReturn = GetField( btFieldName, vtValue );
      if( hReturn == S_OK )
             *plValue = ( long )vtValue;
      return hReturn;
}
//Get Field Helper Function for Text Type
HRESULT CMyADO::GetFieldText( _bstr_t btFieldName, char* szText, DWORD
dwMaxTextSize )
{
      _variant_t vtValue;
      HRESULT hReturn = GetField( btFieldName, vtValue);
      if( hReturn == S_OK )
      {
             _bstr_t btValue = ( _bstr_t )vtValue;
             if( dwMaxTextSize < btValue.length())
```

```
                  hReturn = S_FALSE;
            else
                  strcpy( szText, btValue );
      }
      return hReturn;
}
//Retrieve a Parameter (which was previously set up as either an Output or
InputOutput Direction and is set during Stored Procedure Execution)
HRESULT CMyADO::GetParameter( _variant_t vtParameterName, _variant_t& vtValue )
{
      HRESULT hReturn = S_FALSE;
      if( IsConnected() && IsInitialized())
      {
            try
            {
                  vtValue = m_pCommandPtr->Parameters->GetItem(vtParameterName)
->Value;
                  hReturn = S_OK;
            }
            catch( _com_error& eComError )
            {
                  char szErrorMsg[256];
                  _snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::GetParameter() - %s\n", eComError.ErrorMessage());
                  OutputDebugString( szErrorMsg );
            }
            catch( ... )
            {
            }
      }
      return hReturn;
}
//Retrieve Parameter Helper Function for Long Type
HRESULT CMyADO::GetParameterLong( _bstr_t btParameterName, long* plValue )
{
      _variant_t vtValue;
      HRESULT hReturn = GetParameter( btParameterName, vtValue );
      if( hReturn == S_OK )
            *plValue = ( long )vtValue;
      return hReturn;
}
//Retrieve Parameter Helper Function for Return Value
HRESULT CMyADO::GetParameterReturnValue( long* plReturnValue )
{
      return GetParameterLong( "RETURN_VALUE", plReturnValue );
}
//Retrieve Parameter Helper Function for Text Type
HRESULT CMyADO::GetParameterText( _bstr_t btParameterName, char* szText, DWORD
dwMaxTextSize )
{
      _variant_t vtValue;
      HRESULT hReturn = GetParameter( btParameterName, vtValue );
      if( hReturn == S_OK )
      {
            _bstr_t btValue = ( _bstr_t )vtValue;
```

```
		if( dwMaxTextSize < btValue.length())
			hReturn = S_FALSE;
		else
			strcpy( szText, btValue );
	}
	return hReturn;
}
//Retrieve the Record Count for the Recordset (which was created during Stored
Procedure Execution)
HRESULT CMyADO::GetRecordCount( long* lRecordCount )
{
	HRESULT hReturn = S_FALSE;
	if( m_pRecordsetPtr )
	{
		try
		{
			*lRecordCount = m_pRecordsetPtr->RecordCount;
		}
		catch( _com_error& eComError )
		{
			char szErrorMsg[256];
			_snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::GetParameter() - %s\n", eComError.ErrorMessage());
			OutputDebugString( szErrorMsg );
		}
		catch( ... )
		{
		}
	}
	return hReturn;
}
//Close the Recordset and Initialize the CommandPtr Object
HRESULT CMyADO::Initialize( _bstr_t btStoredProcedureName )
{
	HRESULT hReturn = S_FALSE;
	if( IsConnected())
	{
		m_pCommandPtr = NULL;

		if( m_pRecordsetPtr )
		{
			if( m_pRecordsetPtr->State == adStateOpen )
				m_pRecordsetPtr->Close();
			m_pRecordsetPtr = NULL;
		}
		m_pCommandPtr.CreateInstance( __uuidof( Command ));
		m_pCommandPtr->ActiveConnection = m_pConnectionPtr;
		m_pCommandPtr->CommandText = btStoredProcedureName;
		m_pCommandPtr->CommandType = adCmdStoredProc;

		hReturn = S_OK;
	}
	return hReturn;
}
//Check for Connection Status
```

```
BOOL CMyADO::IsConnected()
{
	return ( m_pConnectionPtr );
}
//Check for EOF on the Recordset
//(which was created during Stored Procedure Execution)
BOOL CMyADO::IsEOF()
{
	BOOL bReturn = TRUE;
	if( m_pRecordsetPtr )
		if(m_pRecordsetPtr->State == adStateOpen && !m_pRecordsetPtr->adoEOF)
			bReturn = FALSE;
	return bReturn;
}
//Check for Initialization Status (CommandPtr Object is valid)
BOOL CMyADO::IsInitialized()
{
	return ( m_pCommandPtr );
}
//Open a new ADO Connection
HRESULT CMyADO::Open( _bstr_t btConnectionString, _bstr_t btUserID, _bstr_t
btPassword )
{
	HRESULT hReturn = S_FALSE;
	if( m_pConnectionPtr == NULL )
	{
		m_pConnectionPtr.CreateInstance( __uuidof( Connection ));
		try
		{
			hReturn = m_pConnectionPtr->Open( btConnectionString, btUserID,
btPassword, 0 );
			if( hReturn == S_OK )
				m_pConnectionPtr->CursorLocation = adUseClient;
		}
		catch( _com_error& eComError )
		{
			char szErrorMsg[256];
			_snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::Open( '%s', '%s', '%s' ) - %s\n", ( char* )btConnectionString, ( char*
)btUserID, ( char* )btPassword, eComError.ErrorMessage());
			OutputDebugString( szErrorMsg );
		}
		catch( ... )
		{
		}
	}
	else
		hReturn = S_OK;
	return hReturn;
}
//Move to the Next Record in the Recordset (which was created during Stored
Procedure Execution)
HRESULT CMyADO::MoveNext()
{
	HRESULT hResult = S_FALSE;
```

```
	if( !IsEOF())
	{
		try
		{
			hResult = m_pRecordsetPtr->MoveNext();
		}
		catch( _com_error& eComError )
		{
			char szErrorMsg[256];
			_snprintf( szErrorMsg, sizeof( szErrorMsg ), "ERROR in
CMyADO::MoveNext() - %s\n", eComError.ErrorMessage());
			OutputDebugString( szErrorMsg );
		}
		catch( ... )
		{
		}
	}
	return hResult;
}
```

Main.cpp

```
#include <stdio.h>
#include "MyADO.h"

void main()
{
	// Initialize COM
	CoInitialize( NULL );

	// Instantiate the Object
	CMyADO MyADOObject;

	// Open( ConnectionString, UserID, Password ) the Connection
	if( MyADOObject.Open( "TestMyADO", "", "" ) == S_OK )
	{
		// Some Example Strings
		char* pNames[] = { "First Name", "Second Name", "Third Name", "Fourth
Name", "Fifth Name", "Sixth Name", "Seventh Name", "Eighth Name", "Nineth Name",
"Tenth Name" };
		char* pValues[] = { "Value One", "Value Two", "Value Three", "Value
Four", "Value Five", "Value Six", "Value Seven", "Value Eight", "Value Nine",
"Value Ten" };
		// The Index
		DWORD dwIndex = 0;

		// The Insert Loop
		while( dwIndex < 10 )
		{
			// Initialize( StoredProcedureName ) the Stored Procedure
			if( MyADOObject.Initialize( "InsertMyADO" ) == S_OK )
			{
				// Add the Return Value Parameter
				if( MyADOObject.AddParameterReturnValue() == S_OK )
				{
					// Add the Output Long Parameter
```

```
                                    if( MyADOObject.AddParameterOutputLong( "ID" ) ==
S_OK )
                                    {
                                          // Add the Input Text Parameters
                                          if( MyADOObject.AddParameterInputText(
"Name", pNames[dwIndex] ) == S_OK &&
                                                MyADOObject.AddParameterInputText(
"Value", pValues[dwIndex] ) == S_OK )
                                          {
                                                // Execute the Stored Procedure
                                                if( MyADOObject.Execute() == S_OK )
                                                {
                                                      long lReturnValue = 0;
                                                      long lID = 0;

                                                      // Retrieve the Return Value
and the Output Paramter set up above
                                                      if(
MyADOObject.GetParameterReturnValue( &lReturnValue ) == S_OK &&

      MyADOObject.GetParameterLong( "ID", &lID ) == S_OK )
                                                      {
                                                            // Sanity check that
does nothing : )
                                                            if( lReturnValue == lID
)
                                                                  printf(
"Inserted Record with ID: %2d, Name: %15s, Value: %15s\n", lID, pNames[dwIndex],
pValues[dwIndex] );
                                                      }
                                                      else
                                                            printf( "ERR: Unable to
Retrieve Return Value And ID\n" );
                                                }
                                                else
                                                      printf( "ERR: Unable to
Execute InsertMyADO\n" );
                                          }
                                          else
                                                printf( "ERR: Unable to Add the
Input Text Values\n" );
                                    }
                                    else
                                          printf( "ERR: Unable to Add the Output Long
Value\n" );
                              }
                              else
                                    printf( "ERR: Unable to Add the Return Value\n" );
                        }
                        else
                              printf( "ERR: Unable to Initialize InsertMyADO\n" );

                        // Increment the Index
                        dwIndex++;
                  }
```

```
            // Initialize( StoredProcedureName ) the Stored Procedure
            if( MyADOObject.Initialize( "SelectMyADO" ) == S_OK )
            {
                  // Execute the Stored Procedure
                  if( MyADOObject.Execute() == S_OK )
                  {
                        // Ensure there are more Records to Retrieve
                        while( !MyADOObject.IsEOF())
                        {
                              long lID = 0;
                              char szName[15];
                              char szValue[15];

                              // Retrieve the Record Fields
                              if( MyADOObject.GetFieldLong( "ID", &lID ) == S_OK
&&
                                    MyADOObject.GetFieldText( "Name", szName,
sizeof( szName )) == S_OK &&
                                    MyADOObject.GetFieldText( "Value", szValue,
sizeof( szValue )) == S_OK )
                                    printf( "Selected Record with ID: %2d,
Name: %15s, Value: %15s\n", lID, szName, szValue );

                              // Move to the Next Record
                              MyADOObject.MoveNext();
                        }
                  }
                  else
                        printf( "ERR: Unable to Execute SelectMyADO\n" );
            }
            else
                  printf( "ERR: Unable to Initialize SelectMyADO\n" );

            // Close the Connection
            MyADOObject.Close();
      }

      // For every action there must be an opposite and equal reaction
      CoUninitialize();
}
```

CHƯƠNG 5. MFC VÀ ACTIVE X

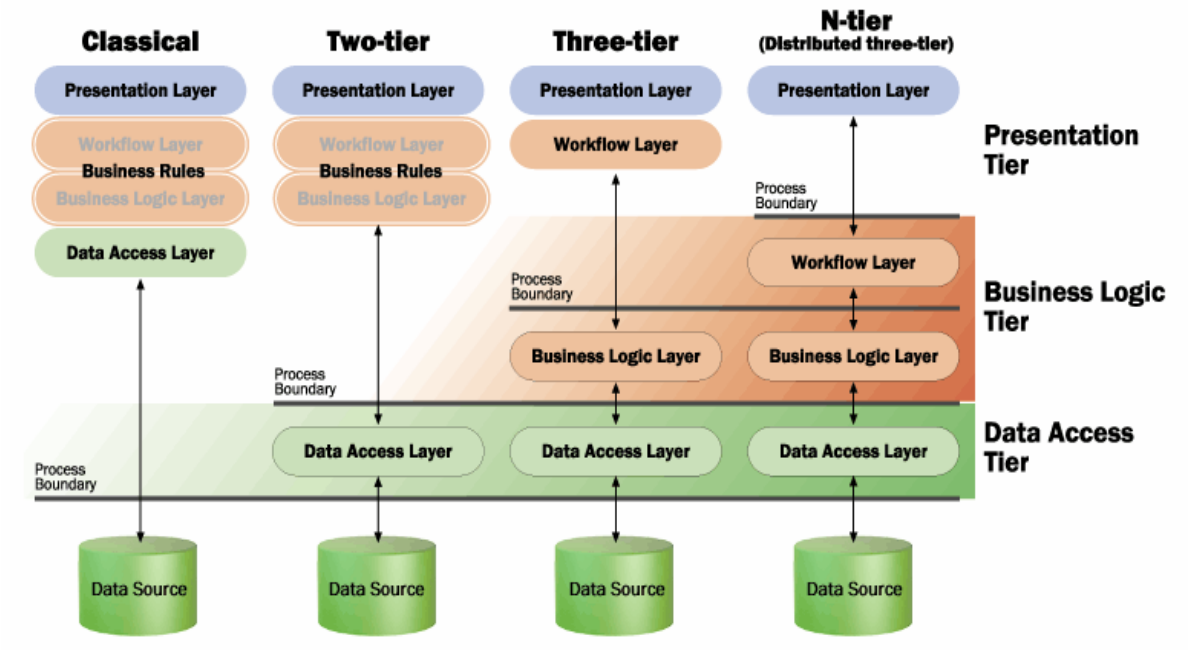
5.1 COMPONENT OBJECT MODEL (COM)

5.1.1 Vấn đề quan tâm

- Hiểu cấu trúc và vai trò của mô hình COM
- Khai thác được ưu điểm của COM trong lập trình ứng dụng.

5.1.2 Giới thiệu mô hình 3-lớp và n-lớp

Là cấu trúc ứng dụng mà trong đó các nhóm ứng dụng được phân chia theo tính năng và vai trò theo từng lớp (*tier*).



Mỗi mô hình có từng ưu/khuyết điểm khác nhau, được sử dụng trong các ngữ cảnh khác nhau.

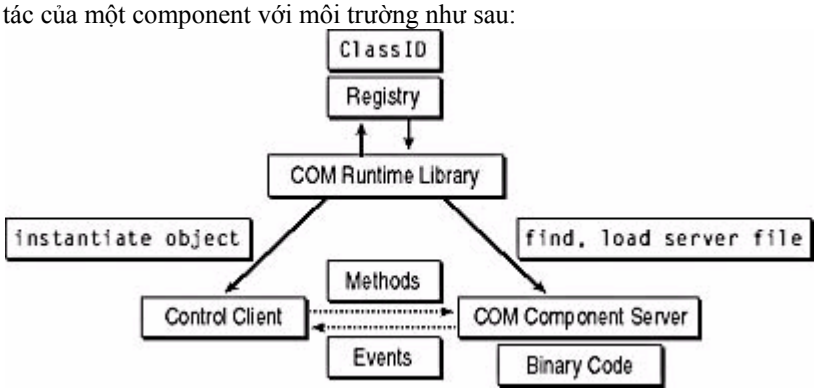
5.1.3 Giới thiệu COM

COM là mô hình đối tượng thành phần, được sử dụng trong các ứng dụng 3-lớp (hoặc n-lớp) nhằm tăng hiệu suất hoạt động, tính bảo mật của hệ thống và tính linh hoạt cao.

Với COM, chúng ta có thể mở rộng khả năng liên kết và tích hợp giữa các thành phần (độc lập ngôn ngữ) trong hệ thống vì bản thân COM được thể hiện ở dạng mã nhị phân.

Có thể dùng VC++ hay VB để tạo COM, và bất cứ ứng dụng nào cũng có thể tương tác với COM (nếu được hỗ trợ).

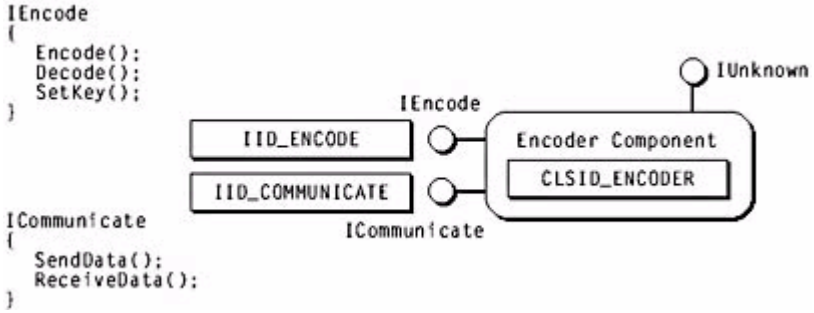
Tổ chức và tương tác của một component với môi trường như sau:



5.1.4 Cấu trúc COM

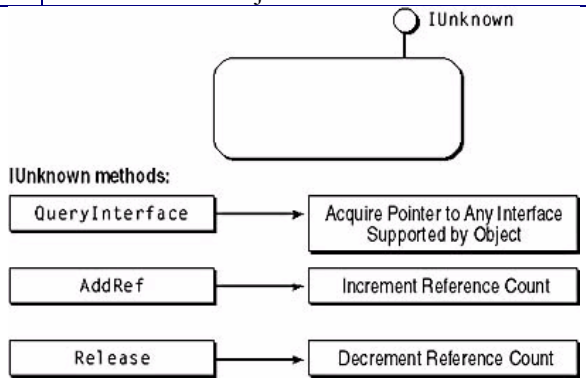
COM được tổ chức dưới dạng tập các class – trong đó mỗi class bao gồm các ***method*** (hàm thành viên), ***attribute*** (biến thành viên) và các ***property***.

Các method và các property được nhóm vào trong các interface (giao diện) – các interface có (như là vỏ bọc của các class) có nhiệm vụ giao tiếp với các ứng dụng khác nhằm che dấu/ngăn chặn sự truy cập trực tiếp vào các thành phần của class trong COM - đồng thời cung cấp các phương thức truy cập các method và property này. Microsoft đã định nghĩa sẵn khoảng 100 interface để hỗ trợ cho COM, nhưng người sử dụng có thể tự tạo interface riêng biệt.

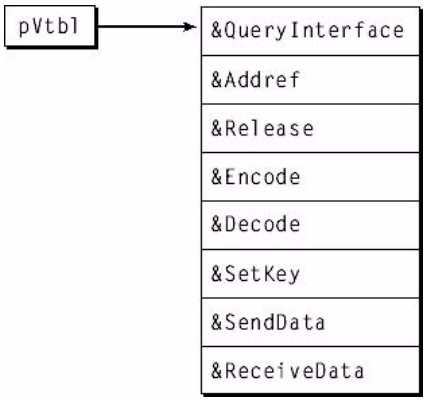


Mỗi COM khi tạo ra luôn có sẵn interface IUnknown và các method như:

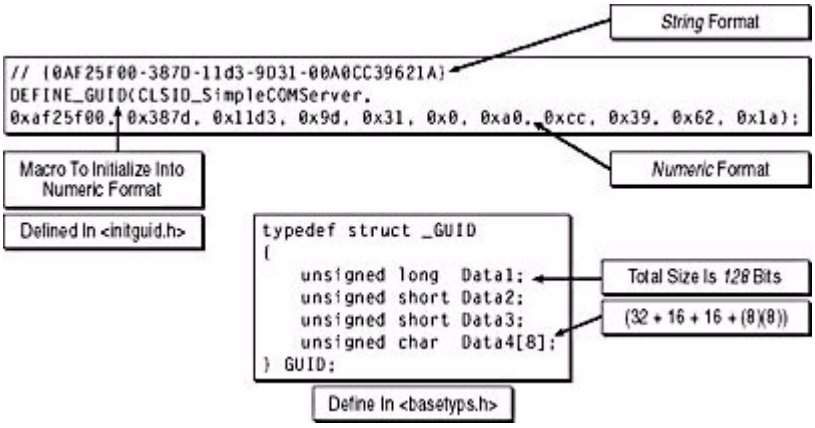
Phương thức	Mô tả
QueryInterface	Returns a pointer to another interface
AddRef	Increments the object's reference count
Release	Decrements the object's reference count



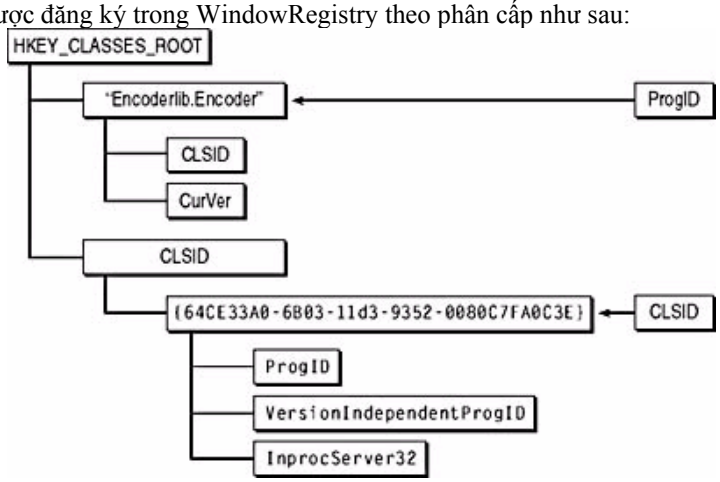
và tổ chức lưu trữ như sau:



Mỗi class trong COM khi tạo ra cần có class ID (hay CLSID) để khai báo với hệ thống, giá trị này được thể hiện bởi chuỗi định nghĩa 128-bit, trong đó thông tin liên quan gồm:



Mỗi component đều được đăng ký trong WindowRegistry theo phân cấp như sau:



Ví dụ 1:

Để khởi tạo đối tượng của COM, gọi hàm kích hoạt **CoCreateInstance** cùng với CLSID của đối tượng đó, kết quả nhận về là một pointer quản lý của đối tượng đó.

Ví dụ 2:

```
Khởi tạo một COM class có CLSID là CLSID_Object và liên kết tới interface IMath bởi
pointer pMath
IMath* pMath;
CoCreateInstance (CLSID_Object, NULL, CLSCTX_SERVER, IID_IMath, (void**) &pMath);
```

Đối tượng COM được khởi tạo bởi thao tác new nhưng được tự động hủy (mà không dùng delete). Có thể một interface hỗ trợ nhiều truy xuất đồng thời, số lượng truy xuất có thể được nhận biết và quản lý bởi một biến thành viên và các sự kiện AddRef, Release:

Ví dụ 3:

```
ULONG __stdcall CComClass::AddRef()
{
    return ++m_lRef;
}
ULONG __stdcall CComClass::Release()
{
    if (--m_lRef == 0) {
        delete this;
        return 0;
    }
    return m_lRef;
}
```

Vấn đề truy xuất đồng thời nhiều đối tượng được xử lý theo trình tự sau:

Ví dụ 4:

```
IMath* pMath;
HRESULT hr = CoCreateInstance (CLSID_Object, NULL,
```

```
    CLSCTX_SERVER, IID_IMath, (void**) &pMath);

if (SUCCEEDED (hr)) { // CoCreateInstance worked.

    ISpelling* pSpelling;
    hr = pMath->QueryInterface (IID_ISpelling, (void**) &pSpelling);
    if (SUCCEEDED (hr)) {
        // Got the interface pointer!
        …
        pSpelling->Release();
    }
    pMath->Release();
}
```

5.1.5 COM server

Một file đối tượng thực thi mà hiện thực một đối tượng COM được gọi là COM server.

Section HKEY_CLASSES_ROOT\CLSID trong registry sẽ chứa các thông tin về CLSID và các đối tượng thực thi liên quan.

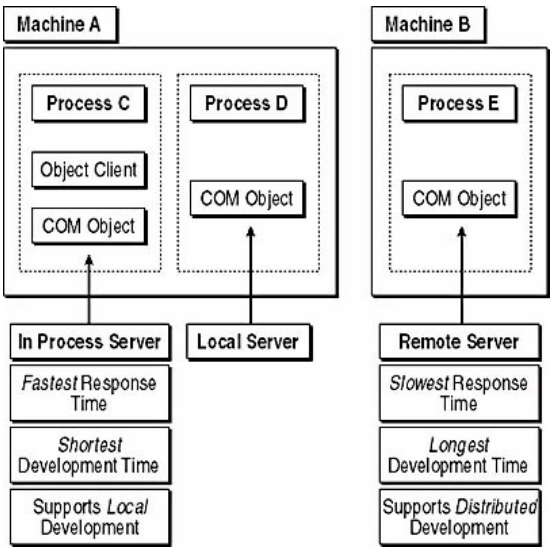
Ví dụ 5:

```
Nếu server file là MathSvr.exe hiện thực cho đối tượng Math objects, và người dùng
gọI CoCreateInstance vớI CLSID của Math, thì COM sẽ tìm CLSID trong registry, và
lấy được đường dẫn của MathSvr.exe rồi thực thi file EXE này.
```

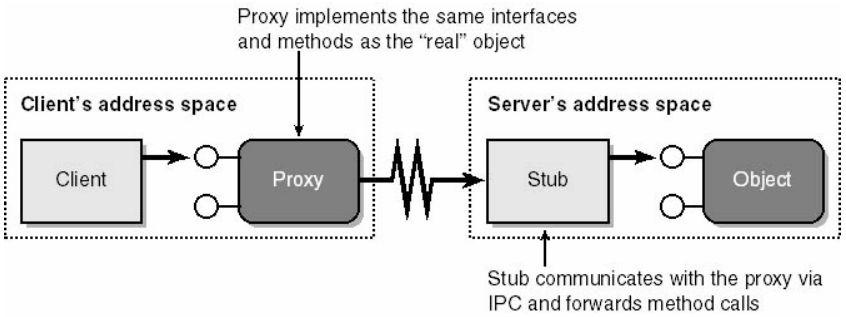
Có 2 kiểu COM-server là:

- **in-process**: là COM có dạng file là DLL, khi hoạt động thì được nạp (cất/lưu) trong cùng không gian địa chỉ của ứng dụng client → tối ưu hơn trong hiện thực.
- **out-of-process**: là COM có dạng file là EXE, khi hoạt động thì được nạp vào một không gian địa chỉ khác với ứng dụng client → có khuyết điểm là ảnh hưởng đến tốc độ thực thi của ứng dụng.

Mô hình DCOM (*Distributed COM*) được giới thiệu với mô hình dạng out-of-process chạy tự do trên các máy server của môi trường mạng.



Một ưu điểm của COM là tính chất "trong suốt vị trí" (*Location Transparency*), tức là chương trình client không quan tâm đến vị trí của các object của COM, và COM sẽ quản lý tất cả thao tác hỗ trợ việc kết nối này. Việc tương tác này thể hiện qua cấu trúc proxy/stub mà COM đã hỗ trợ sẵn.



5.1.6 MFC và COM

Interface có thể được khai báo kế thừa nhau.

Ví dụ 6:

```
interface IUnknown
{
    virtual HRESULT __stdcall QueryInterface(REFIID riid, void** ppv)= 0;
    virtual ULONG __stdcall AddRef()=0;
    virtual ULONG __stdcall Release()=0;
};
…
interface IMath : public IUnknown
{
    virtual HRESULT __stdcall Add(int a, int b, int* pResult)=0;
    virtual HRESULT __stdcall Subtract(int a, int b, int* pResult)=0;
};
…
class CComClass : public IMath
{
protected:
    long m_lRef;    // Reference count
public:
    CComClass ();
    virtual ~CComClass ();
    // IUnknown methods
    virtual HRESULT __stdcall QueryInterface (REFIID riid, void** ppv);
    virtual ULONG __stdcall AddRef();
    virtual ULONG __stdcall Release();
    // IMath methods
    virtual HRESULT __stdcall Add (int a, int b, int* pResult);
    virtual HRESULT __stdcall Subtract (int a, int b, int* pResult);
};
…
class CComClass : public IMath, public ISpelling
{
protected:
    long m_lRef;    // Reference count
public:
    CComClass ();
    virtual ~CComClass();
    // IUnknown methods
    virtual HRESULT __stdcall QueryInterface(REFIID riid, void** ppv);
    virtual ULONG __stdcall AddRef ();
    virtual ULONG __stdcall Release();
    // IMath methods
    virtual HRESULT __stdcall Add(int a, int b, int* pResult);
```

```
    virtual HRESULT __stdcall Subtract(int a, int b, int* pResult);
    // ISpelling methods
    virtual HRESULT __stdcall CheckSpelling(wchar_t* pString);
};
```

và có thể lấy pointer interface của IMath bởi:

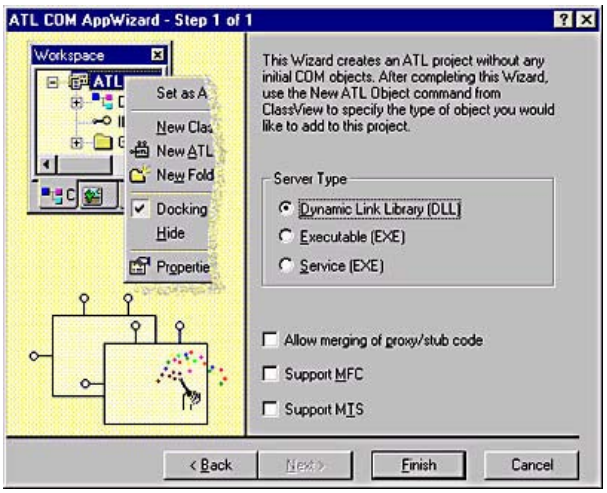
```
*ppv = (IMath*) this;
```

hay:

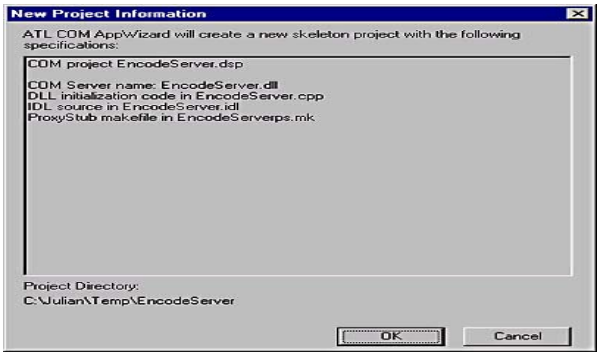
```
*ppv = (ISpelling*) this;
```

Ví dụ tổng hợp:

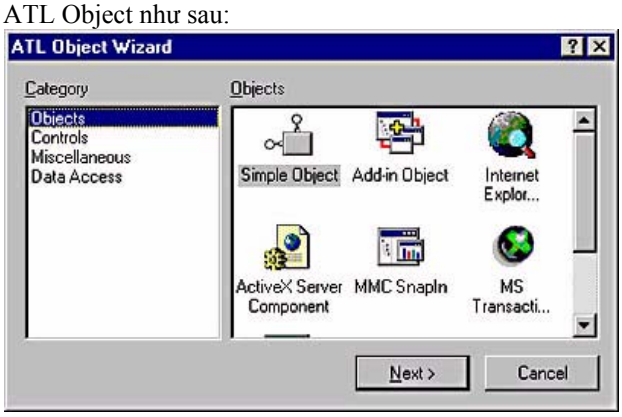
- B1:



và:

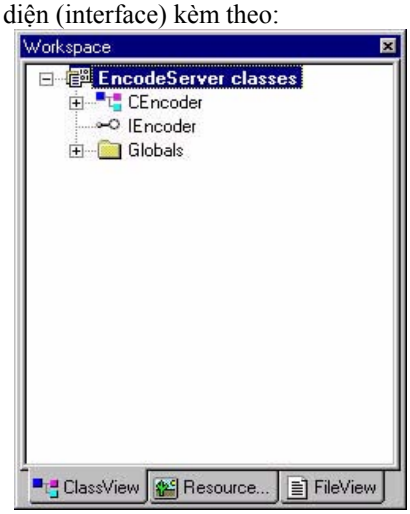


- B2: Chọn tạo mới ATL Object như sau:

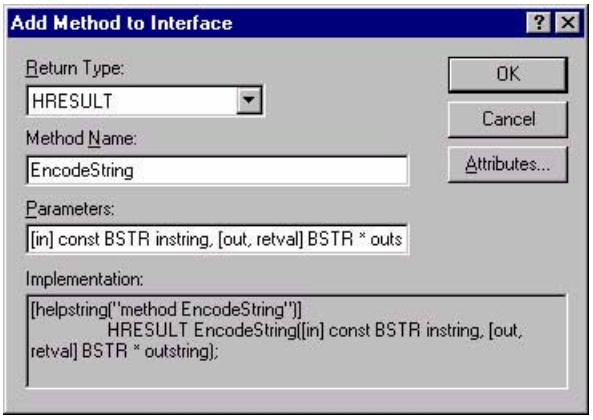


Đối tượng cần tạo có thôntg tin như sau:

Đối tượng được tạo ra bao gồm giao diện (interface) kèm theo:



Chọn để thêm một method vào đối tượng như sau:



trong đó hàm EncodeString cần được cập nhật như sau:

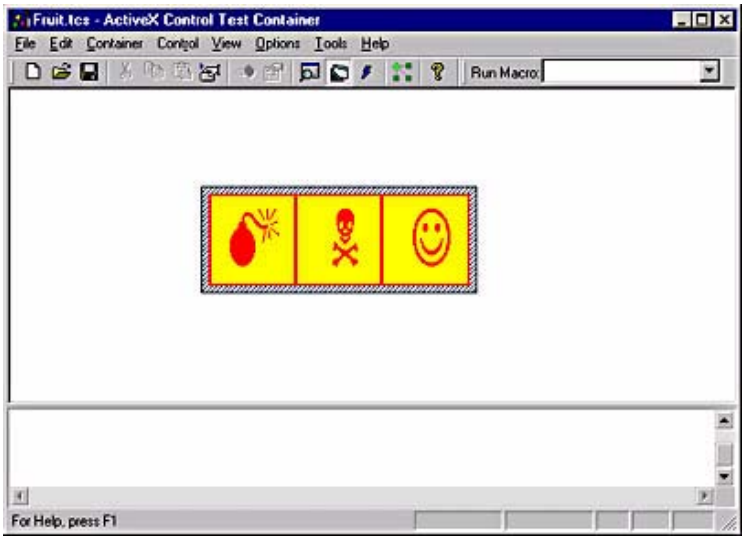
```
STDMETHODIMP CEncoder::EncodeString(const BSTR instring, BSTR *outstring)
{
     BSTR tempstring = ::SysAllocString(instring);
     wcscpy(tempstring, instring);

     for(UINT i = 0; i < ::SysStringLen(tempstring); i++)
          tempstring[i] += m_Key;

     *outstring = ::SysAllocString(tempstring);

     ::SysFreeString(tempstring);

     return S_OK;
}
```

Thực hiện tương tự với **Add Property** để thêm một property *Key* như sau:

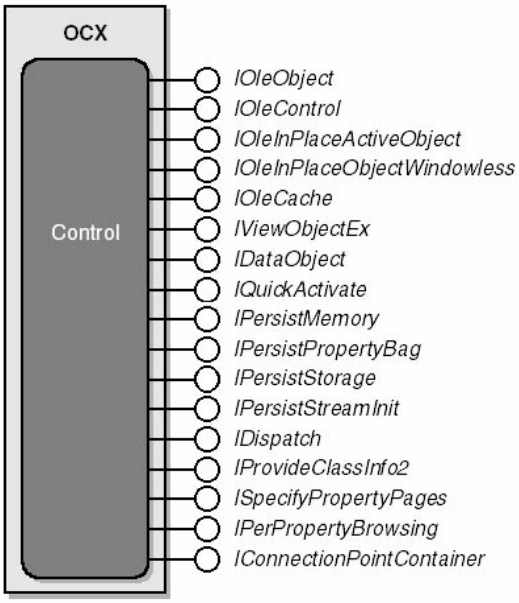
```
STDMETHODIMP CEncoder::get_Key(short *pVal)
{
     *pVal = m_Key;

     return S_OK;
}
STDMETHODIMP CEncoder::put_Key(short newVal)
{
     newVal = newVal > 5 ? 5 : newVal;
     newVal = newVal < -5 ? -5 : newVal;
     m_Key = newVal;

     return S_OK;
}
```

5.2 ACTIVE X CONTROL

ActiveX Control là một điều khiển được xây dựng sẵn hoặc được phát triển theo ý muốn người lập trình, ví dụ như ActiveX control sau đây:



Cấu trúc của một ActiveX Control như sau:

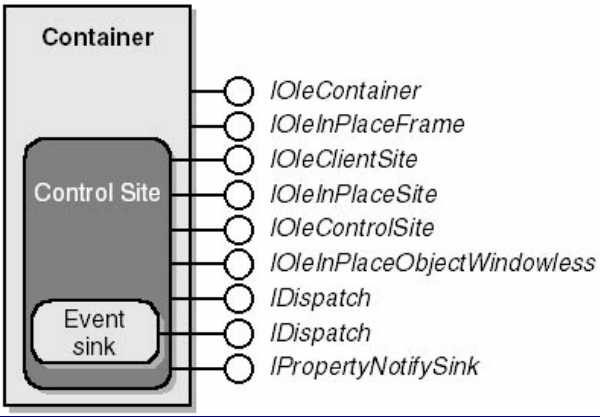


trong đó các giao diện thành phần như sau:

Interface	Comments
IConnectionPointContainer	Exposes connection points for event interfaces
IDataObject	Makes presentation data available to the control container
IDispatch	Exposes the control's methods and properties
IOleCache	Controls the presentation data cache
IOleControl	Base interface for ActiveX controls
IOleInPlaceActiveObject	Base interface for embedded objects that support in-place activation
IOleInPlaceObjectWindowless	Allows the container to manage the activation and deactivation of both windowed and windowless controls
IOleObject	Base interface for embedded objects
IQuickActivate	Speeds control creation in containers that recognize this interface
IPerPropertyBrowsing	Allows containers to acquire information about control properties, such as each property's name
IPersistMemory	Allows the control to write property values to memory and read them back
IPersistPropertyBag	Allows the control to save property values in "property bag" objects provided by the container
IPersistStorage	Allows the control to save property values in storage objects
IPersistStreamInit	Allows the control to save property values in stream objects

IProvideClassInfo2	Makes type information available to the control container
ISpecifyPropertyPages	Allows the control to add pages to property sheets displayed by the container
IViewObjectEx	Allows the container to acquire images of inactive controls and paint windowless controls

Tổ chức lớp chứa và sự kiện như sau:



Interface	Comments
IOleContainer	Base interface for embedding containers
IOleInPlaceFrame	Base interface for OLE containers that support in-place activation
IOleClientSite	Base interface for OLE containers
IOleInPlaceSite	Base interface for OLE containers that support in-place activation
IOleControlSite	Base interface for ActiveX control sites
IDispatch	Exposes the container's ambient properties
IDispatch	Traps events fired by a control
IPropertyNotifySink	Allows the control to notify the container about property changes and to ask permission before changing them

Các hàm ảo hỗ trợ trong COleControl như sau:

Function	Description
OnDraw	Called to paint the control. Override to add control-specific painting logic.
DoPropExchange	Called to save or load a control's persistent properties. Override to support persistent control properties.

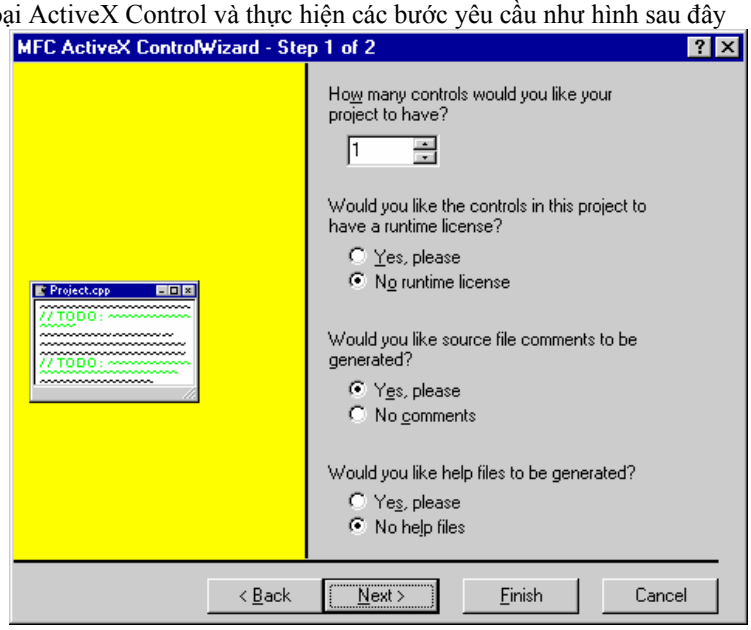
Các hàm hỗ trợ trong lớp COleControl như sau:

Function	Description
Ambientxxx	Retrieves an ambient property value from the container (for example, AmbientBackColor)
Firexxx	Fires a stock event (for example, FireClick)
GetAmbientProperty	Retrieves the values of an ambient property for which no Ambientxxx function is defined
Getxxx	Retrieves the value of a stock property (for example, GetBackColor)
InitializeIIDs	Makes the IDs of the control's event interface and IDispatch interface known to MFC; normally called from the class constructor
InvalidateControl	Repaints the control
SerializeStockProps	Serializes the control's stock properties
SetModifiedFlag	Marks the control as dirty or not dirty (A "dirty" control is one that contains unsaved property changes.)
SetNotSupported	Generates an error when a client attempts to write to a read-only property

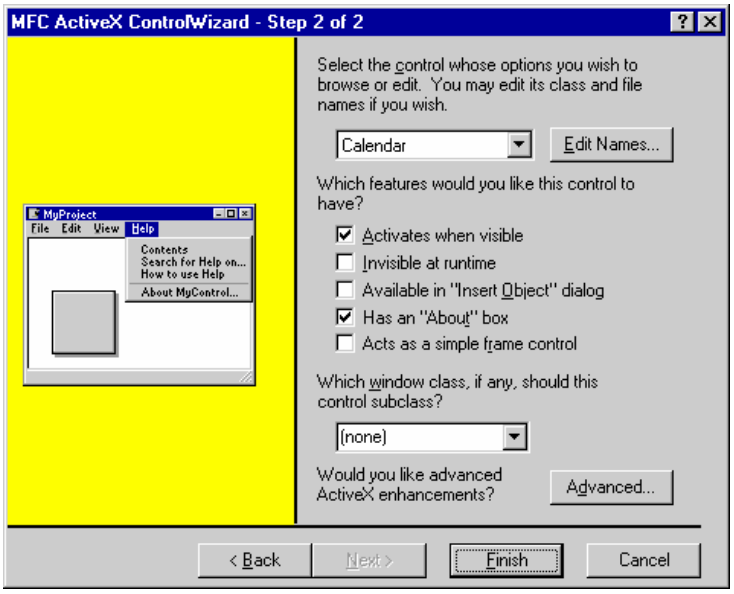
ThrowError	Signals that an error occurred; used in method implementations and property accessor functions
TranslateColor	Translates an OLE_COLOR value into a COLORREF value

Thao tác tạo ActiveX Control:

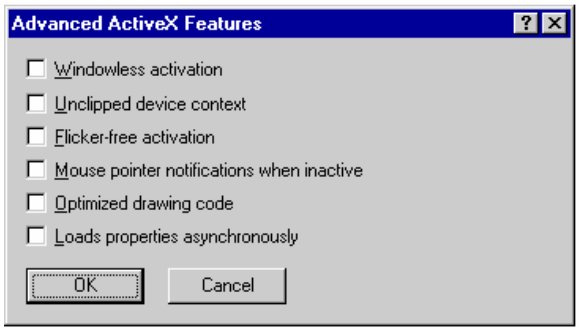
- B1: Chọn loại ActiveX Control và thực hiện các bước yêu cầu như hình sau đây



và:



và:



Ví dụ tổng hợp:

CalendarCtl.h

```
#if !defined(
    AFX_CALENDARCTL_H__68932D29_CFE2_11D2_9282_00C04F8ECF0C__INCLUDED_)
#define AFX_CALENDARCTL_H__68932D29_CFE2_11D2_9282_00C04F8ECF0C__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// CalendarCtl.h : Declaration of the CCalendarCtrl ActiveX Control class.
////////////////////////////////////////////////////////////////////////////
// CCalendarCtrl : See CalendarCtl.cpp for implementation.
class CCalendarCtrl : public COleControl
{
    DECLARE_DYNCREATE(CCalendarCtrl)
// Constructor
public:
    CCalendarCtrl();
// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CCalendarCtrl)
    public:
    virtual void OnDraw(CDC* pdc, const CRect& rcBounds,
        const CRect& rcInvalid);
    virtual void DoPropExchange(CPropExchange* pPX);
    virtual void OnResetState();
    //}}AFX_VIRTUAL

// Implementation
protected:
    BOOL LeapYear(int nYear);
    static const int m_nDaysPerMonth[];
    int m_nDay;
    int m_nMonth;
    int m_nYear;
    ~CCalendarCtrl();

    DECLARE_OLECREATE_EX(CCalendarCtrl)    // Class factory and guid
    DECLARE_OLETYPELIB(CCalendarCtrl)      // GetTypeInfo
    DECLARE_PROPPAGEIDS(CCalendarCtrl)     // Property page IDs
    DECLARE_OLECTLTYPE(CCalendarCtrl)      // Type name and misc status

// Message maps
    //{{AFX_MSG(CCalendarCtrl)
    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
```

```
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()

// Dispatch maps
    //{{AFX_DISPATCH(CCalendarCtrl)
    BOOL m_bRedSundays;
    afx_msg void OnRedSundaysChanged();
    afx_msg DATE GetDate();
    afx_msg BOOL SetDate(short nYear, short nMonth, short nDay);
    //}}AFX_DISPATCH
    DECLARE_DISPATCH_MAP()

    afx_msg void AboutBox();

// Event maps
    //{{AFX_EVENT(CCalendarCtrl)
    void FireNewDay(short nDay)
        {FireEvent(eventidNewDay,EVENT_PARAM(VTS_I2), nDay);}
    //}}AFX_EVENT
    DECLARE_EVENT_MAP()

// Dispatch and event IDs
public:
    enum {
    //{{AFX_DISP_ID(CCalendarCtrl)
    dispidRedSundays = 1L,
    dispidGetDate = 2L,
    dispidSetDate = 3L,
    eventidNewDay = 1L,
    //}}AFX_DISP_ID
    };
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations
// immediately before the previous line.
#endif
// !defined(
//     AFX_CALENDARCTL_H__68932D29_CFE2_11D2_9282_00C04F8ECF0C__INCLUDED)
```

CalendarCtl.cpp

```
// CalendarCtl.cpp : Implementation of the
// CCalendarCtrl ActiveX Control class.

#include "stdafx.h"
#include "Calendar.h"
#include "CalendarCtl.h"
#include "CalendarPpg.h"

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

IMPLEMENT_DYNCREATE(CCalendarCtrl, COleControl)
```

```
const int CCalendarCtrl::m_nDaysPerMonth[] = {
    31,         // January
    28,         // February
    31,         // March
    30,         // April
    31,         // May
    30,         // June
    31,         // July
    31,         // August
    30,         // September
    31,         // October
    30,         // November
    31,         // December
};
////////////////////////////////////////////////////////////////////////////
// Message map
BEGIN_MESSAGE_MAP(CCalendarCtrl, COleControl)
    //{{AFX_MSG_MAP(CCalendarCtrl)
    ON_WM_LBUTTONDOWN()
    //}}AFX_MSG_MAP
    ON_OLEVERB(AFX_IDS_VERB_PROPERTIES, OnProperties)
END_MESSAGE_MAP()
////////////////////////////////////////////////////////////////////////////
// Dispatch map
BEGIN_DISPATCH_MAP(CCalendarCtrl, COleControl)
    //{{AFX_DISPATCH_MAP(CCalendarCtrl)
    DISP_PROPERTY_NOTIFY(CCalendarCtrl, "RedSundays", m_bRedSundays,
        OnRedSundaysChanged, VT_BOOL)
    DISP_FUNCTION(CCalendarCtrl, "GetDate", GetDate, VT_DATE, VTS_NONE)
    DISP_FUNCTION(CCalendarCtrl, "SetDate", SetDate, VT_BOOL,
        VTS_I2 VTS_I2 VTS_I2)
    DISP_STOCKPROP_BACKCOLOR()
    //}}AFX_DISPATCH_MAP
    DISP_FUNCTION_ID(CCalendarCtrl, "AboutBox", DISPID_ABOUTBOX,
        AboutBox, VT_EMPTY, VTS_NONE)
END_DISPATCH_MAP()
////////////////////////////////////////////////////////////////////////////
// Event map
BEGIN_EVENT_MAP(CCalendarCtrl, COleControl)
    //{{AFX_EVENT_MAP(CCalendarCtrl)
    EVENT_CUSTOM("NewDay", FireNewDay, VTS_I2)
    //}}AFX_EVENT_MAP
END_EVENT_MAP()
////////////////////////////////////////////////////////////////////////////
// Property pages
// TODO: Add more property pages as needed.
// Remember to increase the count!
BEGIN_PROPPAGEIDS(CCalendarCtrl, 2)
    PROPPAGEID(CCalendarPropPage::guid)
    PROPPAGEID (CLSID_CColorPropPage)
END_PROPPAGEIDS(CCalendarCtrl)

////////////////////////////////////////////////////////////////////////////
// Initialize class factory and guid
IMPLEMENT_OLECREATE_EX(CCalendarCtrl, "CALENDAR.CalendarCtrl.1",
```

```
    0xed780d6b, 0xcc9f, 0x11d2, 0x92, 0x82, 0, 0xc0, 0x4f, 0x8e, 0xcf, 0xc)

////////////////////////////////////////////////////////////////////////////
// Type library ID and version
IMPLEMENT_OLETYPELIB(CCalendarCtrl, _tlid, _wVerMajor, _wVerMinor)
////////////////////////////////////////////////////////////////////////////
// Interface IDs
const IID BASED_CODE IID_DCalendar =
        { 0x68932d1a, 0xcfe2, 0x11d2,
            { 0x92, 0x82, 0, 0xc0, 0x4f, 0x8e, 0xcf, 0xc } };
const IID BASED_CODE IID_DCalendarEvents =
        { 0x68932d1b, 0xcfe2, 0x11d2,
            { 0x92, 0x82, 0, 0xc0, 0x4f, 0x8e, 0xcf, 0xc } };
////////////////////////////////////////////////////////////////////////////
// Control type information
static const DWORD BASED_CODE _dwCalendarOleMisc =
    OLEMISC_ACTIVATEWHENVISIBLE œ
    OLEMISC_SETCLIENTSITEFIRST œ
    OLEMISC_INSIDEOUT œ
    OLEMISC_CANTLINKINSIDE œ
    OLEMISC_RECOMPOSEONRESIZE;
IMPLEMENT_OLECTLTYPE(CCalendarCtrl, IDS_CALENDAR, _dwCalendarOleMisc)
////////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::CCalendarCtrlFactory::UpdateRegistry -
// Adds or removes system registry entries for CCalendarCtrl
BOOL CCalendarCtrl::CCalendarCtrlFactory::UpdateRegistry(BOOL bRegister)
{
    // TODO: Verify that your control follows apartment-model
    // threading rules. Refer to MFC TechNote 64 for more information.
    // If your control does not conform to the apartment-model rules, then
    // you must modify the code below, changing the 6th parameter from
    // afxRegApartmentThreading to 0.
    if (bRegister)
        return AfxOleRegisterControlClass(
            AfxGetInstanceHandle(),
            m_clsid,
            m_lpszProgID,
            IDS_CALENDAR,
            IDB_CALENDAR,
            afxRegApartmentThreading,
            _dwCalendarOleMisc,
            _tlid,
            _wVerMajor,
            _wVerMinor);
    else
        return AfxOleUnregisterClass(m_clsid, m_lpszProgID);
}
////////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::CCalendarCtrl - Constructor
CCalendarCtrl::CCalendarCtrl()
{
    InitializeIIDs(&IID_DCalendar, &IID_DCalendarEvents);

    CTime time = CTime::GetCurrentTime ();
    m_nYear = time.GetYear ();
```

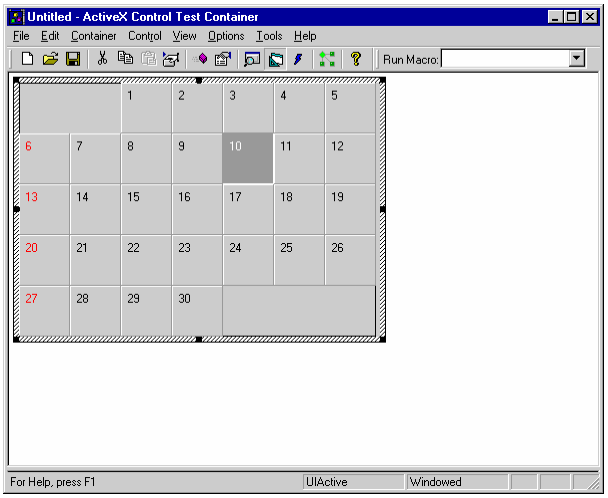
```
    m_nMonth = time.GetMonth ();
    m_nDay = time.GetDay ();
}
////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::~CCalendarCtrl - Destructor
CCalendarCtrl::~CCalendarCtrl()
{
    // TODO: Cleanup your control's instance data here.
}
////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::OnDraw - Drawing function
void CCalendarCtrl::OnDraw(
            CDC* pdc, const CRect& rcBounds, const CRect& rcInvalid)
{
    //
    // Paint the control's background.
    //
    CBrush brush (TranslateColor (GetBackColor ()));
    pdc->FillRect (rcBounds, &brush);
    //
    // Compute the number of days in the month, which day of the week
    // the first of the month falls on, and other information needed to
    // draw the calendar.
    //
    int nNumberOfDays = m_nDaysPerMonth[m_nMonth - 1];
    if (m_nMonth == 2 && LeapYear (m_nYear))
        nNumberOfDays++;
    CTime time (m_nYear, m_nMonth, 1, 12, 0, 0);
    int nFirstDayOfMonth = time.GetDayOfWeek ();
    int nNumberOfRows = (nNumberOfDays + nFirstDayOfMonth + 5) / 7;

    int nCellWidth = rcBounds.Width () / 7;
    int nCellHeight = rcBounds.Height () / nNumberOfRows;

    int cx = rcBounds.left;
    int cy = rcBounds.top;
    //
    // Draw the calendar rectangle.
    //
    CPen* pOldPen = (CPen*) pdc->SelectStockObject (BLACK_PEN);
    CBrush* pOldBrush = (CBrush*) pdc->SelectStockObject (NULL_BRUSH);

    pdc->Rectangle (rcBounds.left, rcBounds.top,
        rcBounds.left + (7 * nCellWidth),
        rcBounds.top + (nNumberOfRows * nCellHeight));
    //
    // Draw rectangles representing the days of the month.
    //
    CFont font;
    font.CreatePointFont (80, _T ("MS Sans Serif"));
    CFont* pOldFont = pdc->SelectObject (&font);

    COLORREF clrOldTextColor = pdc->SetTextColor (RGB (0, 0, 0));
    int nOldBkMode = pdc->SetBkMode (TRANSPARENT);
```

```
    for (int i=0; i<nNumberOfDays; i++) {
        int nGridIndex = i + nFirstDayOfMonth - 1;
        int x = ((nGridIndex % 7) * nCellWidth) + cx;
        int y = ((nGridIndex / 7) * nCellHeight) + cy;
        CRect rect (x, y, x + nCellWidth, y + nCellHeight);

        if (i != m_nDay - 1) {
            pdc->Draw3dRect (rect, RGB (255, 255, 255),
                RGB (128, 128, 128));
            pdc->SetTextColor (RGB (0, 0, 0));
        }
        else {
            pdc->SelectStockObject (NULL_PEN);
            pdc->SelectStockObject (GRAY_BRUSH);
            pdc->Rectangle (rect);
            pdc->Draw3dRect (rect, RGB (128, 128, 128),
                RGB (255, 255, 255));
            pdc->SetTextColor (RGB (255, 255, 255));
        }
        CString string;
        string.Format (_T ("%d"), i + 1);
        rect.DeflateRect (nCellWidth / 8, nCellHeight / 8);

        if (m_bRedSundays && nGridIndex % 7 == 0)
            pdc->SetTextColor (RGB (255, 0, 0));

        pdc->DrawText (string, rect, DT_SINGLELINE œ DT_LEFT œ DT_TOP);
    }
    //
    // Clean up and exit.
    //
    pdc->SetBkMode (nOldBkMode);
    pdc->SetTextColor (clrOldTextColor);
    pdc->SelectObject (pOldFont);
    pdc->SelectObject (pOldBrush);
    pdc->SelectObject (pOldPen);
}
////////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::DoPropExchange - Persistence support
void CCalendarCtrl::DoPropExchange(CPropExchange* pPX)
{
    ExchangeVersion(pPX, MAKELONG(_wVerMinor, _wVerMajor));
    COleControl::DoPropExchange(pPX);
    PX_Bool (pPX, _T ("RedSundays"), m_bRedSundays, TRUE);
}
////////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::OnResetState - Reset control to default state
void CCalendarCtrl::OnResetState()
{
    COleControl::OnResetState(); // Resets defaults found in DoPropExchange

    // TODO: Reset any other control state here.
}
////////////////////////////////////////////////////////////////////////////
// CCalendarCtrl::AboutBox - Display an "About" box to the user
```

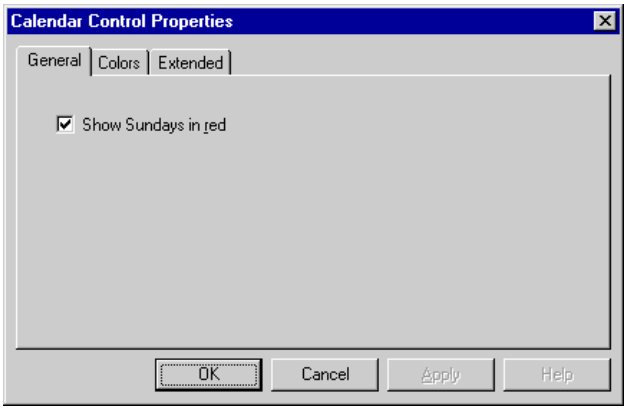
```
void CCalendarCtrl::AboutBox()
{
    CDialog dlgAbout(IDD_ABOUTBOX_CALENDAR);
    dlgAbout.DoModal();
}
////////////////////////////////////////////////////////////////////////
// CCalendarCtrl message handlers
BOOL CCalendarCtrl::LeapYear(int nYear)
{
    return (nYear % 4 == 0) ^ (nYear % 400 == 0) ^ (nYear % 100 == 0);
}
void CCalendarCtrl::OnRedSundaysChanged()
{
    InvalidateControl ();
    SetModifiedFlag();
}
DATE CCalendarCtrl::GetDate()
{
    COleDateTime date (m_nYear, m_nMonth, m_nDay, 12, 0, 0);
    return (DATE) date;
}
BOOL CCalendarCtrl::SetDate(short nYear, short nMonth, short nDay)
{
    //
    // Make sure the input date is valid.
    //
    if (nYear < 1970 œœ nYear > 2037)
        return FALSE;

    if (nMonth < 1 œœ nMonth > 12)
        return FALSE;

    int nNumberOfDays = m_nDaysPerMonth[m_nMonth - 1];
    if (nMonth == 2 && LeapYear (nYear))
        nNumberOfDays++;

    if (nDay < 1 œœ nDay > nNumberOfDays)
        return FALSE;
    //
    // Update the date, repaint the control, and fire a NewDay event.
    //
    m_nYear = nYear;
    m_nMonth = nMonth;
    m_nDay = nDay;
    InvalidateControl ();
    return TRUE;
}
void CCalendarCtrl::OnLButtonDown(UINT nFlags, CPoint point)
{
    int nNumberOfDays = m_nDaysPerMonth[m_nMonth - 1];
    if (m_nMonth == 2 && LeapYear (m_nYear))
        nNumberOfDays++;

    CTime time (m_nYear, m_nMonth, 1, 12, 0, 0);
    int nFirstDayOfMonth = time.GetDayOfWeek ();
```

```
    int nNumberOfRows = (nNumberOfDays + nFirstDayOfMonth + 5) / 7;

    CRect rcClient;
    GetClientRect (&rcClient);
    int nCellWidth = rcClient.Width () / 7;
    int nCellHeight = rcClient.Height () / nNumberOfRows;

    for (int i=0; i<nNumberOfDays; i++) {
        int nGridIndex = i + nFirstDayOfMonth - 1;
        int x = rcClient.left + (nGridIndex % 7) * nCellWidth;
        int y = rcClient.top + (nGridIndex / 7) * nCellHeight;
        CRect rect (x, y, x + nCellWidth, y + nCellHeight);

        if (rect.PtInRect (point)) {
            m_nDay = i + 1;
            FireNewDay (m_nDay);
            InvalidateControl ();
        }
    }
    COleControl::OnLButtonDown(nFlags, point);
}
```

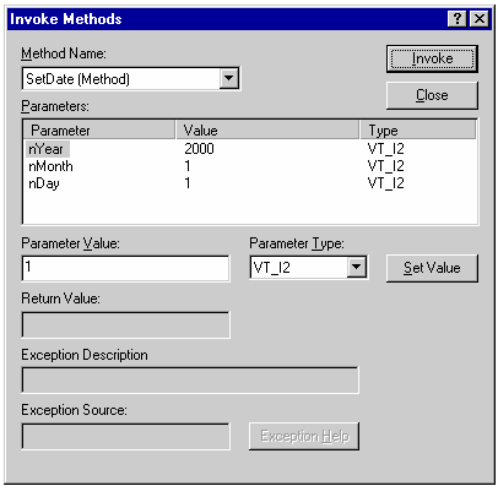
Màn hình kết quả:



và:



và:



Đăng ký OCX:

```
Regsvr32 Calendar.ocx
```

Hủy đăng ký:

```
Regsvr32 /U Calendar.ocx
```

CHƯƠNG 6. BÀI THỰC HÀNH

6.1 Bài thực hành 1

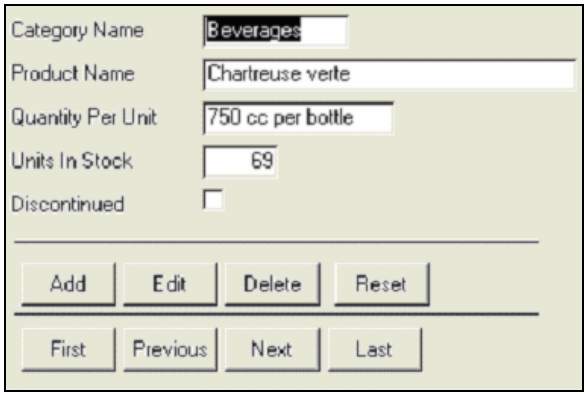
- Yêu cầu: Tìm hiểu môi trường làm việc VC++ và tạo ứng dụng đơn giản.
  a. Tìm hiểu các thành phần của Visual C++.
  b. Dùng MFC App Framework tạo chương trình ứng dụng theo các kiểu: Dialog (tên ExDlg), SDI (tên ExSDI) và MDI (tên ExMDI) (*ExSDI và ExMDI có lớp view cơ sở ở bước 6 là CFormView*).
  c. Lần lượt biên dịch ExDlg, ExSDI và ExMDI với chế độ Release và Debug, quan sát file EXE kết quả (về kích thước) và cho nhận xét.
  d. Mở ứng dụ ExSDI, thêm 1 button (đặt ID là ***IDC_BUTTON_RED*** và caption là ***Mau Do***) sao cho khi bấm vào button này, chương trình sẽ hiện ra thông báo "Nut Mau Do" bởi việc gọi hàm AfxMessageBox.
  e. Làm tương tự(d) cho button ***Mau Xanh***.
  f. Thêm 1 hàm tên ***MyCommonFuntion(BOOL bType)*** kiểu ***void*** để sử dụng chung cho câu (d) và (e) cùng gọi hàm này để thực hiện các công việc (nhằm tránh việc trùng lắp code ở câu d và e)
  g. Thêm 1 editbox (đặt ID là IDC_EDIT_TEXT) và tạo 1 biến liên kết với editbox này. [*HD: Bấm vào menu View → ClassWizard →Member Variables, sau đó chọn ID của editbox này và nhấn button "Add Variable"*] có tên là ***m_szMessage***.
  h. Sửa lại hàm ở (f) sao cho khi bấm vào nút ***Mau Do*** (hay **Mau Xanh**) thông báo hiện lên có dạng "Ban nhap noi dung: xxx, va bam nut Mau Do" hay "Ban nhap noi dung: xxx, va bam nut Mau Xanh" (*với xxx là nội dung có trong editbox ở câu g do người dùng nhập vào)*
  i. Thêm 1 static textbox(đặt ID là IDC_STATIC_MSGSHOW), tạo 1 biến liên kết với static textbox này [làm tương tự như(g)] có tên là m_szMsgShow.
  j. Xử lý tương tự như (h) nhưng không hiện thông báo mà sẽ hiện nội dung này trong static textbox vừa thêm ở (i).

6.2 Bài thực hành 2

- Yêu cầu: Xử lý xuất nội dung, vẽ hình, xử lý thao tác từ bàn phím, chuột và xuất nhập với file.
  a. Tạo mới ứng dụng ExSDI có lớp view cơ sở ở bước 6 là CView
  b. Xuất 1 chuỗi "MSSV: xxx, TENSV: yyy" tại toạ độ (10,10) và vẽ 1 hình elip tuỳ ý không có màu nền *(HD: thao tác trong hàm OnDraw trong lớp View)*
  c. Xử lý vẽ hình con lật đật đứng yên có kích thước và màu nền tuỳ ý tại toạ độ (200, 200)
  d. Xử lý vẽ con lật đật tại vị trí bấm chuột phải bất kỳ (trong vùng client của cửa sổ).
  e. Mở rộng câu d bằng ách xử lý dời con lật đật theo quỹ đạo hình tròn bán kính 20pixel.
  f. Xử lý vẽ đường tự do khi bấm rê chuột trái và phím Control (trong vùng client của cửa sổ).
  g. Hiển thị nội dung ký tự (bao gồm mẫu tự, ký số và các ký hiệu hiển thị được) ở góc phải trên trong vùng client của cửa sổ.
  h. Lập lại tương tự các câu từ (c) đến (e) cho ngôi sao 6 cánh.

6.3 Bài thực hành 3

- Yêu cầu: Tạo và sử dụng các điều khiển Windows và sử dụng hộp thoại (File, Color, Font, Print/Print ...)
  a. Tạo mới ứng dụng ExDlg có lớp view cơ sở ở bước 6 là CFormView
  b. Tạo 1 menu có item "Font…", "Color…", khi nhấn vào lần lượt hiện ra các dialog cho chọn font chữ, màu.
  c. Cập nhật giao diện thành dạng như sau:



d. Tìm hiểu các thành phần View/Document của project ExSDI.

e. Kết hợp với sự hỗ trợ của các lớp CArray, hãy xử lý sao cho khi nhập thông tin và nhấn button "Add" thì thông tin này sẽ được lưu vào bộ nhớ, tương tự cho phép cập nhật, xóa hay reset khi nhấn vào các button Edit/Delete hay Reset.

f. Xử lý tương tự với các button First/Previous/Next/Last cho phép hiển thị các mẫu tin mong muốn.

g. Thêm 1 editbox (đặt ID là IDC_EDIT_NUMBER)

h. Cập nhật lại chương trình sao cho khi bấm di chuyển đến mẫu tin nào thì số thứ tự mẫu tin đó sẽ hiện ra tương ứng trong editbox ở câu g.

i. Lập lại các bước (e) đến (f) với CList.

6.4 Bài thực hành 4

- Yêu cầu: Phát triển từ các Window App ở bài TH3 nêu trên: mở rộng sử dụng các View class, xử lý truyền/nhận giữa Document và View, bổ sung/điều khiển Toolbar, StatusBar.

a. Mở bài thực hành số 3 (ExSDI), sau đó thêm 1 button "Save File"(đặt ID là IDC_BUTTON_SAVEFILE), và cập nhật giao diện lại giống như hình bên dưới đây với combo-box có ID là IDC_COMBOX_CATEGORY

b. Thực hiện thao tác lưu danh sách Product đang có vào file có tên "ProductList.dat" khi bấm nút "Save File"

c. Thực hiện thao tác lưu nội dung đang có trong file "ProductList.dat" vào danh sách Product khi bấm nút "Load File"

d. Thực hiện việc bẫy lỗi với cơ chế ***try…catch***, xuất thông báo lỗi(trong trường hợp có lỗi), xuất thông báo "Du lieu duoc luu xong" (trong trường hợp Add/Edit), xuất thông báo "Ban muon xoa du lieu khong?" (trong trường hợp vừa bấm Delete) và xuất thông báo "Du lieu duoc xoa xong" (sau khi thực hiện Delete xong)

e. Tạo Toolbar (nếu chưa có) hay thêm vào Toolbar hiện hành các biểu tượng tương ứng với các button đã có (Add/Edit/Delete/Save File,First/Previous/Next/Last) sao cho chúng thực thi tương tự như khi bấm các button đã có này.

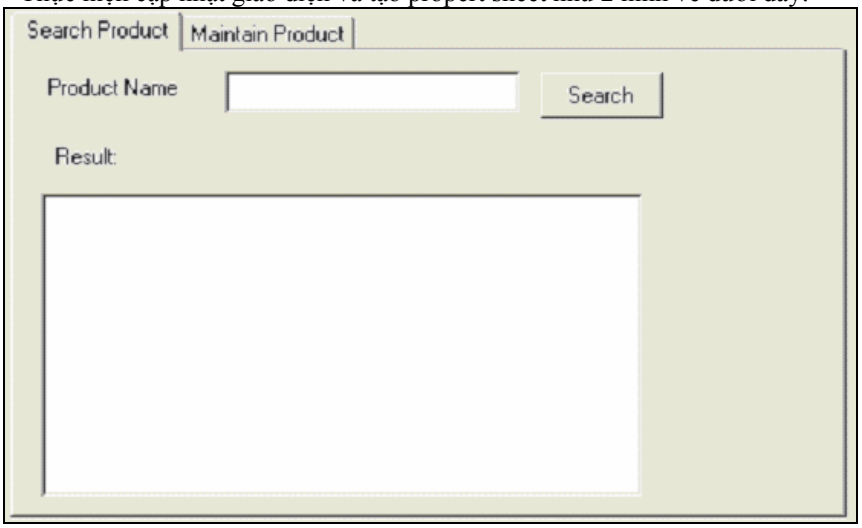
f. Thực hiện tương tự câu (c) nhưng hiện thông báo ra StatusBar phía dưới.

6.5 Bài thực hành 5

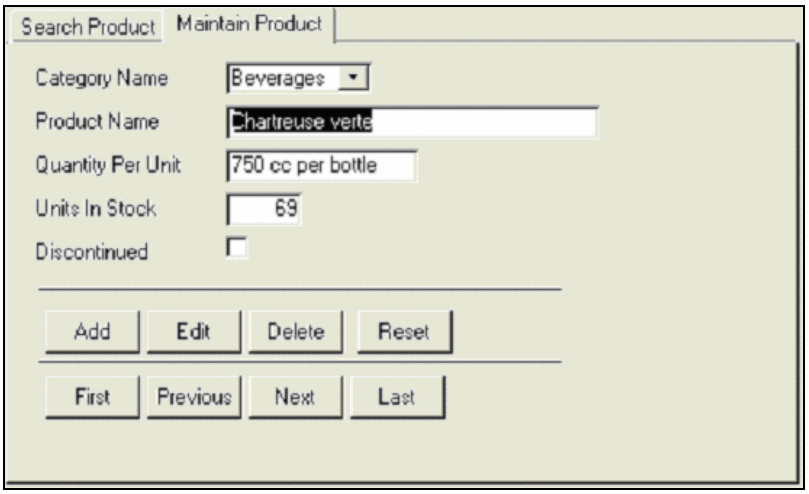
- Yêu cầu: Phát triển từ các Window App ở bài TH1 nêu trên: xây dựng hệ thống đồng hồ theo các địa điểm (timer/idle), và xử lý song hành (threads).
    a. Tạo mới ứng dụng ExSDI có lớp view cơ sở ở bước 6 là CFormView. Tạo giao diện tương tự đồng hồ điện tử với xử lý hiện thời gian trên giao diện và cả StatusBar (tương tự như đồng hồ điện tử ở taskbar)
    b. Thực hiện chương trình vẽ con lật đật tự do – có lắc lư - (dựa theo ví dụ trong phần Timer) tại vị trí bất ký, màu bất kỳ, kích thước bất kỳ với yêu cầu cung cấp 1 menu cho phép khởi tạo bộ định thời (*timer*), ngừng bộ định thời và có thể nhập 1 chu kỳ thời gian (*theo đơn vị millisecond*) tùy ý.
    c. Tìm hiểu chương trình mẫu Chat Client/Server.

6.6 Bài thực hành 6

- Yêu cầu: Xây dựng chương trình quản lý dữ liệu với Northwind database dùng ADO.
    a. Tạo mới ứng dụng ExDlg có lớp view cơ sở ở bước 6 là CFormView
    b. Thực hiện cập nhật giao diện và tạo propert sheet như 2 hình vẽ dưới đây:



trong đó editbox có ID là IDC_EDIT_ SEARCH, button có ID là IDC_BUTTON_SEARCH và combox-box có ID là IDC_LIST_ SEARCH.



và combo-box có ID là IDC_COMBO_CATEGORY

    c. Truy cập cơ sở dữ liệu Northwind (dạng Access hay SQL Server nếu có) để:
        - Trong property page (tab) "Maintain Product", thực hiện nạp dữ liệu từ bảng Categories vào combo-box(dropdown listbox) Category Name sao cho CategoryID được lưu ngầm và CategoryName hiện lên trong combo-box
        - Trong property page (tab) "Maintain Product", thực hiện tương tự câu (e) và (f) của bài 3 nhưng truy xuất trực tiếp cơ sở dữ liệu.

- Trong property page (tab) "Search Product", thực hiện việc tìm kiếm khi bấm button Search và xuất kết quả ra danh sách Result bên dưới.

6.7 Bài thực hành 7

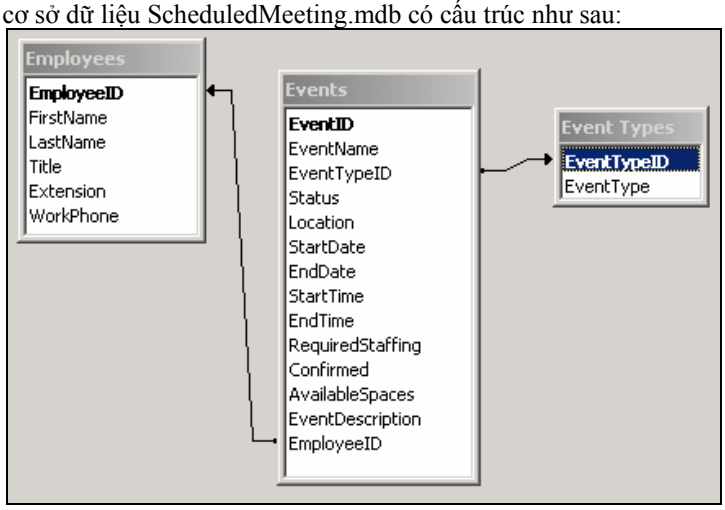
- Yêu cầu: Tạo COM object liên kết trong việc truy xuất dữ liệu.
  a. Tạo mới project có tên ExCOM và thực hiện xây dựng các hàm thư viện hỗ trợ truy cập cơ sở dữ liệu Northwind như thêm/cập nhật/xoá pProduct, tìm kiếm/lấy danh sách Product, lấy danh sách Category.
  b. Mở bài thực hành số 6 (ExSDI), thực hiện liên kết với xxxExCOM và thay thế các thao tác(i), (ii) và (iii) bởi các hàm thư viện ở câu (a).

6.8 Bài thực hành 8

- Yêu cầu: Phát triển từ các Window App ở bài TH6/TH7 nêu trên: xây dựng ActiveX control hỗ trợ các thành phần của chương trình.
  a. Xem chương trình mẫu về ActiveXControl
  b. Tạo ActiveX Control về máy tính cá nhân
  c. Xem chương trình mẫu về Windows NT Service.
  d. Tạo Windows NT Service về thông báo nhắc nhở thời gian 1000ms

6.9 Bài thực hành 9

- Yêu cầu: bài tập tổng hợp: xây dựng chương trình để thông báo lịch họp.
  a. Tạo cơ sở dữ liệu ScheduledMeeting.mdb có cấu trúc như sau:



  b. Xây dựng chương trình quản lý nội dung lịch họp của nhân viên trong 1 cơ quan (không dùng COM) có giao diện tùy chọn (*giao diện xây dựng tùy ý*)

6.10 Bài thực hành 10

- Yêu cầu: bài tập tổng hợp: xây dựng chương trình để thông báo lịch họp.
  a. Phát triển từ bài 9 nhưng dùng COM làm thư việc truy xuất dữ liệu

CHƯƠNG 7. BÀI TẬP TỔNG HỢP

7.1 Bài 1

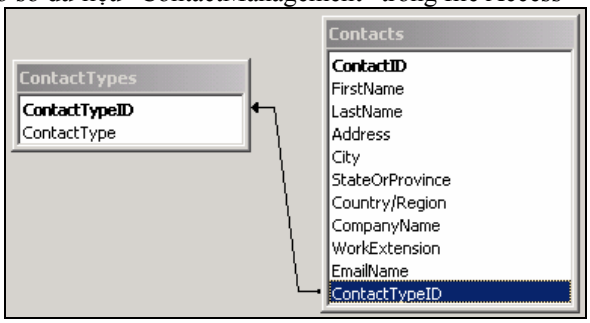
Thực hiện chương trình tương tự như ứng dụng Windows Explorer.

7.2 Bài 2

Thực hiện chương trình tương tự như ứng dụng Microsoft Internet Explorer.

7.3 Bài 3

a. Xem cấu trúc của cơ sở dữ liệu "ContactManagement" trong file Access



b. Xây dựng chương trình ứng dụng dạng SDI trong đó có menu Tools bao gồm các lệnh (menu item) theo cấu trúc sau sau:

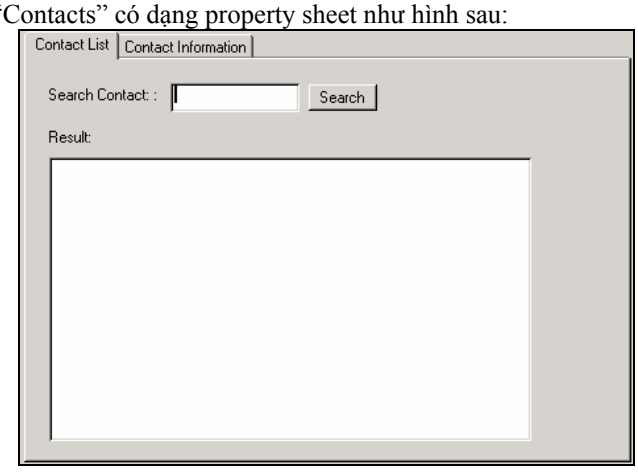
Tools

|- "Enter-View Contacts"

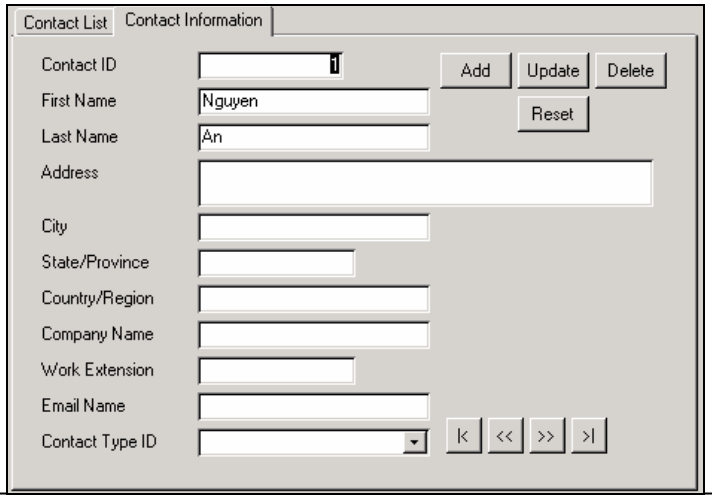
|- "Enter-View Contact Types"

sao cho khi bấm vào từng lệnh sẽ mở các form tương ứng (màn hình như ở câu 3 và 5)

c. Xây dựng form "Contacts" có dạng property sheet như hình sau:

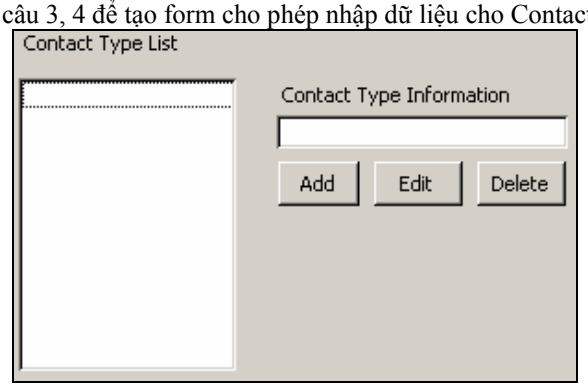


và



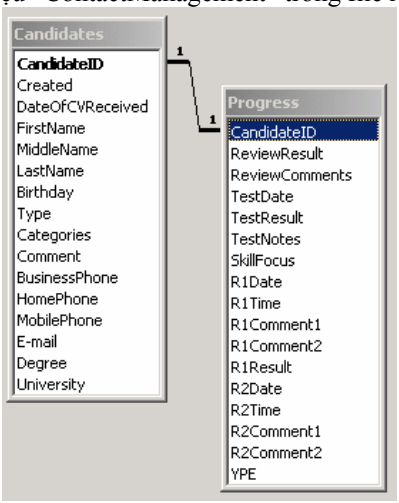
d. Thực hiện truy xuất cơ sở dữ liệu "ContactManagement" và làm các thao tác sau:
   i) Cho phép tìm kiếm dữ liệu Contact
   ii) Cho phép xem thông tin của Contact
   iii) Cho phép thêm, cập nhập, xoá contact. Cho phép reset màn hình. Cho phép thực hiện di chuyển first, last, next, previous để xem thông tin Contact.

e. Thực hiện tương tự câu 3, 4 để tạo form cho phép nhập dữ liệu cho ContactType với màn hình như sau:



7.4 Bài 4

a. Xem cấu trúc của cơ sở dữ liệu "ContactManagement" trong file Access



b. Xây dựng chương trình ứng dụng dạng SDI trong đó có menu Tools bao gồm các lệnh (menu item) theo cấu trúc sau sau:

Tools
|- "Enter-View Candidates"
|- "Enter-View Interview Progress"

c. Xây dựng chương trình quản lý các ứng viên và quá trình phỏng vấn tuyển dụng của công ty ABC (dựa theo bài tổng hợp số 3).

7.5 Bài 5

Viết chương trình MyPainBrush, tương tự như ứng dụng PainBrush của Windows.

7.6 Bài 6

Viết chương trình MyGraph hỗ trợ vẽ đồ thị của một đường bậc nhất (dạng y=ax+b) hay bậc hai (dạng $y=ax^2+bx+c$), dạng y=(ax+b)/(cx+d), dạng $y=(ax^2+bx+c)/(dx+e)$ với các giá trị a, b, c, ... do người dùng nhập vào.

7.7 Bài 7

Viết chương trình MyPainBrush, tương tự như ứng dụng WordPad của Windows.

CHƯƠNG 0. ÔN TẬP LÝ THUYẾT C/C++ 2

0.1 Ôn tập C 2
0.1.1 Kiểu dữ liệu, biến và chuyển đổi kiểu 2
0.2 Hàm và lời gọi hàm 2
0.2.1 Phát biểu điều khiển 2
0.2.2 Array 2
0.2.3 Pointer 2
0.2.4 File 2
0.2.5 Debug – bẫy lỗi 2
0.3 Ôn tập C++ 2
0.3.1 Class 2
0.3.2 Cấu trúc thừa kế 2
0.3.3 Tầm vực truy xuất 2
0.3.4 Object 2

CHƯƠNG 1. CÁC VẤN ĐỀ CƠ BẢN CỦA ỨNG DỤNG WINDOWS VÀ MFC 3

1.1 GIỚI THIỆU KHUNG ỨNG DỤNG WINDOWS (WINDOWS APPLICATION) VÀ XÂY DỰNG CHƯƠNG TRÌNH MẪU VỚI MFC APP FRAMEWORK 3
1.1.1 Lập trình Windows 3
1.1.2 Mô hình lập trình Windows 3
1.1.3 Lập trình Windows với MFC 5
1.1.4 Môi trường lập trình MS Visual C++ 5
1.1.4.1 Miền làm việc 5
1.1.4.2 Cửa sổ xuất (output pane) 6
1.1.4.3 Vùng soạn thảo 6
1.1.4.4 Thanh thực đơn (menu) 7
1.1.4.5 Thanh công cụ 7
1.1.5 Các thành phần của ứng dụng phát triển với MS Visual C++ 8
1.1.5.1 Đối tượng ứng dụng (Application) 9
1.1.5.2 Đối tượng Khung Cửa sổ (Frame Window) 10
1.1.5.3 Quá trình làm việc của các ánh xạ thông báo (Message Map) 10
1.1.5.4 Windows, Character Sets, và _T Macro 11
1.1.5.5 Hàm UpdateData 11
1.1.6 Tạo ứng dụng với MS Visual C++ 12
1.2 XỬ LÝ VẼ HÌNH TRONG ỨNG DỤNG WINDOWS 16
1.2.1 Vấn đề quan tâm 16
1.2.2 Giới thiệu 16
1.2.3 Truy xuất ngữ cảnh thiết bị 17
1.2.3.1 Xác định chế độ đo lường 17
1.2.4 Thao tác vẽ với bút vẽ 18
1.2.5 Thao tác tô màu với cọ vẽ 21
1.2.6 Hiển thị văn bản trong môi trường đồ hoạ 21
1.2.7 GDI Fonts và lớp CFont 22
1.2.8 Ví dụ tổng hợp 23
1.2.8.1 Chương trình 1 23
1.2.8.2 Chương trình 2 29
1.3 XỬ LÝ BÀN PHÍM/CHUỘT TRONG ỨNG DỤNG WINDOWS 30
1.3.1 Vấn đề quan tâm 30
1.3.2 Các sự kiện của chuột 31
1.3.3 Các sự kiện của bàn phím 32
1.4 CÁC LỚP MFC COLLECTION: ARRAY, LIST, MAP*, TYPE POINTER MAP* 34
1.4.1 Vấn đề quan tâm 34
1.4.2 Array collection 34
1.4.3 List collection 35
1.4.4 Map 36
1.4.5 Type pointer map 36
1.5 TRUY XUẤT FILE (I/O) VÀ SERIALIZATION 37

1.5.1 Vấn đề quan tâm 37
1.5.2 Lớp CFile 37
1.5.3 Chuỗi hoá và CArchive 38
1.6 CÁC LỚP MFC CỦA CÁC ĐIỀU KHIỂN WINDOWS. 42
1.6.1 Vấn đề quan tâm 42
1.6.2 Các loại điều khiển 42
1.6.3 Loại CButton 43
1.7 DIALOG BOX, COMMON DIALOG VÀ PROPERTY SHEET 43
1.7.1 Vấn đề quan tâm 43
1.7.2 Hộp thoại (dialog) 43
1.7.3 Các hộp thoại thông dụng (Common Dialog Classes) 45
1.7.4 Property Sheet/Property Page 46
1.8 Một số điều khiển trong Windows 9.x* 58
1.8.1 Các loại điều khiển 58
1.8.2 Ví dụ tổng hợp: 59
1.8.2.1 Chương trình 1: 59
1.8.2.2 Chương trình 2: 66
CHƯƠNG 2. CẤU TRÚC DOCUMENT-VIEW CỦA MFC WINDOWS APP 74
2.1 GIỚI THIỆU DOCUMENT-VIEW VÀ SDI (SINGLE DOCUMENT INTERFACE) 74
2.1.1 Vấn đề quan tâm 74
2.1.2 Giới thiệu 74
2.1.3 Cấu trúc Document/View 75
2.1.4 Sự tương tác giữa phần Document và phần View 76
2.2 CÁC KIỂU VIEW 76
2.2.1 Vấn đề quan tâm 76
2.2.2 Giới thiệu 77
2.2.3 Lớp CScrollView và ứng dụng 77
2.2.4 Lớp CHtmlView và ứng dụng 83
2.2.5 Lớp CTreeView và ứng dụng 87
2.2.6 Lớp CListView và ứng dụng 96
2.3 MULTI-DOCUMENT, MULTI-VIEW VÀ MDI (MULTIPLE DOCUMENT INTERFACE) 106
2.3.1 Vấn đề quan tâm 106
2.3.2 Các hoạt động trong dạng MDI 106
2.3.3 Dạng splitter 121
2.3.3.1 Tạo vùng phân chia động: 121
2.3.3.2 Tạo vùng phân chia tĩnh: 121
2.3.4 Ví dụ tổng hợp 122
2.3.4.1 Chương trình 1: 122
2.3.4.2 Chương trình 2: 135
2.4 TOOLBAR VÀ STATUSBAR 157
2.4.1 Vấn đề quan tâm 157
2.4.2 ToolBar 158
2.4.3 StatusBar 160
CHƯƠNG 3. XỬ LÝ HỆ THỐNG 177
3.1 TIMER/IDLE 177
3.1.1 Vấn đề quan tâm 177
3.1.2 Timer 177
3.1.3 Idles 178
3.2 THREADS 187
3.2.1 Vấn đề quan tâm 187
3.2.2 Giới thiệu 187
3.2.3 Threads 187
3.2.4 Đồng bộ các threads(Thread Synchronization) 196
CHƯƠNG 4. LẬP TRÌNH CƠ SỞ DỮ LIỆU VỚI MFC 214
4.1 TRUY XUẤT VỚI ODBC/DAO 214
4.1.1 Vấn đề quan tâm 214

4.1.2 Giới thiệu 214
4.1.3 Ví dụ tổng hợp 218
4.1.3.1 Chương trình 1: 218
4.1.3.2 Chương trình 2: 221
4.2 TRUY XUẤT VỚI OLEDB 225
4.2.1 Vấn đề quan tâm 225
4.2.2 Giới thiệu OLEDB 225
4.2.3 Thực hiện tác vụ truy xuất dữ liệu với OLEDB: 225
4.3 TRUY XUẤT VỚI ADO 229
4.3.1 Vấn đề quan tâm 229
4.3.2 Giới thiệu ADO 229
4.3.3 Thực hiện truy xuất cơ sở dữ liệu với ADO: 229
CHƯƠNG 5. MFC VÀ ACTIVE X 243
5.1 COMPONENT OBJECT MODEL (COM) 243
5.1.1 Vấn đề quan tâm 243
5.1.2 Giới thiệu mô hình 3-lớp và n-lớp 243
5.1.3 Giới thiệu COM 243
5.1.4 Cấu trúc COM 243
5.1.5 COM server 246
5.1.6 MFC và COM 247
5.2 ACTIVE X CONTROL 250
CHƯƠNG 6. BÀI THỰC HÀNH 263
6.1 Bài thực hành 1 263
6.2 Bài thực hành 2 263
6.3 Bài thực hành 3 263
6.4 Bài thực hành 4 264
6.5 Bài thực hành 5 265
6.6 Bài thực hành 6 265
6.7 Bài thực hành 7 266
6.8 Bài thực hành 8 266
6.9 Bài thực hành 9 266
6.10 Bài thực hành 10 266
CHƯƠNG 7. BÀI TẬP TỔNG HỢP 267
7.1 Bài 1 267
7.2 Bài 2 267
7.3 Bài 3 267
7.4 Bài 4 268
7.5 Bài 5 268
7.6 Bài 6 268
7.7 Bài 7 268