Chapter 11

Windows Programming Concepts

Windows Programming Basics

- Windows API
- Microsoft Foundation Classes (MFC)
- Windows Forms

Elements of a Windows

- Let us go through them to be sure we have a common understanding of what the terms mean.
  - parent window, child window
  - border, size grip
  - title bar, title bar icon, status bar
    - system menu
      - click the title bar icon,
      - or right-click the title bar
  - client area
    - x increasing from left to right,
    - y increasing from top to bottom
  - minimize, maximize, close buttons

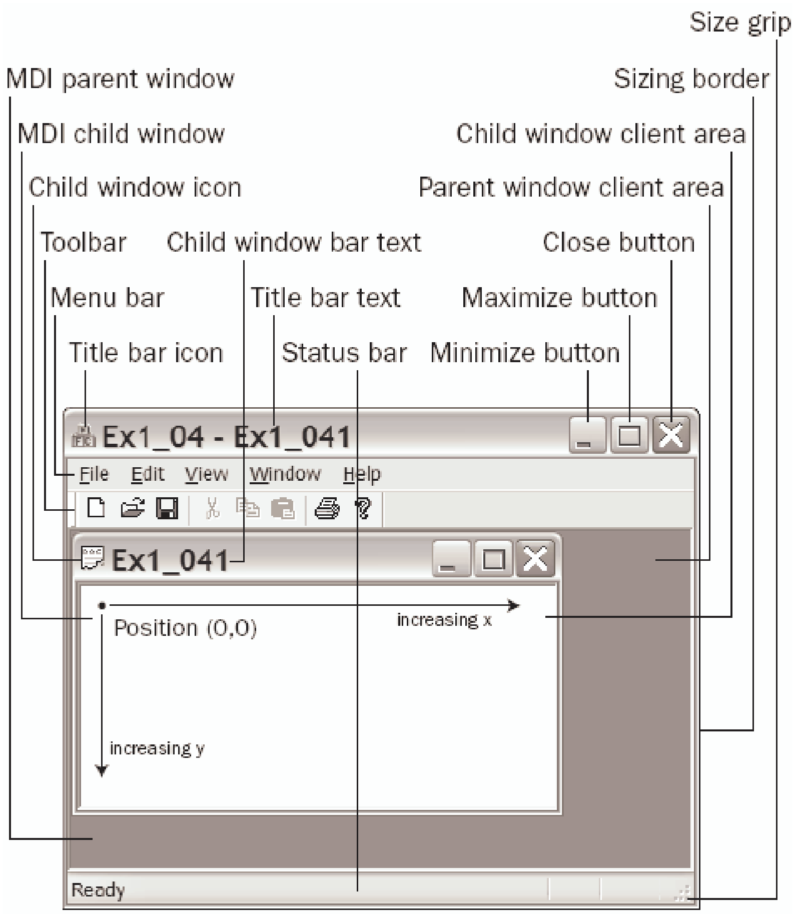
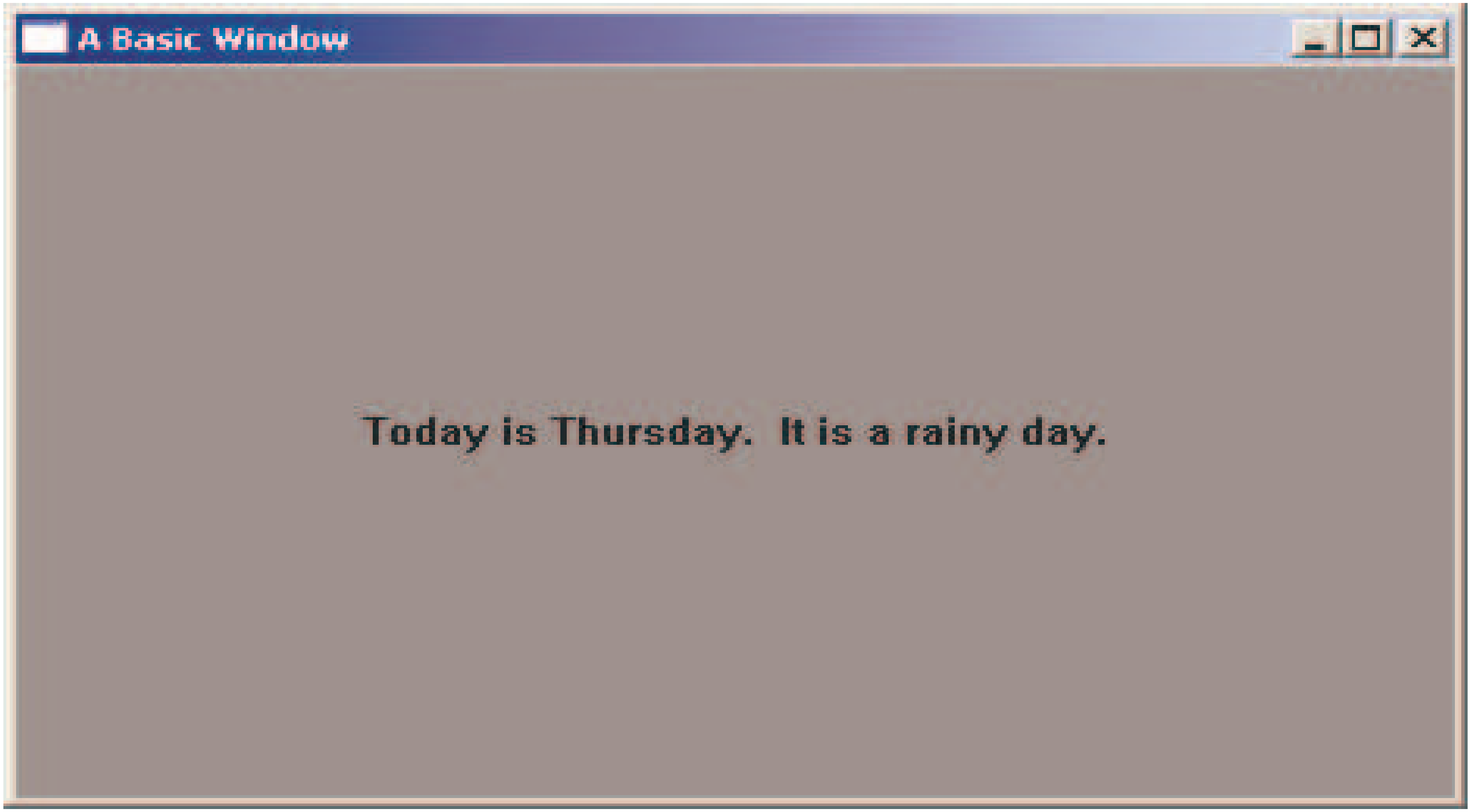


Figure 11-1

Ex11_01



Today is Thursday. It is a rainy day.

The Windows API

- The Windows API was developed in the days when C was the primary language.
  - Structures rather than classes are frequently used.

Windows Data Types

- P.618
  - BOOL
  - BYTE
  - CHAR
  - DWORD
  - HANDLE
  - HBRUSH
  - HCURSOR
  - HDC
  - HINSTANCE
  - LPARAM
  - LPCSTR
  - LPHANDLE
  - LRESULT
  - WORD

- All these types are contained in the header file `windows.h`
- Always use the Windows type.
  - For example, The Windows type WORD has been defined in one version of Windows as type unsigned short,
  - In another Windows version as type unsigned int.
  - On 16-bit machines these two types are equivalent, but on 32-bit machines they are different!

Notation in Windows Programs

- Hungarian notation – variable names have a prefix which indicating what kind of value it holds
  - b – boolean
  - by – byte
  - c – char
  - dw – DWORD, which is unsigned long
  - fn – function
  - h – handle
  - i – int
  - l – long
  - lp – long pointer
  - n – int
  - p – pointer
  - s – string
  - sz – zero terminated string
  - w – WORD, which is unsigned short

The Structure of a Windows Program

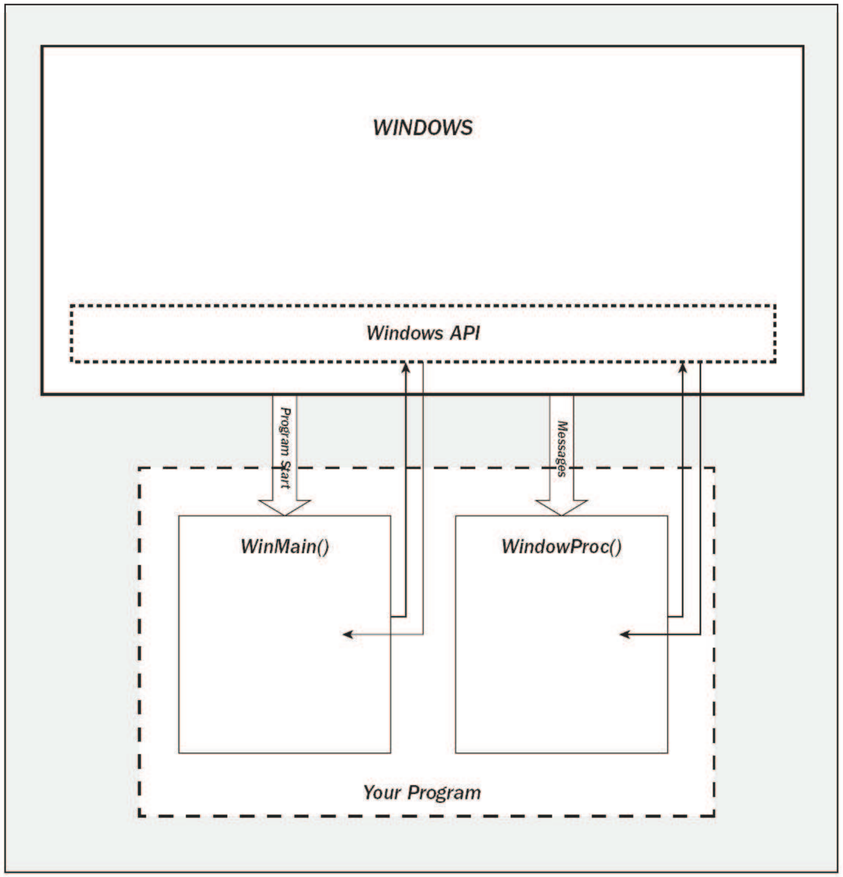


Figure 11-2

- For a minimal Windows program, you will only write two independent functions.
  - WinMain()
    - It initialize the application.
  - WindowProc()
    - This is usually the larger portion to handle user interaction.

The WinMain() Function

- int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPreInstance, LPSTR lpCmdLine, int nCmdShow);
  - hInstance – a handle to an instance
    - An instance stands for a running program
    - A handle is an integer value which identifies something
  - hPreInstance – the handle to the pervious instance.
    - This parameter is always NULL in newer versions of Windows.
    - In Windows 3.x, you need this to know that there is previous instance of the program or not.
  - lpCmdLine – a string containing the command line
  - nCmdShow – determines how the window looks
    - SW_SHOWNORMAL
    - SW_SHOWMINNOACTIVE
    - SW_HIDE
    - SW_SHOWMAXIMIZED
- See MSDN library online at http://msdn.microsoft.com/

WinMain() Needs to Do Four Things

- Tell Windows what kind of window the program requires
- Create the program window
- Initialize the program window
- Retrieve Windows messages intended for the program

Specifying a Program Window

- struct WNDCLASSEX (P.623)
  - WindowClass.cbSize = sizeof(WNDCLASSEX);
  - WindowClass.style = CS_HREDRAW | CS_VREDRAW;
    - Redraw if the vertical height or the horizontal width is altered.
  - WindowClass.lpfnWndProc = WindowProc;
  - WindowClass.lpszClassName = “OFWin”;
    - The name of the application

Creating a Program Window

- `RegisterClassEx(&WindowClass);`

- `HWND hWnd;`

```
hWnd = CreateWindow(
         szAppName,                          // the window class name
         L"A Basic Window",                  // The window title
         WS_OVERLAPPEDWINDOW,                          // Window style as overlapped
         CW_USEDEFAULT,                      // Default screen position of  upper left
         CW_USEDEFAULT,                      // corner of our window as x,y...
         CW_USEDEFAULT,                      // Default window size
         CW_USEDEFAULT,                      // ....
         0,                                  // No parent window
         0,                                  // No menu
         hInstance,                          // Program Instance handle
         0                                   // No window creation data
        );
```

- `ShowWindow(hWnd, nCmdShow);`

Initializing the Program Window

- UpdateWindow(hWnd);
  - This will ask Windows to send your program a message, which will invoke the code in the WindowProc() function to redraw the client.
    - This is the best way to get the client area drawn.

Dealing with Windows Messages

- The Message Loop (P.628)
  - GetMessage()
  - TranslateMessage()
    - Do some conversion for keyboard messages
  - DispatchMessage()
    - Call the WindowProc() function

```
struct MSG
{
    HWND        hwnd;          // handle to the window
    UINT        message;       // The message ID
    WPARAM      wParam;
    LPARAM      lParam;
    DWORD       time;          // Timestamp of the message
    POINT       pt;            // The mouse position
}
```

Conceptual Operation of GetMessage()

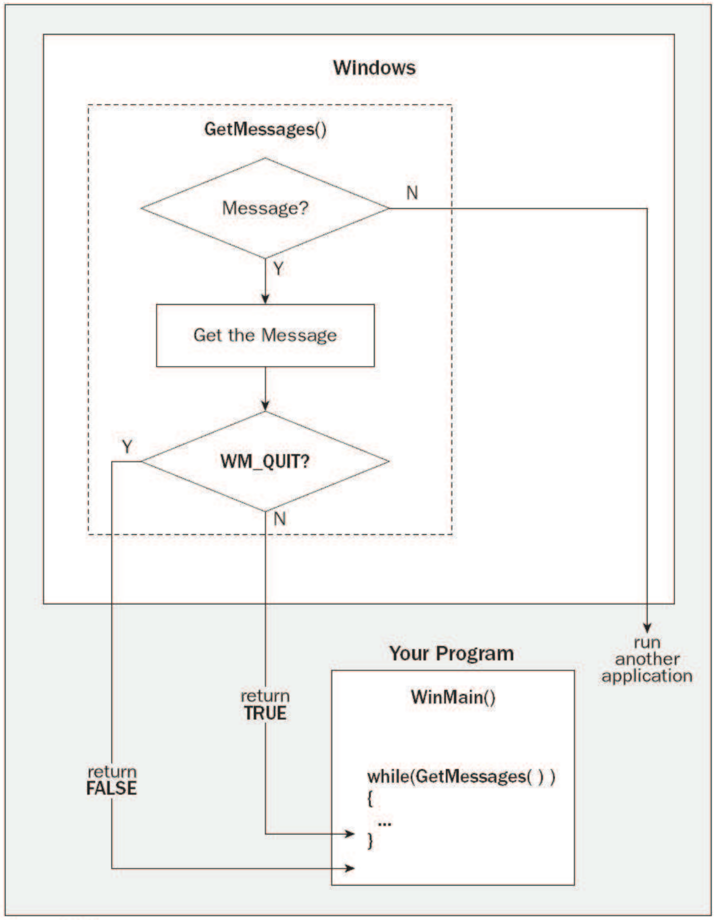


Figure 11-3

A Complete WinMain() Function

- P.631

Message Processing Function `WindowProc()`

- LRESULT CALLBACK WindowProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);
  - hWnd – a handle to the window in which the event causing the message occurred
  - message – the message ID
    - WM_PAINT, WM_LBUTTONDOWN
  - wParam – a 32-bit value
  - lParam – a 32-bit value

Decoding a Windows Message

- Selecting the message types that you want to process by putting a `case` statement for each case in the `switch`.

```
switch(message)
{ case WM_PAINT:
     // Code to deal with drawing the client data
     break;

  case WM_LBUTTONDOWN:
     // Code to deal with the left mouse button being pressed
     break;

  case WM_LBUTTONUP:
     // Code to deal with the left mouse button being released
     break;

  case WM_DESTROY:
     // Code to deal with a window being destroyed
  break;

  default:
     // Code to handle any other messages
}
```

Drawing the Window Client Area

- HDC hDC;
  - Display context handle
- PAINTSTRUCT PaintSt;
  - Structure defining area to be drawn
- hDC = BeginPaint(hWnd, &PaintSt);
  - Prepare to draw the window
- RECT aRect;
  - A working rectangle
- GetClientRect(hWnd, &aRect);
  - Get upper left and lower right of client area
- SetBkMode(hDC, TRANSPARENT);
  - Set text background mode

Draw the text in the window client area

```
DrawText( hDC, // Device context handle
   L"But, soft! What light through yonder window
   breaks?",
   -1, // Indicate null terminated string
   &aRect, // Rectangle in which text is to be drawn
   DT_SINGLELINE|      // Text format - single line
   DT_CENTER|          // - centered in the line
   DT_VCENTER);        // - line centered in aRect
```

- EndPaint(hWnd, &PaintSt);
  - Terminate window redraw operation

A Complee `WindowProc()` Function

- P.636

Ex11_01.cpp

- P.637
  - `#include <windows.h>`
  - Declare `WindowProc()` before `WinMain()`.
- Create a project using the Win32 Project
  - Instead of Win32 Console Application.

5/4 國道六號

- ◎活動名稱：2008台灣自行車日
- ◎主辦單位：南投縣政府
- ◎集合時間：08：30
- ◎集合地點：國道六號埔里鎮西安路入口處
- ◎騎乘路線：國道六號（20公里)
- ◎活動流程及內容：
- 08:30 報到