Winsock

Outline

- Windows programming
  - Window-oriented
  - Event-driven
- Winsock
  - Comparison with Berkeley socket
  - Socket event
- The template
  - WinMain
  - MainDlgProc
  - EditPrintf

Windows programming(1/6)

- There are 2 major properties in Windows programming
  - Window-oriented
  - Event-driven
- A Windows program is a program handling one or more windows to provide users some functionality

Windows programming(2/6)

```
int WINAPI WinMain(…) {
        // Register the window class.
        const wchar_t CLASS_NAME[] = L"Sample Window Class";
        WNDCLASS wc = { };
        wc.lpfnWndProc = WindowProc;
        …
        wc.lpszClassName = CLASS_NAME;
        RegisterClass(&wc);
        // Create the window.
        HWND hwnd = CreateWindowEx(0, CLASS_NAME, ……);
        …
        ShowWindow(hwnd, SW_SHOW);
        // Run the message loop.
        MSG msg = { };
        while (GetMessage(&msg, NULL, 0, 0)) {
                TranslateMessage(&msg);
                DispatchMessage(&msg);
        }
        return 0;
}
```

Windows programming(3/6)

```
LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg,
                            WPARAM wParam, LPARAM lParam)
{
        switch (uMsg) {
        case WM_DESTROY:
                PostQuitMessage(0);
                return 0;
        case WM_PAINT:
                {
                        PAINTSTRUCT ps;
                        HDC hdc = BeginPaint(hwnd, &ps);
                        FillRect(hdc, &ps.rcPaint, (HBRUSH) (COLOR_WINDOW+1));
                        EndPaint(hwnd, &ps);
                }
                return 0;
        }
        return DefWindowProc(hwnd, uMsg, wParam, lParam);
}
```

Windows programming(4/6)

- A window is a basic unit in Windows programming
- In Windows, many kinds of window is provided, such as a dialog, a tree view, and a list view

Windows programming(5/6)

- Event-driven model is used in Windows programming
- The target of a event is a window
- All operations is encapsulated into event
  - Typing one key on keyboard
  - Mouse moving
  - Window maximizing
  - ……

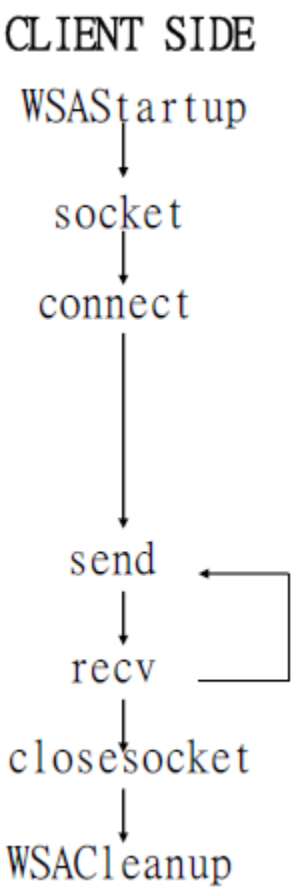
Windows programming(6/6)

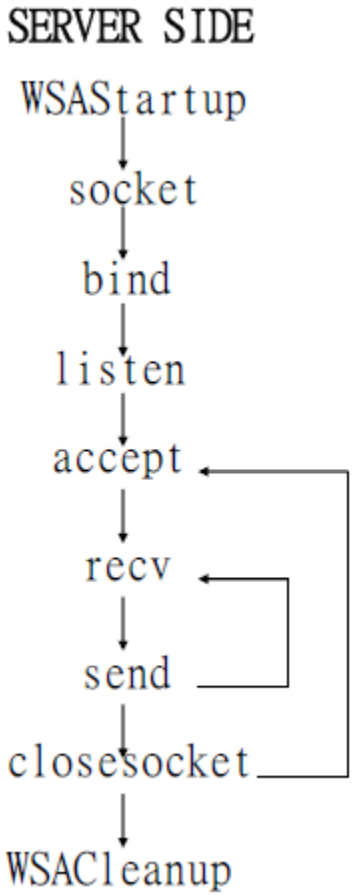
- A common flow of a Windows program
  - Define your event dispatcher
    - Describes what should be done if a event caught
  - Initialize windows
  - Do event polling

Winsock(1/2)

- A solution for networking in Windows
- Like Berkeley socket

read() <-> recv()
write() <-> send()
errno <-> GetLastError()





Winsock(2/2)

- Another mechanism for event driven model is provided
- If you want to handle sockets in event-driven, you must notify it

```
int WSAAsyncSelect( __in SOCKET s, __in HWND hWnd,
                    __in unsigned int wMsg, __in long lEvent );
```

s [in] - A descriptor that identifies the socket for which event notification is required.
hWnd [in] - A handle that identifies the window that will receive a message when a network event occurs.
wMsg [in] - A message to be received when a network event occurs.
lEvent [in] - A bitmask that specifies a combination of network events in which the application is interested.

The template(1/4)

```
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
        LPSTR lpCmdLine, int nCmdShow)
{
        DialogBox(hInstance, MAKEINTRESOURCE(ID_MAIN), NULL, MainDlgProc);
}
```

The template(2/4)

```
BOOL CALLBACK MainDlgProc(HWND hwnd, UINT Message, WPARAM wParam, LPARAM lParam)
{
   static HWND hwndEdit;
   int err;
   switch (Message)
   {
      ……
      case WM_COMMAND:
         switch (LOWORD(wParam))
         {
            case ID_LISTEN:
               // Service start here
               break;
            ……
         };
         break;
      ……
```

The template(3/4)

```
    case WM_SOCKET_NOTIFY:
      switch( WSAGETSELECTEVENT(lParam) )
      {
        case FD_ACCEPT:
          ssock = accept(msock, NULL, NULL);
          Socks.push_back(ssock);
          EditPrintf(hwndEdit, TEXT("=== Accept one new client(%d), List size:%d ===\r\n"),
                                              ssock, Socks.size());
          break;
        case FD_READ:
        //Write your code for read event here.
          break;
        case FD_WRITE:
        //Write your code for write event here
          break;
        case FD_CLOSE:
          break;
      };
      break;
    default:
      return FALSE;
  };
  return TRUE;
}
```

The template(4/4)

```
int EditPrintf (HWND hwndEdit, TCHAR * szFormat, ...)
{
    TCHAR   szBuffer [1024] ;
    va_list pArgList ;

    va_start (pArgList, szFormat) ;
    wvsprintf (szBuffer, szFormat, pArgList) ;
    va_end (pArgList) ;

    SendMessage (hwndEdit, EM_SETSEL, (WPARAM) -1, (LPARAM) -1) ;
    SendMessage (hwndEdit, EM_REPLACESEL, FALSE, (LPARAM) szBuffer) ;
    SendMessage (hwndEdit, EM_SCROLLCARET, 0, 0) ;
    return SendMessage(hwndEdit, EM_GETLINECOUNT, 0, 0);
}
```

Reference

- http://msdn.microsoft.com/en-us/library/ff381399(v=VS.85).aspx
- http://msdn.microsoft.com/en-us/library/windows/desktop/ms740673(v=VS.85).aspx