The scratch program

devblogs.microsoft.com/oldnewthing/20030723-00

July 23, 2003

Raymond Chen

Occasionally, there is need to illustrate a point with a full program. To avoid reproducing the boring parts of the program, let's agree on using the following template for our sample programs.

For expository purposes, I won't use a C++ class. I'll just keep all my variables global. In a real program, of course, instance data would be attached to the window instead of floating globally.

```
#define STRICT
#include <windows.h>
#include <windowsx.h>
#include <ole2.h>
#include <commctrl.h>
#include <shlwapi.h>
HINSTANCE g_hinst;                          /* This application's HINSTANCE */
HWND g_hwndChild;                           /* Optional child window */
/*
 *  OnSize
 *      If we have an inner child, resize it to fit.
 */
void
OnSize(HWND hwnd, UINT state, int cx, int cy)
{
    if (g_hwndChild) {
        MoveWindow(g_hwndChild, 0, 0, cx, cy, TRUE);
    }
}
/*
 *  OnCreate
 *      Applications will typically override this and maybe even
 *      create a child window.
 */
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
    return TRUE;
}
/*
 *  OnDestroy
 *      Post a quit message because our application is over when the
 *      user closes this window.
 */
void
OnDestroy(HWND hwnd)
{
    PostQuitMessage(0);
}
/*
 *  PaintContent
 *      Interesting things will be painted here eventually.
 */
void
PaintContent(HWND hwnd, PAINTSTRUCT *pps)
{
}
/*
 *  OnPaint
 *      Paint the content as part of the paint cycle.
 */
```

```
void
OnPaint(HWND hwnd)
{
    PAINTSTRUCT ps;
    BeginPaint(hwnd, &ps);
    PaintContent(hwnd, &ps);
    EndPaint(hwnd, &ps);
}
/*
 *  OnPrintClient
 *      Paint the content as requested by USER.
 */
void
OnPrintClient(HWND hwnd, HDC hdc)
{
    PAINTSTRUCT ps;
    ps.hdc = hdc;
    GetClientRect(hwnd, &ps.rcPaint);
    PaintContent(hwnd, &ps);
}
/*
 *  Window procedure
 */
LRESULT CALLBACK
WndProc(HWND hwnd, UINT uiMsg, WPARAM wParam, LPARAM lParam)
{
    switch (uiMsg) {
    HANDLE_MSG(hwnd, WM_CREATE, OnCreate);
    HANDLE_MSG(hwnd, WM_SIZE, OnSize);
    HANDLE_MSG(hwnd, WM_DESTROY, OnDestroy);
    HANDLE_MSG(hwnd, WM_PAINT, OnPaint);
    case WM_PRINTCLIENT: OnPrintClient(hwnd, (HDC)wParam); return 0;
    }
    return DefWindowProc(hwnd, uiMsg, wParam, lParam);
}
BOOL
InitApp(void)
{
    WNDCLASS wc;
    wc.style = 0;
    wc.lpfnWndProc = WndProc;
    wc.cbClsExtra = 0;
    wc.cbWndExtra = 0;
    wc.hInstance = g_hinst;
    wc.hIcon = NULL;
    wc.hCursor = LoadCursor(NULL, IDC_ARROW);
    wc.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
    wc.lpszMenuName = NULL;
    wc.lpszClassName = TEXT("Scratch");
    if (!RegisterClass(&wc)) return FALSE;
    InitCommonControls();               /* In case we use a common control */
```

```
    return TRUE;
}
int WINAPI WinMain(HINSTANCE hinst, HINSTANCE hinstPrev,
                   LPSTR lpCmdLine, int nShowCmd)
{
    MSG msg;
    HWND hwnd;
    g_hinst = hinst;
    if (!InitApp()) return 0;
    if (SUCCEEDED(CoInitialize(NULL))) {/* In case we use COM */
        hwnd = CreateWindow(
            TEXT("Scratch"),                /* Class Name */
            TEXT("Scratch"),                /* Title */
            WS_OVERLAPPEDWINDOW,            /* Style */
            CW_USEDEFAULT, CW_USEDEFAULT,   /* Position */
            CW_USEDEFAULT, CW_USEDEFAULT,   /* Size */
            NULL,                           /* Parent */
            NULL,                           /* No menu */
            hinst,                          /* Instance */
            0);                             /* No special parameters */
        ShowWindow(hwnd, nShowCmd);
        while (GetMessage(&msg, NULL, 0, 0)) {
            TranslateMessage(&msg);
            DispatchMessage(&msg);
        }
        CoUninitialize();
    }
    return 0;
}
```

Notice that all painting gets funneled through the `PaintContent` function. This allows us to route the `WM_PRINTCLIENT` message through the same paint function, which has as an immediate consequence the ability to animate the window with `AnimateWindow`. This will also prove useful for printing high-resolution screenshots.

Other than the trickiness with painting, there really isn't anything here that you shouldn't already know. The point of this program is to be a template for future programs.

My first mission will be an eight-part series on scrollbars.

That's right. Scrollbars.

I can't believe I have an eight-part series on scrollbars. And you probably can't believe you're reading about it.

Raymond Chen

Follow