Display an overlay on the taskbar button

devblogs.microsoft.com/oldnewthing/20130211-00

February 11, 2013

Raymond Chen

Today's "Little Program" displays an overlay on the taskbar button. I've seen some people call this a "badge", but "overlay" is the official term.

Start with our scratch program and make the following changes:

```
#include <comip.h>
#include <comdef.h>
#include <shlobj.h>
#include <shellapi.h>


_COM_SMARTPTR_TYPEDEF(ITaskbarList3, __uuidof(ITaskbarList3));
```

I decided to shake things up and use a different smart pointer library: `com_ptr_t`. (That'll teach you to complain that I don't use a smart pointer library in my samples. Now you get to complain that I use the *wrong* smart pointer library.)

```
HICON g_hicoAlert;
UINT g_wmTaskbarButtonCreated;


BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
  g_hicoAlert = LoadIcon(nullptr, IDI_EXCLAMATION);
  g_wmTaskbarButtonCreated = RegisterWindowMessage(
                              TEXT("TaskbarButtonCreated"));
  return TRUE;
}
```

Our overlay icon is the system exclamation point icon. I chose this because I'm lazy.

```
bool g_fHasOverlay = false;


void UpdateOverlayIcon(HWND hwnd)
{
  HICON hicon = g_fHasOverlay ? g_hicoAlert : nullptr;
  PCWSTR pszDescription = g_fHasOverlay ?
                        L”Attention required” : nullptr;
  ITaskbarList3Ptr sptb3;
  sptb3.CreateInstance(CLSID_TaskbarList);
  sptb3->SetOverlayIcon(hwnd, hicon, pszDescription);
}


void OnChar(HWND hwnd, TCHAR ch, int cRepeat)
{
  if (ch == ‘ ‘) {
    g_fHasOverlay = !g_fHasOverlay;
    UpdateOverlayIcon(hwnd);
  }
}


    HANDLE_MSG(hwnd, WM_CHAR, OnChar);


    default:
      if (uiMsg != 0 && uiMsg == g_wmTaskbarButtonCreated) {
        UpdateOverlayIcon(hwnd);
      }
      break;
```

A real program would have error checking, of course.

Press the space bar, and the overlay will be toggled on and off.

If you’re really clever, you might generate your overlay icons on the fly, say, if you wanted to report the number of unread messages or something.

I’ve heard that there’s one program out there that abuses the `ITaskbarList3::SetProgressState` method by changing its progress state repeatedly, causing its taskbar button to cycle through different colors to get the user’s attention.

Just a reminder: The user interface guidelines say that the way to get the user’s attention is to flash your taskbar button. Various parts of the system understand this convention and respond to it. (For example, the taskbar will temporarily unhide if a button starts flashing, and accessibility tools know how to signal the flash state to the user.) As always, the shell reserves the right to block this sort of abusive behavior in the future, just like it has done with abusive notification icons.

Raymond Chen

Follow