How do I get a high resolution icon for a file?

devblogs.microsoft.com/oldnewthing/20140120-00

January 20, 2014

Raymond Chen

Today's Little Program obtains a high resolution icon for a file.

Start with our scratch program and make these changes. Remember, Little Programs do little or no error checking. This week's smart pointer class is (rolls dice) `_com_ptr_t` !

```
...
#include <shlwapi.h>
#include <commoncontrols.h>
#include <comip.h>
#include <comdef.h>
_COM_SMARTPTR_TYPEDEF(IImageList, __uuidof(IImageList));
HICON g_hico;
HINSTANCE g_hinst;                          /* This application's HINSTANCE */
...
int GetIconIndex(PCTSTR pszFile)
{
  SHFILEINFO sfi;
  SHGetFileInfo(pszFile, 0, &sfi, sizeof(sfi), SHGFI_SYSICONINDEX);
  return sfi.iIcon;
}
HICON GetJumboIcon(int iImage)
{
  IImageListPtr spiml;
  SHGetImageList(SHIL_JUMBO, IID_PPV_ARGS(&spiml));
  HICON hico;
  spiml->GetIcon(iImage, ILD_TRANSPARENT, &hico);
  return hico;
}
```

The `GetIconIndex` function does nothing new. It simply retrieves the system image list icon index for a file's icon.

The `GetJumboIcon` retrieves an icon by its system image list index. First, it asks `SHGet-ImageList` for the jumbo image list, then it asks the jumbo image list for the icon.

Now all we have to do is hook the functions up.

```
void
PaintContent(HWND hwnd, PAINTSTRUCT *pps)
{
  DrawIconEx(pps->hdc, 50, 50, g_hico,
             0, 0, 0, nullptr, DI_NORMAL);
}
    ...
    if (SUCCEEDED(CoInitialize(NULL))) {/* In case we use COM */
        g_hico = GetJumboIcon(GetIconIndex(lpCmdLine));
        ...
        DestroyIcon(g_hico);
        CoUninitialize();
    }
    ...
```

Run this program and pass the full path to a file on the command line. (No quotation marks, even if it contains spaces!) Result: A gigantic icon for the file appears.

Instead of converting the system imagelist index into an icon, we could just ask the jumbo imagelist to render it directly.

```
int g_iImage;
void
PaintContent(HWND hwnd, PAINTSTRUCT *pps)
{
  IImageListPtr spiml;
  SHGetImageList(SHIL_JUMBO, IID_PPV_ARGS(&spiml));
  IMAGELISTDRAWPARAMS ildp = { sizeof(ildp) };
  ildp.himl = IImageListToHIMAGELIST(spiml);
  ildp.i = g_iImage;
  ildp.hdcDst = pps->hdc;
  ildp.x = 50;
  ildp.y = 50;
  ildp.rgbBk = CLR_NONE;
  ildp.fStyle = ILD_TRANSPARENT;
  spiml->Draw(&ildp);
}
    ...
    if (SUCCEEDED(CoInitialize(NULL))) {/* In case we use COM */
        g_iImage = GetIconIndex(lpCmdLine);
        ...
        // no cleanup necessary
        CoUninitialize();
    }
    ...
```

This is how Explorer deals with icons most of the time. It doesn't create actual icons; it merely remembers indices into the system imagelist, and when it needs to draw an icon, it calls the `Draw` method on the imagelist whose size corresponds to the image it wants.

Bonus chatter: The system imagelists come in four sizes (as of this writing). And yet *large* is one of the smallest available ones. Why is that?

The system imagelist sizes are

- Small
- Large
- Extra-Large
- Jumbo

The first two (*small* and *large*) were the only ones available in Windows 95. Windows XP added a size larger than large, which was named *extra-large*. And then Windows Vista added another size even larger than extra-large, which I named *jumbo*.

It's an artifact of history that one of the smallest icon sizes has the name *large*. It was the largest icon size at the time, but things got even larger since then.

Raymond Chen

Follow