Icons and cursors know where they came from

devblogs.microsoft.com/oldnewthing/20080820-00

August 20, 2008

Raymond Chen

If you create an icon by calling `LoadIcon`, the window manager loads the specified icon from the module you specified, but it also remembers where the icon came from. (This discussion also applies, *mutatis mutandis* to cursors, but I will just talk about icons to avoid awkardness.) When you pass the `LR_COPYFROMRESOURCE` flag to the `CopyImage` function, the window manager goes back to the original icon source to create the copy you requested rather than blindly stretching the pixels of the icon you passed in.

Remember that an ICO file represents not just one icon but rather a collection of icons (known as an “icon group”), each at a different size or color depth. The intent is that each icon in the icon group depicts the same underlying image, just tuned for particular settings. (Now, mind you, there is no enforcement of this intent. If you make your 16×16 image a smiling-face and your 32×32 image a barking dog, well, then that’s your problem.) For example, a single ICO file might contain a 16×16 image, a 32×32 image, and a 48×48 image. If somebody asks for the icon at one of those sizes, then the corresponding image is used. On the other hand, if somebody asks for, say, the 24×24 image, the window manager will take the 32×32 image and stretch it to the necessary size.

You can recover this “hidden source information” with the `GetIconInfoEx` function (new for Windows Vista). If the icon was loaded by ordinal, then the `szResName` is an empty string and the ordinal is placed in the `wResID` member. If the icon was loaded by name, then `wResID` is zero and `szResName` contains the resource name.

This is just some background information about icons (and cursors). Next time, we’ll put this information to use to solve a problem.

[Raymond is currently away; this message was pre-recorded.]

Raymond Chen

Follow