The evolution of the ICO file format, part 4: PNG images

devblogs.microsoft.com/oldnewthing/20101022-00

October 22, 2010

Raymond Chen

We finish our tour of the evolution of the ICO file format with the introduction of PNG-compressed images in Windows Vista. The natural way of introducing PNG support for icon images would be to allow the `biCompression` field of the `BITMAPINFOHEADER` to take the value `BI_PNG`, in which case the *image* would be represented not by a DIB but by a PNG. After all, that's why we have a `biCompression` field: For forward compatibility with future encoding systems. Wipe the dust off your hands and declare victory. Unfortunately, it wasn't that simple. If you actually try using ICO files in this format, you'll find that a number of popular icon-authoring tools crash when asked to load a PNG-compressed icon file for editing. The problem appeared to be that the new `BI_PNG` compression type appeared at a point in the parsing code where it was not prepared to handle such a failure (or the failure was never detected). The solution was to change the file format so that PNG-compressed images fail these programs' parsers at an earlier, safer step. (This is sort of the opposite of penetration testing, which keeps tweaking data to make the failure occur at a deeper, more dangerous step.) Paradoxically, the way to be more compatible is to be less compatible. The format of a PNG-compressed image consists simply of a PNG image, starting with the PNG file signature. The image must be in 32bpp ARGB format (known to GDI+ as `PixelFormat-32bppARGB`). There is no `BITMAPINFOHEADER` prefix, and no monochrome mask is present.

Since we had to break compatibility with the traditional format for ICO images, we may as well solve the problem we saw last time of people who specify an incorrect mask. With PNG-compressed images, you do not provide the *mask* at all; the *mask* is derived from the alpha channel on the fly. One fewer thing for people to get wrong.

Raymond Chen

Follow