Why are device-independent bitmaps upside down?

devblogs.microsoft.com/oldnewthing/20210525-00

May 25, 2021

Raymond Chen

Something that catches out everybody the first time they encounter it is that Windows device-independent bitmaps are upside down, in the sense that the first row of pixels corresponds to the *bottom* scan line of the image, and subsequent rows of pixels continue *upward*, with the final row of pixels corresponding to the topmost scan line.

Rather than calling these bitmaps *upside down bitmaps*, the less judgmental term is to call them *bottom-up bitmaps*.

Okay, so why are device-independent bitmaps bottom-up?

For compatibility with OS/2.

In OS/2 Presentation Manager, the origin of the graphics coordinate space is at the bottom left corner of the screen, with the *y*-coordinate increasing as you go toward the top of the screen. This is the mathematically correct coordinate system, which removes a lot of confusion when you start doing mathematics with your graphics.

It is my understanding that computer graphics involves a lot of mathematics.

If you align the graphics coordinate space with mathematical convention, then all the mathematical formulas carry over without having to introduce occasional negative signs to account for the reverse handedness. Rotation angles are always counter-clockwise. Transformation matrices operate in the way you learned in linear algebra class. Everything works out great.

Of course, if you aren't one of those deeply mathematical people, then having the *y*-coordinate increase upward means that the order of reading is opposite the order of increasing coordinates.

Which is weird.

Windows 2.0 and OS/2 started out as good friends, and Windows 2.0 adopted OS/2's bitmap format in order to foster interoperability between them. As we all know, that friendship soured over time, but the file format decision lingers on.

Raymond Chen

Follow