My life as a square (pixel)

devblogs.microsoft.com/oldnewthing/20100625-00

June 25, 2010

Raymond Chen

The shape of the Windows screen pixel has changed over the years. It has always had a rectangular shape, but the ratio of height to width has varied. (Someday, somebody will come up with a display adapter with hexagonal or triangular pixels, and then everything will fall apart.) Our story begins with the Color Graphics Adapter, better known by the abbreviation CGA. Depending on the video mode, your pixels could be 1:1.2 in 320×200 color mode (all aspect ratios are given in the form width:height for some reason), or 1:2.4 in 640×200 monochrome mode. Windows used the monochrome mode because the colors available in color mode were pretty useless. You could have black, white, hot pink, and aqua; or you could have black, brown, orange-red and green (no white!). Hideous colors no matter how you slice it. The next major advancement in display adapter technology was the Enhanced Graphics Adapter (EGA), which brought you a stunning 16 colors in a 640×350 grid with an aspect ratio of 1.83:1. But for the step forward in terms of color, you also took a step backwards into the insanity of planar video modes. EGA memory was arranged in planes, and what's worse, each plane had the same memory address! You had to use other registers to program the video card to tell it which plane you wanted to talk to when you accessed a specific memory address. (And the memory behaved differently depending on whether you were reading from it or writing to it.) Next to arrive was VGA, the Video Graphics Array. (I like how the names of all the display adapters are largely meaningless.) VGA brought us square pixels, if you used them at 640×480 resolution (which is what Windows did). Finally, you had a display adapter where a stair-step pattern actually gave you a 45-degree line. Pretty much every display adapter since VGA has supported square pixels, and pretty much every display *driver* preferred those square modes instead of the non-square modes. If you go into the Windows Display control panel, you can force one of the weird non-square modes, but all of the standard dimensions (800×600, 1280×1024, etc.) use square pixels. Okay, so how does a program obtain the pixel aspect ratio? You start by getting a device context (or if you're really clever, an information context) for the device you are interested in, and then ask for the `ASPECTX` and `ASPECTY` device capabilities. These capabilities tell you the relative width and height of a pixel. If the x-aspect and y-aspect are equal, then you have square pixels. If not, then their ratio tells you the aspect ratio. As a special bonus, there is also the `ASPECTXY` device capability, which is the length of the diagonal of a pixel, relative to the other aspect values. This is something you can calculate yourself based on the other two metrics, but it's provided as a courtesy. Let x be the value of the `ASPECTX` device capability, y be the value for `ASPECTY`, and h be the value for `ASPECTXY` (h for hypotenuse). Then the values are related by the formula

> $h^2 = x^2 + y^2$

(Of course, there will be rounding errors since all the device capabilities are integers.) Now that non-square pixels have been pretty much entirely replaced by square pixels in the past fifteen years, I wonder how many programs will start looking weird once a display with non-square pixels is reintroduced to the world of Windows. With newer and more unusual form factors for PCs coming out all the time, it's perhaps just a matter of time before a non-square-pixel display becomes the new must-have device. (And that time appears to have come: Pixels on HD TVs are not always square.)

The reality of non-square pixels explains why most metrics come in both horizontal and vertical versions.