

Re: Aaarrggghhh -- that ugly flicker!

Source: <http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2004-10/0669.html>

From: Michael Culley (mculley_at_NOSPAMoptushome.com.au)

Date: 10/06/04

Date: Wed, 6 Oct 2004 13:50:53 +1000

"Rob" <none> wrote in message news:OILGIM1qEHA.592@TK2MSFTNGP11.phx.gbl...

> *I'm not sure it supports your*

> *view that it's easier to draw directly to the window*

Yes, I should have thought about that more before I wrote it. What I really meant was it is often better just to draw straight to the window. Quite often it is more difficult to draw direct to the screen.

> *but it definitely*

> *illustrates that double buffering is often an unnecessary "first resort".*

> *Here's an interesting article on how to eliminate flicker. It's aimed at*

> *the C language but I think it's still useful.*

What it says in regards to double buffering makes a lot of sense:

A common method to completely eliminate flickering windows is to use a technique called double-buffering. This basic idea is to draw a window's contents into an off-screen buffer, and then transfer this buffer to the screen in one fell-swoop (using BitBlt). This is a pretty good way to reduce flicker, but is often overused, especially by programmers who don't really understand how get efficient drawing working.

This method is a little slow, because the offscreen memory-DC is created from scratch every time the window needs to be drawn. A more efficient method would be to create the memory DC only once, big enough so that the entire window can be painted at any time. When the application terminates, the memory DC would then be destroyed. Both these methods are potentially quite memory-intensive, especially if the memory DC needs to be the size of a screen (1024 * 768 * 32 bytes=2.5 Mb).

Double-buffering will also be twice as slow as it needs to be. Because you are drawing once to the memory-DC, then again during the "blit", you are using up clock cycles when you don't need to. Granted, a fast graphics card will perform a BitBlt very quickly, but it's still wasted CPU.

If your application needs to display quite complicated information (say, like a web-page), then you would need to use the memory-DC method. Take Internet Explorer, for instance. There is no way it would be able to render

microsoft.public.vb.general.discussion: Re: Aaarrggghhh -- that ugly flicker!

a web-page with no flickering without using double-buffering.

Double-buffering doesn't have to be used to paint a whole window. Imagine that you had just a small portion of a window that contained a complex graphic object (maybe a semi-transparent bitmap or something). You could use an off-screen DC to draw just this one region, and BitBlt that to the screen, whilst drawing the rest of the window normally.

Sometimes though, with a little careful thinking, it is often possible to avoid double-buffering and draw straight to the screen. As long as you don't break the golden rule, "Never draw over the same pixel twice", you will achieve flicker-free drawing.

--

Michael Culley