Subject: Re: IDL/MSWin pixmap limitations, Part 2
Posted by David Fanning on Mon, 25 Nov 2002 16:53:16 GMT
View Forum Message <> Reply to Message

Craig Hamilton (someone@microsoft.com) writes:

> In August I asked about a solution to the problem of "unable to create
> pixmap" when running IDL on MSWin 2000. The Most Noble David Fanning explained
> that pixmaps are created in video RAM, and, unless your video card is
> "smart", I am limited by the video card. He also suggested a workaround using unmapped
> draw widgets instead of pixmaps.
>
> I have questions about these two potential solutions:
>
> 1. Get a smarter video card (with loads of video RAM, I presume). Can
> anyone suggest a video card that can use system RAM when necessary? I have
> tried a GEForce4 4200 AGP card with 128 MB of RAM with no success trying
> to allocate about 1500 pixmaps of total size of around 90 MB.
> Interestingly, a Radeon VE with 32MB of video RAM works with roughly the same limits
> as the GEForce4 card. So, it doesn't seem to be as simple as just throwing
> more video RAM at the problem.
>
> 2. Use unmapped draw widgets instead of pixmaps. I started working on this
> and ran into the problem (at least according to the documentation) that
> mapping/unmapping applies only to base widgets, not draw widgets.
>
> So, if I unmapped a draw widget, it goes up to its parent base and unmaps
> it, which is not what I want.

I don't know which video card is "smarter". Maybe Randy Frank
is still listening in. He will know.

If I said "unmapped" draw widget, I'm sorry. At the time I
was answering the question we were mapping and un mappingg
"displays", which consisted of a base widget, a draw widget,
and pixmap. We thought of them as "images", so I probably
confused you by using imprecise language.

But we found that even the noble solution mentioned above
didn't work so well in practice. :-(

After 30-40 images, we still found ourselves running out
of window resources, and--of course--our client wanted to
have *hundreds* of images open at once, after they got a look
at our software and what it could do. :-)

We have since gone to what I call the "smoke and mirrors"
approach to the problem. Fortunately for us, our design

makes it possible to only view one "image" at a time,
although you can select any one of the hundreds of images
in the stack. In practice, the user usually will select
the "previous" or "next" image.

We reasoned that while the user was looking at the currently
selected image, we could be doing some fancy footwork. So we
designed our "pixmaps" so that they actually create a pixmap
window (and use window resources) only when they absolutely
have to. Most of the time, they just carry around a pointer
to an image that they *would* use as the pixmap, if they
had to. Thus, the current, previous, and next images use
pixmaps, but anything else has to create a pixmap window
when requested.

This results in instantaneous display of the previous and
next image, but there is a slight delay if the user suddenly
wants to go to (for example) the first image in the stack.
But the delay is not onerous (a momentary blink), and the
up-side is that we can now load as many images into our system
as required (limited only by the virtual memory available for
paging).

Of course, all of this (displays, draw widgets, pixmaps, etc.)
are wrapped up as objects so they are small, smart, and self-contained.
They are quite easy to work with (well, once you get the hang of it).
The beauty of the system is that we could completely re-work the
way it all worked just by changing the code in a single object.
If you have ever tried to do this is a non-object system, you
can appreciate (again!) the power of object programming. :-)

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155