Subject: Issue in docking example application
Posted by frankdeprins on Mon, 22 Sep 2014 21:01:03 GMT
View Forum Message <> Reply to Message

Hello Mirek,

In the docking example application, there is an issue.
Well, the issue is not exactly in that example or the docking library, but in CtrlLib.
Here is what I noticed:

- When you minimize a docked window and then hover the tab, the docked window will reveal itself (expand). So far, so good.
- But when you move the mouse back out of the window and it's tab, there is something wrong with the auto-hiding: the window hides itself indeed, but then briefly flashes to it's full size once again.

After some digging, I found the cause to be in the Animate procedure in CtrlUtil.cpp.

The following change made the issue go away:

```
void Animate(Ctrl& c, const Rect& target, int type)
{
 if(type < 0)
  type = GUI_PopUpEffect();
 Rect r0 = c.GetRect();
 dword time0 = GetTickCount();
 int anitime = 150;
#ifdef SLOWANIMATION
 anitime = 1500;
#endif
 if(type)
  for(;;) {
   int t = int(GetTickCount() - time0);
   if(t > anitime)
    break;
   if(type == GUIEFFECT_SLIDE) {
    Rect r = r0;
    if(r.left != target.left)
       r.left -= ((r.left - target.left)* t) / anitime;
    if(r.top != target.top)
       r.top -= ((r.top - target.top) * t) / anitime;
    if(r.right != target.right)
       r.right += ((target.right - r.right) * t) / anitime;
    if(r.bottom != target.bottom)
       r.bottom += ((target.bottom - r.bottom) * t) / anitime;
    c.SetRect(r);
   }
   else
   if(type == GUIEFFECT_FADE)
```

```
    c.SetAlpha((byte)(255 * t / anitime));
   else
    break;
   c.Sync();
   Sleep(0);
#ifdef SLOWANIMATION
   Sleep(100);
#endif
  }
 c.SetRect(target);
 c.SetAlpha(255);
}
```

Could you please review this change and, if OK, apply it?
I know this function is used a lot for all animations, so I guess some care will be needed.

Regards,

Frank

Subject: Re: Issue in docking example application
Posted by 281264 on Sun, 28 Sep 2014 15:19:21 GMT
View Forum Message <> Reply to Message

Frank,

Docking, when used together with OpenGL (GLCtrl) produces a unexpected effect: when you AutoHide a docked window and hover the mouse over the hidden window, it affects the main GLCtrl producing its malfunctioning. That is to day: auto-hidding is somehow incompatible with GLCtrl. Do you mind looking into it?

Thx,

Javier

Subject: Re: Issue in docking example application
Posted by frankdeprins on Mon, 29 Sep 2014 13:46:01 GMT
View Forum Message <> Reply to Message

Hello Javier,

I am always prepared to try some things out.
Mind you, there was no OpenGL around when I encountered the issue for which I proposed this fix.

Regards,

Frank

Subject: Re: Issue in docking example application
Posted by mirek on Tue, 30 Sep 2014 07:11:25 GMT
View Forum Message <> Reply to Message

I have had bad feeling about these "!=" comparisons, so I have used a little bit more conservative fix, but should be fixed now.

Mirek

Subject: Re: Issue in docking example application
Posted by frankdeprins on Tue, 30 Sep 2014 19:42:05 GMT
View Forum Message <> Reply to Message

The 'bouncing' is indeed also fixed with this code but there is one difference:
With the code I proposed, the docked window did it's auto-hiding with a sliding animation while, with this fix, it just closes in one movement.

Frank

Subject: Re: Issue in docking example application
Posted by 281264 on Thu, 02 Oct 2014 10:45:40 GMT
View Forum Message <> Reply to Message

Frank,

If you have time to spare please have a look at this example: jus two GLCtrl depicting an spiral. One is the main window and the other is docked. Every time you click with the mouse, the spiral in the clicked window rotates. Simple.

But when you Auto-Hide the secondary GLCtrl then the main window ceases to work properly. Although not shown in this example, when you try to make a zoom or a pan, then the main window gets distorted.

I think that the way Docking is designed imposes a serious limitation when combined with OpenGL. Any suggestion?

Thx.

Javier

File Attachments
1) prueba_OpenGL_DockWindow.zip, downloaded 523 times

Subject: Re: Issue in docking example application
Posted by frankdeprins on Thu, 02 Oct 2014 15:03:37 GMT
View Forum Message <> Reply to Message

I'm afraid I cannot compile your example; this line (gl_struct1.h):
void StartDC()const { wglMakeCurrent(GetDC(), GetP); }
raises errors:

GetDC: function does not take 0 arguments
GetRC: undeclared identifier

I presume this has to do with the fact that I'm on Windows and, I guess, you are on Linux or, at least, something using X11.
I found out that I had to pas GetHWND to GetDC, but for the second error I have no clue what to do.

Subject: Re: Issue in docking example application
Posted by 281264 on Fri, 03 Oct 2014 10:42:32 GMT
View Forum Message <> Reply to Message

Frank,

I am sorry: I forgot to mention that I typically use Glew http://glew.sourceforge.net/ (therfore you have to download and install it) and I also make some minor adjustements in GLCtrl (in this regard, please use the attached version of GLCtrl and try again).

In the Package Organizer you should have: When MSC Library: opengl32 glu32 glew32

Thx,

Javier

File Attachments
1) GLCtrl.zip, downloaded 646 times

Subject: Re: Issue in docking example application
Posted by frankdeprins on Mon, 06 Oct 2014 06:24:52 GMT
View Forum Message <> Reply to Message

I'm sorry Javier, but now we are talking about quite some amount of custom code.

Can't you reduce the testcase and limit it to the use of Upp code/controls only?
I hope you understand that I cannot check all this custom code, especially since I am completely unfamiliar with OpenGL stuff.

What I did find strange is that the docking animation is extremely slow in your test app.
Could it be you introduced something slowing down the drawing code of your control?
During the animation, the control gets resized and, hence, repainted quite a lot and when there is some slowdown there, it will have quite some influence.
As for the rest, it looked like the contents of the control were not always refreshed/repainted when they should, but I could not really observe distortions.

Subject: Re: Issue in docking example application
Posted by frankdeprins on Mon, 06 Oct 2014 07:49:33 GMT
View Forum Message <> Reply to Message

I transformed the GLDrawDemo app from Upp to use docking and noticed the same problems as in your sample:
° GL control is not repainted properly
° Docking animation is terribly slow
Once again, though, I could not observe any distortion or bad drawing other than the repaint issue in the main window. The GL control in the docking window remains black though. I guess that is something I do incorrectly.
On the other hand, I found quite some problem reports about this GLCtrl in the forum and I have the impression that it is not exactly the crown jewel of Upp.

File Attachments
1) GLDockDemo.7z, downloaded 456 times

Subject: Re: Issue in docking example application
Posted by 281264 on Mon, 06 Oct 2014 20:26:42 GMT
View Forum Message <> Reply to Message

Frank,

Thank very much you for your effort.

I think the issue here is the way Docking is designed, its very basic concept that makes it, somehow, not fully compatible with OpenGL when the Auto-Hide mode has been activated.

Is not worthwhile looking into it for we need an expert in both OpenGL and Docking.

Cheers,

Javier

Subject: Re: Issue in docking example application
Posted by frankdeprins on Tue, 07 Oct 2014 05:45:32 GMT
View Forum Message <> Reply to Message

Personally, I have the impression that the GlCtrl is the culprit.
Even without the docking, the same problems as I encountered were already reported in this forum.
It looks like, when you have multiple GlCtrl instances in one topwindow, they all paint to the same surface (some static member somewhere?).
In my converted GlDrawDemo -> GLDockDemo app, I had the impression that the docking GlCtrl was drawing to the one in the toplevel window. Something that came to my mind observing your example app as well.
But, as you say, I am not an OpenGL or even graphics expert. So, I'm afraid I'm at the end of this.