Subject: Re: DockCtrl (A dockable window widget for U++)
Posted by mrjt on Fri, 22 Feb 2008 13:33:52 GMT
View Forum Message <> Reply to Message

Quote:PS: I also noticed a cool thing about qt. If dragged window is to be tabbed the destination place is not only light blue (like in our docking engine) but this blue area is also transparent so you can see previous tab content. I like it.
I had also been thinking about this, but I've only just worked out how to do it (my internal structure has now got to a point where I modify stuff quite easily. Thank FSM). See attached for an example. I'm going to improve the tab behaviour I think, but that requires more changes that I'm not quite ready to make. Looks nice though

Oblivion: The trick is to overload Ctrl::PostPaint in you container class (TabWindow?) and then Draw something transparent over the whole View area. I'm using the following code to generate my highlight images:

```
Image StandardHighlight(Color inside, Color border)
{
 Size sz(5, 5);
 ImageBuffer ib(sz);
 RGBA *q = ~ib;
 for (int i = 0; i < 5; i++)
  for (int j = 0; j < 5; j++)
   *(q++) = (i == 4 || j == 4 || !i || !j) ? border : inside;
 ib.SetHotSpot(Point(1, 1));
 ib.Set2ndSpot(Point(3, 3));
 return ib;
}

Image AlphaHighlight(const Image &img, int alpha)
{
 ImageDraw draw(img.GetSize());
 draw.Alpha().DrawRect(img.GetSize(), Color(alpha, alpha, alpha));
 draw.DrawImage(0, 0, img);
 // Is there a better way to set hotspots than this?
 ImageBuffer ib((Image)draw);
 ib.SetHotSpot(Point(1, 1));
 ib.Set2ndSpot(Point(3, 3));
 return ib;
}

CH_STYLE(DockWindow, Style, StyleDefault)
{
 Image img = StandardHighlight(Blend(SColorHighlight, SColorPaper, 90), SColorHighlight);
 stdhighlight = img;
 tabhighlight = AlphaHighlight(img, 160);
}
```

unodgs wrote on Thu, 21 February 2008 20:04
Ctrl is not bad. Although in QuickTabs I used Ctrl to scroll tabs. I still think it's better (as option or as default) to follow eclipse solution. If you start draging the tab you drag the tab until mouse left tabs area.

I've just tried this and I don't think it possible with the current Upp Drag/Drop implementation. To make it work you need to call PasteClip::Reject from DragLeave. You have to do a nasty bodge just to get the PasteClip object to begin with, and calling Reject doesn't seem to release the mouse, so it's not possible to start dragging the window. The alternative is to not use DnD for tab dragging, but I'd much rather have a 'correct' solution.

Having tried it I agree that this is how it should work though, so I'll keep trying.

unodgs wrote on Thu, 21 February 2008 20:04
It looks quite nice, but the main idea behind grouping was to reduce the scrollbar.. and make switching between tabs more comfortable.
The grayed-out tabs are not included in the scroll limit. It works exactly the same as before except that you can see that there are non-active tabs.

I'm also trying out a new drag highlight:

I think it's an improvement, what do you think?

Incidentally, all these features (grouping, scrollbar/tabbar autohide, etc.) are in my general TabBar class, based around standard Upp Value/Display architecture. QuickTabs is just a specialized bas class of this TabBar. It seems a shame to waste it on one project, should I release it to the SVN Bazaar?

File Attachments

1) tabs.png, downloaded 977 times
2) DockTest.zip, downloaded 332 times