Subject: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Fri, 26 Feb 2010 13:49:51 GMT
View Forum Message <> Reply to Message

hi there,

FEATURE:

Docking is great stuff. But in some cases users mau *not* close any of the tab, only dragging. it is currently not possible to just disable *close* stuff, disabling Close button in DockWindow just disabled it in DockCont, whereas The TabBar down there and the ContextMenus continue to provide this.

down there is a small fix, to chain this to the Close Button availability in Config Menu. based on current 2150 revision.

maybe it is helpfull. i think it makes kind of sense

BUG:

in ConfigMenu, when assigning new groups and moving DockingCtrls there, there seems to be a bug. they disappear. Could not fix it.

File Attachments
1) Docking.rar, downloaded 702 times

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Fri, 26 Feb 2010 14:17:10 GMT
View Forum Message <> Reply to Message

concerning the bug:

maybe there is some TreeCtrl problem, when moving the entries due to some recent changes to TreeCtrl

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Tue, 02 Mar 2010 13:27:40 GMT
View Forum Message <> Reply to Message

another problem:

DockConfig.cpp:230

```
void DockConfigDlg::OnOK()
{
```

```
 dock.Animate(~animate); //NOT CLEAR / CONSEQUENT
...
}
```

is not clear or consequent that animation will be turned off completely, since it edits only first flag. also, the checkbox is initialized with IsAnimate() which OR's flags, so sometimes unselecting animate and opening config again still displays 'Animate'

maybe change it for

```
 dock.Animate(~animate, dock.IsAnimatedFrames(), ~animate);
```

---

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Tue, 02 Mar 2010 13:42:43 GMT
View Forum Message <> Reply to Message

additionally to first post for close issue:

TopWindow DockCont should be made non closable via close button up right.

```
void DockCont::CloseAll()
{
 if(!base->HasCloseButtons()) return; //<<<<ADD
 base->CloseContainer(*this);
}
```

---

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Tue, 02 Mar 2010 14:04:19 GMT
View Forum Message <> Reply to Message

another issue:

when a Docking has got no buttons visible, the Title if the selected Docking also disappears, i could not figure out why
because
DockCont.cpp:196

```
  if (!s.title_font.IsNull()) {
   Ctrl *c = GetLastChild();
   while (c && !c->IsShown() && c->GetParent())
    c = c->GetNext();
```

```
   if (s.handle_vert)
    r.top = c ? c->GetRect().bottom + m.top : m.top;
   else
    r.right = c ? c->GetRect().left + m.right : m.right;
   w.Clip(r);
   WString text = IsNull(dc->GetGroup()) ? dc->GetTitle() : (WString)Format("%s (%s)",
dc->GetTitle(), dc->GetGroup());
   w.DrawText(p.x, p.y, s.handle_vert ? 900 : 0, text, s.title_font, s.title_ink[focus]);
   w.End();
  }
```

seems to be reached and calls stuff to draw..

---

## Subject: Re: Docking: BUG + FEATURE: Disable Close completely
## Posted by mrjt on Wed, 03 Mar 2010 17:29:17 GMT

View Forum Message <> Reply to Message

Hi! I'm glad you're using the code.

The fix for the missing title text is:

```
  if (!s.title_font.IsNull()) {
   Ctrl *c = GetLastChild();
   while (c && !c->IsShown() && c->GetParent())
    c = c->GetPrev();
   if (s.handle_vert)
    r.top = c ? c->GetRect().bottom - m.top : r.top - m.top;
   else
    r.right = c ? c->GetRect().left - m.right : r.right - m.right;
   w.Clip(r);
   WString text = IsNull(dc->GetGroup()) ? dc->GetTitle() : (WString)Format("%s (%s)",
dc->GetTitle(), dc->GetGroup());
   w.DrawText(p.x, p.y, s.handle_vert ? 900 : 0, text, s.title_font, s.title_ink[focus]);
   w.End();
  }The clipping rect was being incorrectly calculated when there were no buttons.
```

Thanks for your other bug reports:
Animate config - This check box was added before the feature was split and I obviously forgot to change it. I'll add a second checkbox to the config form to independently control them.

Config dialog tree - I'll have to look at this, it used to work but as Koldo has said there seem to have been some changes to TreeCtrl.

Closing buttons and stuff - This is pretty complicated, the correct solution involves removng the X button from floating windows and all of the Close, Close Group and Close All options from the menus.

I haven't looked at the code for a while and as you can see it's pretty complicated , but I'll try and get all the fixes incorporated by the end of the week.

There is also an outstanding bug with certain window manager settings on Linux, but I have yet to find a solution to that one.

---

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Wed, 03 Mar 2010 21:16:37 GMT
View Forum Message <> Reply to Message

hey mrjt, thanks a lot..the fix is just fine..

i noticed another slight misbehaviour, which is reproducable in my software.

when i change the Title(..) of a DockableCtrl through a NON MainThread (which should work fine since it uses kind of Posting invokes on processing Draw stuff) it sometimes needs a mouseover to have the change take place. i use it in tabbing environment. the tabs change tha caption on mouseover, but the title bar above does not change on mouseover, only when i doublecklick it (triggers a resize / repaint)

concerning the close stuff: the provided solution in first post well ties up the availability of close behaviour to HasCloseButtons() (selectable in config with "Close"), but maybe you've got some more flexible way. the availability of close button of TopWindow is set on creation, dont know if one can change that, but disabling should just do fine

thanks again

---

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by mrjt on Thu, 04 Mar 2010 14:37:03 GMT
View Forum Message <> Reply to Message

Ah, good find.

To fix the title not updating you need to change DockableCtrl.h:
```
 DockableCtrl&  Title(const char *_title)  { title = _title; if (GetParent())
GetParent()->RefreshFrame(); return *this; }
 DockableCtrl&  Title(const WString& _title) { title = _title; if (GetParent())
GetParent()->RefreshFrame(); return *this; }
```

And this change in DockCont.h makes it keep the new title when it's detached/floated:
```
 void    StateFloating(DockWindow& dock)    { State(dock, STATE_FLOATING); Title(GetTitle());
```
When floating you still won't able to dynamically change the window title though. There just isn't any obvious way for the DockCont window to be notified. I'll think about it though.

The issue with just blocking the Close action as you have is that surely the option is still visible in all the menus? It's just that when the user slects it nothing happens?

It should be possible to remove the close box in Upp now, the option seems to have appearred in the code as NoCloseBox. This change in DockCont.cpp makes it disappear:

```
void DockCont::WindowButtons(bool menu, bool hide, bool _close)
{
 AddRemoveButton(windowpos, menu);
 AddRemoveButton(autohide, hide);
 AddRemoveButton(close, _close);
 NoCloseBox(!_close);
 SyncButtons();
}
```

No idea whether this works on Linux yet. I'll put these changes into the SVN when I've had time to integrate everything.

---

Subject: Re: Docking: BUG + FEATURE: Disable Close completely
Posted by kohait00 on Thu, 04 Mar 2010 19:02:34 GMT
View Forum Message <> Reply to Message

thanks a lot, the fixes work pretty well, the only thing is, that via config disabled NoClose...doesnt get applied untill a doubleclick to the title bar..but ots ok

Quote:
The issue with just blocking the Close action as you have is that surely the option is still visible in all the menus? It's just that when the user slects it nothing happens?

no, the options do even disappear, so the user doesnt get any opportunity to close it, while the code api still can close it.

here is the current stuff, applied all your changes and mine, take a look on it, decide what you want to keep, maybe that saved you time, its a ready readonly svn folder to have a diff...

File Attachments
1) Docking.rar, downloaded 498 times

---

Subject: another thing
Posted by kohait00 on Fri, 05 Mar 2010 08:34:52 GMT
View Forum Message <> Reply to Message

hhi mrjt,

another beauty issue is the following. if one locks the Dock stuff, the DockCont's title bar of tabbed

docks disappears. but i would be good to still have them there, they provide a good to read title, though the tabbars normally have the same info. but the user is used to have title on top..

is there a possib to do this selectable?

i think i found the place where to dig:
to unconditionally disable the setting of the nullframe
one needs to comment !lock &&

DockCont.cpp:833

```
void DockCont::SyncFrames(bool lock)
{
 bool frames = /*!lock && */(IsDocked() || IsAutoHide());
 handle.Show(frames);
 if (frames)
  SetFrame(0, Single<DockCont::DockContFrame>());
 else
  SetFrame(0, NullFrame());
}
```

i would be great to have something like an additional flag

(!lock || base->PermamentTitleBar()) && ...

---

Subject: Re: another thing
Posted by mrjt on Tue, 09 Mar 2010 10:42:48 GMT
View Forum Message <> Reply to Message

Thank you for your fixes/additions! I have uploaded them all to the SVN now.

I have added the option 'ShowLockedHandles' to the DockWindow class. This leaves the handles visible but hides the buttons and prevents undocking. I haven't figured out why the close button doesn't disappear off floating windows straight away, I'll have to dig into the CtrlCore code to fix that.

I have also integrated Hotkey support. See DockableCtrl::SetHotKey functions. You assign a key (or key function from .key file) to a DockableCtrl and the user can then press this key to:
- Hide a window if it's visible
- Restore a window to it's last position if it's been hidden

Unfortunately this has led to me discovering a slight imperfecting in my docking routines that causes repeatedly docked/hidden windows to gradually shink. It's going to be a real PITA to fix that one

---

Subject: Re: another thing
Posted by kohait00 on Fri, 09 Apr 2010 07:47:58 GMT
View Forum Message <> Reply to Message

hey mrjt, i didnt quite understand what the ShowLockedHandles is for and how it works. could you describe a use case?

fot the sake of locking some tabs to prevent detaching in tabify view, while others still beeing able to detach i made the following stuff (see attach)

if a DockableCtrl has no allowed dockable[] than detaching is neither permited. the case seems to be logical, but a decent lock button (pin or something) should be available to fix *one* participant only, to prevent its accidental moving or changing in position/size. what is your oppinion?

another thing is: what about having a 5th mode of docking? something like DOCK_FULL, which lets the participants be docked/tabified in fullscreen, not only left/right and top bottom. due to SizePosHint(), the appearance in current setups can look messy (i.e. docking stuff at top to a docking which is setup inside a splitter, so the splitter)

## File Attachments

1) Docking.rar, downloaded 433 times

Subject: Re: another thing
Posted by kohait00 on Fri, 09 Apr 2010 13:00:13 GMT
View Forum Message <> Reply to Message

hi mrjt

DockWindow.cpp:331

```
 if (DockCont *c = TabifyGroup(group)) {
  if (c->IsDockAllowed(align))
   DockContainer(align, *c, pos);
  else
   FloatContainer(*c);
 }
```

should be

```
 if (DockCont *c = TabifyGroup(group)) {
  DockContainer(align, *c, pos);
 }
```

because, as far as i understand, most actions through code API should be available even if user has selected stuff to bi unavailable (but if the code sais so, it has precedence
i.e. you want to dock in code a window that has been disabled by user a docking ability, but the user pressed "default arrangement" or something, where you need to make tabbing docking (which has been disabled by user)

does that make sense?

---

Subject: Re: another thing
Posted by kohait00 on Sun, 02 May 2010 07:03:06 GMT
View Forum Message <> Reply to Message

Quote:
another thing is: what about having a 5th mode of docking? something like DOCK_FULL, which lets the participants be docked/tabified in fullscreen, not only left/right and top bottom. due to SizePosHint(), the appearance in current setups can look messy (i.e. docking stuff at top to a docking which is setup inside a splitter, so the splitter)

is there anything new about it?

---

Subject: Re: another thing
Posted by mrjt on Wed, 05 May 2010 14:01:28 GMT
View Forum Message <> Reply to Message

I'm not going to ever implement that one I'm afraid. I don't want to interfere with the contents of the window view/client area, that area should be controlled by the application IMO.

---

Subject: Re: another thing
Posted by kohait00 on Thu, 06 May 2010 18:30:46 GMT
View Forum Message <> Reply to Message

it would be great though,  remember those icons when dragging stuff in visual studio i.e. besides showing left right top bottom for where to dock stuff as frame there is also a middle part... as far as i remember.. nevermind, maybe i can get it to a point where i can supply some size info upon adding or so, so that top docks use all the space down to bottom dock border when available.. trying to avoid that SizeHint() thing..

another question:

is there a way to be notified of which DockableCtrl has been selected or dragged or sth..? i noticed Callback WhenState, what is it used for exactly? and can i place it in TabSelected()?

(i tried it  it works fine so far (though beeing called multiple times during inits) but it always calls it for the first DockCont when i start dragging a tabified group..thats kind a not that what i need.

btw. could you somehow line up your ideas behind Docking / Tabifying? which class is involved for and how?

is it right that each DockableCtrl is placed in a DockCont, and what happens if more DockableCtrl are tabbed to the DockCont and when starting dragging the group (only the first DockableCtrl is beeing notified with WhenState()..)

thx in advance

---

Subject: Title() Bug
Posted by kohait00 on Fri, 07 May 2010 11:41:52 GMT
View Forum Message <> Reply to Message

heres another small bug:

changing Title() of a DockableCtrl, which is tabbed somewhere does change the caption of the tab in DockTabBar, but it does not trigger a recomputation of size needed for the tab, so when a bigger title is selected, it draws over the other tabs.

---

Subject: Re: Title() Bug
Posted by mrjt on Mon, 17 May 2010 11:35:36 GMT
View Forum Message <> Reply to Message

I've fixed the title issues in SVN revision 2403. Floating windows now also update their titles, I'm not sure why I didn't do this properly last time

WhenState is executed every time a DockableCtrl changes state (eg from docked to floating, or is close, or is opened). There was a case where it wasn't working (when dragging into tabs), but I've fixed that now. It does correctly call all child tabs when a DockCont changes state.

The idea of the Docking package was to implement docking in a completely Upp style way. So everything would be owned by something (no passing around naked heap pointers), the interface would be based around property methods and it would be relatively simple for the programmer to use. The most difficult bit was that it would sit as an external package to CtrlLib, so there are no alterations or special cases in CtrlLib to accomodate it.

The programmer uses the light-weight DockableCtrl class to wrap the essential requirements for docking around whatever GUI componenets he wants. This can be done by inheritance or by using is as a ParentCtrl with a layout and the class is really just a data container (for size & state data etc) and has almost no control over the docking process. A reference is then passed to the DockWindow and then everything more-or-less takes care of itself.

DockCont is the interesting class. This is a container for one or more DockableCtrl classes and is strictly internal to the Docking system. Unlike DockableCtrl it is non-persistent and they are created/destroyed as required by user actions. For instance: When a DockCont is dragged into another DockCont the tabs from the first are added to the second and then it is destroyed. When a DockableCtrl is dragged out of a DockCont a new one is created to hold it.

What makes this more complicated is that a DockCont can also be 'nested' as a tab in another DockCont. To facilitate this all controls that are owned by a DockCont are actually cast to Values and stored directly in the TabBar derived DockTabBar (all the DockCast and ContCast calls in DockCont.cpp are casting back from these Values). The DockTabBar class knows about these two classes and can draw them into tabs with the correct title and icon.

DockCont really is the 'nastiness' that I have tried to hide from the programmer-fronting interface. There are a lot of complications here such as the fake title bar when docked, managing all the possible user interactions and avoiding infinite recursions (you would not believe how many of these I got during development!).

DockWindow handles all of the wide scope docking control. It's responsible for creating and destroying the DockConts, storing all the global settings and coordinating drag-drops when signalled by the related DockCont. There is a lot of stuff in there to cater for as many user requirements as possible.

There are some other helpful classes: DockMenu is my attempt to make all of the Docking context menus as customizable as possible without complex inheritance, DockConfig is the config window (mainly intended as an example) and DockPane is a modified SplitterFrame that (tries) to do intelligent repositioning based the size hints from DockableCtrls.

TabBar is an external class that can be used completely independently of Docking. This grew out of TheIde QuickTabs package by unodgs as my requirements grew more complex. The TabBar ctrl is pretty awesome IMO

I hope that's explained some of it (though probably not very well). I'll happily answer any more specifi questions you have. It's a very complex package and I'm quite poroud of it for that reason.

---

Subject: Re: Title() Bug
Posted by kohait00 on Mon, 17 May 2010 19:59:58 GMT
View Forum Message <> Reply to Message

many thanks for the details..i think with this bit of background infos it'll be easier to dig through code when necessary. ofcorse you can be proud of it. the Docking stuff is pretty helpful to a lot of people including me.

one thing remains though. the SizeHint problem. i try to explain you my usecase. maybe i use it in the wrong way.

* i use a vertical Splitter to have a resizeable frame area for overview and controling, residing on

the left.

* the right part Splitter'ed again, but horizontally, giving a top view and a bottom view for some components. there can be a variety of them (device views at top and group views of devices at bottom)/

* to be able to detach some views from main window, i use 2 DockWindow's as members of my application class. they are Add()ed inside the splitter at top and bottom. (this was kind of a hassle, because DockWindow is a TopWindow derive, and placing it inside another TopWindow is maybe no good idea, DockInit() stuff was done in a separate init function, which somehow works).

* the contents of the views are added as DockableCtrl's ofcorse, with DockTop *only* and i need to give them a SizeHint.

* PROBLEM: the 2 DockWindows share the right part of application area, and are devided by splitter, but the views dont always use the full available size of their parent DockWindow up to bottom. result is kind of a messy application view altogether. because when resizeing main window, the splitters grow or shirnk and the SizeHint never matches the actual available space (from top to bottom, since there isnt anything else then top)

thats why the idea was born to have kind of a 5th mode (center, with full area usage like SizePos(), or at least to have the Docking calculate the actual possible usable Size, i.e. that a DockTop'ed control could use all the available space to the bottom or to the beginning of the DockBottom'ed controls.

any idea how to make this possible?

---

Subject: Re: Title() Bug
Posted by kohait00 on Mon, 17 May 2010 20:40:19 GMT
View Forum Message <> Reply to Message

before i update and test your version i wanted to save my current state and to have you take a look on it maybe it's helpful for you..

i extanded the DockableCtrl.h:87 just a bit
to let the API inspect some varieties of docking permissions

```
 bool   IsDockAllowed(int a) const  { ASSERT(a >= 0 && a < 4); return dockable[a]; }
 bool          IsDockAllowedLeft() const  { return dockable[0]; }
 bool          IsDockAllowedTop() const  { return dockable[1]; }
 bool          IsDockAllowedRight() const  { return dockable[2]; }
 bool          IsDockAllowedBottom() const  { return dockable[3]; }
 bool          IsDockAllowedAny() const  { for(int i = 0; i<4; i++) { if(dockable[i]) return true; } return
false; }
 bool          IsDockAllowedNone() const  { for(int i = 0; i<4; i++) { if(dockable[i]) return false; }
return true; }
```

DockConfig.cpp:221
to have animation settings extend on the other available params as well, but this is only a quickfix, maybe later it should be possible to disable each animation extra? (in current state only the first highlight animation is changeable via UI, the others remain unchanged (and maybe enabled), while the user expects all animation to be disabled, isnt it?

```
 dock.Animate(~animate, dock.IsAnimatedFrames(), ~animate);
```

DockWindow.cpp:329
the API should have precedence over UI setups.
means if I (in application code) decide to group a component to some kind, it should do it in the desired way, even if DockableCtrl's respective capabilities have been disabled in GUI.
i ended up having DockableCtrl's thrown out floating, when restoring a default view (if desired setup was disabled by user in GUI), so TabDockGroup shouldnt throw out things floating, it should group

```
void DockWindow::TabDockGroup(int align, String group, int pos)
{
 if (DockCont *c = TabifyGroup(group)) {
//  if (c->IsDockAllowed(align))
   DockContainer(align, *c, pos);
//  else
//   FloatContainer(*c);
 }
}
```

DockCont.cpp:362
another floating issue, in void DockCont::TabDragged(int ix)
if a dockable is docked somewhere and is not able to regain its position later when dragged again (because docking capability is completely disabled, it should be able to float the dockable when dragging it out of a tab (it wont be able to go back there). while manual floating command is still possible. (but in tabbed cases its a weired behaviour)

```
   if(c->IsDockAllowedNone()) return;
```

so far the little things. code lines are respect to prior to your change. maybe something of it is depricated now, maybe something is still usable..

cheers
kostah

PS attached is Docking sources including .svn to diff easy

File Attachments
1) Docking.rar, downloaded 406 times

Subject: Re: Title() Bug
Posted by mrjt on Tue, 18 May 2010 10:50:55 GMT
View Forum Message <> Reply to Message

Hmmm. The way you've got your GUI set-up doesn't isn't really fit how the Docking package was designed. The idea is that you use it to dock windows around a client area showing something useful.

Some solutions occur to me:
- Why not put one of your DockableCtrls into the client area? You might have to Deregister it first (maybe not though) and then just add it with SizePos.
- If you need the user to be able to drag into/out of the client area, I think you could do by inheriting from DockWindow and overloading ContainerDragStart/Move/End. I'm sure there would be complications though.
- You could use SetFrameSize to fill the clioent area. The main problem here is working out which frame(s) to expand (which is opartly why I didn't implement it when I looked at the problem yesterday).

Regarding your suggested changes:
DockAllowed changes: I've added these, though modified the implementation slightly. I've also added them to DockWindow.

Animation change: I've added the extra options to DockConfig, so the user can now turn the different animations on/off themselves.

TabDockGroup: I can see you point, but there are two problems:
- TabDockGroup is used by a menu in DockMenu
- What if the programmer wants to dock a group but still respect the DockAllow settings?
To solve this I've added 2 new functions that do what you want: ForceDockGroup and ForceTabDockGroup.

And if you are trying to save/restore states I suggest you use Save/LoadLayout, it should work much better.

TabDragged: This is more complicated. I can see where you're coming from but doing this would introduce an inconsistency to the interface: The user would not be able to drag individual tabs out, but they could drag the whole container (including tabs that couldn't be put back).

Besides that I don't think I like the behaviour. If what you want is to lock some DockableCtrls into particular positions that could be better accomplished in another way, maybe by a setting in DockableCtrl?

I've committed my changes to the SVN and also left you a little present in Docking.h. Hope it helps