Subject: Dockmenu Draw
Posted by luja on Thu, 01 Oct 2009 07:24:13 GMT
View Forum Message <> Reply to Message

Hello, I want to use draw methods in a dockwindow.
But it does not really work.
How to draw pictures, lines etc. in a dockWindow? The demos only give examples how to add some controls.
Any Ideas or Examples?

Subject: Re: Dockmenu Draw
Posted by Oblivion on Thu, 01 Oct 2009 08:37:30 GMT
View Forum Message <> Reply to Message

luja wrote on Thu, 01 October 2009 10:24Hello, I want to use draw methods in a dockwindow.
But it does not really work.
How to draw pictures, lines etc. in a dockWindow? The demos only give examples how to add some controls.
Any Ideas or Examples?

Hi luja,

You have at least two options:

1. Drawing/painting directly into dock window.

Although it is possible to draw stuff "directly into" docks is possible, unless you are trying to modify the dockwindow class, you shouldn't take this approach. It may further complicate your application and can cause instability/crashes if you don't really know what you are doing. So this is not a very good option.

2. Using a custom Ctrl (proper way)

Dock windows or Docks (not the base DockWindow class) are meant to be containers, so you should consider using child ctrls to draw into. This will ensure your further operations to be flexible.

How to:

First Create a Ctrl (The type of ctrl to be used is up to you, bu let's be simple and clear here)

```
class DrawControl : public Ctrl {

	// You can override the Paint() method (easiest way)

 void Paint(Draw& w) {
```

```
		// Lets draw a simple rectangle and fill our control with it.
	Size sz = GetSize();
  w.DrawRect(sz, SRed); // The color of our rectangle is red.

		// Other drawing operations, etc....

  }

};

DrawControl drawctrl;
```

Then you should add it to it's container, namely docks.

```
DockLeft(Dockable(drawctrl, "Drawing Example").SizeHint(Size(300, 200)));
```

And voila! It should work.

Using the containers (the proper way) will give you the opportunity to add/exchange/hide/remove the child ctrls on the fly.

Below is a modified example of DockingExample1. (It should compile).

```
#include <Docking/Docking.h>

/*
** This is very simple example designed to show the basics of setting up a
**  docking window without many of the compicated features
*/

/*
** Class declaration. Notice that we inherit from DockWindow (not TopWindow)
*/
class DockingExample : public DockWindow {
public:
 typedef DockingExample CLASSNAME;
 DockingExample();

private:
 ArrayCtrl arrayctrl1, arrayctrl2;
 TreeCtrl treectrl1, treectrl2;

  class drawcontrol : public Ctrl {
```

```
    void Paint(Draw& w) {
     Size sz = GetSize();
     w.DrawRect(sz, SRed);
   }

  };
 drawcontrol drawctrl;
 Button button;

 // Our initialization function
 virtual void DockInit();

 // Functions to put data in our ctrls
 void FillArray(ArrayCtrl &array);
 void FillTree(TreeCtrl &tree);
};

/*
** Source
*/
DockingExample::DockingExample()
{
 Title("DockingExample1 : Simple Docking");

 // This a button so that we can open the Dock Manager
 Add(button.SetLabel("Manager").LeftPosZ(4, 100).TopPosZ(4, 23));
 button <<= THISBACK(DockManager);

 // Now we just put some bogus data in our controls
 FillArray(arrayctrl1);
 FillTree(treectrl1);
 FillArray(arrayctrl2);
 FillTree(treectrl2);
}

void DockingExample::DockInit()
/* This is the docking initialisation function. It gets called after the window is opened,
**  and it's here you should do your docking/serialization, if you do it in the contructor
**  you get various problems (incorrect layout, floating windows not opening).
*/
{
 // Here we add our dockable controls in the simplest way possible
 // It's usually best to set a minimum size as Upp ctrls aren't very good
 //  at guessing by themselves
 DockLeft(Dockable(arrayctrl1, "ArrayCtrl 1").SizeHint(Size(300, 200)));
 DockLeft(Dockable(treectrl1, "TreeCtrl 1").SizeHint(Size(300, 200)));
 DockTop(Dockable(arrayctrl2, "ArrayCtrl 2").SizeHint(Size(300, 200)));
 DockRight(Dockable(treectrl2, "TreeCtrl 2").SizeHint(Size(300, 200)));
```

```
 // Drawing Example
 DockLeft(Dockable(drawctrl, "Drawing Example").SizeHint(Size(300, 200)));

 /*
 ** Not bad for four lines of code.
 **
 ** Some notes about the above:
 ** Before a control can be dockable it must be added to a DockableCtrl.
 **  Passing a ctrl to Dockable() does this for us and returns the DockableCtrl so
 **  so that we can set size hints etc. If you wanted to use a Upp Layout you
 **  could declare it as WithMyLayout<DockableCtrl> and use that instead.
 ** We then pass this DockableCtrl to DockLeft/DockTop (or right/bottom) to add
 **   it to the window
 */

}

/*
** Everything after here is fluff
*/
void DockingExample::FillArray(ArrayCtrl &array)
{
 array.AddColumn("Number");
 array.AddColumn("Roman numbers");
 array.MultiSelect();
 for(int i = 0; i < 200; i++)
  array.Add(i, FormatIntRoman(i, true));
}

void DockingExample::FillTree(TreeCtrl &tree)
{
 Vector<int> parent, parent2;
 parent.Add(0);
 tree.SetRoot(Image(), "The Tree");
 for(int i = 1; i < 10000; i++) {
  parent.Add(tree.Add(parent[rand() % parent.GetCount()], Image(),
             FormatIntRoman(i, true)));
  if((rand() & 3) == 0)
   tree.Open(parent.Top());
 }
 tree.Open(0);
}

GUI_APP_MAIN
{
 DockingExample().Run();
}
```

Regards,

Oblivion
