Subject: undesired behavior using custom styled MenuBar
Posted by bushman on Sat, 24 Nov 2012 00:09:46 GMT
View Forum Message <> Reply to Message

Hi, folks!

I'm getting an undesired behavior after applying a custom styled MenuBar to a TreeCtrl. My custom MenuBar pops up unexpectedly when I change from a selected TreeBar item to the next using keyboard arrow keys. I put a case code below for your reference and help:

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

using namespace Upp;

class MyMenuBar : public MenuBar
{
 public:
 typedef MyMenuBar CLASSNAME;
 MyMenuBar() {
  menustyle = MenuBar::StyleDefault();
  menustyle.menutext = Color(198, 198, 0);
  menustyle.popupframe = menustyle.popupbody = menustyle.popupiconbar = Black();
  SetStyle(menustyle);
 }

 MenuBar::Style menustyle;
};

class MyApp : public TopWindow
{
 public:
 typedef MyApp CLASSNAME;

 MyApp() {
  tree.LeftPos(10, 300).TopPos(10, 200);
  TreeCtrl::Node node;
  for(int i = 0; i < 20; i++) {
   node.Set(Format("item %d", i));
   tree.Add(0, node);
  }
  tree.OpenDeep(0);
  tree.WhenBar = THISBACK(HandleTreeBar);
  Add(tree);
 }

 void HandleTreeBar(Bar& bar){
         // here I use my styled MenuBar instead of TreeCtrl std bar
  MyMenuBar mymenu;
  mymenu.Add("foo", THISBACK(ActionSink));
```

```
  mymenu.Add("bar", THISBACK(ActionSink));
  mymenu.Execute();
 }

 void ActionSink()
 {
  PromptOK("Upp");
 }

 TreeCtrl  tree;
};

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

I'm not actually sure if that's the proper way to replace the TreeBar std MenuBar by my own, so anyone please point out how to do it the right way.

Shouldn't the MenuBar pop up only when RMB pressed?

Thanks!

Subject: Re: undesired behavior using custom styled MenuBar
Posted by bushman on Sat, 24 Nov 2012 10:50:36 GMT
View Forum Message <> Reply to Message

After inspecting TreeCtrl and MenuBar codes, looks like I could hack out something that works:

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

using namespace Upp;

class MyMenuBar : public MenuBar
{
 public:
 typedef MyMenuBar CLASSNAME;
 MyMenuBar() {
  menustyle = MenuBar::StyleDefault();
  menustyle.menutext = Color(198, 198, 0);
  menustyle.popupframe = menustyle.popupbody = menustyle.popupiconbar = Black();
  SetStyle(menustyle);
 }

// Two MenuBar::Execute overrides to force MenuBar run my styled bar
 static void Execute(Ctrl *owner, Callback1<Bar&> proc, Point p) {
```

```
  MyMenuBar bar;
  proc(bar);
  bar.Execute(owner, p);
 }

 void Execute(Ctrl * owner, Point p) {
  MenuBar::Execute(owner, p);
 }

 MenuBar::Style menustyle;
};

class MyTreeCtrl : public TreeCtrl
{
 public:
 typedef MyTreeCtrl CLASSNAME;

 MyTreeCtrl() {}

// override circumventing regular TreeCtrl behavior, so that it uses my styled bar instead of std
MenuBar
 virtual void RightDown(Point p, dword flags)
 {
  TreeCtrl::RightDown(p, flags);
  if(MyWhenBar)
   MyMenuBar::Execute(GetActiveCtrl(), MyWhenBar, GetMousePos());
 }

 Callback1<Bar&>  MyWhenBar;
};

class MyApp : public TopWindow
{
 public:
 typedef MyApp CLASSNAME;

 MyApp() {
  tree.LeftPos(10, 300).TopPos(10, 200);
  TreeCtrl::Node node;
  for(int i = 0; i < 20; i++) {
   node.Set(Format("item %d", i));
   tree.Add(0, node);
  }
  tree.OpenDeep(0);
// notice here I use my WhenBar callback
  tree.MyWhenBar = THISBACK(HandleTreeBar);
  Add(tree);
 }
```

```
 void HandleTreeBar(Bar& bar){
  bar.Add("foo", THISBACK(ActionSink));
  bar.Add("bar", THISBACK(ActionSink));
 }

 void ActionSink()
 {
  PromptOK("Upp");
 }

 MyTreeCtrl  tree;
};

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

I got the feeling it is too much of a workaround for a rather trivial task! Better, simpler ideas welcome...

Thanks!
