Subject: Proposal: add IsReadOnly() handling inside ButtonOption
Posted by Mindtraveller on Sat, 01 May 2010 14:07:44 GMT
View Forum Message <> Reply to Message

Imagine a common situation: you have a number of choices and you want to make exactly one of them chosen/selected. One way to do this is to make a number of ButtonOption controls and handle switching between them. Generally it is easily done in U++.

The problem is when user clicks already "selected" ButtonOption. This way it becomes unselected, which is commonly not the best idea. The better idea is to make "chosen" button irresponsible to user clicks, while other buttons are, contrary, ready to be clicked and become "chosen".

I've made a number of attempts to make "chosen" button option irresponsible to user clicks and failed. Digging into code revealed that there is no way to make it. So I propose handling Editable/ReadOnly flag for ButtonOption. This requires a little patch which IMO won't break any compatibility yet making ButtonOption more flexible:

```
(Button.cpp)void  ButtonOption::LeftDown(Point, dword) {
 if (IsReadOnly())
  return;
 push = true;
 Refresh();
}

void  ButtonOption::LeftUp(Point, dword) {
 if (IsReadOnly())
  return;
 option = !option;
 push = false;
 UpdateActionRefresh();
}

void  ButtonOption::MouseMove(Point, dword flags) {
 if (IsReadOnly())
  return;
 bool p = !!(flags & K_MOUSELEFT);
 if(push != p) {
  push = p;
  Refresh();
 }
}
```

File Attachments

1) bopt-disable.png, downloaded 889 times



Subject: Re: Proposal: add IsReadOnly() handling inside ButtonOption
Posted by dolik.rce on Sat, 01 May 2010 16:01:26 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Sat, 01 May 2010 16:07I've made a number of attempts to make "chosen" button option irresponsible to user clicks and failed. Digging into code revealed that there is no way to make it.
Hi Pavel!

There is a way Maybe not as straightforward as your patch, but it works fine too. The trick is to check the value of ButtonOption in WhenAction callback and revert the action whenever user tries to "unclick" the button:

```
#include "CtrlLib/CtrlLib.h"
using namespace Upp;

struct App : TopWindow {
 typedef App CLASSNAME;
 Array<ButtonOption> btns;
 void Check(int n){
  if (~btns[n]==0) btns[n].Set(true);
  else for(int i = 0; i < btns.GetCount(); i++)
   if(i!=n) btns[i].Set(false);
 }
 App(){
  btns.Add().SetLabel("Click me! :-)").SetRect(10,10,200,50);
  btns.Add().SetLabel("No, click me! (-:").SetRect(10,70,200,50);
  for(int i = 0; i < btns.GetCount(); i++){
   btns[i]<<=THISBACK1(Check,i);
   Add(btns[i]);
  }
 }
};

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

Subject: Re: Proposal: add IsReadOnly() handling inside ButtonOption
Posted by Mindtraveller on Sat, 01 May 2010 20:57:07 GMT
View Forum Message <> Reply to Message

Thanks, that does the trick.
But IMO adding read-only state for ButtonOption makes it more functional after all, which could be useful and save us from additional tricks.

Subject: Re: Proposal: add IsReadOnly() handling inside ButtonOption
Posted by Mindtraveller on Wed, 05 May 2010 20:35:08 GMT
View Forum Message <> Reply to Message

::up::

Question to U++ authors: are you agree with proposal/patch?

Subject: Re: Proposal: add IsReadOnly() handling inside ButtonOption
Posted by mirek on Thu, 06 May 2010 18:31:11 GMT
View Forum Message <> Reply to Message

Yes, patch applied, thank you.

Mirek