Subject: How to stick focus on a control
Posted by zsolt on Wed, 25 Jan 2006 21:20:48 GMT
View Forum Message <> Reply to Message

I want to avoid leaving a control until user doesn't type valid value in it. And I want to show some exclamation if he wants to leave the control.
Is there a standard way to do that?

I tried overriding LostFocus() and ChildLostFocus() of the control, but the problem is wery bizarre:

The user clicks into a (e.g.) DocEdit.
Then the next code runs:

```
void DocEdit::LeftDown(Point p, dword flags) {
 SetFocus();
 PlaceCaret(GetCursor(Point(p.x - 1, p.y + sb - 1)), flags & K_SHIFT);
 SetCapture();
}
```

So my original control losts focus, runs its own exclamation thing and sets focus back to himself. After this the DocEdit captures the cursor and user can not click to anywhere.

The problem can be solved if I place SetCapture() into the first line of DocEdit::LeftDown() and call ReleaseCtrlCapture(); in the original control's LostFocus() and ChildLostFocus().

So for me, the best solution would be to move "SetCapture()" line to the top of the method, but I don't know if it breaks any other things?

Any Idea?

Subject: Re: How to stick focus on a control
Posted by mirek on Wed, 25 Jan 2006 21:31:52 GMT
View Forum Message <> Reply to Message

Well, seems like one of those numerous cases that does not have the right solution....

Moving SetCapture would indeed help, but I do not like the idea that we should be generally THAT MUCH cautios about SetFocus calling order. Means I would rather advise not calling SetFocus in the LostFocus code...

For this kind of situation, U++ has "Accept" virtual method. It is not the same solution, as it actually allows you to move the focus, but it would not allow you to accept the dialog (e.g. via OK button).

If you insist on not-moving the focus, the one correct solution I see is to disable all sibling Ctrls (disabled Ctrl will not get one..).

Subject: Re: How to stick focus on a control
Posted by zsolt on Wed, 25 Jan 2006 22:33:52 GMT
View Forum Message <> Reply to Message

Yes, I use this code in LostFocus():

```
TreeCtrl::ChildLostFocus();
if(!HasFocusDeep())
 Accept();
```

And Accept() checks the user entered data and calls SetFocus() after exclamation. But I had to change that SetCapture() positions to be working.

Subject: Re: How to stick focus on a control
Posted by zsolt on Wed, 25 Jan 2006 23:52:13 GMT
View Forum Message <> Reply to Message

Maybe the right solution would be to implement a new method in Ctrl class (or use Accept()???). This method should be called becore killing cursor to determine if it is allowed to move focus away.
It could be implemented in Ctrl::SetFocus0(), I think.

This behaviour should be switchable using a flag in Ctrl class, of course.

Subject: Re: How to stick focus on a control
Posted by mirek on Thu, 26 Jan 2006 19:30:54 GMT
View Forum Message <> Reply to Message

Actually, I spend some time thinking about this trouble... and came to conclusion that this is the case we have encountered (and solved) a couple of times ago.

The trouble is that you actually would need to perform needed actions AFTER DocEdit::LeftDown (or other routine that moves focus as the reaction to user input event) finishes. In other words, after input event is processed and all widgets are stable again.

U++ has a tool for this: PostCallback. PostCallback records a callback that has to be invoked when processing of the input queue is finished (queue is emptied) (it actually works by placing callback to timer queue with zero delay).

So what in fact you need to do is use this to post callback, then release capture and you should be fine...

Mirek

Subject: Re: How to stick focus on a control
Posted by zsolt on Fri, 27 Jan 2006 00:47:48 GMT
View Forum Message <> Reply to Message

Thanks! This is a wery good idea.
It's working well, the only additional thing is that I had to add a ReleaseCtrlCapture() call to my (post called) callback containing SetFocus().

Subject: Re: How to stick focus on a control
Posted by mirek on Fri, 27 Jan 2006 08:51:09 GMT
View Forum Message <> Reply to Message

Also, be careful about dangling Callback here - either supply the right "id" to PostCallback or use TimeCallback or PTEBACK...