Subject: Holding a Button down
Posted by Mirari on Sat, 04 Apr 2009 04:55:05 GMT
View Forum Message <> Reply to Message

Hi,

Completely new user. I'm using U++ to develop a GUI application to control a Wireless Car for university. I've been searching for the past hour or so on whether or not there is anyway to detect if a button has been held down.

For example, I have a button called "forward" and at this point I simply want it to increase an integer while its held down. Obviously the IsPush() method is protected so the code below won't work but you can see what I'm trying to accomplish. Is there any method I could use to replace the IsPush() part so that i is continuously incremented as long as the button is held down?

```
void RouterAppWindow::Forward() {
 while (forward.IsPush() == true) {
  i++;
 }
 text = Format("%d", i);
 Refresh();
}
```

Any help will be appreciated.

Subject: Re: Holding a Button down
Posted by mrjt on Sun, 05 Apr 2009 07:39:05 GMT
View Forum Message <> Reply to Message

I can think of two ways.

1- Use the WhenRepeat callback:

```
CtrlLibTest::CtrlLibTest()
{
 CtrlLayout(*this, "Window title");
 btn <<= THISBACK(OnPushEnd);
 btn.WhenRepeat = THISBACK(OnRepeat);
 count = 0;
}

void CtrlLibTest::OnRepeat()
{
 ++count;
 label = AsString(count);
}
```

```
void CtrlLibTest::OnPushEnd()
{
 count = 0;
}
```

2- Create your own button type and add a new Callback:

```
struct HoldButton : public Button {
 Callback WhenPush;
 virtual void LeftDown(Point p, dword keyflags) {
  Button::LeftDown(p, keyflags);
  if (IsPush())
   WhenPush();
 }
};
```

The first method is nice because it doesn't require you to start a timer to determine how many 'ticks' the button has been pushed for, but you don't have much control over the repeat speed (you can set it globally but this might make other bits of the GUI work less well). There is also a delay
The second method will require a timer (SetTimeCallback) to count the ticks. The normal WhenAction callback is triggered on LeftUp so you can use that to detemine when the user stops pushing the button.
Or you can combine both approaches for the best of both worlds (keeping the count internally in your new Ctrl class).

James
