Subject: Keyboard/Input focus: MS windows vs linux
Posted by Paddy Leahy on Thu, 15 Aug 2013 00:06:51 GMT
View Forum Message <> Reply to Message

I've run into an incompatibility between windows and linux in the handling of focus events. Can anyone think of a workaround for the following?

I'm developing a widget application with a draw widget. To allow the user to use standard IDL direct graphics while the application is running, it swaps the graphics state, including colour tables and active window, every time it gains or loses keyboard focus.

The draw widget has motion events enabled. Just as one would wish, IDL generates these as soon as the cursor is positioned over the widget, even if the widget does not have focus. But this does mean that the event handler has to look out for this condition, swap the graphics state, and explicitly grab focus using:

```
        WIDGET_CONTROL, widget_id, /INPUT_FOCUS
```

Under linux, if widget_id is the draw window, this generates a KBRD_FOCUS_EVENT with ENTER = 1 which reaches the top-level base event handler, and the system knows that the widget has gained focus, so when a different window is clicked it generates a KBRD_FOCUS_EVENT with ENTER = 0, cueing a graphics state swap.

But under MS windows, no KBRD_FOCUS_EVENT is generated (at least, none reaches the top-level handler) and the system still regards the focus as being elsewhere, so the graphics state is not swapped back when focus is transferred to another window.

On the other hand, if widget_id is the top-level base, windows does, but linux
does not, generate a KBRD_FOCUS_EVENT. Unfortunately, even then windows doesn't seem to recognise that the widget has focus (unless I click on it), so no ENTER = 0 event is generated when I click elsewhere.

I'm running IDL 7.0 under windows. The behaviour under linux is the same for IDL 7.0 and 8.2, and for different x-windows systems, so I think the key factor is unix vs microsoft.

Subject: Re: Keyboard/Input focus: MS windows vs linux
Posted by David Fanning on Thu, 15 Aug 2013 01:13:09 GMT
View Forum Message <> Reply to Message

Paddy Leahy writes:

>
> I've run into an incompatibility between windows and linux in the handling of focus events. Can anyone think of a workaround for the following?
>
> I'm developing a widget application with a draw widget. To allow the user to use standard IDL direct graphics while the application is running, it swaps the graphics state, including colour tables

and active window, every time it gains or loses keyboard focus.
>
> The draw widget has motion events enabled. Just as one would wish, IDL generates these as soon as the cursor is positioned over the widget, even if the widget does not have focus. But this does mean that the event handler has to look out for this condition, swap the graphics state, and explicitly grab focus using:
>
> WIDGET_CONTROL, widget_id, /INPUT_FOCUS
>
> Under linux, if widget_id is the draw window, this generates a KBRD_FOCUS_EVENT with ENTER = 1 which reaches the top-level base event handler, and the system knows that the widget has gained focus, so when a different window is clicked it generates a KBRD_FOCUS_EVENT with ENTER = 0, cueing a graphics state swap.
>
> But under MS windows, no KBRD_FOCUS_EVENT is generated (at least, none reaches the top-level handler) and the system still regards the focus as being elsewhere, so the graphics state is not swapped back when focus is transferred to another window.
>
> On the other hand, if widget_id is the top-level base, windows does, but linux
> does not, generate a KBRD_FOCUS_EVENT. Unfortunately, even then windows doesn't seem to recognise that the widget has focus (unless I click on it), so no ENTER = 0 event is generated when I click elsewhere.
>
> I'm running IDL 7.0 under windows. The behaviour under linux is the same for IDL 7.0 and 8.2, and for different x-windows systems, so I think the key factor is unix vs microsoft.

I'd say you were probably hosed. But, this seems a little clunky to
mean, and I'd do this another way. I'd store the graphics state in the
user value of the draw widget. Then, at the time you make the draw
widget the current graphics window to draw into it, restore it's graphic
state. That way it is always ready to go when you want to use it.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: Keyboard/Input focus: MS windows vs linux
Posted by Paddy Leahy on Thu, 15 Aug 2013 02:10:40 GMT
View Forum Message <> Reply to Message

On Thursday, 15 August 2013 02:13:09 UTC+1, David Fanning wrote:

> I'd say you were probably hosed. But, this seems a little clunky to
>
> mean, and I'd do this another way. I'd store the graphics state in the
>
> user value of the draw widget. Then, at the time you make the draw
>
> widget the current graphics window to draw into it, restore it's graphic
>
> state. That way it is always ready to go when you want to use it.
>
>
>
> Cheers,
>
>
>
> David
>
>
>
> --
>
> David Fanning, Ph.D.
>
> Fanning Software Consulting, Inc.
>
> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

That's more or less what I do. The problem is not getting the right graphics state for the widget, it is restoring the previous state when I go back to the IDL command line. If that doesn't happen, random plots tend to scrawl all over the widget instead of going to a normal graphics window. Obviously I could use WSET, TVLCT etc on the command line if necessary, or just remember to click on the widget if I happen to run the cursor over it, but that's clunky and shouldn't be necessary (and isn't, under linux).

Subject: Re: Keyboard/Input focus: MS windows vs linux
Posted by David Fanning on Thu, 15 Aug 2013 02:35:21 GMT
View Forum Message <> Reply to Message

Paddy Leahy writes:

> That's more or less what I do. The problem is not getting the right graphics state for the widget, it is restoring the previous state when I go back to the IDL command line. If that doesn't happen, random plots tend to scrawl all over the widget instead of going to a normal graphics window. Obviously I could use WSET, TVLCT etc on the command line if necessary, or just remember to

click on the widget if I happen to run the cursor over it, but that's clunky and shouldn't
be necessary (and isn't, under linux).

I would write a "draw" procedure for each of your widget windows. When
you want to draw graphics in that window it first saves the current
graphics state, makes itself the current graphics window, draws its
content, then restores the graphics state it entered with. Simple,
compact, leaves no mess, and works everywhere. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Subject: Re: Keyboard/Input focus: MS windows vs linux
Posted by Paddy Leahy on Sat, 17 Aug 2013 15:36:44 GMT

View Forum Message <> Reply to Message

On Thursday, 15 August 2013 03:35:21 UTC+1, David Fanning wrote:
> Paddy Leahy writes:
>
>
>
>> That's more or less what I do. The problem is not getting the right graphics state for the widget, it is restoring the previous state when I go back to the IDL command line. If that doesn't happen, random plots tend to scrawl all over the widget instead of going to a normal graphics window. Obviously I could use WSET, TVLCT etc on the command line if necessary, or just remember to click on the widget if I happen to run the cursor over it, but that's clunky and shouldn't
>
> be necessary (and isn't, under linux).
>
>
>
> I would write a "draw" procedure for each of your widget windows. When
>
> you want to draw graphics in that window it first saves the current
>
> graphics state, makes itself the current graphics window, draws its
>

> content, then restores the graphics state it entered with. Simple,
>
> compact, leaves no mess, and works everywhere. :-)
>
>
>
> Cheers,
>
>
>
> David
>
>
>
>
>
>
>
> --
>
> David Fanning, Ph.D.
>
> Fanning Software Consulting, Inc.
>
> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

Thanks David, that worked. I was trying to avoid doing that because originally (5 years ago) this code was supposed to work with Direct Color visuals & private colour maps, but that never worked well and doesn't seem to work at all on modern systems.