

Re: Cursor libraries

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2004-07/1148.html

From: Dan Pop (*Dan.Pop_at_cern.ch*)

Date: 07/09/04

Date: 9 Jul 2004 13:51:36 GMT

In <10et3r293ujh35e@corp.supernews.com> Thomas Dickey <dickey@saltmine.radix.net> writes:

>Dan Pop <Dan.Pop@cern.ch> wrote:

>> In <10er4jhhdq6ch0b@corp.supernews.com> Thomas Dickey <dickey@saltmine.radix.net> writes:

>

>>>Dan Pop <Dan.Pop@cern.ch> wrote:

>>>> In <10eqff2fhu2d2cb@corp.supernews.com> Thomas Dickey <dickey@saltmine.radix.net> writes:

>>>>

>>>> Nope, it doesn't. When talking about the curses library, without any

>>>> qualifiers, one refers to the common subset of features.

>>>>

>>>>for that sort of attitude, you shouldn't post in comp.lang.c

>

>> Why? What's wrong with advocating portable programming in c.l.c?

>

>rephrasing your comment, let's simply code against the common denominator

>of K&R C, C89 and C99.

Exactly, when the code **must** work on K&R C, C89 and C99 implementations.
What's wrong with that?

>>>>>> Another approach is to find an old BSD documentation of the curses

>>>>>> library: every other implementation is a superset of the original

>>>>>> BSD curses library.

>>>>>>

>>>>>>not really. There are analogous features, but it is not compatible.

>>>>

>>>>> Concrete examples, please.

>>>>

>>>>getch behaves differently (as you would know if you'd done much porting).

>

>> All my curses-based code simply worked anywhere I tried it. But I used

>> only the most basic features and had no problems with getch behaving

>> differently (as long as we're talking about the curses getch).

>

>BSD getch doesn't do a refresh, doesn't provide for returning function keys.

>So it's not compatible.

You're missing my point which was that, for portable code, you have to rely on the original BSD specification. Which means that you cannot support function keys. It's the age old trade off between functionality and portability.

```
>>>delwin also.  
>  
>> Never used it. All the descriptions I'm familiar with say basically the  
>> same thing:  
>  
>> Calling delwin deletes the named window, freeing all memory  
>> associated with it (it does not actually erase the window's  
>> screen image). Subwindows must be deleted before the main  
>> window can be deleted.  
>  
>some of the BSD implementations erase the window (more than one that I  
>used in the late 80's).
```

So, clear the window before calling delwin and you couldn't care less about whether delwin clears the window or not.

```
>>>The low-level flags (cbreak, crmode, etc) are also analogous but not compatible.  
>  
>> I've asked for *concrete* examples, not for hand waving. Saying "behaves  
>> differently" is not what I call *concrete*. "Analogous but not  
>> compatible" even less.  
>  
>There's no 1-1 mapping between the half-cooked modes in the BSD and SysV  
>implementations (they're different enough to notice). That's generally  
>because the BSD code uses sgty, which has fewer bits to work with. So  
>a program would have to deal with different control characters that wouldn't  
>be filtered by the library's settings.
```

Don't use half-cooked modes. In a full screen application, you really want raw mode, so that you have full control over what happens when the user presses a key.

Dan

```
--  
Dan Pop  
DESY Zeuthen, RZ group  
Email: Dan.Pop@ifh.de
```