Chapter 5 Dialogs & the Dialog Manager

From:Ben Cranston <zben@Trantor.UMD.EDU
Subject: Re: Enter/Return in Modeless dialog

> ...IM-I p. 417 has some pretty good detail on how to interpret events
> in modeless dialogs. What [one] usually [does] is to call IsDialogEvent
> each time through the event loop. If it returns TRUE, go ahead and do any
> special processing of keys...

Some time ago I reported a gotcha about having to trap off activate events before IsDialogEvent in order to properly hilight the scroll bar of a List Manager list embedded in a modeless dialog. I got bit by another gotcha last week. If a modeless dialog event is frontmost, Multifinder events (like suspend and resume) are also considered "Dialog Events".
Thus, you have to check for THEM before calling IsDialogEvent...

The documentation on I-416/7 is still exactly correct:

'If theEvent is an activate or update event for a dialog window, a mouse-down event in the content region of an active dialog window, OR ANY OTHER TYPE OF EVENT WHEN A DIALOG WINDOW IS ACTIVE, IsDialogEvent returns TRUE...'

Emphasis added. ANY type of event. Caveat Codor.
--
Ben Cranston <zben@Trantor.UMD.EDU>

●●●

From: oster@dewey.soe.berkeley.edu (David Phillip Oster)
Subject: Re: Informational dialog help (drawdialog stuff with code)

In article <1850@neoucom.UUCP> sam@neoucom.UUCP (Scott A. Mason) writes:

>I would like to display an informational dialog box that contains
>predefined static text items only. While this box is on the screen, I want
>to call a few procedures in my C program. When the procedures finish, I
>want the box to go away. Sounds pretty simple.

```
Handle GetDIHandle(i)Integer i;{
    short theType;
    Handle theHandle;
    Rect theRect;

    GetDItem(thePort, i, &theType, &theHandle, &theRect);
    return theHandle;
}

DialogPtr gMesgDia;

BeginMessage(){
    GrafPtr savePort;

    GetPort(&savePort);
    if(NIL == (gMesgDia = GetNewDialog(resID, NIL, (WindowPtr) -1L))){
        return;
    }
    SetPort(gMesgDia);
    SetIText(GetDIHandle(textITEM), (StringPtr) "\pYour message here.");
    ShowWindow(thePort);
    DrawDialog(thePort);
    SetPort(savePort);
}

EndMessage(){
```

```
if(NIL != gMesgDia){
    DisposDialog(gMesgDia);
```

```
        gMesgDia = NIL;
    }
}
```

Now, just say:

```
BeginMessage();
... do your work ...
EndMessage();
```

You might also look at your event loop, and see if you are callingmisDialogEvent() and DialogSelect() there. You should be. See Inside Mac for more info on those two calls.

--- David Phillip Oster -- No, I come from Boston. I just work

●●●

From: blob@apple.com (Brian Bechtel)
Subject: Re: Password a'la AppleShare (Dialog Filter with code)

> I kept this code from the last time that this question was asked.
> >From: matthews@eleazar.dartmouth.edu (Jim Matthews)
> >Subject: Re: suppress display of password entry
>
> In article <536@sdacs.ucsd.EDU> wade@sdacs.ucsd.EDU (Wade Blomgren) writes:
> >
> >What is the best (easiest) way to allow entry of a password on the screen
> >while suppressing the display of the actual characters in the
> >edittext item?

It actually isn't very difficult to intercept the key events and keep a hidden copy of the password string. It isn't necessary to remember any context since you have access to the TextEdit record, and it tells you what the current selection is. The following routine is a filter for a name/password dialog box. It displays bullets (ala AppleShare) and stores the real password in a global string, pwStr. It also handles hitting return or enter.

```
{ signonFilter -- dialog filter for doSignon, hides password }

FUNCTION signonFilter (dp : DialogPtr;
           VAR theEvent : EventRecord;
           VAR itemHit : integer) : boolean;
   CONST
       nameItem = 3;
       passwordItem = 4;
       bs = $08;
       tab = $09;
       cr = $0D;
       enter = $03;
       larrow = $1C;
       rarrow = $1D;
       uparrow = $1E;
       downarrow = $1F;
   VAR
       dpeek : DialogPeek;
       theChar : char;
       theStr : Str255;
       selStart, selEnd : integer;
       h : Handle;
       itemType : integer;
       box : Rect;
BEGIN
   signonFilter := false;
```

```
dpeek := DialogPeek(dp);
IF ((theEvent.what = keydown) OR (theEvent.what = autoKey)) THEN
    IF (dpeek^.editField = passwordItem - 1) THEN
    BEGIN
```

```
            theChar := char(BitAnd(theEvent.message, charCodeMask));
            selStart := dpeek^.textH^^.selStart;
            selEnd := dpeek^.textH^^.selEnd;
            CASE ord(theChar) OF
                bs :                { Backspace }
                    BEGIN
                        IF selEnd = selStart THEN  { back over a character }
                        BEGIN
                            IF selStart > 0 THEN
                             pwStr := concat(copy(pwStr,1, selStart - 1),
                                             copy(pwStr, selStart + 1,
                                              length(pwStr) - selStart));
                        END
                        ELSE            { delete the selection }
                            pwStr := concat(copy(pwStr, 1, selStart),
                             copy(pwStr, selEnd + 1,
                                   length(pwStr) - selEnd));
                    END;
                cr, enter :     { Return or Enter -- treat as "OK }
                    BEGIN
                        itemHit := ok;
                        signonFilter := true;
                    END; { cr, enter }
                tab, uparrow, downarrow, rarrow, larrow :
                    ;     { just pass on tabs & arrows }
                OTHERWISE   { "normal" character }
                    BEGIN        { remember character, insert a bullet }
                        pwStr := concat(copy(pwStr, 1, selStart),
                         theChar,
                         copy(pwStr, selEnd + 1, length(pwStr) - selEnd));
                        theEvent.message :=
                         BitAnd(theEvent.message, $FFFFFF00) + ord('%');
                    END; { normal character }
            END; { case ord(theChar) of }
        END { in password field }
        ELSE     { not in password field -- still check for cr, enter }
            CASE BitAnd(theEvent.message, charCodeMask) OF
                cr, enter :
                    BEGIN
                        itemHit := ok;
                        signonFilter := true;
                    END; { cr, enter }
                OTHERWISE
                    ;
            END; { case BitAnd }
END; { signonFilter }
```

--Brian Bechtel blob@apple.com "My opinion, not Apple's"

●●●

From: kk@mcnc.org (Krzysztof Kozminski)
Subject: Re: How can I draw contents of modeless dialog?

In article <24766@unix.cis.pitt.edu> er225711@unix.cis.pittsburgh.edu (M Kikuchi) writes:

> >I'm trying to implement a modeless dialog using complete DLOG and
> >DITL resources, but the contents of the item list are not drawn

You need to use IsDialogEvent/DialogSelect calls. Your event loop should look something like:

```
if (GetNextEvent(everyEvent,& CurrEvent)) {
   if (IsDialogEvent(&CurrEvent)) {
  /* You may want to intercept some events here */
```

```
if (DialogSelect(&CurrEvent, &theDialog, &itemHit)) {
  /* Process the event/item that got hit */
}
  } else {
/* Handle non-dialog events */
  }
```

DialogSelect will process the update/activate events for you.

For more details, see IM vol.1, pages somewhere around 400,if I remember correctly.

KK
--
Kris Kozminski kk@mcnc.org

●●●

From: kazim@Apple.COM (Alex Kazim)
Subject: Re: how can I draw contents of modeless dialog?
Summary: Tucked away in Inside Mac
Keywords: Dialog

In article <24766@unix.cis.pitt.edu> er225711@unix.cis.pittsburgh.edu (M Kikuchi) writes:

> >I'm trying to implement a modeless dialog using complete DLOG and
> >DITL resources, but the contents of the item list are not drawn

Actually, both _ModalDialog and _DialogSelect make calls to, yes,_DrawDialog (see p. I-418). I remember having trouble about four years ago, and spending all afternoon trying to find this call (it's nicely tucked away.)

Alex Kazim, Apple Computer

●●●

From: jackiw@cs.swarthmore.edu (Nick Jackiw)
Subject: Re: Floating Point #'s as Strings in Dialogs using Think C 3.0

isler@grad1.cis.upenn.edu () writes:

> > I am writing a program in Think C 3.0 in which I would like to convert
> > floating point numbers to strings and place them in EditText items in
> > a dialog box.
> > sprintf(str, "%3.1f", value);
> >
> > Any suggestions?
> > S-K ISLER

As mentioned previously, you'll need a Pascal string instead of a c-string. You may be better served by using the Toolbox, instead of C, to do your conversions for you. SANE implements the calls:

```
procedure Str2Dec(s:DecStr;var Index:integer; var d:Decimal;
                  var validPrefix:boolean);

procedure Dec2Str(f:Decform; d:Decimal; var s:DecStr);

procedure CStr2Dec(s:CStrPtr;var index:integer; var d:Decimal;
                  var validPrefix:boolean);
```

which will do all the necessary conversions for you. In addition to providing the appropriate types to your task, SANE is script-manager compatible, which may or may not be true of THINK C's sprintf. This means that if your application is for

public use, it will format (and accept formatted numbers) appropriately to the country in which it's in use.

See your SANE interface files for declarations of the individual types.

-----Nicholas Jackiw [jackiw@cs.swarthmore.edu|jackiw@swarthmr.bitnet]-----

●●●

From: stoms@castor.ncgia.ucsb.edu (David Stoms)
Subject: Re: Floating Point #'s as Strings in Dialogs using Think C 3.0

In article <25837@netnews.upenn.edu> isler@grad1.cis.upenn.edu () writes:

> >I am writing a program in Think C 3.0 in which I would like to convert
> >floating point numbers to strings and place them in EditText items in
> >a dialog box.
>
> >The result is that garbage is printed in the dialog instead of the
> >converted string. The above conversion method works fine when I am
> >displaying floats as strings in windows, but it does not work here.

```
typedef char[32] FStr;

Double2String(double num, int digits, FStr string)
   {
   char   bcdNum[12];
   int    i, j;
   char   c;

   if (!string) Debugger();

/* I believe SANE does this too */
   asm {
      fmove.x   num,fp0
      fmove.p   fp0,bcdNum{#17}
      }

   j = 1;
   c = bcdNum[3];
   if (bcdNum[0] & 0x8000) string[j++] = '-';
   string[j++] = (c & 0xf) + '0';
   string[j++] = '.';

   digits = 16 - digits;
   if (nbtwn(digits, 0, 16)) Debugger();

   for (i=4; i <= 11 - (digits>>1); i++) {
      c = bcdNum[i];
      string[j++] = (c>>4 & 0xf) + '0';
      string[j++] = (c & 0xf) + '0';
      }
   j -= digits % 2;
   /* while (string[j-1] == '0') j--; */
   if (string[j-1] == '.') j--;

   string[j++] = 'e';
   string[j++] = (bcdNum[0] & 0x4000) ? '-' : '+';

   i = false;
   if ( (bcdNum[0] & 0xf) != 0 ) {
      string[j++] = (bcdNum[0] & 0xf) + '0';
      i = true;     /* significant digit */
      }
```

```
if ( i || (bcdNum[1]>>4 & 0xf) != 0) {
    string[j++] = (bcdNum[1]>>4 & 0xf) + '0';
    i = true;
    }
```

```
    if ( !i && bcdNum[1] & 0xf == 0)
       j -= 4;
    else
       string[j++] = (bcdNum[1] & 0xf) + '0';

    string[0] = j-1;
    }
```

Josh.

●●●

From: lippin@brahms.berkeley.edu (The Apathist)
Subject: Re: Query- Double lines on default button...How?

I find that an inactive PICT item does a wonderful job of highlighting the default button. It has two advantages over drawing the ring "by hand" -- it uses no code, and it updates whenever necessary.

I even use this method in alerts -- although the dialog manager draws the outline for you, it doesn't update it if your window gets covered (by MultiFinder or a screen saver). A similar problem applies to the icons for NoteAlert, CautionAlert, and StopAlert, and in this case can be solved with an appropriate ICON item.

--Tom Lippincott

●●●

From: keith@Apple.COM (Keith Rollin)
Subject: Re: Query- Double lines on default button...How?

In article <3486@adobe.UUCP> hawley@adobe.UUCP (Steve Hawley) writes:

> >In article <9883@odin.corp.sgi.com> myoung@joker.sgi.com (Mark Young) writes:
> >>that describes how to make a button with the double lines that indicates
> >>the default selection. I found a comment indicating that item #1 was the
> >>default selection, but my "ok" button, which is item #1 doesn't appear with
> >>the telltale double lines?
> >>
> >>any clues?
> >
> >Bad news: The bold outline does NOT get drawn for you. You must do that
> >yourself.
> >
> >Inside Macintosh Volume I has some code to do that. It boils down to getting
> >the bounding rect of the item, InsetRect(&boundingRect, -4, -4), PenSize(2, 2),
> >and then calling FrameRoundRect(...) to draw it.

Following is the routine that DTS will be using in future versions of our sample code. Note that it DOES NOT use the (16, 16) method recommended by Inside Mac, which is why I'm posting it here.

```
PROCEDURE OutlineButton(button: UNIV ControlHandle);

{Given any control handle, this will draw an outline around it.  This is used for the
default button of a window.  The extra nice feature here is that I'll erase the outline for
buttons that are inactive.  Seems like there should be a Toolbox call for getting a
control's hilite state. Since there isn't, I have to look into the control record myself.
This should be called for update and activate events.

The method for determining the oval diameters for the roundrect is a  little different than
that recommended by Inside Mac. IM I-407 suggests that you use a hardcoded (16,16) for the
```

diameters. However, this only looks good for small roundrects. For larger ones, the outline doesn't follow the inner roundrect because the CDEF for simply buttons doesn't use (16,16). Instead, it uses half the height of the button as the diameter. By using this formula, too, our outlines look better.

```
WARNING: This will set the current port to the control's window.}

    CONST

    kButtonFrameSize= 3;          { button frameUs pen size }
    kButtonFrameInset= - 4;{ inset rectangle adjustment around button }

    VAR
       theRect:         Rect;
       curPen:          PenState;
       buttonOval:      integer;

    BEGIN
       IF button <> NIL THEN BEGIN
          SetPort(button^^.contrlOwner);
          GetPenState(curPen);
          PenNormal;
          theRect := button^^.contrlRect;
          InsetRect(theRect, kButtonFrameInset, kButtonFrameInset);
          buttonOval := (theRect.bottom - theRect.top) DIV 2;
          IF (button^^.contrlHilite = kCntlActivate) THEN
             PenPat(black)
          ELSE
             PenPat(gray);
          PenSize(kButtonFrameSize, kButtonFrameSize);
          FrameRoundRect(theRect, buttonOval, buttonOval);
          SetPenState(curPen);
       END;
    END;
```

--
Keith Rollin --- Apple Computer, Inc. --- Developer Technical Support

●●●

From: jackiw@cs.swarthmore.edu (Nick Jackiw)
Subject: Re: Password Dialogs--what is the best way to do them?

tj@cs.ucla.edu (Tom Johnson) writes:

> My big question is: How do I know when the user is entering text into
> the Password text edit item (as opposed to the UserName item)? The event
> record gives me no indication of which item is selected (or does it?),
> the itemHit variable doesn't tell me, and I don't where to find out.

There's a field in the DialogRecord called editField, which according to I-408 "is one less than the item number of the current editText item, or -1 if there's no editText item in the dialog." So use this and bob's your uncle.

Do beware of tabs, though. ModalDialog interprets these separately, as keys to switch from the current editText item to the next. This switching probably happens after your filterProc and before your itemHit. Make sure you ignore tabs, or handle them appropriately.

> Tom Johnson UCLA Computer Science Department

-----Nicholas Jackiw [jackiw@cs.swarthmore.edu|jackiw@swarthmr.bitnet]-----

●●●