Farben

Die Funktionen zur Farbauswahl wurden bereits in den Beispielen vorgestellt, und dort auch benutzt. Es exitieren vier Funktionen zur Farbwahl, je zwei zum Einstellen der Hintergrund- und Vordergrundfarbe. Zur Wahl steht jeweils eine Funktion zur Auswahl eines vordefinierten Farbwertes, bzw. die direkte Angabe eines RGB Wertes:

```
j_setcolor(integer japi_object, integer r, integer g, integer b)
j_setcolorbg(integer japi_object, integer r, integer g, integer b)
j_setnamedcolor(integer japi_object, integer farbe);
j_setnamedcolorbg(integer japi_object, integer farbe);
```

Die beiden ersten Funktionen setzen die Farben für Vordergrund und Hintergrund durch RGB Werte. Jeder Kanal kann einen Wert von 0 bis 255 annehmen. Die nächsten beiden Funktionen setzen vordefinierte Farben. Die Variable "farbe" kann dabei einen Wert von 0 bis 15 annehmen. Diese 16 Farben entsprechen dem alten CGA Farbmodell und sind wie folgt zugeordnet:

```
J_BLACK                 0
J_WHITE                 1
J_RED                   2
J_GREEN                 3
J_BLUE                  4
J_CYAN                  5
J_MAGENTA               6
J_YELLOW                7
J_ORANGE                8
J_GREEN_YELLOW          9
J_GREEN_CYAN           10
J_BLUE_CYAN            11
J_BLUE_MAGENTA         12
J_RED_MAGENTA          13
J_DARK_GRAY            14
J_LIGHT_GRAY           15
```

Die Angabe eines RGB Wertes ist unabhängig von der tatsächlichen Farbtiefe des Bildschirms. Die Umrechnung von der übergebenen 24 Bit Farbtiefe auf die vorhandene Farbtiefe erfolgt automatisch. Der JAPI Kernel wählt dann die jeweils ""ahnlichste" Farbe aus.

Da diese Funktionen in den vorangegangenen Beispielen bereits ausführlich benutzt und beschrieben wurden, erübrigt sich an dieser Stelle eigentlich ein Beispielprogramm. Wir wollen stattdessen die Gelegenheit nutzen, hier die Japilib durch ein selbsgeschriebene Farbdialog Element zu erweitern.

Eine Farbauswahl Dalogbox (auch Colorpicker genannt) soll die Möglichkeit bieten über drei Farbregler eine Farbe einzustellen, und dessen RGB Werte an das aufrufende Programm zurückliefern. Der Aufruf der Dialogbox sollte demnach wie folgt erfolgen:

```
j_colordialog(frame, red, green, blue);
```

Neben der Farbeinstellung sollte die Dialogbox folgende Eigenschaften haben:

- Sie sollte einen "OK" Button und einen "Cancel" Button besitzen.
- Die Rückgabe der Funktion sollte **.true.** sein, falls der "OK" Button gedrückt wurde, bei "Cancel" soll **.false.** zurückgegeben werden.
- Die Werte die in den übergebenen Variablen r,g und b stehen, sollten beim Funktionsaufruf dargestellt werden.
- In der Dialogbox sollte ein Element zur Darstellung der gewählten Farbe vorhanden sein. Dieses Element sollte mit der Dialogbox größenänderlich sein.
- Die RGB Werte sollen zudem als Zahlenwerte dargestellt werden.
- Die Colordialogbox soll ihre eigenen Events verwalten.
- Sie sollte mittig im Frame erscheinen.

Aus diesen Forderungen ergibt sich folgendes Layout:

Die Dialogbox erhält einen Borderlayoutmanager, der im Zentrum einen Canvas zu Farbdarstellung enthählt. Die RGB Scollbars werden in den linken Bereich des Borderlayouts positioniert. Dieser Bereich erhält nun ebenfalls einen Borderlayoutmanager. In dessen Kopfteil werden drei Labels, die die RGB Zahlenwerte darstellen, in einem Gridlayoutmanager positioniert. In den zentralen Bereich dieses zweiten Borderlayoutmanager werden die Scrollbars positioniert. Auch hierbei wird wiederum ein Gridlayout verwendet. Die beiden Button "OK" und "Cancel" werden in einem Flowlayoutmanager im Fussbereich der äußeren Borderlayout angeordnet.

Durch diese Anordnung hat das Labelfeld immer die gleiche Größe, und alle Label sind gleich groß. Das Scrollbarfeld hat eine konstante Breite aber immer die volle Höhe des Dialograhmens. Die Button liegen immer zentriert im Fußbereich. Der verbleibende Rest wird komplett dem Canvas zugeordnet.

Hier zunächst das komplette Listing der Funktion ***j_colordialog()***

```
C       aus Example colorpicker.f
C
C
C       Funktion Colorpicker( container , red , green , blue )
C
C
        logical function j_colorpicker(frame,r,g,b)
        integer frame
        integer r,g,b

        INCLUDE "japi.f"

        integer dialog,obj
        integer rscroll,gscroll,bscroll
        integer rlabel,glabel,blabel
        integer panel1,panel2,canvas
```

```
      integer ok,cancel
      integer xpos,ypos
      integer lr,lg,lb
      character*256 str

      call j_disable(frame)
      j_colorpicker = .false.

      dialog = j_dialog(frame,"Colorpicker")
      call j_setnamedcolorbg(dialog,J_WHITE)
      call j_setborderlayout(dialog)
      panel1=j_panel(dialog)
      call j_setborderpos(panel1,J_LEFT)
      call j_setborderlayout(panel1)

      panel2=j_panel(panel1)
      write(*,*) J_TOP
      call j_setborderpos(panel2,J_TOP)
      call j_setgridlayout(panel2,0,3)
      rlabel=j_label(panel2,"255")
      glabel=j_label(panel2,"255")
      blabel=j_label(panel2,"255")

      panel2=j_panel(panel1)
      call j_setgridlayout(panel2,0,3)
      call j_sethgap(panel2,20)
      rscroll=j_vscrollbar(panel2)
      gscroll=j_vscrollbar(panel2)
      bscroll=j_vscrollbar(panel2)
      call j_setmax(rscroll,265)
      call j_setmax(gscroll,265)
      call j_setmax(bscroll,265)
      call j_setnamedcolorbg(rscroll,J_RED)
      call j_setnamedcolorbg(gscroll,J_GREEN)
      call j_setnamedcolorbg(bscroll,J_BLUE)
      call j_setvalue(rscroll,r)
      call j_setvalue(gscroll,g)
      call j_setvalue(bscroll,b)

      panel1=j_panel(dialog)
      call j_setborderpos(panel1,J_BOTTOM)
      call j_setflowlayout(panel1,J_HORIZONTAL)
      ok = j_button(panel1,".   OK   .")
      cancel = j_button(panel1,"Cancel")

      canvas=j_canvas(dialog,200,200)

      call j_pack(dialog)
      call j_getpos(frame,xpos,ypos)
      xpos = xpos + j_getwidth(frame)/2-j_getwidth(dialog)/2
      ypos = ypos + j_getheight(frame)/2-j_getheight(dialog)/2
      call j_setpos(dialog,xpos,ypos)
      call j_show(dialog)

      obj = 0
10    if ((obj.eq.cancel) .or. (obj.eq.dialog)) goto 20
          lr = j_getvalue(rscroll)
          lg = j_getvalue(gscroll)
          lb = j_getvalue(bscroll)

          call j_setcolorbg(canvas,lr,lg,lb)
```

```
            write(str,*) lr
            call j_settext(rlabel,str)
            write(str,*) lg
            call j_settext(glabel,str)
            write(str,*) lb
            call j_settext(blabel,str)

            obj = j_nextaction()

            if(obj.eq.ok) then
                j_colorpicker = .true.
                r = lr
                g = lg
                b = lb
                goto 20
            endif
        goto 10

20      call j_dispose(dialog)
        call j_enable(frame)
        end
```

Der Aufruf der Funktion:

```
    call j_disable(frame)
```

sorgt dafür, das vom aufrufenden Programm keine Events mehr kommen können. Dies ermöglicht uns, in der Funktion eine eigene Eventloop zu programmieren. Der Aufbau des Layout dürfte selbsterklärend sein. Die Maximalwerte der Scrollbars werden auf den wert 265 gesetzt. Dadurch ergibt sich, abzüglich der Hoehe des Schiebereglers von 10, ein maximal erreichbarer Wert von 255. Damit die Dialogbox immer mittig im aufrufenden Frame erscheint, wird die Position der Dialogbox mit folgenden Funktionsaufrufen gesetzt:

```
    call j_getpos(frame,xpos,ypos)
    xpos = xpos + j_getwidth(frame)/2-j_getwidth(dialog)/2
    ypos = ypos + j_getheight(frame)/2-j_getheight(dialog)/2
    call j_setpos(dialog,xpos,ypos)
```

Dabei wird zunächst die Schirmposition des Frames ermittelt. Als der Hälfte der Breite von Frame und Colordialogbox kann die horizontale Position ermittelt werden. Die vertikale Position errechnet suich analog.

In der Eventloop wird bei jedem Event die Werte der drei Scrollbars ermittelt und die Farbe des Canvas sowie die Inhalte der Labels neu gesetzt. Diese Programmierung ist sicherlich ineffizient un könnte durch das Abfangen der Scrollbar Events verbessert werden. Da der Overhead jedoch nicht so groß ist, soll dies hier genügen.

Abschließend wird der Colordialog entfernt, und der aufrufende Frame wieder enabled:

```
    call j_dispose(dialog)
    call j_enable(frame)
```

Das folgende Beispiel nutzt den Colordialog, um seine Hintergrundfarbe einzustellen (Abbildung 5.1):

```
C       Example text.f

        :
        frame   = j_frame("Colorpicker Demo")
        menubar = j_menubar(frame)
        file    = j_menu(menubar,"File")
        color   = j_menuitem(file,"Color")
        quit    = j_menuitem(file,"Quit")

        r=0
        g=0
        b=0
        call j_setcolorbg(frame,r,g,b)
        call j_show(frame)

40      obj =  j_nextaction();

            if(obj .eq. color) then
                if(j_colorpicker(frame,r,g,b))
     &              call j_setcolorbg(frame,r,g,b)
            endif

            if((obj .eq. quit) .or. (obj .eq. frame)) goto50
        goto 40
        :
```