Wimp Function/Procedure

by Mark Moxon

In this, the second instalment of our Wimp library, the routines are concerned with icons. Before describing the routines, a quick word on the subject of libraries is probably appropriate.

Libraries are collections of routines that can be shared between many programs. A Basic library program is the same as any other Basic program, except that it only contains functions and procedures. If the command:

```
    LIBRARY "filename"
```

is included at the start of a program, then the routines in the Basic library program *filename* are automatically loaded into memory by Basic, and can be used as if they had actually been appended to the end of the program.

So whereabouts on disc should we store this library program? Well, for Wimp applications there are two main possibilities: inside the application's directory or in the disc's library. In the case of the former, the library is for the sole use of that application, but it does mean that the application is self-contained. In this case a system variable must be set up in the application's *!Run file*, like this:

```
    Set WimpLib$Dir <Obey$Dir>
```

and then the command in the *!RunImage* program that installs the library will be:

```
    LIBRARY "<WimpLib$Dir>.WimpLib"
```

If the program is put into the disc's library (normally a directory called Library in the root of a disc) then all that is needed is the line:

```
    LIBRARY "%.WimpLib"
```

in *!RunImage* to call the library, and the library can be shared by many applications. However, the application is not self contained, and will only run on systems that have the Wimp library in their library directory. I prefer the former method, and the example applications in this series use it.

In either case, the same library can be used by any applications, as the routines contained in the library are not application specific. The difference is just that in the first method each application has its own individual copy of the library, and in the second many applications share one copy.

ENTERING THE EXAMPLE APPLICATION

The library listing should be added to the library from last month, and saved as *WimpLib* as before. The line numbers are such that this month's listing continues where last month's ended.

The example application should be entered exactly as for last month (i.e. create a *!Run file*, a *!Sprites file*, save *WimpLib* inside the directory, and type in the example listing given here as *!RunImage*). Then create a sprite file called *Sprites* in the application directory containing three mode 12 sprites with height 17 and width 34, called *test1*, *test2* and *test3*. The application will display these sprites on the icon bar in sequence.

PART 2: ICONS

Routine 8: FNcreate_icon

Creates an icon definition in window *whandle*. If you have used WimpLib routines to create windows, then the origin for co-ordinates (0,0) is at the top left of the window, and positive x% and y% place the icon below and to the right of this corner. Returns icon handle.

block%	Block for SWI call
whandle%	Window handle
x%	X co-ordinate of top left corner of icon

y%	Y co-ordinate of top left corner of icon
w%	Width of icon (+ve)
h%	Height of icon (+ve)
flag%	Icon flags (see Routine 9)
text$	Icon text or sprite name
icon1%	Icon data word 1
icon2%	Icon data word 2
icon3%	Icon data word 3

Routine 9: FNicon_flags

Returns value of icon flags word.

text%	Icon contains text
sp%	Icon is a sprite
bor%	Icon has a border
chor%	Icon is centred horizontally
cver%	Icon is centred vertically
fill%	Icon has a filled background
anti%	Icon is in an anti-aliased font
redraw%	Icon cannot be redrawn entirely by the Wimp (i.e. redraw requests are returned)
ind%	Icon data is indirected
rjust%	Icon text is right justified
adj%	If selected with Adjust don't cancel others in the same ESG
half%	Icon is displayed at half size (if a sprite) button% Icon's button type
esg%	Icon's ESG number (0-31)
invert%	Icon is selected (inverted)
shade%	Icon is shaded (not selectable)
del%	Icon has been deleted
fcol%	Icon foreground colour (anti%=0)
bcol%	Icon background colour (anti%=0)
font%	Font handle for anti-aliased text (anti%=1)

Routine 10: FNload_sprites

Creates a user sprite area and loads a sprite file into it. Returns a pointer to the sprite area.

file$	Name of sprite file to load

Routine 11: FNicon_bar_icon

Places a sprite icon on the icon bar. Returns icon handle.

block%	Block for SWI call
spname$	Sprite name (must be in Wimp sprite pool)
side%	Which side of icon bar (-2 left, -1 right)

Routine 12: FNchangeable_bar_icon

Places a changeable sprite (i.e. sprite name is indirected) on the icon bar. Returns icon handle.

block%	Block for SWI call
spname$	Sprite name
side%	Which side of icon bar (-2 left, -1 right)
spname%	Pointer to buffer in which to store sprite name
sprite%	Pointer to sprite area containing sprite

Routine 13: PROCchange_icon

Changes first piece of indirected icon data (i.e. indirected text or indirected sprite name).

block%	Block for SWI call
whandle%	Window handle
ihandle%	Icon handle
new$	New indirected data (i.e. new icon text or sprite name)

!Run Image for example application

```
 10 REM        >!RunImage
 20 REM Program   Wimp Library Part 2
 30 REM Version   A 1.0
 40 REM Author    Mark Moxon
 50 REM RISC User June 1992
 60 REM Program   Subject to Copyright
 70 REM           Not Public Domain
 80 :
 90 LIBRARY "<WimpLib$Dir>.WimpLib"
100 PROCinit
110 PROCwindow
120 mask%=FNpoll_flags(1,0,1,1,0,1,1,1
,0,0,1)
130 REPEAT
140  PROCpoll
150 UNTIL quit%
160 SYS "Wimp_CloseDown",task%,&4B5341
54
170 END
180 :
 190 DEF PROCinit
 200 task_name$="WimpLib2"
 210 ON ERROR IF FNwimperror(block%,ERR
,REPORT$+" (internal error code "+STR$(E
RL)+")",task_name$,0,1)=2 THEN SYS "Wimp
_CloseDown":END
 220 quit%=FALSE:state%=0
 230 DIM block% &1000,ind% 2,spname% 20
 240 sprite%=FNload_sprites("<WimpLib$D
ir>.Sprites")
 250 SYS "Wimp_Initialise",200,&4B53415
4,task_name$ TO,task%
 260 ENDPROC
 270 :
 280 DEF PROCwindow
 290 bicon%=FNchangeable_bar_icon(block
%,"test1",-1,spname%,sprite%)
 300 wflag%=FNwindow_flags(1,1,1,0,0,0,
0,1,1,0,0,0,0,0,0,0)
 310 whandle%=FNst_create_window2(block
%+4,200,700,300,100,300,100,0,0,"Wimp Te
st",0,0,wflag%)
 320 !block%=whandle%
 330 SYS "Wimp_OpenWindow",,block%
 340 flags%=FNicon_flags(1,0,1,1,1,1,0,
0,0,0,0,0,9,1,0,0,0,7,12,0)
 350 wicon1%=FNcreate_icon(block%,whand
le%,75,5,150,48,flags%,"Change",0,0,0)
 360 flags%=FNicon_flags(1,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,7,0,0)
 370 wicon2%=FNcreate_icon(block%,whand
le%,75,53,100,48,flags%,"State:",0,0,0)
 380 flags%=FNicon_flags(1,0,0,1,1,0,0,
0,1,0,0,0,0,0,0,0,0,11,0,0)
 390 wicon3%=FNcreate_icon(block%,whand
le%,200,53,24,48,flags%,"1",ind%,0,0)
 400 ENDPROC
 410 :
 420 DEF PROCpoll
 430 SYS "Wimp_Poll",mask%,block% TO re
ason%
 440 CASE reason% OF
 450  WHEN 2     : SYS "Wimp_OpenWindow
",,block%
 460  WHEN 3     : SYS "Wimp_CloseWindo
w",,block%:quit%=TRUE
 470  WHEN 6     : PROCclick
 480  WHEN 17,18 : IF block%!16=0 THEN
quit%=TRUE
 490 ENDCASE
 500 ENDPROC
 510 :
 520 DEF PROCclick
 530 IF (block%!8=4 OR block%!8=1) AND
block%!12=-2 AND block%!16=bicon% THEN P
ROCchange
 540 IF (block%!8=4 OR block%!8=1) AND
block%!12=whandle% AND block%!16=wicon1%
 THEN PROCchange
 550 ENDPROC
 560 :
 570 DEF PROCchange
 580 state%=(state%+1)MOD3
 590 PROCchange_icon(block%,-1,bicon%,"
test"+STR$(state%+1))
 600 PROCchange_icon(block%,whandle%,wi
con3%,STR$(state%+1))
 610 ENDPROC
```

W imp Library

```
 650 :
 660 DEF FNcreate_icon(block%,whandle%,
x%,y%,w%,h%,flag%,text$,icon1%,icon2%,ic
on3%)
 670 LOCAL handle%,ist%
 680 !block%=whandle%:block%!4=x%
 690 block%!8=-y%-h%:block%!12=x%+w%
 700 block%!16=-y%:block%!20=flag%
 710 ist%=((flag%>>6)AND4)+(flag%AND3)
 720 CASE ist% OF
 730  WHEN 1,2,3:$(block%+24)=text$
 740  WHEN 5,6,7:block%!24=icon1%:block
%!28=icon2%:block%!32=icon3%:$icon1%=tex
t$
 750 ENDCASE
 760 SYS "Wimp_CreateIcon",,block% TO h
andle%
 770 =handle%
 780 :
 790 DEF FNicon_flags(text%,sp%,bor%,ch
or%,cver%,fill%,anti%,redraw%,ind%,rjust
%,adj%,half%,button%,esg%,invert%,shade%
,del%,fcol%,bcol%,font%)
 800 LOCAL flag%
 810 flag%=text%+(sp%<<1)+(bor%<<2)+(ch
or%<<3)+(cver%<<4)+(fill%<<5)+(anti%<<6)
+(redraw%<<7)+(ind%<<8)+(rjust%<<9)+(adj
%<<10)+(half%<<11)+(button%<<12)+(esg%<<
16)+(invert%<<21)+(shade%<<22)+(del%<<23
)
```

```
 820 IF anti%=0 THEN
 830  flag%+=(fcol%<<24)+(bcol%<<28)
 840 ELSE
 850  flag%+=(font%<<24)
 860 ENDIF
 870 =flag%
 880 :
 890 DEF FNload_sprites(file$)
 900 LOCAL file%,size%,sprite%
 910 file%=OPENIN file$
 920 size%=EXT#file%+16:CLOSE#file%
 930 DIM sprite% size%
 940 !sprite%=size%:sprite%!8=16
 950 SYS "OS_SpriteOp",&109,sprite%
 960 SYS "OS_SpriteOp",&10A,sprite%,fil
e$
 970 =sprite%
 980 :
 990 DEF FNicon_bar_icon(block%,spname$
,side%)
 1000 LOCAL flag%
 1010 flag%=FNicon_flags(0,1,0,0,0,0,0,0
,0,0,0,0,3,0,0,0,0,0,0,0)
 1020 =FNcreate_icon(block%,side%,0,-68,
68,68,flag%,spname$,0,0,0)
 1030 :
 1040 DEF FNchangeable_bar_icon(block%,s
pname$,side%,spname%,sprite%)
 1050 LOCAL flag%
 1060 flag%=FNicon_flags(0,1,0,0,0,0,0,0
,1,0,0,0,3,0,0,0,0,0,0,0)
 1070 =FNcreate_icon(block%,side%,0,-68,
68,68,flag%,spname$,spname%,sprite%,LEN(
$spname%)+1)
 1080 :
 1090 DEF PROCchange_icon(block%,whandle
%,ihandle%,new$)
 1100 !block%=whandle%:block%!4=ihandle%
 1110 SYS "Wimp_GetIconState",,block%
 1120 $(block%!28)=new$
 1130 !block%=whandle%:block%!4=ihandle%
 1140 block%!8=0:block%!12=0
 1150 SYS "Wimp_SetIconState",,block%
 1160 ENDPROC
```