Introducing DeskLib

Thomas Down takes a look at an alternative Wimp programming library for your C compiler

When most people think of RISC OS Wimp programming in C, they immediately think of Acorn's (now fairly old) RISC_OSLib library, which is supplied with Desktop C. An earlier version was also supplied with the old Beebug C package. However, although this library does have some powerful features, it can be very hard to implement some of the new window tools, such as pop-up menus, which are documented in the RISC OS 3 Style Guide and are appearing with increasing frequency in modern RISC OS designs. Also, RISC OS 3 support seems - at least in current versions - fairly minimal.

There is, however, an alternative library available, known as DeskLib. Although it lacks such features as the ability to write an Edit clone in a few lines of code, it is becoming increasingly popular for general Wimp programming. This article looks at the advantages of this new library, and tries to explain some of its central concepts.

Getting started with DeskLib

DeskLib for Acorn C is classified as "freeware", and can be obtained from a number of public domain sources including APDL disc B166. A recompiled version is also included with Beebug's Easy C compiler. However you get your copy, you should find, in addition to the library itself and its header files, the full source code for the library (although you may need some sort of de-archiving package to read this). Remember, however, that a significant proportion of the library is written in assembler, and cannot be changed unless you have the appropriate package. Nonetheless, some people will find the ability to customise the library useful, and some of the source files - especially those coding for the higher level parts of the library - can serve as useful examples for parts of your own programs.

All the DeskLib header files are contained within the !DeskLib application, and this application must be "seen" by the filer before DeskLib programs are compiled. Double-clicking on the !DeskLib icon causes a window to open on the extreme right hand side of the screen, showing all the available DeskLib headers.

When including a DeskLib header, it is usual to use a line of the form:

```
#include "DeskLib:Event.h"
```

If you have a large number of these to enter, you may wish to set up a DeskEdit macro (or equivalent in whichever editor you are using) to reduce the amount of typing needed.

The DeskLib library itself does not provide any error handling routines. You are expected to provide a suitable set, and compile and link them together with your main program. Luckily, for most purposes the default set provided with the library in the OtherSrc.c directory is adequate.

The DeskLib philosophy

Like RISC_OSLib, DeskLib relies heavily on the concept of attaching handler functions to particular Wimp events. However, the approach to attaching these functions varies significantly. In the case of RISC_OSLib, a variety of specific function attachment routines are available to deal with a number of different circumstances. Some of these routines are very specific to a particular type of event, such as a click on the icon bar icon. Others, such as the function responsible for collecting events concerning a particular window, pick up a wide range of vaguely related events, and usually require a large switch statement to sort everything out.

However, DeskLib provides only one main function-attachment routine, known as Event_Claim. In addition to the name of the handler function you wish to attach, this routine takes 3 parameters: an event type, a window handle, and an icon handle. If all of these are specified, only one very specific type of event will be picked up by the function. This might be useful for providing mouse-click handlers for specific buttons in a dialogue box. However, attachments can be made more general by not specifying one or more of these parameters, the only restriction being that if an icon handle is specified, a window handle must also be given.

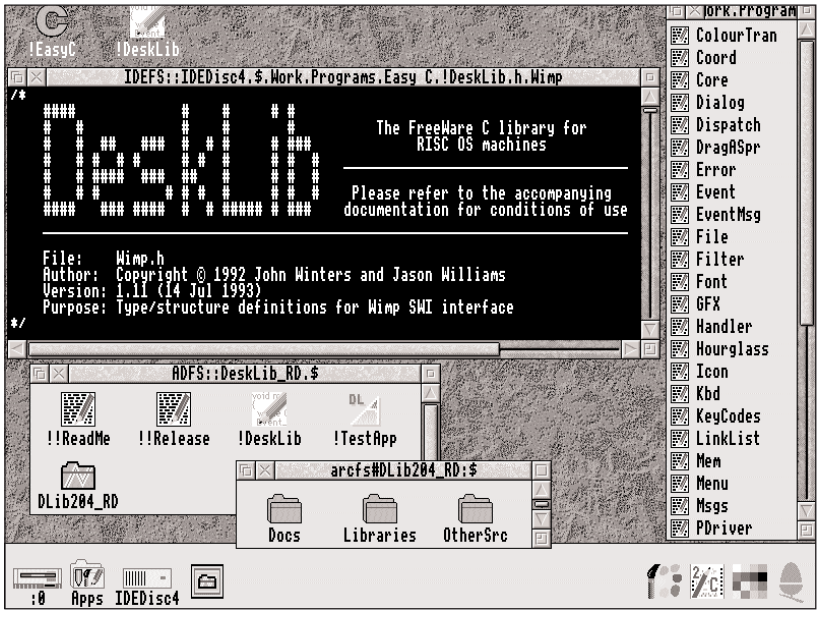
Although the Event_Claim system appears to be less advanced than the RISC_OSLib approach, it is simple, consistent and easy to use,

and is actually far more powerful. The Event module is so sophisticated that a Wimp poll mask is automatically generated based on which events have been claimed, so that the window manager does not bother telling a program about an event which does not interest it. This can significantly improve the performance of the desktop.

Much of the rest of the library is written around the Event_Claim system. For example, if you use the Window_Delete function to remove an old window definition, any event claims relating to that window are automatically released. This may seem a minor point, but can be very useful, particularly in complex programs where the exact set of handler functions attached to a particular window can vary according to the program s operating mode.

In theory, Event_Claim is the only function attachment routine needed to write even the most complex of programs. However, to tidy up message handling routines, the EventMsg module is supplied. This Event_Claim s the Wimp message received events. It then allows you to set a handler function for any specific Wimp message. This means that, for example, a message requesting a program to load a file and a message informing the program about a change of screen mode can be dealt with by different parts of the program, without having to be sorted out by a central dispatching function.

Many functions in both RISC_OSLib and DeskLib correspond closely to SWI calls provided by the operating system, and all these functions need to do is sort the function arguments out into appropriate registers, then make the SWI call. However, even here there is an important difference between the two libraries. DeskLib s programmers have always tried to make the function name the same as that of its corresponding SWI, even to the point of using the same capitalisation. This makes DeskLib familiar for anyone used to using Basic or assembler. RISC_OSLib, on the other hand, frequently chooses names which are very different from those of the SWI. For example, Hourglass_On becomes visdelay_begin! This is really only a minor point, but it is indicative of DeskLib s generally higher standard of Programmer-friendliness .



Using DeskLib

Advanced facilities

Although DeskLib lacks some of the very high level RISC_OSLib functions, such as those relating to the display of text files in windows, it makes up for them by a range of other advanced features, many of which are of more general use than those found in RISC_OSLib. Some of these facilities are simply interfaces into code provided as standard with RISC OS 3, such as the DragASprite module, and the Wimp event filter system, while others, such as the slider facilities are original code, provided by the library itself. The version which I use does not support the new features of RISC OS 3.5 (as used on the Risc PC), but I suspect this will be rectified before long (if it hasn t already).

Some parts of DeskLib have, at first sight, nothing to do with Wimp programming. These include the File module, which provides relatively low level access to RISC OS file management facilities. Such functions can be useful whether or not a program is running on the desktop, making DeskLib a useful library for C programmers writing any sort of RISC OS specific program.