DRAG'N DROP

RISC OS Pi • Beagle • Armini • Iyonix

January 2014 Volume 5 Issue 2 £3.00

67 pages!

Anatomy of a Drawfile

Write your own apps

Winglebith Sticky programming

Maestro to PMS

Mr Frog's Armcode Corner

GPIO for beginners

This issue has been blessed with contributions from the following people:
Gavin Wraith (Sticky Programming)
Matthew Thompson (BBC Games)
Jon Robinson (Unicode)
Christopher Dewhurst (everything else)

Contact Information

Editor: Christopher Dewhurst
Email: editor@dragdrop.co.uk
www.dragdrop.co.uk

EDITORIAL

Hello, I hope you had a great Christmas and you are starting 2014 off on a secure footing. Thanks for your support for *Drag 'N Drop.*

As we rely ever more on the internet for our sources of information, some people think the traditional "magazine" will become obsolete, be it paper or PDF like this one. Blogs and forums and the like are all useful but because they are free you get what you pay for! They also haven't been around long enough to prove their historical value.

Anyway, in this action-packed issue we round off one series (Philips Music Scribe), continue apace with another (GPIO for beginners) and begin two new series: one on writing your own applications and another on machine code by the irreverent Mr Frog. Not only that there is are articles on graphics and sound plus a full "type-in" game (Winglebith) to keep you entertained through the winter . . . so enjoy the read !

Chris.
Christopher Dewhurst

Beginner's Tips: Listing programs

Once you have typed in a BBC Basic program, whether it's a listing from *Drag 'N Drop* or your own concoction, you can type LIST followed by the Return key to see all the program lines. But if it is a longish program then the lines go flying off the top of the screen because the listing scrolls by so fast.

To read your Basic program listing a screenful at a time, first of all turn on **paged mode** by pressing **Ctrl+N**. You can now press the Shift key to display the next page. However that may still be too fast. So after typing LIST, hold down the **Ctrl** key. Keep holding down **Ctrl** and tap the **Shift** key to advance the screen a few lines at a time. Press **Escape** to stop and to turn off paged mode press **Ctrl+O**.

News and Applications

RISC OS Awards

Have you voted in the RISC OS Awards yet? You have until 15th January to cast your vote in any or all of the nine categories. Of course we advise voting for your favourite magazine in the "Best Publication or offline Resource" category. Vince Hudd is hosting the awards at www.riscosawards.co.uk

RISC OS SW Show 2014

The date for the South West show has been set for Saturday 1st March 2014, 10.30am to 4pm at the Webbington Hotel in Somerset. Come along for special offers and a chat on the *Drag N Drop* stand. More details at www.riscos-swshow.co.uk/

SystemDisc 1.01

A couple of utilities are available from Piccolo Systems.

Creation of OS cards (Secure Digital memory cards containing the RISC OS rom image and boot files) is very easy with SystemDisc. No longer do you have to rely on "foreign" machines (Windows or Linux) to make a bootable SD card for your Raspberry Pi.

CloneDisc is a similar utility for copying discs and disc images. CloneDisc and SystemDisc cost £12.00 each from piccolosystems.com.

Fireworkz 1.35

The freeware spreadsheet and word processing package has received an update. No less than 26 improvements bug fixes are list including better help files and tutorials, easier access to the Choices, plus when exporting pages to Draw, Fireworkz now includes Draw files (including sprites and JPEGs) present in the document.

Fireworkz 1.35 (01-Jan-2014) can be downloaded from downloads.abacusline.me.uk/freeware/fireworkz.

Fireworkz Google group is currently inaccessible on RISC OS with Netsurf. However we've been running a series on using Fireworkz in *Drag 'N Drop* from vol 4 issue 3 to vol 5 issue 1. Visit the www.dragdrop.co.uk to buy back issues.

Pluto 3.10

Pluto is a free email and news client suitable for the RISC OS Raspberry Pi. which has received a minor update to the "eSpeak" module.

sourceforge.net/projects/plutoemail

Don't forget you need an "email transport" to use it, whih mean PopStar from www.heenan.me.uk/acorn/download.html

DrawPrint 1.43

Hilary Phiillips' utility allows you to overcome the limitation of modern printer paper by printing Drawfies over several sheets of paper. Version 1.43 fixes the bug with disappearing JPEGs and scaling problems.
sinenomine.co.uk/software/

Passman 1.00

Kevin Wells' application allows easier management of passwords for restricted websites. Instead of entering them each time on a web page (and potentially storing confidential information in your browser's cookies) you keep the details in Passman (which itself has a password).

kevsoft.co.uk/news/2013/12/04/password-manager-released/

GPIO for Beginners

In the introductory article we used a breadboard kit to power a single LED from the Raspberry Pi's GPIO (General Purpose Input/Output) port.

The GPIO has a total of 26 pins and we wired up the LED so it is powered from one of the 3.3 volt pins (sometimes called 3V3 pins). These and two other 5 volt pins always have power supply going through them whenever your computer is turned on, which is why the LED was always lit.

This time we will learn how to control the GPIO so we can turn off and on the LED. All we do to start with is wire up the LED to one of the other IO (input-output) pins and use a simple SYS call in BBC Basic.

Before we go any further we should clarify one or two things. Firstly, you will see pictures on the internet of gadgets wired directly (sometimes even *soldered*) to the pins on the Pi's GPIO. This actually requires less components but fiddling around inside the bowels of your computer every time you need to connect or disconnect something results in more wear and tear in the long run.

Far better to use a breadboard as described last time, also known as a *Gertboard*, *Break-Out PCB* or *Cobbler* depending on who manufactures it. Whatever the name, they are boards which plug into the Pi via a ribbon cable and you wire up the components on the board and the ribbon cable stays permanently connected to the Pi.

Secondly, I am assuming you are using Model B of the Raspberry Pi, the one with two mounting holes in the circuit board. There has been a lot of confusion about the numbering of the pins, with several numbering schemes in existence. There are 17 IO pins and figure 1 shows their arrangement which I have verified by experimentation. It's oriented to the rest of the figures in this article (i.e. pin 7 is at the top left corner). There is no rhyme or reason about which pin is where, so keep figure 1 handy for future reference.

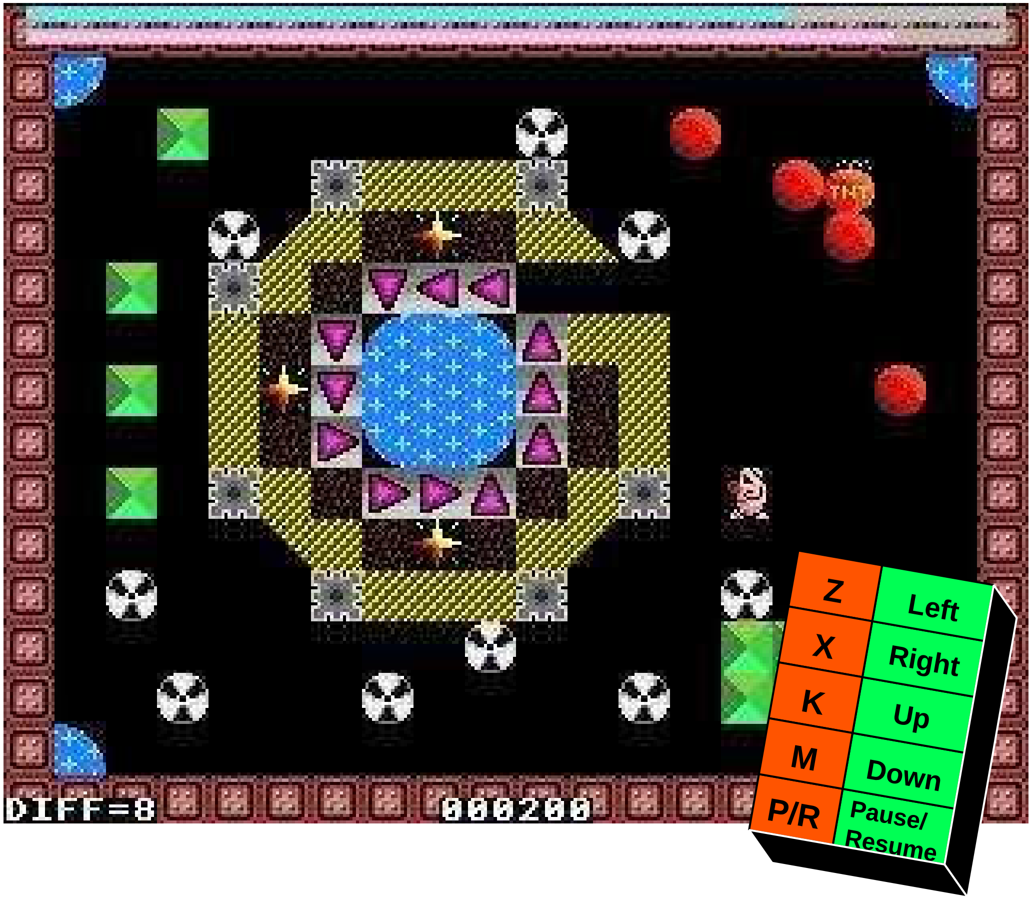
You'll be quite safe with the components we will be wiring up but I can't be held responsible for any damage you might inflict on your computer!

So, let's get down to business. If you still have the breadboard wired up from last time all you need to do is change the jumper lead going from the 3.3v pin to the LED so it goes from input/output pin 4 (IO4) to the LED. Figure 2 and the photo show this if you aren't sure. The breakout plug is up the top end of the breadboard instead of in the middle. It doesn't matter exactly where it goes as long as the short sides of the rectangular

7	GND
8	11
25	9
GND	10
24	**3.3V**
23	22
GND	27
18	17
15	GND
14	4
GND	3
5V	2
5V	**3.3V**

Fig.1 GPIO pinout

Winglebith



Move Winglebith around the screen disposing of radioactive cannisters by pushing them into holes and avoiding monsters.

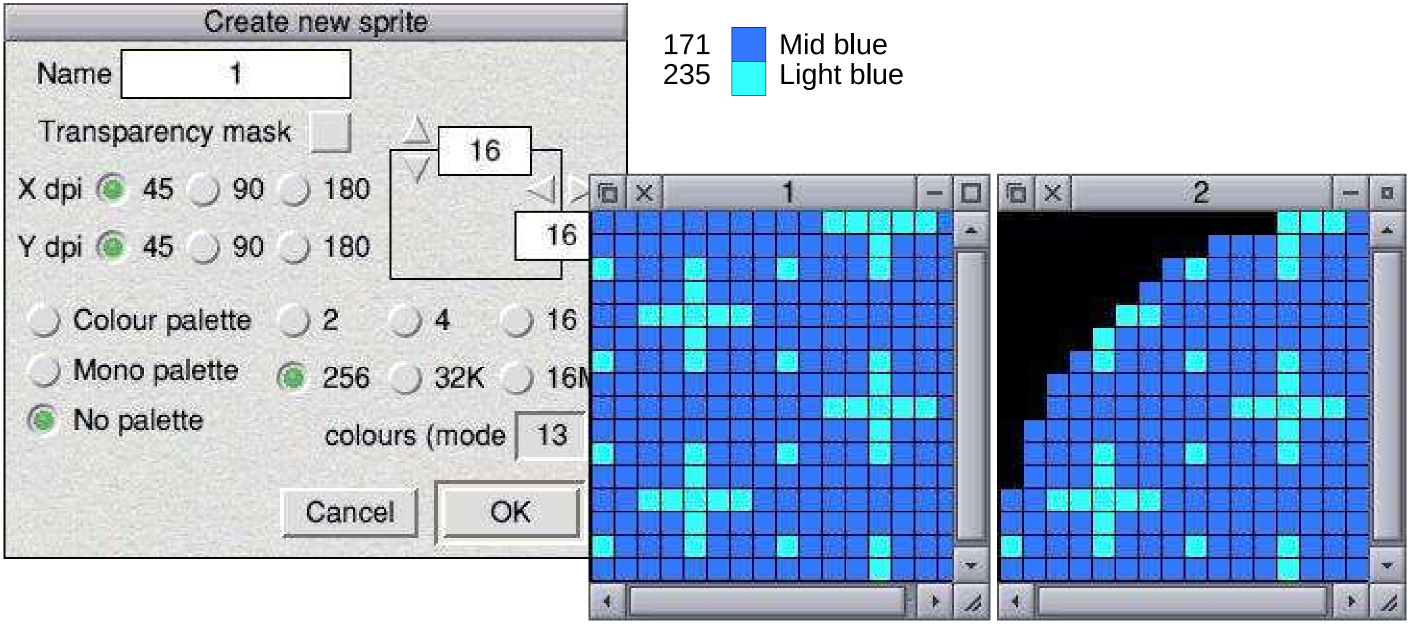
To complete each screen you must also collect the earth and crystals, defuse the bomb (surround it with boulders) and kill all the monsters (enclose with cannisters). And hurry there isn't much time!

Before typing in the listing, check your monitor setting can display Mode 13 and there is memory allocated to the system sprites. See PiBall in the Autumn 2013 issue of *Drag 'N Drop* if you aren't sure how to do this. Create the sprites as shown in figure 1 and save them as WBspr. Set the current directory to where WBspr is saved (Menu on filer and choose Set directory).

For more sound effects, run the UserVoices module which we present in the next issue.

Winglebith is a conversion of the BBC micro game called *System Wadgebury* by Andrew Cook.

Sprites and full listing begin on page 11


Create new sprite

Name: 1

Transparency mask [ ]

X dpi: [✓ 45] 90 180

Y dpi: [✓ 45] 90 180

Size: 16 × 16

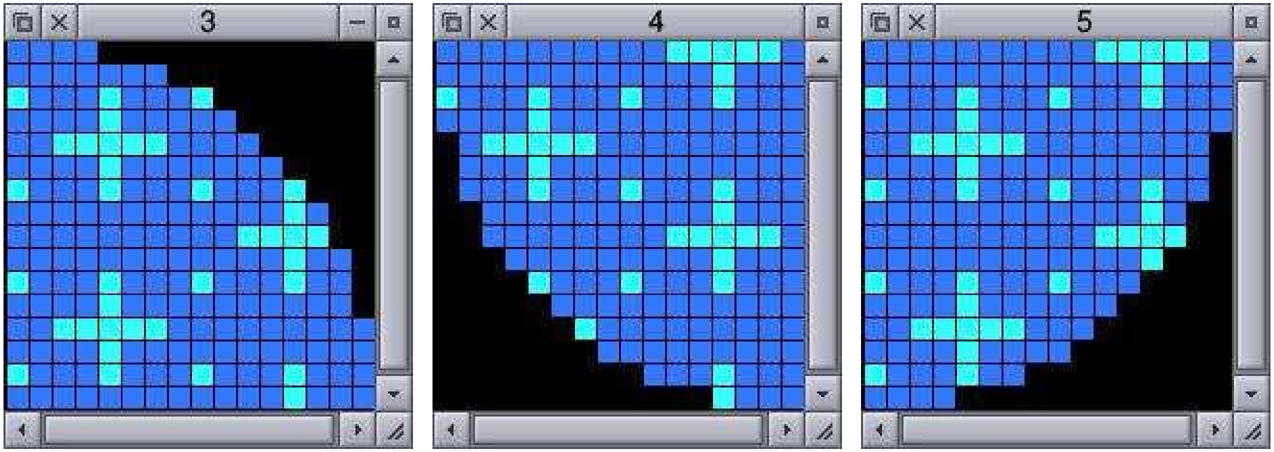
Colour palette / Mono palette / [✓ No palette]

2 / 4 / 16 / [✓ 256] / 32K / 16M

colours (mode 13

Cancel OK

171	🟦	Mid blue
235	🟦	Light blue



PROCmoving	Keeps Winglebith moving in set direction according to key pressed
PROCmovemon	Move monster
PROCmoveob	Move boulders or cannisters
PROCright	Move player right
PROCscore	Print score at centre bottom of display
PROCscreen(N%)	Initialises screen
PROCshowscreen	Plots the map
PROCsprite(a%, x%,y%)	Plot sprite a% and graphics coordinates (x%,y%)
PROCtitle	Print title page and hall of fame
PROCtune	Plays a note from the tune
PROCup	Move player up
PROCunder	Check contents of cell underneath player
PROCvariables	Initialise variables, read in tune data
PROCwait(delay%)	Wait until *delay%* centiseconds have elapsed
PROCwrite(a%, x%,y%)	Write contents a% to map cell (x%,y%)

```
 1250a%=FNread(wcx%,wcy%-1)
 1260death%=(a%=12)
 1270IF a%=14 PROCmoveob(14,0,
-1):ENDPROC
 1280IF a%=15 PROCmoveob(15,0,
-1):ENDPROC
 1290IF a%=0 OR a%=16 OR a%=17
 OR a%=19 dx%=0:dy%=1:moves%=9
 1300ENDPROC
 1310:
 1320DEFPROCface(x%)
 1330PROCsprite(wif%,wgx%,wgy%
)
 1340wif%=x%
 1350PROCsprite(wif%,wgx%,wgy%
)
 1360ENDPROC
 1370:
 1380DEF PROCmoveob(a%,dx%,dy%
)
 1390i%=1:REPEAT i%+=1
 1400b%=FNread(wcx%+dx%*i%,wcy
%+dy%*i%)
 1410UNTIL b%<>a%
 1420IF b%<>0 AND b%<>12 ENDPR
OC
 1430IF b%=0 THEN
 1440PROCsprite(a%,wgx%+dx%*i%
*64,wgy%-dy%*i%*64)
 1450PROCwrite(a%,wcx%+dx%*i%,
wcy%+dy%*i%)
 1460ENDIF
 1470PROCwrite(0,wcx%+dx%,wcy%
+dy%)
 1480PROCsprite(a%,wgx%+dx%*64
,wgy%-dy%*64)
 1490IF b%=12 ?count%=?count%-
1:PROCaddsc(200):SOUND4,-10,90
,10:PROCscore:ENDPROC
 1500IF a%=14 IF FNsurround(14
,bmx%,bmy%) PROCsprite(13,bmx%
*64,959-bmy%*64):PROCwrite(0,b
mx%,bmy%):bmx%=0:SOUND6,-10,10
,20:PROCaddsc(1000)
 1510ENDPROC
 1520:
 1530DEF PROCmonsters
```

Stick and Drag

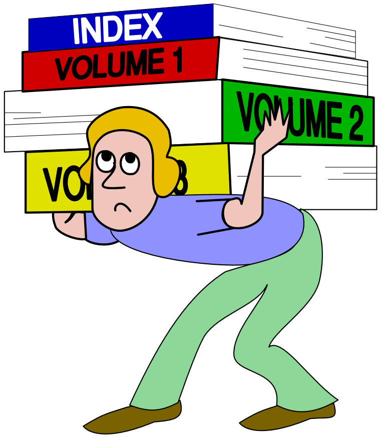
Programming can be a very tedious business. Those obstinate machines refuse to read your mind. How fortunate it is that you do not have to program everything yourself, but can use software that has been written by others – if you read the manual, that is.

Alas, the more that a piece of software can do, the more effort is generally required for reading how to use it. Who has not formulated for themselves a law of invariance of mental effort?

As for me, laziness makes me prefer the simpler tools.

A piece of software with a toolbar resembling the cockpit of a jumbo jet, and a set of manuals that need a wheelbarrow, fills me with dread. How are the shelves of surviving bookshops weighed down with *BehemothScript IV For Dummies*, and soon after our landfill sites are overflowing?

I recall the Marquis interviewing prospective coachmen, asking each how long he would take to drive from Ghent to Aix; they vie with each other in boasting of their speed, but it is the slowest and most careful who gets the job. So it is with software. Choose the simplest.



Sticky programming

To write a program that uses RISC OS's graphical user interface you usually find yourself having to tackle a lot of necessary but tedious detail. Luckily there are tools to ease the programmer's burden, such as templates, the toolbox, Acorn's FrontEnd module and Confix etc.

But simpler than any of these is **stick**, so long as your program does not create a window but just puts an icon on the iconbar and responds to clicks and drags on that. It is a little Basic program that I wrote in 1998 and have been using ever since. The idea is that **stick** sticks an icon on the iconbar and handles all the interactions with the task manager, leaving you to fill in the rest of the application with more simple-minded programs, of whatever kind.

I call such applications **stickies**. I have made dozens of them, mostly for my own use. This article is about how to make them.

You need two tools, apart from a text editor:

● Something to make or modify an icon, e.g. Paint

● a template editor for the info dialog-box for your sticky's iconbar menu. I use the excellent WinEd.

This was originally written by Tony Houghton and now maintained Adam Richardson. See www.riscos.info/index.php/WinEd for how to get it.

I tend just to modify pre-existing icons and dialog-boxes rather than create from scratch, as that is less work. You need **stick** itself if you do not already have it.

To get that go to www.wra1th.plus.com/stick.html and click on **TaskW**. That will download an archive file containing a little sticky that provides a simple example for our purposes. It runs in a taskwindow what you drag onto its iconbar icon.

Open **!TaskW** by shift-doubleclicking to find a copy of **stick** inside the

Writing a simple RISC OS app

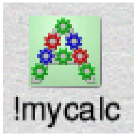
Has the thought of writing multi-tasking applications filled you with horror? Have you been putting off learning all that scary Wimp stuff?

Well it's not that hard and in this short series we're going to write a simple calculator application.

Each RISC OS application is kept in a special directory called the *application directory* starting with an exclamation mark (pling). To create one simply press Menu over a filer window and select New directory > !mycalc.

The green icon with cogs making up the letter A is the default icon for any application which does not have its own icon. See Figure 1.

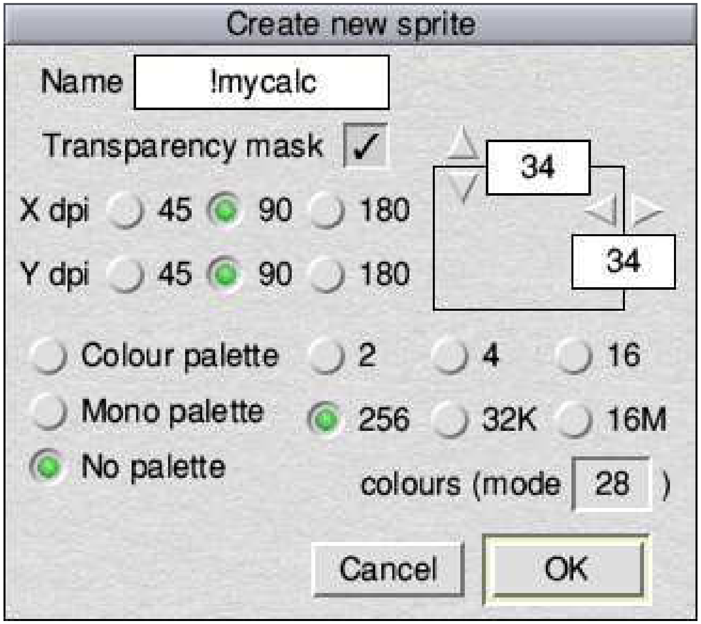
Figure 1



The application directory is like an ordinary directory except you have to hold down the shift key whilst double clicking to open it.

To give an application its own icon you have to design one in Paint. Load in Paint and select click on Paint's iconbar icon. Enter the parameters as per figure 2.

Figure 2



Create new sprite

Name !mycalc

Transparency mask ✓

34 × 34

X dpi 45 [✓ 90] 180

Y dpi 45 [✓ 90] 180

Colour palette / Mono palette / [✓ No palette]

2 / 4 / 16 / [✓ 256] / 32K / 16M colours (mode 28)

Cancel OK

The Name of the sprite has to be the same as the application directory you created.

Now design a suitable calculator icon. If you're stuck there are some examples in figure 3.

Icons are pictorial representations of what the application should do but they aren't photographs. Just make them look stylish and simple.

Save the icon inside the application directory as !sprites. The next stage is to Create an OBEY file as in figure 4. Type in the line

```
iconsprites <Obey$Dir>
.!Sprites
```

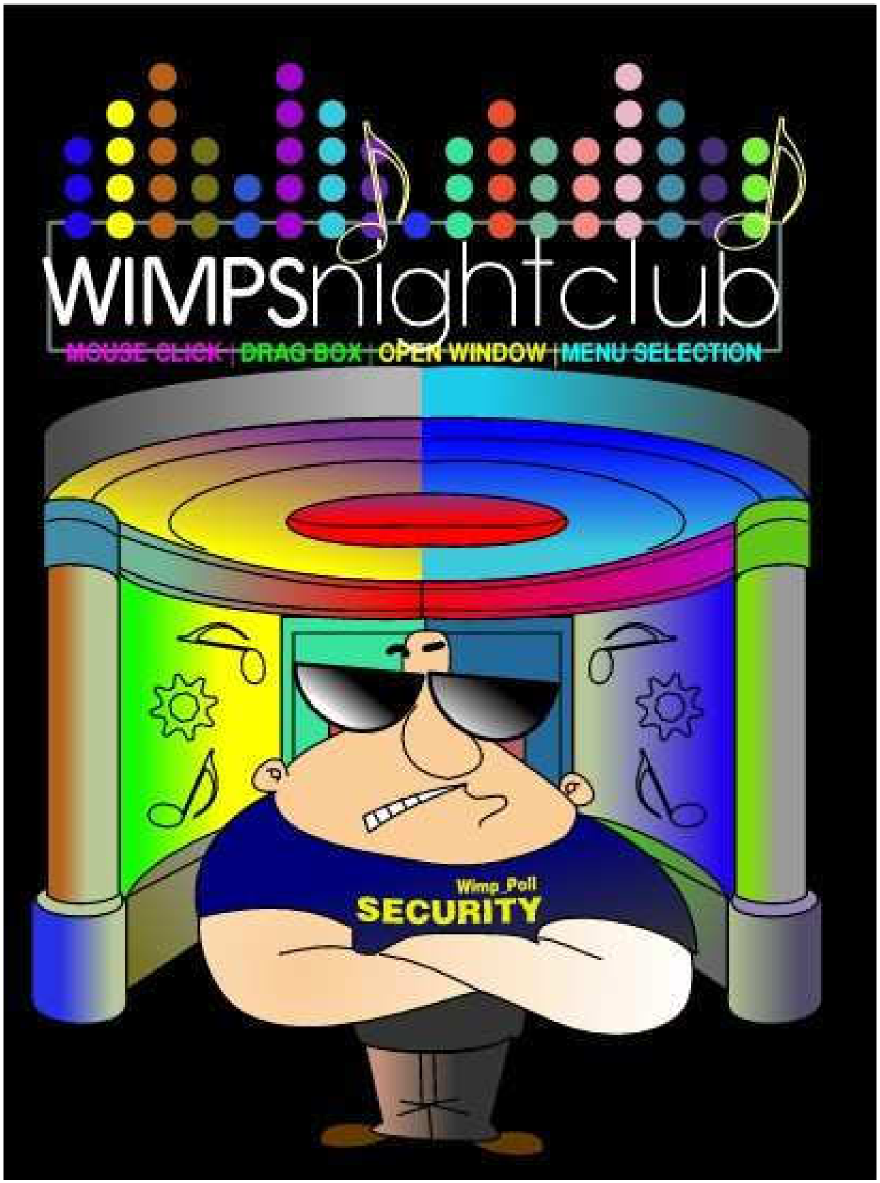
all on one line followed by the Return key and save it inside the application directory. What this does is to put our calculator sprite in the *sprite pool*.

When you call up a filer display RISC OS looks inside each application directory and runs their !Boot files, adding icons to the *sprite pool*.

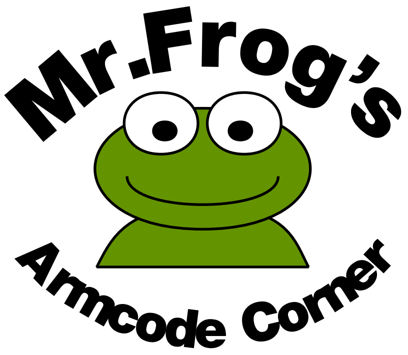
```
  220:
  330DEF PROCinit
  340DIM block% 255
  350
  360REM register task with ta
sk manager
  370SYS"Wimp_Initialise",500,
&4B534154,"Mycalc"
  380
  390REM install the icon on t
he icon bar
  400!block%=-1:REM icon on r/
h side
  410block%!4=0:REM clickable
area
  420block%!8=0:REM our icon i
s 34x34 pix
  430block%!12=68:REM which is
 68x68 OS
  440block%!16=68:REM graphics
 units
  450block%!20=&3002
  460$(block%+24)="!mycalc"
  470SYS"Wimp_CreateIcon",,blo
ck% TO ihandle%
  480
  650ENDPROC
```

The first thing PROCinit does is to define a block of memory. We communicate with the Wimp manager by this block of memory and a series of SYS "Wimp_SomethingOrOther" commands.

SYS"Wimp_Initialise" registers our app, or *task* with the Wimp manager. 500 is the lowest version of RISC OS that the *task* should run on multiplied by 100. As we are using RISC OS 5 it's 5× 100 = 500.



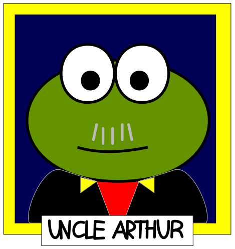
Some events captured by Wimp_Poll.



Many moons ago, my old cousin Mr Toad presented Machine Code Corner for the equally elderly BBC Micro. I thought I would continue in his footsteps. Except that the machine code your Raspberry Pi isn't quite the same as the machine code on the old BBC Micros. But machine code it is and it's called Armcode. If you learn how to write Armcode you can conquer the world!

Armcode was developed in Britain in the 1980s and is now used all over the world in smart phones, Nintendo DSs, iPads, iPods, Android tablets, as well the humble Raspberry Pi, because it is so fast and economic on power. (Just type "Arm code" into a search engine and you will come up with hundreds of references.)

Mr Frog remembers the very first RISC OS computers, years and years before smart phones were even dreamt of! So you could say we RISC OS users are the fastest and greenest computer users on the planet.



Enough gloating. Now, there are plenty of books on Armcode, some of which teach grandmothers how to suck eggs, and others are so full of unintelligable babble that would make Grandma Frog never touch an egg again.

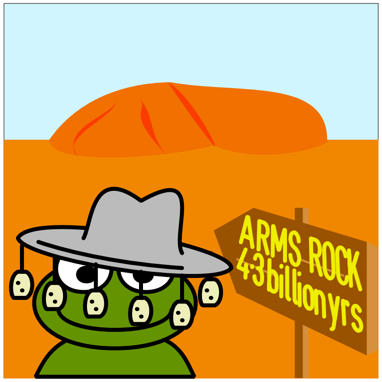
Mr Frog knows his bits. As well as bytes and hexadecimal arithmetic. I've done some Basic and 8-bit machine code programming in the past. But I'm not way up there with those super geeks who, when having dinner at an Indian restaurant look at the *Set Meal for One* menu and start discussing what an *Unset Meal for One* could be.

This is more or less the approach I take in Armcode Corner. It isn't a complete beginners series on computing so I expect you to be familiar with BBC Basic. Of course if you find your grandmother being taught how to suck eggs then you're best off keeping that unintelligable babble by your bedside for some night-time reading.

A single Arm code instruction is always held in one word of memory, which is four bytes or 32 bits. In the modern computing world, "word" means a collection of 32 bits but that hasn't always been the case.

In the 1960s, mainframe computers used six bits to the byte and 24 bits to the word. It could change again, of course, and we could be doing metric computing with 10 bits to the byte. But we won't worry about the future.

What 32 bits means is that there are 2^32 or 4,294,967,296 possible values. That's nearly 4.3 billion. Mr Frog's hobby is geology and he has noticed that 4.3 billion is about the same as age of the earth in years. Interesting, eh?



Anyway, not every single one of those 4.3 billion values is a defined instruction, and even then not all of them are *really useful* instructions.

What do Arm instructions actually look like? Did you know that your computer comes with a built in disassembler? It's a star command and it's called *MEMORYI (that's pronounced "star memory eye"). Press Ctrl+F12 on your computer. A task window opens up with a star ready for you to type a star command. Type in:

Anatomy of a Drawfile 2

In the introductory article in the Summer 2013 issue of *Drag ’N Drop* we discovered that Draw files consist of a ‘header’ followed by a number of ‘objects’.

Object number 2 is a path object and object number 11 is the options object. Although Draw stores the options when you save a drawing, object number 11 isn't compulsory.

We used this knowledge to write a Basic program to generate a Draw file of repetitive objects such as lines of a grid.

This time we'll meet two more objects, numbers zero and one, which concern text, the sort you type in using the Text tool (AZ symbol on the Draw toolbar). We will write Basic programs to produce displays shown in fig.1.

Object number zero is a "font table". It tells Draw what fonts are used in the Drawfile. Look at figure 2 which shows a dump of a Draw file consisting of a line of text printed in 12pt red text in the font Homerton.Medium.

In the computer's memory, the first word (group of four bytes) of an object is the object type. The second word is object's size in bytes. This is so the computer can add the size to the current location in memory to get to where the next object is stored. In figure 2, word zero is 0 (font table) and its size is &1C (28 bytes).

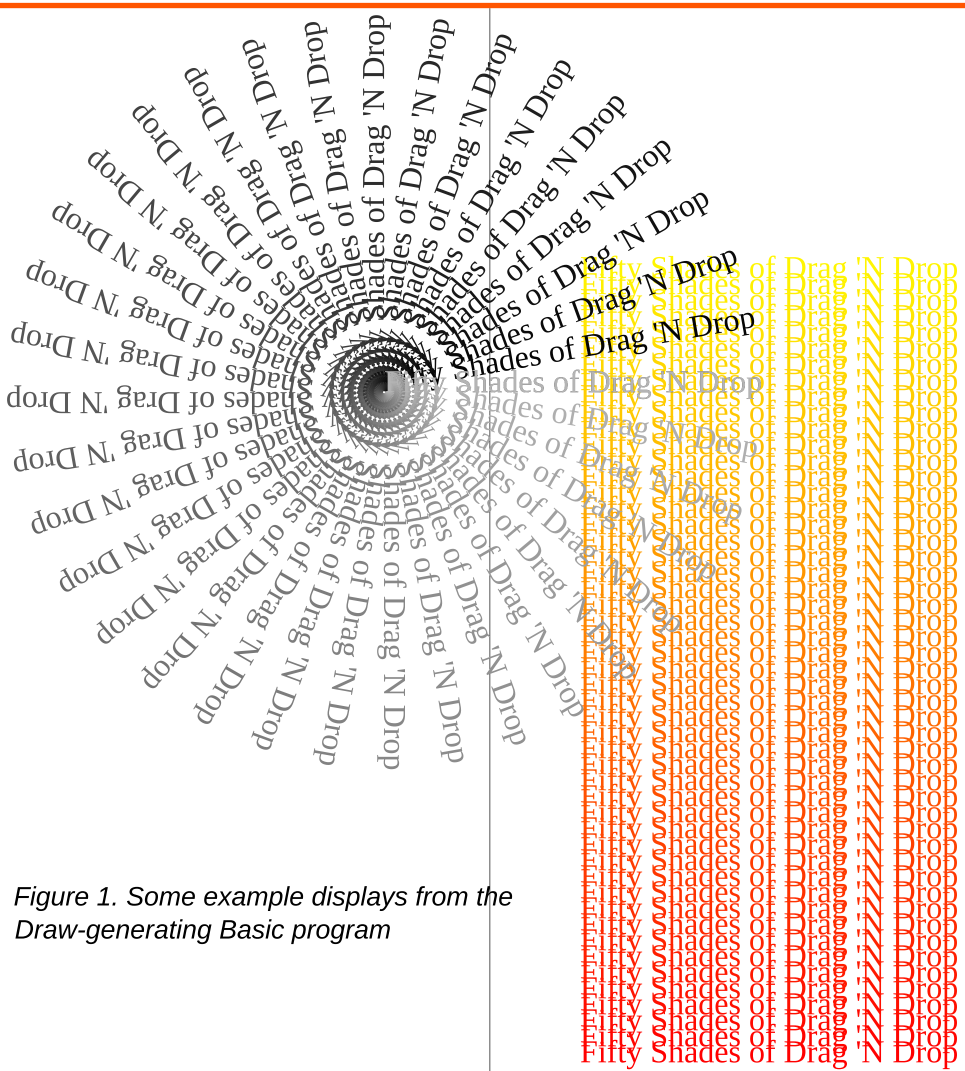


Figure 1. Some example displays from the Draw-generating Basic program

A Maestro to PMS converter

Maestro is a simple but surprisingly capable music writing application bundled free with RISC OS. One drawback of using it as a serious scoring tool is the print output. Maestro just produces a score with bitmapped notes resulting in amateurish looking sheet music.

Mae2PMS, the BBC Basic type-in utility presented here, converts a Music (type &AF1) file to a PMS text file for a more professional result. It handles multiple staves but not percussion or voices.

It isn't multitasking but in a future issue of *Drag 'N Drop* we'll add some fancy Christmas wrapping as part of the Wimp writing series. (It's also an ideal candidate to adapt for Gavin Wraith's *Stick* utility discussed elsewhere in this issue –Ed.)

You should have a folder called Documents.Music containing example Maestro files. Part of EdWelch (Blockbusters theme) is shown in Figure 1 on the next page, firstly as Maestro prints it and secondly as it appears after having been converted to PMS.

Since BBC Basic is so structured the program is listed without line numbers because none are needed.

To use it you need to set the Currently Selected Directory (CSD) to the directory that you have Mae2PMS in. Do this by selecting Menu > set directory on the filer. Copy any Maestro files you want into the directory as well. Now double-click Mae2PMS. Enter the name of the Maestro file (which will be used as the heading), press Return, then type the name of the PMS file to be created, and tap the Return key again.

A copy of the lines is output on the screen as well as being written to the output file. (This can be suppressed by inserting a REM in the penultimate line of the program.)

Rather than go through the program line-by-line, I've sprinkled it librally with REMs to explain what's going on. Most of the program is concerned with extracting bits out of the bytes and looking up the corresponding PMS notation in its tables.

The Programmers Reference Manuals (Appendix E) give details about the Maestro file format, which is what I followed when developing Mae2PMS. A copy of the PRMs are at http://www.riscos.com/support/developers/prm.

I was expecting Maestro to save its global variables at the *start* of the file but actually they are all at the end. What Mae2PMS has to do is to wade through the bulk of the file, reading in all the channel, gate and note data, store it in memory, then set up the number of staves etc. based on the global variables.

Gates are explained in the PRM as a "point in the music where something is interpreted e.g. note, time signature, bar line or clef". Pitches and durations of notes themselves aren't stored in the gates, only the fact that there is a space, or 'slot'. If it's a note, the slot value denotes what channel it's on, and in turn there is a 'mapping' between channel and stave depending on how many staves there are.

The pitches of notes aren't stored as pitches at all but as positions on the stave, together with durations, ties, and dots accidentals in channel arrays. The 'barb bit' seems to be unused in Maestro as all notes are unbeamed. PMS, on the other hand, will beam everything it can by default so you may need to insert semicolons to break beams.

It is important to realise that while the original Maestro file may play correctly in Maestro it may not be correctly scored due to sloppy transcribing. Several tweaks may be required. For example, Mae2PMS converts bar 14 of the EdWelch score thus:

```
e'= e'= d'= e' d'. | @ bar 14
```

The first three semiquavers should be quavers:

Colour Co-Ordinated

Modern RISC OS machines such as the Raspberry Pi and Iyonix can display a minimum 256 colours in *all* screen modes regardless of the screen Mode number you choose.

The system is still based on 16 palette registers dating from when there were 16 colour Modes on RISC OS computers and a one-to-one relationship between logical colour and register.

This cannot be continued in 256 colour modes – there simply aren't the registers. In 256 colour Modes, for each logical colour, the top four bits go straight to the red, green and blue guns, while the bottom four bits specify which palette register is used to make up the remaining bits.

The relationship for the top four bits is as follows (see also fig.1):

- bit 7 goes to blue bit 3
- bit 6 goes to green bit 3
- bit 5 goes to green bit 2
- bit 4 goes to red bit 1

There are two points to make here. First, it should be apparent by now that you cannot actually redefine the colours in 256 colour Modes since you cannot affect the top four bits. All that you can do is redefine the bottom four bits, or the shading of the colours.

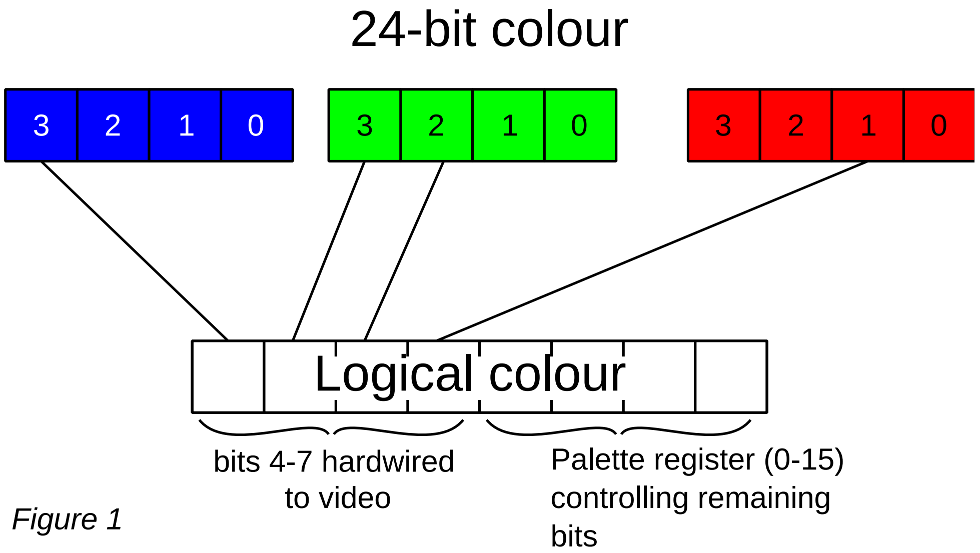

24-bit colour
3
2
1
0
3
2
1
0
3
2
1
0
Logical colour
bits 4-7 hardwired to video
Palette register (0-15) controlling remaining bits

Figure 1

Second, since two of the top four bits go to the green gun, you have more control over the red and blue guns.

So, if four of the twelve bits going to the guns are fixed, how can 4096 colours be achieved? Program 1 (Col256) demonstrates how the colours can be cycled through.

it works by taking the appropriate red, green and blue levels and deciding which logical colour should be used (on the basis that the most important bits of the displayed colour are determined by the logical colour used) and then reprogramming one of the palette registers to make up the remaining bits.

Probably the most important point to note here is that there is a lot of bit shifting involved.

Program 1

```
   10REM > Col256
   20REM Written by Philip J C
olmer
   30REM Originally published
in A&B Computing
   40REM Updated for Drag N Dr
op 2014
   50MODE 13:OFF
   60FOR red%=0 TO 15
   70FOR green%=0 TO 15
   80FOR blue%=0 TO 15
   90PRINTTAB(0,0)"Red = ",red
%;TAB(21);FNbin(red%)
```

WinED Tutorial

WinEd is a template editor for designing windows for use in Wimp programs.

WinEd can be downloaded from www.riscos.info/index.php/WinEd In this tutorial we will create a simple "About this program" window which is referred to in other articles in this issue of *Drag 'N Drop* on Wimp programming.

Double click !WinEd to install it on the icon bar. Click select on Iconbar icon. MENU on the new window and select Create. Type "info" into the box and click Create. See Figure 1.

Now double click on icon which has appeared. Three windows will appear, as in figure 2. They may be scattered around the screen; move them closer together if you want.

The icon picker contains a "palette" of icons which you can drag and drop onto the work window· Figure 3 shows the icon toolbar; once you have put icons from the icon picker you use the buttons to line them up or position them precisely.

Figure 4 describes how to build up the window using the icon picker and alignment tools. If you are familiar with Draw's alignment tools then WinEd's work in the same way. By selecting combinations of rows or columns of icons you can neaten the appearence of the icons in the window.

Figure 1

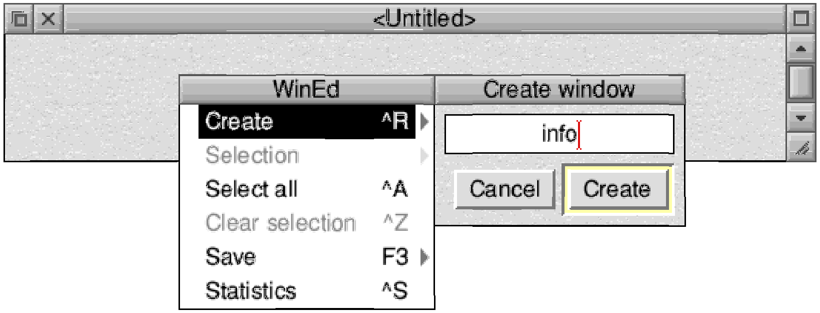
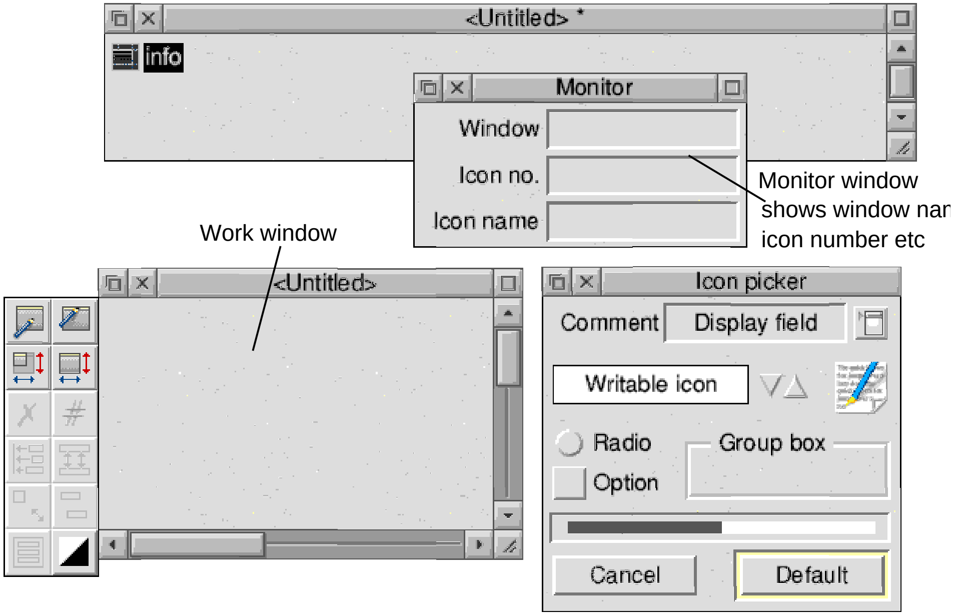


Figure 2 - The Main Windows Control Window!



Music writing with PMS

Part 5

This is the final part of our series on Philips Music Scribe in which we look at a couple of sundries: macros which are useful for typesetting repeated notes, and a brief look at the more advanced "drawing" facility.

If after that you want to pursue macros and drawings further then there are numerous examples in the PMS.Extensions folder.

It often happens in music that you have repeating figures which are tedious to score by writing out every single note. Fortunately PMS provides a couple of shorthands to make life easier. Look figure 1, the first four bars of a Carl Czerny exercise (yes him again). To write out the repeated left hand part you might use:

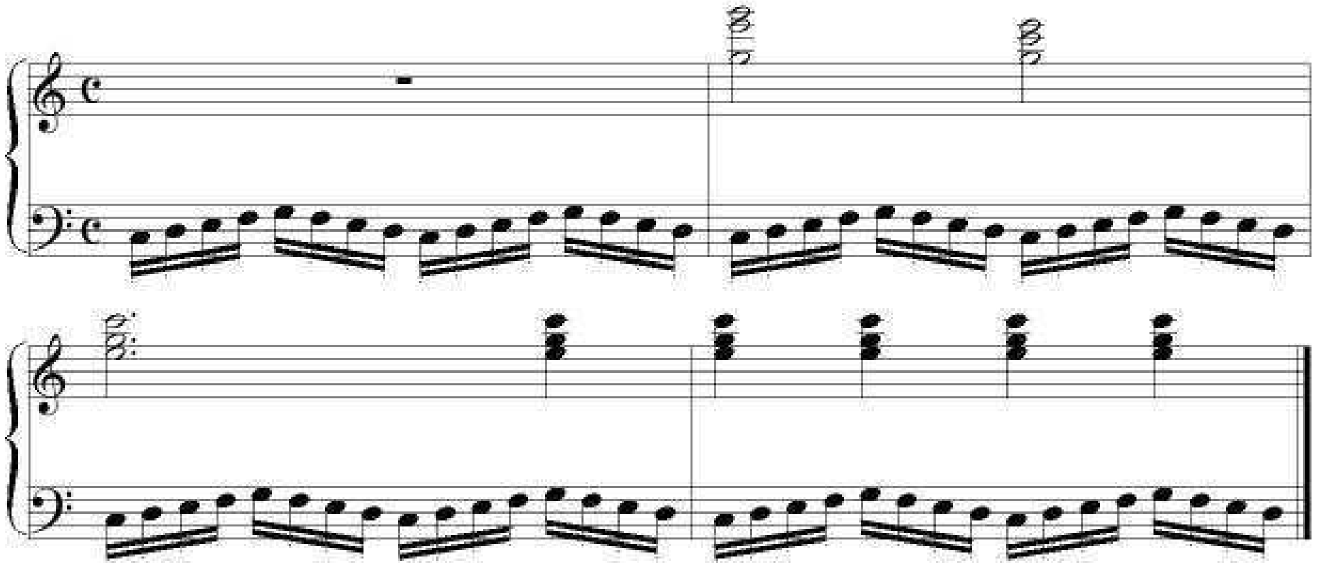
```
[stave 2 bass 0 time C]
c=d=e=f=; g=f=e=d; g=f=e=d; g=
f=e=d; |
c=d=e=f=; g=f=e=d; g=f=e=d; g=
f=e=d; |
c=d=e=f=; g=f=e=d; g=f=e=d; g=
f=e=d; |
c=d=e=f=; g=f=e=d; g=f=e=d; g=
f=e=d; |
[endstave]
```

While text editors make it very easy to copy and paste there are better ways. Firstly you can write in square brackets the number of times the bar is to be repeated, thus:

```
[stave 2 bass 0 time C]
[4] c=d=e=f=; g=f=e=d; g=f=e=
d; g=f=e=d; |
[endstave]
```

The [4] tells PMS to render the bar four times. The second method, suited to figures which repeat more often than a bar (half bars, say) is to use a **macro**. Macros go in the header section of the music and start with a star (like star commands):

Figure 1



Working with Unicode 1

Although we are still in the early stages of making the RISC OS system, and its software, fully Unicode compatible, with a little bit of fiddling there is quite a lot you can do with it already.

The was a general introduction in *Drag N Drop 5i1* where we got Draw to display Unicde symbols. In this article, we are going to see how to extract Unicode content from web pages and PDF documents, and incorporate it into our own text documents.

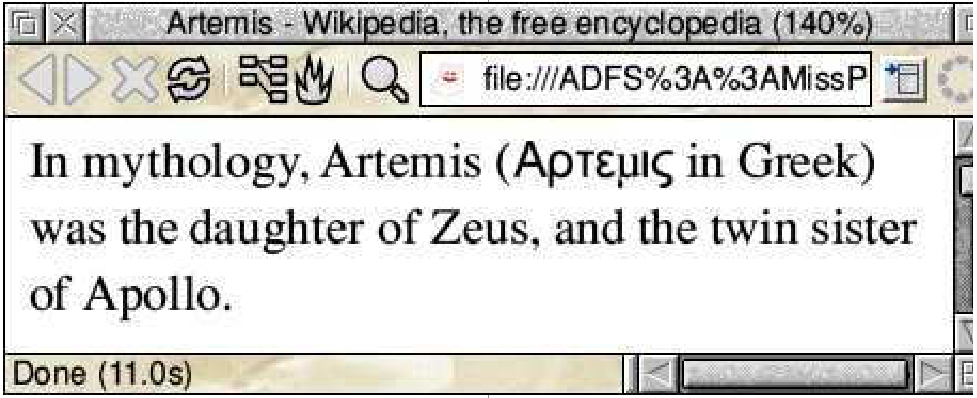
Of course, the first thing that needs to be said, is that you need either RISC OS 5 installed on your machine, or RISC OS 5's Font Manager modules, on an older system.

The most important piece of Unicode-compatible software we have on RISC OS is, undoubtedly, the open-source browser, NetSurf.

One of the advantages that PC users have over us, is that they can just cut and paste Unicode text from Explorer, straight into WordPad, if they want to make notes as they go along. This is something that I do all the time.

This doesn't work in NetSurf at the moment. If you try it, you just get a line of question marks where the text should be. (I'm looking forward to the day when it does work with NetSurf, however!)
But there is a kludgey way of getting Unicode content into your own documents, even if it's not quite as elegant as the way PC users do it - we simply load its HTML file into Edit.

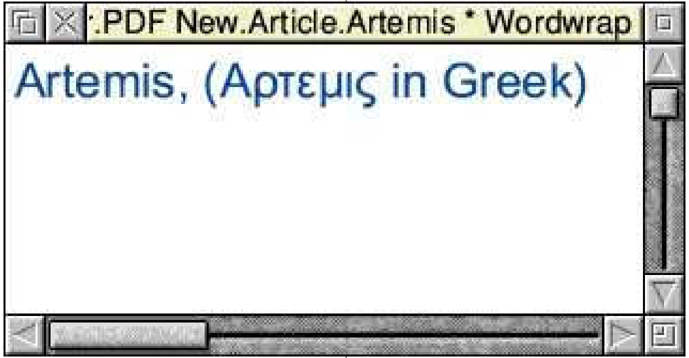
In this bit of a web page, displayed in NetSurf, the original Greek name for the god, Artemis, has been written in Unicode:



In mythology, Artemis (Αρτεμις in Greek) was the daughter of Zeus, and the twin sister of Apollo.

If we load its HTML file into Edit, and do a text search for something like 'twin sister,' we can find the bit of Unicode that represents Artemis, and save it into another Edit document.

Chris Gransden's PDFviewer also partly supports Unicode.



Artemis, (Αρτεμις in Greek)

Book Reviews

Title: Raspberry Pi RISC OS System Programming Revealed

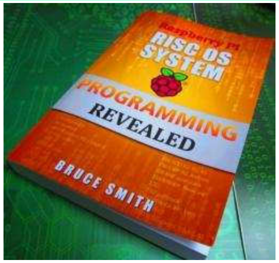
Pages:320

Author: Bruce Smith

Publisher: Bruce Smith Books

Price £19.99

Stockists: Amazon



Ever since the release of *Assembly Language Beginners Hands On Guide* (reviewed way back in the Winter 2013 edition of *Drag 'N Drop*) I had been keenly waiting the follow up book. It was billed as *Assembly Language Advanced* however the book that actually emerged is this one, which covers the whole of RISC OS in an in-depth way.

I was a little taken aback at the £20 price tag but with 320 pages it looked like it had a lot to offer.

I was not disappointed at the quantity of information (25 chapters) though there are a fair few typos.

The first few chapters look at star commands and SWIs, input and output of characters, a chapter on events and interrupts. I was glad the Arm's Status Register receives attention as it was noticably absent from the *Assembly Language HOG*.

Chapter 16 and 17 look at the Wimp manager although in a way that relies heavily on third party resources (e.g. AppBasic and Dr.Wimp).

After a well written chapter on the font manager, a chapter is devoted to writing your own modules. Chapter 19 and 20 concern sound although there is nothing about writing voice modules.

After a short chapter on using the floating point facilities of the Pi's Arm chip, there is a chapter on using the GPIO which I found useful but too short!

I wasn't as impressed with this book as the prevous one. It has the feeling of being rushed. A lot of the content supplements rather than replaces your old Acorn programming books.

Still, it's a very useful book to have around and we may yet see the book on advanced assembly language.

Title: The Big Book of the Raspberry Pi

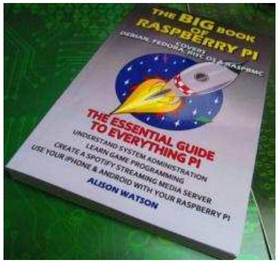
Author: Alison Watson

Publisher: Alison Watson

Price £6.99

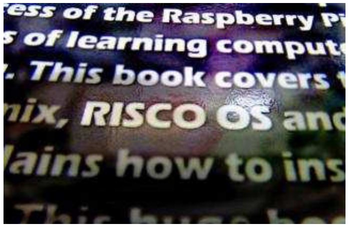
Stockists: Amazon

This book is written by somebody who is has a clear working knowledge of Linux but isn't really that bothered with RISC OS. Half of the book is a general introduction to the hardware and the other half is about writing a game in Python – the illustrations to which are quite frankly *awful*!


THE BIG BOOK OF RASPBERRY PI
COVERS DEBIAN, FEDORA, RISC OS & RASPBMC
THE ESSENTIAL GUIDE TO EVERYTHING PI
UNDERSTAND SYSTEM ADMINISTRATION
LEARN GAME PROGRAMMING
CREATE A SPOTIFY STREAMING MEDIA SERVER
USE YOUR IPHONE & ANDROID WITH YOUR RASPBERRY PI
ALISON WATSON

The one chapter devoted to RISC OS is cursory and incomplete. No mention of the built-in BBC Basic just a few keypresses away, and not enough explanation of the three-button RISC OS mouse.

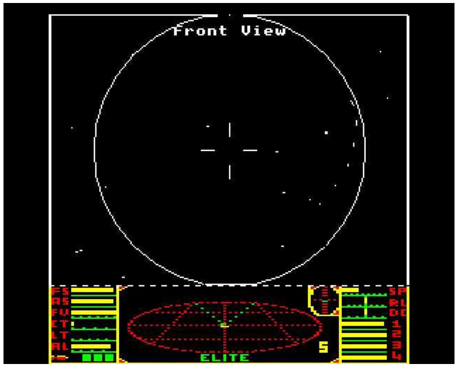
Sadly there aren't even any page numbers in this book. But we should be grateful of the author for bringing RISC OS to the attention of newcomers and the price of this book is reasonable.



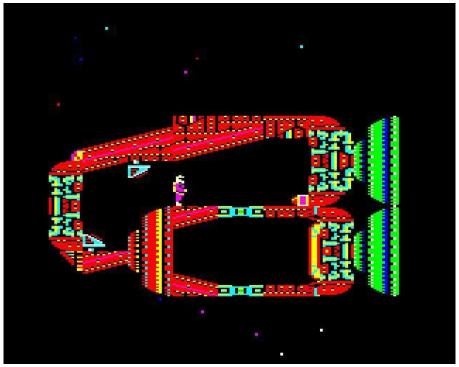
Part 3:Top 25 BBC Games

To finish off my BBC Micro retrospective articles, I have picked (in no particular order) 25 games which I think are ones which show how good the BBC Micro is for games, and dispel the myth that the BBC was no good for games and all you could play on it was Elite, Repton and Granny's Garden.

The games selected aren't necessarily favourites or ones I have played much, but I think they show what the BBC can do. There are plenty more games which deserve to be mentioned as well but it was hard enough trying to get the list down to just 25.

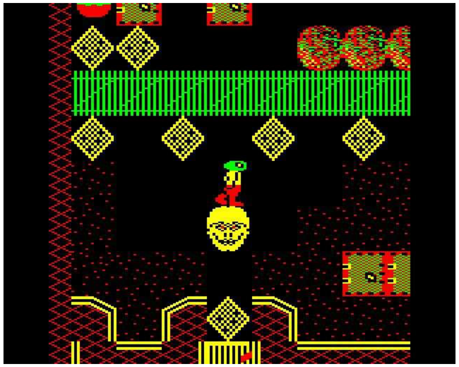


Elite - To be honest, Elite never really appealed to me, and still doesn't but it clearly was a groundbreaking title as when it was released nothing had ever been seen before with the endless galaxies to explore instead of single screen shooters. With the trading element as well this was a game that really stood out and made people look at the BBC in a different light. When you had played the game you could read the *Dark Wheel* novella.

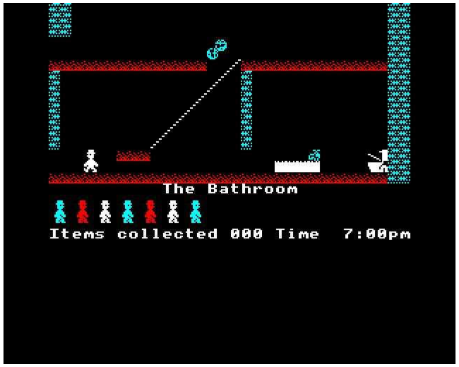


Exile - This was a game that really pushed the BBC to the limit, something which sadly never really happened with the Archimedes. *Exile* was a large scrolling arcade adventure; it also came with a novella to read before you played the game. The game featured enemies with intelligence, wind effects and gravity, which all affected how you could move through the game. A real classic and showpiece of what you could do with the BBC.

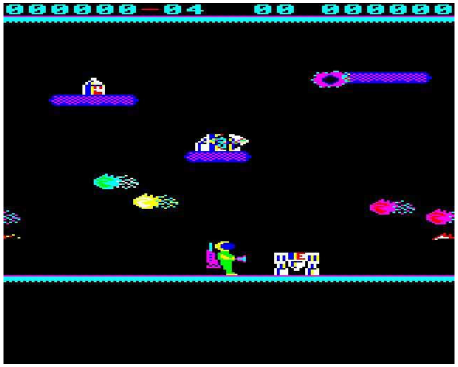
Repton 3 - well, there *had* to be one of the games listed really, as Repton is to the BBC as Mario is to Nintendo. Anyway I chose Repton 3 as I think it is the best release in the series. It builds on the previous two games with the new features.



in addition to the collecting diamonds and avoiding rocks there is a screen and character editor, which was used to create three more additional games. It works better as individual screens with passwords rather than the huge massive landscape that had to be completed in one go as seen in Repton 2.

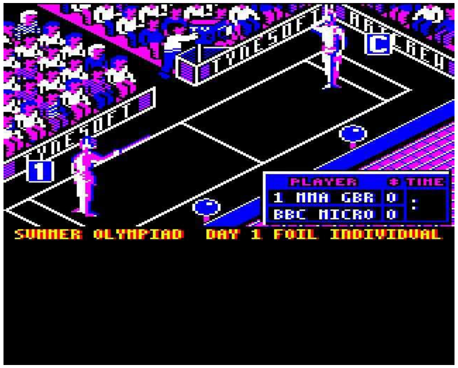


Jet Set Willy - one of the first games I ever played is quite simply legendary on whatever platform you played it. The Spectrum version is the original, but the BBC had a very good version: no rooms were missed out and apart from the title page rendition of *Moonlight Sonata*, Jet Set Willy on the BBC was every bit as good, and you could actually complete it too. No Attic bugs here !

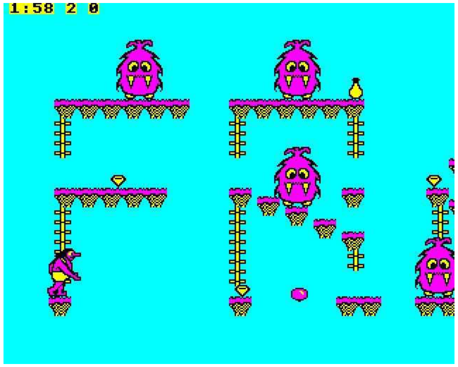


Jet Pac - A very simple and basic game, put the three bits of the rocket together, and then fill it up with fuel and get in it for take off, do this whilst avoiding hordes of aliens, when it's completed you can do it all over again ! That is all there is to it, but it is a very good addictive fun game, originally on the ZX Spectrum, the BBC version was every bit as good if not better as it was more colourful.

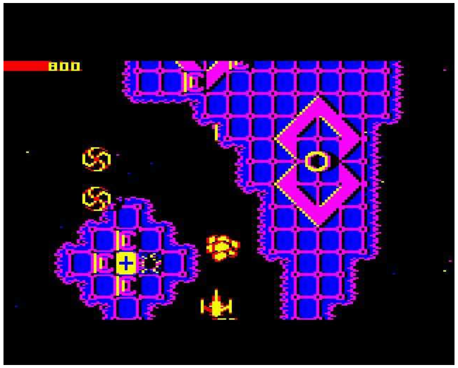
Summer Olympiad - Tynesoft were famous for their many multi player, multi event releases. This was one of the better ones and consisted of five different events which were Fencing, Triple Jump, Hurdles, Skeet Shooting and Diving.



You could play against a friend or the computer, a good selection of events in this one, and for those who enjoy this sort of game should check out the many other multi event games, *Rodeo Games* in particular is a good one.



Frak - An interesting little platformer, you control a caveman who has to move the monsters out the way so he can get to the end of the level. This is done by using your yo-yo to push them off the platforms. This game was a rarity in that the Electron version was actually better than the BBC version (apart from the colours) as it had more levels and a screen designer. When your caveman fell of a ledge a large speech bubble saying "Frak!" appeared above his head. I think we can all guess what he really means, and there are hacked versions which have been slightly altered, it's not too difficult to find if you really want to see it!



Firetrack - A vertical scrolling shoot-em-up. Whilst other platforms had many games in this style the BBC didn't have too many that differed from the regular space invaders style games. Whilst this is still shooting aliens, there are many different types in different formations who come at you as the screen scrolls up. You can shoot things on the ground as well and each level ends with you having to destroy the end of level boss. A brilliant game which is accompanied by some superb music.

The Sentinel - This was a totally original, unique game, were nothing else had been produced like it. The objective was to move up in height in the

Community Contacts

Developers & Publishers (RISC OS 5)

Archive www.archivemag.co.uk

CJE Micros www.cjemicros.co.uk

Datawave www.datawave.nl

David Pilling Software
www.davidpilling.net/riscos.html

Electronic Font Foundry thefonts.com

MW Software www.mw-software.com

Orpheus Internet
www.orpheusinternet.co.uk

Organizer www.organizerpim.co.uk

Piccolo Systems
piccolosystems.com/risc-os

PiLearn www.pilearn.com

R Comp www.rcomp.co.uk

RISC OS Code www.riscoscode.com

RISC OS Open Ltd
www.riscosopen.org

Webwonder/ProCad
www.zynet.co.uk/dsnell/Welcome.html

User Groups

HHRUG (Hemel Hempstead RISC OS user Group)
Area: Hemel Hempstead
Meets every 3rd Wednesday of the month
£3.50 per evening (£3.00 if paid 3 months in advance)
www.hhrug.org/

ICENI
Area: Ipswich
Meets every 1st Wednesday of the month
Visitors free for the first time
www.icenicomputerclub.org.uk

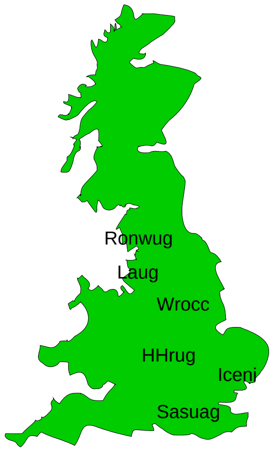
LAUG (Liverpool Acorn User Group)
Area: Liverpool
Meets every second Tuesday of the month
Free entry
www.orpheusweb.co.uk/bob.williams/laug/index.htm

ROUGOL (RISC OS User Group of London)
Area: London
Meets every third Monday of the month
Free entry
http://rougol.jellybaby.net

RONWUG (RISC OS North West User Group)
Area: North West
Meets 3rd Weds every month
Free admission
www.ronwug.org

SASAUG (Surrey and Sussex Acorn Users Group
Area: Surrey and Sussex
Meets every second Monday of the month
£1.50 per meeting for members
www.sasaug.org.uk

WROCC (Wakefield RISCOS Computer Club)
Area: Wakefield
Meets First Wednesday of the month
£2 for non members
www.wrocc.org.uk/



Are your RISC OS club details up to date? If not please let us know!