DRAG'nDROP

Volume 10 Issue 4
Winter 2021 £3.75

Raspberry Pi 400

Tutorials
Type-in apps
Reviews

Toolbox with Basic

Advertisement

40 years of improving on the best.

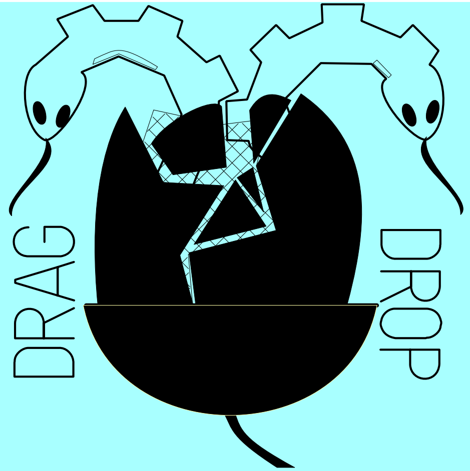
In 1981 the first BBC Microcomputer was released with 16K RAM, 8 colours, and a clock speed of 2MHz. Over the next 40 years a pedigree of fast machines running the world's best operating system, RISC OS, appeared. We won't bore you with the rest of the facts. Except to tell you about the latest computer. Which runs RISC OS*, of course. It has 253,952 as much RAM, 2 million more colours, runs 900 times faster, and is 10 times lighter than the BBC Microcomputer.

BBC Microcomputer Model A. 8 colours, 16K RAM, 2MHz, 3700g.

Raspberry Pi 400. 16M colours, 3968MB RAM, 1.8GHz, 386g.

The new Raspberry Pi 400. Still improving on the best.

Raspberry Pi 400 machine available from all good internet retailers. RISC OS downloadable separately. *Other operating systems available. "Raspberry Pi" is a trademark of the Raspberry Pi Foundation. E&OE.



This issue has been blessed with contributions from the following people:
Norman Lawrence (Schema2)
Christopher Dewhurst (everything else)

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

EDITORIAL

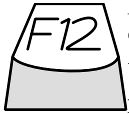
Welcome to another edition of *Drag 'N Drop*. Amongst the gloom of the pandemic, there's something to look forward to in 2021 and that's 40 years of the BBC Micro.

Incredible to think the little beige machine and its sucessors like the Archimedes and RISC OS introduced many people to computers and programming in a fun way, your editor being just one! Were it not for that I doubt I would have been remotely interested in computers as they'd just be drab, inaccessible things running horrible operating systems. So do please keep on supporting RISC OS developers, large and small. Enjoy the read.

Chris.

CONTENTS

How do I get the BBC Basic prompt?

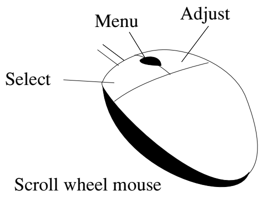


Press F12 and type *BASIC and press Return. You can change the screen mode with MODE n where n is a number e.g. MODE 7 or MODE 0.

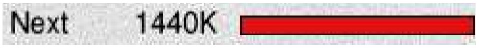
Type AUTO for automatic line numbering. Press Escape to stop and type SAVE "myprog" followed by Return to store *myprog* on hard disc. To return to the desktop type *QUIT.

Programs listed in *Drag 'N Drop* are assumed to work on all machines with RISC OS 5 e.g. Raspberry Pi, unless otherwise stated.

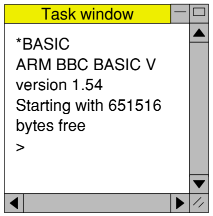
How do I open a Task window?



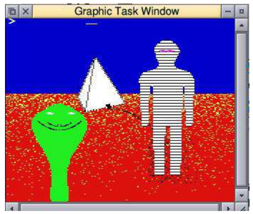
Menu click over the Raspberry icon on the right side of the iconbar and select click on Task window. Or press Ctrl+F12.



You may need to reserve more memory for the task. Adjust-click on the Raspberry icon and under *Application tasks* click and drag the *Next* slide bar out to the right.



```
*BASIC
ARM BBC BASIC V
version 1.54
Starting with 651516
bytes free
>
```



You can also type programs in a *task window*, press Ctrl and F12. You can't use the cursor editing facility or change MODE so you might like GraphTask from armclub.org.uk/free/. It allows you to type in and run Basic programs that use simple graphics (not sprites) in a window on the desktop.

To run Basic programs from the desktop, double-clicking with select on the filer icon runs it. Holding down Shift and double clicking loads it into a text editor like !Edit.

What does 'currently selected directory' mean?

Articles may tell you to set the CSD (currently selected directory). Click menu over filer window and choose *Set directory ^W*. It's where the computer stores the file when you type SAVE "myprog".

How do I open an Applcation Directory?

Application directories begin with a ! called 'pling'. Hold down the shift key and double click select to open the directory.

I get a blank screen when running games listings

Check you have the Anymode module installed, download it from www.pi-star.co.uk/anymode. It goes in !Boot.Choices.Boot. Predesk.

Open the !Boot application directory, in the root directory of the SD Card, that is SDFS::RISCOSPi. $.!Boot. Locate the *Loader* file and with Shift held down double click it to open it. Create a text file in Edit with the following line (press Return at the end):

```
disable_mode_changes
```

Save it inside Loader as CMDLINE/TXT and restart your machine.

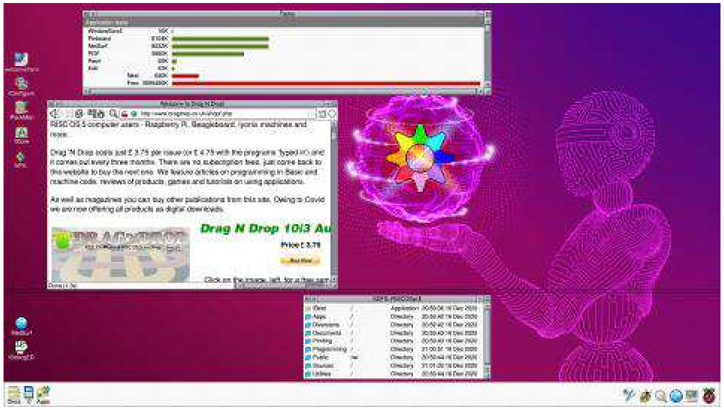
Sounds are strange

Some listings need the free RDSP module installed. Download it from www.amcog-games.co.uk/rdsp.htm where you'll find instructions on how to install it.

News and applications

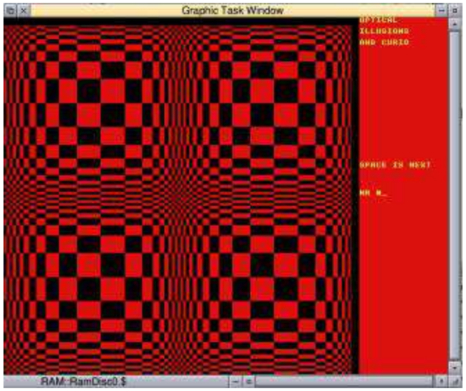
New Raspberry

The latest addition to the Pi family is the Pi 400, released by the Pi Foundation at the end of 2020. It's an all-in-one Arm A72-based compact computer with integrated keyboard which can run RISC OS. We take a first look at it on page 11.



GraphTask

GraphTask is a free application which allows many single-tasking programs (including ones using VDU graphics) to be run in a window on the desktop. It has been recompiled to fix issues with the Raspberry Pi 4. Version 4.04 can be freely downloaded from armclub.org.uk/free/. GraphTask comes with 35 programs written in BBC Basic demonstrating what can be done inside a GraphTask window.



Tutorial Central

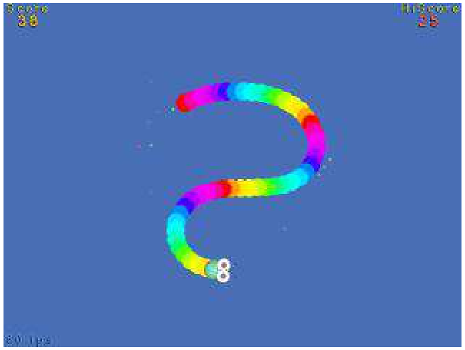
One of the criticisms of RISC OS is the lack of documentation (except in your favourite magazine, of course). A central repository of tutorials for new users and advanced users (DDE developers, C programming and the like) has been pulled together at www.riscository.com/tutorials/.

Tails from the Wimp

TailWimp Lite is a utility for arranging desktop windows keyboard shortcuts (Ctrl+Shift and arrow keys). The program is very much in development stages at the moment and you can try it out by visiting github.com/skymandr/TailWimpLite/releases/download/v0.1.0/TailWmpLt.zip

Maryland Escapee

Rainbow snakes live in swamps in the Atlanic region of the USA and one seems to have escaped to star in this 'snake' type game. You guide



your creature around the screen eating strawberries (it doesn't like raspberries, evidently) avoiding your ever-growing tail and the edges of the playing area .

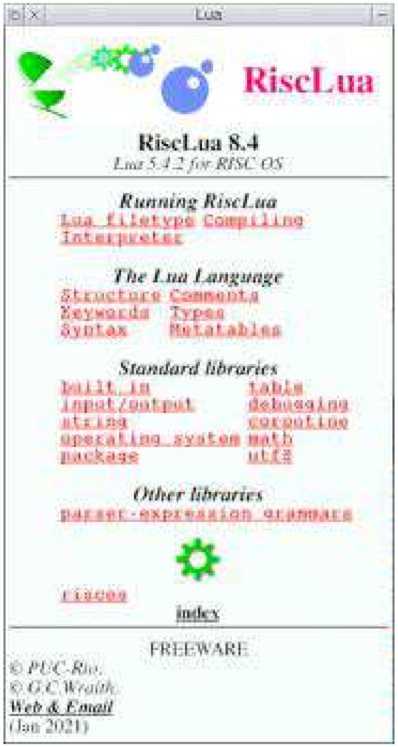
Simple but highly addictive with great graphics and sound effects, Rainbow Snake is free from www.proggies.uk/riscosstuff/riscos/zip/rainbowsnake_riscos.zip.

Lua Help

The RISC OS port of Lua, the high-level programming language available for a wide computer platforms now comes with a StrongHelp manual. Download it from www.wra1th.plus.com/lua/risclua.html.

Pimp your Draws

Draw is the free vector-based application which has come bundled with every RISC OS version and this freeware add-on has received an update. DrawPrint 1.53 from www.sinenomine.co.uk/software/riscosm/downloads.php allows you to print Draw files over several sheets of paper, it even puts cut marks in the margin. Useful if (like most of us) your printer is only big enough for A4 paper.


Lua

RiscLua

RiscLua 8.4
Lua 5.4.2 for RISC OS

Running RiscLua
Lua filetype Compiling
Interpreter

The Lua Language
Structure Comments
Keywords Types
Syntax Metatables

Standard libraries
built in table
input/output debugging
string coroutine
operating system math
package utf8

Other libraries
parser-expression grammars

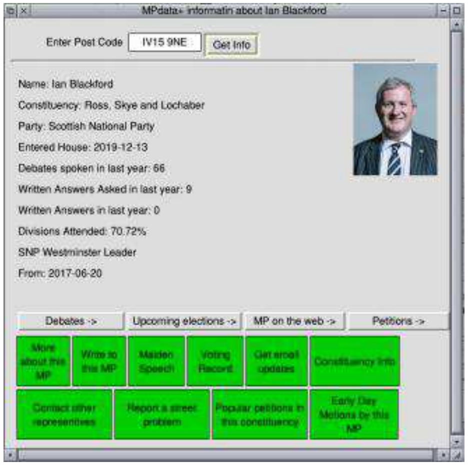
riscos
index

FREEWARE
© PUC-Rio.
© G.C.Wraith.
Web & Email
(Jan 2021)

They work for you

MPdata+ is an application to keep tabs on your Member of the UK Parliament. Type in your postcode and it retrieves information from Westminster on how your local MP is doing with attendance, debates and so forth. Version 2.01 is available from www.kevsoft.co.uk.

Small-time developers are the backbone of the RISC OS economy, and whilst Kevin's applications are free so you should consider making a small donation at www.ko-fi.com/kevsoft (ko-fi, the price of a coffee, geddit?)


MPdata+ informatin about Ian Blackford

Enter Post Code: IV15 9NE

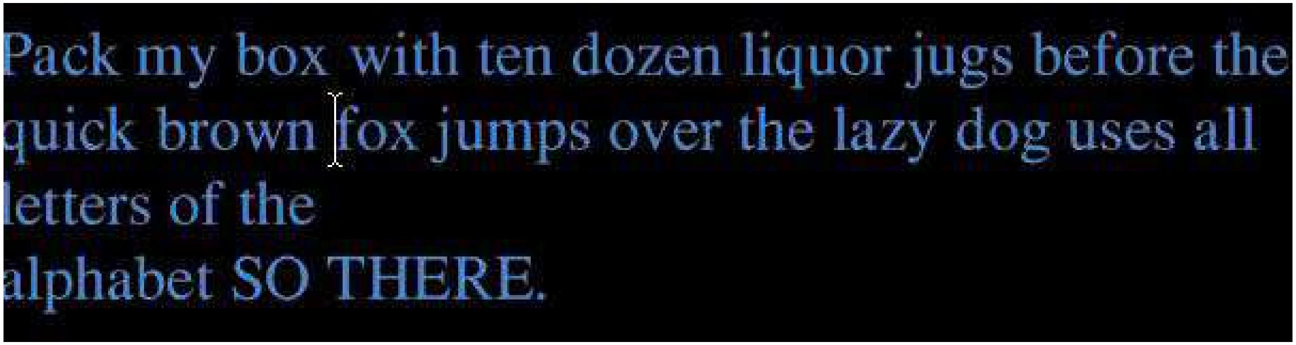
Name: Ian Blackford
Constituency: Ross, Skye and Lochaber
Party: Scottish National Party
Entered House: 2019-12-13
Debates spoken in last year: 66
Written Answers Asked in last year: 9
Written Answers in last year: 0
Divisions Attended: 70.72%
SNP Westminster Leader
From: 2017-06-20

Counting

CLOC is a command-line program which counts blank lines and comment lines in code in a variety of computer languages. Download it from www.coleman.orpheusweb.co.uk/downloads/cloc_v100.zip.

Text Editing

with outline fonts


Pack my box with ten dozen liquor jugs before the quick brown fox jumps over the lazy dog uses all letters of the
alphabet SO THERE.

This program demonstrates how to edit text in an outline font on RISC OS. In essence it's a very simple word processor.

Two pangrams – sentences that use all letters of the alphabet – are displayed. The user can insert new text by clicking the mouse to position the caret then typing at the keyboard (including Return) to insert text and erase it with the Delete or backspace keys. Press Tab to finish the program and display the contents of TEXT, the address in memory where the text is stored.

The program uses eight SYStem calls concerning outline fonts. They all take the form SYS "Font_SomethingOrOther". Some of these calls are very tricky to get your head round. Award yourself a large chocolate bar for each call you manage to get the hang of !

1. SYS "Font_FindFont",, Font$,X,Y TO FH

This 'opens' a font for future use. Font$ holds the name of the font required, for example Trinity. Medium or Corpus.Bold. X and Y are the horizontal and vertical type sizes, in printer's points multiplied by 16 (for instance to get a 10 by 12 point font, X=160 and Y=192). The computer returns the 'handle' of the font in FH. All references to the font are now made by its handle. Chocolate bar number one was an easy one wasn't it !

2. SYS "Font_SetPalette",,, 1,C,B,F

Sets up the colours in which the text is to be painted. C is the number of shades used for the anti-aliasing starting at B (background) and finishing at F (foreground). B and F are 24-bit values in &BBGGRR00 format (BB=Blue, GG=Green and RR=Red). So for yellow text on black, B=0 and F=&00FFFF00. The program chooses a random colour.

C can be anything from one to 255 with one giving the the jagged appearance of fonts on inferior systems like Microsoft Windows and 255 the maximum. In practice, 6 is sufficient for most purposes.

Raspberry Pi 400 – first look

Whilst the beginning of a new decade is marred by the global pandemic, it also heralds the appearance of a new generation of Raspberry Pi's.

The Raspberry Pi 400 was actually released just before Christmas 2020 and branded a "Personal Computer" coming inside an integrated keyboard, much like the Acorn Electron nearly 40 years before.

The machine is styled in white and dark pink and if you buy the full kit for around £93 you'll get a mouse in matching livery plus (more importantly) the cables. These are different from the Pi's 1 to 4. Video output is through mini-HDMI sockets and the power is via a USB type-C.

All ports are now located on the backside of the machine, a vast improvement on the previous Pi's where they were spread around all four sides. This includes the user (GPIO) and ethernet ports, plus two normal-sized USB sockets.

The machine can be purchased without accessories for around £67 so the Pi 400 is at least twice as much as previous models.

At the heart of the Pi 400 is the Arm Cortex A72. Arm chips have powered every native RISC OS machine since the Archimedes in the 1980s so RISC OS is the natural choice of operating system for the Pi, though it's yet to take advantage of features like quad-core processing. (Being so fast anyway, I suspect it may not make that much difference to RISC OS.)

The Pi 400 ships with its own proprietary operating system on a micro-SD card so you will need to reflash it with a 'special edition' of RISC OS freely downloadable from RISC OS Open, www.riscosopen.org.

Booting up, there's a good 10 seconds or so before the RISC OS splashbox appears announcing an incredible 3,968 meg of Ram! The RISC OS desktop appears soon after, with a dark pink theme; a wire-frame robot holds the RISC OS cog inside a spinning orb.

As with every new iteration of Pi, there are bound to be compatibility issues with software and we haven't had time to test out many applications. Software developers will iron out any problems in due course (programs written in BBC Basic V like *Drag 'N Drop* magazine listings aren't generally affected).

Some will deem the Pi 400's 'scrabble tile' keyboard too flimsy for serious work and others will see it as just a hobbyist machine with a retro look but the Pi 400 is a promising machine. It just depends on how quickly (or slowly) yet another Raspberry Pi appears !

Product: Acorn - A World in Pixels
Price: £29.99
Supplier: Idesine Publishing, www.idesine.com

With the country in lockdown, what better way to celebrate 40 years of the BBC Micro than with this sumptuous coffee-table book.

It's all about the arcade games that were released for Britain's ubiquitous 8-bit computer of the 1980s. The lavish hardback volume with slipcase contains over 450 full-colour glossy pages packed with screenshots and maps of games plus interviews with nearly 50 games authors and BBC Micro journalists. They chat about their lives when they discovered the BBC Micro, many of them just schoolkids and programming those new micro computer things just for fun.

A World in Pixels paints a fascinating geography of Britain of the time. Most of the activity took place in London, Leeds, the North East and of course Cambridge where the BBC Micro was born. The proto-gaming industry was vastly different from today's scene and many of the 'grown-up' coders are still here in 2021 in one computing field or another.

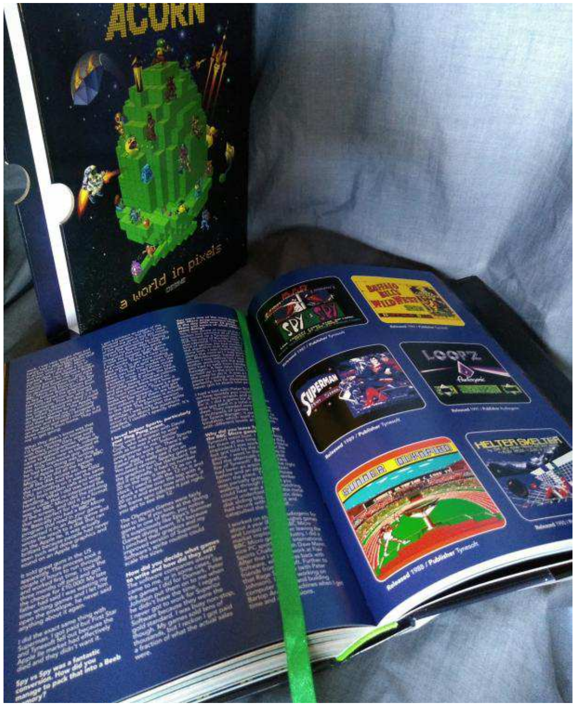
If like me, you're interested in the technical aspects of the Beeb as well as its social history you'll like the programming pearls dropped in by the interviewees, such as assembly language tricks and compressing graphics data into the Beeb's tiny memory. I often wondered if Citadel used some kind of 'cut out' routine to draw its amazing locations and I didn't realise JCB digger had to use two scrolling mechanisms and nicknamed JCB Judder because of it!

The foreword is by Richard Hanson who's still the MD of Superior Software. The stories told in *A World in Pixels* reveal Richard a supportive publisher to the schoolboy coders. Alas, Superior don't publish any RISC OS games nowadays, odd when you think how well Repton and the like were ported to 32-bit machines.

Several well-known personalities make an appearance including Acorn's design manager and Elite authors; the absence of other Beeb exponents is notable. (Probably were far too busy to bother with a book on games.)

Some shortcomings in the proof-reading of this book made me cringe and it makes no mention of the Raspberry Pi although the RISC OS scene isn't entirely forgotten about. One or two early Archimedes programmers talk about graduating to the 32-bit scene from their BBC days.

● This is a magnificently presented book and a must for BBC Micro gamesters, past and present.



FAST ICONS

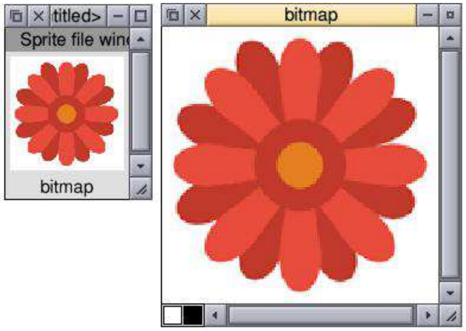
If you write applications for RISC OS then you may well suffer a mental block when it comes to designing icons.

In this article I'll show you how to create them in next to no time. All you need is Netsurf and Paint.

First navigate to images.google.co.uk in Netsurf. Decide the theme for your icon and type it in the search box followed by "ico". For example I am writing an application for cataloguing the seeds from my garden so I searched for "flower ico". (N.B. it's "ico" and not "icon".)

Important point here. Check the images to ensure they're not restricted by copyright. Many pictures are free for personal use but some are not.

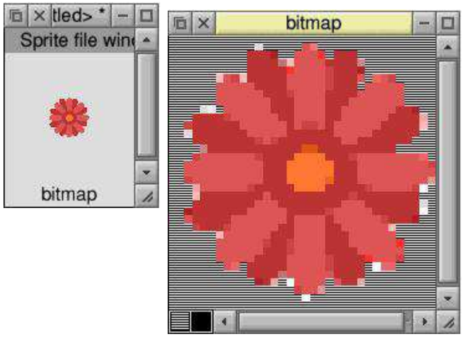
Press the Menu button over a suitable icon, preferably one with a white background, and choose Object > Object > Export > Sprite and drag and drop the file onto Paint's icon bar icon. You'll get a window called <*untitled*> and a sprite window called *bitmap*. Click anywhere in the background (white) of *bitmap* so the window title turns yellow.



The *RISC OS Style Guide* recommends that application sprites should be no more than 70 OS units square, about 35 × 35 pixels in practice.

Icons exported from Netsurf are 140 pixels square which, unless Netsurf or Google has changed by the time you read this, is too big.

Press Menu over *bitmap* and choose Edit > Scale X > 0.25, click with Adjust and then choose Scale Y and click Select. (The 0.25 is already there. Remember in RISC OS there's no need to close the menu and navigate through the whole structure again.)



If the sprite doesn't have a mask add one by Menuing over *bitmap* > Edit > Mask. Bring up the colours window with Shift+Ctrl+F1 and choose None for the colour (which selects the mask). Press Ctrl+F1 for the tools window. Click the paint can icon and ensure the Global is selected. Now click Select a couple of times in the white area surrounding the flower.

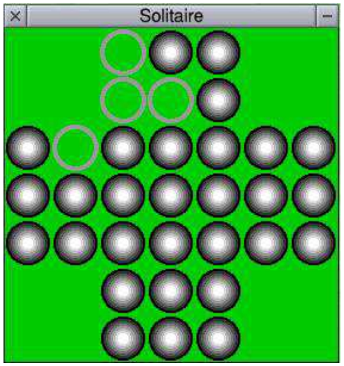
Rename *bitmap* to *!flowers*, create a directory called *!flowers* on your hard disc, and save the sprites as *!flowers.sprites*, add a one-line obey file

```
iconsprites <obey$dir>.sprites
```

!flowers

and save it as *!flowers.!boot* then double click it.

Desktop Solitaire



This is a desktop version of the classic patience game where you must clear the board of pebbles.

The game starts with a '+' layout of pebbles. Move a pebble, by clicking and dragging the mouse, to a hole two positions away horizontally or vertically. The computer will remove the pebble in the middle.

Click menu over the Solitaire window, or its iconbar icon, for program information, to start a new game, or quit the application.

SOLITAIRE listing

```
   10REM Solitaire
   20REM (C) Drag N Drop 2021
   30ON ERROR PROCERROR
   40SYS "Wimp_Initialise",200,&4B53
4154,"Solitaire"
   50DIM B 500, N 1000, S 2000, T 10
00, U 1000,B$(6,6),V 10
   60Q=FALSE
   70PROCSPRITES : PROCWINDOWS : PRO
CICONS : PROCMENUS :U0=U
   80PROCNEW : PROCOPEN(W1)
   90REPEAT
  100SYS "Wimp_Poll",,B TO E
  110CASE E OF
  120WHEN 2: SYS "Wimp_OpenWindow",,
B
  130WHEN 3: SYS "Wimp_CloseWindow",
,B
  140WHEN 6: PROCCLICK
  150WHEN 7: PROCDRAG
  160WHEN 9: PROCMENUSELECT
  170WHEN 17,18:Q=(B!16=0)
  180ENDCASE
  190UNTIL Q
  200SYS "Wimp_CloseDown"
  210END
  220:
  230DEF PROCOPEN(W)
  240!B=W
  250SYS "Wimp_GetWindowState",,B
  260SYS "Wimp_OpenWindow",,B
  270ENDPROC
  280:
  290DEF PROCNEW
  300RESTORE +1
  310DATA --PPP--,--PPP--,PPPPPPP,PP
PHPPP,PPPPPPP,--PPP--,--PPP--
  320FOR Y=0 TO 6 : READ A$
  330FOR X=0 TO 6
  340B$(X,Y)=MID$(A$,X+1,1)
  350NEXT:NEXT
  360PROCBOARD
  370ENDPROC
  380:
  390DEF PROCBOARD
  400FOR I=0 TO 49
  410!B=W1 : B!4=I
  420SYS "XWimp_DeleteIcon",,B
  430NEXT
  440SYS "Wimp_ForceRedraw",W1,0,0,1
279,1023
  450U=U0
  460FOR Y=0 TO 6 : FOR X=0 TO 6
  470IF B$(X,Y)<>"-" PROCMKICON(W1,X
*80,(6-Y)*80,80,80,&B7006102,B$(X,Y)
,S)
  480NEXT:NEXT
  490ENDPROC
  500:
  510DEF PROCCLICK
  520PROCGET
  530CASE W OF
  540WHEN -2 : CASE MB OF
  550WHEN 1,4 : PROCOPEN(W1)
  560WHEN 2 : SYS "Wimp_CreateMenu",
,N1,MX-80,MY+180
  570ENDCASE:ENDPROC
  580WHEN W1:IF MB=2 SYS "Wimp_Creat
eMenu",,N1,MX,MY:ENDPROC
  590IF (MB AND 4)=0 ENDPROC
  600CX = (MX - B!4) DIV 80
  610CY = 6- (MY - B!8) DIV 80
  620IF B$(CX,CY)<>"P" ENDPROC
  630!B=CX*80 + B!4
  640B!4 = (6-CY)*80 + B!8
  650B!8 = !B + 80
  660B!12 = B!4 + 80
  670SYS "DragASprite_Start",16,S,"P
",B
  680ENDCASE
  690ENDPROC
  700:
  710DEF PROCDRAG
  720PROCGET
  730TX = (MX - B!4) DIV 80
  740TY = 6 - (MY - B!8) DIV 80
```

Programming in Postscript

In the introductory article in the Autumn edition of *Drag 'N Drop* we remarked that PostScript is an arse-about-face language because commands look back-to-front.

In Basic we would write

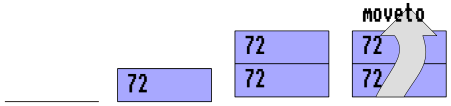
```
MOVE 72,72
```

but in PostScript it's

```
72 72 moveto
```

(72,72) is a position one inch from the bottom left corner of the page. Postscript's default measurements are in printer's points (72 per inch) so an A4 sheet measures approx 595 × 842 points.

The commands are back to front, or post-fixed as we say in he trade, because PostScript uses a stack. Imagine a stack as a pile of bricks. It starts off empty and items are added to the top of the pile. As soon as you issue an instruction like **moveto**, Postscript goes looking for items underneath it on the stack:



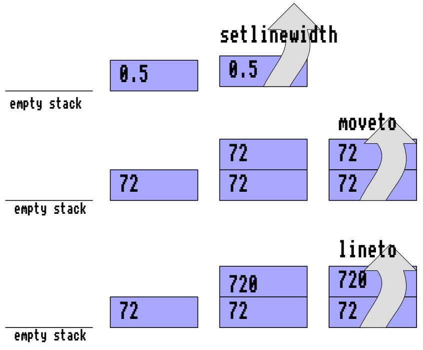
It pulls off (or *pops*) the **moveto** and the two bricks under it. It's known as a first in, last out (or first on, last off) stack. If there aren't enough bricks there, Postscript throws a wobbly and produces an error message.

Here is listing 1 again from last time:

```
%!PS-Adobe
newpath
.5 setlinewidth
72 72 moveto
72 720 lineto
stroke
showpage
```

newpath, **stroke** and **showpage** take no parameters, which is to say they don't need anything from the stack. **setlinewidth** on the other hand expects a number between zero and 1 on the stack. **moveto** and **lineto** each expect two numbers on the stack.

The program can be summarised in bricks as follows:

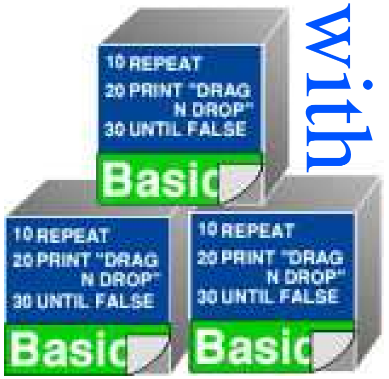


Now, it's perfectly feasible to re-write the program to put all the numbers on the stack first:

```
%!PS-Adobe
newpath
72 720 72 72 .5
setlinewidth
moveto lineto
stroke
showpage
```

This can be thought of in bricks as in the following diagram:

Toolbox with Basic



The User Interface Toolbox, Toolbox User Interface, or just Toolbox as it's known to its friends, comes free with RISC OS and it takes some of the drudgery out of programming desktop applications.

For example, it’s common for applications to display a small window giving program information and this often leads off a menu item accessed through an icon bar icon.

Traditionally, one of the many setup tasks the application's main program, usually called *!RunImage*, would have to do is to tell RISC OS to place an icon on the icon bar and it would have to be taught where the ‘About this program’ window lives (whether designed with a template editor or coded in-program like in *Drag ’N Drop* listings). *!RunImage* would also have to hold procedures (maybe in a separate Basic LIBRARY) to handle the setting up of menus and deal with subsequent menu clicks.

If you’re writing applications regularly this results in displays and chunks of code which are largely the same from app to app and coding then becomes tedious.

The Toolbox, on the other hand, automates a lot of these processes: placing an icon on the icon bar, creation of menus and opening of the ‘About this program’ window being just several examples.

Unfortunately, about the only documentation that has ever been released for the Toolbox is an impenetrable 500-page manual dating back to the 1990s which gives the illusion that Toolbox applications can only be written in C.

This is not so. Toolbox applications can easily be written in BBC Basic. The downside is (you knew I was going to say that!) that there’s a lot of setting up to do. Unlike *Drag ’N Drop* applications which are, for simplicity, presented as single listings, Toolbox apps require an application directory to be set up. Certain files with certain filenames must be present, and initialisation procedures carried out in the Basic *!RunImage*.

A minimum of six files have to be present in an application directory. The diagram on page 21 summarises them for a simple app called (wait for it) *!Simple*.

Three files should be familiar: *!Boot* puts the icons in the *!Sprites* file into the Wimp sprite pool. Here *!Sprites* contains just one icon called *!simple* (the application directory name in lower case) to be shown by the computer in its filer displays.

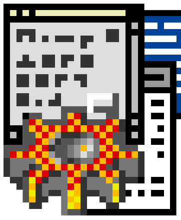
Incidentally if you’re stuck when it comes to designing icons, have a look at the Fast Icons article elsewhere in this issue.

!Boot sets up a system variable, App$Dir so we can refer to the application directory by App$Dir.

No mysteries there, so let’s quickly move on. It's the *Res* and *Messages* file which require explanation.

We’ll come to them in a second. What you should do first of all is to set up your own *!Simple* directory,

Using WimpWorks2



Writing graphical user interfaces from scratch can be a tedious and error prone process on any operating system but with the help of a suitable integrated development environment (IDE), it can be made a lot easier.

A demo version is available for download from the Jaffa Software website www.jaffasoft.co.uk or the application can be purchased for £39.99 and comes on a CD together with an A5 comb bound manual.

Installation is a simple case of creating a directory on a hard drive and dragging the application from the original disc. A big plus of WimpWorks for me is that Andrew Flegg of Jaffa Software provides an excellent after-sales support service.

Setting up WimpWorks2 is fairly straightforward. The commercial version of WimpWorks2 must be activated using the activation code supplied.

The application has an extensive Strong Help file, a PDF version of the hard copy manual and many demo applications.

In the *Editors* directory are some additional examples of editors for WimpWorks, see fig.1 They are installed by copying them to *!WimpWorks.Resources.Editors*, see fig. 2. The WEMs directory (fig.1) contains example WEMs (Window Extension Modules) which may be useful for particular application.

These modules are installed by placing them within the *!WimpWorks.Resources.WEMs* directory. When WimpWorks is next started, the modules are automatically included and ready to be used in a program.

Many users will find !DrawWEM2 of interest since this module that adds several commands to enable easy creation and manipulation of Draw files.

Information on the additional commands offered by each extension can be found from the "Online Help ..." menu option.

The *Xtras* directory shown in Fig.1 contains an application for applying upgrades when available from the Jaffa Soft website.

WimpWorks2 makes it easy for windows and menus to be created graphically and all the programmer is required to write short procedures that are automatically called when the user clicks on a window or icon.

WimpWorks2 achieves all this through four editors, a task editor, a template editor, a menu editor and a subroutine editor.

The subroutine code editor may

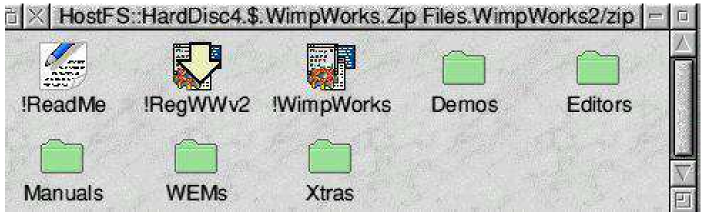


Figure 1

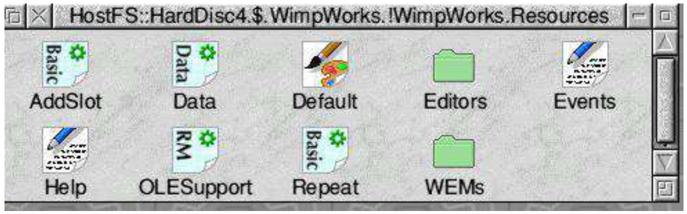


Figure 2. Contents of WimpWorks2 resources directory.