Stardodger I – the Basic version

Stewart Russell shows you how to program the same game three times, in three very different languages

STARDODGER is a very simple game that remains challenging despite the lack of fiddley bits. It was produced as a test exercise to determine the suitability of three languages – Basic, BCPL and Assembly language – for writing a simple game.

The gameplay is about as simple as is humanly possible. The player guides a zig-zag line towards a goal on the other side of the screen, avoiding static objects placed in its path. The higher the level, the more objects are placed on the screen, until the player collides with an object or the white edge of the screen, ending the game.

The game is made even simpler for both programmer and player by the use of the Shift key as the only means of control. If the Shift key is pressed, the line climbs; if it is released, the line falls.

Basic was used to write the original program as it has the advantage of instant access. This allows a deal of experimentation and correction while testing without lengthy recompilation sessions.

Despite its speedy reputation, Locomotive Basic is a little slow for this sort of thing, so the game was later rewritten in a compiled language for speed.

Subprograms

The Basic program was written as a series of simple subprograms tacked together with that most marvellous keyword, GOTO. This keyword gives rise to cheers from some and hisses from others – generally computer scientists who have been bitten by a rabid GOTO at an early age.

The most complex part of the game is the check on whether the pixel in front of the player's line is white; if it is, the game ends. A test is also made to determine whether the right-hand edge of the screen has been reached; this results in the next screen being drawn.

It was found that the game was very slow when written in this manner since two independent tests – TESTR and XPOS – were required every game loop.

This was solved by drawing a vertical line in a different ink to the rest of the game graphics at the right-hand edge of the screen. Now all that was required was one ink test per loop, using TESTR, to see if the player had hit the line or had hit something nasty.

There were plans to include sound – these were shelved when it was discovered that sound programming is not one of my strongest points. This programming technique is know among the cognoscenti as The Cop Out.

As the variable names used are not descriptive, a table of their function may help (see Figure I). These variables hold integer values only, so it could be an interesting (and simple?) exercise to use a Basic compiler on the program.

● *Next month we'll look at the BCPL version.*

```
1 ' Basic Stardodger by SCR.
2 '
10 ' ** Initialise **
20 MODE 1
30 INK 0,0
40 BORDER 0
50 INK 1,26
60 INK 3,0
70 q=5 'set initial asterisks to 5
80 ' ** Title screen **
90 LOCATE 16,1
100 PRINT"Stardodger"
110 LOCATE 1,5
120 PRINT "Avoid the killer Asterisqs,
    and seek the"
130 LOCATE 9,6
140 PRINT "wondrous Nextscreen Gap."
150 LOCATE 12,13
160 PRINT "Use SHIFT to climb"
170 GOSUB 700
180 ' ** Draw game screen **
190 MODE 1
```

```
200 DRAWR 629,0
210 DRAWR 0,170
220 MOVER 0,60
230 DRAWR 0,169
240 DRAWR -629,0
250 DRAWR 0,-399
260 DRAWR 0,2
270 DRAWR 627,0
280 DRAWR 0,168
290 MOVER 0,60
300 DRAWR 0,167
310 DRAWR -625,0
320 DRAWR 0,-399
330 MOVE 636,0
340 DRAW 636,399,3
350 MOVE 638,0
360 DRAW 638,399
370 PLOT -1,-1,1
380 TAG
390 FOR s=1 TO q
400 MOVE 50+RND*561,20+RND*361
410 PRINT "*";
420 NEXT
430 TAGOFF
```

Q	Number of stars plotted on screen. Equal to (screen number)*5.
DY	Vertical line increment. Negative if line is falling, positive otherwise.
T	Ink number tested in front of plotted point. Can be: 0 – No action taken. 1 – Hit something white, hence game over. 2 – Hit invisible lines drawn at right of screen.

Figure I: The main variables in the Basic version

```
440 MOVE 0,200
450 dy=4 'set initial line dir to up
460 ' ** Main game loop **
470 DRAWR 4,dy
480 IF INKEY(21)<>-1 THEN dy=4 ELSE dy
    =-4 'move up if shift pressed
490 t=TESTR(2,dy/2)
500 IF t=1 GOTO 550 'hit summat nasty
510 IF t=3 GOTO 620 'completed the scr
520 MOVER -2,-dy/2
530 GOTO 470 'repeat main loop
540 ' ** End of game screen **
550 MODE 1
560 PRINT TAB(16);"YOU GOOFED"
570 LOCATE 5,13
580 PRINT "Number of Screens completed
    = "+STR$((q/5)-1)
590 GOSUB 700 'Press any key to cont..
600 RUN
610 ' ** Success screen **
620 MODE 1
630 PRINT TAB(16);"WELL  DONE"
640 LOCATE 10,13
650 PRINT "Stand by for Screen "+STR$(
    (q/5)+1)
660 GOSUB 700
670 q=q+5 'add 5 stars to next screen
680 GOTO 190 'screen drawing routine
690 ' ** Wait for key **
700 LOCATE 8,25
710 PRINT "Press any key to continue."
720 WHILE INKEY$<>""
730 WEND
740 WHILE INKEY$=""
750 WEND
760 RETURN
```