Débogage d'un programme Basic avec Euphoric

Un petit truc simple et efficace !

par André C.

Quelques manuels d'initiation à la programmation Basic comportent un chapitre concernant le débogage ou au minimum quelques conseils. On trouve aussi des infos et des conseils dans les magazines informatiques. Le débogage est un problème récurrent sur lequel le CEO-mag est souvent revenu. Mais je ne pense pas qu'il existe un manuel spécialement dédié au débogage. Pour faire un peu d'humour, il est de bon ton de citer l'adage : "Si le débogage consiste à retire les bogues d'un programme, alors la programmation est forcément le processus qui consiste à les y mettre". Pourtant ce n'est pas de l'humour : rien n'est plus vrai !

Il n'existe pas de solution miracle (sauf celle que je vous propose, œuf corse). La cause d'une bogue n'est pas toujours évidente et sa recherche peut parfois être exaspérante. Pour les programmes en langage machine, j'ai largement utilisé le débogueur d'Euphoric (voir le mode d'emploi dans le CEO-mag n°291-292 de Juillet-Août 2014, pages 40 à 48).

Mais c'est moins simple pour les programmes Basic. En effet, le programme est peu lisible en mémoire, l'accès aux variables n'est pas facile et surtout la machine passe l'essentiel de son temps à exécuter du code en Rom (les commandes Basic correspondent à des routines en Rom). En pratique lorsqu'on appuie sur F11 pour entrer dans le déboguer, on se retrouve dans 99% des cas dans les routines de la Rom. Et déboguer celles-ci n'est pas l'objectif du jour !

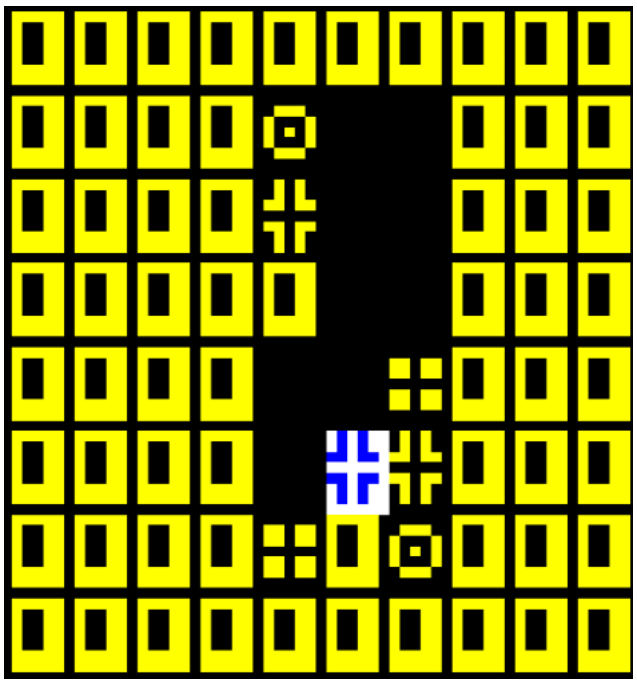
Donc la première chose à faire est de programmer de façon claire, même si ce n'est pas forcement optimal. Facile à dire ! Essayez quand même d'avoir toujours à l'esprit que le débogage prend bien plus de temps que la programmation…

Au menu des conseils généralement dispensés figurent les commandes STOP, CONT, END, RUN n°_de_ligne, WAIT, GET R$, TRON, TROFF, etc.). Et lorsqu'un programme est arrêté, il est possible d'afficher la ou les variables qui posent problème (PRINT, PRINT@, PLOT). Ce dont je voudrais vous parler aujourd'hui, c'est d'un petit truc pour utiliser le débogueur d'Euphoric avec un programme Basic.

Récemment, je me suis penché sur Supervex un programme que j'avais écrit il y a quelques années pour le Super-Oric, afin de l'adapter à l'Atmos et que je viens de diffuser sous le nom de VEXED1. A l'époque, ce jeu m'avait couté beaucoup d'efforts au niveau du moteur de désintégration. En effet, suite à chaque mouvement, il peut y avoir des blocs qui chutent, ce qui entraîne d'éventuelles désintégrations et peut-être de nouvelles chutes etc. Le test du jeu avait été poussé très loin.

Et pourtant il subsistait une bogue ! Une bogue dont la probabilité d'occurrence est tellement faible qu'elle est restée invisible jusqu'à maintenant. Vous me direz que c'est le propre de tout programme informatique (cf. les failles trouvées dans les programmes de Micromou).

Toujours est-il qu'il existe une situation dans laquelle deux blocs identiques subsistent côte à côte sans se désintégrer. Le jeu n'est pas bloqué, mais cela fait désordre !



Pour résoudre ce problème, j'ai commencé par

```
1460 BX=CX:BY=CY:BL=CU:DOKE#9900,BL
1504 KILL=FALSE:GOSUB 3700:DOKE#9904,BL
1506 TX=BX:TY=BY-1:DOKE#9910,TX:DOKE#9912,TY:DOKE#9914,BL:'Sup
1507 DOKE#9916,SCRN(TX,TY):DOKE#9918,BL:GOSUB 1600
1508 TX=BX:TY=BY+1:DOKE#9920,TX:DOKE#9922,TY:DOKE#9924,BL:'Inf
1509 DOKE#9926,SCRN(TX,TY):DOKE#9928,BL:GOSUB 1600
1510 TX=BX-1:TY=BY:DOKE#9930,TX:DOKE#9932,TY:DOKE#9934,BL:'Gauche
1511 DOKE#9936,SCRN(TX,TY):DOKE#9938,BL:GOSUB 1600
1512 TX=BX+1:TY=BY:DOKE#9940,TX:DOKE#9942,TY:DOKE#9944,BL:'Droite
1513 DOKE#9946,SCRN(TX,TY):DOKE#9948,BL:GOSUB 1600
1516 DOKE#994A,BL:GET R$:RETURN
1517 BL=32:DOKE#994A,BL:GET R$
3720 IF SCRN(BX,BY+1)<>32 THEN DOKE#9902,BL:RETURN
```

afficher à l'écran (à côté de la zone de jeu) les valeurs des variables possiblement incriminées. Mais la place à l'écran est limitée, car il faut utiliser des PRINT@ et non des PRINT, afin de ne pas faire scroller l'écran, donc le jeu. Même en plaçant des GET R$ dans les boucles, pour faire des pauses, cela s'est vite révélé inutilisable (les boucles tournent très vite et c'est difficile à suivre).

Il me semblait que si je pouvais disposer d'un affichage récapitulatif il serait plus facile de se faire une idée d'ensemble et suivre l'évolution des variables. Tilt ! C'est possible ! Il suffit de copier les informations cruciales dans une zone mémoire inutilisée par le programme et de visualiser cette zone avec le débogueur d'Euphoric.

En pratique, cela veut dire placer des DOKEs dans le programme par exemple après chaque changement de la valeur des paramètres importants. Ces DOKE peuvent éventuellement être suivis d'un GET R$ afin d'introduire une pause dans le déroulement du programme et de pouvoir consulter l'écran du débogueur. Sur la figure ci-dessus vous pouvez voir les ajouts de débogage que j'ai utilisés pour VEXED1. Les DOKEs ont été listés avec le commande SEEK CHR$(138).

Dans cet exemple, j'ai donc placé des "DOKE#99xx,variable" suivit d'un GET R$ aux points stratégiques. Un appui sur F11 suivit d'un D9900 permet d'afficher la mémoire à partir de #9900 en Ram. Un second F11 permet de revenir au jeu, toujours en attente d'une réponse pour le GET R$.

Les figures ci-dessous montrent ce que cela donne. Sur la première ligne, on voit l'évolution (ici la stabilité) de la variable BL (code ascii du bloc sous le curseur). Sur chacune des lignes suivantes on a successivement les coordonnées TX et TY, puis la variable BL, la lecture du code ascii du bloc sous le curseur et enfin à nouveau BL.

Je n'entre pas dans le détail qui est sans intérêt pour vous. C'est le principe qui compte. On peut bien sûr disposer les données afin qu'elles soient lisibles de manière évidente, par exemple indexer l'adresse des DOKEs avec l'index des boucles (DOKE#9900+J,variable).

En résumé, il faut écrire en mémoire ce qu'on aimerait savoir pour comprendre ce qui se passe, placer des GET R$ pour "figer" l'exécution afin de pouvoir presser sur F11 et lire ce que ça donne en mémoire. Difficile de faire plus simple comme procédé de débogage !

J'espère que mon petit truc vous sera utile et vous souhaite de profiter pleinement du plaisir de programmer. A+ André