Déplacer un objet dans l'écran texte

Par André C.

NOSTALGIE

Je suis sûr que vous vous souvenez du jour où vous avez déballé votre Oric. Une des premières choses que vous avez sans doute apprise a été à redéfinir un caractère et à le promener dans l'écran texte. Qu'il s'agisse de reproduire un jeu publié ou d'en bricoler un de son cru, dans la plupart des cas on est amené à utiliser le plus simple des sprites : le caractère redéfini. Quoi de plus simple en effet que de redéfinir un caractère et d'utiliser les commandes PRINT@ ou POKE ? Notez qu'un vrai sprite est une facilité matérielle fonctionnant avec certains ordinateurs, mais qui n'existe pas dans le hardware de l'Oric.

PSEUDO SPRITES

Dans notre cas, on est obligé de simuler un sprite, par exemple en assemblant plusieurs caractères. Dans ce qui suit, c'est donc abusivement que j'utiliserai le mot "sprite" au lieu de "pseudo sprite". Les choses peuvent sembler se compliquer si le sprite que l'on veut mettre en place est constitué de plusieurs caractères. Mais avec un minimum d'organisation, c'est très facile.

PRINT@ ou PLOT?

Sauf si vous aimez les complications, laissez tomber les PRINT@. En effet, à l'usage, la commande PLOT est bien plus agréable à utiliser. Pour en avoir le cœur net, j'ai expérimenté les deux et il n'y a pas photo, PLOT gagne par une longueur d'avance sur PRINT@!

DEPLACER UN CARACTERE UNIQUE

Pour un caractère unique, tout ce dont on a besoin (outre les 8 octets de redéfinition) ce sont les coordonnées X,Y de l'endroit où on veut l'envoyer dans l'écran. Pour le bouger d'une case, il suffit d'afficher un espace à sa position initiale (pour effacer), puis d'incrémenter/décrémenter X, Y (selon la direction du mouvement) d'une unité et de réafficher le caractère. Il est possible de faire bondir le caractère de plusieurs cases, mais si l'on procède par déplacement d'une case à la fois le Basic se révèle assez rapide pour produire un effet fluide. Je laisse ici de côté les tests de collision, rebond, etc., qui ne sont pas du propos de cet article.

DEPLACER UN SPRITE

Le déplacement "simultané" de plusieurs caractères prend plus de temps, beaucoup de temps. Pour que le sprite se retrouve à côté de sa position initiale, il doit être déplacé de plusieurs cases. Faut-il le faire en un seul bond ou bien décomposer le mouvement en plusieurs petits déplacements d'une case? Le programme peut exiger des déplacements pas à pas (je pense à Rush Hour de Fabrice F. par exemple), mais ce n'est pas toujours le cas. Généralement, la solution la plus viable consiste à faire bondir son sprite de plusieurs cases d'un coup. Si cela ne s'accorde pas avec le contexte du programme, il faut envisager de passer au langage machine, beaucoup plus rapide et donc plus fluide…

IDENTIFICATION DES SPRITES

Le code Ascii suffisait pour identifier un caractère isolé. Mais il va falloir ruser pour identifier un sprite. Voici quelques solutions possibles :

a) On peut inclure un numéro dans le dessin de redéfinition du sprite (c'est la solution retenue par Fabrice pour Rush Hour).
b) On peut utiliser le code Ascii de son 1er élément (le caractère situé en haut à gauche). Si un même caractère redéfini est utilisé dans le coin en haut à gauche de plusieurs sprites, il faut redéfinir plusieurs Ascii avec le même dessin de ce coin gauche.
c) On peut attribuer un numéro à chacun des sprites, avec création d'un tableau de correspondance. C'est possible, mais plus compliqué.

Voici ensuite quelques possibilités pour désigner le sprite à déplacer :

a) Taper le numéro du sprite (cf. Rush Hour).
b) Déplacer un curseur et presser la barre d'espace (cf. Vexed2).
c) Mettre en œuvre la fonction SCRN(X,Y) pour identifier le code Ascii.
d) Ou utiliser la logique du jeu lui-même (cf. Pinball etc.).

AFFICHAGE D'UN SPRITE

C'est aussi simple que l'affichage d'un caractère isolé. Pour un sprite de 6 caractères, sur 2 lignes de 3 caractères, il faut disposer d'un sous-programme du genre :

```
10000 ' Affiche le sprite n°1 en X,Y
10010 PLOTX,Y,"abc"
10020 PLOTX,Y+1,"def"
10030 RETURN
```

DEPLACEMENT D'UN SPRITE PAS A PAS

C'est le truc utilisé par Fabrice pour Rush Hour. Pour déplacer un caractère "c" d'une case vers la gauche (par exemple), on utiliserai simplement PLOTX-1,Y,"c□" (le symbole □ représente ici un espace ou un caractère de fond d'écran pour effacer l'ancienne position). Par analogie, pour déplacer un sprite formé de 2 rangées de 3 caractères d'une seule case vers la gauche, on fait appel à un sous-programme du type:

```
10100 ' Déplace d'une case vers la gauche
10110 PLOTX-1,Y,"abc□"
10120 PLOTX-1,Y+1,"def□"
10130 RETURN
```

Le caractère de fond d'écran (symbole □) après le "c" et après le "f", sert à effacer la trace précédente du pseudo sprite. Pour que le sprite se place complètement à gauche de sa position initiale, il faudra donc appeler 3 fois ce sous-programme.

RATIONALISATION DES S/PROGRAMMES

Il s'ensuit que, pour un sprite donné, vous devrez disposer d'autant de sous-programmes que de directions à gérer, c'est-à-dire 4 pour déplacer dans les 4 directions et même 8 s'il y a des déplacements en diagonale. Avec plusieurs sprites, le nombre de sous-programmes devient rédhibitoire. Il faut donc absolument rationaliser. Pour cela, selon la direction de déplacement, nous allons utiliser DX et DY (delta X et delta Y) selon le tableau ci-contre.

L'unique sous-programme ressemblera alors à cela :

```
10000 ' Efface le sprite en X,Y (ancienne position)
10010 PLOTX,Y,"   "
10020 PLOTX,Y+1,"   "
10030 ' Affiche le sprite à sa nouvelle position
10040 PLOTX+DX,Y+DY,"abc"
10050 PLOTX+DX,Y+DY+1,"def"
10060  RETURN
```

Simple non ? Observez que la multiplicité des directions nous a obligé dédoubler l'opération, qui comprend maintenant une phase d'effacement, puis une phase de réaffichage. Reste que si vous avez de nombreux sprites, il va falloir rationaliser encore plus la 2ème partie de ce sous-programme, ainsi par exemple :

```
10030 ' Affiche le sprite à sa nouvelle position
10040 PLOTX+DX,Y+DY,A$
10050 PLOTX+DX,Y+DY+1,B$
10060 RETURN
```

Il suffit alors d'initialiser correctement X, Y, DX, DY, A$ et B$ avant d'appeler cet unique sous-programme.

DX	DY	Sens	DX	DY	Sens
1	0	→	1	1	↘
0	1	↓	-1	1	↙
-1	0	←	-1	-1	↖
0	-1	↑	1	-1	↗

EXEMPLE PEDAGOGIQUE

```
1000 ' Avec 3 sprites et 4 directions
1010 ' Coordonnees initiales
1020 X(1)=10:Y(1)=10:X(2)=10:Y(2)=15:X(3)=10:Y(3)=20
1030 ' Initialisation des 2 chaînes de 3 caractères pour chacun des 3 sprites
1040 A$(1)="abc":B$(1)="def":A$(2)="ghi":B$(2)="jkl":A$(3)="mno":B$(3)="pqr"
1050 ' Affichage initial des 3 sprites
1060 CLS:FOR I=1 TO 3
1070 PLOTX(I),Y(I),A$(I):PLOTX(I),Y(I)+1,B$(I)
1080 NEXT
1090 ' Saisie numéro du sprite a deplacer
1100 PLOT2,0,"No (1 a 3)?":GET N$:PLOT2,0,"          "
1110 IF N$>="1" AND N$<="3" THEN N=ASC(N$)-48 ELSE 1100
1120 ' Saisie de la direction, initialisation de DX, DY et deplacement du sprite
1130 PLOT2,0,"Direction ?":GET D$: PLOT2,0,"          "
1140 IF D$>=CHR$(8) AND D$<=CHR$(11) THEN D=ASC(D$)-7 ELSE 1130
1150 IF D=1 THEN DX=-1: DY=0:GOSUB 10010:GOTO 1100:' Gauche
1160 IF D=2 THEN DX=+1: DY=0:GOSUB 10010:GOTO 1100:' Droite
1170 IF D=3 THEN DX=0: DY=+1:GOSUB 10010:GOTO 1100:' Bas
1180 IF D=4 THEN DX=0: DY=-1:GOSUB 10010:GOTO 1100:' Haut
10000 ' Efface le sprite No N situé en X(N),Y(N) (ancienne position)
10010 PLOTX(N),Y(N),"   "
10020 PLOTX(N),Y(N)+1,"   "
10030 ' Affiche le sprite No N à sa nouvelle position
10040 X(N)=X(N)+DX:Y(N)=Y(N)+DY
10050 PLOTX(N),Y(N),A$(N)
10060 PLOTX(N),Y(N)+1,B$(N)
10060 RETURN
```

Ce petit programme (que vous trouverez sur la disquette accompagnant ce mag, sous le nom DEMBALA0.COM) montre seulement le principe, il doit bien sûr être adapté. Ainsi, avant de déplacer un sprite, il faut d'abord s'assure que la place est libre dans la direction envisagée ou alors gérer un sous-programme de collision, voire de rebond.

RETOUR SUR LE DEPLACEMENT PAS A PAS : DEMO PLUS REALISTE

Voici un petit programme de démo permettant de déplacer un sprite de 2x2 caractères dans l'écran texte. Un contrôle des limites de l'écran a été ajouté, avec respect des 2 colonnes de gauche (attributs) et, par suite, respect de 2 colonnes à droite, ainsi que de 2 lignes en haut et en bas (ah! la symétrie). Les nouvelles coordonnées X et Y sont à jour à chaque déplacement (ligne 1030). Le sprite faisant 2x2 caractères, 2 déplacements sont nécessaires pour positionner le sprite. Vous trouverez cet exemple avec la disquette, sous le nom DEMBALA1.COM.

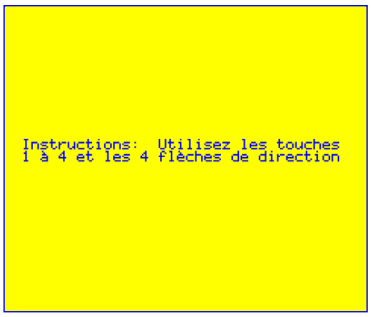
```
100 ' Petite demo baladeuse 1, Andrec 2017
110 ' Efface l'ecran et la ligne service
120 CLS:FOR I=#BB81 TO #BBA7:POKE I,32:NEXT
130 ' Encre rouge sur fond jaune
140 INK1:PAPER3:POKE#BB80,PEEK(#26B)
150 ' Efface le curseur
160 POKE#26A,(PEEK(#26A)AND#FE):PRINT""
170 ' Redefinition des caracteres
180 GOSUB 6000
190 ' Affichage initial du sprite de 2x2 caracteres
200 X=16:Y=13: PLOTX,Y,"ab":PLOTX,Y+1,"cd"
210 ' Saisie direction
220 GET D$:
230 IF D$>=CHR$(8) AND D$<=CHR$(11) THEN D=ASC(D$)-7 ELSE 220
240 IF D=1 AND X>= 4 THEN DX=-1: DY=0 :GOTO 1010:' Gauche
250 IF D=2 AND X<=34 THEN DX=+1: DY=0 :GOTO 1010:' Droite
260 IF D=3 AND Y<=21 THEN DX=0 : DY=+1:GOTO 1010:' Bas
270 IF D=4 AND Y>= 3 THEN DX=0 : DY=-1:GOTO 1010:' Haut
280 GOTO 220
1000 ' Efface le sprite situe en X,Y (ancienne position)
1010 PLOTX,Y,"  ":PLOTX,Y+1,"  "
1020 ' Affiche le sprite a sa nouvelle position
1030 X=X+DX:Y=Y+DY
1040 PLOTX,Y,"ab":PLOTX,Y+1,"cd"
1050 ' Saisie d'une nouvelle direction
1060 GOTO 220
6000 ' Redefinition des 4 caracteres abcd
6010 FOR I=#B708 TO #B727:READ V:POKE I,V:NEXT
6020 DATA 63,32,47,40,43,42,42,42
6030 DATA 63,1,61,5,53,21,21,21
6040 DATA 42,42,42,42,42,43,40,47
6050 DATA 21,21,21,53,5,61,1,63
6060 RETURN
```

DEMO AVEC DEPLACEMENT PAR BONDS

Pour finir, voici un petit programme permettant de déplacer 4 sprites de 2x3 caractères dans l'écran texte. Pour simplifier les caractères n'ont pas été redéfinis. Les sprites se déplacent par bons de 3 cases à l'horizontale et de 2 cases à la verticale. Les nouvelles coordonnées X et Y sont mises à jour à chaque déplacement (ligne 570). Le contrôle des limites de l'écran est toujours présent et un test de collision a été ajouté. Pour déplacer un sprite, il faut le sélectionner en tapant son numéro, (visible dans le coin gauche en haut), puis presser une flèche de direction. Le sprite reste sélectionné jusqu'à l'indication d'un autre numéro. Vous trouverez cet exemple avec la disquette, sous le nom DEMBALA2.COM, ainsi que DEMBALA3.COM (déplacement de 4 sprites de 3 lignes de 3 caractères).

```
100 ' Demo baladeuse 2 pour 4 sprites de 6 caracteres
110 ' *** Andrec V25042017
120 ' Initialisation de A$(N) et B$(N) selon le numero du sprite
130 ' Initialisation de X(N) et Y(N) avec les coord initiales des sprites
140 DIM A$(4):DIM B$(4):DIM X(4):DIM Y(4)
150 A$(1)="1ab":B$(1)="cde":X(1)=14:Y(1)=9
160 A$(2)="2fg":B$(2)="hij":X(2)=26:Y(2)=9
170 A$(3)="3kl":B$(3)="mno":X(3)=14:Y(3)=15
180 A$(4)="4pq":B$(4)="rst":X(4)=26:Y(4)=15
190 ' Les delta DX & DY dependront des fleches pressees au clavier
200 ' Efface l'ecran et la ligne service
```

```
210 CLS:FOR I=#BB81 TO #BBA7:POKE I,32:NEXT
220 ' Encre bleue sur fond jaune
230 INK4:PAPER3:POKE#BB80,PEEK(#26B)
240 GOSUB 630:' Instructions
250 ' Affiche les 4 sprites a leur position initiale
260 FOR N=1 TO 4
270 PLOT X(N),Y(N),A$(N)
280 PLOT X(N),Y(N)+1,B$(N)
290 NEXT
300 N=1:DX=0:DY=0: ' No du sprite et deltas par defaut
310 ' Saisie clavier : No du sprite de 1 a 4 ou l'une des 4 fleches
320 GET R$
330 IF R$>="1" AND R$<="4" THEN N=ASC(R$)-48:GOTO 320
340 ' Ce n'est pas un No de sprite, est-ce une des 4 fleches
350 IF R$>=CHR$(8) AND R$<=CHR$(11) THEN D=ASC(R$)-7 ELSE 320
360 ON D GOTO 370,410,450,490
370 ' Gauche
380 IF X(N)<4 THEN 320: 'Impossible, nouv saisie
390 IF SCRN(X(N)-3,Y(N))<>32 THEN 320: 'Impossible, nouv saisie
400 DX=-3:DY=0:GOSUB 540:GOTO 320
410 ' Droite
420 IF X(N)>33 THEN 320: 'Impossible, nouv saisie
430 IF SCRN(X(N)+3,Y(N))<>32 THEN 320: 'Impossible, nouv saisie
440 DX=+3:DY=0:GOSUB 540:GOTO 320
450 ' Bas
460 IF Y(N)>21 THEN 320: 'Impossible, nouv saisie
470 IF SCRN(X(N),Y(N)+2)<>32 THEN 320: 'Impossible, nouv saisie
480 DX=0:DY=+2:GOSUB 540:GOTO 320
490 ' Haut
500 IF Y(N)<3 THEN 320: 'Impossible, nouv saisie
510 IF SCRN(X(N),Y(N)-2)<>32 THEN 320: 'Impossible, nouv saisie
520 DX=0:DY=-2:GOSUB 540:GOTO 320
530 ' Efface le sprite situe en X,Y (ancienne position)
540 PLOT X(N),Y(N),"   "
550 PLOT X(N),Y(N)+1,"   "
560 ' Actualise les coordonnees du sprite
570 X(N)=X(N)+DX:Y(N)=Y(N)+DY
580 ' Affiche le sprite a sa nouvelle position
590 PLOT X(N),Y(N),A$(N)
600 PLOT X(N),Y(N)+1,B$(N)
610 RETURN
620 ' Affiche les instructions
630 POKE#26A,(PEEK(#26A)AND#FE):PRINT"": ' Efface le curseur
640 ACCENT SET
650 PLOT2,10,"Instructions:  Utilisez les touches"
660 PLOT2,11,"1 "+CHR$(64)+" 4 et les 4 fl}ches de direction"
670 GET R$:CLS:RETURN
```



J'espère que ce petit bout de programme vous inspirera une application. Rappelez-vous: il n'existe pas de programme modeste, l'important est de faire soi-même. C'est tout l'intérêt de nos vieux 8-bits !

Bon amusement avec votre Oric !