

● S'il s'agit d'une chaîne de caractères contrôlée lors de la saisie (longueur, type de caractères, touche de validation), formatée en édition transférée dans une variable Basic ou, inversement, chargée à partir d'une variable, nous recourons au groupe d'instructions TOOL concernant les zones-écran : ces zones sont préalablement définies par une déclaration de type DECZ sur laquelle nous nous étions longuement attardés dans le précédent numéro de ce magazine.

Décrivons (ligne 30) notre zone date, en lui attribuant le numéro d'identification 1 : elle débute ligne 4 en colonne 11 et a une longueur de 15 caractères maximum. Le format de saisie est libre : 24/06/1985 ou lundi 24 juin ou 24 juin 85 etc..

● La composition de la date s'exprime ensuite de façon très simple par l'instruction REQZ suivie du numéro de la zone : 10 REQZ1.

L'exécution de cette instruction positionne le curseur en début de zone. Toutes les touches de l'éditeur de texte, CLR, HOME, CRSR, INST/DEL... Peuvent être utilisées en cours de saisie dans la zone.

● Le transfert du contenu de la zone dans une variable Basic chaîne de caractères (ligne 50) est réalisé par l'instruction INZ : 50 INZ1, d\$.

La variable chaîne d\$ qui reçoit le contenu de la zone-écran aura la même longueur que cette zone, quelque soit le contenu de celle-ci : 15 caractères dans notre exemple.

Il nous est facile maintenant de tester le contenu de la variable d\$:

● aucune date n'est composée : d\$ contient 15 espaces (et non d\$="" si l'on tient compte de la précédente remarque). Nous sommes dans le cas d'une demande d'affichage de la page modèle créée par FOLIO1 et archivée sous le nom "agenda" (cf. ligne 570 de ce programme). La page "agenda" est chargée en mémoire et affichée (ligne 60) : SLOAD8, "agenda".

● dans le cas contraire, l'instruction SLOAD8, d\$ de la ligne 70 affiche la page de l'agenda correspondant à la date composée.

Nous ne pourrions pas tester ce dernier point dans l'immediat, puisqu'aucune page n'a encore été créée. Nous allons nous y employer. Appuyons sur la touche RETURN : nous retrouvons notre page modèle à l'écran.

Listing - lignes 90 à 100-

Il nous faut maintenant dresser la liste des opérations que nous serons amenés à réaliser sur une page de l'agenda :

- composer la date qui servira de nom, d'identificateur d'une page nouvelle
- se positionner sur une ligne du planning horaire et composer ou corriger un texte
- placer un index de couleur en fin de ligne afin de mieux faire ressortir l'emploi du temps journalier par des codes couleurs
- effacer texte et index d'une ligne
- effacer une page de l'agenda, la supprimer de la disquette
- abandonner la consultation d'une page
- mémoriser la nouvelle page ou la page modifiée sur disquette
- sans oublier les fonctions mémos et frais qui seront traitées par la suite.

Pour la commodité de l'utilisateur, le choix d'une de ces opérations se fera par appui sur la touche du clavier dont le caractère se rapproche le plus de la fonction qui lui est associée :

- (J) Pour jour,
- (CRSR) Pour le positionnement sur une ligne,
- (C) Pour fixer la couleur des index,
- (G) Pour gommer une ligne,
- (E) Pour effacer une page,
- (Q) Pour quitter la page en cours,
- (s) Pour la sauvegarder sur disquette,
- (M) et (F) Pour mémos et frais.

Le couple d'instructions CARGET et ON..GOTO gère ces choix.

CARGET est une instruction TOOL, cousine de l'instruction Basic GET ; la seule ressemblance : elle prévoit toutes deux l'entrée d'un seul caractère au clavier. CARGET interrompt, par contre, l'exécution du programme, identifie les caractères autorisés par référence à une liste, et signale son attente, à la demande, par un curseur clignotant.

Ainsi CARGET "az3*", 5,3 signifie : arrêt du programme tant qu'un des caractères a, z, 3 ou * n'est pas frappé, et curseur clignotant sur la ligne 5 en colonne 3, ces deux paramètres étant facultatifs. Vous pouvez faire varier la fréquence du clignotement par un POKE 703, n : c'est ce qui est prévu en ligne 10 du programme avec une valeur de n égale à 100.

CARGET g\$, où g\$ est une chaîne vide (g\$=""), autorise tous les caractères du clavier.

Le contenu des deux variables ok et zo, réservées par TOOL, varie en fonction de la touche sélectionnée : ok est égal au rang de la touche dans la liste (ok=1 si a, 2 si z, 4 si * pour reprendre notre exemple), tandis que zo contient le code ASCII du caractère choisi.

Il est évident qu'une seule instruction ON ok GOTO... à la suite de l'instruction CARGET, nous évitera la fastidieuse répétition des IF g\$="a" goto... etc., habituelle avec get !

Un regret toutefois (et une suggestion à l'intention des auteurs) : cette même instruction CARGET ne prévoit pas de paramètre rendant facultatif l'arrêt de l'exécution du programme ; je puis vous assurer que cela éviterait, dans un grand nombre de cas, le recours rétrograde à GET !

Examinons les différents choix offerts à l'utilisateur (ligne 90) en fonction des valeurs de la variable ok (ligne 100).

Listing - lignes 100 à 120-

L'appui sur la touche J renvoie à la ligne 110 : composition de la date dans la zone 1 et transfert du contenu de cette zone dans la variable d\$. Attention ! la déclaration de la zone 1 (DECZ1) qui permet d'utiliser les instructions REQZ1 et INZ1 qui s'y réfèrent, n'est pas celle de la ligne 30 malgré les apparences, mais bien celle de la ligne 270 du programme FOLIO1. Une zone donnée, en effet, déclarée par l'instruction DECZ qui lui attribue un numéro, une longueur, un format, fait partie intégrante de la page-écran sur laquelle elle figure, au même titre qu'une trace de ligne, une couleur ou un libellé de colonne. Lorsque nous avons lu la page "agenda" sur la disquette (ligne 60), nous avons transféré dans la mémoire la totalité des informations s'y