Utilisation de Turbo Pascal 7.0

La procédure ReadLn et le clavier

Par haypo

Date de publication : 24 juin 2001

Dernière mise à jour : 17 février 2016

DÉBUTANT

Cette série de tutoriels est destinée à aider les débutants à prendre en main Turbo Pascal 7.0.

Cette partie détaille le fonctionnement de la procédure ReadLn et la gestion du clavier.

Commentez

I - Procédure ReadLn

La procédure **ReadLn** permet de capturer des chaînes de caractères tapées par l'utilisateur. Un exemple :

```
var Nom: String;
begin
  Write ('Entrez votre nom : ');
  ReadLn (Nom);
  Writeln ('Salut ',Nom);
end.
```

Téléchargez l'exemple **ReadLn.pas**.

La fonction ReadLn a donc besoin d'une variable où stocker le résultat (ici « Nom ») de type String et l'utilisateur peut entrer des données à l'aide de toutes les touches du clavier : retour pour effacer du texte et entrée pour valider.

II - Le clavier

Le clavier est accessible par les fonctions **KeyPressed** et **ReadKey** (fonctions de l'unité **CRT**). Mais vous pouvez aussi utiliser les fonctions de mon unité « Clavier.tpu » : « TouchPresse » et « LitTouche ». Un exemple :

```
uses Clavier;
var touche: char;
begin
  Write ('Pressez une touche');
  repeat
    { Ne fait rien en attendant une touche }
  until TouchPresse;
  touche := LitTouche;
  Writeln;
  Writeln ('Vous avez tapé la touche de code ASCII ',ord(touche),' ce qui donne : ',touche);
end.
```

Téléchargez l'exemple **LiTouche.pas**.

On voit donc que la fonction « TouchPresse » (ou KeyPressed) renvoie un type **boolean** - True (vrai) ou False (faux) - informant si une touche a été pressée ou non.

La fonction « LitTouche » lit une touche dans le tampon clavier et renvoie son code ASCII. Le gros avantage sur la fonction ReadKey est que si la touche est étendue (touches fléchées, suppression, fin, etc.), le code renvoyé n'est pas 0 (code ASCII #0) mais le code ASCII additionné de 128. Par exemple, la touche « flèche haut » sera codée 0 puis 72 par la fonction ReadKey alors que « LitTouche » renvoie directement 200 (72+128).

Veuillez également consulter le code source de l'unité « Clavier.tpu » (fichier Clavier.pas) pour voir les constantes :

- « _HAUT » pour la touche haut (vaut #200) ;
- « _ECHAPE » pour la touche escape (vaut #8) ;
- « _ENTREE » pour la touche entrée (vaut #13), etc.

Pour connaître tous les codes ASCII du clavier, téléchargez le programme **ClavAsci.zip** (il est situé dans la rubrique Périphérique de TPascal).

Souvent, on a besoin de coder une interface permettant de se déplacer. Pour cela, les touches directionnelles sont les plus adaptées. Je ne vais pas vous donner un exemple complet, mais seulement la structure :

```
uses Clavier;
var ch: char;   { Touche lue au clavier }
```

```
procedure DeplaceListe (Plus: Integer); var TmpL: LongInt; begin
TmpL := IndexListe +Plus;
if TmpL<MINIMUM then TmpL := MINIMUM;
if MAXIMUM<TmpL then TmpL := MAXIMUM;
if IndexListe=TmpL then exit; { Aucun changement, autant quitter }

{ Efface l'ancien élément }
IndexListe := TmpL;
{ Dessine le nouvel élément }
end;

begin
   { code de début }
  repeat
    if TouchPresse then begin
      ch := LitTouche;
      case ch of
        _PAGEHAUT: DeplaceListe (-10);
        _HAUT: DeplaceListe (-1);
        _BAS: DeplaceListe (+1);
        _PAGEBAS: DeplaceListe (+10);
      end;
    end;
    { code exécuté en attendant une touche (animation ?) }
  until ch=_ECHAPE;
  { code de fin }
end.
```

Pour ceux qui veulent aller plus loin, téléchargez le programme **Clav_int.zip**. Celui-ci utilise mon autre unité de gestion du clavier : « SClavier » (Super Clavier !). Elle reprogramme la gestion du clavier pour permettre d'accéder à toutes les touches et offre un tableau avec la liste des touches et leur état : enfoncée ou non. Très pratique pour les jeux, pour le déplacement en diagonale par exemple : HAUT+GAUCHE.

III - Remerciements

Merci à **Damien Genthial** pour ses corrections.