Impression avec Delphi 1 sous Windows 9x

Par Jean-Marc Quere

Utilisation de l'interruption $17

Tous les utilisateurs n'ayant pas adopté Windows 9x ou NT, il est encore parfois nécessaire d'utiliser Delphi 1 pour réaliser une application pour W3.1 ou WfW. Heureusement fourni sur le CD de Delphi 2 et Delphi 3, son installation ne pose aucun problème sous W95 et encore moins pour W3.x (environnement pour lequel il a été initialement conçu). Afin de ne pas être incité à donner une apparence trop "à la 95" à une application qui finalement devra fonctionner sous W3.x, j'utilise cette plate-forme pour le développement. Un jour, pressé par le temps, j'ai réalisé des modifications d'une application pour W3.x sous W95 (avec Delphi 1) et là surprise... les éditions "plantent" le PC lorsque l'imprimante n'est pas en ligne !

L'objet de la réalisation est d'assurer une gestion correcte de l'impression sous Windows 95 pour une application réalisée en Delphi 1 pilotant l'imprimante en direct (ce qui est nécessaire pour les imprimantes sans pilote pour Windows : imprimantes d'étiquettes barres-codées thermiques et à transfert thermiques par exemple).

En conclusion la méthode classique et rapide d'édition ci-dessous ne fonctionne pas sous Windows 95 :

```
...
var
FPrn : Text;
begin
try
AssignFile(FPrn, 'LPT1');
Rewrite(FPrn);
Writeln(FPrn, 'TEST');
except
on EIOError do MessageDlg('Erreur !', mtWarning, [mbOK], 0);
end;
CloseFile(Fprn);
end;
...
```

Lorsque l'imprimante est hors service, l'exception EIOError n'est pas déclenchée alors que sous Windows 3.x tout fonctionne à la perfection. L'objet de l'unité "Imprimer" décrite est de disposer du même mécanisme en évitant d'utiliser Windows 95 en passant directement par le BIOS du PC.

En pratique

La nouvelle unité doit être ajoutée à la clause "uses" de votre unité. Et le programme précédant devient :

```
...
begin
try
prn_open(prn_LPT1);
prn_writeln(prn_LPT1, 'TEST');
except
on EPrinterError do MessageDlg('Erreur !', mtWarning, [mbOK], 0);
end;
prn_close(prn_LPT1);
end;
...
```

Vous n'avez plus besoin de déclarer une variable de type "fichier texte". Le tableau ci-dessous indique la correspondance des nouvelles instructions et celles de bases.

Méthode usuelle avec l'unité "imprimer"

AssignFile(...) .....**prn_open**(prn_LPTx) où x prend pour valeur 1, 2, 3 (pour LPT1 à LPT3)

Writeln(...) .....**prn_writeln**(prn_LPTx, " texte ")

Close(...) .....**prn_close**(prn_LPTx)

A noter la mise en place d'une nouvelle exception utilisée en lieu et place de EIOError : EprinterError.

Etude du source

Les constantes prn_LPT1, prn_LPT2 et prn_LPT3 assurent la concordance entre les numéros des imprimantes identifiées par le BIOS et l'application Delphi. Bien que ceux-ci puissent être employés directement dans les fonctions **prn_...** , je vous invite à utiliser les constantes pour deux raisons : la lisibilité et la compatibilité avec une évolution de l'unité (BIOS avec d'autres numéros, etc...).

Le tableau prn_MEMO a pour vocation de mémoriser les délais de time-out des différents ports (mémorisés à l'ouverture par **prn_open** et restitués en fermeture par **prn_close**).

Le tableau prn_TIME indique le délai de time-out souhaité lors de l'exploitation des ports. Il peut être nécessaire d'ajuster ceux-ci en fonction du temps de réponse de l'imprimante (évite d'avoir des messages d'avertissement alors que l'imprimante est prête). Dans ce cas il faut augmenter la valeur. La fourchette de valeur rencontrée est généralement de 2 à 4 pour les imprimantes particulièrement lentes. Vous pouvez modifier cette valeur soit dans l'unité, soit prévoir de lire celle-ci à partir d'un fichier ".ini"

```
var
Fini:TIniFile;
...
begin
Fini:=TIniFile.Create( "nom du fichier .ini" )
prn_TIME[prn_LPT1]:=FIni.ReadInteger("prn_TIME","LPT1",2);
prn_TIME[prn_LPT1]:=FIni.ReadInteger("prn_TIME","LPT1",2);
prn_TIME[prn_LPT1]:=FIni.ReadInteger("prn_TIME","LPT1",2);
Fini.Free;
end;
...
```

Si vous n'intégrez pas cette fonctionnalité à l'unité "imprimer", il faudra utiliser la procédure prn_timeout(prn_LPTx, Fini.ReadInteger("prn_TIME", "LPTx",2) car prn_TIME ne sera pas accessible. "prn_stt" renvoie l'état de l'imprimante : time-out, erreur de transmission, etc. En fait tout va bien si la condition suivante est remplie :

((res and $20)=0) and((res and $80)<>0) and ((res and $10)<>0))

Soit en clair : (il y a du papier) et (accusé réception) et (imprimante en ligne). A noter que cette expression peut être optimisée, mais son écriture sous cette forme est explicite *(devinette : comment simplifier le test ?... vous avez trouvé ? non ? réponse le mois prochain...)*.

"prn_open" initialise le port imprimante et met en place la nouvelle valeur de time-out après avoir préalablement sauvegardé la valeur courante.

"prn_close".....restitue la valeur initiale du time-out pour le port concerné.

"prn_chr".....émet un caractère via le port considéré.

"prn_write".....émet une chaîne de caractères en utilisant "prn_chr" et en contrôlant l'état du port. Après quatre essais infructueux et abandon de l'opérateur, déclenche l'exception EPrinterError.

"prn_writeln".....idem prn_write avec un CR/LF à la fin.

Remarques

La solution proposée est compatible avec l'emploi de pilotes résidents DOS utilisant l'interruption $17 pour gérer des fonctions spécifiques. Vous pouvez continuer d'utiliser la solution d'édition de codes à barres sur imprimantes matricielles "BARRES" de Worthington...

Pour en savoir plus ...

L'interruption n° $17 est abondamment documentée dans "La Bible PC programmation système" aux éditions Micro Application *(page 621 à 625 pour la 4ème édition).*