Tech Info Library

How to read Apple PILOT data files from Pascal

Revised: 11/30/84
Security: Everyone

How to read Apple PILOT data files from Pascal

==========

Though it may appear that data files are randomly accessible text files, they are actually block structured files, two strings long, with 255 per block. Records 0 and 1 are contained in relative block zero, 2 and 3 are in block 1, etc. The Pascal construct which defines this file format is:

```
VAR PILOTFILE: FILE OF STRING [255];
```

The data record associated with this file has the description PILOTFILE^, with a string of length 255. This file may be sequentially or randomly accessed using the commands Get, Put and Seek. See the Pascal Language Reference manual for details.

To open an existing file, use the Reset command; a new data file may be created using Rewrite. These commands are the same as the PILOT commands Fix and Fox respectively. Always Close a new file using the Lock option to record the data file in the diskette directory.

PASCALPGM Links to the Pascal program contained in the file PASCALPGM.code. Compile any intrinsic library units this program requires with the libraries found on your Apple PILOT Author Diskette or Lesson Diskette using the Pascal 1.0 compiler. The Apple PILOT system uses a special 48K run-only version of Pascal 1.0, which requires different intrinsics due to its unique storage allocation.

The linkage from PILOT to Pascal is one-way: data cannot be transmitted from PILOT to Pascal, except by data files on diskette. When the Pascal program ends, the PILOT lesson diskette reboots automatically and begins executing the Hello program if any. Programs running in this environment have approximately 10K less user storage space than in the standard Pascal system.

The first compiled procedure in the outer block of any Pascal program running under the PILOT system should be:

```
Procedure Syserror;
Begin
End;
```

System errors automatically divert control to this procedure, with IOResult set to the corresponding error number. The default action is:

A. If the error is non-fatal, execution of the procedure currently executed is aborted. Execution of its calling procedure is resumed.

B. When the error is fatal, the program terminates with an appropriate error message. Any additional error checking or recovery should be done in this procedure.

The following example prints the contents of any Apple PILOT data file:

```
Program Datalist;

Var PILOTFILE: file of string [255];
    Filename: string;
    RECNO: integer;

Procedure Syserror;
Begin
End;

Procedure Pause;
Begin
  Writeln;
  Write ('Press RETURN to continue...');
  Readln;
  Writeln
End;

  Begin
  Page (output);
  Write ('List which data file? ');
  Readln (Filename);
  Writeln;
  Filename := CONCAT (Filename,'.data');
  RESET (PILOTFILE, Filename);
  RECNO := 0;
  While not EOF (PILOTFILE) Do Begin
    Writeln ('Record ',RECNO,': ',PILOTFILE^);
    Get (PILOTFILE);
    RECNO := RECNO + 1;
    If RECNO MOD 20 = 0 Then PAUSE
  End;
  Writeln;
  Writeln ('End of file: ',RECNO,' records listed.');
  Pause
End.
```

Apple Tech Notes

Tech Info Library Article Number:600