Technical Note FL29
Problem with GetVInfo

The high-level call `GetVInfo` (and its low-level counterpart `PBGetVInfo`) may return inaccurate results for `freeBytes` when running HFS.

[Sep 01 1987]

GetVInfo

The high-level File Manager call `GetVInfo` returns the number of free bytes on a volume as one of its parameters. Since `GetVInfo` is really only glue that fills in a parameter block for you and then calls `PBGetVInfo`, the values returned from it are subject to the limitations (imposed for MFS) discussed in the File Manager chapter of *Inside Macintosh Volume IV* (p. 130): "**Warning**: `IOVNmAlBlks` and `ioVFrBlks`, which are actually unsigned integers, are clipped to 31744 (`$7C00`) regardless of the size of the volume." This will be fixed in future versions of the glue (newer than MPW 2.0.2), but for now, you need to call `PBHGetVInfo` yourself instead, as shown below.

The value that `GetVInfo` returns in `freeBytes` (`ioVFrBlks` * `ioVAlBlkSize`) will thus be **less than or equal to** the actual number of free bytes on a volume. This isn't catastrophic, but can be highly inconvenient if you really need to know how much free space is on a given volume.

Note:
`IOVNmAlBlks` returned from `PB[H]GetVInfo` does not reflect the actual total number of blocks on an HFS disk, but rather only the blocks that are "available" for program use (it doesn't count blocks used for catalog information).

Here are two functions (one in MPW Pascal, one in MPW C) that return the actual number of free bytes on a volume, regardless of File System (if MFS is running, the `PBHGetVInfo` call will actually generate a `PBGetVInfo` call which will return the proper values for MFS volumes):

In MPW Pascal:

```
FUNCTION FreeSpaceOnVol(vRef:Integer; VAR freeBytes:Longint): OSErr;
TYPE
    {we need this to convert an unsigned integer to an unsigned
    longint}
    TwoIntsMakesALong    = RECORD CASE Integer OF 1:    (long: LongInt);
    2: (ints: ARRAY [0..1] OF Integer);
            END; {TwoIntsMakesALong}
VAR
    HPB        : HParamBlockRec;
    convert    : TwoIntsMakesALong;
    err        : OSErr;

BEGIN {FreeSpaceOnVol}
    WITH HPB DO BEGIN {set up parameter block for the PBGetVInfo call}
    ioNamePtr := NIL; {we don't care about the name}
    ioVRefNum := vRef;    {this was passed in as a parameter}
    ioVolIndex := 0;       {use ioVRefNum only}
    END;                  {WITH}
    err := PBHGetVInfo(@HPB,false);
    FreeSpaceOnVol:= err; {return error from HGetVInfo}

{
This next section needs some explanation.  ioVFrBlk is an unsigned integer.
If we were to assign it(or coerce it) to a longint, it would get sign extended
by MPW Pascal and would thus be negative.  Since we don't want that, we use a
variant record that maps two integers into the space of one longint.  The high
word (convert.ints[0]) is cleared in the first line, then the low word is
assigned the value of HPB.ioVFrBlk.  The resulting longint (convert.long)
is now a correctly signed (positive) long integer representing the number
of free blocks.  NOTE:  this is only necessary if the number of free blocks
on a disk exceeds 32767 ($7FFF).
}
    IF err = 0 THEN BEGIN
        convert.ints[0] := 0;
        convert.ints[1] := HPB.ioVFrBlk;
            freeBytes:= convert.long * HPB.ioVAlBlkSiz;
    END ELSE BEGIN
        {PBHGetVInfo failed}
        freeBytes:= 0; {return this if the routine failed}
    END;
END;  {FreeSpaceOnVol}
```

In MPW C:

```
OSErr freeSpaceOnVol(vRef,pfreeBytes)
short int vRef;
unsigned long int *pfreeBytes; /* C does this correctly!! */

{    /* freeSpaceOnVol */
    HVolumeParam HPB;
    OSErr err;

    HPB.ioNamePtr = 0L;        /* we don't care about the name */
    HPB.ioVRefNum = vRef;     /* this was passed in as a parameter */
    HPB.ioVolIndex = 0;        /* use ioVRefNum only */
    err = PBHGetVInfo(&HPB,false);
    if (err == noErr)
        *pfreeBytes = (unsigned long int)HPB.ioVFrBlk * HPB.ioVAlBlkSiz;
    else
        *pfreeBytes = 0L;    /* return this if the routine failed */
    return(err);            /* function result */
}    /* freeSpaceOnVol */
```

Back to top

References

File Manager

Back to top

Downloadables

Acrobat version of this Note (48K) Download

Back to top