FS_CHDIR

Purpose

Change or verify the directory path for the requesting process

Calling Sequence

```
int far pascal FS_CHDIR(flag, pcdfsi, pcdfsd, pDir, iCurDirEnd)

unsigned short flag;
struct cdsfi far * pcdfsi;
struct cdfsd far * pcdfsd;
char far * pDir;
unsigned short iCurDirEnd;
```

Where

flag	indicates what action is to be taken on the directory.
flag == 0	indicates that an explicit directory-change request has been made.
flag == 1	indicates that the working directory needs to be verified.
flag == 2	indicates that this reference to a directory is being freed.

The *flag* passed to the FSD will have a valid value.

pcdfsi is a pointer to a file-system-independent working directory structure.

- For *flag* == 0, this pointer points to the previous current directory on the drive.
- For *flag* == 1, this pointer points to the most recent working directory on the drive. The cdi_curdir field contains the text of the directory that is to be verified.
- For *flag* == 2, this pointer is null.

The FSD must never modify the *cdfsi*. The OS/2 kernel handles all updates.

pcdfsd is a pointer to a file-system-dependent working directory structure.

This is a place for the FSD to store information about the working directory. For *flag* == 0 or 1, this is the information left there by the FSD. The FSD is expected to update this information if the directory exists. For *flag* == 2, this is the information left there by the FSD.

pDir is a pointer to directory text.

For *flag* == 0, this is the pointer to the directory. For *flag* == 1 or *flag* == 2, this pointer is null. The FSD does not need to verify this pointer.

iCurDirEnd is the index of the end of the current directory in *pDir*.

This is used to optimize FSD path processing. If *iCurDirEnd* == -1, there is no current directory relevant to the directory text, that is, a device. This parameter only has meaning for *flag* == 0.

Remarks

The FSD should cache no information when the directory is the root. Root directories are a special case. They always exist, and never need validation. The OS/2 kernel does not pass root directory requests to the FSD. An FSD is not allowed to cache any information in the cdfsd data structure for a root directory. Under normal conditions, the kernel does not save the CDS for a root directory and builds one from scratch when it is needed. (One exception is where a validate CDS fails, and the kernel sets it to the root, and zeroes out the cdfsd data structure. This CDS is saved and is cleaned up later.)

The following is information about the exact state of the cdfsi and cdfsd data structures passed to the FSD for each flag value and guidelines about what an FSD should do upon receiving an FS_CHDIR call:

```
IF (flag == 0)     /* Set new Current Directory */
   pcdfsi, pcdfsd = copy of CDS we're starting from; may be useful as
starting
                    point for verification.

           cdfsi contents:

              hVPB - handle of Volume Parameter Block mapped to this drive
              end - end of root portion of CurDir
              flags - various flags (indicating state of cdfsd)
              IsValid - cdfsd is unknown format (ignore contents)
                        IsValid == 0x80
              IsRoot - cdfsd is meaningless if CurDir = root (not kept)
                        IsRoot == 0x40
              IsCurrent - cdfsd is know format, but may not be current
(medium
                          may have been changed).
                          IsCurrent == 0x20
              text - Current Directory Text

   icurdir = if Current Directory is in the path of the new Current
Directory,
             this is the index to the end of the Current Directory. If not,
             this is -1 (Current Directory does not apply).
   pDir = path to verify as legal directory

THEN

   Validate path named in pDir.
       /* This means both that it exists AND that it is a directory.
pcdfsi,
          pcdfsd, icurdir give old CDS, which may allow optimization */

   IF (Validate succeeds)
      IF (pDir != ROOT)
         Store any cache information in area pointed to by pcdfsd.
      ELSE
         Do Nothing.
         /* Area pointed to by pcdfsd will be thrown away, so don't bother
            storing into it */
         Return success.
    ELSE
      Return failure.
      /* Kernel will create new CDS using pDir data and pcdfsd data. If the
         old CDS is valid, the kernel will take care of cleaning it up. The
         FSD must not edit any structure other than the *Pcdfsd area, with
         which it may do as it chooses. */
/* flag == 0 */
ELSE
IF (flag == 1)       /* Validate current CDS structure */

   pcdfsi = pointer to copy of cdfsi of interest.
   pcdfsd = pointer to copy of cdfsd. Flags in cdfsi indicate the state of
            this cdfsd. It may be: (1) completely invalid (unknown
            format), (2) known format, but non-current information,
            (3) completely valid, or (4) all zero (root).

THEN

    Validate that CDS still describes a legal directory (using cdi_text).

    IF (valid)
       Update cdfsd if necessary.
       Return success.
       /* kernel will copy cdfsd into real CDS */
    ELSE
       IF (cdi_isvalid)
          Release any resources associated with cdfsd.
          /* kernel will force Current Directory to root, and will zero out
             cdfsd in real CDS */
          Return failure.
    /* The FSD must not modify any structure other than the cdfsd pointed to
by
       pcdfsd.  */
ELSE
IF (flag == 2)   /* previous CDS no longer in use; being freed */

   pcdfsd = pointer to copy of cdfsd of CDS being freed.

THEN

   Release any resources associated with the CDS.
   /* For example, if cdfsd (where pcdfsd points) contains a pointer to
      some FSD private structure associated with the CDS, that structure
      should be freed. */

/* kernel will not retain the cdfsd */
```