#226: Moving Your Cat

Written by: John Harvey February 1989

This Technical Note clarifies the documentation in *Inside Macintosh* for `_PBCatMove` and provides a demonstration on how to use it.

The documentation for `_PBCatMove` is not particularly clear; *Inside Macintosh*, Volume IV-157, states:

> *"The name and directory ID of the directory to which the file or directory is to be moved are specified by ioNewName and ioNewDirID."*

To understand this explanation, imagine a situation where your program maintains a work file in the default directory. Since this work file can get really big, if the user asks to save it to the same disk as the default directory, you use `_PBCatMove` to move the file instead of copying it.

After the user dismisses `SFPutFile` by clicking on the Save button, you have the situation illustrated in Figure 1.

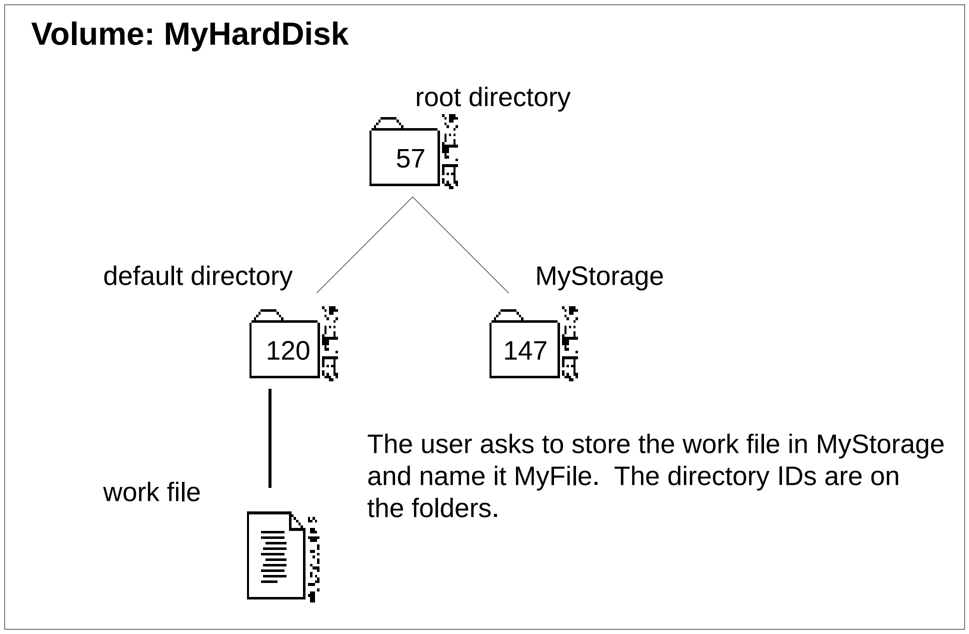
Volume: MyHardDisk
root directory
57
default directory
MyStorage
120
147
work file
The user asks to store the work file in MyStorage and name it MyFile. The directory IDs are on the folders.

Figure 1–Directory Structure After SFPutFile

Following the directions in *Inside Macintosh*, we put the directory ID (147) in `ioNewDirID` and a pointer to the directory name (the string "MyStorage") in `ioNewName`, but if we do this, we get a `fnfErr` (-43) since the File Manager looks for the directory inside of itself.

To understand this situation, consider how the File Manager accesses files or directories. It uses four different methods: full pathname, volume reference number and partial pathname, directory ID and partial pathname, and working directory reference number and partial pathname. Most File Manager calls let the programmer choose any of these methods, but `_PBCatMove` only lets you use the third method, directory ID and partial pathname, to access a file or directory. Figure 2 illustrates why a call to `_PBCatMove` fails if you use 147 as the directory ID.

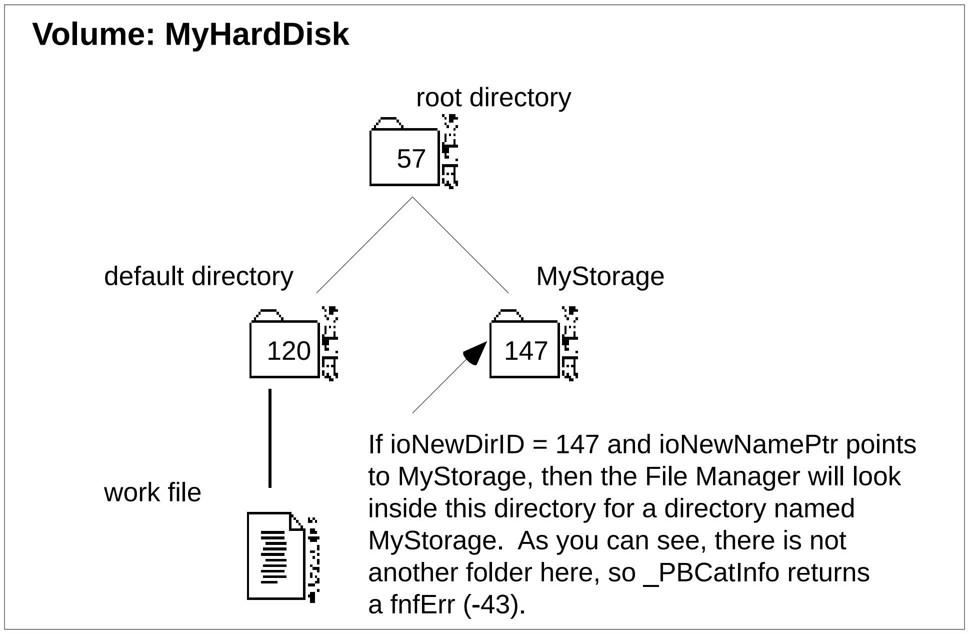
Volume: MyHardDisk
root directory
57
default directory
MyStorage
120
147
work file
If ioNewDirID = 147 and ioNewNamePtr points to MyStorage, then the File Manager will look inside this directory for a directory named MyStorage. As you can see, there is not another folder here, so _PBCatInfo returns a fnfErr (-43).

Figure 2–_PBCatMove Failure

What `_PBCatMove` really needs is the directory ID of the directory which **contains** the name of the directory to which `ioNewName` points. You also could call the value stored in `ioNewDirID` the parent directory of the destination directory. As Figure 3 illustrates, `_PBCatMove` finds what folder to look in from `ioNewDirID` and what to look for from `ioNewName`.

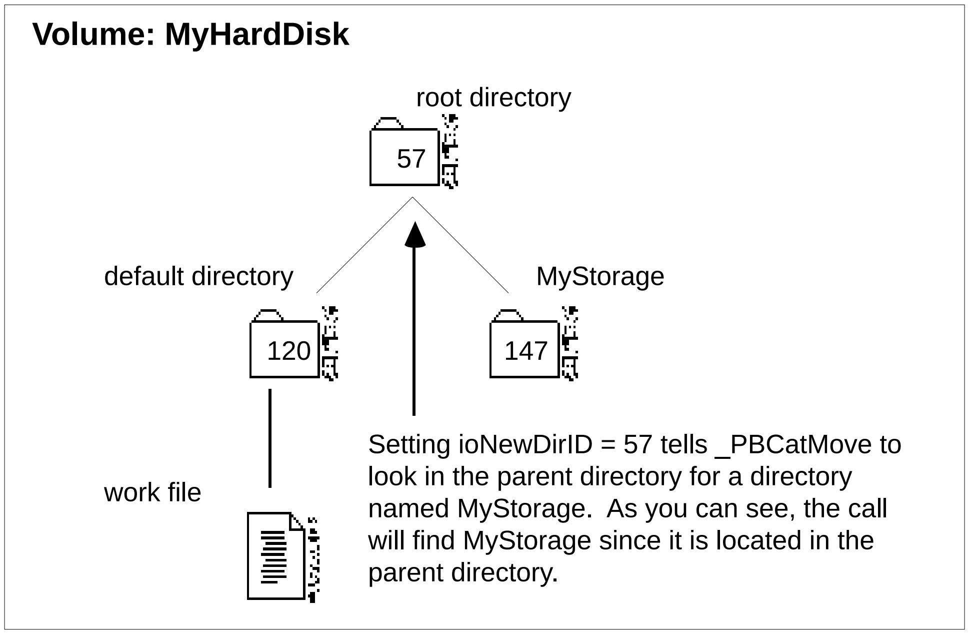
Volume: MyHardDisk
root directory
57
default directory
120
MyStorage
147
work file
Setting ioNewDirID = 57 tells _PBCatMove to look in the parent directory for a directory named MyStorage. As you can see, the call will find MyStorage since it is located in the parent directory.

Figure 3–Parent and Destination Directories

The line from *Inside Macintosh*, Volume IV-157, can be rephrased to say, “The name of the directory to which the file or directory is to be moved is specified by `ioNewName`. That directory’s **parent directory** is specified by `ioNewDirID`.”

Following are examples in Pascal and C which demonstrate how to obtain parent directory IDs and use `_PBCatMove` as discussed in this Technical Note:

Pascal:

```
PROCEDURE MoveTheCat(WorkFileName: Str255; WorkVRef: Integer;
           NewFileName: Str255; DestinationVRef: Integer);
{-------------------------------------------------------------------------------
|  MoveTheCat is called if a user asks to save a large file onto the same
|  volume that the work file is located on but a different directory. It uses
|  PBCatMove to move the file to the directory that the user chose from SFPutFile
|  and then renames that file to have the name the user specified in SFPutFile.
|
|  Parameters:
|  WorkFileName: is the name of the file that will be moving.  It has been
|  serving as our applications scratch file.
|
|  WorkVRef is the working directoryreference number for the directory
|  it is located in.  We found that out when we opened it.
|
|  NewFileName: is the file name supplied via SFPutFile.
|
|  DestinationVRef: the working directory reference number supplied by
|  SFPutFile.
--------------------------------------------------------------------------------}
```

```
VAR
   SourceDirectoryID: Longint;      {directory id that for WorkVRef}
   ParentDirectoryID: Longint;      {parent directory id of the destination
                                           directory}
   DestinationDirectoryName: Str255;{Name of the directory we are moving to}
   VolumeReference:      Integer;      {working directory reference for the volume}
   CInfoPB:              CInfoPBRec;   {parameter block used by PBCatInfo}
   CMovePB:              CMovePBRec;   {parameter block used by PBCatMove}
   PBRec:                ParamBlockRec;{parameter block used by PBRename}

BEGIN

   {get the volumes working directory reference with GetVol}
   Signal(GetVol(NIL,VolumeReference);  {see Technical Note #88 for a discussion of and
                                     an example of Signal error handling}

   WITH CInfoPB DO BEGIN
       ioNamePtr := NIL;            {name is ignored when ioFDirIndex = -1
                                     see Technical Note #69}
       ioVRefNum := WorkVRef;
       ioDrDirID := 0;
       ioFDirIndex := -1;           {we want to know about the parent directory
                                     Again Technical Note #69}
   END;

   Signal(PBGetCatInfo(@CInfoPB,FALSE);

   {if we get here we want to save the source directory}
   SourceDirectoryID := CInfoPB.ioDirID;

   {Now we need to get the parent directory id of the directory the file is
   moving to.  Do this by filling in CInfoPB and calling PBGetCatInfo once more}

   WITH CInfoPB DO BEGIN
   {PBCatInfo will supply the name of our Destination directory
   if we give it space to put it in.}
       ioNamePtr := @DestinationDirectoryName;
       ioVRefNum := DestinationVRef;{we want to find out about this directory's
                                     parent directory}
       ioDrDirID := 0;
       ioFDirIndex := -1;           {again Technical Note #69}
   END;

   Signal(PBGetCatInfo(@CInfoPB,FALSE));

   {save the parent directory id}
   ParentDirectoryID := CInfoPB.ioDrParID;

   {fill CMovePB with the information we got from the PBCatInfo calls}

   WITH CMovePB DO BEGIN
       ioNamePtr := @WorkFileName;
       ioDirID := SourceDirectoryID;
       ioNewName := @DestinationDirectoryName;     {filled in by last PBCatInfo}
       ioNewDirID := ParentDirectoryID;
   END;

   {Move that cat}
   Signal(PBCatMove(@CMovepb,FALSE);
    {After we moved the file we need to rename it. Do this by filling in
    PBRec and calling PBRename}
    WITH PBRec DO BEGIN
        ioNamePtr := @WorkFileName;      {what we've called it before}
        ioVRefNum := DestinationVRef;{Since GetFile was kind enough to open this
                                     working directory we will use it here}
        ioVersNum := 0;                 {always set this to 0}
        ioMisc := @NewFileName;         {The name the user chose}
    END;

    Signal(PBRename(@PBRec,FALSE));

END;    {of procedure}
```

C:

```
OSErr MoveTheCat(Str255 WorkFileName, short WorkVRef,
           Str255 NewFileName, short DestinationVRef)
/*--------------------------------------------------------------------------------
|  MoveTheCat is called if a user asks to save a large file onto the same
|  volume that the work file is located on but a different directory. It uses
|  PBCatMove to move the file to the directory that the user chose from SFPutFile
|  and then renames that file to have the name the user specified in SFPutFile.
|
|  Parameters:
|  WorkFileName: is the name of the file that will be moving.  It has been
|  serving as our applications scratch file.
|
|  WorkVRef is the working directoryreference number for the directory
|  it is located in.  We found that out when we opened it.
|
|  NewFileName: is the file name supplied via SFPutFile.
|
|  DestinationVRef: the working directory reference number supplied by
|  SFPutFile.
--------------------------------------------------------------------------------*/
{
    long SourceDirectoryID;                  /*directory id that for WorkVRef*/
    long ParentDirectoryID;                  /*parent directory id of the destination
                                       directory*/
    Str255 DestinationDirectoryName;  /*Name of the directory we are moving to*/
    short VolumeReference;            /*working directory reference for
                                       the volume*/
    CInfoPBRec CInfoPB;               /*parameter block used by PBCatInfo*/
    CMovePBRec CMovePB;               /*parameter block used by PBCatMove*/
    ParamBlockRec PBRec               /*parameter block used by PBRename*/
    OSErr err;                        /* storage for errors returned by
                                         file manager */

    /*get the volumes working directory reference with GetVol*/
    if (err = GetVol(NIL,VolumeReference))
        return(err);

        CInfoPB.ioNamePtr = NIL;             /*name is ignored when ioFDirIndex = -1
                                               see Technical Note #69*/
        CInfoPB.ioVRefNum = WorkVRef;
        CInfoPB.ioDrDirID = 0;
        CInfoPB.ioFDirIndex = -1;            /*we want to know about the parent directory
                                               Again Technical Note #69*/

    if (err = PBGetCatInfo(&CInfoPB,FALSE) )
        return(err);
    /*if we get here we want to save the source directory*/
    SourceDirectoryID = CInfoPB.ioDirID;

    /*Now we need to get the parent directory id of the directory the file is
      moving to.  Do this by filling in CInfoPB and calling PBGetCatInfo once more*/

    /*PBCatInfo will supply the name of our Destination directory
      if we give it space to put it in.*/
        CInfoPB.ioNamePtr = &DestinationDirectoryName;
        CInfoPB.ioVRefNum = DestinationVRef; /*we want to find out about this
                                               directory's parent directory*/
        CInfoPB.ioDrDirID = 0;
        CInfoPB.ioFDirIndex = -1;            /*again Technical Note #69*/

    if (err = PBGetCatInfo(&CInfoPB,FALSE))
        return(err);

    /*save the parent directory id*/
    ParentDirectoryID = CInfoPB.ioDrParID;

    /*fill in CMovePB with the information we got from the PBCatInfo calls*/

        CMovePB.ioNamePtr = &WorkFileName;
        CMovePB.ioDirID = SourceDirectoryID;
        CMovePB.ioNewName = &DestinationDirectoryName;     /*filled in by last PBCatInfo*/
        CMovePB.ioNewDirID = ParentDirectoryID;

    /*Move that cat*/
    if ( err = PBCatMove(&CMovepb,FALSE) )
        return(err);

    /*After we moved the file we need to rename it.  Do this by filling in
      PBRec and calling PBRename*/

        PBRec.ioNamePtr = &WorkFileName;     /*what we've called it before*/
        PBRec.ioVRefNum = DestinationVRef;   /*Since GetFile was kind enough
                                               to open this working directory we
                                               will use it here*/
        PBRec.ioVersNum = 0;                 /*always set this to 0*/
        PBRec.ioMisc = &NewFileName;         /*The name the user chose*/

    if ( err = PBRename(&PBRec,FALSE))
        return(err);

    return(noErr);                     /* everything was okay */

};  /*of function*/
```

Further Reference:

- *Inside Macintosh*, Volume IV-87, The File Manager
- Technical Note #69, Setting ioFDirIndex in _PBGetCatInfo Calls