PBUnmountVol Flush volume, close its files, release its memory

#include <Files.h> **File Manager (PBxxx)**

```
OSErr       PBUnmountVol(pb);
ParmBlkPtr  pb ;          address of a 64-byte VolumeParam structure
            returns       Error Code; 0=no error
```

PBUnmountVol flushes a volume buffer to disk and releases all memory occupied by the volume buffer and related structures. The volume must be re-mounted before it can be accessed.

pb is the address of a 64-byte VolumeParam structure or any of the variants which contain all the relevant fields:

Out-In	Name	Type	Size	Offset	Description
->	ioCompletion	ProcPtr	4	12	Completion routine address (if *async* =TRUE)
->	ioNamePtr	StringPtr	4	18	Address of volume name (or NIL)
->	ioVRefNum	short	2	22	Volume reference number of volume to unmount
<-	ioResult	OSErr	2	16	Error Code (0=no error, 1=not done yet)

Returns: an operating system Error Code. It will be one of:

noErr	(0)	No error
bdNamErr	(-37)	Bad name
fBsyErr	(-47)	Files are open on volume
extFSErr	(-58)	External file system
nsDrvErr	(-56)	No such drive
nsvErr	(-35)	No such volume
paramErr	(-50)	No default volume

Notes: Don't unmount the startup volume.

If you are using a string in the ioNamePtr field to specify a volume name, the string must be in the following form

```
myVolParam.ioNamePtr = "\pMy HardDisk:"
```

The trailing colon on the string indicates that we are referring to a directory, not a file.

All files on a volume must be closed in order for **PBUnmountVol** to succeed.
A fBsyErr will be received if this is not the case.

PBUnmountVol is typically called just before **PBEject** - when the disk will not be needed again for a while. Use **PBMountVol** if you need to remount the volume.

PBOffLine is related; the volume buffer is ditched but its control block remains in memory so that **GetVInfo** will continue to return information. Later, the volume can be brought back online transparently, by the File Manager, when needed.