**PBHRstFLock** Unlock a file (HFS only)

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

| | | |
|---|---|---|
| OSErr | **PBHRstFLock(***pb*, *async* **);** | |
| HParmBlkPtr | *pb* ; | address of an 80-byte HFileParam structure |
| Boolean | *async* ; | 0=await completion; 1=immediate return |
| | ***returns*** | Error Code; 0=no error |

**PBHRstFLock** unlocks a file. It undoes the effect of **PBSetFLock**, allowing the file to be deleted or modified. It is the same as **PBRstFLock** except it provides a way to specify a directory ID in the parameter block.

*pb* is the address of an 80-byte HFileParam structure (or a fileParam member of an HParamBlockRec union). The relevant fields are as follows:

| Out-In | Name | Type | Size | Offset | Description |
|---|---|---|---|---|---|
| –> | ioCompletion | ProcPtr | 4 | 12 | Completion routine address (if *async* =TRUE) |
| –> | ioNamePtr | StringPtr | 4 | 18 | Address of full or partial path/filename |
| –> | ioVRefNum | short | 2 | 22 | Volume, drive, or working directory reference |
| –> | ioDirID | long | 4 | 48 | ID of directory containing file to unlock |
| <- | ioResult | OSErr | 2 | 16 | Error Code (0=no error, 1=not done yet) |

*async* is a Boolean value. Use FALSE for normal (synchronous) operation or TRUE to enqueue the request and resume control immediately. See Async I/O.

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

| | | |
|---|---|---|
| noErr | (0) | No error |
| dirNFErr | (-120) | Directory not found |
| extFSErr | (-58) | External file system |
| fnfErr | (-43) | File not found |
| ioErr | (-36) | I/O error |
| nsvErr | (-35) | No such volume |
| vLckdErr | (-46) | Volume is locked |
| wPrErr | (-44) | Diskette is write-protected |

Notes: See **PBHSetFLock** for related information.

Be sure to call **PBFlushVol** to guarantee that the change is written to the disk.