FSRead Read from an open file or device driver

#include <Files.h> **Device Manager**

OSErr	**FSRead(***refNum*, *count*, *buffer* **)**;	
short	*refNum* ;	reference number as obtained from **OpenDriver**
long	***count** ;	bytes to read; receives actual bytes read
Ptr	*buffer* ;	address of buffer to receive data
	returns	Error Code; 0=no error

FSRead is used to read from files and from device drivers. The following description pertains to reading from files, but the two uses are conceptually similar. A description of how to use it with device drivers is also given below.

#include <Files.h> **File Manager**

OSErr	**FSRead(***fRefNum*, *inOutCount*, *buffer* **)**;	
short	*fRefNum* ;	reference number as obtained from **FSOpen**
long	***inOutCount* ;	bytes to read; receives actual bytes read
Ptr	*buffer* ;	address of buffer to receive data
	returns	Error Code; 0=no error

FSRead reads data from an open file into the caller's buffer. The read begins at the current position of the file mark and advances that mark by the actual number of bytes read.

fRefNum is the reference number of an open file. See **FSOpen** and **OpenRF**.

inOutCount is the address of a positive long integer. On entry, it specifies the number of bytes you wish to read. Upon return, it will contain the actual number of bytes read.

buffer is the address of a memory area, at least *inOutCount* bytes long. Upon return, the buffer will contain data read from the file.

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

noErr	(0)	No error
eofErr	(-39)	End of file (partial read occurred)
extFSErr	(-58)	External file system
fnOpnErr	(-38)	File not open
ioErr	(-36)	I/O error
paramErr	(-50)	*inOutCount* was negative
rfNumErr	(-51)	Bad file reference number

Notes: You can call **FSRead** repeatedly to read the file sequentially, or you can use **SetFPos** before the read to access any part of the file randomly.

In the event that the end-of-file is reached during the read, the return code will be eofErr and *inOutCount* value will contain the actual number of bytes read (probably less than the amount requested on entry).

Note: This is different from other operating systems that return an EOF error only when attempting to start the read at the EOF.

Use the low-level **PBRead** function for asynchronous reading or to use the 'newline' mode (to read from the current file position up to a carriage

return or other character).

The following example illustrates how to open a file, allocate a 24-byte relocatable data area to hold disk data, and read the 17th such data record from the file into that area. See **OpenRF** for an example that uses **FSRead** and **FSWrite** to copy an entire file. See **Using FSRead to Read from a File** for a longer example.

Example

```
#include <Files.h>
#include <Memory.h>
#define REC_SIZE 24

short   fRef, rc;
long    count;
Handle  hInBuf;

rc = FSOpen( "\pHardDisk:MyFile", 0, &fRef );
if ( rc ) { /* . . . handle the error . . . */ }

/* --- position to 17th 24-byte record in the file --- */
rc = SetFPos( fRef, fsFromStart, (REC_SIZE * 16) );
if ( rc ) { /* . . . handle the error . . . */ }

hInBuf = NewHandle( REC_SIZE );
if ( hInBuf == 0 ) { /* . . .  error allocating memory . . . */ }

count = REC_SIZE;
HLock( hInBuf );              /* better safe than sorry */
rc = FSRead( fRef, &count, *hInBuf );
if ( rc ) { /* . . . handle the error . . . */ }
HUnlock( hInBuf );

FSClose( fRef );
```

Here is the version of **FSRead** that is used for reading data from a device driver's i/o buffer:

FSRead reads data from an open device driver into the caller's buffer.

refNum is the reference number of an open device driver. See **OpenDriver**.

count is the address of a positive long integer. On entry, it specifies the number of bytes you wish to read. Upon return, it will contain the actual number of bytes read.

buffer is the address of a memory area, at least *count* bytes long. Upon return, the buffer will contain data read from the device driver.

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

noErr	(0)	No error
readErr	(-19)	No response from driver to Read call
badUnitErr	(-21)	*refNum* doesn't match unit table
unitEmptyErr	(-22)	*refNum* specifies NIL handle in unit table
notOpenErr	(-28)	Driver not open

Notes: This procedure, along with **FSWrite**, is also used by the **File Manager** to read and write files.