



This is part of **Family API** which allow to create dual-os version of program runs under OS/2 and DOS

Note: This is legacy API call. It is recommended to use 32-bit equivalent

2021/09/17 04:47 · prokushhev · [0 Comments](#)

2021/08/20 03:18 · prokushhev · [0 Comments](#)

DosWrite

This call transfers the specified number of bytes from a buffer to the specified file, synchronously with respect to the requesting process's execution.

Syntax

DosWrite (FileHandle, BufferArea, BufferLength, BytesWritten)

Parameters

;FileHandle (HFILE) - input : File handle from DosOpen. ;BufferArea (PVOID) - input : Address of the output buffer. ;BufferLength (USHORT) - input : Number of bytes to write. ;BytesWritten (PUSHORT) - output : Address of the number of bytes written.

Return Code

rc (USHORT) - return Return code descriptions are: * 0 NO_ERROR * 5 ERROR_ACCESS_DENIED * 6 ERROR_INVALID_HANDLE * 26 ERROR_NOT_DOS_DISK * 33 ERROR_LOCK_VIOLATION * 109 ERROR_BROKEN_PIPE

Remarks

On output, BytesWritten is the number of bytes actually written. If BytesWritten is different from BufferLength, this usually indicates insufficient disk space.

A BufferLength value of 0 is not considered an error. No data transfer occurs. There is no effect on the file or the file pointer.

Buffers that are multiples of the hardware's base physical unit for data written to the file on these base boundaries, are written directly to the device. (The base physical unit is defined as the smallest block that can be physically written to the device.) Other buffer sizes force some I/O to go through an internal system buffer and greatly reduce the efficiency of I/O operation.

The file pointer is moved by read and write operations. It can be moved to a desired position by calling DosChgFilePtr.

If the file is read-only, the write to the file is not performed.

Family API Considerations

Some options operate differently in the DOS mode than in OS/2 mode. Therefore, the following restriction applies to DosWrite when coding for the DOS mode.

* Only single-byte DosWrite requests can be made to the COM device, because the COM device driver for the DOS environment does not support multiple-byte I/O.

Named Pipe Considerations

DosWrite is also used to write bytes or messages to a named pipe.

Each write to a message pipe writes a message whose size is the length of the write; DosWrite automatically encodes message lengths in the pipe, so applications need not encode this information in the buffer being written.

Writes in blocking mode always write all requested bytes before returning. In non-blocking mode, if the message size is bigger than the buffer size, the write blocks. If the message size is smaller than the pipe but not enough space is left in the pipe, the write returns immediately with a value of zero, indicating no bytes were written.

In the case of a byte pipe, if the number of bytes to be written exceeds the space available in the pipe, DosWrite writes as many bytes as it can and returns with the number of bytes actually written.

An attempt to write to a pipe whose other end has been closed returns ERROR_BROKEN_PIPE.

Example Code

C Binding

```
<PRE> #define INCL_DOSFILEMGR

USHORT rc = DosWrite(FileHandle, BufferArea, BufferLength, BytesWritten);

HFILE FileHandle; /* File handle */ PVOID BufferArea; /* User buffer */ USHORT BufferLength; /* Buffer
length */ PUSHORT BytesWritten; /* Bytes written (returned) */

USHORT rc; /* return code */ </PRE> This example writes to a file. <PRE> #define INCL_DOSFILEMGR

#define OPEN_FILE 0x01 #define CREATE_FILE 0x10 #define FILE_ARCHIVE 0x20 #define FILE_EXISTS
OPEN_FILE #define FILE_NOEXISTS CREATE_FILE #define DASD_FLAG 0 #define INHERIT 0x80
#define WRITE_THRU 0 #define FAIL_FLAG 0 #define SHARE_FLAG 0x10 #define ACCESS_FLAG 0x02
```

```
#define FILE_NAME "test.dat" #define FILE_SIZE 800L #define FILE_ATTRIBUTE FILE_ARCHIVE #define  
RESERVED 0L
```

```
HFILE FileHandle; USHORT Wrote; USHORT Action; PSZ FileData[100]; USHORT rc;
```

```
Action = 2;  
strcpy(FileData, "Data...");  
if(!DosOpen(FILE_NAME, /* File path name */  
            &FileHandle, /* File handle */  
            &Action, /* Action taken */  
            FILE_SIZE, /* File primary allocation */  
            FILE_ATTRIBUTE, /* File attribute */  
            FILE_EXISTS | FILE_NOEXIST, /* Open function type */  
            DASD_FLAG | INHERIT | /* Open mode of the file */  
            WRITE_THRU | FAIL_FLAG |  
            SHARE_FLAG | ACCESS_FLAG,  
            RESERVED)) /* Reserved (must be zero) */  
rc = DosWrite(FileHandle, /* File handle */  
              (PVOID) FileData, /* User buffer */  
              sizeof(FileData), /* Buffer length */  
              &Wrote); /* Bytes written */
```

```
</PRE>
```

MASM Binding

```
<PRE> EXTRN DosWrite:FAR INCL_DOSFILEMGR EQU 1
```

```
PUSH WORD FileHandle ;File handle PUSH@ OTHER BufferArea ;User buffer PUSH WORD BufferLength  
;Buffer length PUSH@ WORD BytesWritten ;Bytes written (returned) CALL DosWrite
```

```
Returns WORD </PRE>
```

Note

Text based on [http://www.edm2.com/index.php/DosWrite_\(FAPI\)](http://www.edm2.com/index.php/DosWrite_(FAPI))

Family API	
DOS	Process Manager DosBeep DosExit DosSleep DosExecPgm
	File Manager DosChDir DosChgFilePtr DosClose DosDelete DosDupHandle DosMkDir DosMove DosQCurDir DosQCurDisk DosSet FileMode DosOpen DosQFileInfo DosRead DosQ FileMode DosQFSInfo DosQVerify DosRmDir DosSelectDisk DosFindClose DosFindFirst DosFindNext DosSet FileInfo DosSet Verify DosWrite DosFileLocks DosSet FHandState DosNewSize DosBufReset DosQFHandState DosSet FInfo
	Memory Manager DosFreeSeg DosSubAlloc DosSubFree DosSubSet DosAlloc Huge DosAlloc Seg DosRealloc Huge DosRealloc Seg DosGet Huge Shift DosCreate CS Alias
	NLS DosCaseMap DosGet Ctry Info DosGet DBCSEv DosSet Ctry Code DosGet Collate DosGet Message DosIns Message DosPut Message
	Date and Time DosSet Date Time DosGet Date Time
	Devices DosDevConfig DosDevIOCtl DosDevIOCtl2
	Signals DosHoldSignal DosSet Sig Handler
	Misc BadDynLink DosGet Env DosGet Machine Mode DosGet Version DosError DosErr Class DosSet Vec
KBD	KbdCharIn KbdFlushBuffer KbdGet Status KbdSet Status KbdStringIn KbdPeek
VIO	VioGet Buf VioGet Config VioGet Cur Pos VioGet Cur Type VioGet Phys Buf VioRead Cell Str VioRead Char Str VioScroll Up VioScroll Dn VioScroll If VioScroll Rt VioScr Un Lock VioSet Cur Pos VioSet Cur Type VioSet Mode VioGet Mode VioShow Buf VioWrt Cell Str VioWrt Char Str VioWrt Char Str Att VioWrt N Attr VioWrt N Cell VioWrt N Char VioWrt TTY VioScr Lock VioPop Up
Tools	BIND
Modules	DOSCALLS.DLL VIOCALS.DLL KBDCALS.DLL MSG.DLL
Libraries	API.LIB OS2386.LIB FAPI.LIB DOSCALLS.LIB SUBCALLS.LIB

2018/08/25 15:05 · prokushev · 0 Comments

From:
<https://ftp.osfree.org/doku/> - osFree wiki



Permanent link:
<https://ftp.osfree.org/doku/doku.php?id=en:docs:fapi:doswrite&rev=1629440559>

Last update: **2021/08/20 06:22**