GetFileSize function

Retrieves the size of the specified file, in bytes.

It is recommended that you use **GetFileSizeEx**.

Syntax

C++

```
DWORD WINAPI GetFileSize(
  _In_      HANDLE  hFile,
  _Out_opt_ LPDWORD lpFileSizeHigh
);
```

Parameters

hFile [in]
A handle to the file.

lpFileSizeHigh [out, optional]
A pointer to the variable where the high-order doubleword of the file size is returned. This parameter can be **NULL** if the application does not require the high-order doubleword.

Return value

If the function succeeds, the return value is the low-order doubleword of the file size, and, if *lpFileSizeHigh* is non-**NULL**, the function puts the high-order doubleword of the file size into the variable pointed to by that parameter.

If the function fails and *lpFileSizeHigh* is **NULL**, the return value is **INVALID_FILE_SIZE**. To get extended error information, call **GetLastError**. When *lpFileSizeHigh* is **NULL**, the results returned for large files are ambiguous, and you will not be able to determine the actual size of the file. It is recommended that you use **GetFileSizeEx** instead.

If the function fails and *lpFileSizeHigh* is non-**NULL**, the return value is **INVALID_FILE_SIZE** and **GetLastError** will return a value other than **NO_ERROR**.

Remarks

You cannot use the **GetFileSize** function with a handle of a nonseeking device such as a pipe or a communications device. To determine the file type for *hFile*, use the GetFileType function.

The **GetFileSize** function retrieves the uncompressed size of a file. Use the GetCompressedFileSize function to obtain the compressed size of a file.

Note that if the return value is **INVALID_FILE_SIZE** (0xffffffff), an application must call GetLastError to determine whether the function has succeeded or failed. The reason the function may appear to fail when it has not is that *lpFileSizeHigh* could be non-**NULL** or the file size could be 0xffffffff. In this case, **GetLastError** will return **NO_ERROR** (0) upon success. Because of this behavior, it is recommended that you use GetFileSizeEx instead.

Transacted Operations: If there is a transaction bound to the file handle, then the function returns information for the isolated file view.

In Windows 8 and Windows Server 2012, this function is supported by the following technologies.

Technology	Supported
Server Message Block (SMB) 3.0 protocol	Yes
SMB 3.0 Transparent Failover (TFO)	Yes
SMB 3.0 with Scale-out File Shares (SO)	Yes
Cluster Shared Volume File System (CsvFS)	Yes
Resilient File System (ReFS)	Yes

Examples

For an example, see Creating a View Within a File.

Requirements

Minimum supported client	Windows XP [desktop apps only]
Minimum supported server	Windows Server 2003 [desktop apps only]

Header	FileAPI.h (include Windows.h); WinBase.h on Windows Server 2008 R2, Windows 7, Windows Server 2008, Windows Vista, Windows Server 2003, and Windows XP (include Windows.h)
Library	Kernel32.lib
DLL	Kernel32.dll

See also

File Management Functions
GetCompressedFileSize
GetFileSizeEx
GetFileType