GetQueuedCompletionStatus function

Attempts to dequeue an I/O completion packet from the specified I/O completion port. If there is no completion packet queued, the function waits for a pending I/O operation associated with the completion port to complete.

To dequeue multiple I/O completion packets at once, use the **GetQueuedCompletionStatusEx** function.

Syntax

C++

```
BOOL WINAPI GetQueuedCompletionStatus(
  _In_  HANDLE       CompletionPort,
  _Out_ LPDWORD      lpNumberOfBytes,
  _Out_ PULONG_PTR   lpCompletionKey,
  _Out_ LPOVERLAPPED *lpOverlapped,
  _In_  DWORD        dwMilliseconds
);
```

Parameters

CompletionPort [in]
A handle to the completion port. To create a completion port, use the **CreateIoCompletionPort** function.

lpNumberOfBytes [out]
A pointer to a variable that receives the number of bytes transferred during an I/O operation that has completed.

lpCompletionKey [out]
A pointer to a variable that receives the completion key value associated with the file handle whose I/O operation has completed. A completion key is a per-file key that is specified in a call to **CreateIoCompletionPort**.

lpOverlapped [out]
A pointer to a variable that receives the address of the **OVERLAPPED** structure that was specified when the completed I/O operation was started.

Even if you have passed the function a file handle associated with a completion port and a valid **OVERLAPPED** structure, an application can prevent completion port notification. This is done by specifying a valid event handle for the **hEvent** member of the **OVERLAPPED** structure, and setting its

low-order bit. A valid event handle whose low-order bit is set keeps I/O completion from being queued to the completion port.

dwMilliseconds [in]
The number of milliseconds that the caller is willing to wait for a completion packet to appear at the completion port. If a completion packet does not appear within the specified time, the function times out, returns **FALSE**, and sets **lpOverlapped* to **NULL**.

If *dwMilliseconds* is **INFINITE**, the function will never time out. If *dwMilliseconds* is zero and there is no I/O operation to dequeue, the function will time out immediately.

Return value

Returns nonzero (**TRUE**) if successful or zero (**FALSE**) otherwise.

To get extended error information, call **GetLastError**.

For more information, see the Remarks section.

Remarks

This function associates a thread with the specified completion port. A thread can be associated with at most one completion port.

If a call to **GetQueuedCompletionStatus** fails because the completion port handle associated with it is closed while the call is outstanding, the function returns **FALSE**, **lpOverlapped* will be **NULL**, and **GetLastError** will return **ERROR_ABANDONED_WAIT_0**.

Windows Server 2003 and Windows XP: Closing the completion port handle while a call is outstanding will not result in the previously stated behavior. The function will continue to wait until an entry is removed from the port or until a time-out occurs, if specified as a value other than **INFINITE**.

If the**GetQueuedCompletionStatus** function succeeds, it dequeued a completion packet for a successful I/O operation from the completion port and has stored information in the variables pointed to by the following parameters: *lpNumberOfBytes*, *lpCompletionKey*, and *lpOverlapped*. Upon failure (the return value is **FALSE**), those same parameters can contain particular value combinations as follows:

- If **lpOverlapped* is **NULL**, the function did not dequeue a completion packet from the completion port. In this case, the function does not store information in the variables pointed to by the *lpNumberOfBytes* and *lpCompletionKey* parameters, and their values are indeterminate.
- If **lpOverlapped* is not **NULL** and the function dequeues a completion packet for a failed I/O operation from the completion port, the function stores information about the failed operation in the variables pointed to by *lpNumberOfBytes*, *lpCompletionKey*, and *lpOverlapped*. To get extended error information, call **GetLastError**.

For more information on I/O completion port theory, usage, and associated functions, see **I/O Completion Ports**.

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

Requirements

Minimum supported client	Windows XP [desktop apps only]
Minimum supported server	Windows Server 2003 [desktop apps only]
Header	WinBase.h (include Windows.h)
Library	Kernel32.lib
DLL	Kernel32.dll

See also

Overview Topics
File Management Functions
I/O Completion Ports
Using the Windows Headers
Functions
ConnectNamedPipe
CreateIoCompletionPort
DeviceIoControl

GetQueuedCompletionStatusEx
LockFileEx
ReadFile
PostQueuedCompletionStatus
TransactNamedPipe
WaitCommEvent
WriteFile