



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](#)

MouReadEventQue

This call reads an event from the mouse device FIFO event queue, and places it in a structure provided by the application.

Syntax

```
MouReadEventQue (Buffer, ReadType, DeviceHandle)
```

Parameters

- Buffer ([PMOUSEVENTINFO](#)) - output: Address of the status of the mouse event queue.
- DeviceHandle ([HMOU](#)) - input : Handle of the mouse device from a previous MouOpen.

Return Code

rc ([USHORT](#)) - return:Return code descriptions are:

- 0 NO_ERROR
- 385 ERROR_MOUSE_NO_DEVICE
- 387 ERROR_MOUSE_INV_PARMS
- 393 ERROR_MOUSE_NO_DATA
- 466 ERROR_MOU_DETACHED
- 501 ERROR_MOUSE_NO_CONSOLE
- 505 ERROR_MOU_EXTENDED_SG

Remarks

The types of queued events are directly affected by the current value of the Mouse EventMask. MouSetEventMask is used to indicate the types of events desired, and MouGetEventMask is used to query the current value of the mask. Refer to these functions for further explanation of the masking of events.

Recognition of the mouse transition depends on the use of MouState returned in the event record. The application should focus on bit transitions that occur in this word. It is important to properly set the event mask with MouSetEventMask for reporting the state transitions.

MouState reports the state of the mouse that resulted from the action that caused the event. The action can be pressing or releasing a button, and/or moving the mouse. All status is given, regardless of the EventMask that was used to determine whether or not to report the event.

For example, assume the EventMask indicates that the application wishes only button 1 event. The EventMask has only bits 1 and 2 set in this case. Also assume the current state of the mouse is no buttons down, and mouse is not moving. At this point, button 1 is pressed causing an event; the status shows button 1 down (bit 2 set). Next the mouse is moved, thereby causing more events; status shows bit 1 set. Finally, mouse is stopped and button 1 is released. The event shows status with no bits set.

Next, button 2 is pressed. No event occurs. Mouse is then moved; again, no event. Then, while mouse is still in motion, button 1 is pressed; an event is generated with bits 1 and 3 set in the state word. While mouse is still in motion, both buttons are released. Because button 1 changes states, an event occurs. The state word has bit 0 set. Finally, mouse is stopped. No event occurs, again because no button 1 transition has taken place.

The Row and Column fields in the Buffer Parameter may contain either absolute display coordinates or relative mouse motion in mickeys. See [MouSetDevStatus](#) for additional information.

Bindings

C

```
typedef struct _MOUEVENTINFO { /* mouev */
    USHORT fs;                      /* State of mouse at time event was reported */
    ULONG  time;                    /* Time since boot in milliseconds */
    USHORT row;                     /* Absolute/relative row position */
    USHORT col;                     /* Absolute/relative column position */
}MOUEVENTINFO;

#define INCL_MOU

USHORT  rc = MouReadEventQue(Buffer, ReadType, DeviceHandle);

PMOUEVENTINFO  Buffer;          /* 10 byte Structure address */
PUSHORT        ReadType;        /* Read type */
HMOU           DeviceHandle;   /* Mouse device handle */

USHORT         rc;              /* return code */
```

MASM

```

MOUEVENTINFO struct
    mouev_fs    dw    ? ;State of mouse at time event was reported
    mouev_time  dd    ? ;time since boot in milliseconds
    mouev_row   dw    ? ;absolute/relative row position
    mouev_col   dw    ? ;absolute/relative column position
MOUEVENTINFO ends

EXTRN MouReadEventQue:FAR
INCL_MOU      EQU 1

PUSH@ OTHER    Buffer        ;10 byte Structure address
PUSH@ WORD     ReadType     ;Read type
PUSH WORD     DeviceHandle  ;Mouse device handle
CALL  MouReadEventQue

```

Returns WORD

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 FHand State DosNewSize DosBufReset DosQFHand State DosSet FSinfo DosShutdown
	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 Vio Scroll Up Vio Scroll Dn Vio Scroll If Vio Scroll Rt Vio Scr Un Lock Vio Set Cur Pos Vio Set Cur Type Vio Set Mode Vio Get Mode Vio Show Buf Vio Wrt Cell Str Vio Wrt Char Str Vio Wrt Char Str Att Vio Wrt N Attr Vio Wrt N Cell Vio Wrt N Char Vio Wrt TTY Vio Scr Lock Vio Pop Up
Tools		BIND
Modules		DOSCALLS.DLL VIOCALS.DLL KBDCALLS.DLL MSG.DLL
Libraries		API.LIB OS2386.LIB FAPI.LIB DOSCALLS.LIB SUBCALLS.LIB

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

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



Permanent link:
<https://cocorico.osfree.org/doku/doku.php?id=en:docs:fapi:moureadeventque>

Last update: **2021/11/04 13:14**