 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 · prokushev · 0 Comments
2021/08/20 03:18 · prokushev · 0 Comments

MouRegister

This call registers a mouse subsystem within a session.

Syntax

```
MouRegister (ModuleName, EntryName, Mask)
```

Parameters

- ModuleName (PSZ) - input : Address of the dynamic link module name. The maximum length is 9 bytes (including ASCIIZ terminator).
- EntryName (PSZ) - input : Address of the dynamic link entry point name of a routine that receives control when any of the registered functions are called. The maximum length is 33 bytes (including ASCIIZ terminator).
- Mask (ULONG) - input : A mask of bits, where each bit set to 1 identifies a mouse function being registered.

Return Code

rc (USHORT) - return:Return code descriptions are:

- 0 NO_ERROR
- 385 ERROR_MOUSE_NO_DEVICE
- 413 ERROR_MOUSE_INVALID_ASCIIZ
- 414 ERROR_MOUSE_INVALID_MASK
- 415 ERROR_MOUSE_REGISTER
- 466 ERROR_MOU_DETACHED
- 505 ERROR_MOU_EXTENDED_SG

Remarks

The Base Mouse Subsystem is the default mouse subsystem. There can be only one MouRegister

outstanding for each session without an intervening MouDeRegister. MouDeRegister must be issued by the same process that issued MouRegister.

When any registered function is called, control is routed to EntryName. When this routine is entered, four additional values are pushed onto the stack. The first is the index number (Word) of the function being called. The second is a near pointer (Word). The third is the caller's DS register (Word). The fourth is the return address (DWord) to the mouse router. For example, if MouGetNumMickeys were called and control routed to EntryName, the stack would appear as if the following instructions were executed:

```
PUSH@ WORD    NumberOfMickeys
PUSH  WORD    DeviceHandle
CALL  FAR     MouGetNumMickeys
PUSH  WORD    Function Code
CALL  NEAR    Entry point in Mouse Router
PUSH  DS
CALL  FAR     EntryName.
```

When a registered function returns to the Mouse Router, AX is interpreted as follows:

- AX = 0 No error. Do not invoke the Base Mouse Subsystem routine. Return AX = 0.
- AX = -1 Invoke the BaseMouse Subsystem routine. Return AX = return code from the Base Mouse Subsystem.
- AX = error (if not 0 or -1) Do not invoke the Base Mouse Subsystem Routine. Return AX = error.

When the mouse router receives a mouse call, it routes it to the Base Mouse Subsystem unless an application or other mouse subsystem has previously issued MouRegister for that call. If the call was registered, the subsystem is entered at the EntryName specified, and provided with the applicable function code.

The registered function mask is used to determine whether a requested function is performed by the registered mouse subsystem or default to the Base Mouse Subsystem.

The following list shows the relationship of the mouse API calls and the Function Code passed to either the Base Mouse Subsystem or a registered mouse subsystem.

MOU API calls	Function Code	Function Mask
MouGetNumButtons	00H	00000001H
MouGetNumMickeys	01H	00000002H
MouGetDevStatus	02H	00000004H
MouGetNumQueEl	03H	00000008H
MouReadEventQue	04H	00000010H
MouGetScaleFact	05H	00000020H
MouGetEventMask	06H	00000040H
MouSetScaleFact	07H	00000080H
MouSetEventMask	08H	00000100H
MouGetHotKey	09H	00000200H
MouSetHotKey	0AH	00000400H
MouOpen	0BH	00000800H
MouClose	0CH	00001000H

MOU API calls	Function Code	Function Mask
MouGetPtrShape	0DH	00002000H
MouSetPtrShape	0EH	00004000H
MouDrawPtr	0FH	00008000H
MouRemovePtr	10H	00010000H
MouGetPtrPos	11H	00020000H
MouSetPtrPos	12H	00040000H
MouInitReal	13H	00080000H
MouFlushQue	14H	00100000H
MouSetDevStatus	15H	00200000H

A registered mouse sybsystem must leave the stack, on exit, in the exact state it was received.

Bindings

C

```
#define INCL_MOU

USHORT  rc = MouRegister(ModuleName, EntryName, Mask);

PSZ     ModuleName;     /* Module Name */
PSZ     EntryName;      /* Entry Name */
ULONG   Mask;           /* Function Mask */

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

MASM

```
EXTRN   MouRegister:FAR
INCL_MOU            EQU 1

PUSH@  ASCIIZ  ModuleName    ;Module Name
PUSH@  ASCIIZ  EntryName     ;Entry Name
PUSH   DWORD   Mask          ;Function Mask
CALL   MouRegister

Returns WORD
```

Related Functions

MouDeRegister

Family API		
DOS	Process Manager	DosBeep DosExit DosSleep DosExecPgm
	File Manager	DosChDir DosChgFilePtr DosClose DosDelete DosDupHandle DosMkDir DosMove DosQCurDir DosQCurDisk DosSetFileMode DosOpen DosQFileInfo DosRead DosQFileMode DosQFSInfo DosQVerify DosRmDir DosSelectDisk DosFindClose DosFindFirst DosFindNext DosSetFileInfo DosSetVerify DosWrite DosFileLocks DosSetFHandState DosNewSize DosBufReset DosQFHandState DosSetFSinfo DosShutdown
	Memory Manager	DosFreeSeg DosSubAlloc DosSubFree DosSubSet DosAllocHuge DosAllocSeg DosReallocHuge DosReallocSeg DosGetHugeShift DosCreateCSAlias
	NLS	DosCaseMap DosGetCtryInfo DosGetDBCSEv DosSetCtryCode DosGetCollate DosGetMessage DosInsMessage DosPutMessage
	Date and Time	DosSetDateTime DosGetDateTime
	Devices	DosDevConfig DosDevIOCtl DosDevIOCtl2
	Signals	DosHoldSignal DosSetSigHandler
	Misc	BadDynLink DosGetEnv DosGetMachineMode DosGetVersion DosError DosErrClass DosSetVec
KBD		KbdCharIn KbdFlushBuffer KbdGetStatus KbdSetStatus KbdStringIn KbdPeek
VIO		VioGetBuf VioGetConfig VioGetCurPos VioGetCurType VioGetPhysBuf VioReadCellStr VioReadCharStr VioScrollUp VioScrollDn VioScrollLf VioScrollRt VioScrUnLock VioSetCurPos VioSetCurType VioSetMode VioGetMode VioShowBuf VioWrtCellStr VioWrtCharStr VioWrtCharStrAtt VioWrtNAttr VioWrtNCell VioWrtNChar VioWrtTTY VioScrLock VioPopUp
Tools		BIND
Modules		DOSCALLS.DLL VIOCALLS.DLL KBDCALLS.DLL MSG.DLL
Libraries		API.LIB OS2386.LIB FAPI.LIB DOSCALLS.LIB SUBCALLS.LIB

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