2025/10/16 00:50 1/3 KbdPeek

KbdPeek

Bindings:

C:

```
typedef struct _KBDKEYINFO { /* kbci */
         chChar;
 UCHAR
                             /* ASCII character code */
                             /* Scan Code */
 UCHAR
          chScan;
 UCHAR fbStatus;
UCHAR bNlsShift;
                             /* State of the character */
                            /* Reserved (set to zero) */
                             /* State of the shift keys */
 USHORT fsState;
                             /* Time stamp of keystroke (ms since ipl) */
 ULONG
         time:
}KBDKEYINF0;
#define INCL KBD
USHORT rc = KbdPeek(CharData, KbdHandle);
PKBDKEYINFO
                              /* Buffer for data */
                CharData:
HKBD
                KbdHandle:
                               /* Keyboard handle */
USHORT
                               /* return code */068∏068∏
                rc;
```

MASM:

```
KBDKEYINFO struc
 kbci chChar
               db ? ; ASCII character code
 kbci chScan
                db ? ;Scan Code
 kbci fbStatus db ? ;State of the character
 kbci bNlsShift db ? ; Reserved (set to zero)
 kbci_fsState
                dw ? ; state of the shift keys
 kbci time
                dd ? ;time stamp of keystroke (ms since ipl)
KBDKEYINFO ends
EXTRN KbdPeek: FAR
                   E0U 1
INCL KBD
              CharData
                            :Buffer for data
PUSH@ OTHER
PUSH
      WORD
              KbdHandle
                            ;Keyboard handle
CALL
      KbdPeek
Returns WORD
```

This call returns any available character data record from the keyboard without removing it from the buffer.

KbdPeek (CharData, KbdHandle)

CharData (**PKBDKEYINFO**) - output Address of the character data information:

asciicharcode (**UCHAR**) ASCII character code. The scan code received from the keyboard is translated to the ASCII character code.

scancode (UCHAR) Code received from the keyboard hardware.

status (**UCHAR**) State of the keystroke event:

Bit	Description	
7-6	00	= Undefined.
	01	= Final character, interim character flag off.
	10	= Interim character.
	11	= Final character, interim character flag on.
5	1	= Immediate conversion requested.
4-2		Reserved, set to zero.
1	0	= Scan code is a character.
	1	= Scan code is not a character; it is an extended key code from the keyboard.
0	1	= Shift status returned without character.

reserved (UCHAR) NLS shift status. Reserved, set to zero.

shiftkeystat (USHORT) Shift key status.

Description
Description
SysReq key down
CapsLock key down
NumLock key down
ScrollLock key down
Right Alt key down
Right Ctrl key down
Left Alt key down
Left Ctrl key down
Insert on
CapsLock on
NumLock on
ScrollLock on
Either Alt key down
Either Ctrl key down
Left Shift key down
Right Shift key down

time (**ULONG**) Time stamp indicating when a key was pressed. It is specified in milliseconds from the time the system was started.

KbdHandle (HKBD) - input Default keyboard or the logical keyboard.

rc (USHORT) - return Return code descriptions are:

0	NO_ERROR
439	ERROR KBD INVALID HANDLE

https://ftp.osfree.org/doku/ Printed on 2025/10/16 00:50

2025/10/16 00:50 3/3 KbdPeek

445	ERROR_KBD_FOCUS_REQUIRED
447	ERROR_KBD_KEYBOARD_BUSY
464	ERROR_KBD_DETACHED
504	ERROR_KBD_EXTENDED_SG

Remarks

On an enhanced keyboard, the secondary enter key returns the normal character 0DH and a scan code of E0H.

Double-byte character codes (DBCS) require two function calls to obtain the entire code.

If shift report is set with *KbdSetStatus* the *CharData* record returned, reflects changed shift information only.

Extended ASCII codes are identified with the status byte, bit 1 on and the ASCII character code being either 00H or E0H. Both conditions must be satisfied for the character to be an extended keystroke. For extended ASCII codes, the scan code byte returned is the second code (extended code). Usually the extended ASCII code is the scan code of the primary key that was pressed.

A thread in the foreground session that repeatedly polls the keyboard with *KbdCharln* (with no wait), can prevent all regular priority class threads from executing. If polling must be used and a minimal amount of other processing is being performed, the thread should periodically yield the CPU by issuing a *DosSleep* call for an interval of at least 5 milliseconds.

Family API Considerations

Some options operate differently in the DOS mode than in the OS/2 mode. Therefore, the following restrictions apply to KbdPeek when coding for the DOS mode:

- The CharData structure includes everything except the time stamp.
- Interim character is not supported.
- Status can be 0 or 1.
- KbdHandle is ignored.

From:

https://ftp.osfree.org/doku/ - osFree wiki

Permanent link:

https://ftp.osfree.org/doku/doku.php?id=en:ibm:prcp:kbd:peek&rev=1400264582

Last update: 2014/05/16 18:23

