KndStringIn

Bindings: C, MASM

This call reads a character string (character codes only) from the keyboard.

KbdStringIn (CharBuffer, StringLength, IOWait, KbdHandle)

CharBuffer (**PCH**) - output Address of the character string buffer.

StringLength (**PSTRINGINBUF**) - input/output Address of the length of the character string buffer. On entry, buflen is the maximum length, in bytes, of the buffer. The maximum length that can be specified is 255. Template processing has meaning only in the ASCII mode.

buflen (**USHORT**) Length of the input buffer.

inputlen (**USHORT**) Number of bytes read into the buffer.

IOWait (**USHORT**) - input Wait if a character is not available.

Value	Definition
0	Wait. In Binary input mode, the requestor waits until *CharBuffer* is full. In ASCII input mode, the requestor waits until a carriage return is pressed.
1	No wait. The requestor gets an immediate return if no characters are available. If characters are available, *KbdStringIn* returns immediately with as many characters as are available (up to the maximum). No wait is not supported in ASCII input mode.

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

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

0	NO_ERROR
375	ERROR_KBD_INVALID_IOWAIT
439	ERROR_KBD_INVALID_HANDLE
445	ERROR_KBD_FOCUS_REQUIRED
464	ERROR_KBD_DETACHED
504	ERROR_KBD_EXTENDED_SG

Remarks

The character strings may be optionally echoed on the display if echo mode is set. When echo is on each character is echoed as it is read from the keyboard. Echo mode and BINARY mode are mutually exclusive. Reference *KbdSetStatus* and *KbdGetStatus* for more information.

The default input mode is ASCII. In ASCII mode, 2-byte character codes only return in complete form. An extended ASCII code is returned in a 2-byte string. The first byte is 0DH or E0H and the next byte is an extended code.

In input mode (BINARY, ASCII), The following returns can be set and retrieved with *KbdSetStatus* and *KbdGetStatus*:

Turnaround Character Echo Mode Interim Character Flag Shift State

The received input length is also used by the *KbdStringIn* line edit functions for re-displaying and entering a caller specified string. On the next *KbdStringIn* call the received input length indicates the length of the input buffer that may be recalled by the user using the line editing keys. A value of 0 inhibits the line editing function for the current *KbdStringIn* request.

KbdStringIn completes when the handle has access to the physical keyboard (focus), or is equal to zero and no other handle has the focus.

Family API Considerations

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

- *KbdHandle* is ignored

Refer to the *DosRead* Family API Considerations for differences between DOS and OS/2 node when reading from a handle opened to the CON device.

C bindings

```
typedef struct _STRINGINBUF {   /* kbsi */
  USHORT cb;                    /* input buffer length */
  USHORT cchIn;                 /* received input length */
} STRINGINBUF;

#define INCL_KBD

USHORT  rc = KbdStringIn(CharBuffer, Length, IOWait, KbdHandle);

PCH             CharBuffer;     /* Char string buffer */
PSTRINGINBUF    Length;         /* Length table */
USHORT          IOWait;         /* Indicate if wait for char */
HKBD            KbdHandle;      /* Keyboard handle */

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

MASM bindings

```
STRINGINBUF struc
  kbsi_cb    dw  ? ;input buffer length
  kbsi_cchIn dw  ? ;received input length
STRINGINBUF ends

EXTRN  KbdStringIn:FAR
INCL_KBD            EQU 1

PUSH@  OTHER   CharBuffer    ;Char string buffer
PUSH@  OTHER   Length        ;Length table
PUSH   WORD    IOWait        ;Indicate if wait for char
```

```
PUSH    WORD    KbdHandle      ;Keyboard handle
CALL    KbdStringIn

Returns WORD
```

From:
https://osfree.ru/doku/ - **osFree wiki**

Permanent link:
https://osfree.ru/doku/doku.php?id=en:ibm:prcp:kbd:strin

Last update: **2016/09/15 03:00**