

# keybd\_event function

12/05/2018 2 minutes to read

## In this article

[Syntax](#)

[Parameters](#)

[Return Value](#)

[Remarks](#)

[Requirements](#)

[See Also](#)

Synthesizes a keystroke. The system can use such a synthesized keystroke to generate a [WM\\_KEYUP](#) or [WM\\_KEYDOWN](#) message. The keyboard driver's interrupt handler calls the `keybd_event` function.

**Note** This function has been superseded. Use **SendInput** instead.

## Syntax

 Copy

```
void keybd_event(  
    BYTE      bVk,  
    BYTE      bScan,  
    DWORD     dwFlags,  
    ULONG_PTR dwExtraInfo  
);
```

## Parameters

`bVk`

Type: **BYTE**

A virtual-key code. The code must be a value in the range 1 to 254. For a complete list, see [Virtual Key Codes](#).

`bScan`

Type: **BYTE**

A hardware scan code for the key.

`dwFlags`

Type: **DWORD**

Controls various aspects of function operation. This parameter can be one or more of the following values.

Value	Meaning
<b>KEYEVENTF_EXTENDEDKEY</b> 0x0001	If specified, the scan code was preceded by a prefix byte having the value 0xE0 (224).
<b>KEYEVENTF_KEYUP</b> 0x0002	If specified, the key is being released. If not specified, the key is being depressed.

`dwExtraInfo`

Type: **ULONG\_PTR**

An additional value associated with the key stroke.

## Return Value

This function does not return a value.

## Remarks

An application can simulate a press of the PRINTSCRN key in order to obtain a screen snapshot and save it to the clipboard. To do this, call **keybd\_event** with the *bVk* parameter set to **VK\_SNAPSHOT**.

### Examples

The following sample program toggles the NUM LOCK light by using **keybd\_event** with a virtual key of **VK\_NUMLOCK**. It takes a Boolean value that indicates whether the light should be turned

off (**FALSE**) or on (**TRUE**). The same technique can be used for the CAPS LOCK key (**VK\_CAPITAL**) and the SCROLL LOCK key (**VK\_SCROLL**).

```
#include <windows.h>
void SetNumLock( BOOL bState )
{
    BYTE keyState[256];
```



Copy

```
    GetKeyboardState((LPBYTE)&keyState);
    if( (bState &&& !(keyState[VK_NUMLOCK] & 1)) ||
        (!bState &&& (keyState[VK_NUMLOCK] & 1)) )
    {
        // Simulate a key press
        keybd_event( VK_NUMLOCK,
                    0x45,
                    KEYEVENTF_EXTENDEDKEY | 0,
                    0 );

        // Simulate a key release
        keybd_event( VK_NUMLOCK,
                    0x45,
                    KEYEVENTF_EXTENDEDKEY | KEYEVENTF_KEYUP,
                    0);
    }

}

void main()
{
    SetNumLock( TRUE );
}
```

## Requirements

Minimum supported client	Windows 2000 Professional [desktop apps only]
Minimum supported server	Windows 2000 Server [desktop apps only]

<b>Target Platform</b>	Windows
<b>Header</b>	winuser.h (include Windows.h)
<b>Library</b>	User32.lib
<b>DLL</b>	User32.dll

## See Also

### Conceptual

[GetAsyncKeyState](#)

[GetKeyState](#)

[Keyboard Input](#)

[MapVirtualKey](#)

### Reference

[SetKeyboardState](#)

[keybd\\_event](#)