



ActiveX Keyboard Control Developers Toolkit For Windows NT

Description

The ACCESS range of keyboard Active X controls allows full and easy integration of ACCESS Keyboard hardware into your Windows application. This allows the user to instruct the system to deliver all keystrokes as events directly regardless of focus or other applications that may be running on the system. The keyboard hardware has an additional 'wedge' socket that acts as an AT host allowing connection of any standard keyboard (even when the system is live). Keystrokes from the keyboard connected to the wedge can be blocked or passed through to be processed by Windows.

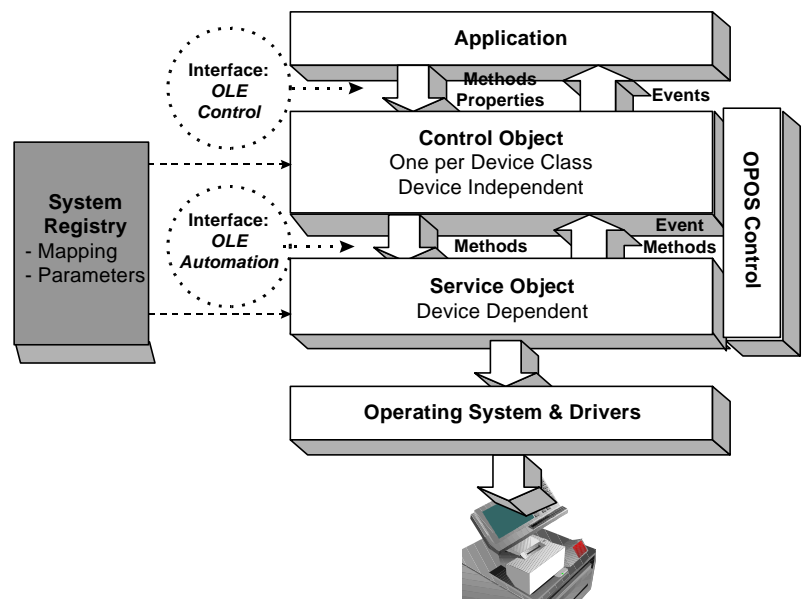
Overview

Windows NT presents many barriers when integrating keypads to generate robust and secure applications. Access Keyboards ActiveX Keyboard Toolkit provides the following functionality **via the keyboard port** with just a few lines of code:

- The master keypad has device status which allows it to be exclusively claimed
- Keyboard events are reported to the controlling application regardless of foreground focus
- Disable (lock) the keyboard and/or the wedge keyboard.
- Queuing of Keyboard events until the application is ready to process or discard the events
- Inhibit windows control functions
- Control the keyboard Auxiliary serial port
- Sound internal keyboard beeper with a variety of standard beeps. (useful if keyboard is remote from system unit)
- Control up to status 4 LEDs on the keyboard
- Momentary layer shift and 'sticky' shift keys to extend keyboard functionality (key combinations)
- Full tracking of 6 position key lock
- OPOS Compatible
- Support for a standard layout 'slave' keyboard that operates in standard NT mode

Applications

- Point of Sale
- Financial Dealing Room Keypads
- Instrumentation
- Industrial Control
- Medical
- Security



Keyboard Controller Hardware:

Standard Features:

- Standard PC/AT or PS/2 Interface
- Slave Keyboard 'wedge' port

Options:

- Slave RS232 Serial interface
- + 5V DV 150mA Peripheral Power Output
- Up to 4 LED Indicators
- 6 Position Key lock
- Integrated 2 Track Magnetic Swipe Reader

WWW.ACCESSKEYBOARDS.CO.UK

Code Extract:

```

' Code to enable the keyboard event reporting
KEYBOARD1.DeviceOpen ("AKE116")
KEYBOARD1.Claim
KEYBOARD1.DataEventEnabled
KEYBOARD1.FreezeEvents = False
KEYBOARD1.DeviceEnabled = True
'
' The keypresses are reported as a logical key number in
' the control's Dataevent Event

```

How an Application Uses The Control

The first action the application must take on the Control is to call its **Open** method. The parameter of this method selects a device name to associate with the Control. The **Open** method performs the following steps:

- Establishes a link to the device name.
- Initializes the properties **Claimed**, **DeviceEnabled**, **DataEventEnabled**, **FreezeEvents**, **AutoDisable**, **DataCount**, and **BinaryConversion**, as well as descriptions and version numbers of the ActiveX Control layers. Additional class-specific properties may also be initialized.

Several applications may have the Control open at the same time. Therefore, after the device is opened, the application will often need to call the **Claim** method to gain exclusive access to the device. Many devices must be **Claimed** before the Control allows access to its methods and properties. Claiming the device ensures that other applications do not interfere with the use of the device. The application may **Release** the device when the device can be shared by other applications – for instance, at the end of a transaction.

Before using the device, the application must set the **DeviceEnabled** property to TRUE. This value brings the device to an operational state, while FALSE disables the device. For example, if the Keyboard Control is disabled, then the device will be physically disabled (when possible). Whether physically disabled or not, any input from the device will be discarded until the device is enabled.

After the application has finished using the device, the **Close** method should be called to release the device and associated resources. If the **DeviceEnabled** property is TRUE, then **Close** disables the device. If the **Claimed** property is TRUE, then **Close** releases the lock. Before exiting, an application should close all open Access Controls.
