# Chapter 6—qkey Reference

## Introduction

The QKEY class gives your external component access to keyboard messages and some keyboard checking functions. It refers to two kinds of key, a VCHAR and a PCHAR. A VCHAR is a virtual key code for special keys such as the PageUp key. PCHAR refers to printable characters.

Keyboard messages WM\_KEYDOWN and WM\_KEYUP pass a pointer to a qkey object.

## Enumerations

#### vChar

An enum defining some virtual keyboard values.

* vcF1  
  The F1 key on the keyboard
* vcUp  
  The up arrow key on the keyboard
* vcDown  
  The down arrow key on the keyboard
* vcLeft  
  The left arrow key on the keyboard
* vcRight  
  The right arrow key on the keyboard
* vcPup  
  The page up key on the keyboard
* vcPdown  
  The page down key on the keyboard
* vcPleft  
  The page left key on the keyboard
* vcPright  
  The page right key on the keyboard
* vcHome  
  The home key on the keyboard
* vcEnd  
  The end key on the keyboard
* vcTab  
  The tab key on the keyboard
* vcReturn  
  The return key on the keyboard
* vcEnter  
  The enter key on the keyboard
* vcBack  
  The backspace key on the keyboard
* vcClear  
  The clear key on the keyboard
* vcCancel  
  The escape key on the keyboard
* vcDel  
  The forward delete key on the keyboard
* vcIns  
  The insert key on the keyboard

## qkey Class Reference

### qkey::qkey()

|  |
| --- |
| qkey::qkey( LPARAM pKeyValue ) |

The constructor for the external keyboard class. After construction, the class can be used to interrogate the keyboard message.

1. **pKeyValue** - This is the keyboard scan value passed in LPARAM on a WM\_KEYDOWN, WM\_KEYUP message.

### qkey::qkey()

|  |
| --- |
| qkey::qkey( pchar pPchar, qbool pShift, qbool pOption, qbool pControl ) |

Creates a qkey object from the printable character and key states passed.

1. **pPchar** - The printable character to be added into the new qkey.
2. **pShift** - The state of the shift key for the new qkey object.
3. **pOption** - The state of the option key for the new qkey object.
4. **pControl** - The state of the control key for the new qkey object.
5. **return** - Returns a new qkey object.

**See also** qkey::getPChar()

### qkey::qkey()

|  |
| --- |
| qkey::qkey( vchar pVchar, qbool pShift, qbool pOption, qbool pControl ) |

Creates a qkey object from the virtual key code and key states passed.

1. **pVchar** - The virtual keyboard value to be added into the new qkey.
2. **pShift** - The state of the shift key for the new qkey object.
3. **pOption** - The state of the option key for the new qkey object.
4. **pControl** - The state of the control key for the new qkey object.
5. **return** - Returns a new qkey object.

### qkey::qkey()

|  |
| --- |
| qkey::qkey( ) |

Creates a qkey object with only the modifier states ( SHIFT, CONTROL and OPTION ) set.

1. **return** - Returns a new qkey object.

### qkey::getPChar()

|  |
| --- |
| pchar qkey::getPChar() |

Returns the printable character from the key message.

1. **returns** - Returns the character.

### qkey::getVChar()

|  |
| --- |
| vchar qkey::getVChar() |

Returns the virtual key code from the key message.

1. **returns** - Returns the key code.

### qkey::isAlt()

|  |
| --- |
| qbool qkey::isAlt() |

Returns the state of the ALT key for this keyboard message.

1. **returns** - Returns qtrue if the ALT key is down.

### qkey::isControl()

|  |
| --- |
| qbool qkey::isControl() |

Returns the state of the CONTROL key for this keyboard message.

* **returns** - Returns qtrue if the CONTROL key is down.

### qkey::isShift()

|  |
| --- |
| qbool qkey::isShift() |

Returns the state of the SHIFT key for this keyboard message.

1. **returns** - Returns qtrue if the SHIFT key is down.

### qkey::operator !()

|  |
| --- |
| qbool qkey::operator ! ( ) |

Tests if the qkey object is invalid.

1. **return** - qtrue if the qkey object is invalid and qfalse if the object is valid.

### qkey::operator !=()

|  |
| --- |
| qbool qkey::operator != ( const qkey& pTestKey ) |

Compares the key message stored in this qkey object with the key message passed in.

1. **pTestKey** - The qkey object to compare against.
2. **return** - qtrue if the qkey key messages are not the same.

### qkey::operator ==()

|  |
| --- |
| qbool qkey::operator == ( const qkey& pTestKey ) |

Compares the key message stored in this qkey object with the key message passed in.

1. **pTestKey** - The qkey object to compare against.
2. **return** - qtrue if the qkey key messages match and qfalse if the objects are different.

### qkey::uppc()

|  |
| --- |
| void qkey::uppc() |

Uppercases the printable character stored in the qkey object.

**See also** qkey::getPChar()

## Other Functions

### isShift()

|  |
| --- |
| qbool isShift() |

Returns the current state of the SHIFT key.

1. **returns** - Returns qtrue if the SHIFT key is down and qfalse if up.

### isAlt()

|  |
| --- |
| qbool isAlt() |

Returns the current state of the ALT key.

1. **returns** - Returns qtrue if the ALT key is down and qfalse if up.

**Example:**

###### extern "C" qlong OMNISWNDPROC GenericWndProc( HWND hwnd, LPARAM Msg, WPARAM wParam, LPARAM lParam, EXTCompInfo\* eci ) {   ECOsetupCallbacks(hwnd,eci);   switch (Msg)   {     case WM\_KEYDOWN:     case WM\_KEYUP:     {       qkey\* keyMessage = (qkey\*)lParam;       if ( keyMessage->getPChar()==’P’ )       {         // The ‘P’ key was pressed.         return 0L; // tell Omnis we have processed the key       }       else if ( keyMessage->isShift() && keyMessage->getPChar()==’L’ )       {         // The ‘L’ key and ‘SHIFT’ keys were pressed.         return 0L; /// tell Omnis we have processed the key       }       return 1L; // let Omnis process the key     }   }   return WNDdefWindowProc(hwnd,Msg,wParam,lParam,eci); }