

## Retrieving Information from Textboxes with SendMessageA

### The Native Liberty BASIC Command

## Table of Contents

[Retrieving Information from Textboxes with SendMessageA](#)

[The Native Liberty BASIC Command](#)

[A Multiline Textbox](#)

[CallDLL #user32, "SendMessageA"](#)

[Get the Number of Lines in the Textbox](#)

[Get the Number of Characters in Each Line of Text](#)

[Liberty BASIC's Mid\\$\(\) Function](#)

Liberty BASIC has a native command for retrieving textbox contents -

```
Print #main.tbx1, "!Contents? text$"
```

In most instances, this is all the code programs require.

```
WindowWidth = 400
WindowHeight = 300
UpperLeftX = Int((DisplayWidth - WindowWidth) / 2)
UpperLeftY = Int((DisplayHeight - WindowHeight) / 2)
Statictext #main.stx1, "What is your name?", 20, 34, 70, 60
Textbox #main.tbx1, 100, 50, 200, 30
Button #main.default, " OK ", GetName, UL, 320, 50
Open "Retrieving Textbox Contents" for Dialog as #main
#main "Trapclose XbyTrap"
#main "Font Times_New_Roman 12 Bold"
```

```
Wait
```

```
Sub XbyTrap handle$
  Close #handle$
```

```
End
End Sub

Sub GetName handle$
    #main.tb1 "!Contents? name$"
    Notice "Hello, ";name$
End Sub
```

### A Multiline Textbox

There are times when a [multiline, word wrapping textbox](#) may be preferred over the native Liberty BASIC texteditor. See also [An Improved Stylebits Textbox](#).

There may also be a need to

1. Get the number of lines in the textbox
2. Get the number of characters in each line of text
3. Get each separate line of text

These functions are possible with an API call to #user32, "SendMessage".

### CallDLL #user32, "SendMessage"

The SendMessage function sends a specified message to a window or a control. The arguments to be passed include

- handle The handle of the window or control ULONG
- Msg The specific message LONG
- wParam additional information about the message LONG
- lParam additional information about the message LONG

When the message is received, the return value specifies the result of the message processing.

- result LONG

Visit the [MSDN Library](#) for more information about [SendMessage](#).

### Get the Number of Lines in the Textbox

The Windows constant `_EM_GETLINECOUNT` is the message used to retrieve the number of lines from a textbox.

```
CallDLL #user32, "SendMessageA", _  
    hText as ULong, _ ' handle of the textbox  
    _EM_GETLINECOUNT as Long, _ ' message to be passed  
    0 as Long, _ ' no further information to be passed  
    0 as Long, _ ' no further information to be passed  
    result as Long ' Windows returns the number of lines
```

Here is a demo to get the number of lines in a textbox. The mainwindow is kept visible for the printing of returned information. You may need to reposition your GUI window to see this information.

```
' Define a textbox  
    Textbox #main.tb, 10, 20, 150, 68  
  
' Stylebits for word wrapping and vertical scrolling  
    Stylebits #main.tb, _WS_VSCROLL OR  
    _ES_MULTILINE, _ES_AUTOHSCROLL, 0, 0  
  
' Define text  
    demoText$ = "All Gaul is divided into three parts, one of " + _  
"which the Belgae inhabit, the Aquitani another, those who " + _  
"in  
    their own language are called Celts, in our Gauls, the third. " + _  
"All these differ from each other in language, customs, and laws."  
  
    Open "Lines of Multiline Textbox" for Window as #main  
    #main "Trapclose XbyTrap"  
    #main "Font Times_New_Roman 10 bold"  
  
' Print the text in the textbox  
    #main.tb demoText$  
  
' Get the handle of the textbox  
    hText=hWnd(#main.tb)  
  
' Pass the required arguments to SendMessageA API  
    nLines = ReturnedMessage(hText, _EM_GETLINECOUNT, 0, 0)  
    Print "There are ";nLines;" lines in this textbox."
```

Wait

```
' Program ends here
Sub XbyTrap handle$
    Close #handle$
End
End Sub

' SendMessageA Function
Function ReturnedMessage(handle, Msg, wParam, lParam)
    CallDLL #user32, "SendMessageA", _
        handle as Ulong, _
        Msg as Long, _
        wParam as Long, _
        lParam as Long, _
        ReturnedMessage as Long
End Function
```

## Get the Number of Characters in Each Line of Text

SendMessageA will not return the actual text string. Instead, it returns the number of characters in that string. This next demo is the same as the previous demo, with a function added to get the number of characters at the end of each line. It is important to recognize that the function does not return the number of characters in that line, but the number of characters from the *beginning of the text* to the *end of that line*. Again using the API call SendMessageA, the handle of the textbox is passed as well as the message `_EM_LINEINDEX`. SendMessageA also needs to know which line is being questioned. The first available extra information argument, wParam, passes the line number to be queried. There is no additional information needed, so the second informational argument, lParam, remains 0.

```
CallDLL #user32, "SendMessageA", _
    hText as Ulong, _ ' handle of the textbox
    _EM_LINEINDEX as Long, _ ' message to be passed
    lineNumber as Long, _ ' the queried line number
    0 as Long, _ ' no further information to be passed
    result as Long ' Windows returns the number of lines
```

Since the last line is treated a little differently, the loop must terminate at the next to the last line `For i = 1 to nLines - 1`. The code for the last line immediately follows the loop.

```
' Define a textbox
Textbox #main.tb, 10, 20, 150, 68
```

```
' Stylebits for word wrapping and vertical scrolling
  Stylebits #main.tb, _WS_VSCROLL OR
  _ES_MULTILINE, _ES_AUTOHSCROLL, 0, 0

' Define text
  demoText$ = "All Gaul is divided into three parts, one of " + _
"which the Belgae inhabit, the Aquitani another, those who " + _
"in
  their own language are called Celts, in our Gauls, the third. " + _
"All these differ from each other in language, customs, and laws."

  Open "Lines of Multiline Textbox" for Window as #main
  #main "Trapclose XbyTrap"
  #main "Font Times_New_Roman 10 bold"

' Print the text in the textbox
  #main.tb demoText$

' Get the handle of the textbox
  hText=hWnd(#main.tb)

' Pass the required arguments to SendMessageA API
  nLines = ReturnedMessage(hText, _EM_GETLINECOUNT, 0, 0)
  Print "There are ";nLines;" lines in this textbox."

' Get the total number of characters at the end of each line
  For i = 1 to nLines - 1
    nCharacters = ReturnedMessage(hText, _EM_LINEINDEX, i, 0)
    Print "Line #";i;": ";nCharacters;" Characters"
  Next i

Wait

' Program ends here
Sub XbyTrap handle$
  Close #handle$
End
End Sub

' SendMessageA Function
Function ReturnedMessage(handle, Msg, wParam, lParam)
  CallDLL #user32, "SendMessageA", _
    handle as Ulong, _
```

```
Msg as Long, _  
wParam as Long, _  
lParam as Long, _  
ReturnedMessage as Long  
End Function
```

How will knowing what the total number of characters reveal the actual number of characters in each line? By keeping track of how many characters were used in preceding lines, the number of characters already assigned is known. Subtract that number from the number of characters up to and including the selected line. The difference is the number of characters in just the selected line. Start the position, startPos, as 1.

```
' Define a textbox  
Textbox #main.tb, 10, 20, 150, 68  
  
' Stylebits for word wrapping and vertical scrolling  
Stylebits #main.tb, _WS_VSCROLL OR  
_ES_MULTILINE, _ES_AUTOHSCROLL, 0, 0  
  
' Define text  
demoText$ = "All Gaul is divided into three parts, one of " + _  
"which the Belgae inhabit, the Aquitani another, those who " + _  
"in  
their own language are called Celts, in our Gauls, the third. " + _  
"All these differ from each other in language, customs, and laws."  
  
Open "Lines of Multiline Textbox" for Window as #main  
#main, "Trapclose XbyTrap"  
#main, "Font Times_New_Roman 10 bold"  
  
' Print the text in the textbox  
#main.tb demoText$  
textLength = Len(demoText$)  
  
' Get the handle of the textbox  
hText=hWnd(#main.tb)  
  
' Pass the required arguments to SendMessageA API  
nLines = ReturnedMessage(hText, _EM_GETLINECOUNT, 0, 0)  
Print "There are ";nLines;" lines in this textbox"  
  
' The starting position is 1
```

```
startPos = 1

For i = 1 to nLines - 1
' Get the total number of characters in each line
    nCharacters = ReturnedMessage(hText, _EM_LINEINDEX, i, 0)
' The characters in that line start with startPos and continue through
to nCharacters
    Print "Line ";i;": Characters ";startPos;" - ";nCharacters
' startPos moves up to the nCharacters pos for the next line
    startPos = nCharacters + 1
Next i
' Get the number of characters in the last line by subtracting the nCh
aracters
' found thus far from the Length of the entire text
    nCharacters = textLength - nCharacters
    Print "Line ";i;": Characters ";startPos;" - ";textLength

'Wait for button to be pushed
Wait

' Program ends here
Sub XbyTrap handle$
    Close #handle$
End
End Sub

' SendMessageA Function
Function ReturnedMessage(handle, Msg, wParam, lParam)
    CallDLL #user32, "SendMessageA", _
        handle as Ulong, _
        Msg as Long, _
        wParam as Long, _
        lParam as Long, _
        ReturnedMessage as Long
End Function
```

## **Liberty BASIC's Mid\$() Function**

Now that the starting and ending character position number is known, the Liberty BASIC Mid\$() function can be used to extract that one line. The Mid\$() function has 3 elements

1. The whole text to be parsed (demoText\$)
2. The starting position (startPos)

3. The number of characters to be extracted from the starting position to the right (nCharacters - startPos)

A simple demo of the Liberty BASIC Mid\$() function -

```
txt$ = "trouble"  
m$ = Mid$(txt$, 4, 3)  
Print m$  
End
```

Executing this code prints ubl, where u is the character in the fourth position of the string trouble, and ubl are the three characters in sequence beginning in this fourth position.

In this textbox demo, startPos becomes the start position and nCharacters - startPos becomes the sequence range of characters. Here is the code again, this time extracting and printing each line of text.

```
' Define a textbox  
    Textbox #main.tb, 10, 20, 150, 68  
  
' Stylebits for word wrapping and vertical scrolling  
    Stylebits #main.tb, _WS_VSCROLL OR  
    _ES_MULTILINE, _ES_AUTOHSCROLL, 0, 0  
  
' Define text  
    demoText$ = "All Gaul is divided into three parts, one of " + _  
"which the Belgae inhabit, the Aquitani another, those who " + _  
"in  
their own language are called Celts, in our Gauls, the third. " + _  
"All these differ from each other in language, customs, and laws."  
  
    Open "Lines of Multiline Textbox" for Window as #main  
    #main "Trapclose XbyTrap"  
    #main "Font Times_New_Roman 10 bold"  
  
' Print the text in the textbox  
    #main.tb demoText$  
    textLength = Len(demoText$)  
  
' Get the handle of the textbox  
    hText=hWnd(#main.tb)
```



```
' Pass the required arguments to SendMessageA API
  nLines = ReturnedMessage(hText, _EM_GETLINECOUNT, 0, 0)
  Print "There are ";nLines;" lines in this textbox"

' The starting position is 1
  startPos = 1

  For i = 1 to nLines - 1
' Get the total number of characters in each line
    nCharacters = ReturnedMessage(hText, _EM_LINEINDEX, i, 0)
' Print that line of text using the Mid$() function
    Print Mid$(demoText$, startPos, nCharacters - startPos)
' startPos moves up to the nCharacters pos for the next line
    startPos = nCharacters + 1
  Next i
' Get the number of characters in the last line by subtracting the nCh
aracters
' found thus far from the Length of the entire text
  nCharacters = textLength - nCharacters
  Print Mid$(demoText$, startPos, nCharacters)

'Wait for button to be pushed
  Wait

' Program ends here
Sub XbyTrap handle$
  Close #handle$
End
End Sub

' SendMessageA Function
Function ReturnedMessage(handle, Msg, wParam, lParam)
  CallDLL #user32, "SendMessageA", _
    handle as Ulong, _
    Msg as Long, _
    wParam as Long, _
    lParam as Long, _
    ReturnedMessage as Long
End Function
```

You may never need to extract single lines of text in a multiline textbox, but at least you know you can.