UserName and CompanyName An example from and by Ch. Germelmann

Access Windows UserName and CompanyName with MS Visual Basic (or every other programming language).

Additionally you gain the ability of obtaining the true Windows version and of creating multilingual applications.

Please select...

- **I want to know something about the access...**
- **I want to see the code...**

The described code will work with *any* Windows version *but* Windows NT !
(If YOU have a solution please let me know. I will immediately update this file !)

Author: Christian Germelmann
Am Glaskopf 26
35039 Marburg /Lahn
Germany
CompuServe 100520,2644
Phone +49 6421 45457

P.S.: This is meant as a demonstration and not as an 'extraordinaryly designed' program !

That's one reason why I didn't put this source code into shareware.

This code is for free but if you like (and/or needed) this hint you are free to register one of my shareware programs to express your appreciation.
Just have a look on them -- maybe you need one of them anyway (!)
Search for my CIS-ID on VBPJ, MSBASIC, WINSHARE, and other forums...

Suggestions of any kind are welcome !!!

How To Access The Required Data

The Declares For VB

There are *only two* ***API functions*** we need to use:
(Users of other programming languages click '*...more about this function*'.)

Declare Function **LoadString** Lib "USER" Alias "#176" (ByVal *hInstance* As Integer, ByVal *wID* As Integer, ByVal *lpBuffer* As Any, ByVal *nBufferMax* As Integer) As Integer

or shorter:

Declare Function **LoadString**% Lib "USER" (ByVal *hInstance%*, ByVal *wID%*, ByVal *lpBuffer* As Any, ByVal *nBufferMax%*)

...more about this function

and

Declare Function **GetModuleHandle** Lib "KERNEL" Alias "#47" (ByVal *lpModuleName* As String) As Integer

or shorter:

Declare Function **GetModuleHandle**% Lib "KERNEL" (ByVal *lpModuleName$*)

...more about this function

The Return Values

We access the *module USER* (`USER.EXE`) which contains the desired string data since the Windows setup.

String Number	Return Value
514	**UserName** The *first line of your user input* in the Windows setup.
515	**CompanyName** The *second line of your user input* in the Windows setup.
516	**WindowsVersion** The *true Windows Version* implemented when the USER.EXE was built.

Of course, there are even more strings to access with different numbers. But they are not of such an interest for this time...

WindowsVersions

Version Number	Windows Version
3.10	"old" normal Windows (16-bit)
3.11	"old" Windows for Workgroups (16-bit)
4.00.950	Windows 95 (32-bit)
(*none)*	Windows NT (32-bit)

(In case you find out more or receive different data
please report them to me (the author) !)

Further Strings From Modules

Using the shown method you can access strings from *any loaded module*.
You simply exchange "USER" against the module name of your choice
and select a string number. Other modules could be loaded as well for
the short moment it takes to obtain a string.

- This is a good way to build up **multilingual applications** !

*(Since the meaning of all strings are identical in all Windows language versions
this is the author's preferred way of creating universal error handlers.
I'm giving some example numbers for 'USER' in the source code text !)*

LoadString

int LoadString(hinst, idResource, lpszBuffer, cbBuffer**)**

```
HINSTANCE hinst;        /* handle of module containing string resource     */
UINT idResource;        /* resource identifier                             */
LPSTR lpszBuffer;       /* address of buffer for resource                  */
int cbBuffer;           /* size of buffer                                  */
```

The **LoadString** function loads the specified string resource.

Parameter	Description
hinst	Identifies an instance of the module whose executable file contains the string resource to be loaded.
idResource	Specifies the integer identifier of the string to be loaded.
lpszBuffer	Points to the buffer that will receive the null-terminated string.
cbBuffer	Specifies the buffer size, in bytes. The buffer should be large enough for the string and its terminating null character. The string is truncated if it is longer than the number of bytes specified.

Returns

The return value specifies the number of bytes copied into the buffer, if the function is successful. It is zero if the string resource does not exist.

From SDK-Help `WIN31WH.HLP.`

Microsoft Windows
Software **D**evelopment **K**it

Version 3.1

GetModuleHandle

HMODULE GetModuleHandle(lpszModuleName**)**

LPCSTR *lpszModuleName;* /* address of name of module */

The **GetModuleHandle** function retrieves the handle of the specified module.

Parameter	Description
lpszModuleName	Points to a null-terminated string that specifies the name of the module.

Returns
The return value is the handle of the module if the function is successful. Otherwise, it is NULL.

From SDK-Help `WIN31WH.HLP.`

Close | Copy | Print

The Sample Code 'UserName'

The example uses the **LoadString** and **GetModuleHandle** API functions to access some internal Windows data. To try this example, open a new project, create three text boxes 'Text(0)' through 'Text(2)', a command button 'cmd' and paste the code into the Declarations section of the form. Then press F5 to run the program.

Please notice that all Declares must appear on *one* line: In case you use the copy dialog of WinHelp the lines may be departed !

(The sample project USERNAME.MAK containing USERNAME.FRM and USERNAME.FRX should be included in the same archive as this help file. USERNAME.FRM contains the following code.)

```
Option Explicit

Dim retInt%       ' holds an Integer variable '
Dim retLng&       ' holds a Long variable      '

' ********************************************************************
' * The declarations for disabling the editing of the text boxes: *
' ********************************************************************
Declare Function SendMessage& Lib "USER" Alias "#111" (ByVal hWnd%, ByVal wMsg%,
ByVal wParam%, lParam As Any)
Const WM_USER& = &H400
Const EM_SETREADONLY& = (WM_USER + 31)

' ************************************************
' * The API declarations to the 'secret access': *
' ************************************************
Declare Function GetModuleHandle% Lib "KERNEL" Alias "#47" (ByVal lpModuleName$)
Declare Function LoadString% Lib "USER" Alias "#176" (ByVal hInstance%, ByVal wID%,
ByVal lpBuffer As Any, ByVal nBufferMax%)

' --> Correct numbers for the Alias-declarations '
'       can e.g. be achived with the author's      '
'       program APIMAN.EXE.                        '

Sub Form_Load ()

    ' ****************************************
    ' Fill the text boxes with the desired data
    ' ****************************************
    Text(0) = UserName()
    Text(1) = CompanyName()
    Text(2) = TrueWinVer()

    ' ********************************
    ' Make the text boxes 'untouchable'
    ' ********************************
    retLng = SendMessage(Text(0).hWnd, EM_SETREADONLY, True, 0)
    retLng = SendMessage(Text(1).hWnd, EM_SETREADONLY, True, 0)
    retLng = SendMessage(Text(2).hWnd, EM_SETREADONLY, True, 0)
```

```
    ' ************************
    ' Place the form on screen
    ' ************************
    Move (Screen.Width - Width) / 2, (Screen.Height - Height) / 2.5

    Show
End Sub

Sub cmd_Click ()
    Unload Me
End Sub

Sub Text_Change (Index As Integer)
    Text(Index).SelStart = 0
    Text(Index).SelLength = 30
    DoEvents
End Sub

Function GetUserString$ (StringNumber%)
'                                                                '
' This is the function that returns a string from the USER.EXE.  '
' To access other loaded (!) modules exchange "USER" against the  '
' other module's name and select a string number.                 '
'                                                                '
' In the USER.EXE we e.g. find:                                   '
'    514   the UserName                                           '
'    515   the CompanyName                                        '
'    516   the true Windows version                               '
'    518   the serial number                                      '
'                                                                '
' and for multilingual purposes:                                  '
'     85   'Cancel'                                              '
'     86   '&Abort'                                              '
'     87   '&Retry'                                              '
'     88   '&Ignore'                                             '
'     89   '&Yes'                                                '
'     90   '&No'                                                 '
'     78   'Error'                                               '
' --> All in the respective language of a country !              '
' --> Use them for international labeling !                      '
'                                                                '

Dim ReturnedString$

    ' ******************************** '
    ' The maximum length of a string at '
    ' that location is 30 characters.   '
    ' So we preload only 30 spaces.     '
    ' ******************************** '
    ReturnedString = Space(30)
```

```
    retInt = LoadString(GetModuleHandle("USER"), StringNumber, ReturnedString,
Len(ReturnedString))
    ' ******************************************************** '
    ' We actually do not need this now but when accessing other '
    ' strings for a mulilingual purpose we should keep it here. '
    ' ******************************************************** '
    ReturnedString = (Left$(ReturnedString, retInt))

GetUserString = Trim$(ReturnedString)
End Function

Function UserName$ ()
    UserName = GetUserString(514)
End Function

Function CompanyName$ ()
    CompanyName = GetUserString(515)
End Function

Function TrueWinVer$ ()
    TrueWinVer = GetUserString(516)
End Function
```

After saving the project press F5.

If you have any suggestions or questions contact the author.

C o p y r i g h t

Christian Germelmann
Am Glaskopf 26
35039 Marburg/Lahn
Germany
Phone +49 06421 45457
CompuServe 100520,2644

Disclaimer

This code has been used and tested by me in VB 2.0 Pro and VB 3.0 Pro. Use it at your own risk.
If you use portions of this document or the sample code elsewhere, please indicate where you got it.