GBLIB2.EXE

A registration utility for Microsoft Visual Basic(tm) V3.0
By G.Bamber

Exit Help - **Exit this Help System (or ALT+x)**

Function Reference for GBLIB2.EXE

ReadRegData

Back
Contents
Exit Help

Pascal Declaration

Function ReadRegData (VBEXEPath, USERNAME, USERORG YMD :Pchar; var i_NAMELEN, i_ORGLEN :Integer) :Integer ;Export;

Declaration

Declare Function ReadRegData Lib "GBLIB2.EXE" (ByVal sz_EXEPath As String, ByVal sz_Name As String, ByVal sz_Org As String, YMD as String, NameLen As Integer, OrgLen As Integer) As Integer

Usage and notes

● The function returns Zero for valid information, otherwise returns 1 for unbranded files.
The EXEPath must be a valid compiled Visual Basic(tm) Version 3 program. It need not be the current program, and you can use this function to read the registration details of any branded or unbranded EXE file you like.

● Note that this function does not require a registration code, as the WriteRegData function does.

The sz_Name variable **must** be 33 characters long.
The sz_Org variable **must** be 41 characters long.
The sz_YMD variable **must** be 4 characters long.
sz_Name, sz_Org and sz_YMD would normally be initialised with the VB String$ statement thus:

```
Dim sz_Name as String
Dim sz_Org as String
Dim sz_YMD as String
sz_Name=String$(33,0)
sz_Org=String$(41,0)
sz_YMD=String$(4,0)
```

NameLen and OrgLen are DIMmed thus:

```
Dim NameLen as Integer
Dim OrgLen as Integer
```

The function may then be called.

On return, sz_Name and sz_Org are normally trimmed to length thus:
sz_Name=Left$(sz_Name,NameLen)
sz_Org=Left$(sz_Org,OrgLen)

- In WriteRegData , the Users current system date was branded, along with the name and organisation information. This was done automatically by GBLIB2.EXE.

It is returned in the 4th parameter **sz_YMD**.

sz_YMD is returned as 3 ASCII codes thus:
- YMD(1) = ASCII code representing the Year minus 108   plus 1900 (e.g. Chr$(198) is 1990, Chr$(199) is 1991)
- YMD(2) = ASCII code representing the Month (e.g. 4 would be April, 5 would be May)
- YMD(3) = ASCii code representing the Date (e.g. 1 is the 1st, 2 is the 2nd)

I know that this is a bit complicated to decode, but there were good reasons for coding the date into just 3 ASCII codes, and the REGMAX.BAS example contains VB code to make sense of sz_YMD. (I hope)

If the file is unbranded, then:
- Return value is 1
- sz_Name = UNLICENSED
- sz_Org = UNLICENSED
- sz_YMD = 3 Spaces + Chr$(0)

Otherwise, the registration information is returned accordingly.

See.. WriteRegData

WriteRegData

Back
Contents
Exit Help

Pascal Declaration

Function WriteRegData (USERNAME, USERORG, VBEXEPATH,
CODE :Pchar) :Integer; Export;

Declaration

Declare Function WriteRegData Lib "GBLIB2.EXE" (ByVal sz_Name As String, ByVal sz_Org As String, ByVal sz_EXEPath As String, ByVal sz_Code as String) As Integer

Usage and notes

Back

The function returns Zero for success, otherwise returns 1 if either sz_Name or sz_Org had to be truncated. (they will still be written)

The EXEPath must be a valid compiled Visual Basic(tm) Version 3 program. It need not be the current program, and an installation program, for instance, could use this function to brand any valid VB3-compiled EXE program.

Both sz_Name and sz_Org will be capitalised. (UPPERCASED)

sz_Name has a maximum length of 33 characters.
sz_Org has a maximum length of 41 characters.

Back

The Current System Date is also written to the EXE file in this routine automatically by the DLL.

sz_Code is your individual code that identifies you as a bona fide Visual Basic developer.

Clearly, if you are using GBLIB2 to help your client brand registration information, you would not want anyone else to be able to re-brand their own information on the file willy-nilly.

If you wish to use GBLIB2, therefore, you must follow the procedure outlined in APPFORM.TXT in order to get your own personal sz_Code.

In the spirit of FREEWARE, you will not be charged for this service, though you must pay your own email charges, if any are incurred.

See..ReadRegData

MakeAKey

Back

Back

Back

Pascal Declaration

Function MakeAKey (Astring :Pchar) :LongInt ;Export;

Declaration

Declare Function MakeAKey Lib "GBLIB2.EXE" (ByVal AString As String) As Long

Usage and notes

Back

This is a simple function to turn a string into (hopefully) a unique number. I am no cryptographer, so I make no apologies for the quality of the algorithm.
Normally Astring would be composed of USERNAME & USERORG, and the result stored in a separate file. This can then be crosschecked with the data pulled from the EXE file if required.

ShowWinDir

Back
Back
Back

Pascal Declaration

Procedure ShowWinDir ;Export;

Declaration

Declare Sub ShowWinDir Lib "GBLIB2.EXE" ()

Usage and notes

Contents
This routine merely puts up a message box containing the Users Windows directory.
This, and its partner ShowSysDir can assist in telephone helpline support, where you sometimes have to establish where the users Windows path is.

ShowSysDir

Contents
Contents
Contents

Pascal Declaration

Procedure ShowSysDir ;Export;

Declaration

Declare Sub ShowSysDir Lib "GBLIB2.EXE" ()

Usage and notes

- This routine merely puts up a message box containing the Users Windows\System directory.

This, and its partner ShowWinDir can assist in telephone helpline support, where you sometimes have to establish where the users Windows\System path is.

About GBLIB2.EXE

-
- (or ALT+x)

Programmer G.Bamber
Help File - G.Bamber

- GBLIB2.EXE (c)1994 was written in Borland Turbo Pascal for Windows(tm) V1.5 by Gordon Bamber. .

- It is released as FREEWARE on the following conditions:

1) Copyright remains with the author
2) Commercial use is prohibited without the author"s permission.
3) No responsibility is accepted by the author for support in its use.
4) No responsibility is accepted under any circumstances by the author for any damage caused in its use.
5) Subject to conditions (1) to (4), you may use it in any way you please.
6) There is to be **no** condition (7)
8) Source code for GBLIB2.EXE is not freeware.

Gordon Bamber

Gordon Bamber is a staff programmer at Maxim Training Consultants, a Brighton-based multimedia company.

He can be contacted:

1) email:

- 74437.672@compuserve.com
- gbamber@cix.compulink.com.uk

2) Snailmail:

- Gordon Bamber,
- 11, Helena road
- Woodingdean,
- Brighton, England
- BN1 2BS

Browse Sequence

-
-
-

-

This is the start of the Browse Sequence.

-

Use the buttons [<<] and [>>] to move through the main topics in this help file.

OverView of GBLIB2

-
-
-

1) Purpose

- The purpose of GBLIB2.EXE is to brand a Visual Basic V3.0 compiled EXE with registration details in such a way as to be difficult for an unscrupulous user to alter.

- Once the EXE file is branded by GBLIB2, then even if it is copied to another computer, the brand information will be copied with it.

2) How does it work?

- The WriteRegData routine writes an encrypted version of the registration information into a little-used area of the compiled EXE. Because it is encrypted, it is quite difficult for a hacker to alter. (Though not impossible) If you wish for better security, you could include a checksum in the registration information, though this is not built in to GBLIB2 as such.

- If you will be using GBLIB2, then you would not want me to broadcast details here about the exact areas involved, or the encryption process, so I wont.

- The ReadRegData routine reads these areas, and if the EXE has not been branded, returns the string UNLICENSED and the result code 1. Otherwise, it will retrieve whatever was written by WriteRegData into the EXE.

3) Is is dangerous to use?

- Yes, it could be - unless you know what you are doing. (That is why you are reading this Help file, isnt it?)

- It has only been written for, and tested with Microsoft(tm) Visual Basic Professional Edition Version 3.0. That is not to say that it wont work with other flavours of VB - only that I havnt been able to test any others. If you wish to try it - back up your data first!

- A GBLIB2-branded file works identically under Windows. The registration details are encrypted, and therefore difficult to alter/view.   File-size makes no difference. (I have branded a 2MB VB EXE with no ill-effects)

4) Can a file be unbranded?

- No. But it can be re-branded as many times as you wish.   The only way to recover a virgin copy of the EXE is to recompile it with the source code in VB V3.0.

5) Have you written any other FREEWARE?

- Try GBLIB1a.DLL, and SYSCOLOR.BAS. (Available on Compuserve and in VBTIPS)

See.. Getting Started

New Users Guide to GBLIB2.EXE

-
-
-

- GBLIB2.EXE has been designed to be powerful enough for the sophisticated user, yet simple to use.

- I suggest you look at the Overview , then the Getting Started topics.

Overview of GBLIB2

Getting Started

Getting Started

-
-
-

You will need to study the REGMAX.EXE source code using Microsoft(tm) Visual Basic V3.0 and the project REGMAX.MAK.

Program Structure for Normal Use

- (Optional) Use the applications installation program to get the registration information from the user, and write it to the programs INI file.

- Before the main form loads, call ReadRegData . Test USERNAME for the string UNLICENSED. If it is, then the program is unbranded.
Either Pull the information from the INI file to brand the applications EXE, using WriteRegData . (The user need not be aware that this is happening). Display the registration information from the EXE using ReadRegData .
Or Use a form like REGFORM to get the information from the user, and brand the EXE then.

- Thereafter, when the main form loads, it will pick up the registration information automatically. If you have used an INI file, you can compare it with the INI file information to check its integrity.

Other uses and ideas.

- Read/WriteRegData can be used to store any kind if string information you like.

- A high-score and High-Scorer could be stored.

- A pointer (provided it is in string form) to an array or table containing progress so far through a sequential program. (Such as an adventure game)

- Embedded copyright information. (Like Version Information in resource tables)

- Remember:

String area 1 is 41 characters maximum.
String area 2 is 34 characters maximum.

Use your imagination!

Fully enabling GBLIB2 for use.

-
-
-

-

Using APPFORM.TXT (supplied with this archive) you can aquire your own personal code that will enable you to use GBLIB2.EXE to brand any information you like.

The security of the code thus obtained is then solely **your responsibility**.

To qualify, you must be a bona fide Visual Basic(tm) V3.0 developer, who intends to use GBLIB2 in a responsible way.

In the spirit of **FREEWARE**, you cannot be charged for this service.
