GBLIB1.DLL - A utility library for Visual Basic Programmers

By Gordon Bamber (c)1994

Exit Help (or ALT+x)

Index of all Subs and Functions in GBLIB1.DLL

Back

Exit Help (or ALT+x)

Click the sub or function name to obtain more information

See... Functions By Category

About GBLIB1.DLL

Back

Exit Help (or ALT+x)

GBLIB1.DLL (c)1994 was written in Borland Turbo Pascal for Windows(tm) V1.5 by The Author .

It is released as FREEWARE on the following conditions:

1) Copyright remains with the author.
2) Commercial use is prohibited without the authors written 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 GBLIB1.DLL is not freeware.

Function PlayResource

Back
Exit Help (or ALT+x)

Purpose

There are a number of WAVES already embedded in GBLIB1.DLL.. They can be called by their own subroutines (ie. Sub Click) or using this function.

The embedded sounds are:

GORDON	Me saying "Hello"
MARTS	A collegue (Martin Pointer) saying "Hello"
CLICK	A sound like a camera shutter
CLIK	A shorter and harder version of CLICK (Click with a flick-knife)
DONE	A man saying Done Processing

GBLIB.DLL Pascal Prototype

Function PlayResource(lpname:LPSTR):INT

Help Declare and Use

```
RegisterRoutine("gblib1.dll","PlayResource","i=S")
PlayResource(DONE)
```

Visual Basic Declare

VB

```
Declare Function PlayResource Lib "gblib1.dll" (ByVal szResName As String) as Integer
```

Visual Basic Example

VB

```
i_Retval = PlayResource (CLICK)
```

See... PlaySound SpeakerBeep

Sub About

Back
Exit Help (or ALT+x)

CLICK ME for an example

Purpose

An about screen for GBLIB.DLL. You have no compulsion to include this in your application if it uses GBLIB.DLL, but I would consider you a complete asshole if you didnt.

GBLIB.DLL Pascal Prototype

procedure About(hWind:UINT)

Help Declare and Use

```
RegisterRoutine("gblib1.dll","AboutHelp","v=")
AboutHelp()
```

Visual Basic Declare

VB

```
Declare Sub About Lib "gblib1.dll" (ByVal i_HWnd As Integer)
```

Visual Basic Example

VB

```
About Me.hWnd
```

See... About GBLIB1 LoadDLLDialog

Sub Gordon

Back
Exit Help (or ALT+x)

Purpose

Me saying "Hello".

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure Gordon

Help Declare and Use

RegisterRoutine("gblib1.dll","Gordon","v=")
Gordon()

Visual Basic Declare

VB

Declare Sub Gordon Lib "gblib1.dll" ()

Visual Basic Example

VB
Gordon

See... About

Sub Marts

Back
Exit Help (or ALT+x)

Purpose

Martin Pointer saying "Hello". (Who the heck is Martin Pointer?)

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure Marts

Help Declare and Use

RegisterRoutine("gblib1.dll","Marts","v=")
Marts()

Visual Basic Declare

VB
Declare Sub Marts Lib "gblib1.dll" ()

Visual Basic Example

VB
Marts

See... PlayResource

Sub StopLoop

Back
Exit Help (or ALT+x)

Purpose

Plays a NULL sound, which has the effect of stopping any sound started by calling the PlayLoop sub. No sound is produced by StopLoop itself.

CLICK ME to start a looped sound as an example.
CLICK ME to stop looping

GBLIB.DLL Pascal Prototype

Procedure StopLoop

Help Declare and Use

```
RegisterRoutine("gblib1.dll","StopLoop","v=")
StopLoop()
```

Visual Basic Declare

VB

```
Declare Sub StopLoop Lib "gblib1.dll" ()
```

Visual Basic Example

Back

StopLoop

See... PlayLoop PlaySound

Sub PlayLoop

Back

Back (or ALT+x)

Purpose

Back

Plays the specified WAVE repeatedly until another wave is played, or StopLoop is called.

CLICK ME to start a looped sound as an example.
CLICK ME to stop looping

GBLIB.DLL Pascal Prototype

Procedure PlayLoop(sz_WAVFile:LPSTR)

Help Declare and Use

RegisterRoutine("gblib1.dll","PlayLoop","i=S")
PlayLoop(c:\windows\snore.wav)

Visual Basic Declare

Back

Declare Sub PlayLoop Lib "gblib1.dll" (ByVal szWAVEFile as string)

Visual Basic Example

Back

PlayLoop "c:\windows\snore.wav"

See... StopLoop PlaySound

Sub Clik

Back

Back (or ALT+x)

Purpose

Back

Plays a short click sound. - Good for menu selections.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure Clik

Help Declare and Use

```
RegisterRoutine("gblib1.dll","Clik","v=")
Clik()
```

Visual Basic Declare

Back

Declare Sub Clik Lib "gblib1.dll" ()

Visual Basic Example

Back

Clik

See... SpeakerBeep PlaySound Click

Sub Click

Back

Back (or ALT+x)

Purpose

Back

Plays a camera shutter click sound. - Good for command buttons.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure Click

Help Declare and Use

RegisterRoutine("gblib1.dll","Click","v=")
Click()

Visual Basic Declare

Back

Declare Sub Click Lib "gblib1.dll" ()

Visual Basic Example

Back

Click

See... Clik SpeakerBeep PlaySound

Sub Done

Back

Back (or ALT+x)

Purpose

Back

Plays the sound of a man saying "Done Processing" I like this sound. I would love to hear the same person saying something like "Please can I go to the toilet? " or "Hi - pleased to meet you - havnt I got a silly voice? "

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure Done

Help Declare and Use

RegisterRoutine("gblib1.dll","Done","v=")
Done()

Visual Basic Declare

Back

Declare Sub Done Lib "gblib1.dll" ()

Visual Basic Example

Back

Done

See... PlayResource

Sub SystemStart

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
SystemStart=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemStart

Help Declare and Use

RegisterRoutine("gblib1.dll","SystemStart","v=")
SystemStart()

Visual Basic Declare

Back

Declare Sub SystemStart Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemStart

See... SystemEnd SystemBeep SystemAsterisk
SystemHand SystemQuestion SystemExclamation
MouseClick ProgramLaunch PlaySound

Sub SystemEnd

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI file under
[Sounds]
SystemEnd=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemEnd

Help Declare and Use

RegisterRoutine("gblib1.dll","SystemEnd","v=")
SystemEnd()

Visual Basic Declare

Back

Declare Sub SystemEnd Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemEnd

See... SystemStart SystemBeep SystemAsterisk
SystemHand SystemQuestion SystemExclamation
MouseClick ProgramLaunch PlaySound

Sub SystemBeep

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
SystemBeep=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemBeep

Help Declare and Use

```
RegisterRoutine("gblib1.dll","SystemBeep","v=")
SystemBeep()
```

Visual Basic Declare

Back

Declare Sub SystemBeep Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemBeep

See... SystemEnd, SystemStart, SystemAsterisk, SystemHand, SystemQuestion, SystemExclamation, MouseClick, ProgramLaunch, PlaySound

---

Sub SystemQuestion

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
SystemQuestion=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemQuestion

Help Declare and Use

RegisterRoutine("gblib1.dll","SystemQuestion","v=")
SystemQuestion()

Visual Basic Declare

Back

Declare Sub SystemQuestion Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemQuestion

See... SystemEnd SystemStart SystemAsterisk
SystemHand SystemBeep SystemExclamation
MouseClick ProgramLaunch PlaySound

Sub SystemExclamation

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
SystemExclamation=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemExclamation

Help Declare and Use

RegisterRoutine("gblib1.dll","SystemExclamation","v=")
SystemExclamation()

Visual Basic Declare

Back

Declare Sub SystemExclamation Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemExclamation

See... SystemEnd SystemStart SystemAsterisk
SystemHand SystemBeep SystemQuestion
MouseClick ProgramLaunch PlaySound

Sub SystemAsterisk

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
SystemAsterisk=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemAsterisk

Help Declare and Use

RegisterRoutine("gblib1.dll","SystemAsterisk","v=")
SystemAsterisk()

Visual Basic Declare

Back

Declare Sub SystemAsterisk Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemAsterisk

See... SystemEnd SystemStart SystemExclamation
SystemHand SystemBeep SystemQuestion
MouseClick ProgramLaunch PlaySound

Sub SystemHand

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
SystemHand=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SystemHand

Help Declare and Use

RegisterRoutine("gblib1.dll","SystemHand","v=")
SystemHand()

Visual Basic Declare

Back

Declare Sub SystemHand Lib "gblib1.dll" ()

Visual Basic Example

Back

SystemHand

See... SystemEnd SystemStart SystemExclamation
SystemAsterisk SystemBeep SystemQuestion
MouseClick ProgramLaunch PlaySound

Sub MouseClick

Back

Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
MouseClick=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure MouseClick

Help Declare and Use

```
RegisterRoutine("gblib1.dll","MouseClick","v=")
MouseClick()
```

Visual Basic Declare

Back

Declare Sub MouseClick Lib "gblib1.dll" ()

Visual Basic Example

Back

MouseClick

See... SystemEnd SystemStart SystemExclamation
SystemAsterisk SubSystemBeep SystemQuestion
SystemHand SubProgramLaunch PlaySound

Sub ProgramLaunch

Back
Back (or ALT+x)

Purpose

Back

Plays the sound specified in your WIN.INI under
[Sounds]
ProgramLaunch=

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure ProgramLaunch

Help Declare and Use

RegisterRoutine("gblib1.dll","ProgramLaunch","v=")
ProgramLaunch()

Visual Basic Declare

Back

Declare Sub ProgramLaunch Lib "gblib1.dll" ()

Visual Basic Example

Back

ProgramLaunch

See... SystemEnd SystemStart SystemExclamation
SystemAsterisk SubSystemBeep SystemQuestion
SystemHand MouseClick PlaySound

Sub PlaySound

Back

Back (or ALT+x)

Purpose

Back

Plays a WAV file from disk.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure PlaySound(sz_WAVFile:LPSTR)

Help Declare and Use

RegisterRoutine("gblib1.dll","PlaySound","v=S")
PlaySound(c:\windows\tada.wav)

Visual Basic Declare

Back

Declare Sub PlaySound Lib "gblib1.dll" (ByVal sz_WAVFile as String)

Visual Basic Example

Back

PlaySound "c:\windows\sparkle.wav"

See... SystemEnd SystemStart SystemExclamation
SystemAsterisk SystemBeep SystemQuestion
SystemHand MouseClick ProgramLaunch

Function PlayDLLWave

Back

Back (or ALT+x)

Purpose

Back

This function will play a user resource of type WAVE

Returns Zero=Success, NonZero=Failure

GBLIB.DLL Pascal Prototype

Function PlayDLLWave (sz_Filename, sz_ResName :LPSTR) :INT

Visual Basic Declare

Back

Declare Function PlayDLLWave Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal sz_WAVEName As String) As Integer

Visual Basic Example

Back

```
Dim i_Retval as integer

Rem     //The sound is in vbres.dll and the resource is named WAVE_1//
i_Retval = PlayDLLWave("c:\windows\vbres.dll", "WAVE_1")

Rem     //If successful, the return value is zero//
If i_Retval Then
    MsgBox "Bad return from PlayDLLWave", 16, "ERROR"
    Exit Sub
End If
```

See How to add a non-standard resource to a DLL

Function LoadDLLBitmap

Back
Back (or ALT+x)

Purpose

Back
This function will display a Bitmap from a DLL resource into a display hWnd.

If Stretch=0 then no resizing
If Stretch=1 then the bitmap resizes to fit the hWnd

Returns

Zero=Success, NonZero=Failure

GBLIB.DLL Pascal Prototype

Function LoadDLLBitmap (sz_Resname, sz_Filename :LPSTR; hWind :UINT; Stretch :INT) :INT

Visual Basic Declare

Back

```
Declare Function LoadDLLBitmap Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal
sz_BitMapName As String, ByVal hWnd As Integer, ByVal Stretch As Integer) As Integer
```

Visual Basic Example

Back

```
Rem     //Loading into the Picture Box called pic1//
Call LoadAPicture pic1, "BITMAP_1", 0
Rem     //If the last parameter was 1, then it would AutoSize to Pic1//
```

Which Calls...

```
Sub LoadAPicture (c As Control, sz As String, bstretch As Integer)

Dim i_Retval As Integer
Dim hwind As Integer

Rem     //Get the controls hWnd property//
hwind = c.hWnd

i_Retval = LoadDLLBitmap("c:\windows\vbres.dll", sz, hwind, bstretch)

Rem     //Test for failure//
If i_Retval Then
    MsgBox "Returned an error", 16, "LoadDLLBitmap Function"
    Exit Sub
```

```
End If

End Sub
```

See... LoadDLLBitmapFrom LoadDLLBitmapTo

Function GetDLLText

Back
Back (or ALT+x)

GBLIB.DLL Pascal Prototype

Function GetDLLText (sz_Filename, sz_Resname, sz_Text :LPSTR; Var i_Len :UINT) :INT

Purpose

Back

This function will load a user resource of type TEXT
The text resource must end in ASCII 0

Returns

SZTEXT as a String/LPSTR
SZLEN as the length of szText
Zero=Success, NonZero=Failure

Visual Basic Declare

Back

Declare Function GetDLLText Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal sz_TextName As String, ByVal lpText As String, iLen As Integer) As Integer

Visual Basic Example

Global Declaration
Back

```
Rem     //The function requires a fixed length string//
Global lpSzr As String * 512
```

Typical Use in an event Sub
Back

```
Dim msg As String

Rem     //Remember the DLL text could include embedded carriage returns//

msg = VBSetTextFromDLL("DLLNAME") & Chr$(13)
msg = msg & "Version " & VBSetTextFromDLL("VERSION") & Chr$(13)
msg = msg & VBSetTextFromDLL("COPYRIGHT")
lbl_OutPut.Caption = msg
```

Visual Basic Wrapper Function
Back

```
Function VBSetTextFromDLL (sz As String) As String
```

```
Dim i_Retval As Integer
Dim szlen As Integer

Rem     //Initialise lpSzr to ASCII Zeros//
szlen = 512
lpSzr = String$(512, 0)

i_Retval = GetDLLText("c:\windows\vbres.dll", sz, lpSzr, szlen)
If i_Retval Then
     MsgBox "Bad return from GetDLLText", 16, "ERROR"
     VBSetTextFromDLL = "unknown"
     Exit Function
End If

Rem     //The DLL Text MUST end in an ASCII Zero //
VBSetTextFromDLL = Left$(lpSzr, szlen)

End Function
```

See How to add a non-standard resource to a DLL

Function GetDLLBitmapSize

Back

Back (or ALT+x)

Purpose

Back

This function will interrogate a Bitmap resource. Used in conjunction with LoadDLLBitmap , for instance, you could resize the VB control to the bitmaps size before loading it in, and thus use a generic picturebox control to display a series of bitmaps.

Returns

TRECT Structure contains size:
TRECT.Right = Width,
TRECT.Bottom = height
Zero=Success, NonZero=Failure

GBLIB.DLL Pascal Prototype

Function GetDLLBitmapSize (sz_Filename, sz_Resname :LPSTR ;Var MyRect :TRECT) :INT

Visual Basic Declare

Back

```
Rem     //Declare TRECT before using it in the Function Declare//
Type TRECT
      Left As Integer
      Top As Integer
      Right As Integer
      Bottom As Integer
End Type

Declare Function GetDLLBitmapSize Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal lpname
As String, MyRECT As TRECT) As Integer
```

Visual Basic Example

Back

```
Dim msg As String
Dim ARECT as TRECT

rem      //TRECT must first be declared as a TYPE prototype//
i_Retval = GetDLLBitmapSize("c:\windows\vbres.dll", "BITMAP_1", ARECT)

If i_Retval Then
     MsgBox "Bad return from GetDLLBitmapSize", 16, "ERROR"
     Exit Sub
End If
```

```
rem      //The return values are always in pixels//
msg = Format$(ARECT.Right)
msg = msg & " Pixels x "
msg = msg & Format$(ARECT.Bottom)
msg = msg & " Pixels"

lbl_OutPut.Caption = msg
```

See... LoadDLLBitmap LoadDLLBitmapFrom LoadDLLBitmapTo

Function LoadDLLBitmapTo

Back

Back (or ALT+x)

Purpose

Back

This function will display a Bitmap from a DLL resource into a display hWnd.

Left, Top, Right and Bottom place the bitmap RELATIVE TO the Top and Left of the display Window

Returns

Zero=Success, NonZero=Failure

GBLIB.DLL Pascal Prototype

Function LoadDLLBitmapTo (sz_Filename, sz_Resname: LPSTR; hWind: UINT; Left, Top, Right, Bottom :UINT) :INT

Visual Basic Declare

Back

```
Declare Function LoadDLLBitmapTo Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal
sz_BitMapName As String, ByVal hWnd As Integer, ByVal i_Left As Integer, ByVal i_Top As Integer,
ByVal i_Right As Integer, ByVal i_Bottom As Integer,) As Integer
```

Visual Basic Example

Back

```
i_Retval = LoadDLLBitmapTo("c:\windows\vbres.dll", MYPICTURE, PictureBox.hWnd,
10,10,200,200)

Rem     //Test for failure//
If i_Retval Then
     MsgBox "Returned an error", 16, "LoadDLLBitmapTo Function"
     Exit Sub
End If

End Sub
```

See... LoadDLLBitmap LoadDLLBitmapFrom

Calling Functions by Index

This Function or Sub can be called by this Index number instead of by name.

Overview of GBLIB1.DLL

Back

Back (or ALT+x)

Aim and Purpose of GBLIB1.DLL

Back

GBLIB1.DLL was written as a companion to Visual Basic(tm) V3.0.
There were 2 criteria for including a function or subroutine in the library.

Either: **1)** Visual Basic was unable to do it.
or **2)** It saved tedious and/or complicated API declarations.

In category (1) are the SubClassing and Cursor functions.
In Category (2) are the Resource and Sound functions.

- As much effort went into this help file GBLIB1.HLP as the DLL itself. This is because a DLL library is only useful if the functions are understood in their syntax and use.

- Each Subroutine or Function has an explanation of it's use, the original Pascal Prototype (so that it can be used with other languages) the Visual Basic Declaration, and, where appropriate, an example of how it can be used in VB, and/or an HPJ declaration.

- Some of the sound functions have a live demo (CLICK ME) in this help file.

There were two important aims in writing the code:

- All routines should be 'system-resource-friendly', in that they should return any resources they consume. (eg DCs, Handles, Objects)
- All routines should be as robust as possible, and capable of detecting their own errors.

- Most routines check the parameters passed to them, and will return cleanly if they are bad, usually with a Message Box outlining the problem, and any resources returned.

- Most functions return Zero for success, and 1 for failure. (except those which retrieve a value, of course)

- GBLIB1.DLL and associated help file are distributed as FREEWARE, under conditions outlined in the help file (See About)

- The DLL was written in Borland Turbo Pascal for Windows(tm) V1.5 and this
help file was written using Microsoft Word for Windows(tm) V6.0

Visual Basic and Binary Resources

Back

One of the few failings of Visual Basic(tm) is its non-standard use of Windows resources.

VB resources are held in the FRX file, which is non-standard, and then bound into the EXE file at compilation in a non-standard way.

●
If your application uses many Bitmaps and Sounds, you have two choices:

1) Compile them into the EXE file.
2) Use LoadPicture to load them in from disk.

Method (1) can result in an EXE file that will be unacceptably large to compile, and whats more, the resources cannot be edited or changed without re-compilation.

Method (2) leaves your work subject to piracy or alteration. Many bitmaps and waves can make distribution of your application lengthy and expensive on disks.

GBLIB1.DLL tries to overcome these problems.

●
Using a resource editor, (ie. Resource Workshop(tm) from Borland) all Bitmap, Cursor,Icon, Wave and Text items are bound into one or more DLL resource files.

These files are then used by your application (in association with the GBLIB1.DLL functions) to retrieve the resources when needed.

●
An added bonus is that the resources remain secure, in that they can only be altered or removed by a knowlegable user of a Resource Editor.

VB
Resources loaded from a DLL load MUCH faster than from disk, so flickbook animation with bitmaps is practical in VB at last!

VB
Another boon is the ability to load CURSORS and ICONS via resources - loading persistent cursors is very difficult in VB.

In addition, GBLIB1.DLL contains functions which eliminate tedious API/DLL programming in VB. Functions like the SoundCard querying ones make life easier for the professional VB programmer who has to produce distributable VB applications.

VB
Each function in GBLIB.DLL is documented where appropriate, with examples in the Function Reference included in this helpfile.

Gordon Bamber

Gordon Bamber is currently a staff programmer with Maxim Training - a company specialising in Multimedia CBT.

Gordon can currently(08/94) be reached

Email:
gbamber@cix.compulink.uk.com
74437.672@compuserve.com

Snailmail:
11, Helena road,
Brighton, E.Sussex, England.
BN2 6BS

Maxim Training can be reached at
74437.672@compuserve.com
57 Ship Street,
Brighton,
E.Sussex.
England.

A Complete Asshole

Defined as: An individual who uses or alters my software without acknowledging or crediting me in any way.

WAVE Files

MCI RIFF files having the extension WAV
You must have a sound output device attached to your PC in order to play these.
If such a sound device is absent, Windows 3.1(tm) will substitute a default speaker beep for the sound.

System Sounds

In Windows 3.1(tm) sounds can be specified as system sounds in Control Panel(tm). These are stored in you WIN.INI file as:

```
[Sounds]
SystemStart=tada.wav
SystemEnd=terminat.wav
etc.
```

You can, of course, edit these entries manually via notepad, and even add your own eg

```
MouseClick=click.wav
ProgramLaunch=sparkle.wav
etc.
```

[Sounds]
Entry=

Dont worry if your WIN.INI does not contain this entry.
Use SYSEDIT (or your application code) to put it in. The Subroutine will then find it OK.

If it is absent, then a default System Beep will be played instead

ASCII 0

When you add a user resource and define it of type TEXT , Resource Workshop does not add an ASCII Zero (Binary Zero) to the end of the data.
You must edit the raw data yourself, and manually add &H00 as the last character in your text.

The function GetDLLText will retrieve all text up to (not including) this ASCII zero. If you are getting unwanted extra characters, then the ASCII 0 is missing.

How to add a non-standard resource to a DLL

VB
VB (or ALT+x)

VB

WAV files and Text files using Resource Workshop (tm)Borland

1) Select **File/Open Project**, and open a DLL (It can be an empty one with no functions, but it must be a proper Windows DLL file)

2) Select **File/Add to Project**, and select Filetype as **USER RESOURCE DATA**.

3) Select a WAV file or an unformatted Text file from disk, and load it.

4) A dialog box titled **Custom Resource Type** will appear. Select the **New Type** button.

5) Type in **WAVE** or **TEXT** as the type (use Uppercase)

6) Rename the resource as necessary.

7) If **TEXT**, edit the raw data so that it ends with ASCII 0

Thats it! Save the project as a DLL, and the GBLIB1 functions can extract the **WAVE** and **TEXT** resources as needed.

LoadDLLBitmapFrom

VB
VB (or ALT+x)

Purpose

VB
This function will display a whole or a part of a Bitmap from a DLL resource into a display hWnd.

Left, Top, Right and Bottom defines the area of the Bitmap that you want to display. It is stretched to fit the Hwnd

Returns

Zero=Success, NonZero=Failure

GBLIB.DLL Pascal Prototype

Function LoadDLLBitmapFrom (sz_Filename, sz_Resname: LPSTR; hWind: UINT; Left, Top, Right, Bottom :UINT) :INT

Visual Basic Declare

VB

```
Declare Function LoadDLLBitmapFrom Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal sz_BitMapResName As String, ByVal hWnd As Integer, ByVal i_Left As Integer, ByVal i_Top As Integer, ByVal i_Right As Integer, ByVal i_Bottom As Integer,) As Integer
```

Visual Basic Example

VB

```
i_Retval = LoadDLLBitmapFrom("c:\windows\MyBMPS.dll", TOOLBITMAP, PictureBox.hWnd, 64,0,64,64)

Rem     //Test for failure//
If i_Retval Then
    MsgBox "Returned an error", 16, "LoadDLLBitmapFrom Function"
    Exit Sub
End If

End Sub
```

See... LoadDLLBitmap LoadDLLBitmapTo

LoadDLLDialog

VB
VB (or ALT+x)

Purpose

VB
Loads a Dialog resource. The dialog must be a simple AboutBox type of dialog only. Pass the form hWnd to the function.

GBLIB.DLL Pascal Prototype

Function LoadDLLDialog (sz_Filename, sz_Resname :LPSTR; hWind :UINT) :INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function LoadDLLDialog Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal
sz_DialogResName As String, ByVal hWnd As Integer) As Integer
```

Visual Basic Example

VB

```
i_RetVal = LoadDLLDialog(RESFILE.DLL,ABOUTDLG,Me.hWnd)
Rem     //Test for failure//
If i_Retval Then
        MsgBox "Returned an error", 16, "LoadDLLDialog Function"
        Exit Sub
End If
```

See... About

LoadDLLIcon

VB
VB (or ALT+x)

Purpose

VB
Loads an Icon resource into the passed hWnd. The icon is placed at the Top and Left of the hwnd, and is not resized.

GBLIB.DLL Pascal Prototype

Function LoadDLLIcon (sz_Filename, sz_Resname :LPSTR; hWind :UINT) :INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function LoadDLLIcon Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal
sz_IconResName As String, ByVal hWnd As Integer) As Integer
```

Visual Basic Example

VB

```
i_RetVal = LoadDLLIcon(RESFILE.DLL,PROGICON,Me.hWnd)
Rem     //Test for failure//
If i_Retval Then
        MsgBox "Returned an error", 16, "LoadDLLIcon Function"
        Exit Sub
End If
```

See... LoadDLLBitmap GetDLLText LoadDLLDialog LoadDLLCursor

LoadDLLCursor

VB
VB (or ALT+x)

Purpose

VB
Loads a Cursor resource into the passed hWnd. The effect lasts until the mouse is moved outside the boundries of the class to which the hWnd belongs.

GBLIB.DLL Pascal Prototype

Function LoadDLLCursor (sz_Filename, sz_Resname :LPSTR;; hWind :UINT) :INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function LoadDLLCursor Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal
sz_CursorResName As String, ByVal hWnd As Integer) As Integer
```

Visual Basic Example

VB

```
i_RetVal = LoadDLLCursor(RESFILE.DLL,HANDPTR,Me.hWnd)
Rem     //Test for failure//
If i_Retval Then
        MsgBox "Returned an error", 16, "LoadDLLCursor Function"
        Exit Sub
End If
```

See... LoadDLLBitmap GetDLLText LoadDLLDialog LoadDLLIcon

DestroyDLLCursor

VB
VB (or ALT+x)

Purpose

VB
Destroys a cursor set previously by LoadDLLCursor()

GBLIB.DLL Pascal Prototype

Procedure DestroyDLLCursor

Visual Basic Declare

VB

```
Declare Sub DestroyDLLCursor Lib "gblib1.dll" ()
```

Visual Basic Example

VB

```
DestroyDLLCursor
```

See... LoadDLLCursor

SetArrow

VB
VB (or ALT+x)

Purpose

VB
Sets the cursor for the passed hWnd to the default arrow.

GBLIB.DLL Pascal Prototype

Function SetArrow(hWind:UINT):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

Declare Function SetArrow "gblib1.dll" (ByVal hWnd As Integer) As Integer

Visual Basic Example

VB

SetArrow

See... LoadDLLCursor

GBSetCursorPos

VB
VB (or ALT+x)

Purpose

VB
To set the cursor to a position relative to the Top and Left of the passed hWnd in pixels. The source hWnd need not have Scalemode=3

GBLIB.DLL Pascal Prototype

Function GBSetCursorPos(hWind:UINT;NewX,NewY:INT):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function GBSetCursorPos Lib "gblib1.dll" (ByVal hWind As Integer, ByVal XPos As Integer, ByVal YPos As Integer) As Integer
```

Visual Basic Example

VB

```
i_Retval = GBSetCursorPos Me.hWnd, 10, 10

If i_Retval Then
    MsgBox "Bad return from GBSetCursorPos ", 16, "ERROR"
    Exit Sub
End If
```

GBClientToScreenRECT

VB
VB (or ALT+x)

Purpose

VB
Converts client co-ordinates into screen co-ordinates.
N.B. - Use VBs built-in Screen.TwipsperpixelX and Screen.TwipsperpixelY for forms with ScaleMode=0.

GBLIB.DLL Pascal Prototype

```
Procedure GBClientToScreenRECT(hWind:UINT;Var MyRect:TRECT)
```

Visual Basic Declare

VB

```
Type TRECT
      Left As Integer
      Top As Integer
      Right As Integer
      Bottom As Integer
End Type

Declare Sub GBClientToScreenRECT Lib "gblib1.dll" (ByVal hWind As Integer, ScreenRECT As
TRECT)
```

Visual Basic Example

VB

```
Dim MyRect as TRECT

Myrect.Left = Me.Left
MyRect.Top = Me.Top
MyRect.Right = Me.Width
MyRect.Bottom = Me.Height

GBClientToScreenRECT Me.hWnd, MyRect

InvertRect Me.hWnd, MyRect
```

GBDecompress

VB
VB (or ALT+x)

Purpose

VB
Acts like the Microsoft(tm) DOS program EXPAND.EXE(c). Enables a VB program to uncompress files on demand with just one call.

VB
A program could store, say, version-specific data files, and decompress the appropriate ones as needed after installation.

VB
Sadly, there is no ability to compress files without COMPRESS.EXE(c) (A DOS program)

GBLIB.DLL Pascal Prototype

Function GBDecompress(sz_InFilePath,sz_OutFilePath:LPSTR):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function GBDecompress Lib "gblib1.dll" (ByVal InfilePath As String, ByVal OutfilePath As String) As Integer
```

Visual Basic Example

VB

```
i_Retval = GBDecompress (c:\install\readme.tx_, c:\app\readme.txt)

If i_Retval Then
    MsgBox "Bad return from GBDecompress", 16, "ERROR"
    Exit Sub
End If
```

GBPutOnTop

VB
VB (or ALT+x)

Purpose

VB
Pass a forms hWnd to this function, and it will stay on top of other windows, even when it does not have the focus.

GBLIB.DLL Pascal Prototype

Procedure GBPutOnTop(hWind:UINT)

Visual Basic Declare

VB

Declare Sub GBPutOnTop Lib "gblib1.dll" (ByVal hWind As Integer)

Visual Basic Example

VB

GBPutOnTop Me.hWnd

GBNotOnTop

VB
VB (or ALT+x)

Purpose

VB
This will undo a GBPutOnTop call.

GBLIB.DLL Pascal Prototype

Procedure GBNotOnTop(hWind:UINT)

Visual Basic Declare

VB

Declare Sub GBNotOnTop Lib "gblib1.dll" (ByVal hWind As Integer)

Visual Basic Example

VB

GBNotOnTop Me.hWnd

ModalCalc

VB
VB (or ALT+x)

Purpose

VB
Puts up the Windows Calculator on top of all other windows until it is closed by the user.

VB
If Calculator is already running, it will restore the running version to the top.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Function ModalCalc:INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function ModalCalc Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
i_RetVal = ModalCalc

If i_Retval Then
    MsgBox "Bad return from ModalCalc", 16, "ERROR"
    Exit Sub
End If
```

See... ModalNotePad ModalNotePadExec

ModalNotePad

VB
VB (or ALT+x)

Purpose

VB
Runs Windows Notepad on top of all other windows, and keeps on top until the user shuts it down.
If Notepad is already running, it is brought to the top.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Function ModalNotePad:INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function ModalNotePad Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
i_Retval = ModalNotePad()

If i_Retval Then
    MsgBox "Bad return from ModalNotePad", 16, "ERROR"
    Exit Sub
End If
```

See... ModalCalc ModalNotePadExec

ModalNotePadExec

VB
VB (or ALT+x)

Purpose

VB
Runs Windows Notepad with a specified file on top of all other windows, and keeps on top until the user shuts it down.
If Notepad is already running, it is brought to the top.
ModalNotePadExec could be used to show README files upon successful installation of your application.

GBLIB.DLL Pascal Prototype

Function ModalNotePadExec(sz_Txtname:LPSTR):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function ModalNotePadExec Lib "gblib1.dll" (ByVal szFilePath As String) As Integer
```

Visual Basic Example

VB

```
i_RetVal = ModalNotePadExec ( c:\appdir\readme.txt )

If i_Retval Then
    MsgBox "Bad return from ModalNotePadExec", 16, "ERROR"
    Exit Sub
End If
```

See... ModalCalc ModalNotePad

MakeUAE

VB
VB (or ALT+x)

Purpose

VB
This is a pointless and tasteless joke routine. It simulates a General Protection Fault, but unlike a real one, returns without harm. Someone with a purile sense of humour must have written this one.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure MakeUAE

Visual Basic Declare

VB

```
Declare Sub MakeUAE Lib "gblib1.dll" ()
```

Visual Basic Example

VB

```
MakeUAE
MsgBox Hah! I was only fooling!
```

WaitForL

VB
VB (or ALT+x)

Purpose

VB
This is a modal timer, in other words, a PAUSE statement. Code will stop executing, and all user actions are defrerred until the WaitForL returns.
The L stands for Long - the data type that must be passed to this subroutine. The time is in MilliSeconds. As the system clock ticks as 18.2 ticks/second, the shortest time that can be passed is about 55ms.

GBLIB.DLL Pascal Prototype

Procedure WaitForL(I_Millisecs:LONGINT)

Visual Basic Declare

VB

Declare Sub WAITFORL Lib "gblib1.dll" (ByVal MilliSeconds As Long)

Visual Basic Example

VB

```
WaitForL 1000
rem     //Wait a second//
WaitForL 100
rem     //Wait 1/10th of a second//
```

See.. WaitFor

WaitFor

VB
VB (or ALT+x)

Purpose

VB

WaitFor is a Visual Basic PAUSE statement. Pass it the number of seconds that you want to wait. No code will get executed until it returns.

Possible use - A Splash Screen timer.

GBLIB.DLL Pascal Prototype

Procedure WaitFor(i_seconds:INT)

Visual Basic Declare

VB

```
Declare Sub WaitFor Lib "gblib1.dll" (ByVal Seconds As Integer)
```

Visual Basic Example

VB

```
WaitFor 1
rem     //Hold on a second!//

Splashfrm.Show
rem     //Make sure that the user cannot escape from seeing it//
WaitFor 5
Unload Splashfrm
```

See.. WaitForL

SpeakerBeep

VB
VB (or ALT+x)

Purpose

VB
Sounds the system built-in speaker. If the user has a soundcard, it will be ignored, and the speaker will be used.
The user may have the default mci beep turned off, but a SpeakerBeep will still make a sound.

CLICK ME for an example.

GBLIB.DLL Pascal Prototype

Procedure SpeakerBeep

Visual Basic Declare

VB

```
Declare Sub SpeakerBeep Lib "gblib1.dll" ()
```

Visual Basic Example

VB

```
SpeakerBeep
Msgbox An error has occurred,16,That woke you up!
```

See.. PlaySound WillPlay Click Clik

StartWait

VB
VB (or ALT+x)

Purpose

VB
Together with the WAITONE and STOPWAIT procedures, StartWait implements a CLOCK Cursor that will animate 12-o-clock thru 12-o-clock with each call to WaitOne.
Used instead of the hourglass cursor to show the passing of time during a lengthy process.

GBLIB.DLL Pascal Prototype

Procedure StartWait(hWind:UINT)

Visual Basic Declare

VB

```
Declare Sub StartWait Lib "gblib1.dll" (ByVal hWind As Integer)
```

Visual Basic Example

VB

```
StartWait Me
rem     //Initialise clockcursor//

Call DoCalc
StopWait
rem     //Return to normal cursor, and free-up resources//

Sub DoCalc
For x = 1 to aLot
        rem     //Twizzle the clock another hour//
        WaitOne
        DoAnotherCalc
Next X
Exit Sub
```

See.. WaitOne StopWait

StopWait

VB
VB (or ALT+x)

Purpose

VB
Called to return the cursor to normal, and clean up resources after using StartWait/WaitOne.

GBLIB.DLL Pascal Prototype

Procedure StopWait

Visual Basic Declare

VB

```
Declare Sub StopWait Lib "gblib1.dll" ()
```

Visual Basic Example

VB

```
StartWait Me
rem     //Initialise clockcursor//

Call DoCalc
StopWait
rem     //Return to normal cursor, and free-up resources//

Sub DoCalc
For x = 1 to aLot
        rem     //Twizzle the clock another hour//
        WaitOne
        DoAnotherCalc
Next X
Exit Sub
```

See.. StartWait WaitOne

WaitOne

VB
VB (or ALT+x)

Purpose

VB

Advances the clock cursor set up in STARTWAIT one hour.

GBLIB.DLL Pascal Prototype

Procedure WaitOne

Visual Basic Declare

VB

```
Declare Sub WaitOne Lib "gblib1.dll" ()
```

Visual Basic Example

VB

```
StartWait Me
rem     //Initialise clockcursor//

Call DoCalc
StopWait
rem     //Return to normal cursor, and free-up resources//

Sub DoCalc
For x = 1 to aLot
        rem     //Twizzle the clock another hour//
        WaitOne
        DoAnotherCalc
Next X
Exit Sub
```

See.. StartWait StopWait

LoadEXEIcon

VB
VB (or ALT+x)

Purpose

VB
Borrows the first Icon it can find in the specified EXE file, and displays it at position 0,0 in the passed hWnd.

GBLIB.DLL Pascal Prototype

Function LoadEXEIcon(hWind:UINT;sz_EXEPath:LPSTR):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function LoadEXEIcon Lib "gblib1.dll" (ByVal hWind As Integer, ByVal szFilename As String) As Integer
```

Visual Basic Example

VB

```
i_RetVal = LoadEXEIcon ( Picture1.hWnd, c:\windows\progman.exe )

If i_Retval Then
    MsgBox "Bad return from LoadEXEIcon", 16, "ERROR"
    Exit Sub
End If
```

See.. LoadEXEIconXY

SubClassIt

VB
VB (or ALT+x)

Purpose

VB
Unfortunately, Visual Basic(tm) does not have a Message Handling loop, as other languages (Pascal, C, C++) have. This means that there are (a very few) messages that cannot be picked up by the VB programmer.
VB
Using SubClassIt, you can detect Non-Client (eg MenuBar) mouseclick events, the Form_Move event, and Memory-Compacting and WM_POWER events.
VB
While the window is subclassed, the right_mousebutton_down event is mapped to the left_mousebutton_down event - that may be useful in itself.

VB
The extra message events are all diverted to the Visual Basic Sub form_mousedown event. The type of message is indicated by the X parameter. (Normally this is the X-Position of the mouse, of course)

VB
In addition, a right mousebutton double-click will trigger the default Windows Screen-Saver.

GBLIB.DLL Pascal Prototype

```
Procedure SubClassIt(hWind:UINT)
```

Visual Basic Declare

VB

```
Declare Sub SubClassIt Lib "gblib1.dll" (ByVal hWind As Integer)
rem     //Use the current form hWnd//
```

Visual Basic Example

VB

```
rem     //Use the current form hWnd//
SubClassIt Me.hWnd

Sub Form_MouseDown (Button As Integer, Shift As Integer, X As Single, Y As Single)
Dim msg As String

If Button = 1 Then
        MsgBox "Left_MouseButton Clicked", 64, "Form_MouseDown"
        Exit Sub
Else
        rem     //If subClassing then X= following values: //
        rem     //0=NC_Left, 1=NC_Right, 2=Form_Move, 3=Compacting, 4=POWER//
        Select Case X
        Case 0
```

```
            msg = "Non-Client Left Button clicked."
        Case 1
            msg = "Non-Client Right Button clicked."
        Case 2
            Dim WX As Integer
            Dim WY As Integer
            GetClientZero Me.hWnd, WX, WY
            msg = "Window moved to Screen Co-Ordinates" & Chr$(10)
            msg = msg & "X=" & Format$(WX) & " Y=" & Format$(WY)
        Case 3
            msg = "Windows is Compacting Memory."
            MsgBox msg, 64, "Subclassing via GBLIB1.DLL"
            UnSubClassIt
        Case Else
            msg = "Right_MouseButton Clicked"
        End Select
        MsgBox msg, 64, "Subclassing via GBLIB1.DLL"
End If
End Sub
```

See... UnSubClassIt

UnSubClassIt

VB
VB (or ALT+x)

Purpose

VB
This routine restores things to nomal after a call to SubClassIt . The Form_MouseDown event will revert to Visual Basic(tm) behaviour.

GBLIB.DLL Pascal Prototype

Procedure UnSubClassIt

Visual Basic Declare

VB

Declare Sub UnSubClassIt Lib "gblib1.dll" ()

Visual Basic Example

VB

UnSubCalssIt

See... SubClassIt

LoadEXEIconXY

VB
VB (or ALT+x)

Purpose

VB
This routine will load the first Icon it finds (usually the application Icon) from the specified EXE file, and display it in the specified hWnd at the co-ordinates X, Y.

GBLIB.DLL Pascal Prototype

Function LoadEXEIconXY(hWind:UINT;sz_EXEPath:LPSTR;X,Y:INT):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function LoadEXEIconXY Lib "gblib1.dll" (ByVal hWind As Integer, ByVal sz_EXEname As String, ByVal Xpos as Integer, ByVal Ypos as Integer) As Integer
```

Visual Basic Example

VB

```
i_Retval = LoadEXEIconXY(picLoadIcon1.hWnd, "C:\WINDOWS\PROGMAN.EXE", 10,10)
If i_Retval Then
    MsgBox "Bad return from LoadEXEIconXY", 16, "ERROR"
    Exit Sub
End If
```

See... LoadEXEIcon LoadDLLIcon

LoadDLLIconXY

VB
VB (or ALT+x)

Purpose

VB
This routine will load an Icon from the resources of the specified DLL into the specified hWnd at the co-ordinates X, Y.

GBLIB.DLL Pascal Prototype

Function LoadDLLIconXY(sz_Filename, sz_Resname :LPSTR;hWind:UINT;X,Y:INT):INT

Returns

Zero=Success, NonZero=Failure

Visual Basic Declare

VB

```
Declare Function LoadDLLIconXY Lib "gblib1.dll" (ByVal sz_DLLFile As String, ByVal
sz_ICO_Resname As String, ByVal hWind As Integer, ByVal i_Left As Integer, ByVal i_Top As
Integer) As Integer
```

Visual Basic Example

VB

```
i_RetVal = LoadDLLIconXY(VBRES.DLL,PROGICON,PictureBox1.hWnd,10,10)

If i_Retval Then
    MsgBox "Bad return from LoadDLLIconXY", 16, "ERROR"
    Exit Sub
End If
```

See... LoadDLLIcon

SetUpJoystick

VB
VB (or ALT+x)

Purpose

VB
Visual Basic provides no built-in Joystick support. Using the MMSYSTEM.DLL mci services, this function sets up a Joystick interface.
VB
If there is no joystick connected, or some other error, a message is displayed to that effect.
VB
Calls to GetJoyPos (In a Timer event) retrieve the joystick position.
VB
UnSetUpJoystick de-initialises the services, and returns all resources.

GBLIB.DLL Pascal Prototype

procedure SetUpJoyStick(hWind:UINT)

Visual Basic Declare

VB

Declare Sub SetUpJoyStick Lib "gblib1.dll" (ByVal hWind As Integer)

Visual Basic Example

VB

SetUpJoyStick Me.hWnd

See... UnSetupJoystick GetJoyPos

UnSetUpJoyStick

VB
VB (or ALT+x)

Purpose

VB
After a call to SetUpJoystick , this routine de-initialises the Joystick handlers and returns all resources.

GBLIB.DLL Pascal Prototype

Procedure UnSetUpJoyStick

Visual Basic Declare

VB

Declare Sub UnSetUpJoystick Lib "gblib1.dll" ()

Visual Basic Example

VB

UnSetUpJoyStick

See... SetupJoystick GetJoyPos

WillPlay...

VB
VB (or ALT+x)

Purpose

VB
A series of functions that return a TRUE/FALSE value. You can use one or more to test a Users hardware before playing WAVE files on their system.
VB
You could have 2 or more versions of the same sound, and, depending on the users hardware, play the best quality that they can support.
VB
Eliminates a lot of tedious VB programming.

GBLIB.DLL Pascal Prototypes

Function WillPlay811Mono:INT
Function WillPlay1611Mono:INT
Function WillPlay811Stereo:INT
Function WillPlay1611Stereo:INT
Function WillPlay822Mono:INT
Function WillPlay1622Mono:INT
Function WillPlay822Stereo:INT
Function WillPlay1622Stereo:INT
Function WillPlay844Mono:INT
Function WillPlay1644Mono:INT
Function WillPlay844Stereo:INT
Function WillPlay1644Stereo:INT

Returns

Zero=False, NonZero=True
N.B. The SoundBlaster AWE32 returns True for them all.

Visual Basic Declares

VB

```
Declare Function WillPlay811Mono Lib "gblib1.dll" () As Integer
Declare Function WillPlay811Stereo Lib "gblib1.dll" () As Integer
Declare Function WillPlay1611Mono Lib "gblib1.dll" () As Integer
Declare Function WillPlay1611Stereo Lib "gblib1.dll" () As Integer

Declare Function WillPlay822Mono Lib "gblib1.dll" () As Integer
Declare Function WillPlay822Stereo Lib "gblib1.dll" () As Integer
Declare Function WillPlay1622Mono Lib "gblib1.dll" () As Integer
Declare Function WillPlay1622Stereo Lib "gblib1.dll" () As Integer

Declare Function WillPlay844Mono Lib "gblib1.dll" () As Integer
Declare Function WillPlay844Stereo Lib "gblib1.dll" () As Integer
Declare Function WillPlay1644Mono Lib "gblib1.dll" () As Integer
Declare Function WillPlay1644Stereo Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Sub PlayThisSound
If WillPlay1644Stereo() > 0 then
        i_RetVal=PlayResource(mysounds.dll,HIFISOUND)
        If i_RetVal then Exit Sub
Else
        i_RetVal=PlayResource(mysounds.dll,LOWFISOUND)
        If i_RetVal then Exit Sub
EndIf
End Sub
```

GetWaveVendorID

VB
VB (or ALT+x)

Purpose

VB

Use to determine the type of soundcard in a machine. Each manufacturer has a unique ID.

GBLIB.DLL Pascal Prototype

Function GetWaveVendorID:INT

Returns

The Vendor ID
N.B. The SoundBlaster AWE32(tm) returns the number 2

Visual Basic Declare

VB

```
Declare Function GetWaveVendorID Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
i_VendorID = GetWaveVendorID()
Select Case i_VendorID
Case....
```

GetWaveProductID


 (or ALT+x)

Purpose

Used to query the soundcard setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetWaveProductID:INT

Returns

The Product ID
N.B. The SoundBlaster AWE32(tm) returns the number 104

Visual Basic Declare

```
Declare Function GetWaveProductID Lib "gblib1.dll" () As Integer
```

Visual Basic Example

```
i_ProductID = GetWaveProductID()
Select Case i_ProductID
Case....
```

GetWaveDriverVersion

VB
VB (or ALT+x)

Purpose

VB
Used to query the soundcard setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetWaveDriverVersion(var Major,Minor:INT):INT

Returns

Zero=Success, NonZero=Failure
N.B. The SoundBlaster AWE32(tm) driver returns the number 1.17

Visual Basic Declare

VB

```
Declare Function GetWaveDriverVersion Lib "gblib1.dll" (Major As Integer, Minor As Integer) As Integer
```

Visual Basic Example

VB

```
i_RetVal = GetWaveDriverVersion(Major, Minor)
If i_RetVal Then
    MsgBox "Error in GetWaveDriverVersion", 48, "GBLIB1.DLL"
    Exit Sub
End If
MsgBox "Driver version is " & Format$(Major) & "." & Format$(Minor)
```

GetWaveProductName

VB
VB (or ALT+x)

Purpose

VB
Used to query the soundcard setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetWaveProductName(sz_Buffer:LPSTR;var i_Len:UINT):INT

Returns

Zero=Success, NonZero=Failure
N.B. The SoundBlaster AWE32(tm) returns the string SB16 WAVE OUT

Visual Basic Declare

VB

```
Declare Function GetWaveProductName Lib "gblib1.dll" (ByVal szBuffer As String, i_BufferLen As Integer) As Integer
```

Visual Basic Example

VB

```
Dim szBuf as String * 255
ALen = GetWaveProductName(szBuf, 255)
msg = Left$(szBuf, ALen)
MsgBox msg, 64, "Sound Card Driver Name"
```

GetWaveNumChannels

VB
VB (or ALT+x)

Purpose

VB
Used to query the soundcard setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetWaveNumChannels:INT

Returns

1 = Mono, 2 = Stereo
N.B. The SoundBlaster AWE32(tm) returns the number 2

Visual Basic Declare

VB

```
Declare Function GetWaveNumChannels Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
msg = "No. of Channels: " & Format$(GetWaveNumChannels())
MsgBox msg, 64, "GBLIB.DLL Report"
```

GetMIDIVendorID

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDIVendorID:INT

Returns

The MIDI Vendor ID number.
N.B. The SoundBlaster AWE32(tm) returns the number 30

Visual Basic Declare

VB

```
Declare Function GetMIDIVendorID Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim nl As String
nl = Chr$(10)
msg = "MIDI Capabilities" & nl
msg = msg & "Vendor   ID: " & Format$(GetMIDIVendorID()) & nl
msg = msg & "Product ID: " & Format$(GetMIDIProductID()) & nl
msg = msg & "No. of Voices: " & Format$(GetMIDIVoices()) & nl
msg = msg & "No. of Notes: " & Format$(GetMIDINotes()) & nl
msg = msg & "No. of Channels: " & Format$(GetMIDINumChannels())
MsgBox msg, 64, "GBLIB.DLL Report"
```

GetMIDIProductID

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDIProductID:INT

Returns

The Product ID
N.B. The SoundBlaster AWE32(tm) returns the number 15

Visual Basic Declare

VB

```
Declare Function GetMIDIProductID Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim nl As String
nl = Chr$(10)
msg = "MIDI Capabilities" & nl
msg = msg & "Vendor   ID: " & Format$(GetMIDIVendorID()) & nl
msg = msg & "Product ID: " & Format$(GetMIDIProductID()) & nl
msg = msg & "No. of Voices: " & Format$(GetMIDIVoices()) & nl
msg = msg & "No. of Notes: " & Format$(GetMIDINotes()) & nl
msg = msg & "No. of Channels: " & Format$(GetMIDINumChannels())
MsgBox msg, 64, "GBLIB.DLL Report"
```

GetMIDIDriverVersion

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDIDriverVersion(var Major,Minor:INT):INT

Returns

The Version number of the MIDI driver.
N.B. The SoundBlaster AWE32(tm) driver returns 1.51

Visual Basic Declare

VB

```
Declare Function GetMIDIDriverVersion Lib "gblib1.dll" (Major As Integer, Minor As Integer) As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim ALen As Integer
Dim Major As Integer
Dim Minor As Integer
Dim i_RetVal As Integer
Dim szBuf As String * 255
i_RetVal = GetMIDIDriverVersion(Major, Minor)
If i_RetVal Then
    MsgBox "Error in GetMIDIDriverVersion", 48, "GBLIB1.DLL"
    Exit Sub
End If
MsgBox "Driver version is " & Format$(Major) & "." & Format$(Minor), 64, "GetMIDIDriverVersion"
```

GetMIDIProductName

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDIProductName(sz_Buffer:LPSTR;var i_Len:UINT):INT

Returns

The Product name of the MIDI driver.
N.B. The SoundBlaster AWE32(tm) returns the string Voyerta Super Sapi FM Driver

Visual Basic Declare

VB

```
Declare Function GetMIDIProductName Lib "gblib1.dll" (ByVal szBuffer As String, i_BufferLen As Integer) As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim ALen As Integer
Dim szBuf As String * 255
ALen = GetMIDIProductName(szBuf, 255)
msg = "Driver: " & Left$(szBuf, ALen)
MsgBox msg, 64, "GBLIB1 - GetMIDIProductName"
```

GetMIDIVoices

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDIVoices:INT

Returns

Number of simultaneous MIDI Voices supported.
N.B. The SoundBlaster AWE32(tm) returns the number 14

Visual Basic Declare

VB

```
Declare Function GetMIDIVoices Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim nl As String
nl = Chr$(10)
msg = "MIDI Capabilities" & nl
msg = msg & "Vendor   ID: " & Format$(GetMIDIVendorID()) & nl
msg = msg & "Product ID: " & Format$(GetMIDIProductID()) & nl
msg = msg & "No. of Voices: " & Format$(GetMIDIVoices()) & nl
msg = msg & "No. of Notes: " & Format$(GetMIDINotes()) & nl
msg = msg & "No. of Channels: " & Format$(GetMIDINumChannels())
MsgBox msg, 64, "GBLIB.DLL Report"
```

GetMIDINotes

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDINotes:INT

Returns

Number of simultaneous MIDI notes supported.
N.B. The SoundBlaster AWE32(tm) returns the number 14

Visual Basic Declare

VB

```
Declare Function GetMIDINotes Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim nl As String
nl = Chr$(10)
msg = "MIDI Capabilities" & nl
msg = msg & "Vendor   ID: " & Format$(GetMIDIVendorID()) & nl
msg = msg & "Product ID: " & Format$(GetMIDIProductID()) & nl
msg = msg & "No. of Voices: " & Format$(GetMIDIVoices()) & nl
msg = msg & "No. of Notes: " & Format$(GetMIDINotes()) & nl
msg = msg & "No. of Channels: " & Format$(GetMIDINumChannels())
MsgBox msg, 64, "GBLIB.DLL Report"
```

GetMIDINumChannels

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototype

Function GetMIDINumChannels:INT

Returns

Number of MIDI Channels supported
N.B. The SoundBlaster AWE32(tm) returns the number 8

Visual Basic Declare

VB

```
Declare Function GetMIDINumChannels Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim msg As String
Dim nl As String
nl = Chr$(10)
msg = "MIDI Capabilities" & nl
msg = msg & "Vendor   ID: " & Format$(GetMIDIVendorID()) & nl
msg = msg & "Product ID: " & Format$(GetMIDIProductID()) & nl
msg = msg & "No. of Voices: " & Format$(GetMIDIVoices()) & nl
msg = msg & "No. of Notes: " & Format$(GetMIDINotes()) & nl
msg = msg & "No. of Channels: " & Format$(GetMIDINumChannels())
MsgBox msg, 64, "GBLIB.DLL Report"
```

IsMIDI...

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototypes

Function IsMIDIExternalSynth:INT
Function IsMIDISquareWaveSynth:INT
Function IsMIDIFMSynth:INT
Function IsMIDIGenericSynth:INT
Function IsMIDIMapper:INT

Returns

No=0, Yes=1

Visual Basic Declares

VB

```
Declare Function IsMIDIExternalSynth Lib "gblib1.dll" () As Integer
Declare Function IsMIDISquareWaveSynth Lib "gblib1.dll" () As Integer
Declare Function IsMIDIFMSynth Lib "gblib1.dll" () As Integer
Declare Function IsMIDIMapper Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim msg as String
If IsMIDIExternalSynth() then msg =   an external synthesiser
If IsMIDISquareWaveSynth() then msg = a square wave synthesiser
If IsMIDIFMSynth() then msg = an FM synthesiser
If IsMIDIMapper() then msg = diverted through MIDI Mapper
msg = Your MIDI is   & msg
MsgBox msg, 64, IsMIDI Functions
```

WillMidiDo...

VB
VB (or ALT+x)

Purpose

VB
Used to query the MIDI setup installed in a target machine.

GBLIB.DLL Pascal Prototypes

Function WillMidiDoVolume:INT
Function WillMidiDoLRVolume:INT
Function WillMidiDoCache:INT

Returns

No=0, Yes=1

Visual Basic Declares

VB

```
Declare Function WillMidiDoVolume Lib "gblib1.dll" () As Integer
Declare Function WillMidiDoLRVolume Lib "gblib1.dll" () As Integer
Declare Function WillMidiDoCache Lib "gblib1.dll" () As Integer
```

Visual Basic Example

VB

```
Dim Msg as String
Dim nl as String
nl = Chr$(13)
Msg = Your MIDI setup. & nl
If WillMidiDoVolume() then Msg = Msg & It can Vary the Volume & nl
If WillMidiDoLRVolume() then Msg = Msg & and also do this in Stereo! & nl
If WillMidiDoCache() then Msg = Msg & It can Cache Patches. & nl
MsgBox Msg, 64 GBLIB1.DLL MIDI Report
```

GetClientZero

VB
VB (or ALT+x)

Purpose

VB
Someone, somewhere, may find a good use for this routine. I use it for the Subclassed Form_Move routine.

GBLIB.DLL Pascal Prototype

Procedure GetClientZero(hWind:UINT;var Left:INT;Var Top:INT)

Visual Basic Declare

VB

```
Declare Sub GetClientZero Lib "gblib1.dll" (ByVal hWind As Integer, i_Left As Integer, i_Top As Integer)
```

Visual Basic Example

VB

```
Dim i_Left as Integer
Dim i_Top as Integer
Dim Msg as String
GetClientZero Me.hWnd, i_Left, i_Top
Msg = Form is at    & Format$(i_Left) & , & Format$(i_Top)
MsgBox Msg, 64, GBLIB1 - GetClientZero
```

GetJoyPos

VB
VB (or ALT+x)

Purpose

VB

GBLIB.DLL Pascal Prototype

Procedure GetJoyPos(var XPos:LONGINT;var YPos:LONGINT;var LButton:INT;var RButton:INT)

Visual Basic Declare

VB

Declare Sub GetJoyPos Lib "gblib1.dll" (XPos As Long, YPos As Long, LButton As Integer, RButton As Integer)

Visual Basic Example

VB

```
Global JX as Long
Global JY as Long
Global JRB as Integer
Global JLB as Integer

Sub JoyTimer_Timer ()
GetJoyPos JX, JY, JRB, JLB
If JLB = 1 Then MsgBox "Left Button Pressed"
If JRB = 1 Then MsgBox "Right Button Pressed"
Me.Caption = "Joy-X=" & Format$(JX) & " Joy-Y=" & Format$(JY)
End Sub
```

See... UnSetupJoystick SetUpJoystick

GBLIB1.DLL Function Reference by Category

VB
VB (or ALT+x)

Retrieving Resources from a DLL

Sound Functions

VB Add-Ons

See... Complete Function Reference

Retrieving Resources from a DLL


 (or ALT+x)

Click the sub or function name to obtain more information	
Function PlayDLLWave	index 2
Function LoadDLLBitmap	index 3
Function GetDLLText	index 4
Function GetDLLBitmapSize	index 5
Function PlayResource	index 7
LoadDLLBitmapFrom	index 25
LoadDLLDialog	index 26
LoadDLLIcon	index 27
LoadDLLCursor	index 28
DestroyDLLCursor	index 29
LoadDLLIconXY	index 50

See... Complete Function Reference

Sound Functions

VB
VB (or ALT+x)

Click the sub or function name to obtain more information

See... Complete Function Reference

VB Add-Ons


 (or ALT+x)

Click the sub or function name to obtain more information

See... Complete Function Reference

System requirements for using/distributing GBLIB1.DLL

VB
VB (or ALT+x)

VB
In order to run applications using services provided by GBLIB1.DLL you will need to ensure that the following libraries are installed on both your and the target machine:

Library	Purpose
GBLIB1.DLL	Functions in this Help File
MMSYSTEM.DLL	Sound and Joystick services
LZEXPAND.DLL	GBDecompress function
SHELL.DLL	Icon services

VB
The target machine should be running Windows 3.1 or higher in Enhanced mode.

VB
You can install the DLLs in the users Windows/System directory (Obtained by a call to the API function GetSystemDirectory) **or** you can install them all in the same directory as your application.

VB
I would recommend the second option, as it eliminates any version conflicts with existing DLLs on the users system.

VB
I know that Microsoft(tm) recommend that DLLs be installed in the System directory, but they, and a lot of other software distributors break this rule for the practical reason that Version checking **just does not work in practice** to the benefit of the user

---

See... About GBLIB1.DLL and Conditions of Use
