

PREFACE

This document is a compilation of frequently asked questions and their answers about Visual Basic for Windows and Visual Basic for Applications which have been gathered from posts to the comp.lang.basic.visual newsgroup. Although efforts have been made to find obvious errors, there is no guarantee that the information in this document is error-free. Neither the FAQ maintainer nor anyone else associated with this document assume ANY liability for the content or use of this document. If you find any errors, please report them to the address given below.

This FAQ document is protected by international copyright regulations. Permission is granted to distribute it freely, both in electronic and written format, provided no charge is made. Also, do not make changes to this document without the consent of the maintainer. Usenet netiquette applies.

Most FAQs (including this one) are available at the anonymous ftp archive site "rtfm.mit.edu". All parts of the VB FAQ may be found in the directory "pub/usenet/comp.lang.basic.visual".

You can also have the VB FAQs e-mailed to you by sending a message to "mail-server@rtfm.mit.edu" with ONLY the text **SEND USENET/NEWS.ANSWERS/VISUAL-BASIC-FAQ/*** in the BODY of the message.

As the FAQ maintainer, I don't have time to explore all of the aspects of Visual Basic. I rely on your submissions to improve the quality and inclusiveness of this document. If you have found a VB hint, tip, trick, work-around, etc., please write it up and send it to me!

Jan Steinar Haugland (Jan.Haugland@uib.no), VB/Win FAQ maintainer

Please note that my first language is not English. You can safely ignore all typos, but if you find an error that is so embarrassing you can't help pulling your hair and screaming loud, just send me a note and I'll correct it quietly. OK?

I would like to thank all contributors, and also all those who have given constructive feedback. This FAQ is now widely distributed in the net community.

Special thanks to the following people who have made many and invaluable contributions to the VB/Win FAQ: Kris Nosack (the previous maintainer), Peter Millard, Nic Gibson, Mr. "D" (the anonymous contributor), George Tatge (gat@csn.org), Andy Dingley (dingbat@codesmth.demon.co.uk), and those I may have forgotten (sorry).

Tim Roberts is responsible for the Windows Help File version of the FAQ. If you have any problems with the Help file he can be contacted directly at TJR@AC.DAL.CA.

Your help is GREATLY appreciated!

About this document

This document is a compilation of frequently asked questions and their answers about Visual Basic for Windows and Visual Basic for Applications which have been gathered from posts to the comp.lang.basic.visual.* newsgroups. Although efforts have been made to find obvious errors, there is no guarantee that the information in this document is error-free. Neither the FAQ maintainer nor anyone else associated with this document assume ANY liability for the content or use of this document. If you find any errors, please report them to the address given below.

This FAQ document is protected by international copyright regulations. Permission is granted to distribute it freely, both in electronic and written format, provided no charge is made. Also, do not make changes to this document without the consent of the maintainer. Usenet netiquette applies.

Where to get the VB/Win FAQ [++]

Most FAQs (including this one) are available at the anonymous ftp archive site "rtfm.mit.edu". All parts of the VB FAQ may be found here:

`ftp://rtfm.mit.edu/pub/usenet/comp.lang.basic.visual.misc/`

****Alternative FTP Sites**:**

The following are alternative sites to rtfm:

N. America:	<code>ftp.uu.net</code>	<code>/usenet/news.answers</code>
Europe:	<code>ftp.uni-paderborn.de</code>	<code>/pub/FAQ</code>
	<code>ftp.Germany.EU.net</code>	<code>/pub/newsarchive/news.answers</code>
	<code>grasp1.univ-lyon1.fr</code>	<code>/pub/faq</code>
	<code>ftp.win.tue.nl</code>	<code>/pub/usenet/news.answers</code>
	<code>ftp.sunet.se</code>	<code>/pub/usenet</code>
Asia:	<code>nctucca.edu.tw</code>	<code>/USENET/FAQ</code>
	<code>hwarang.postech.ac.kr</code>	<code>/pub/usenet/news.answers</code>

(FYI: "rtfm" stands for "Read The ***** Manual". I kid you not!)
[Andre van Meulebrouck (vanmeule@ils.nwu.edu)]

You can also have the VB FAQs e-mailed to you by sending a message to "mail-server@rtfm.mit.edu" with ONLY the text

```
send usenet/comp.lang.basic.visual.misc/*
```

in the BODY of the message.

Alternative mailservers for those who have no ftp access:

- `ftpmail@decwrl.dec.com` or `ftpmail@cs.uow.edu.au` bit
- `ftp@pucc.princeton.edu` or `ftpmail@lth.se`
- `bitftp@dearn` or to `bitftp@vm.gmd.de` (Europe only)
- `ftpmail@ftp.uni-stuttgart.de`
- `ftpmail@grasp.insa-lyon.fr` or `ftpmail@ieunet.ie`
- `bitftp@plearn.edu.pl` or `bitftp@plearn` (Europe)
- `ftpmail@doc.ic.ac.uk` or `ftpmail@sunsite.unc.edu`

[thanks to Jakob Faarvang (jakobf@apexsc.com)]

For all mailservers:

Use the "index" command to get a list of files available at the site. Use the "help" command to get more detailed instructions. (NOTE: commands are in BODY of mail message!)

Note that a Web-browser like Netscape or Microsoft Internet Explorer easily can access an ftp site, view and download files. The ftp address on the top of this point (`ftp://...`) is formatted for these browsers; just copy and paste the text directly into the "URL" line.

There is currently work going on to make these available in HTML format (for Web-browsers). This work is done by Peter Millard <millard@buffnet.net>. Look at:

<http://www.buffnet.net/~millard/vblinks.htm>

or go directly to the copies of his HTML FAQ's (when finished!):

<http://www.buffnet.net/~millard/vbgenfaq.htm>
<http://www.buffnet.net/~millard/vbdosfaq.htm>

Kudos & Comments

In this document, whenever a long line of code must be split into two or more lines of text in the code examples, a | symbol will precede each line which should be appended to the line above it.

As the FAQ maintainer, I don't have time to explore all of the aspects of Visual Basic. Neither have I time or knowhow to personally answer direct technical questions thoroughly. I rely on your submissions to improve the quality and inclusiveness of this document. If you have found a VB hint, tip, trick, work-around, etc., please write it up and send it to me!

Jan Steinar Haugland (Jan.Haugland@uib.no), VB/Win FAQ maintainer

Please note that my first language is not English. You can safely ignore all typos, but if you find an error that is so embarrassing you can't help pulling your hair and screaming loud, just send me a note and I'll correct it quietly. OK?

This document is a collective effort! I would like to thank all contributors, and also all those who have given constructive feedback. This FAQ is now widely distributed in the net community.

Special thanks to the following people who have made many and invaluable contributions to the VB/Win FAQ: Kris Nosack (the previous maintainer), Peter Millard, Nic Gibson, Mr. "D" (the anonymous contributor), George Tatge (gat@csn.org), Andy Dingley (dingbat@codesmth.demon.co.uk), Ayn Shipley (ashipley@hookup.net) and those I may have forgotten (sorry).

John McGuire (jmcguire@jax.jaxnet.com), a longtime VB user, recently went through the FAQ and found lots of things worth his comments. Many of his corrections and suggestions are implemented throughout. Thanks!

Everybody: Your help is GREATLY appreciated!

Does VB/Win make standalone .EXE files?

VB/Win produces .exe files, but they are not standalone. All VB/Win programs must be distributed with the VBRUNx00.DLL file (where x is the major version number). This DLL must accompany all VB/Win programs, but only one such file should reside on every system where VB programs are used.

What is the current version of Visual Basic for Windows? [++]

VB/Win is just between version 3.0 and 4.0 as of writing this. 4.0 is about to be released around the same time as Windows 95. This will contain both a 16-bit and a 32-bit (for NT and Win95) version of the developing environment and the integrated compiler.

Where can I get updated VB and other Microsoft files?

Microsoft Software Library (MSL) is accessible from the following services:

- Compu\$erve

GO MSL

Search for <filename.EXE>

- Microsoft Download Service (MSDL)

Dial (206) 936-6735 to connect to MSDL

- Internet (anonymous FTP)

ftp ftp.microsoft.com

Change to the \softlib\mslfiles directory

(There are a LOT of files in this directory! It is not advisable to list all the files unless you have good time.)

Help! I am lost on ftp.microsoft.com.

You no longer have to be! The site have been reorganised, and you'll find the file DIRMAP.TXT in root (or whatever it's called on a 3.5 NT server). Here's the file as I found it (stolen ruthlessly from the server for your convenience):

This file is to help you find your way around ftp.microsoft.com. This file only covers the directory structure two levels deep. If you see a 'kb' directory in a second level directory, it contains all of the information regarding that second level directory. For example, the /developr/win32dk directory has a kb directory in it. This kb directory contains all of the articles for any 32 bit development kit.

<u>Root Directory</u>	<u>Sub Directory</u>	<u>Contents</u>
ADVSYS		Advanced Systems, Networks,Mail
	LANMAN	LanMan & other networks
	MAIL	Mail and Schedule+
	MSCLIENT	Microsoft Networking Client
	SQL_ODBC	SQL and ODBC
	WINNT	Windows NT
	WINSOCK	Windows Sockets information
DESKAPPS		Desktop Applications
	ACCESS	Access
	DOSWORD	Word for MS-DOS
	EXCEL	Excel
	GAMES	Entertainment Packs, Flight Sim, etc.
	HOMEAPPS	Home applications (Fine Artist, Creative Writer)
	MISCAPPS	Other applications
	MMAPPS	Multimedia Titles
	OFFICE	Microsoft Office
	POWERPT	PowerPoint
	PROJECT	Project
	PUBLISHER	Publisher
	WORD	Word for Windows & Macintosh
	WRKS_MNY	Works and Money
DEVELOPR		Developer Tools and Information
	BASIC	Quick Basic & other Basics
	DEVCAST	DevCast information
	DEVUTIL	MS Test, Delta, EXEMOD, EXEPACK, & LIB Utility
	DRG	Developer Relations Group
	FORTTRAN	Fortran and Fortran PowerStation
	FOX	FoxPro and FoxBase

	MAPI	Messaging API information
	MASM	Macro Assembler
	MSDN	Microsoft Developer Network
	MSJ	Microsoft Systems Journal
	OLE	OLE
	TAPI	Telephony API information
	VB	Visual Basic
	VISUAL_C	Visual C++, MFC, & other C products
	WIN_DK	Windows SDK, DKs & At Work
	WIN32DK	32 bit Development Kits
MSEDCERT		Microsoft Education and Certification
	EDUCATIO	Microsoft Education information
	CERTIFIC	Microsoft Certified Professional info
MSFT		Microsoft shareholder information
	ANNREPT	Microsoft Annual Report
	BACKGRND	Background information on Microsoft
	PRESSREL	Microsoft Press Releases
	SEC	Recent filings with the Securities and Exchange Commission
SOFTLIB		Instructions & index for software library
	MSLFILES	Software library files (> 1500 files)
PEROPSYS		Personal Operating Systems and Hardware
	HARDWARE	Mouse & other Hardware
	MSDOS	MS-DOS
	WINDOWS	Windows (all versions)
	WIN_NEWS	Information on Windows "Chicago"
TECHNET		Information on TechNet
	SERVDIR	Microsoft Services Directory

NOTE: Directory and file names at ftp.microsoft.com are NOT case sensitive.

Thanks to Richard Mason (richard@whitney.demon.co.uk) for the map he made for us before this long overdue reorganisation, and thanks for bringing this to my attention.

Where can I get good up-to-date information about VB? [++]

If you do any VB programming at all, you really should get the latest copy of the Microsoft Knowledge Base from one of the sources listed above! The filename is VBKB.EXE or VBKB_FT.EXE for the version with full text searching. The Knowledge Base is a Windows help formatted document that is updated almost monthly. (Beware: The files are huge!)

The EXE files on ftp.microsoft.com (and other places) are simple self-extracting files. They don't do any automatic updating of your system, just dump the file(s) inside them in the current directory on your disk, uncompressed and ready for use. Just replace the old files on your system with the new files (same name, usually).

This FAQ (ahem) is a pretty good source as a digest of many of the VB issues that are discussed in the comp.lang.basic.visual.* newsgroups, but as such the information contained in this FAQ may not have been thoroughly tested or verified.

For easier access to and use of this FAQ document, get the Windows Help file format FAQ doc by anonymous ftp to quasar.sba.dal.ca and look under /www/hlp. The .HLP version is made by Tim Roberts (TJR@SBACOO.P.SBA.DAL.CA). This is HIGHLY RECOMMENDED.

Dave McCarter puts out a nice Windows help formatted document called "Visual Basic Tips and Tricks". A good resource with information that isn't found in the Knowledge Base. It can be found by anonymous ftp to ftp.cica.indiana.edu /pub/pc/win3/programr/vbasic/ as VBTIPS??.ZIP, where '??' is the version number (yes, get the one with the highest number).

[[Mark Schoonover has started a monthly magazine called _VBWin Programmer's Magazine_ that is compiled around the Discussion of Microsoft Visual Basic and Related Issues mailing list. This magazine is available at the usual ftp sites like cica and at his BBS (619) 571-2846. Mark can be contacted at schoon@cts.com. It also has its own anonymous ftp site: ftp.cts.com in the /pub/schoon/VBWM.Issues directory. - ED: this appears to be down, alas.]]

Jakob Faarvang (jakobf@apexsc.com) maintains the useful CLBVB Digest:

`http://www.apexsc.com/vb/clbv-digest/`

and

`ftp.apexsc.com:/pub/cgvb/clbv-digest/`

Address for him regarding CLBVB Digest is clbv-digest-editor@apexsc.com

There is presently an effort to put together a VB code library so that VB users can share their best - and trickiest - programming work. The code library project is being spear headed by Hein Ragas who has managed to get a directory on CICA for code snippets to be deposited. Stayed tuned to comp.lang.basic.visual.misc for more information.

There's a VISBAS-L mailing list for Visual Basic (Thanks to David Liden (DL9U@Virginia.EDU) for tracking it down for me when it moved). You can subscribe to this mailing list by sending an email to

`listserv@listserv.tamu.edu`

Place the following text in the *body* of the message (no, not the subject line):

`SUB VISBAS-L Real Name`

Where Real Name is just that, *your full real name* not your email address. Note that the traffic on this list

may overflow your mailbox if you have a limited mail buffer. Expect around 40-50 messages every weekday, a bit less during weekends. Also, to unsubscribe, do as above but with "UNSUB" in the body of the message to the LISTSERV address, *not* to the VISBAS-L list address itself (Believe it or not, we usually receive 2-3 sub/unsub mails every day on the list, and we're really fed up!) Also, this mail server was split and a new list called VBDATA-L was made for Vb database (Jet) related topics. Same procedure for registering (SUB VBDATA-L Real Name).

The address for the MS ACCESS listserver is listserv@indycms.bitnet or listserv@indycms.iupui.edu The list name is ACCESS-L. To subscribe, you follow the same procedure as for the other lists.

Sorry, I know no mailservers or other Internet resources dedicated to VBA specifically. Send me any information you may have, and I'll bring it on.

There are several Usenet newsgroups dedicated to MS Windows programming and use. In fact, far too many to list here :-)

NOTE: PLEASE don't post VB stuff to comp.lang.visual. This group has *nothing* to do with Visual Basic, and the academics discussing "real" visual programming there are very, very annoyed at what they call "quasi-visual" stuff and postings about those languages to their group.

If you have a Compu\$erve account, you will find a forum for Visual Basic there, including some support from Microsoft:

MS BASIC Forum (GO MSBASIC)

<u>Message Sections Available:</u>	Libraries Available:
1 Forum News/Info	1 MS Info and Index
2 Setup Wizard/Kit	2 Setup Wizard/Kit
3 Data Access Objects	3 Data Access Objects
4 The Data Control	4 The Data Control
5 Programming Issues	5 Programming Issues
6 ODBC Connectivity	6 ODBC Connectivity
7 SQL Queries	7 SQL Queries
8 ProEdition Controls	8 ProEdition Controls
9 Calling API's/DLL's	9 Calling API's/DLL's
10 Using OLE/DDE	10 VBWIN-ODBC/Database
11 MSCOMM control	11 MSCOMM control
12 MCI/MAPI controls	12 MCI/MAPI controls
13 DOS Visual Basic	13 DOS Visual Basic
14 DOS and Mac Basic	14 DOS and Mac Basic
15 Suggestions/Mktg.	15 Suggestions/Mktg.
16 CDK	16 CDK
17 3rd Party Products	17 3rd Party Products

There are magazines dedicated to VB. The best known is Fawcett Technical Publications' _Visual Basic Programmer's Journal_ (VSPJ). Phone 800-848-5523 (for US credit card orders) or 303-541- 0610 (int'l and US other orders), Email 74003.224@compuserve.com to Shirley Modric for subscription info. Address is 280 Second Street, Suite 200, Los Altos, CA 94022-3603 USA.

From Randy Coates (rcoates@telerama.lm.com): I currently subscribe to "Inside Visual BASIC for Windows" from the Cobb Group. Although it is a helpful monthly paper (about 14 pages per publication), I find it to be overpriced when compared to VB Programmers Journal. Here is the information anyway: Domestic \$59/yr (\$7.00 each); Outside US \$79/yr (\$8.50 each) Phone: Toll Free: 800-223-8720), Local: 502-491-1900, Customer Relations Fax: 502-491-8050, Editorial Department Fax: 502-491- 4200. Address: _Inside Visual BASIC for Windows_, 9420 Bunsen Parkway, Suite 300, Louisville, KY 40220.

(Note: for completeness other VB magazines should be listed, and I would like to receive info on those!)

Are there any examples of commercial applications built using Visual Basic?

Profit by Microsoft was written mostly in Visual Basic. In fact, Profit was one of three programs selected as PC Magazine's Editor's Choice among Windows small business accounting packages.

Microsoft uses VB extensively for smaller utilities. 3 of the small apps in Windows and Windows for Workgroups Resource Toolkits are written in VB. Also, if you have the Microsoft Bookshelf CD-ROM, you will notice that the MVOPTION.EXE program, which is an "options" program for MS Viewer, is created in VB.

Note: The existence of VBX files in a package doesn't necessarily mean that it was written in VB. The most popular C++ compilers also support VBXes.

WWW pages for VB! []**

Carl 'n Gary's Visual Basic HomePage is a good place to start:

<http://www.apexsc.com/vb/>

This page has hotlinks to lots of goodies, including the FAQs, clbv.* archives (with search tool), etc. Send any e-mail inquiries (about the page!) to:

vb-admin@apexsc.com

[Gary Wisniewski (gary@apexsc.com)]

Limits of VB? [**]

Are you kidding? VB have *no* limitations... Uh, yeah ;-)

For starters:

- It's not a true compiler, hence it's slow for non-interface stuff (it's of course slow for interface stuff as well, but that's *Windows* not VB)
- It's not really object-oriented (Try looking for the parent of ie. a line control, and you'll wonder why it has no hWnd - SpyWorks is an add-on you may need if this is annoying)
- A statement must be on a single line! (Note: Fixed in VB4!!!!)
- No arrays of constants.
- Your Complaint Here!

An enormous amount of contributors to this topic! Can you guess why?

[Entry suggested by Andre van Meulebrouck (vanmeule@netcom.com)]

What's the difference between MODAL and MODELESS forms? [++]

MODAL forms are forms which require user input before any other actions can be taken place. In other words, a modal form has exclusive focus in that application until it is dismissed. When showing a modal form, the controls outside this modal form will not take user interaction until the form is closed. The internal MsgBox and InputBox forms are examples of modal forms. To show a form modally, use the syntax:

```
MyForm.SHOW 1
```

MODELESS forms are those which are shown but do not require immediate user input. MDI child forms are always modeless. To show a form modeless, use the syntax:

```
MyForm.SHOW
```

(Thanks to John M. Calvert (calvertj@magi.com) for correcting a slightly embarrassing mistake in previous versions of this topic)

When/Why should I use Option Explicit?

Option Explicit forces you to declare all variables before using them. Opinions vary greatly on this subject. The main reason to use the OPTION EXPLICIT statement at the top of all modules is to minimize the amount of bugs introduced into your code by misspelling a variable name. Most variants of BASIC (including VB) have the capability to create variables 'on the fly' (without any declarations). This capability can be a double edged sword.

At the minimum, some suggest using the DEFINT A-Z statement in leu of OPTION EXPLICIT. This statement will cause any variables which are created on the fly to be created as integers as opposed to variant (VB 3.0) or single precision (VB 1.0 and 2.0). (Integers take up less memory).

The OPTION EXPLICIT statement causes VB to 'disable' its ability to create variables on the fly. Thus, all variables must be declared using a DIM or REDIM statement. All variables not declared will cause an error when the OPTION EXPLICIT statement is used. This will eliminate bugs caused by a misspelled variable. The option works module-wide, so you can have some modules with and some without this option in your project.

Why does everybody say I should save in TEXT not BINARY?

Actually, saving in binary mode is a bit faster, so why do we recommend you to save in text?

If you save the source and the project as text, it becomes ASCII (or really, ANSI) code that you can edit with any text editor or (if you are careful when you save) word processor. If you save in binary, only the VB development environment, current or later versions, will understand the code. The Setup Wizard can not scan binary projects. Also, source documenters and other programming tools usually require text mode. If you use text, you can use a simple text editor (ie. notepad) to cut and paste code from other source/form modules into your current project. Some 'tricks' (like making an array of 1 control into a single non-array control again) is easily done with an editor but not that easy in the environment. If you want to print your project to paper the file|print option in the VB environment is often not good enough; you may want to import the text files into your word processor. And, finally, if something goes wrong (only one byte is changed!) you may be out of luck in binary mode. In text mode you will more easily be able to fix it.

Is the Variant type slower than using other variable types?

Generally, yes, if we are talking numeric variable types. The Variant type also increases memory overhead. To test the speed difference, try the following piece of code in something like a button_click event and keep the debug window on the screen:

```
Dim Va As Variant
Dim In As Integer
T1! = Timer
For i% = 1 To 32766
    Va = i%
Next i%
T2! = Timer
Debug.Print "With variant: "; Format$((T2! - T1!), "0.0000")
T1! = Timer
For i% = 1 To 32766
    In = i%
Next i%
T2! = Timer
Debug.Print "With integer: "; Format$((T2! - T1!), "0.0000")
```

This test shows (on our test system) that integers are ~60% faster! However, for strings there where no real difference, or in some instances, variants were faster than strings for routines with heavy conversion usage. For the best result in your application, test your routines directly.

How do I make a text box not beep but do something else when I hit the Enter key?

Put "something else" in your `_KeyPress` event, depending on what you really want. This code example makes *nothing* happen, for an extended period of time:

```
Sub Text1_KeyPress (KeyAscii As Integer)
    If KeyAscii = 13 Then '13 is Key_Return
        KeyAscii = 0
    End If
End Sub
```

This might not be a very nice thing to do, since your users usually have some intention when they press Enter. Often they will want to jump to the next control, like the Tab key does. To have the Enter key emulate the Tab key action, you will need to add the line `'SendKeys "{tab}"` above `'KeyAscii=0` in the example above (Yes, I thought `KeyAscii=9` works but it doesn't! Tab is obviously handled by Windows on a lower level).

By the way, you'll also find this in the Microsoft VB Knowledge Base (see KB Q78305 and Q85562).

Note: If `MultiLine=True` you will *not* want to disable the normal behaviour of the Enter key.

How do I get the Tab key to be treated like a normal character?

You must set TabStop = False for ALL controls on the active form. Then you will be able to insert "tab" (chr 9) characters in controls like the text box.

If you feel you need the Tab key to behave "normal" (ie. jump to next control) outside this specific control, it is trivial to emulate its functionality in code:

```
Sub Command1_KeyDown (KeyCode As Integer, Shift As Integer)
    If KeyCode = 9 Then
        If Shift = 0 Then
            Command2.SetFocus 'Tab=Next control
        ElseIf Shift = 1 Then
            Command3.SetFocus 'Shift-Tab=Prev.ctrl.
        End If
    End If
End Sub
```

...etc.

How do I implement an incremental search in list/dir/combo/file boxes?

This is your lucky day. Dan Champagne (Dan_Champagne@dell.com) made some VB code (no DLLs are necessary!) which easily provides this feature for your applications:

```
' Code by Dan Champagne
' 4/18/94

'This code can be used to do an incremental search in either a
'list box, dir, combo, or a file box. The following code is set
'for a file box called FILE1. To make it work with a list box, or
'a file box with a different name, change all occurrences of FILE1
'with whatever you or VB has named your list, combo, dir, or file box.
'There are two places where you will need to change these. They are
'on the last couple of lines in the KeyPress code.
'Also, thanks to John Tarr for helping debug the code.

'In a .BAS file, add the following:
'searchme$ is a global variable that will keep track of what the
'user has typed so far.
Global searchme$

'The following needs to be on one line.
Declare Function SendMessageBystring& Lib "User" ALIAS "SendMessage"
(ByVal hWnd%, ByVal wParam%, ByVal lParam%)

Global Const WM_USER = &H400
Global Const LB_SELECTSTRING = (WM_USER + 13)
Global Const LB_FINDSTRING = (WM_USER + 16)

'In File1 under keydown, add the following:
'This checks if the user has pressed the up or down arrow.
'If they have, reset searchme$ to "".
If KeyCode = 40 Or KeyCode = 38 Then
    searchme$ = ""
End If

'In File1 under lostfocus, pathchange, patternchange, and click add:

'If the user has done any of the above, reset the searchme$
'string.
searchme$ = ""

'In File1 under keypress add:

Dim result&

Select Case KeyAscii
    Case 8      'Backspace
        If searchme$ <> "" Then
            searchme$ = Left$(searchme$, Len(searchme$) - 1)
        Else
            File1.ListIndex = 0
        End If
    Case Else
        result& = SendMessageBystring&(File1.hwnd, wParam, lParam)
    End Select
```

```

        KeyAscii = 0
        Exit Sub
    Case 27      'Escape
        searchme$ = ""
        KeyAscii = 0
        Exit Sub
    Case 13      'Enter
        searchme$ = ""
        KeyAscii = 0
        Exit Sub
    Case Asc("a") To Asc("z"), Asc("A") To Asc("Z"), Asc("'"), Asc("."),
Asc(" "), Asc("0") To Asc("9")
        searchme$ = searchme$ & Chr$(KeyAscii)
        KeyAscii = 0
End Select

result& = SendMessageBystring(FILE1.hWnd, LB_FINDSTRING, 0, searchme$)

If result& = -1 Then
    searchme$ = Left$(searchme$, Len(searchme$) - 1)
Else
    result& = SendMessageBystring(FILE1.hWnd, LB_SELECTSTRING, -1,
searchme$)
End If

```


How do I make an animated icon for my program?

For an example on how you change the icon for your application as it is displayed when it is minimized, see the example REDTOP in the \samples\picclip directory for VB/Win 3 Pro. This demonstrates a fancy animated icon.

What is passing by reference?

Arguments are either passed by reference or by value. When they are passed by value, they cannot be changed by the procedure or function they are passed to. They *can* be altered when passed by reference, since passing by reference is just passing the address.

Note that procedures are less strict about variable types when you use BYVAL. If you declare that your Sub takes a Variant, VB takes that seriously and gives a nasty "mismatch error" if you try to pass ie. a string to it. Make it ByVal (at the cost of some speed) and your sub will be more tolerant.

Also note the following nasty trap: Arguments are passed by reference unless enclosed by parentheses or declared using the ByVal keyword. [VBWin Language Ref., p. 55]

I get a "file not found" error on the IIF function when I distribute by program. Uh?

There's a documentation error, since the manual does not tell you that the IIF function requires the file MSAFINX.DLL to be distributed with your application. No, IIF is not financial (I should know, I study finance right now, or at least I should be doing that ;-]).

Is there any way to pass a variable to a form apart from using global variables?

The standard workaround is to put an invisible text box (or caption or any other control that suits your use.) on the target form and access it by `Form.textbox = "value"`. Then you can use the Change event of that control to do anything you want in that form. Also, check out the `.Tag` property which is a "what-you-want" property where you can hook any string you want onto a control. This property can also be accessed from other modules.

[Dave Mitton (mitton@dave.enet.dec.com)]

How should dates be implemented so they work with other language and country formats?

If you use ie. MM/DD/YY format dates in a program, you will get either a runtime-error (ie. month>12) or the wrong date (ie. March 12 instead of December 3) when your program is used in Europe. And vice versa, of course. Even Microsoft's own example programs (like the MAPI sample) make this stupid mistake and fail miserably. Use the Format command to make sure you get the date you want. For example:

```
strTodaysDate = Format[$](Now, "Short Date")
```

As a side note, Microsoft has taken much heat on the newsgroup for VB's bad support for internationalization! Just try to make a date literal in source code that works everywhere as a little exercise. Answer elsewhere in this document. No prizes :-)

Can a VB application be an OLE server?

No. You'll have to use an external DLL/VBX. If you see any examples, please tell the newsgroup.

How do I dial a phone number without using the MSCOMM VBX?

The MSCOMM VBX that comes with VB/Pro is great for creating communication programs, but it's overkill for dialing a phone number. Try the following code:

```
PhoneNumber$ = "(123)456-7890"  
Open "COM2" For Output As #1 'or COM1  
Print #1, "ATDT" & PhoneNumber$ & Chr$(13)  
Close #1
```

Ian Storrs <exuian@exu.ericsson.se> informed me that he had experienced problems with this when the VB program was run from a network drive. A file named "COM1" was created on the disk! This trick is probably not a good idea for bigger applications, but it's nice for small personal utilities.

I have [several] megabytes of memory. Why do I get an "out of memory" error? [++]

This problem and its solution(s) should not be applicable to Windows 95.

Unfortunately, Microsoft has been more famous for memory barriers than anything else. This is a late descendant of the infamous 640K barrier that has been plaguing us for years. Although Windows allows the user to access several megabytes of memory, it uses two limited (64K) memory areas called User Heap and GDI Heap for some specific data structures. Go to the Help|About box in Program Manager to see the percentage of free resources in the *most* exhausted heap. If these areas are exhausted, you are out of luck. VB programs are unfortunately rather greedy on these structures. Windows 4 is supposed to free us from this limitation...

Note that every visible control (ie every button) is a window to Windows. Every new control takes up some bytes in the precious User heap.

Also, there is another way to run out of memory in Windows, not related to VB. Windows requires free Upper Memory Area (UMA, also called Upper Memory Blocks, not to be confused with High RAM, which is the first 64K of extended memory) to do certain tasks. If you use QEMM or DOS 6+ MemMaker and you have many device drivers (network, etc) this area may have been filled up before you launch Windows. You will then be unable to start applications, even though you have plenty of free RAM. The problem can be solved with careful memory setup, but this is far beyond the scope of this FAQ.

On a completely unrelated problem: When you run a program with an outline control with some ATI graphics cards, it may crash with just that error message. (see Knowledge Base Q100194 PRB: "Some ATI Video Drivers Hang When Using MSOUTLIN.VBX")

How do I mimic a toggle button? [++]

The only "fix" we know for this problem is to use a picture or image control to mimic the action of a button or button3d control. You need two bitmaps, one for buttonup and one for buttotdown (and perhaps one more for inactive state). This is a kluge, we know. Look at the button bar used in the MDINOTE sample program supplied with VB for an example of this.

How do I get my application on top?

To force a form to the front of the screen, do the following command:

```
Form1.ZOrder
```

To make the application *stay* on top, put the Zorder command in a Timer event repeatedly called, say, every 1000 msec. This makes a "softer" on-top than other methods, and allows the user to make a short peek below the form.

There are two different "Zorder"s of forms in Windows, both implemented internally as linked lists. One is for "normal" windows, the other for real "topmost" windows (like the Clock application which is distributed with MS Windows). The Zorder command above simply moves your window to the top of the "normal" window stack. To make your window truly topmost, use the SetWindowPos API call like this:

```
'Make these declares:
Declare Function SetWindowPos Lib "user" (ByVal h%,ByVal hb%, ByVal x%,
ByVal y%, ByVal cx%, ByVal cy%,ByVal f%) As Integer
Global Const SWP_NOMOVE = 2
Global Const SWP_NOSIZE = 1
Global Const FLAGS = SWP_NOMOVE Or SWP_NOSIZE
Global Const HWND_TOPMOST = -1
Global Const HWND_NOTOPMOST = -2

'To set Form1 as a TopMost form, do the following:
res% = SetWindowPos (Form1.hWnd, HWND_TOPMOST, 0, 0, 0, 0, FLAGS)
'if res%=0, there is an error

'To turn off topmost (make the form act normal again):
res% = SetWindowPos (Form1.hWnd, HWND_NOTOPMOST, 0, 0, 0, 0, FLAGS)
```

Is there a way to break long lines in VB code?

There is unfortunately no line continuation character in VB/Win 3.0. Excel 5 VBA does, however, use Space+Underscore (" _") as a line continuation character, and we hope this will be included in the next version of VB.

There are a few tricks you can use to reduce line length, but unfortunately there is very little to do with DECLARE statements which can get very long.

Print your source in landscape :-/

How do I remove/change the picture property of a control at design time?

Mark the (bitmap) or (icon) text in the property window and press Del or Backspace. "No!" I hear you cry, "It doesn't work". Well, it does if you first select the object from the combo box at the top of the Properties Window, and then immediately afterwards doubleclick (or paint over) the "(bitmap)" text and press Del. Alternatively, just click on another control, then click back to the first control. Now Del works. Who said "bug"?

If you want to paste your picture directly into the VB program by pressing Ctrl-V when you are editing the picture property, you will have to use a similar procedure: select the control, select the property, press Ctrl-V. If you try it again without deselecting the control first (or selecting it from the combo box), it doesn't work.

Is a [foo] VBX/DLL available as shareware/freeware?

Part 4 of the FAQ is Adam Harris' excellent "Shareware Custom Controls List". Please consult this list before you post this question.

The following type of controls are NOT known to be available as sw/pd/fw for Visual Basic, only as commercial toolboxes (If you feel like making any of these for VB and sharing it for a modest fee, you will become very popular!):

- a. ZModem communication control/source
- b. Rich Text Format-control or other mixed font/word processor control (rumours indicate that this will be in the Windows 4 API, and therefor available from VB)
- c. Matrix math

If any of these should be available, please tell us.

How do I make my applications screen-resolution independent?

There are two methods: Either get a custom control that does the job for you, or you write lots of complicated code in the Load and Resize events.

For the first option, check out VideoSoft's \$hareware VSVBX.VBX (download VSVBX.ZIP from Cica or mirrors). It has a will of its own, as you will experience, but it's generally better than trying what is described below.

For the brave (or stupid), try to write "screen resolution-smart code" in the form's Load event. If the form is resizable (normally it should be), you'll have to put some magic into the Resize event as well. There are 4 rules of thumb:

- a. **Do not trust the form's height and width properties.** These measure the entire form, not the client area where your controls are. To see this in action, create a simple applet with the only code being in the resize event which resets a line control from 0,0 to the form's width,height properties. The top left corner is in the client area, the bottom right corner disappears. The API call `GetClientRect` will return the size of the client area in pixels. You can use the screen object's `TwipsPerPixelX` and `TwipsPerPixelY` properties to convert from pixels to twips. If that's not enough, `GetWindowRect` will return the actual size of the entire form, client and non-client areas combined. `GetSystemMetrics` will return individual pieces of things like border width/height, caption height, etc.
- b. **Use the `TextWidth` and `TextHeight` properties.** You can use them off the form if all your controls share the same font, otherwise use them off of the given control. I typically do a `TextWidth("X")` and `TextHeight("X")` to get a value which I use as a margin between controls. I grab these values on startup, and multiply it by 2, 1.5, .75, .5, .25 to get varying margin sizes, depending on how close or far apart I want to space things. If your control has an autosize property, you may want to use it, and then calculate the maximum width of a control in a given "column" of controls on your screen and position all of them accordingly.
- c. **Try not to resize your controls in the resize event.** You will spawn another resize event in the process. Of course, you can use a flag to determine whether the resize event is the original event or the spawned one. Using the load event, and setting the forms borders to fixed minimizes the amount of work you have to do.
- d. **Make sure you use a consistent scale.** I don't even bother with the scale properties, but instead just convert pixels (from API calls) into twips and be done with it. If you do use scale properties, be sure you convert your numbers correctly. I had no end of difficulty when I failed to convert into twips with one number that was used in a series of calculations to position controls. Also be sure all your controls share the same SCALE -- another nasty problem I had before I gave up on them completely. [Thanks to our generous anonymous source "D"]

How do I do Peek and Poke? [++]

VB provides no mechanism for this. There are several 3rd party pkgs. which provide this. Also, this often comes up in regards to the comm ports and you can many times do what you want with the mscomm.vbx.

[George Tatge (gat@csn.org)]

On The Developer Network Library CD, you'll find the following:

VBASM is a Microsoft(R) Visual Basic(R) dynamic-link library (DLL) that helps Visual Basic programmers accomplish tasks that are difficult or impossible using Visual Basic alone. This sample application includes two programs, SNOOPER and TXTALIGN, that demonstrate the DLL's use. The DLL contains many low-level routines such as access to real and protected mode interrupts, port input/output (I/O), peek/poke, control manipulation, and so on. These routines and their associated functions include:

- Control Manipulation vbGetCtrlHwnd, vbGetCtrlModel, vbGetCtrlName, vbRecreateCtrl
- Pointer and Memory vbGetData, vbGetLongPtr, vbPeek, vbPeekw, vbPoke, vbPokew, vbSAdd, vbSetData, vbSSeg, vbVarPtr, vbVarSeg
- Byte/Word/Long Manipulation vbHiByte, vbHiWord, vbLoByte, vbLoWord, vbMakeLong, vbMakeWord
- I/O Access vbInp, vbInpw, vbOut, vbOutw
- Interrupts vbInterrupt, vbInterruptX, vbRealModelntX
- Others vbGetDriveType, vbShiftLeft, vbShiftRight

[Deane Gardner <deaneg@ix.netcom.com> quotes Microsoft]

There's a shareware package for in/out routines, btw.

Why doesn't "my string" & Chr\$(13) do what I want?

You need to also add a Chr\$(10): "my string" & Chr\$(13) & Chr\$(10) will give you a CR and LF.

[George Tatge (gat@csn.org)]

How do I prevent multiple instances of my program?

In VB 3, the property `App.PrevInstance` is set to `True` if an older instance of the program already exist.

The following piece of code, stolen from MS KB article Q102480, will activate the old instance and then terminate itself:

```
Sub Form_Load()  
    If App.PrevInstance Then  
        SaveTitle$ = App.Title  
        App.Title = "... duplicate instance."  
'Pretty, eh?  
        Form1.Caption = "... duplicate instance."  
        AppActivate SaveTitle$  
        SendKeys "% R", True  
    End  
End If  
End Sub
```

As Robert Knienider(rknienid@email.tuwien.ac.at) informed me, this piece of code WILL NOT work for non-English versions of MS Windows where the word for "Restore" does not have "R" as the underlined word. Replace the "R" in the `SendKeys` line above with "{ENTER}" or "~".

Note that you shouldn't prevent multiple instances of your application unless you have a good reason to do so, since this is a very useful feature in MS Windows. Windows will only load the code and dynamic link code **once**, so it (normally) uses much less memory for the later instances than the first.

How do I implement an accelerator key for a text box?

You want to use a label caption to identify a text box and to act as if it were the text box caption:

Example:

```
&Label1 [text1]
```

How should I do to set the focus to text1, by typing <ALT>L

Make sure that the TabIndex property for the text box is 1 greater than the label's TabIndex. Since a label can't have the focus, the focus will go to the next item in the tab order, which would be the text box.

Here's any easy way to set the TabIndex for a busy form. Select the object that should be last in the tab order and then select the TabIndex property. Press 0 (zero), click on the next to last object, press 0, click on the the next object back, press 0, etc. When you're done, all of the TabIndexes will be in order, because VB increments all of the higher TabIndexes when you put in a lower number.

Many thanks to Jonathan Kinnick and Gary Weinfurther that provided the answer on the FIDO net echo VISUAL_BASIC.

[Tiago Leal (Tiago.Leal@p25.f1.n283.z2.gds.nl)]

How do I force a file dialogue box to reread the current disk?

If you make a simple dialogue box modelled after common dialogue (normally you should *use* the common dialogue VBX!), you will notice that reselecting the diskette drive will not really rescan the disk. Very annoying to change to C:, and to reselect A: just to make it read the directory of a new diskette.

To solve this problem, put

```
drive1.refresh  
dir1.refresh  
file1.refresh
```

in the code for the "Rescan" button (or whatever).

How do I get the number of free bytes on a disk? [++]

As far as I know, there is three possibilities:

- 1) SETUPKIT.DLL as mentioned above
- 2) Arjen Broeze's VBIO.VBX or something like that, with a lot of options.
- 3) Make your own DISKINFO.DLL from the example code in VBKB article Q106553.

See Article Q113590 (or Q106553) in Microsoft's VB KnowledgeBase. A short extract follows:

```
Declare Function DiskSpaceFree Lib "SETUPKIT.DLL" () As Long
Dim free_space& ' Holds number of bytes returned from
DiskSpaceFree().
ChDrive "c:"    ' Change to the drive you want to test.
free_space& = DiskSpaceFree()
```

[Geir Tutturen(itfgt@nlh10.nlh.no)]

Data Control missing from toolbox when I use VB under NT 3.5. Huh?

Open the VB.INI file and add these lines under the [Visual Basic] heading:

ReportDesign=1

DataAccess=1

[Danny Ames (dames@pic.net)]

How do I tell when an application executed using the SHELL command is finished?

Shell() doesn't really return a task handle, it returns an instance handle. Any documentation that says otherwise is wrong. But never mind that; the answer to your question is to use the API call GetModuleUsage.

```
'Put this in the general declarations of your form/module
Declare Function GetModuleUsage Lib "Kernel" (ByVal hModule As
Integer) As Integer
```

```
'Here's where you shell out to the other program
intHandle = Shell("PROGRAM.EXE")
Do While GetModuleUsage(intHandle) > 0
    DoEvents
Loop
```

[Kenn Nesbitt, Microsoft Consulting Services (kennn@netcom.com)]

How do I access C style strings?

Use the 'Istrlen' and 'Istrcpy' calls found in the Kernel DLL.

How can I change the printer Windows uses in code without using the print common dialog?

You can change the printer the VB 3.0 Printer object is pointing to programmatically (without using the common dialogs). Just use the WriteProfileString API call and rewrite the [WINDOWS], DEVICE entry in the WIN.INI file! VB will instantly use the new printer, when the next Printer.Print command is issued. If you get the old printer string before you rewrite it (GetProfileString API call), you can set it back after using a specific printer. This technique is especially useful, when you want to use a FAX printer driver: Select the FAX driver, send your fax by printing to it and switch back to the normal default printer.

[Hajo Schmidt (hajo@bwl.bwl.th-darmstadt.de)]

It is recommended (and polite, as we're multitasking) to send a WM_WININICHANGE (&H1A) to all windows to tell them of the change. Also, under some circumstances the printer object won't notice that you have changed the default printer unless you do this.

```
Declare Function SendMessage(ByVal hWnd As Integer, ByVal wParam As Integer, lParam As Any) As Long
Global Const WM_WININICHANGE = &H1A
Global Const HWND_BROADCAST = &HFFFFFF
' Dummy means send to all top windows.

' Send name of changed section as lParam.
lRes = SendMessage(HWND_BROADCAST, WM_WININICHANGE, 0, ByVal
"Windows")
```

[Nic Gibson (nic@skin.demon.co.uk)]

Any tips for speeding up VB?

Who said "code in C"???? ;-)

- a. When SHOWing a form with lots of bound controls, have a blank frame covering everything. Then, in the Form_Activate event, set the Frame.Visible = False. This greatly speeds the display of the form and hides ugly thrashing as the data controls initialize.
[Christopher Biow (biow@cs.umd.edu)]
- b. Try to keep any Global definitions to a minimum. Massive numbers of global variables really seem to slow VB Windows down (besides chewing up memory). In other words, if you've pasted a lot of stuff from the globals.txt file, trim all definitions and variables you don't use in your application.
- c. Keep the total number of controls and forms used to a minimum (you've probably already guessed that).
- d. Keep fancy graphics to a minimum (another one you know).
- e. Try "pre-processing" in the background (using Do_Events). Doesn't really speed anything up, but often there is a lot of "idle" time while the user is selecting menu's, buttons and such - if you can do some calculations, image loading or whatever during this idle time your user perceives the application is faster than it really is.
- f. Hide often-used forms rather than unloading them. Unloading saves memory, but it takes longer to re-load a form than to simply "un-hide" it.
[Tips b to f by Galen Raben (galenr@gr.hp.com)]

The following tip is along the same lines, but with a code sample. They are provided by Andy Dingley (dingbat@codesmith.demon.co.uk):

You're limited by the system as to how quickly you can go from calling frmMyForm.Show to being able to type into the controls, but you can make the form *appear* to display faster. One technique is to keep forms loaded, and just switch their visibility on and off. This is heavy on resource usage, and doesn't help for the first time they're shown.

Most forms have some processing (eg. querying a table to fill a list box) that goes on when they're first opened, and this is what causes the most serious delay. It's possible to display the form, make its controls appear on screen, then do the slow processing before finally making the form "live". As the user can see things happening, the perceived delay is less obvious.

Include the following code in your form:

```
Option Explicit
Dim FirstActivation as integer

Sub Form_Activate
    DoEvents 'Allow the _Load event to be seen on screen
    If FirstActivation Then
        ' Do all the slow loading stuff here
        If FillComboBox <> 0 Then
            Unload Me 'If it all goes horribly wrong, then you
                    'can call Unload from an _Activate event
                    '(Which you can't do from the _Load event)
        End If
        FirstActivation = False
    End If
    Screen.MousePointer = DEFAULT
End Sub
```

```
Sub Form_Load
    FirstActivation = True
End Sub
```

Show the form by using:

```
Screen.MousePointer = HOURGLASS
frmMyForm.Show MODAL
```

Bruce Garrett (bruceg@access2.digex.net) had the following tips from his VBITS 93 notes:

- Polling a control for its properties directly is 10 to 20 times slower than placing the property values you need into variables and testing the value of the variables.
- Swap tuning: Modules are not loaded until used; put related code in the same modules, reduce the number of intermodule calls and keep modules small.
- Binary file I/O is faster than Text/Random.

There was also a lot of discussion about "apparent" speed i.e: how it looks on the screen as opposed to how fast it's chugging internally. It was noted that the cute little flashing menu items and exploding windows in the Mac amounted to a little razzle-dazzle to distract you from how long it took to actually load something and get it on the screen. Keeping all your forms loaded but hidden until needed was suggested. Also the use of progress indicators and a simple quickly loaded and drawn startup form. Also preloading data you expect to need.

How do I speed up control property access?

Instead of using a property in a loop, you will be better off using a normal variable in the loop and then assign the variable once to the property afterwards. Also, when reading a property, you should read it once into a variable instead of using it in a loop.

Sometimes it is not possible to simply put contents of a property into a variable. For example, if you are using a list box or you need to conserve memory. In these cases you can send the WM_SetRedraw message to the control to prevent redrawing. You can typically increase the speed 6-10 times - or even more.

```
'Add the following declares:
Declare Function SendMessage Lib "User" (ByVal hWnd As Integer,
ByVal wParam As Integer, ByVal lParam As Any) As
Long
Const WM_SetRedraw = &HB

'Add this to your code:
Result% = SendMessage(Text1.hWnd, WM_SetRedraw, 0, 0)
'redraw off
'Do your stuff here!
Result% = SendMessage(Text1.hWnd, WM_SetRedraw, 1, 0)
'redraw on
```

This same method applies to list boxes and other controls.

How much gain in performance will I get if I write my number crunching routines in instead of Visual Basic?

Probably the best solution to the number crunching problem is to write the number crunching routines as a custom control or a DLL, and plug it into a VB app. VB interface handling is not significantly slower than, say C++, and most of the wait is associated with Windows.

Some real world experience speaks volumes about this one:

I wrote some time consuming code in VB to solve a combinatorical (does this word exist in English?) problem. The code consists of one main recursive function, which calls itself very often. It took a night to compute a certain problem. I was rather disappointed and then decided to write the central routine in C++. It was a 1:1 transcription. The routine was compiled with the MS C++-Compiler. It took only 22 Minutes for the same problem. Amazing, isn't it? The routine doesn't do any floating point arithmetic, only integer, and handles some arrays. The PC was a 33MHz 486. And the second amazing thing is, that a IBM RS6000 (560)-Risc-machine needed 17 Min for the same code. I was the only one on the machine. I thought it should be much faster. The MS C++ seems to make very fast, optimized code. The optimization was configured to make fast code.

[Christoph Steinbeck (steinbeck@uni-bonn.de)]

How do you make a TEXTBOX read only? Or, how do I prevent the user from changing the text in a TEXTBOX?

There's a lot of ideas on this one. You can grab the `_KeyPress` and `_KeyDown` events and set them to zero. However, the best idea is to use the Windows API `SendMessage` function to tell the control to become read-only:

```
'After making the following declarations...
Global Const WM_USER = &H400
Global Const EM_SETREADONLY = (WM_USER + 31)
Declare Function SendMessage Lib "User" (ByVal hWnd As Integer
ByVal wParam As Integer, ByVal lParam As Any) As
Long

'Then Try:
SendMessage(Text1.hWnd, EM_SETREADONLY, 1, 0)
[Pete Jones (pjones@csi.compuserve.com)]
```

This will still allow the user to copy *from* the text box. If you need to disable this (why?), steal the Ctrl-C in the `_KeyPress` event.

How can I create a VBX?

VBXs (Visual Basic eXtensions) are practically always written in C (Borland C++, but mainly MS VC++). You should refer to the `_Control Development Guide_` (in VB Professional Features Vol. I) and any relevant documentation for your compiler. Followup questions should normally be directed to `comp.os.ms-windows.programmer.*` or `comp.lang.c*`.

There are some example VBX's with C code supplied with VB3 Pro. You'll find them under the directory `[VB]\CDK`.

How do you change the system menu (on the Control-Menu Box)?

You can turn off the minimize and maximize menu options by changing properties, but what if you need to remove the "close" option?

```
'Make the following declares.
Declare Function GetSystemMenu Lib "User" (ByVal hWnd As Integer,
ByVal bRevert As Integer) As Integer
Declare Function RemoveMenu Lib "User" (ByVal hMenu As Integer,
ByVal nPosition As Integer, ByVal wFlags As Integer) As Integer
Global Const MF_BYPOSITION=&H400

'Use the following code to remove the "close" option.
SystemMenu% = GetSystemMenu (hWnd, 0)
Res% = RemoveMenu(SystemMenu%,6, MF_BYPOSITION)
'(also remove the separator line)
Res% = RemoveMenu(SystemMenu%,6, MF_BYPOSITION)
```

Adding menu items to the control menu is more complicated, since you need to respond to the events triggered when the user selects the new options. The newest Message Blaster (msgblast.vbx, see details in beginning of FAQ about how to get files) contains example code.

How do I play MID, WAV or other multimedia files?

Use the MSMCI.VBX, provided with VB/Win Pro 3.0. You can also declare and call the MM-functions manually:

```
Declare Function mciExecute Lib "MMSystem" (ByVal FileName as  
String) As Integer  
  
Sub Form1_Click ()  
    iResult = mciExecute("Play c:\windows\mkmyday.wav")  
End Sub
```


How can I call a 'hidden' DOS program from VB?

If you run a DOS program minimized using the SHELL command, it will never complete. This is because DOS tasks by default are NOT setup to run in the background. The easiest way to get around this is to make a PIF file for the program you need to run with the "Background" option checked. Then SHELL to the PIF file to run the DOS program and it will return control to your VB application when it terminates.

Tip: If you edit or replace the _DEFAULT.PIF file in the Windows directory to allow execution in background, this will apply to all DOS boxes that is not run with it's own .pif!

How do I do drag & drop between applications?

MSGBLAST.ZIP (the famous Message Blaster by Ed Staffin and Kyle Marsh) available on Cica and mirrors tell you *everything* you want to know about this and other advanced stuff. This is now (inexpensive) shareware, but the older freeware version is still supposed to be available. Get the file mentined above for more info.

Short glossary for the confused ones :-)

Drag & Drop Client: the form you drop objects to/on
Drag & Drop Server: the form you drag object(s) from

How do I use GetPrivateProfileString to read from INI files?

There's a good example of accessing *.INI files in the Knowledge Base, but here's the basic idea:

```
'You declare these API function as usual:  
Declare Function GetPrivateProfileString Lib "Kernel" (ByVal  
lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpDefault As  
String, ByVal lpReturnedString As String, ByVal nSize As Integer, ByVal  
lpFileName As String) As Integer
```

```
'Then in your code you do like below:  
strIniFile = "WIN.INI"  
strSection = "MyProgram"  
strKey = "Language"  
strDefault = "English"  
iLength = 70  
strReturn = String$(iLength, " ") 'Pad the string first!  
iResult = GetPrivateProfileString(strSection, strKey, strDefault,  
strReturn, iLength, strIniFile)
```

WARNING: Be aware that there was an ERROR in the Windows 3.1 API documentation that came with VB. Here's the scoop:

Knowledge Base article Q110826 (DOCERR: GetPrivateProfileString Declaration Incorrect in API) corrects a documentation error for the GetPrivateProfileString function call as described in the Windows version 3.1 API Reference help file that shipped with Microsoft Visual Basic version 3.0 for Windows. The CORRECT declaration is as follows:

```
Declare Function GetPrivateProfileString Lib "Kernel" (ByVal  
lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpDefault As  
String, ByVal lpReturnedString As String, ByVal nSize As Integer, ByVal  
lpFileName As String) As Integer
```

Note that the "ByVal" keyword was omitted from the second parameter in the online reference. This means that the function is passing the second parameter (lpKeyName) by reference. It needs to be passed by value.

The most common problem that occurs when using the incorrect declaration is that when the function is called, it returns a copy of "lpdefault" in the "lpReturnedString" parameter instead of the actual value referenced by KeyName.

OOPS: As P. Wierenga (pwiereng@sol.UVic.CA) told me, the same doc error applies to Writeblablabla:

DOCERR: WriteProfileString Declaration Incorrect in API Article ID: Q115328

The correct declaration is as follows:

```
Declare Function WritePrivateProfileString Lib "Kernel" (ByVal lpApplicationName As String,  
ByVal lpKeyName As Any, ByVal lpString As Any, ByVal lpFileName As String) As Integer
```

How do I implement Undo?

For most controls, you will have to keep track of changes yourself. There's no magic involved, just some coding. However, if you use the standard Text box or Combo box, Windows provides a "free" undo function for you!

```
'Do the following declares:  
Declare Function SendMessage Lib "User" (ByVal hWnd As Integer,  
ByVal wParam As Integer, ByVal lParam As Any) As  
Long  
Global Const WM_USER = &h400  
Global Const EM_UNDO = WM_USER + 23  
  
'And in your Undo Sub do the following:  
UndoResult = SendMessage(myControl.hWnd, EM_UNDO, 0, 0)  
'UndoResult = -1 indicates an error.
```

How do I create a window with a small title bar as in a floating toolbar?

Download the MSGBLAST VBX from <ftp.microsoft.com> (filename "8- 3.ZIP") or (better) from <ftp.cica.indiana.edu>. The example files provide an example of a form with a small title. When you see it, you'll understand why I haven't include a full explanation here!

What is Pseudocode?

VB/Win does not generate machine code like most compilers do. Instead it creates what is called pseudocode (a real misnomer, IMO). A good explanation is given below:

A bit of history: the original P-code was an instruction set for a "virtual Pascal" machine. This came with a portable Pascal compiler written at ETH in Zuerich. The portable compiler produced instructions for this phony machine which had an instruction set ideally suited to the stack and heap management of Pascal. To execute portable Pascal programs, you had two choices: either write an interpreter for P-code, or translate the small set of P-code instructions (there were about 80) into assembler; assemble it; and run it at native speed. Thus "P-code" originally stood for "Portable" or "Pascal" code.

The broader meaning, "pseudo-code" came later. P-code was widely popularized by the UCSD Pascal system, a small workstation that was implemented entirely in P-code and interpreted. It was sold for some years, and one company even re-did the microcode for a PDP-11 microchip to interpret P-code. The original Borland Turbo Pascal had obvious similarities to the UCSD system although it was not interpreted. The dialect was virtually identical.

Today P-code is used extensively in Microsoft apps, for two reasons. First, it is much more compact than native code; so the apps are smaller. Second, having an interpreter at the core of an app makes it much easier to customize and extend. That is why VB is becoming the heart of the MS major apps. It is simply not true that P-code apps run much slower than native apps. The slowdown is determined by the granularity of the interpreted routines. If every little thing is an interpreted op, the slowdown might be as much as 3-to-1 for the 80x86 architecture, or about 2-to-1 for the Motorola 68000 family (which is better suited to writing interpreters).

But in practice, modern P-code systems have large-scale instructions, each of which is executed by a big compiled subroutine. These subs run at native speed, so the overhead of the interpreter is occasional at worst. [Roger E. Ison (r_ison@csn.org)]

It is also possible that since the code may not need recompilation to run on other platforms *iff* the runtime interpreter is first ported, VB applications can become very portable. This depends on Microsoft's long-term plans.

A note on the word "pseudocode": I wrote above that it is a misnomer, and I stand on that. Pseudocode is *really* the pascal-like (mostly) explanation of an algorithm that is intended for human readers, not computers. But since somehow the term pseudocode stuck to the pseudo-machine-code created by VB the word is used here.

Does VB support pointers to functions?

No, it does not. [George Tatge (gat@csn.org)]

How do I program the Novell NetWare API from VB?

Tom Tregilgas (Tom.Tregilgas@InfoB.unisg.ch) had a lot of information on this one. Normally I leave it to the other FAQ parts to list books & how-to-get-info's, but since this topic is very specific and more NetWare than VB I include all the stuff here for your convenience:

If you are interested in seeing how Visual Basic can be used for NetWare programming, obtain the following files from your friendly neighborhood Novell FTP Mirror site.

Mirror sites are (according to ftp.novell.com):

<u>Mirror Site</u>	<u>FTP Address</u>
Novell Germany	ftp.novell.de
Netherlands	ftp.rug.nl
United Kingdom	ftp.salford.ac.uk
Logan, Utah	netlab2.usu.edu
New Zealand	tui.lincoln.ac.nz
Tuscaloosa, Alabama	risc.ua.edu
Ottawa, Ontario, CA	novell.nrc.ca
Boston, Mass	bnug.proteon.com

novlib\11\nivb.zip Netware Interface for Visual Basic
novlib\11\nwtest.zip NetWare Test for Visual Basic

There are also two Novell App Notes on the subject of using NetWare with Visual Basic (although this is NOT supported by Novell...) which are:

October 92 Interfacing Visual Basic for Windows and NetWare
July 93 A NetWare Interface for Visual Basic

The AppNotes can be obtained by contacting the Novell Research Order Desk, FAX: +1 303 294-0903, Voice 800 377-4136, +1 303 297-2725. Address as follows:

Novell Research Order Desk
1601 Park Avenue West
Denver, CO 80216-5199

AppNotes are \$95/year (\$135 outside US)

Here are a few books which might help you out to figure the calls out:

Windows Development on NetWare Systems, Lori Gauthier and Sue Whitehead (c) 1994, Windcrest, Blue Ridge Summit, PA 17294-0850 (McGraw-Hill) \$34.95 Comes with disk This book also tells you how to "upgrade" to the currently supported SDK calls

NetWare System Interface Technical Overview, Novell (c) 1990,1989 (Addison-Wesley), \$32.95
(describes Novell's C Network Compiler API's)

Visual Basic Programmer's Guide to the Windows API, Daniel Appleman Ziff-Davis Press, 5903
Christie Ave, Emeryville, CA 94608, \$34.95 Comes with disk

It should be mentioned that the APIs included with the NIVB are not current, and for this purpose, you should get the Novell SDK kit. Also, Novell will not support NIVB, but you can sometimes get some help from Compu\$erve, or from others on the Infobahn <g>

Good luck!

p.s. It behooves you to become a member in the PDP (Professional Developer's Program) since you get the AppNotes (& Bullets!) for...free.

p.p.s. Novell does NOT support the NIVB...

p.p.p.s. Also, no docs come with it. You'll probably need the Client C SDK kit to be able to really use the code.

p.p.p.p.s. To make things even better, the calls in NIVB are fairly old, and not of the Client C SDK kit variety. However, there are books which could help you out, e.g. "NetWare System Interface Technical Overview", by Novell. ISBN:0-201-57027-0, published by Addison-Wesley Publishing co, \$32.95 US, \$42.95 in Canada.

Update:

AppNotes are dead, however, Developer Notes live on. There is one article about NetWare programming with Visual Basic here:

July/Aug 94 NetWare Programming in Visual Basic: Using Apiary's NetWare Client SDK for Visual Basic

Visual Basic 4 News [**]

Here is the news we have observed (and which was a guess in previous versions of this FAQ)

a. **Two versions: 32-bit and 16-bit**

Like Visual C++ 1.5, VB 4 comes with compilers for the 16-bit API from Windows 3.1 as well as a real 32-bit compiler (or pseudocompiler) for Windows 95 (and NT) bundled together.

b. **New features inherited from VBA**

- 1) VB 4.0 is really VB Applications Edition 2.0
- 2) Line continuation character " _" (space+underscore)
- 3) WITH statement, known from Pascal, to save typing and make code cleaner. An example:

```
With Form1
    With Text1
        Bold = true
        FontName = "New Times Roman"
    End With
End With
```

- 4) FOR EACH .. NEXT statement allows you to make changes to a group of objects at once better than FOR..NEXT. Syntax:

```
For Each element In group
    [statements]
    [Exit For]
    [statements]
Next [element]
```

- 5) OPTIONAL statement allows you to leave some Variant parameteres undefined when calling a user-defined sub or function. Also an ARRAY function that returns an array form a list supplied as parameteres, and a special optional ParamArray optional parameter of Variants.
- 6) Boolean data type. Coded as 16-bit (2-byte), but can only have TRUE or FALSE value.
- 7) Byte data type (8-bit). Can have values from 0 to 255.
- 8) FUNCTION can be of user-defined type.
- 9) OLE Automation allows detailed control over objects taken from ie. Excel or Word.
- 10) Reusable Objects and Collections, called Classes
- 11) Property Procedures allows you to add custom properties to form and some other modules.
- 12) New enhanced development environment, which includes an object browser and allows add-ins. Pro edition also allows creation of your own add-ins. (For some strange reason, the 32-bit version of this does not use Win95 file dialog boxes!)
- 13) Conditional compilation: allows one source for 32- and 16-bit versions.
- 14) Data bound DBList, DBCombo, and DBGrid controls! VB 4 uses Jet 2.5 engine.

c. **VBX is dead. Long live OCX!**

You may not feel for celebrating this either, but the 32 bit VB 4 will not support the old 16-bit VBX'es. The 16-bit version of VB 4 will support them for backwards compatibility, but be aware

that OLE 2 and OCX is the way of the future, at least if Microsoft gets it as they want.

How do I call help files from a VB program? [**]

There are 2 ways you can display help files:

- 1) In code
 - a) Using the common dialog control
 - b) Making a winhelp api call (see below for examples)

- 2) Pressing F1 when App.HelpFile is set to your help file.

Note - if a control's HelpContextID property is set to the context id of a help topic, that topic will be displayed if the control has focus and the user presses F1, otherwise the contents is displayed.

(see Mapping Context-Sensitive Topics, Chapter 6 pg. 117 of the Professional Features Book 1 for information on setting the context ids for your help file. Many help authoring tools will generate the context ids for you.)

- Include constant.txt in your project
- Set App.HelpFile = "c:\yourfile.hlp"
- Declare the following function in a .bas module (make sure it is all on one line):

'Function for display windows help files

```
Declare Function WinHelp Lib "User" (ByVal hWnd As Integer, ByVal  
lpHelpFile As String, ByVal wCommand As Integer, dwData As Any)  
As Integer
```

COMMON DIALOG:

Note - The HelpContext property can only be assigned an integer. The equivalent WinHelp Api call can use a long. In this example the context id is 1000

jump to a context id within help file.

```
CMDialog1.HelpFile = "c:\yourfile.hlp"  
CMDialog1.HelpCommand = HELP_CONTEXT  
CMDialog1.HelpContext = 1000  
CMDialog1.Action = 6
```

bring up contents

```
CMDialog1.HelpFile = "c:\yourfile.hlp"  
CMDialog1.HelpCommand = HELP_CONTENTS  
CMDialog1.Action = 6
```

bring up search window

```
CMDialog1.HelpFile = "c:\yourfile.hlp"  
CMDialog1.HelpCommand = HELP_PARTIALKEY  
CMDialog1.HelpKey = ""  
CMDialog1.Action = 6
```

quit help

```
CMDialog1.HelpFile = "c:\yourfile.hlp"  
CMDialog1.HelpCommand = HELP_QUIT  
CMDialog1.HelpContext = 0  
CMDialog1.Action = 6
```

WINHELP API:

In the following examples iRetCde is declared as an integer.

Jump to context id (e.x., context id is 1000)

```
iRetCde = WinHelp(Me.hWnd, "c:\yourfile.hlp", HELP_CONTEXT, ByVal  
CLng(1000))
```

bring up contents

```
iRetCde = WinHelp(Me.hWnd, "c:\yourfile.hlp", HELP_CONTENTS,  
CLng(0))
```

bring up search window

```
iRetCde = WinHelp(Me.hWnd, "c:\yourfile.hlp", HELP_PARTIALKEY,  
ByVal "")
```

quit help

```
iRetCde = WinHelp(Me.hWnd, "c:\yourfile.hlp", HELP_QUIT, CLng(0))
```

[Blake Miller (bemiller@amoco.com)]

How do you change the icon and otherwise manipulate the DOS box?

Enclosed is the results of my digging around to enable me to change the icon of a DOS box launched with the SHELL function. It illustrates most aspects of dealing with DOS boxes.

Example of launching PIF file minimized and running it in the background (Needs Execute Background box checked in PIF file), and assigning the PIF file a new Icon rather than the default DOS Box Icon.

(Note: it seems this method changes the icon of ALL the active DOS boxes)

```
x-----
-----x
Option Explicit

Global Const SWP_NOSIZE = 1
Global Const SWP_NOMOVE = 2
Global Const SWP_NOACTIVATE = &H10
Global Const SWP_SHOWWINDOW = &H40
Global Const SWP_HIDEWINDOW = &H80
Global Const SWP_FLAGS = SWP_NOMOVE Or SWP_NOSIZE Or
SWP_NOACTIVATE
Global Const SWP_SHOW = SWP_SHOWWINDOW Or SWP_FLAGS
Global Const SWP_HIDE = SWP_HIDEWINDOW Or SWP_FLAGS
Global Const HWND_BOTTOM = 1
Global Const GCW_HICON = (-14)
Global Const GCW_HMODULE = (-16)

Declare Function GetModuleUsage Lib "Kernel" (ByVal hWnd%)
As Integer
Declare Function ExtractIcon Lib "Shell" (ByVal hInst%,
|ByVal lpExeName$, ByVal hIcon%) As Integer
Declare Function DestroyIcon Lib "User" (ByVal hIcon%) As
Integer
Declare Function FindWindow Lib "User" (ByVal lpClassName As
Any,
|ByVal lpCaption As Any) as Integer

Declare Function SetWindowPos Lib "User" (ByVal h%, ByVal hb
%,
|ByVal X%, ByVal y%, ByVal cx%, ByVal cy%, ByVal F%) as
Integer
Declare Function GetClassWord Lib "User" (ByVal hWnd%, ByVal
nIndex%)
|As Integer
Declare Function SetClassWord Lib "User" (ByVal hWnd%, ByVal
nIndex%,
|ByVal wNewWord%) As Integer

Sub LaunchPif(PifFile as String, IconName as String)
Dim Res As Integer 'Return value from API call
Dim MyInst As Integer 'Instance handle for this app
Dim PifIcon As Integer 'Pif Icon Resource name
Dim OldIcon as Integer 'Original Pif Icon resource
(default = 0)
Dim PifhWnd As Integer 'Dos Box Window handle
'Launch PIF file
```

```

PifInst = Shell(PifFile, 6)
'Get Instance handle for main MDI Window (assumed to be
frmMDI)
MyInst = GetClassWord((frmMDI.hWnd), GCW_HMODULE)
'Create new Icon resource by loading Icon from disk file
'Check Icon file exists
If Dir$(IconName) <> "" Then
    'Make sure string is null terminated
    IconName = IconName & Chr$(0)
    'Create Icon resource in this window
    PifIcon = ExtractIcon(MyInst, IconName, 0)
Else
    PifIcon = 0
End If
'Reset Default Icon on DOS Box to PifIcon
'Check PIF file still running and Icon resource created
If GetModuleUsage(PifInst) <> 0 and PifIcon > 0 Then
    'Get Pif Window handle
    PifhWnd = FindWindow(0&, "Window name in PIF file")
    'Set Icon handle to PifIcon resource
    OldIcon = SetClassWord(PifhWnd, GCW_HICON, PifIcon)
    'Hide and show window to get a redraw of the Icon
    Res = SetWindowPos(PifhWnd, HWND_BOTTOM, 0, 0, 0, 0,
SWP_HIDE)
    Res = SetWindowPos(PifhWnd, HWND_BOTTOM, 0, 0, 0, 0,
SWP_SHOW)
End If

'Wait for PIF file to complete
Do While GetModuleUsage(PifInst) <> 0
    DoEvents
Loop

If PifIcon > 0 then
    'Reset Icon to original
    Res = SetClassWord(PifhWnd, GCW_HICON, OldIcon)
    'Tidy up by removing PIF Icon resource
    Res = DestroyIcon(PifIcon)
Endif
End Sub
x-----
-----x

```

[Dr David Baird (BairdD@AgResearch.CRI.NZ)]

How do I make the mouse cursor invisible/visible?

Use the API call ShowCursor(False) or ShowCursor(True). Just be aware that the Windows cursor is a shared object, so if your process hides it then it must also redisplay it as well. Also the function is not truly "True" or "False" in nature - it is LIFO, so that if your process has for some reason set it "False" multiple times then you must set it "True" the same number of times in order to re-display the cursor. Hard to explain but play with it - you'll see what I mean...

[Galen Raben (galenr@hpgrla.gr.hp.com)]

How do I create controls dynamically (at run-time)?

Search of VBKB_FT on "control* near array*" finds a number of articles, including Article ID: Q79029 "Creating Nested Control Arrays in Visual Basic" This is THE article to read to understand control arrays (plus a number of other neat concepts). In particular, look at the procedure "loadall_click".

The key is that you need to create the first instance of your_control (ie. your_control(0)) at design time.

[Ayn Shipley (ashipley@hookup.net)]

Steps for securing an Access 2.0 database [**]

What I started with:

- A database that I wanted to make secure.
- Access 2.0 installed on my hard drive.
- No Access permissions or accounts ever played with under Access.

- 1) I ran MS Access Workgroup Administrator and created a new SYSTEM.MDA and assigned a unique workgroup id.
- 2) Ran Access and created a new database, DB1.MDB.
- 3) Changed the password for the Admin user using Change Password under the Security menu. Changed the password from nothing to "PASSWORD".
- 4) Clicked on Users under Security. Added a new user, "JOEUSER" and made it a member of the Admins and Users groups.
- 5) Closed Access, deleted the new database I had just created, DB1.MDB, and restarted Access. Logged in as the new user, "JOEUSER". Created a new database named DB1.MDB. Changed the password from nothing to=7F "PASSWORD".
- 6) From the File menu, clicked Add-ins and then Import Database. Imported my database, REAL.MDB, that I wanted to make secure.
- 7) Went to Permissions under the Security menu and turned off all permissions for all Groups and all Users for the database, all tables, forms, macros, etc. The only permissions I left on was full permissions for all objects for the new user, "JOEUSER". After changing the permissions, I closed Access (note that the database I just used was DB1.MDB).
- 8) I made a backup of my database REAL.MDB and renamed DB1.MDB to REAL.MDB.
- 9) Opened my VB code and added the line:

```
SetDefaultWorkspace "JOEUSER", "PASSWORD"
```

right before the line where I open the database.

- 10) Copied the SYSTEM.MDA from my C:\ACCESS directory to the directory where my VB code is. (Note, I do not have an INI file for my program).
- 11) Ran my VB program and my database can be opened by my program and nobody else. The SYSTEM.MDA has to be delivered with my program however.

See SetDefaultWorkspace and SetDataAccessOption in VB help for more information on how to set up the data access parameters in an INI file. Make sure you add error handling for a missing or messed up SYSTEM.MDA file.

I think this is a good basic way to prevent people from looking/changing your database. Possible flaws are looking at a dump of the .EXE file and finding the name and password used to open the database (there are various workarounds for this) and also running a program to look at parameters passed when opening the database (don't know if this is possible and can't think of a workaround if it is).

[Justin F. Smith (jsw117@psu.edu)]

How do I set the Windows wallpaper at runtime?

I'm surprised this isn't in the FAQ yet. [ED: now it is!] You need the SystemParametersInfo API function and the SPI_SETDESKWALLPAPER constant.

For this you will need the following constants and function declaration...

```
Const SPI_SETDESKWALLPAPER = 20
Const SPIF_SENDWININICHANGE = &H2
Const SPIF_UPDATEINIFILE = &H1
Declare Function SystemParametersInfo Lib "User" (ByVal uAction
As Integer, ByVal uParam As Integer, lpvParam As Any, ByVal
fuWinIni As Integer) As Integer

'You then call the function as follows...

Result% = SystemParametersInfo(SPI_SETDESKWALLPAPER, 0, ByVal
BMPFile$, SPIFlags%)
'where SPIFlags% is 0 if the change is not to be made permanent,
or
'(SPIF_UPDATEINIFILE Or SPIF_SENDWININICHANGE) if it is to be
carried across
'into future Windows sessions.
```

NOTE: Please be certain to include the ByVal keyword before the bitmap filename, as this argument is declared as Any, NOT as ByVal String.

[Luke Webber(webber@werple.apana.org.au)]

What is the Windows API?

The Windows API (Application Program Interface) is a collection of Dynamic-Link Libraries (DLLs) that do most of the common things in Windows. Calls to the Windows API gives you access to routines that do things like drawing menu bars, manipulating bitmaps, playing sound files, and pretty much every other function of Windows.

How do I call a DLL?

Basically, you declare a DLL procedure which you can call in your VB program which in turn passes data to and/or retrieves data from the DLL. You should read the section of the VB manual that talks about calling DLLs ("Chapter 24 Calling Procedures in DLLs" in the VB 3.0 Programmer's Guide). This chapter covers the basics of using the Windows API DLLs and calling DLLs in general. Beyond that you may want to find a good book on this subject since it is too large to cover here (see the Book Listing in the Appendix of the General FAQ - Part 1). Don't be too intimidated! Using DLLs (especially many of the Windows API functions) is quite easy, once you learn how to call them. In fact, many of the newer DLLs include VB-compatible modules!

What about DLL calls that require callbacks?

VB does not support callbacks, but various extensions can help.

Dan Appleman's "Visual Basic Programmer's Guide to the Windows API" comes with a floppy disk which code samples and tools. It also includes a VBX which supports the callbacks which many API calls require. Dan is also founder and president of Desaware which sells more extensive tools, including SpyWorks, for VB developers.

[Walter Hill (whill@netcom.com)]

Tips for calling DLLs (such as the Windows API)

- a. Using the BYVAL keyword is critical. Using it when you're supposed to call by reference and (more common) not using it when you are to give a value to the external function are the single most common mistakes. Wrong calling convention can often result in a General Protection Fault (GPF) or, even worse, corruption of another applications' data.
- b. Check return and parameter types. For return types, a C function declared as "void" means you use a Sub not a Function.
- c. Initialize strings by padding it to the necessary length! If you pass a string that is too short to the API it will happily write past the end of the string and possibly corrupt data.
- d. Use Option Explicit. A typing error that results in a bug in the VB source will occasionally cause a GPF when you call external code.
- e. It's a jungle out there! Check parameter values as there is no type checking outside VB. If you make a mistake, you'll often get a GPF.
- f. Save before you run! You may even need to restart Windows after a GPF, since DLL's often aren't unloaded properly. As a second option you can check out WPS (Windows Process Status) which is distributed with VB/Pro and placed in the CDK directory. This utility allows you to kick out any module (EXE, DLL) from memory (shooting yourself in the foot if you want to. WPS is a nice way to find out what DLLs are actually used, but save your work first!).

Why can't I use an index with my VB accessed database?

There is a mistake in the docs which says you can set the active index for a recordset. You can't. The data control uses the primary key for tables and physical order (I think) for dynasets.

[Nic Gibson (nic@skin.demon.co.uk)]

Why does my compiled VB database app generate an error when it ran just fine in the design environment?

You can thank Microsoft for documenting this topic so poorly. When you compile your VB database application, you must also have an INI file for it which provides the correct pointers to the appropriate database drivers. Therefore, if your application is named "INVOICE.EXE", you will need to have a properly configured "INVOICE.INI" file in your Windows directory. The file, EXTERNAL.TXT, that came with VB should explain all about it.

Is the Access Engine and Visual Basic Pro good enough for database work?

That, of course, depends. Generally the answer is "yes", but you may need some third-party add-on products.

These are the major weaknesses of VBPro's database functions:

- a. Limited data controls: No add, delete or search button; no bound list box or masked edit control and - the worst - no bound grid!
- b. No run-time query builder ("how good is your user in SQL?") or report builder.
- c. No direct advanced control of the Access 1.1 (or 2) Database Engine (ie. security, optimization, etc.).

The good news is that lots of companies are willing to sell you products which address one or more of the above weaknesses.

Also, if you build a database application with advanced database relations, it can be a good idea to build the database itself with Access and the front-end with VB.

How do you avoid the "Invalid use of null" error when reading null values from a database?

If you try to retrieve a null value (empty field) from a database, you will get the error: "Invalid use of Null". Here is one way to get around this problem:

I've worked around this problem with the following code:

```
TextBox.Text = MyTest.Fields("TestFld") & ""
```

This code converts the Null-Value into an empty string.

[Ralf Metzger (rmm@dragon.stgt.sub.org)]

What is "NULL"?

Contrary to popular belief, Null is not nothing. It's even less than nothing. 8^)

The VB documentation describes all the horrors of misunderstanding the infamous NULL. Since people don't read the documentation, we feel like informing that

```
If ThisVarIsNull = NULL then DoSomething
```

will **always** fail, and the DoSomething can't possibly be executed. You **must** use IsNull(ThisVarIsNull) which will return True if the var is Null (phew!).

If you want to find out why someone came to think of this strange value, read some relational database theory.

How can I access a record by record number?

Use a counter or index field and access the record with this.

It is *impossible* to ask a relational database system to give you ie. "field number 3 in record number 10" since by definition a relational database does not have row or column numbers. Databases allowing direct access like that is not even remotely relational.

Access (and therefore, VB) is about as close to a real relational database system as you can get.

Tips for VB database programming:

- a. Use Access and QBE. Once it's "working" (even if the parameters are hardcoded), then open up View.SQL and copy the stuff from the SQL window into your VB code. If you need to insert VB variables, try testing this under Access by using parameters instead. They're then nice & easy to spot when it comes to converting into VB - I always call my parameters "PR_xxxx", so I can just search my VB code for this to find any instances that I've missed.
- b. It never works first time. So put an error handler into your VB code that copies the contents of SQLStr onto the clipboard, should the query fail. Now it's quick & easy to switch back to Access, find a scratch query and paste the erroneous SQL into that. It's *much* easier to debug a SQL query in Access, after the variables have been merged in, than it is to do it blind from VB.
- c. Use carriage returns to break up your SQL. One before each reserved word is sensible. They're not significant in SQL. I assume you're not stupid enough to put them in the middle of field names - unfortunately Debug.Print is!
- d. When merging in the contents of a variable (building a SQL query in a VB string), it should *always* be surrounded by an ampersand and 3 double quotes, or an ampersand and 2 mixed quotes, depending on your local conventions:

```
SQLStr = SQLStr & "WHERE Username <= """" & Username$ """" "
```

or

```
SQLStr = SQLStr & "WHERE Username <= " & Username$ " " "
```

- e. If you're using dates, then it will *always* be one quote, a hash and an ampersand:

```
SQLStr = SQLStr & "WHERE Start_Date <= #" & Format$(CutOffDate,"Long Date") & "# "
```

- f. Another tip with dates is to format them with the long date format, not the short date. This is then safe against the transatlantic reversal of month & day position.
- g. If you're merging in a field/table name, enclose it in square brackets. That way the SQL will still be valid if the variable contains spaces:

```
SQLStr = "SELECT * FROM [" & TableName$ "]" ;"
```

When building SQL strings in VB, then you'll often do this on several lines, concatenating SQLStr with the new string. If you leave a space at the end of every string, then you can guarantee you won't have problems with the text from successive lines running into each other.

- h. If you're using Access 1, you'll keep running into the 1024 character limit on the length of a SQL string. Keep the table & field names short, especially if many JOINS are concerned. Using underscores in names is shorter than spaces, as you don't need the extra 2 characters for the square

brackets around them. If your SQL is slightly too long, then you'll probably see a "Missing semicolon" error, even though the semicolon is obviously there (To you, anyway !).

i. Making a QueryDef is a complicated process that is often slower than executing the query ! Don't mess with the .SQL property, as that is equally slow (Access needs to do a lot of work to turn SQL into its internal query format). Two ways around this: Use ready-built queries, written with Access. If you need to merge in values from variables, then use a query with parameters. Setting parameter values is quick to execute.

j. If you really need to build SQL on the fly -- you need to build an ad hoc query, or to supply table or field names (which can't be done with query parameters), then try using:

`database.Execute SQLStr`

As this doesn't build a QueryDef, then it's quick.

[All tips a to j by Andy Dingley(dingbat@codesmth.demon.co.uk)]

How come I get a "No Current Record" error when I use a Data Control on an empty table?

Well, this is a "feature" courtesy of Microsoft. KB article Q106494 explains this in detail. Basically, the workaround is to add an empty record to the table before the user can do anything (or before you try to do any Moves on the Table).

[George Tatge (gat@csn.org)]

How can I speed up my VB database application?

KB article Q109830 gives some hints. Things you should do include:

- Use Snapshots when possible.
- Use transactions whenever possible.
- Use Dynasets when possible.
- Use SQL action queries when possible.

[George Tatge (gat@csn.org)]

How about Access 2.0 compatibility?

You need the compatibility layer availability. The file COMLYR.EXE is in the MSBASIC library on CompuServe. This file provides all the items necessary for compatibility between VB 3.0 and Access 2.0.
[Fred Griffin (72321.3230@compuserve.com)]

The file COMLYR.EXE can be downloaded from <ftp.microsoft.com>. It is located in the directory /softlib/msfiles.

How do I get a bitmap picture in a field in an Access database?

See p.466 of the Visual Basic (3.0) Programmer's Guide. It contains a section called "Using Bound Picture Box and Image Controls". Basically you have to bind the VB PictureBox to a field in the Access DB, set the .Picture property in the PictureBox, and then move to the next record or something. VB will then store your picture in Access in a form in which it can be retrieved by VB in the future.

If you store the pictures in Access directly (using Access), VB won't be able to read them (using VB 3.0 and Access 1.1).

You can also store the picture's filename as a text field in the database and use LoadPicture() to load that file into the VB PictureBox. [Tim Shea (shea@marcam.com)]

What are some tips for using Setup Wizard?

There were loads of bugs in the setup utilities supplied with VB3. Be sure to get the newest version of SETUPKIT (usually called SETUPK.EXE or -.ZIP). It is available from the sources listed in the beginning of this document, and in the General FAQ.

Alternatively, if you have the older versions, you may have to manually remove the line referring to OLE2UI.DLL in the file SETUPWIZ.INI. See later in this document for dates of newest files on ftp.microsoft.com. Follow the instructions in SETUPK.TXT exactly. The files actually belong in two separate directories. Not placing them correctly can create strange and unusual side effects -- none of them good(!)

Set all involved EXE, DLL and VBX files to Read-Only so that the setup program doesn't modify them.

[Charles F. Mulks (21667cfm@msu.edu)]

A *very* good tip. Actually, make all executables on your system read-only. If not, you can get a sharing violation if you try to run the same DOS executable twice at the same time.

Also, the source code for a SETUP program is *included* with VB3 Pro. It is quite trivial to tailor it to your specific needs.

The question remains: Is SetupWiz good? No! Good enough? Perhaps.

Are there restrictions on what I can distribute with my VB program?

The documentation tells what parts of the Visual Basic kit you can freely distribute: the VBX files, some DLL's and what the SetupKit includes on your distribution diskettes. Reading software license agreements may be more boring than asking the newsgroup, but is nevertheless a good idea. 8^)

There have been some rumours on the newsgroup that you can't redistribute programs written with VB freely. This is nonsense. All applications created with VB can be redistributed freely without royalties (as long as you don't distribute proprietary external files).

The rumours probably originated when Microsoft announced that they will not sell kits allowing third-party software to include the Visual Basic for Applications (VBA) system.

What alternatives to setup wizard do I have?

Perhaps the best one is to simply modify the setup app which is supplied with VB. Look in your VB directory for the setupkit\setup1 directory. There you will find everything you need to do a complete setup program. This sample setup is coded to install a few sample app files and create a program group. You can comment out those lines and change to your files and program mgr. group. There are also a few global variables you will want to change. All of this is contained in the comments in the code.

Using this, and the distribution information in the manual telling you about which files to distribute with your app will make things much easier than using the setup wizard (IMNSHO).

There are also several third party setup products available.

[George Tatge (gat@csn.org)]

Do I need to worry about users who have Progman replacements such as Norton Desktop and PC Tools?

Earlier versions of those products and some others do not respond properly to the DLL commands to create groups and items. More recent versions do. All you can do in this case is to include some information in your readme.txt file that instructs users of those products to shut them down and start up program manager before installing.

[George Tatge (gat@csn.org)]

Can I distribute my app without vbrunXXX.dll?

If you are sure that your users have it or can get it, you can easily distribute your app without vbrunXXX.dll. Simply remove the file from your distribution disk or zip file and ALSO remove it from the setup.lst file.

[George Tatge (gat@csn.org)]

Why won't my setup program install commdlg.dll et. al.?

There are a couple of DLLs that are almost always in use by windows. Commdlg.dll is the most common example. When faced with this problem, there is no easy way out. The full explanation is several pages long and beyond the scope of this FAQ. The general idea is as follows:

Your setup program will need to create a .BAT file to expand and then copy these files. Then, it will need to shutdown Windows (see ExitWindowsExec API call) and run the .BAT file. Then it will need to restart windows and continue your setup program. Your setup program should delete the temporary .BAT file that is no longer needed.

[George Tatge (gat@csn.org)]

Where do I install VBXs and DLLs?

PLEASE- this is one place where everybody's life is much easier if you will follow Microsoft's recommendations. All PUBLIC VBXs and DLLs should be installed in the windows/system directory! A "PUBLIC" DLL or VBX is any which can be purchased on the open market. In other words, if another VB programmer might possibly use the same VBX or DLL, install it in the system directory.

If you have written private VBXs or DLLs that will never be used by any program but yours, you can install them in the same directory where you install your application files.

There are lots of good reasons for doing this, but it makes a short novel to rehearse them all.

[George Tatge (gat@csn.org)]

Multiple identifiers after the DIM statement can be confusing

Some with background from Pascal can try the following

```
Dim iA, iB, iC as Integer
```

and think that all these 3 variables end up as Integer. In fact, the first two end up as default data type, normally Variant.

Instead you should do

```
Dim iA as Integer  
Dim iB as Integer  
Dim iC as Integer
```

which takes up more space, but gives you room to comment your variables (hint, hint); *or*

```
Dim iA%, iB%, iC%
```

which does the whole job.

"Clean up" your project before final EXE compilation.

When you are ready to compile your VB project into your 'finished' EXE, be sure to save the project files, exit VB, restart Windows, run VB, load your project and go straight to compiling. Otherwise, your EXE may be larger in file size than necessary due to 'garbage' getting included in the EXE. For some reason, VB does not fully clean up all of the previously used variables or objects that you may have been playing with while developing your program so these get included in your EXE even though they aren't used. Other VB users have even advocated saving all the project files as ASCII, then loading the ASCII files before compiling to further "clean up" the resulting EXE file.

Multiple END statements can be dangerous; or, The program that refused to terminate.

Suggestion: put the END statement used to exit your program *only* in the Form_Unload event of the main form. Whenever you want to end the program, just tell the main form to unload.

Some have reported that after their program have (supposedly) terminated, it still appears in the task list. This can happen if you only hide secondary forms and forget to unload them when you end/unload the main form.

Also note that the Stop-button on the button-bar of the integrated development environment doesn't really unload anything. It *nukes* the program, which generally is a good idea since it could be a bug in it that caused it to be stuck in an eternal loop or something.

What are the latest versions of the various files used by VB?

<u>Date</u>	<u>File to download</u>	<u>Updates files</u>	<u>Description</u>
3/7/94	BTR110.EXE	BTRV110.DLL	Btrieve IISAM Driver
3/7/94	DATAINDX.EXE	DATAINDX.DOC	"Data Access Guide" Index
3/7/94	GENERIC.EXE	\VB\CDK\GENERIC	Sample custom control source
3/7/94	VBGRID.EXE	GRID.VBX	Grid control
3/7/94	VBHC505.EXE	HC.EXE, HCP.EXE	WinHelp compiler
3/7/94	MSAJT.EXE	MSAJT110.DLL	Access Database Engine
3/8/94	MSCOMM.EXE	MSCOMM.VBX	Serial Communications\control
3/7/94	ORA110.EXE	ORACLE.TXT	Updated ORACLE.TXT file
6/27/94	SETUPK.EXE	SETUP.EXE	Setup Toolkit
3/7/94	VBRUN300.EXE	VBRUN300.DLL	Visual Basic Runtime Library
3/7/94	XBS110.EXE	XBS110.DLL	XBase IISAM Driver

There is an article in the Microsoft Knowledge Base that points to each of these files and provides more detailed information about the update. To find these articles, query the Microsoft Knowledge Base using the file name and the word "update3.00". Note the NEW SETUPKIT update! [Thanks to Marks Harrop <harrop@werple.apana.org.au>]

Please inform the FAQ maintainer about newer versions.

Any tips for VB/Win 3 programmers moving to VBA?

You are in for some surprises. VBA is more unlike VB than most people thought. Especially the development environment is very different, and the language puts more emphasis on objects. The latter is a trend you can get used to for VB also.

For Excel 5 VBA, be aware that the environment is based on the "workbook" idea Microsoft stole from Borland. Your controls will be placed in one sheet, and the code will be in another. Double-clicking on the control to open the code window doesn't help. You have to use the "Tools|Assign Macro" menu option.

Also, be aware that the list of events is nowhere close to what VB3 supports! No GotFocus, no MouseMove, no nothing. You'll be very confused if you try to look for "events" in the VBA docs!

Does VBA support VBXs?

No. If Microsoft have its way, VBX is a dead end. There will never be 32-bit VBXs, but OCXs using OLE 2. VBA is more a subset of VB 4 than VB 3, but it does not fully support OCX yet. It will, though.

How do I access properties on my dialog boxes in VBA?

As noted above, VBA is a cultural chock for VB programmers. If you create a textbox in VBA, call it txName and try to

```
cMyVar=txName.Text
```

the impolite interpreter will give you a "variable not defined" error.

The magic is objects. You have to

```
Dim txName as Object  
Set txName = DialogSheets("NameDialog").EditBoxes("txName")
```

And then you can access your properties like you used to in good ol' VB 3. (Anyone volunteer to beat senseless the guy who thought out this?)

How do I use database routines from Excel VBA?

The documentation is somewhere between sparse and nonexistent on this topic. Any info on VBA and SQL would be much appreciated.

Here Microsoft breaks the tradition and you *can't* use database objects, at least not the way you do in VB. Also, forget dynasets.

I know nothing about databases in VBA. I just bring on the following tips from various magazines:

Both SQLOpen and QueryGetData require a 'connection string'. That's about what the doc's say about the parameter. What is it? The doc is also tragically void of useful examples. Someone dug up the following example:

```
"DSN=My data file;DBQ=c:\access\data.mdb;FIL=RedISAM;"
```

which is about as understandable as it looks. If you use an empty string, you get a dialog which also can give you the string into a spreadsheet cell.

Also, search for SQLREQUEST in the *main* help file for Excel 5 (not the VBA help!) for these examples of connection_string's:

dBASE	DSN=NWind;PWD=test
SQL Server	DSN=MyServer;UID=dbayer;PWE=123;Database=Pubs
ORACLE	DSN=My Oracle Data Source;DBQ=MYSER VER;UID=JohnS;PWD=Sesame

There's a KnowledgeBase on Excel 5 on ftp.microsoft.com. Last time I looked, it was void of database stuff. Still, it may be a good idea to download it as the situation may have changed now.

What is "Reserved Error -1209"?

You will get a Reserved Error [-1209] ("There is no message for this error") when your database is corrupted. Try opening the database using MS Access; if it's corrupted you should get the option to repair it.

[Joe Abley (joe_abley@originuk.demon.co.uk)]

You should also compact it, after repair. I recommend you add the following to your File menu on your main form:

```
Case ...
    RepairDatabase Currentdatabasename
Case ....
    On Error resume next
    Kill "temp.MDB"
    Name currentdatabasename as "temp.mdb"
    on error goto errcompact
    compactdatabase "temp.mdb", Currentdatabasename
    kill "temp.mdb"
    exit sub

errcompact:
    msgbox "compaction failed"
    name "temp.mdb" as Currentdatabasename

Case ...
```

[Ayn Shipley (ashipley@hookup.net) , Kym Wilson]

"Cannot perform operation. illegal.." with Paradox 3.5 table(s)

Your Paradox table must have a primary key, or it will be read-only no matter what you set its properties to.

[Ayn Shipley (ashipley@hookup.net)]

I'm getting error message "Reserved Error [-nnnn] ("There is no message for this error")" from Jet Engine 2.0. Huh?

See the Knowledge Base article Q117900 "Reserved Error Numbers Returned by the Jet 2.0 Engine" for a complete list of the new error messages.

Extract:

```
"Jet_Error/Message_String
-1010 Invalid database ID.
-1016 Can't have more than 10 fields in an index.
-1029 Database engine hasn't been initialized.
-1030 Database engine has already been initialized.
-1034 Query support unavailable."
```

[Ayn Shipley (ashipley@hookup.net)]

Why do I get "object not an array" when I try reference the fields of a global object variable which I have set to a table?

VB has a parser bug which makes it difficult to use database objects declared in a module from within a form.










WORKAROUND: Just perform some `_method_` on the table object somewhere `_before_` you try to reference fields. Say in a form-based subroutine `AAAA_IIIbFirst` you have a `Tbl.MoveFirst`, which is never even executed. Then VB suddenly realises what the object is and all is forgiven.

Credit to Luke Webber and "Joe Foster of Borg".

[Ayn Shipley (ashipley@hookup.net)]

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS: [Note]

-  [PREFACE](#)
-  [GENERAL VISUAL BASIC QUESTIONS](#)
-  [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
-  [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
-  [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
-  [VISUAL BASIC AND DATABASES](#)
-  [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
-  [MISCELLANEOUS TIPS AND INFORMATION](#)
-  [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS: [Note]

 [PREFACE](#)

 [About this document](#)

 [Where to get the VB/Win FAQ \[++\]](#)

 [Kudos & comments](#)






- [GENERAL VISUAL BASIC QUESTIONS](#)
- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
- [VISUAL BASIC AND DATABASES](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- [PREFACE](#)

-  [GENERAL VISUAL BASIC QUESTIONS](#)

-  [Does VB/Win make standalone .EXE files?](#)
-  [What is the current version of Visual Basic for Windows? \[++\]](#)
-  [Where can I get updated VB and other Microsoft files?](#)
-  [Help! I am lost on ftp.microsoft.com .](#)
-  [Where can I get good up-to-date information about VB? \[++\]](#)
- [Are there any examples of commercial applications built using Visual Basic?](#)
- [WWW Pages for VB! \[**\]](#)
- [Limits of VB? \[**\]](#)

- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
- [VISUAL BASIC AND DATABASES](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- [PREFACE](#)
- [GENERAL VISUAL BASIC QUESTIONS](#)
- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
 - [What's the difference between MODAL and MODELESS forms? \[++\]](#)
 - [When/Why should I use Option Explicit?](#)
 - [Why does everybody say I should save in TEXT not BINARY?](#)
 - [Is the Variant type slower than using other variable types?](#)
 - [How do I make a text box not beep but do something else when I hit the Enter key?](#)
 - [How do I implement an incremental search in list/dir/combo/file boxes?](#)
 - [How do I get the Tab key to be treated like a normal character?](#)
 - [How do I make an animated icon for my program?](#)
 - [What is passing by reference?](#)
 - [I get a "file not found" error on the IIF function when I distribute by program. Uh?](#)
 - [Is there any way to pass a variable to a form apart from using global variables?](#)
 - [How should dates be implemented so they work with other language and country formats?](#)

 - [Can a VB application be an OLE server?](#)
 - [How do I dial a phone number without using the MSCOMM VBX?](#)
 - [I have \[several\] megabytes of memory. Why do I get an "out of memory" error? \[++\]](#)
 - [How do I mimic a toggle button? \[++\]](#)
 - [How do I get my application on top?](#)
 - [Is there a way to break long lines in VB code?](#)
 - [How do I remove/change the picture property of a control at design time?](#)
 - [Is a \[foo\] VBX/DLL available as shareware/freeware?](#)
 - [How do I make my applications screen-resolution independent?](#)

 - [How do I do Peek and Poke? \[++\]](#)
 - [Why doesn't "my string" & Chr\\$\(13\) do what I want?](#)
 - [How do I prevent multiple instances of my program?](#)
 - [How do I implement an accelerator key for a text box?](#)
 - [How do I force a file dialogue box to reread the current disk?](#)
 - [How do I get the number of free bytes on a disk? \[++\]](#)

 - [Data Control missing from toolbox when I use VB under NT 3.5. Huh?](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
- [VISUAL BASIC AND DATABASES](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- PREFACE
- GENERAL VISUAL BASIC QUESTIONS
- COMMON VISUAL BASIC PROGRAMMING QUESTIONS
- ADVANCED VISUAL BASIC PROGRAMMING ISSUES
 - How do I tell when an application executed using the SHELL command is finished?
 - How do I access C style strings?
 - How can I change the printer Windows uses in code without using the print common dialog?
 - Any tips for speeding up VB?
 - How do I speed up control property access?
 - How much gain in performance will I get if I write my number crunching routines in C instead of Visual Basic?
- How do you make a TEXTBOX read only? Or, how do I prevent the user from changing the text in a TEXTBOX?
 - How can I create a VBX?
 - How do you change the system menu (on the Control-Menu Box)?
 - How do I play MID, WAV or other multimedia files?
 - How can I call a 'hidden' DOS program from VB?
 - How do I do drag & drop between applications?
 - How do I use GetPrivateProfileString to read from INI files?
 - How do I implement Undo?
 - How do I create a window with a small title bar as in a floating toolbar?
 - What is Pseudocode?
 - Does VB support pointers to functions?
 - How do I program the Novell NetWare API from VB?
 - How do you change the icon and otherwise manipulate the DOS box?
 - How do I make the mouse cursor invisible/visible?
 - How do I create controls dynamically (at run-time)?
 - How do I set the Windows wallpaper at runtime?
 - Visual Basic 4 News [**]
 - How do I call help files from a VB program? [**]
- CALLING THE WINDOWS API AND DLLs IN GENERAL
- VISUAL BASIC AND DATABASES
- DISTRIBUTING VISUAL BASIC APPLICATIONS
- MISCELLANEOUS TIPS AND INFORMATION
- VISUAL BASIC FOR APPLICATIONS (VBA)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- [PREFACE](#)
- [GENERAL VISUAL BASIC QUESTIONS](#)
- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
 - [What is the Windows API?](#)
 - [How do I call a DLL?](#)
 - [What about DLL calls that require callbacks?](#)
 - [Tips for calling DLLs \(such as the Windows API\)](#)
- [VISUAL BASIC AND DATABASES](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- [PREFACE](#)
- [GENERAL VISUAL BASIC QUESTIONS](#)
- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
- [VISUAL BASIC AND DATABASES](#)
 - [Why can't I use an index with my VB accessed database?](#)
 - [Why does my compiled VB database app generate an error when it ran just fine in the design environment?](#)
 - [Is the Access Engine and Visual Basic Pro good enough for database work?](#)
 - [How do you avoid the "Invalid use of null" error when reading null values from a database?](#)
 - [What is "NULL"?](#)
 - [How can I access a record by record number?](#)
 - [How about Access 2.0 compatibility?](#)
 - [Tips for VB database programming:](#)
 - [How come I get a "No Current Record" error when I use a Data Control on an empty table?](#)
 - [How can I speed up my VB database application?](#)
 - [How do I get a bitmap picture in a field in an Access database?](#)
 - [What is "Reserved Error -1209"?](#)
 - ["Cannot perform operation. illegal.." with Paradox 3.5 table\(s\)](#)
 - [I'm getting error message "Reserved Error \[-nnnn\] \("There is no message for this error"\)" from Jet Engine 2.0. Huh?](#)
 - [Why do I get "object not an array" when I try reference the fields of a global object variable which I have set to a table?](#)
 - [Steps for securing an Access 2.0 database \[**\]](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- [PREFACE](#)
- [GENERAL VISUAL BASIC QUESTIONS](#)
- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
- [VISUAL BASIC AND DATABASES](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
 - [What are some tips for using Setup Wizard?](#)
 - [Are there restrictions on what I can distribute with my VB program?](#)
 - [What alternatives to setup wizard do I have?](#)
 - [Do I need to worry about users who have Progman replacements such as Norton Desktop and](#)
- [PC Tools?](#)
 - [Can I distribute my app without vbrunXXX.dll?](#)
 - [Why won't my setup program install commdlg.dll et. al.?](#)
 - [Where do I install VBxS and DLLs?](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- PREFACE
- GENERAL VISUAL BASIC QUESTIONS
- COMMON VISUAL BASIC PROGRAMMING QUESTIONS
- ADVANCED VISUAL BASIC PROGRAMMING ISSUES
- CALLING THE WINDOWS API AND DLLs IN GENERAL
- VISUAL BASIC AND DATABASES
- DISTRIBUTING VISUAL BASIC APPLICATIONS
- MISCELLANEOUS TIPS AND INFORMATION
 - Multiple identifiers after the DIM statement can be confusing
 - "Clean up" your project before final EXE compilation.
 - Multiple END statements can be dangerous; or, The program that refused to terminate.
 - What are the latest versions of the various files used by VB?
- VISUAL BASIC FOR APPLICATIONS (VBA)

VISUAL BASIC FOR WINDOWS (VB/Win)
Frequently asked Questions & Answers Section IX - B
Last-modified: 12.03.95

TABLE OF CONTENTS:

- [PREFACE](#)
- [GENERAL VISUAL BASIC QUESTIONS](#)
- [COMMON VISUAL BASIC PROGRAMMING QUESTIONS](#)
- [ADVANCED VISUAL BASIC PROGRAMMING ISSUES](#)
- [CALLING THE WINDOWS API AND DLLs IN GENERAL](#)
- [VISUAL BASIC AND DATABASES](#)
- [DISTRIBUTING VISUAL BASIC APPLICATIONS](#)
- [MISCELLANEOUS TIPS AND INFORMATION](#)
- [VISUAL BASIC FOR APPLICATIONS \(VBA\)](#)
 - [Any tips for VB/Win 3 programmers moving to VBA?](#)
 - [Does VBA support VBXs?](#)
 - [How do I access properties on my dialog boxes in VBA?](#)
 - [How do I use database routines from Excel VBA?](#)

Questions

- [About this document](#)
- [Any tips for speeding up VB?](#)
- [Any tips for VB/Win 3 programmers moving to VBA?](#)
- [Are there any examples of commercial applications built using Visual Basic?](#)
- [Are there restrictions on what I can distribute with my VB program?](#)
- [Can a VB application be an OLE server?](#)
- [Can I distribute my app without vbrunXXX.dll?](#)
- ["Cannot perform operation. illegal.." with Paradox 3.5 table\(s\)](#)
- ["Clean up" your project before final EXE compilation.](#)
- [Data Control missing from toolbox when I use VB under NT 3.5. Huh?](#)
- [Do I need to worry about users who have Progman replacements such as Norton Desktop and](#)

PC Tools?

- [Does VB support pointers to functions?](#)
- [Does VB/Win make standalone .EXE files?](#)
- [Does VBA support VBXs?](#)
- [Help! I am lost on ftp.microsoft.com .](#)
- [How about Access 2.0 compatibility?](#)
- [How can I access a record by record number?](#)
- [How can I call a 'hidden' DOS program from VB?](#)
- [How can I change the printer Windows uses in code without using the print common dialog?](#)
- [How can I create a VBX?](#)
- [How can I speed up my VB database application?](#)
- [How come I get a "No Current Record" error when I use a Data Control on an empty table?](#)
- [How do I access C style strings?](#)
- [How do I access properties on my dialog boxes in VBA?](#)
- [How do I call a DLL?](#)
- [How do I call help files from a VB program? \[**\]](#)
- [How do I create a window with a small title bar as in a floating toolbar?](#)
- [How do I create controls dynamically \(at run-time\)?](#)
- [How do I dial a phone number without using the MSCOMM VBX?](#)
- [How do I do drag & drop between applications?](#)
- [How do I do Peek and Poke? \[++\]](#)
- [How do I force a file dialogue box to reread the current disk?](#)
- [How do I get a bitmap picture in a field in an Access database?](#)
- [How do I get my application on top?](#)
- [How do I get the number of free bytes on a disk? \[++\]](#)
- [How do I get the Tab key to be treated like a normal character?](#)
- [How do I implement an accelerator key for a text box?](#)
- [How do I implement an incremental search in list/dir/combo/file boxes?](#)
- [How do I implement Undo?](#)
- [How do I make a text box not beep but do something else when I hit the Enter key?](#)
- [How do I make an animated icon for my program?](#)
- [How do I make my applications screen-resolution independent?](#)
- [How do I make the mouse cursor invisible/visible?](#)
- [How do I mimic a toggle button? \[++\]](#)

- [How do I play MID, WAV or other multimedia files?](#)
- [How do I prevent multiple instances of my program?](#)
- [How do I program the Novell NetWare API from VB?](#)
- [How do I remove/change the picture property of a control at design time?](#)
- [How do I set the Windows wallpaper at runtime?](#)
- [How do I speed up control property access?](#)
- [How do I tell when an application executed using the SHELL command is finished?](#)
- [How do I use database routines from Excel VBA?](#)
- [How do I use GetPrivateProfileString to read from INI files?](#)
- [How do you avoid the "Invalid use of null" error when reading null values from a database?](#)
- [How do you change the icon and otherwise manipulate the DOS box?](#)
- [How do you change the system menu \(on the Control-Menu Box\)?](#)
- [How do you make a TEXTBOX read only? Or, how do I prevent the user from changing the text in a TEXTBOX?](#)
- [How much gain in performance will I get if I write my number crunching routines in C instead of Visual Basic?](#)
- [How should dates be implemented so they work with other language and country formats?](#)
- [I get a "file not found" error on the IIF function when I distribute by program. Uh?](#)
- [I have \[several\] megabytes of memory. Why do I get an "out of memory" error? \[++\]](#)
- [I'm getting error message "Reserved Error \[-nnnn\] \("There is no message for this error"\)" from Jet Engine 2.0. Huh?](#)
- [Is a \[foo\] VBX/DLL available as shareware/freeware?](#)
- [Is the Access Engine and Visual Basic Pro good enough for database work?](#)
- [Is the Variant type slower than using other variable types?](#)
- [Is there a way to break long lines in VB code?](#)
- [Is there any way to pass a variable to a form apart from using global variables?](#)
- [Kudos & comments](#)
- [Limits of VB? \[**\]](#)
- [Multiple END statements can be dangerous; or, The program that refused to terminate.](#)
- [Multiple identifiers after the DIM statement can be confusing](#)
- [Steps for securing an Access 2.0 database \[**\]](#)
- [Tips for calling DLLs \(such as the Windows API\)](#)
- [Tips for VB database programming:](#)
- [Visual Basic 4 News \[**\]](#)
- [What about DLL calls that require callbacks?](#)
- [What alternatives to setup wizard do I have?](#)
- [What are some tips for using Setup Wizard?](#)
- [What are the latest versions of the various files used by VB?](#)
- [What is "NULL"?](#)
- [What is "Reserved Error -1209"?](#)
- [What is passing by reference?](#)
- [What is Pseudocode?](#)
- [What is the current version of Visual Basic for Windows? \[++\]](#)
- [What is the Windows API?](#)
- [What's the difference between MODAL and MODELESS forms? \[++\]](#)
- [When/Why should I use Option Explicit?](#)
- [Where can I get good up-to-date information about VB? \[++\]](#)
- [Where can I get updated VB and other Microsoft files?](#)
- [Where do I install VBXs and DLLs?](#)
- [Where to get the VB/Win FAQ \[++\]](#)
- [Why can't I use an index with my VB accessed database?](#)

- [Why do I get "object not an array" when I try reference the fields of a global object variable which I have set to a table?](#)
- [Why does everybody say I should save in TEXT not BINARY?](#)
- [Why does my compiled VB database app generate an error when it ran just fine in the design environment?](#)
- [Why doesn't "my string" & Chr\\$\(13\) do what I want?](#)
- [Why won't my setup program install commdlg.dll et. al.?](#)
- [WWW Pages for VB! \[**\]](#)

Help format by Tim Roberts
Phone: (902) 494 2842
FAX: (902) 494-1107
E-Mail: TJR@AC.DAL.CA

VB/Win FAQ Maintainer
Jan Steinar Haugland
E-Mail: Jan.Haugland@uib.no

Note

From this issue of the FAQ I introduce symbols in topic header and TOC:

[++] means topic is updated in this issue

[]** means topic is new in this issue

Hope this makes it easier for Our Regular Readers ;-)

