VBX Studio 1.2b - Contents

Welcome!
in the world of VBX Studio 1.2, the ultimate package for Visual programmers. To get some help on a VBX, just click on the corresponding button...

Overview

Features

VBX are a nice way to include complex controls or objects with minimal hard coding into your Windows 3.10 programs. Also, there are 3 things to do to use most VBX:
-> Click on the palette and draw the control at design time,
-> Change some 'properties' to adapt the control to your needs (at design time) and,
-> Distribute the VBX with your application and put this file in the Windows system directory.

VBX Studio 1.2 provides 15 really useful VBXs:
-> Bitmapped listbox (sometimes without any hard coding),
-> Bitmapped combobox (as the listbox, you choose bitmap at design time: this is quite easy!),
-> Files and directories bitmapped listbox (of course, no coding required),
-> Files and drives bitmapped combobox (''),
-> 3D and automatic spins (with or without coding),
-> 3D Tabs to build single or multiple tabbed lines last generation dialog boxes,
-> Pictured edit controls (always to avoid any checking routines),
-> 3D or classic gauge (sometimes called percentage controls),
-> Animated pictures for special effects,
-> Pictured buttons (with\without OnOff mode, button border...),
-> Nice static text controls (with\without 3D, auto sizing),
-> Automatic 3D generator (to draw a 3D border for known and unknown controls...),
-> Winfile drag and drop control and,
-> Design time specific note control to annotate your resources.

For an overview, run the VBXDEMO program.
Of course, all these controls are VB 1.0 and VC++ compatible and to make them as easy to use as standard controls (maybe more easy..), we provide all required object wrappers and DDX functions.
As a conclusion, these controls are the cheapest: Less than 4$- per VBX !

VBX Installation

If you use Visual Basic, choose "Add Files..." and select the VBX you wish to install. If you prefer VC++ AppStudio, use "Install controls..", double click on the VBXs and close the dialog box with the "Ok" button (do not use the "Close" button). If you use AppStudio and you want to use the VBX Studio DDX routines, have a look at DDX.TXT text file and, if you want to use classes wrappers, put VBXSTD12.H and ASVBDO12.H into your INCLUDE path and put the '#include <VBXSTD12.H>' at the end of your STDAFX.H precompiled header.

Registration

Unregistered (evaluation) version will display a nag screen each time you create a VBX at run time. However, once you' ve created 'unregistered' VBX controls in your dialog boxes, and then you register, you don' t have to rebuild your dialog boxes: You just have to reopen these and they' ll be automatically updated. Unregistered is not crippled in any way: there are no special functions\properties for registered users. You are not allowed to distribute any program made with the unregistered version.

To register, you must use the VBXREG program we distribute with the VBX Studio 1.2 package: This utility will create the required LIC file: Of course, you must not distribute this file! Once you are a registered user, you can distribute your programs using VBX Studio without any royalties.
VBX Studio costs 55$-, you can register on Compuserve SWREG and PsL (call 800.2424.PSL).

Support?

You can join us thanks to COMPUSERVE (the best way, our CIS id is 100333,27), or, on the net at

<hnet@dialup.francenet.fr>, or, you can write to us. If you ask for example, "How can we load a VBX with Visual Basic" or "Can we use your product to create a listbox?" you will never receive any reply! We hope you understand that the technical support is provided only for real questions regarding our product (and only VBX Studio) and if there is no answer in this help file!

How to redistribute the VBX?

When you use the VBX Studio VBXs you must not forget to distribute the VBX(s) with your software. Also, these VBXs must be put into your customer WINDOWS\SYSTEM directory. To do this you can find a big help in Setup Studio 2.2 (Another 'Studio' shareware, see Author for more informations). With Setup Studio you can even use VBX Studio VBXs with your Setup program! To install the VBX(s), the best way is too check the VBX last modification date: If you don' t find the VBX you must install it, if the VBX is found but is older you don' t install it (but you maybe need to display a warning), and if your VBX is newer, you backup the old one and you install the new one.

Terms

Description

Used terms are C, C++ and SDK language type. If you do not program with these languages, perhaps you will have some difficulties to understand them. These are some explanations:

WINAPI	FAR PASCAL (just ignore this)
int, Short	Integer value.
long, Long	Long integer value.
float, Float	Float (real) decimal value
UINT	Signed long integer value.
BOOL	Integer value, 0 for FALSE, anything else for TRUE.
char	Character.
char far*	String.
LPCSTR	Constant String
LPSTR, String, HSZ	String.
HWND	Window' s Handle(Integer value).
HINSTANCE	Application' s Handle(Integer value).
void	Null type (empty).
COLORREF	Long integer value (Color).
TRUE	An integer value other than 0 (-1).
FALSE	0

Package

The ZIP file you' ve received contains the following files:

(uncompressed files, about):

```
SETUP.EXE      71.000 The setup program.
CSETUP.DLL     113.90 A Setup module.
                    4
UZSETUP.XXX    28.334 A Setup module.
CTL3D.DLL      14.336 A Setup module.
SETUP.INF       3.000 A Setup module.
SETUP.LST         500 A Setup module.
MAIN.EXE       98.000 A Setup module.
FINISH.EXE      5.100 A Setup Module.

READ.US         2.751 Basic informations file.
ORDER.TXT       2.000 The order form.
OFFER.TXT       1.399 'Studio' bundle special offer.
DESC.SDI          168 BBS file.
FILE_ID.DIZ       170 BBS file.

VBLIST.VBX     47.552 VBList VBX.
VBFLIST.VBX    57.424 VBFList VBX.
VBCOMBO.VBX    47.408 VBCombo VBX.
VBFCOMBO.VBX   49.712 VBFCombo VBX.
VBTAB.VBX      30.896 VBTab VBX.
VBSPIN.VBX     29.664 VBSpin VBX.
SPINDLL.DLL     6.415 VBSpin VBX required DLL.
VBGAUGE.VBX    38.048 VBGauge VBX.
VBEDIT.VBX     63.568 VBEdit VBX.
VBPICBTN.VBX   35.808 VBPicBtn VBX.
VBANIM.VBX     30.032 VBAnim VBX.
VB3D.VBX       21.712 VB3D VBX.
VBDDROP.VBX    20.464 VBDDrop VBX.
VBNOTE.VBX     19.296 VBNote VBX.
VBTEXT.VBX     30.960 VBText VBX.
VBLINE.VBX     31.936 VBLine VBX.

ASVBDO12.H     10.342 Include file for class
                      wrappers.
VBXSTD12.H     60.001 Include file for VC++.
VBXSTD12.BAS    6.418 Header file for VB
                      (constants).
DDX.TXT         3.730 Text file to set up DDX
                      routines.
VBXREG.EXE     51.216 Registration utility program.
VBXSTD12.HLP   240.00 This help file.
                    0

VBXDEMO.EXE    230.25 VBXDEMO - The program.
                    6
3DDEMO.CPP      1.943 VBXDEMO - VB3D CPP file.
3DDEMO.H          670 VBXDEMO - VB3D H file.
ANIMDEMO.CPP    1.719 VBXDEMO - VBAnim CPP file.
ANIMDEMO.H        697 VBXDEMO - VBAnim H file.
COMBODEM.CPP    1.739 VBXDEMO - VBCombo CPP file.
COMBODEM.H        699 VBXDEMO - VBCombo H file.
```

```
DDROPDEM.CPP   1.975 VBXDEMO - VBDDrop CPP file.
DDROPDEM.H       734 VBXDEMO - VBDDrop H file.
EDITDEMO.CPP   3.315 VBXDEMO - VBEdit CPP file.
EDITDEMO.H     1.105 VBXDEMO - VBEdit H file.
FCOMBODE.CPP   1.893 VBXDEMO - VBFCombo CPP file.
FCOMBODE.H       674 VBXDEMO - VBFCombo H file.
FLISTDEM.CPP   3.022 VBXDEMO - VBFList CPP file.
FLISTDEM.H       860 VBXDEMO - VBFList H file.
GAUGEDEM.CPP   2.305 VBXDEMO - VBGauge CPP file.
GAUGEDEM.H       807 VBXDEMO - VBGauge H file.
LINEDEMO.CPP   1.724 VBXDEMO - VBLine CPP file.
LINEDEMO.H       693 VBXDEMO - VBLine H file.
LISTDEMO.CPP   1.680 VBXDEMO - VBList CPP file.
LISTDEMO.H       693 VBXDEMO - VBList H file.
MAINDLG.CPP    8.716 VBXDEMO - Main dialog CPP
                     file.
MAINDLG.H      1.187 VBXDEMO - Main dialog H file.
PICBTNDE.CPP   4.221 VBXDEMO - VBPicBtn CPP file.
PICBTNDE.H     1.069 VBXDEMO - VBPicBtn H file.
RESOURCE.H     6.045 VBXDEMO - Resources
                     definitions.
SPINDEMO.CPP   4.317 VBXDEMO - VBSpin CPP file.
SPINDEMO.H     1.008 VBXDEMO - VBSpin H file.
STDAFX.CPP       204 VBXDEMO - CPP Precompiled
                     header.
STDAFX.H         404 VBXDEMO - H precompiled
                     header.
TABDEMO.CPP    4.947 VBXDEMO - VBTab CPP file.
TABDEMO.H      1.129 VBXDEMO - VBTab H file.
TEXTDEMO.CPP   1.706 VBXDEMO - VBText CPP file.
TEXTDEMO.H       667 VBXDEMO - VBText H file.
VBXDEMO.CPP    7.003 VBXDEMO - Application CPP
                     file.
VBXDEMO.H      1.373 VBXDEMO - Application H file.
VBXDEMO.DEF      353 VBXDEMO - Application DEF
                     file.
VBXDEMO.MAK   10.214 VBXDEMO - The project MAK
                     file.
VBXDEMO.RC   388.73 VBXDEMO - The res. script
                 5   file.
BMP00001.BMP     286 VBXDEMO - RES - Resource
                     file...
BMP00002.BMP     198 VBXDEMO - RES - Resource
                     file...
BMP3D.BMP        286 VBXDEMO - RES - Resource
                     file...
BMPANIM.BMP      286 VBXDEMO - RES - Resource
                     file...
BMPBLACK.BMP     198 VBXDEMO - RES - Resource
                     file...
BMPBLUE.BMP      198 VBXDEMO - RES - Resource
                     file...
BMPCOMBO.BMP     286 VBXDEMO - RES - Resource
                     file...
BMPCYAN.BMP      198 VBXDEMO - RES - Resource
                     file...
BMPDARKB.BMP     198 VBXDEMO - RES - Resource
```

```
                    file...
BMPDARKC.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPDARKF.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPDARKG.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPDARKR.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPDARKY.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPDDROP.BMP      286 VBXDEMO - RES - Resource
                    file...
BMPEDIT.BMP       286 VBXDEMO - RES - Resource
                    file...
BMPFUSHI.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPGAUGE.BMP      286 VBXDEMO - RES - Resource
                    file...
BMPGREEN.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPGREY.BMP       198 VBXDEMO - RES - Resource
                    file...
BMPLINE.BMP       286 VBXDEMO - RES - Resource
                    file...
BMPLIST.BMP       286 VBXDEMO - RES - Resource
                    file...
BMPLISTF.BMP      286 VBXDEMO - RES - Resource
                    file...
BMPNOTE.BMP       286 VBXDEMO - RES - Resource
                    file...
BMPPICBTN.BM      286 VBXDEMO - RES - Resource
P                   file...
BMPRED.BMP        198 VBXDEMO - RES - Resource
                    file...
BMPSPIN.BMP       286 VBXDEMO - RES - Resource
                    file...
BMPTAB.BMP        286 VBXDEMO - RES - Resource
                    file...
BMPTEXT.BMP       286 VBXDEMO - RES - Resource
                    file...
BMPWHITE.BMP      198 VBXDEMO - RES - Resource
                    file...
BMPYELLO.BMP      198 VBXDEMO - RES - Resource
                    file...
VBXDEMO.ICO       766 VBXDEMO - RES - Resource
                    file...
VBXDEMO.RC2     1.548 VBXDEMO - RES - Resource
                    file...

DDROP.FRM       2.416 VBSAMPLE - VBDDrop form.
PICBTN.FRM     22.361 VBSAMPLE - VBPicBtn form.
PICBTN.FRM     20.790 VBSAMPLE - FRX file.
TABED.FRM      12.452 VBSAMPLE - VBTab form.
VBXSTD12.MAK      277 VBSAMPLE - The MAK file.
```

Compressed files

You can find VBXSTD12.ZIP or VBXSTD12.EXE on COMPUSERVE (MSBASIC Forum), BBS and thanks to diskettes vendors. These files are about 750 Kb and a full installation requires about 2 Mb.

Author

Versions

Two versions are available: The registered version and the evaluation version. The evaluation version can be freely distributed if you give the full VBX Studio package and you don' t modify any file. You can find the evaluation version:

- on COMPUSERVE, we always put our latest releases on the WUGNET forum. You can also register thanks to COMPUSERVE (go SWREG and search for 'STUDIO").

- on CICA, we always put our latest releases on the <winftp.cica.indiana.edu> site.

- Thanks to DP Tool Club,
BP 745, 59657 VILLENEUVE D' ASCQ (FAX: (16) 20.05.38.27) FRANCE.

With the unregistered version, you can fully test VBX Studio. You will see Shareware principles reminders when you create a VBX (at run time). Distributing programs made with VBX Studio unregistered version is forbidden!

The registered version required the evaluation version because you change the old version thanks to two personal passwords. You will receive these passwords, by mail, about 2 days after you register (French users). If you have a COMPUSERVE ID, we will give you your passwords there and without delay.

Registration

To register you can print Order form and you must send your payment to:

HEXANET®

HEXANET
BP 385.16
75768 PARIS CEDEX 16
FRANCE

You can also use the compuserve registration service (go SWREG), or PsL registration service to pay with a credit card.To use SWREG, type 'Go SWREG' and then search for 'STUDIO' sharewares or give our name
or CIS id (100333,27). PSL ID for VBX Studio is #11593.

You can order with MC, Visa, or American Express from the Public (software) Library (PsL) by calling 1-800-2424-PSL or FAXing 1-713-524-6398. These numbers are for orders only. For questions about credit card orders, call PsL at 713-524-6394. You can also mail credit card orders to PsL at P.O.Box 35705, Houston, TX 77235-5705.

Save $$$!

When you register VBX Studio, it' s a good idea to register Setup Studio 2.2 at the same time: Buy the VBX Studio + Setup Studio package and you save 20$-. This special offer is available for the bundle and you must buy it in one time. Compuserve user can use SWREG to register this bundle, but maybe we will stop this offer after the 12/31/1994... Also, maybe you need String Studio package (if you are a MFC user)to handle strings and time DDX, you can register on Compuserve.

Prices depend on currency you use: You can pay in French Franc currency, which is the cheapest way to pay or in another allowed currency in this list:

Currency:	VBX STUDIO	VBX STUDIO + SETUP STUDIO
Francs français:	300-	450-
US Dollars:	55-	85-
Deutch mark:	90-	145-
GB Pounds:	37-	60-

Currencies 'Francs belges', 'Francs suisse' and 'Canadian dollars' are no more allowed for all 'Studio' sharewares. If you need a diskette, you must buy the bundle and pay with a check in USD drawn an US Bank. If you are living in a foreign country, maybe your sharewares distributor has a foreign payment service (in USD for example) to help you to register. Such distributors can ask for a payment for this service. For informations, please ask your distributor.

Upgrades

To know about upgrades you can connect to COMPUSERVE (go MSBASIC) or you can also read the DP Tool Club (see higher) news. Generally, you don' t have to pay again for upgrades.

License

The license is an using authorisation but not a product cession. Programs made with the unregistered version never will be distributed: Distributing programs made with VBX Studio unregistered version is forbidden! The registered version kills all nag screens.

You can use this product on several PC but a unique copy can be used at the same time.
The sell of this product without program is allowed for physical persons if it is an exception.

Author and technical support

You can join the author on COMPUSERVE (100333,27) and on the net, our address is <hnet@dialup.francenet.fr>. COMPUSERVE is the best way to join us because we read our mail every day. Do not hesitate to write to us your critics. You can also send a mail at our address. We offer a free technical support for registered users.

Other products

- Setup Studio 2.2

  This is a powerful toolkit to create professional installation programs for Windows 3.10. (The installation program you' ve used to install VBX Studio uses this one). SWREG id is 2455 and it costs 50$-. Buy this program with VBX Studio (in order to redistribute your VBXs) and you will save 20$ (to do this, Compuserve users must search for 'Studio' sharewares or sharewares from 100333,27).

- String Studio 2.x

  This is a MFC CString class extension with about 110 new functions to make strings management easier than VB. This class is not available with _AFXDLL. Source code is available for just $35- and SWREG id is 3003 or just $20 when registered with VBX Studio (SWREG ID 3521).

- Zip Studio 2.x

  The new Zip\Unzip API for Windows 3.x with many features and PKZIP 2.04g compatible. Now you can use DLLs or the VBX interface. This useful library is available for $70- and SWREG id is 2456.

- Zip Studio Shell 2.x

  The most enhanced archives shell: Built-in Zip/Unzip engine, 25 files formats, MDI... French version available. Licenses from $3.8! ... SWREG id is 3832 for a single user ($25-).

See also...

- Overview
- Order form
- Shareware

Shareware?

VBX Studio is a ShareWare:
You can try it as you want before you decide or not to register. The only one way we have to sell enough licenses to support this software, is to distribute it all over the world. So, if you think this product will help one of your friends, don' t hesitate to give him a copy of this product: This is your interest!

When you distribute VBX Studio, don' t change any file and give the complete package. If you UL VBX Studio on a BBS, please name it VBXSTD12.ZIP or VBXSTD12.EXE. Obviously, if you register, you can' t distribute the LIC file! Distributing programs made with VBX Studio unregistered version is forbidden!

Order form

To register you can print this help page and manually fill it, or use ORDER.TXT.
Unreadable orders will be ignored!

--- VBX STUDIO 1.2b ORDER FORM ---

Name

```
COMPANY:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
NAME:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
FIRST NAME:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
ATTENTION TO:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
ADDRESS:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
ZIP:_ _ _ _ _ _ _ _ _ _
TOWN:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
COUNTRY:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
COMPUSERVE ID: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
EMAIL: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
WHERE DID YOU FIND VBX STUDIO?:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
WHICH DEVELOPMENT SYSTEM DO YOU USE?:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

Product(s)

```
QUANTITY:_ _ _
CURRENCY:_ _ _ _ _ _ _ _
PRODUCT(S):_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

PRICES:

Currency:	VBX STUDIO	VBX STUDIO + SETUP STUDIO
Francs français:	300-	450-
US Dollars:	55-	85-
Deutch mark:	90-	145-
GB Pounds:	37-	60-

(Others not allowed)
These prices include shipping and VAT.
If you' ve got a COMPUSERVE ID or an EMail address, we will mail you your passwords before we send you your invoice, in order to avoid Postal delays. We always send out a printed invoice.

Send this form with your check to:

HEXANET®

HEXANET
BP 385.16
75768 PARIS CEDEX 16
FRANCE

How to register

First step: Buy the license

Click on the 'Overview' button and have a look at the Order form to do it.

Second Step: Use the VBXREG utility

VBX Studio is provided with the VBXREG utility: This program will create the required LIC file. If you' ve registered Zip Studio in a VBX (Zip Studio 1.3) and VBX Studio, you must use this program twice. To use this program, you must give your 2 passwords: The first one (called <szYourName>) in the first edit zone and the second one (called <szPassword>), in the second one. Then click on OK. That' s all.

Remark

If you get nag screen and you are a registered user, you probably have blanked the LIC file, so you must use this utility again!

VC++ Issues

As you know, VC++ thanks to MFC 2.0 emulates VBX via the CVBControl class. To get additional tips on using VBXs with MFC, you' ll find help in the Compuserve MSMFC forum.

**** WARNING ****
VC++ ApStudio is not the same as VB: When you build your application and when you want to test it, close the ApStudio resources editor. If you don' t do it, your VBXs can generate some troubles! Also, when you use the VBSpin VBX, you can' t run ApStudio and your program too! Also, you must use the <Refresh> method for some controls.

Basic considerations

When you use VC++ and a VBX you must check this VBX before you call the dialog box. MFC doesn' t display a dialog box if some VBXs are missing! To use VBXs, don' t forget to add EnableVBX() in your CWinApp::InitInstance() function. Also, using _AFXDLL and VBX requires special implementation.

Object wrappers vs CVBControl

With VC++ AppStudio, when you give a variable member for a VBX, this one will be in all cases a CVBControl* pointer. Thanks to DDX (see DDX.TXT) you can use a specific class for each VBX Studio VBX. For instance, your VBEdit VBX control will be associated with the CVBEdit CEdit derived class. Then you can use most of CEdit methods because your VBEdit control is a true Edit control (and a true CVBControl too..). Of course you can access the CVBControl class thanks to the GetVBX() CVBEdit member function (All VBX Studio classes contain this function).
To install VBX Studio DDXs, have a look at DDX.TXT file and copy some lines to APSTUDIO.INI as specified. VBX Studio DDXs are also provided for special string and time values: You can use CStr String Studio CString extensions and, CTime object (without hard coding). These DDX requires String Studio and don' t support _AFXDLL.

Required files

As we see above, to put some new DDX routines, have a look at DDX.TXT and change your APSTUDIO.INI file. Also, 2 other files are required to use VBX Studio VC++ extensions: ASVBDO12.H is the header file for the VBXs properties (and some other things..) and VBXSTD12.H which contains class definitions. You don' t have to worry about these files: Just put them into your INCLUDE directory and for each MFC project add #include <VBXSTD12.H> at the end of your STDAFX.H precompiled header. Also, if you wish to use String Studio CStr and CTime DDX routines, you must define _CSTR and include <CSTR.H> before you include <VBXSTD12.H>. String Studio file name is CSTR20.ZIP and you can find it on Compuserve MSMFC Forum.

Using properties with object wrappers

The basic interest of VBX Studio specific class is the dramatically improved properties Read\Write operations: You can use

```
m_pMyEdit->MinRange() = m_pMyEdit->MaxRange();
```

instead of

```
m_pMyEdit->SetNumProperty( "MinRange", m_pMyEdit-
>GetNumProperty( "MaxRange" ) );
```

for instance... The only thing you must do is to add '()' at the end of the property because this is a function, not a variable. Some properties are 'arrays' properties and they required an index value, like

m_pMyList->ListString(1)... Another VC++ advantage for these classes is the built-in VC++ browser: You don' t remember something, just click [F11] and you will see all the class!

Using VBX events

To retrieve an event parameter, just instanciate a VBX Studio class, then class the VC++ Class wizard and create an Event function. Now, give a value for the LPVOID parameter. Now, select the VBX variable and click on [F11] to have a look at its class. At the end of the class you can read all possible functions useful to retrieve an event value for this object, choose the corresponding one and use this function to retrieve the event parameter...like this:

```
void CMainDlg::OnDblclickVblist1(UINT, int, CWnd*, LPVOID lp)
  {
  int iListID = m_pMainList->EventDblClkID( lp ); // retrieve the current
VBList item number
 ...
```

As a conclusion, you can extend your other third party VBXs like VBX Studio wrappers...

VBList

VBList is the VBX Studio bitmapped ListBox.

Required file(s): *VBLIST.VBX*
Control name: *VBList*
MFC Class name: *CVBList (Base class is CListBox)*

See also...

VBFList

VBList Properties

Only non standard properties are described below in alphabetic order:

Border3D

Type: Bool

Set this to TRUE (-1) if you wish the listbox has a 3D border at run time.

Text3D

Type: Short

Enables or disables the 3D effect for items text.

AlwaysBitmapped

Type: Bool

Set this to TRUE (-1) if you wish all items with no bitmap specification use the first bitmap.

BackColor

Type: Short (COLOR40_...)

Specifies the background color.

Bitmap01 to Bitmap30

Type: Picture (bitmap)

These properties define a picture: <Bitmap01> is the picture 1, which can be called, depending on the data type like this: AddItem "1" for a bitmap only listbox and "Text1;1" for a bitmap and text listbox. This is the same for all 29 available other bitmaps.

Caption

Type: String

This string property is used to intialize the listbox: If you don' t give any caption, you will be in the test mode at design time (you will see bitmaps or text in the listbox) and of course, there is no data at run time. If you give a caption, you stop the test mode and this caption will be used to initialize the listbox: You can give multiple items data: Use '¤' to separate each item, like this 'Item1¤Item2¤Item3...'. The caption can' t be longer than 255 characters. Item data depends on the data type: For text datas, nothing special; For bitmap only datas, give a number to specify the bitmap number like: '1¤2¤2' and for 'both' datas use ';' to separate text data and bitmap data like this: 'Our text 1 use bitmap 1;1¤Our text 2 uses bitmap 2;2'... If you choose the always bitmapped option, in the 'both' data type, you can give 'Item1' and this will display the bitmap 1 and 'Item1' text (no ';' required).

DataType

Type: Short

Specifies the data type (bitmap, text or both).

DiableNoScroll

Type: Bool

Set this property to TRUE (-1) if you wish to use a listbox with a permanent vertical scroll bar (this is not available if you choose the Horizontal scroll bar too).

DisplayMode

Type: Short

Sets the display mode (for bitmaps). If you choose the automatic mode, you can' t use transparent bitmaps. The 'normal' mode uses margins.

Help

Will display this Windows HLP file.

ItemHeight

Type: Short

Specifies the height of the items height in the list (in pixels).

MultipleSel
Type: Bool
Set this property to TRUE (-1) if you wish to use a multiple selection\extended selection listbox.

NoRedraw
Type: Bool
Disables all redraw operations (not useful!).

ParentNotify
Type: Bool
Set this to TRUE (-1) to receive notifications messages (generally you must set this property to TRUE).

PictureHeight
Type: Short
Specifies the bitmap height (in pixels).To use it you must set <DisplayMode> to normal.

PictureWidth
Type: Short
Specifies the bitmap with (in pixels).To use it you must set <DisplayMode> to normal.

PictureXMargin
Type: Short
Specifies the bitmap horizontal margin (in pixels). To use it you must set <DisplayMode> to normal.

PictureYMargin
Type: Short
Specifies the bitmap vertical margin (in pixels). To use it you must set <DisplayMode> to normal.

Sort
Type: Bool
Set this property to TRUE (-1) if you wish to use a sorted listbox.

TextAlignment
Type: Short
Specifies the text alignment.

TextXMargin
Type: Short
Specifies the text horizontal margin (in pixels). To use it you must set <DisplayMode> to normal.

TextYMargin
Type: Short
Specifies the text vertical margin (in pixels). To use it you must set <DisplayMode> to normal.

UseHorzScrollBar()
Type: Bool
Set to TRUE (-1) if you wish to use an horizontal scroll bar if needed.

UseTranspBitmap
Type: Bool
Set this to TRUE (-1) if you use transparent bitmaps (bitmaps using the light fuchsia color as the 'transparent' color). Transparent bitmaps are not available with the 'stretch' (automatic) display mode.

vbxAbout

will display the VBX Studio 'About...' box.

VBList Special Properties

ListCount
Type: Short
Returns the number of items in the listbox

ListIndex(Index)
Type: Bool, property arrays
Use this property array to retrieve the state of an listbox item. If the specified item (0 indexed) is selected, the property will be set to TRUE (-1), otherwise FALSE (0) is returned.

ListString(Index)
Type: String, property arrays
Use this property array to retrieve the string (text) of an listbox item.

SetSel
Type: Long
For a single selection listbox, calling this property with a valid index (0 based) will select specified item. For a multiple selection listbox, this property will toggle the status of the specified item: If the item was selected it won' t be any more or if it wasn' t it will be.

SetRedraw
Type: Bool
When you wish to add or to delete more than 1 item in the listbox, set this property to FALSE (0) before the operations then reset it to TRUE (-1) to refresh the listbox.

VBList Events

DblClick (VBN_DBLCLICK)
Parameters: ID as integer
This event is fired when the user double click on a list item. <ID> is the 0 indexed item number.

SelCancel (VBN_SELCANCEL)
Parameters: <None>
This event is fired when the user cancels his selection (see SDK LBN_SELCANCEL).

SelChange (VBN_SELCHANGE)
Parameters: ID as integer
This event is fired when the user change the selected list item. <ID> is the 0 indexed item number if you use a single selection listbox, otherwise <ID> is the number of selected items.

VBList Methods

AddItem
Parameters: <Item string> [,<index>]
You will use this method to add an item to your listbox. If your listbox is a sorted listbox, you can bypass the sort process if you give a value to <index>. Also, to add an item at the beginning of the list set <index> to 0, if you prefer to add this one to the end, set it to -1. <index> is 0 based. If you choose to use a bitmapped listbox, your item string must be a number (from 1 to 30) and if you use both bitmaps and text datas, you will give your text, the ';' character and the bitmap number like 'MyFirstItem;12'...

RemoveItem
Parameters: [<index>]
Will remove the specify 0 based <index> item or all items if you don' t give any <index> value.

Refresh
will repaint the listbox.

VBList VC++ Extensions

CVBList is the class you must use with VBList controls.

Properties wrappers

CBool Border3D()
CInt Text3D()
CBool AlwaysBitmapped()
CInt BackColor()
CPicture Bitmap01() (to Bitmap30())
CHsz Caption()
CHsz CtlName()
CBool DisableNoScroll()
CBool Enabled()
CBool FontBold()
CHsz FontName()
CFloat FontSize()
CLong ForeColor()
CLong Height()
CInt ItemHeight()
CInt DataType()
CInt DisplayMode()
CLong Left()
CInt ListCount()
BOOL ListIndex(int index = 0)
CString ListString(int index = 0)
CInt MousePointer()
CBool MultipleSel()
CBool NoRedraw()
CBool ParentNotify()
CInt PictureHeight()
CInt PictureWidth()
CInt PictureXMargin()
CInt PictureYMargin()
CBool SetRedraw()
CLong SetSel()
CBool Sort()
CBool TabStop()
CHsz Tag()
CLong Top()
CInt TextAlignment()
CInt TextXMargin()
CInt TextYMargin()
CBool UseHorzScrollBar()
CBool UseTranspBitmap()
CBool Visible()
CLong Width()

Events wrappers

EventDblClkID(LPVOID lp)
used for the VBN_DBLCLIK event and returns <ID>.

EventSelChangeID(LPVOID lp)
used for the VBN_SELCHANGE event and returns <ID>.

EventKeyCode(LPVOID lp)

used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventKeyPress(LPVOID lp)
used for VBN_KEYPRESS and returns the <key code>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBList VBX.

VBList Tips

Using the <Caption> property...

<Caption> can be used to fill the listbox when you don' t have many list items. At design time if <Caption> is blank, the test mode will be active (it will display design-time only items). You can use the '¤' character to specify more than 1 item, like 'Item1¤Item2...'

How to fill the listbox...

Of course, you must fill the listbox according to the <DataType> you choose: If you use text datas there is no special tips, if you use bitmaps you will fill the list with bitmaps numbers (from 1 to 30) and if you use the both, you give the text then the ';' character and the bitmap number, or just the text if you wish to use the standard bitmap.

No events received...

To receive the notification messages you must set <ParentNotify> property to TRUE (-1).

Using the standard bitmap...

You can use a standard bitmap when you set the <AlwaysBitmapped> property to TRUE (-1). In this case, when you add an item without ';' and any bitmap specification, the VBList control will use the first bitmap with this item. If you set <AlwaysBitmapped> to FALSE (0), no bitmap will be displayed for such item.

Style properties

Some properties like <Sort>, <ListHeight>, <MultipleSel> and so on.. change the control style. That' s why you can' t change them dynamically with your application. Also, in some cases, under at design time you will have to reload your forms to reflect the changes.

VBFList

VBFList is the VBX Studio bitmapped files ListBox.

Required file(s): VBFLIST.VBX
Control name: VBFList
MFC Class name: CVBFList (Base class is CListBox)

See also...

VBList

VBFList Properties

Only non standard properties are described below in alphabetic order:

Border3D
Type: Bool
Set this to TRUE (-1) if you wish the listbox has a 3D border at run time.

Text3D
Type: Short
Enables or disables the 3D effect for items text.

BackColor
Type: Short (COLOR40_...)
Specifies the background color. (You will have better results with WHITE).

DataType
Type: Short
You must choose between DIRECTORIES, FILES and USERFILES. UserFiles type allows you to use special filenames (like 'FILE.TXT (size: 0K)') with an associated bitmap.

DiableNoScroll
Type: Bool
Set this property to TRUE (-1) if you wish to use a listbox with a permanent vertical scroll bar (this is not available if you choose the Horizontal scroll bar too).

FilesDisplay
Type: Short
When you use the FILES or the USERFILES data types, you can specify how these files will be displayed: The 'quick' mode only use 2 bitmaps: The first one for program and the second one for data files. The 'Check Winfile' mode checks Winfiles association and use 3 bitmaps (like WinFile does). The 'Extensive' mode is the 'Check Winfile' mode but use many bitmaps instead of 3 (so it requires a little more time for redrawing operations).

Help
Will display this Windows HLP file.

InValue
Type: String
Use this property to specify the initial directory or initial mask depending on the chosen <DataType>. This property has no effect for USERFILES mode. Valid masks are 'C:\WINDOWS' or 'C:\WINDOWS*.TXT' for instance.

ItemHeight
Type: Short
Specifies the height of the items height in the list (in pixels).

MultipleSel
Type: Bool
Set this property to TRUE (-1) if you wish to use a multiple selction\extended selection listbox. Of course this property is not available for DIRECTORIES data type.

ParentNotify
Type: Bool
Set this to TRUE (-1) to receive notifications messages (generally you must set this property to TRUE).

Sort

Type: Bool
Set this property to TRUE (-1) if you wish to use a sorted listbox (this property is not available with the DIRECTORIES data type).

StaticID
Type: Short
If you wish, VBFListBox can automatically fill a static (label) control with the user item choice. To use this function, you must give the ID of this control to this property or use <StaticHwnd> (never use the both!) .

StaticHwnd
Type: Short
If you wish, VBFListBox can automatically fill a static (label) control with the user item choice. To use this function, you must give the HWND of this control to this property or use <StaticID> (never use the both!).

UseHorzScrollBar
Type: Bool
Set to TRUE (-1) if you wish to use an horizontal scroll bar if needed.

vbxAbout
will display the VBX Studio 'About...' box.

VBFList Special Properties

ListCount
Type: Short
Returns the number of items in the listbox

ListIndex(Index)
Type: Bool, property array
Use this property array to retrieve the state of an listbox item. If the specified item (0 indexed) is selected, the property will be set to TRUE (-1), otherwise FALSE (0) is returned.

ListString(Index)
Type: String, property array
Use this property array to retrieve the string (text) of an listbox item. Don' t forget the string won't be the item text as you see it: For a directories listbox the string will be '[<dir>]' for instance. Also, you will use <OutValue> to retrieve this text.

OutValue
Type: String
This property is useful to retrieve the current opened directory because ListString nor SDK message LB_GETCURSEL can give you this value. Also, use this property to retrieve file names and user files names.

SetSel
Type: Long
For a single selection listbox, calling this property with a valid index (0 based) will select specified item. For a multiple selection listbox, this property will toggle the status of the specified item: If the item was selected it won' t be any more or if it wasn' t it will be.

SetRedraw
Type: Bool
When you wish to add or to delete more than 1 item in the listbox, set this property to FALSE (0) before the operations then reset it to TRUE (-1) to refresh the listbox.

VBFList Events

DblClick (VBN_DBLCLICK)
Parameters: ID as integer
This event is fired when the user double click on a list item. <ID> is the 0 indexed item number.

SelCancel (VBN_SELCANCEL)
Parameters: <None>
This event is fired when the user cancels his selection (see SDK LBN_SELCANCEL).

SelChange (VBN_SELCHANGE)
Parameters: ID as integer
This event is fired when the user change the selected list item. <ID> is the 0 indexed item number if you use a single selection listbox, otherwise <ID> is the number of selected items.

VBFList Methods

AddItem

Parameters: <Item string> [,<index>]

You will use this method to add an item to your listbox when you set the <DataType> property to USERFILES. To add your 'user defined filename' you will can use what you want, the ';' character and the file extension in uppercase. So to add a specific file name you can call (for instance):

```
VBFList1.AddItem "FILES.TXT (218.236 bytes);TXT", -1
```

If you use a sorted listbox, you can bypass the sort process if you give a value to <index>. Also, to add an item at the beginning of the list set <index> to 0, if you prefer to add this one to the end, set it to -1. <index> is 0 based.

RemoveItem

Parameters: [<index>]

Will remove the specify 0 based <index> item or all items if you don' t give any <index> value. You will use this method with the USERFILES <DataType>.

Refresh

will repaint the listbox.

VBFList VC++ Extensions

CVBFList is the class you must use with VBFList controls.

Properties wrappers

CBool Border3D()
CInt Text3D()
CInt BackColor()
CHsz CtlName()
CInt DataType()
CBool DisableNoScroll()
CBool Enabled()
CInt FilesDisplay()
CHsz FontName()
CBool FontBold()
CFloat FontSize()
CLong ForeColor()
CLong Height()
CHsz InValue()
CInt ItemHeight()
CLong Left()
CInt ListCount()
BOOL ListIndex(int index = 0)
CString ListString(int index = 0)
CBool MultipleSel()
CHsz OutValue()
CBool ParentNotify()
CBool SetRedraw()
CLong SetSel()
CBool Sort()
CInt StaticHwnd()
CInt StaticID()
CBool TabStop()
CHsz Tag()
CLong Top()
CBool UseHorizScrollBar()
CBool Visible()
CLong Width()

Events wrappers

EventDblClkID(LPVOID lp)
used for the VBN_DBLCLIK event and returns <ID>.

EventSelChangeID(LPVOID lp)
used for the VBN_SELCHANGE event and returns <ID>.

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventKeyPress(LPVOID lp)
used for VBN_KEYPRESS and returns the <key code>.

EventMouseBtn(LPVOID lp)

used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBFList VBX.

VBFList Tips

No mouse!

With the directory <DataType> and when no mouse is available, you can use the spacebar to 'open' a directory.

Using <DataType>...

If you set this property to USERFILES, you will be able to use customised files name and files list with your VBFList VBX. Basically there are 2 ways to use this: For instance you wish to add size informations to your list. Then you will use the <AddItem> method with something like this "MYFILE.EXT (size);EXT". The other way to use this mode is to make a list from multiple directories or something else (like a files list in a ZIP file): Then you have to manually fill the list. See <DataType> and <AddItem> for more informations.

Using the static control...

VB standard text control (named 'Label") can' t be use with the VBFList <StaticHwnd> because this control doesn' t have a standard <Hwnd> property, so, you will use the VBText control instead. VBFlist will change this control text each time the user 'open' a directory: So this text control will reflect the current directory. However there is a limitation with this process: The VBFList control will put the text in the control even if there is no more room for this text (you won' t have something like 'C:\dir\...\final.dir'), to make this nicer, you can use the String Studio 1.2 library (<CheckDirSize> function).

Style properties

Some properties like <Sort>, <ListHeight>, <MultipleSel> and so on.. change the control style. That' s why you can' t change them dynamically with your application. Also, in some cases, under at design time you will have to reload your forms to reflect the changes.

Using ApStudio...

If you experience some troubles when you test your application, close the ApStudio program (VBX Studio VBXs don't like to be loaded in the design time and the run time systems at the same time!).

VBCombo

VBCombo is the VBX Studio bitmapped ComboBox.

Required file(s): *VBCOMBO.VBX*
Control name: *VBCombo*
MFC Class name: *CVBCombo (Base class is CComboBox)*

See also...

VBFCombo

VBCombo Properties

Only non standard properties are described below in alphabetic order:

Border3D
Type: Bool
Set this to TRUE (-1) if you wish the combobox has a 3D border at run time.

Text3D
Type: Short
Enables or disables the 3D effect for items text.

AlwaysBitmapped
Type: Bool
Set this property to TRUE (-1) if you wish all items with no bitmap specifications display the first bitmap.

AutoSearchInList
Type: Bool
If this property is set to TRUE, each time the user press a key the listbox will appear.

BackColor
Type: Short (COLOR40_...)
Specifies the background color.

Bitmap01 to Bitmap30
Type: Picture (bitmap)
These are the bitmaps you will use in your combobox. Calling a picture depends on the Data type you 've chosen (see <Caption> property below).

BitmapWidth
Type: Short
Specifies the bitmap width in pixels when you are on the 'normal' display mode.

BitmapHeight
Type: Short
Specifies the bitmap height in pixels when you are on the 'normal' display mode.

Caption
Type: String
This string property is used to intialize the listbox: To use the caption you must disable the TestMode property. You can give multiple items data: Use '¤' to separate each item, like this 'Item1¤Item2¤Item3...'. The caption can' t be longer than 255 characters. Item data depends on the data type: For text datas, nothing special; For bitmap only datas, give a number to specify the bitmap number like: '1¤2¤2' and for 'both' datas use ';' to separate text data and bitmap data like this: 'Our text 1 use bitmap 1;1¤Our text 2 uses bitmap 2;2'... If you choose the always bitmapped option, in the 'both' data type, you can give 'Item1' and this will display the bitmap 1 and 'Item1' text (no ';' required).

DataType
Type: Short
You must choose between TEXT, BITMAP and BOTH (text and bitmap).

DisplayMode
Type: Short
Specifies the bitmap display mode. Choosing the 'NORMAL' mode allows you to use margin values and stretch mode ('AUTO') is not available with transparent bitmaps.

Help

Will display this Windows HLP file.

InitialValue
Type: Long
Specifies the 0 based initial value for the combobox. Specified item will be selected when the dialog box appears.

ItemHeight
Type: Short
Specifies the height of the items height in the list (in pixels).

ListHeight
Type: Short
Specifies the height of the listbox (in pixels).

StaticHeight
Type: Short
Specifies the height of the static zone of the combobox (in pixels).

TestMode
Type: Bool
If this property is set to TRUE (-1) the test mode is active: the combobox will use special built-in items and will display all selected bitmaps. When your combobox seems to be OK, don' t forget to set this property to FALSE (0)!

TextAlignment
Type: Short
Specifies the text alignment.

UseTranspBitmap
Type: Bool
Set to TRUE (-1) if you wish to 'transparent' bitmaps (e.g. Bitmaps using the light fuchsia color as the transparent color). Transparent bitmaps are not available with stretch display mode.

XMarginBitmap
Type: Short
Specifies the bitmap horizontal margin (in pixels). To use it you must set <DisplayMode> to normal.

YMarginBitmap
Type: Short
Specifies the bitmap vertical margin (in pixels). To use it you must set <DisplayMode> to normal.

XMarginText
Type: Short
Specifies the text horizontal margin (in pixels). To use it you must set <DisplayMode> to normal.

YMarginText
Type: Short
Specifies the text vertical margin (in pixels). To use it you must set <DisplayMode> to normal.

vbxAbout
will display the VBX Studio 'About...' box.

VBCombo Special Properties

ListCount
Type: Short
Returns the number of items in the listbox

ListIndex(Index)
Type: Bool, property arrays
Use this property array to retrieve the state of an listbox item. If the specified item (0 indexed) is selected, the property will be set to TRUE (-1), otherwise FALSE (0) is returned.

ListString(Index)
Type: String, property arrays
Use this property array to retrieve the string (text) of an listbox item.

OutValue
Type: String
This property is useful to retrieve the current selected string. If you' ve chosen the BOTH data type, you will need to extract the left part of the item (to the ';' character) because the second part can be the item bitmap number.

SetSel
Type: Long
For a single selection listbox, calling this property with a valid index (0 based) will select specified item. For a multiple selection listbox, this property will toggle the status of the specified item: If the item was selected it won' t be any more or if it wasn' t it will be.

VBCombo Events

Click (VBN_CLICK)
Parameters: ID as integer, Buffer as String
This event is fired when the user click on an item. <ID> is the 0 indexed item number, <String> is the label of the selected item. Don' t forget this label can include a bitmap specification like 'MyItem;1'.

CloseUp (VBN_CLOSEUP)
Parameters: <None>
This event is fired when the user close the listbox (see SDK CBN_CLOSEUP).

DropDown (VBN_DROPDOWN)
Parameters: <None>
This event is fired when the user open the listbox (see SDK CBN_DROPDOWN).

SelCancel (VBN_SELCANCEL)
Parameters: <None>
This event is fired when the user cancels his selection (see SDK CBN_SELCANCEL).

SelCancel (VBN_SELENDOK)
Parameters: <None>
This event is fired when the user validate his selection (see SDK CBN_SELENDOK).

SelChange (VBN_SELCHANGE)
Parameters: ID as integer, Buffer as String
This event is fired when the user click on an item. <ID> is the 0 indexed item number, <String> is the label of the selected item. Don' t forget this label can include a bitmap specification like 'MyItem;1'.

VBCombo Methods

AddItem
Parameters: <Item string> [,<index>]
You will use this method to add an item to your combobox listbox. Also, to add an item at the beginning of the list set <index> to 0, if you prefer to add this one to the end, set it to -1. <index> is 0 based. If you choose to use a bitmapped listbox, your item string must be a number (from 1 to 30) and if you use both bitmaps and text datas, you will give your text, the ';' character and the bitmap number like 'MyFirstItem;12'...

RemoveItem
Parameters: [<index>]
Will remove the specify 0 based <index> listbox item or all items if you don' t give any <index> value.

Refresh
will repaint the combobox.

VBCombo VC++ Extensions

CVBCombo is the class you must use with VBCombo controls.

Properties wrappers

CBool Border3D()
CInt Text3D()
CBool AlwaysBitmapped()
CBool AutoSearchInList()
CInt BackColor()
CPicture Bitmap01() to Bitmap30()
CInt BitmapHeight()
CInt BitmapWidth()
CHsz CtlName()
CHsz Caption()
CInt DataType()
CInt DisplayMode()
CBool Enabled()
CHsz FontName()
CBool FontBold()
CFloat FontSize()
CLong ForeColor()
CLong Height()
CLong InitialValue()
CInt ItemHeight()
CLong Left()
CInt ListCount()
CInt ListHeight()
BOOL ListIndex(int index = 0)
CString ListString(int index = 0)
CHsz OutValue()
CLong SetSel()
CInt StaticHeight()
CBool TabStop()
CHsz Tag()
CBool TestMode()
CInt TextAlignment()
CLong Top()
CBool UseTranspBitmap()
CBool Visible()
CLong Width()
CInt XMarginBitmap()
CInt XMarginText()
CInt YMarginBitmap()
CInt YMarginText()

Events wrappers

EventClickID(LPVOID lp)
used for the VBN_DBLCLIK event and returns <ID>.

EventClickString(LPVOID lp)
used for the VBN_DBLCLIK event and returns <Buffer>.

EventSelChangeID(LPVOID lp)
used for the VBN_SELCHANGE event and returns <ID>.

EventSelChangeString(LPVOID lp)
used for the VBN_SELCHANGE event and returns <Buffer>.

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventKeyPress(LPVOID lp)
used for VBN_KEYPRESS and returns the <key code>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBCombo VBX.

VBCombo Tips

Sorting...
VBCombo listboxes are always sorted. If you wish to bypass this, just give an index to the <AddItem> method (also, -1 will add the item at the end of the listbox). See <AddItem> method for more informations.

Using the <TestMode> property...
When <TestMode> is TRUE (-1), the combobox will be filled with test items to help you to build your combox at design time. Before you distribute your application, do not forget to set this property to FALSE (0).

Using the <Caption> property...
<Caption> can be used to fill the listbox when you don' t have many list items. To activate this, set <TestMode> property to FALSE (0). You can use the '¤' character to specify more than 1 item, like 'Item1¤Item2...'

How to fill the listbox...
Of course, you must fill the listbox according to the <DataType> you choose: If you use text datas there is no special tips, if you use bitmaps you will fill the list with bitmaps numbers (from 1 to 30) and if you use the both, you give the text then the ';' character and the bitmap number, or just the text if you wish to use the standard bitmap.

Using the standard bitmap...
You can use a standard bitmap when you set the <AlwaysBitmapped> property to TRUE (-1). In this case, when you add an item without ';' and any bitmap specification, the VBCombo control will use the first bitmap with this item. If you set <AlwaysBitmapped> to FALSE (0), no bitmap will be displayed for such item.

Style properties
Some properties like <Sort>, <ListHeight>, <MultipleSel> and so on.. change the control style. That' s why you can' t change them dynamically with your application. Also, in some cases, under at design time you will have to reload your forms to reflect the changes.

VBFCombo

VBFCombo is the VBX Studio bitmapped files ComboBox.

Required file(s): *VBFCOMBO.VBX*
Control name: *VBFCombo*
MFC Class name: *CVBFCombo (Base class is CComboBox)*

See also...

VBCombo

VBFCombo Properties

Only non standard properties are described below in alphabetic order:

Border3D
Type: Bool
Set this to TRUE (-1) if you wish the combobox has a 3D border at run time.

Text3D
Type: Short
Enables or disables the 3D effect for items text.

AutoSearchInList
Type: Bool
If this property is set to TRUE, each time the user press a key the listbox will appear.

BackColor
Type: Short (COLOR40_...)
Specifies the background color.

DataType
Type: Short
You must choose between DRIVES, FILES and USERFILES. UserFiles type allows you to use special filenames (like 'FILE.TXT (size: 12 Kb.)') with an associated bitmap.

FilesDisplay
Type: Short
When you use the FILES or the USERFILES data types, you can specify how these files will be displayed: The 'quick' mode only use 2 bitmaps: The first one for program and the second one for data files. The 'Check Winfile' mode checks Winfiles association and use 3 bitmaps (like WinFile does). The 'Extensive' mode is the 'Check Winfile' mode but use many bitmaps instead of 3 (so it requires a little more time for redrawing operations).

Help
Will display this Windows HLP file.

InitialValue
Type: Long
Specifies the 0 based initial value for the combobox. Specified item will be selected when the dialog box appears.

InValue
Type: String
Use this property to specify the initial mask or directory depending on the chosen DataType. This property has no effect for USERFILES mode. Valid masks are like 'C:\WINDOWS*.*' or '*.*'.

ItemHeight
Type: Short
Specifies the height of the items height in the list (in pixels).

ListHeight
Type: Short
Specifies the height of the listbox (in pixels).

StaticHeight
Type: Short
Specifies the height of the static zone of the combobox (in pixels).

TextAlignment
Type: Short
Specifies the text alignment.

vbxAbout
will display the VBX Studio 'About...' box.

VBFCombo Special Properties

ListCount
Type: Short
Returns the number of items in the listbox

ListIndex(Index)
Type: Bool, property arrays
Use this property array to retrieve the state of an listbox item. If the specified item (0 indexed) is selected, the property will be set to TRUE (-1), otherwise FALSE (0) is returned.

ListString(Index)
Type: String, property arrays
Use this property array to retrieve the string (text) of an listbox item. For a drives combobox the string will be '[-<drive>-]' for instance. Also, you will use <OutValue> to retrieve this text.

OutValue
Type: String
This property is useful to retrieve the current opened directory because ListString nor SDK message LB_GETCURSEL can give you this value. Also, use this property to retrieve file names and user files names.

SetSel
Type: Long
For a single selection listbox, calling this property with a valid index (0 based) will select specified item. For a multiple selection listbox, this property will toggle the status of the specified item: If the item was selected it won' t be any more or if it wasn' t it will be.

VBFCombo Events

CloseUp (VBN_CLOSEUP)
Parameters: <None>
This event is fired when the user close the listbox (see SDK CBN_CLOSEUP).

DropDown (VBN_DROPDOWN)
Parameters: <None>
This event is fired when the user open the listbox (see SDK CBN_DROPDOWN).

SelCancel (VBN_SELCANCEL)
Parameters: <None>
This event is fired when the user cancels his selection (see SDK CBN_SELCANCEL).

SelCancel (VBN_SELENDOK)
Parameters: <None>
This event is fired when the user validate his selection (see SDK CBN_SELENDOK).

SelChange (VBN_SELCHANGE)
Parameters: ID as integer, Buffer as String
This event is fired when the user click on an item. <ID> is the 0 indexed item number, <String> is the label of the selected item. When you use a 'drives' combobox, <Buffer> will be the item text as you see it (like <OutValue>). Also, if you use the 'UserFiles' mode, this text will include all the item text (with ';EXT').

VBFCombo Methods

AddItem

Parameters: <Item string> [,<index>]

You will use this method to add an item to your listbox when you set the <DataType> property to USERFILES. To add your 'user defined filename' you will can use what you want, the ';' character and the file extension in uppercase. So to add a specific file name you can call (for instance):

```
VBFCombo1.AddItem "FILES.TXT (218.236 bytes);TXT", -1
```

Also, to add an item at the beginning of the list set <index> to 0, if you prefer to add this one to the end, set it to -1. <index> is 0 based.

RemoveItem

Parameters: [<index>]

Will remove the specify 0 based <index> item or all items if you don' t give any <index> value. You will use this method with the USERFILES <DataType>.

Refresh

will repaint the combobox.

VBFCombo VC++ Extensions

CVBFCombo is the class you must use with VBFCombo controls.

Properties wrappers

CInt Text3D()
CBool Border3D()
CBool AutoSearchInList()
CHsz CtlName()
CInt DataType()
CBool Enabled()
CInt FilesDisplay()
CHsz FontName()
CBool FontBold()
CFloat FontSize()
CLong ForeColor()
CLong Height()
CLong InitialValue()
CHsz InValue()
CInt ItemHeight()
CLong Left()
CInt ListCount()
CInt ListHeight()
BOOL ListIndex(int index = 0)
CString ListString(int index = 0)
CHsz OutValue()
CLong SetSel()
CInt StaticHeight()
CInt BackColor()
CInt TabIndex()
CBool TabStop()
CHsz Tag()
CInt TextAlignment()
CLong Top()
CBool Visible()
CLong Width()

Events wrappers

EventSelChangeID(LPVOID lp)
used for the VBN_SELCHANGE event and returns <ID>.

EventSelChangeString(LPVOID lp)
used for the VBN_SELCHANGE event and returns <Buffer>.

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventKeyPress(LPVOID lp)
used for VBN_KEYPRESS and returns the <key code>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBFCombo VBX.

VBFCombo Tips

Sorting...

VBFCombo comboboxes (like VBCombo) are always sorted. If you wish to bypass this, just give an index to the <AddItem> method (also, -1 will add the item at the end of the listbox).

Using <DataType>...

If you set this property to USERFILES, you will be able to use customised files name and files list with your VBFCombo VBX. Basically there are 2 ways to use this: For instance you wish to add size informations to your list. Then you will use the <AddItem> method with something like this "MYFILE.EXT (size);EXT". The other way to use this mode is to make a list from multiple files lists or something else (like a files list in a ZIP file): Then you have to manually fill the list. See <DataType> and <AddItem> for more informations.

Style properties

Some properties like <Sort>, <ListHeight>, <MultipleSel> and so on.. change the control style. That' s why you can' t change them dynamically with your application. Also, in some cases, under at design time you will have to reload your forms to reflect the changes.

VBEdit

VBEdit is the VBX Studio pictured Edit control.

Required file(s): *VBEDIT.VBX*
Control name: *VBEdit*
MFC Class name: *CVBEdit (Base class is CEdit)*

See also...

VBText
VBSpin

VBEdit Properties

Only non standard properties are described below in alphabetic order:

Border3D

Type: Bool

Set this to TRUE (-1) if you wish the edit control has a 3D border at run time.

AllowAutoChange

Type: Bool

If TRUE (-1) the edit control can add characters if needed (specially with time or drive input).

AllowBlankField

Type: Bool

If TRUE (-1) <IsValid> property will return TRUE if there is no text in the edit control, otherwise the 'normal' checking process is performed.

BeepIfError

Type: Bool

Specify if a little beep is generated in case the user press an invalid key.

Case

Type: Short

You must choose a case between NO CASE, UPPERCASE and LOWERCASE. Depending on this choice, all user input will or won' t be converted in the specified case style.

DataType

Type: Short

Specify an input mask. NONE will disable input checking. The alphabetic mask doesn' t allow the '_' character, the drives mask accepts valid drives only, time and date masks required a valid date and the last 2 files masks accept names with or without a final '\'. Filenames and directory names don' t accept the '/' directories separator (only '\' is allowed) but they accept a name with a final '.'. Depending on the chosen mask, <IsValid> property will be set to TRUE or FALSE.

Help

Will display this Windows HLP file.

IncrementValue

Type: Short

Sets the increment value.

MaxLength

Type: Short

Specify the maximum length for the user input.

MaxRange

Type: Float (real)

Sets the maxrange for numerical input. This value must be positive. To disable this property, give the same value to <MinRange>.

MinRange

Type: Float (real)

Sets the minrange for numerical input. This value can be negative. To disable this property, give the same value to <MaxRange>.

NoHideSelection

Type: Bool

If this property is set to TRUE (-1) the edit selection will be kept when the edit control losts the focus.

OEMConvert
Type: Bool
If this property is set to TRUE (-1) the edit control will convert OEM characters.

PasswordChars
Type: Bool
If this property is set to TRUE (-1) the edit control will display '*' instead of the user input.

ReadOnly
Type: Bool
If this property is set to TRUE (-1) the edit text won' t be changed nor deleted.

Text
Type: String
This is the control text. This string can' t be a multiline string.

UseSystemColors
Type: Bool
If this property is set to TRUE (-1) the edit control will check and use user's configuration colors instead of colors you' ve specified at design time.

vbxAbout
will display the VBX Studio 'About...' box.

VBEdit Special Properties

Decrement
Type: Bool
When you set this property to TRUE (-1), the edit control current value will be decremented using the mask. If the edit value is not valid, it will be changed into a valid text. When you use this property with the VBSpin VBX you must reset the VBSpin <Value> property to 0.

Increment
Type: Bool
When you set this property to TRUE (-1), the edit control current value will be incremented using the mask. If the edit value is not valid, it will be changed into a valid text. When you use this property with the VBSpin VBX you must reset the VBSpin <Value> property to 0.

IsValid
Type: Bool
This property is set to TRUE if the user input is correct, this depends on the current mask.

SetSel
Type: Bool
When you set this property to TRUE (-1), the edit control text will be selected. In most cases, you' ll have to use a SetFocus afterwise.

VBEdit Events

Change (VBN_CHANGE)
Parameters: <None>
This event is fired when the user change the VBEdit text (see SDK EN_CHANGE).

Update (VBN_UPDATE)
Parameters: <None>
This event is fired when the user change the VBEdit text (see SDK EN_UPDATE).

MaxText (VBN_MAXTEXT)
Parameters: <None>
This event is fired when the user attempt to put too much datas in the VBEdit control (see SDK EN_MAXTEXT).

VBEdit Methods

Refresh
will repaint the edit control.

VBEdit VC++ Extensions

CVBEdit is the class you must use with VBEdit controls.

Properties wrappers

CBool Border3D()
CBool AllowBlankField()
CBool AllowAutoChange()
CLong BackColor()
CBool BeepIfError()
CInt BorderStyle()
CInt Case()
CHsz CtlName()
CBool Decrement()
CInt DataType()
CBool Increment()
CInt IncrementValue()
CBool IsValid()
CBool Enabled()
CBool FontBold()
CHsz FontName()
CFloat FontSize()
CLong ForeColor()
CLong Height()
CLong Left()
CInt MaxLength()
CFloat MaxRange()
CFloat MinRange()
CBool NoHideSelection()
CBool OEMConvert()
CBool PasswordChars()
CBool ReadOnly()
CBool SetSel()
CBool TabStop()
CHsz Tag()
CHsz Text()
CLong Top()
CBool UseSystemColors()
CBool Visible()
CLong Width()

Events wrappers

<None>

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBEdit VBX.

VBEdit Tips

Using system colors...

In many cases it' s a good idea to use the user's configuration system colors (and set the <UseSystemColors> property to TRUE (-1)). To do this perfectly, disable the <Border3D> property and use the VB3D VBX with its <UseSystemColors> set to TRUE. So, the edit control and its border will be adjusted to the user colors choice.

Style properties

Some properties like <Sort>, <ListHeight>, <MultipleSel> and so on.. change the control style. That' s why you can' t change them dynamically with your application. Also, in some cases, under at design time you will have to reload your forms to reflect the changes.

Using VBEdit with VBSpin...

VBEdit can be used with VBSpin and it manages (itself) the VBSpin: You just have to put a VBEdit handler like 'VBEdit1.Increment = True' behind the VBSpin scrolling notification messages and then reset the VBSpin <Value> property to 0 (or something else inside the VBSpin range). When you use this method you must disable the VBSpin range checking. VBEdit doesn' t handle float datas increment.

Using the <IsValid> property...

This property will be set to TRUE (-1) if the user give a correct value. If you set <AllowBlankField>, this property will be TRUE if there is no text. VBEdit uses a double checking process: The first one occurs when the user press a key and the second one will be performed when you have a look at this property. Also, it' s a good idea to read this property each time you receive a VBEdit <Update> notification message in order to enable or disable dialog box buttons..

Using String Studio...

If you build MFC applications, you will find a big advantage in using String Studio CStr class with the VBEdit control: This library can help you to format string, to use numbers and dates and to manage your control from a string! Also, this library is required to use CTime DDX routines.

VBSpin

VBSpin is the VBX Studio 3D spin control.

Required file(s): *VBSPIN.VBX, SPINDLL.DLL*
Control name: *VBSpin*
MFC Class name: *CVBSpin (Base class is CStatic)*

See also...

VBEdit

VBSpin Properties

Only non standard properties are described below in alphabetic order:

ArrowsColor

Type: Long (COLORREF)

Changes the arrows color.

AssociatedHwnd

Type: Short

Giving a valid edit control HWND value to this property will activate the 'automatic' mode. The automatic mode will fill this edit control each time the spin control is pressed. The automatic mode won' t change the edit text if this text is not valid when the spin control is called (to avoid this, use the VBEdit <Increment> and <Decrement> functions instead). If you give an HWND to this property, do not give an ID to the <AssociatedID> property.

AssociatedID

Type: Short

Giving a valid edit control ID value to this property will activate the 'automatic' mode. The automatic mode will fill this edit control each time the spin control is pressed. The automatic mode won' t change the edit text if this text is not valid when the spin control is called (to avoid this, use the VBEdit <Increment> and <Decrement> functions instead). If you give an ID to this property, do not give an HWND to the <AssociatedHwnd> property.

AutoPos

Type: Bool

This property doesn' t work with VB! With VC++ set this property to TRUE (-1) and the spin control will look for the previous control and then will position the control close to this control. Also, if you wish to use this property, build your edit control just before you build the spin.

BorderColor

Type: Long (COLORREF)

Changes the spin border (the frame) color.

HandCursor

Type: Bool

If set to TRUE (-1) a hand cursor will be used with the spin control (run time only).

Help

Will display this Windows HLP file.

Maximum

Type: Float (real)

Give the maximum value for the spin. This value must be positive. If you use the VBEdit <Increment> and <Decrement> properties to handle your spin, in most case you will give 1 to this property and each time you receive a scrolling notification message you will reset the <Value> property to 0.

Minimum

Type: Float (real)

Give the minimum value for the spin. This value can be negative. If you use the VBEdit <Increment> and <Decrement> properties to handle your spin, in most case you will give -1 to this property and each time you receive a scrolling notification message you will reset the <Value> property to 0.

UseSystemColors

Type: Bool

If this property is set to TRUE (-1) the spin control will check and use user's configuration colors (button section) instead of colors you' ve specified at design time.

Value

Type: Float (real)

For a manual spin handling you must update this value each time the spin control is called and each time the user changes the associated text control. This value can be negative. If you use the VBEdit <Increment> and <Decrement> properties to handle your spin, in most case you will give 0 to this property, -1 to <Minimum> and 1 to <Maximum>, because the VBEdit control does all necessary works. To do this, you must reset the <Value> property to 0 each time you receive a VBSpin scrolling notification message. Also, for the 'automatic' mode (when you give a value for <AssociatedID> or <AssociatedHwnd>), you don' t have to worry about this property. See VBSpin tips for additional infos.

vbxAbout

will display the VBX Studio 'About...' box.

VBSpin Special Properties

<none>

VBSpin Events

ScrollUp (VBN_SCROLLUP)

Parameters: <None>

This event is fired when the user click on the upper spin arrow. If you don' t wish to use the VBSpin range checking, don' t forget to reset the <Value> property (to 0) each time you receive this message.

ScrollDn (VBN_SCROLLDN)

Parameters: <None>

This event is fired when the user click on the lower spin arrow. If you don' t wish to use the VBSpin range checking, don' t forget to reset the <Value> property (to 0) each time you receive this message.

VBSpin Methods

Refresh
will repaint the spin control.

VBSpin VC++ Extensions

CVBSpin is the class you must use with VBSpin controls.

Properties wrappers

CLong ArrowsColor()
CInt AssociatedHwnd()
CInt AssociatedID()
CBool AutoPos()
CLong BorderColor()
CHsz CtlName()
CBool Enabled()
CBool HandCursor()
CLong Height()
CLong Left()
CFloat Maximum()
CFloat Minimum()
CBool TabStop()
CHsz Tag()
CLong Top()
CBool UseSystemColors()
CFloat Value()
CBool Visible()
CLong Width()

Events wrappers

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventKeyPress(LPVOID lp)
used for VBN_KEYPRESS and returns the <key code>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBSpin VBX.

VBSpin Tips

Handling the spin

Basically there are 3 ways to manage the VBSpin control: For simple numeric values, the best way is to use the 'automatic' mode. In this case, you must give a value for <AssociatedID> or <AssociatedHwnd>. With this mode, if the user give an invalid value in the text control, the spin doesn' t perform any action.
The second way is to use the VBEdit control and its <Increment> and <Decrement> properties. To use this mode, disable the spin ranges (set <Minimum> to -1, <Maximum> to 1 and <Value> to 0) because the VBEdit doesn' t need this range checking. VBEdit can change the text if this one is not valid and this control can perform special processing for dates and drives for instance.
To do this, you must reset the <Value> property to 0 each time you receive a VBSpin scrolling notification message. However, the VBEdit doesn' t handle float increment nor specific datas. So the third way is to handle VBSpin notifications message and do it yourself. In this case, you must give values for <Minimum>, <Maximum> and you must update the <Value> property each time the text control is updated.

Using String Studio...

If you build MFC applications, you will find a big advantage in using String Studio CStr class with the VBSpin control: This library can help you to format string and to use numbers and dates for instance.

VBGauge

VBGauge is the VBX Studio gauge (percentage) control.

Required file(s): *VBGAUGE.VBX*
Control name: *VBGauge*
MFC Class name: *CVBGauge (Base class is CStatic)*

VBGauge Properties

Only non standard properties are described below in alphabetic order:

Border3D
Type: Bool
Set this to TRUE (-1) if you wish the edit control has a 3D border at run time.

BackColor
Type: Short (COLOR40_...)
Specifies the background color.

Gauge3DBorder
Type: Bool
Set this to TRUE (-1) if you wish the gauge has a small 3D border.

GaugeColor
Type: Short (COLOR40_...)
Specifies the gauge color (this is the foreground color).

Help
Will display this Windows HLP file.

Maximum
Type: Float (real)
This is the maximum value and it must be positive. This property will be used to compute the rate.

Minimum
Type: Float (real)
This is the minimum value and it can be negative.This property will be used to compute the rate.

ShowTheRate
Type: Bool
Set this to TRUE (-1) if you wish to show the gauge rate (percentage). If you choose this option, because the rate string are always black, specify light colors for both <BackColor> and <GaugeColor>.

Value
Type: Float (real)
Current value. If you prefer to use a rate, set <Minimum> to 0 and <Maximum> to 100. If you use VC++ ApStudio, you must call the <Refresh> method to reflect the change.

VariableColors
Type: Bool
If set to TRUE (-1), the gauge foreground color will be red if the rate is lower than 33%, yellow if the rate is higher than 33% and smaller than 66% and green if not.

vbxAbout
will display the VBX Studio 'About...' box.

VBGauge Special Properties

<none>

VBGauge Events

<none>

VBGauge Methods

Refresh
will repaint the gauge control.

VBGauge VC++ Extensions

CVBGauge is the class you must use with VBGauge controls.

Properties wrappers

CBool Border3D()
CInt BackColor()
CInt BorderStyle()
CHsz CtlName()
CBool Enabled()
CBool Gauge3DBorder()
CInt GaugeColor()
CLong Height()
CLong Left()
CFloat Maximum()
CFloat Minimum()
CBool ShowTheRate()
CBool TabStop()
CHsz Tag()
CLong Top()
CFloat Value()
CBool VariableColors()
CBool Visible()
CLong Width()

Events wrappers

<None>

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBGauge VBX.

VBGauge Tips

How to display the rate...

To display the gauge rate (value), set the <ShowTheRate> property to TRUE (-1). Also to use this, choose a light background color (light grey or white for instance) and do the same for the foreground color because the rate use the black pen.

VBTab

VBTab is the VBX Studio tab control.

Required file(s): *VBTAB.VBX*
Control name: *VBTab*
MFC Class name: *CVBTab (Base class is CButton)*

See also...

VBLine

VBTab Properties

Only non standard properties are described below in alphabetic order:

Text3D
Type: Short
Enables or disables the 3D effect for the tab text (caption property).

Active
Type: Bool
You will use this property in 2 ways: The first one at design time, you will set this property to TRUE (-1) to activate this tab (only one tab can be active, the others will have this property set to FALSE (0)). At run time, when the user click on a tab, this tab <Active> property will be set to TRUE (-1) and the other tabs <Active> property will be set to FALSE (0).

FirstLeft
Type: Bool
Set this property to TRUE (-1) if the tab is the most left tab of your tabs line.

FirstLine
Type: Bool
Set this property to TRUE (-1) if the tab is in the first line tabs. When this property is set to TRUE (-1), when the user click on the tab button, this tab will become <Active>, otherwise (if <FirstLine> is FALSE) the text will be exchanged with the current (<FirstLine>) active tab. If you don' t use multiple lines, all your tabs must have this property set to TRUE (-1).

FirstRight
Type: Bool
Set this property to TRUE (-1) if the tab is the most right tab of your tabs line.

FlashColor
Type: Long (COLORREF)
Specifies the background color of selected tab text.

FlashTxtColor
Type: Long (COLORREF)
Specifies the foreground color of selected tab text.

HandCursor
Type: Bool
If set to TRUE (-1) a hand cursor will be used with the tab control (run time only).

Help
Will display this Windows HLP file.

UseSystemColors
Type: Bool
If this property is set to TRUE (-1) the tab control will check and use user's configuration colors (button section) instead of colors you' ve specified at design time.

TabCorners
Type: Bool
Set this property to TRUE (-1) if you prefer rounded corners instead of square corners.

vbxAbout
will display the VBX Studio 'About...' box.

VBTab Special Properties

<none>

To retrieve the state of a tab, read the <Active> property.

VBTab Events

<none>

The most useful property is <Click> (VBN_CLICK): When the user selects a VBTab button, you will receive this notification message.

VBTab Methods

Refresh
will repaint the VBTab control.

VBTab VC++ Extensions

CVBTab is the class you must use with VBTab controls.

Properties wrappers

CInt Text3D()
CBool Active()
CLong BackColor()
CHsz Caption()
CHsz CtlName()
CBool Enabled()
CBool FirstLeft()
CBool FirstLine()
CBool FirstRight()
CLong FlashColor()
CLong FlashTxtColor()
CBool FontBold()
CBool FontItalic()
CHsz FontName()
CFloat FontSize()
CBool FontUnderline()
CLong ForeColor()
CBool HandCursor()
CLong Height()
CLong Left()
CInt MousePointer()
CBool TabCorners()
CBool TabStop()
CHsz Tag()
CLong Top()
CBool UseSystemColors()
CBool Visible()
CLong Width()

Events wrappers

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventKeyPress(LPVOID lp)
used for VBN_KEYPRESS and returns the <key code>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)

used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*

returns a CVBControl pointer for the current CVBTab VBX.

VBTab Tips

Drawing the tabbed dialog box

To draw a tabbed dialog, have a look at provided samples. You must use VBLine tab style controls to draw the rectangles. The biggest problem is to use different controls for each tab. The easiest way, we think, is to build some 'normal' dialog boxes, to test them and, when everything seems to be OK, to set the control <Visible> property to FALSE (0) and to copy them into your main tabbed dialog box. Inside AppStudio, you can make a Drag 'n Drop and edit the resource script (*.RC).

Handling the tabs

Using a single line of tabs is quite more easier (and maybe more 'natural') than using multiple tabs lines dialog box. With a single tabs line, the tabs label (<Caption>) can' t change, so you just have to put a simple handler behind each tab button. With multiple tabs lines, these texts can change so you must check the <Caption> property each time a tab is clicked. You can have a look at provided samples source code to get informations to build your tab handlers.

See also...

For your first tabbed dialog box you can find some tips with the VC++ and VB samples.

VBPicBtn

VBPicBtn is the VBX Studio pictured button.

Required file(s): *VBPICBTN.VBX*
Control name: *VBPicBtn*
MFC Class name: *CVBPicBtn (Base class is CButton)*

See also...

VBAnim

VBPicBtn Properties

Only non standard properties are described below in alphabetic order:

AlwaysRepaint

Type: Bool

If this property is set to TRUE (-1), a background rectangle will be displayed each time the picture change. This is required when the picture has a different size from <PictureUp> to <PictureDown>.

BitmapHeight

Type: Short

Specifies the picture height in pixels in the 'normal' display mode.

BitmapWidth

Type: Short

Specifies the picture width in pixels in the 'normal' display mode.

CornersColor

Type: Long (COLORREF)

This is the color for the button corners when you set <ShowBorder> to TRUE (-1). If this color is the same than the <BackColor> the border will have squared corners.

DisplayMode

Type: Short

Set this property to AUTO, CENTRE, NORMAL or USE BITMAP SIZE. AUTO will stretch the bitmap. With the NORMAL mode you can specify margins and bitmap size. You can' t use the stretch mode with transparent bitmaps.

HandCursor

Type: Bool

If set to TRUE (-1), a hand cursor will be used at run time.

Help

Will display this Windows HLP file.

OnOffMode

Type: Bool

Set this property to TRUE (-1) if you wish this VBPicBtn can be used like an On-Off button (like a checkbox control). You must use this property with the special property <ButtonState>.

PictureDisabled

Type: Picture (Bitmap or Icon)

Specifies the picture to use when the button is disabled (and 'up' without any focus).

PictureDown

Type: Picture (Bitmap or Icon)

Specifies the picture to use when the button is 'down' (selected).

PictureFocus

Type: Picture (Bitmap or Icon)

Specifies the picture to use when the button has the focus (not selected).

PictureUp

Type: Picture (Bitmap or Icon)

Specifies the picture to use when the button is 'up'.

RadioMode

Type: Bool
To use this property you will set (in most cases) the <OnOffMode> to TRUE (-1). When this property is TRUE, the button will acts like a radio control but you must handle it with the <SetState> special property.

ShowBorder
Type: Bool
If set to TRUE (-1) a button border will be automatically displayed. If your picture already has a normal or a customised border, set this parameter to FALSE (0).

ShowFocus
Type: Bool
If set to TRUE (-1) a button focus rectangle will be automatically displayed. If your picture already has a normal or a customised focus rectangle, set this parameter to FALSE (0).

UseSystemColors
Type: Bool
If set to TRUE (-1), the button will check user's colors configuration and will use these instead of colors you have specified at design time.

UseTranspBitmap
Type: Bool
If set to TRUE (-1), the button will use the light fuchsia as the 'transparent' color and won' t display it. This property is not available when you use the stretch mode (AUTO) or when you use Icons.

vbxAbout
will display the VBX Studio 'About...' box.

XMargin
Type: Short
Specifies the picture horizontal margin in pixels in the 'normal' display mode.

YMargin
Type: Short
Specifies the picture vertical margin in pixels in the 'normal' display mode.

VBPicBtn Special Properties

ButtonState
Type: Bool
This read-only property is useful when you choose the <OnOffMode>. If set to TRUE (-1) the button is On (selected), otherwise the button is Off.

SetState
Type: Bool
This hidden property is used when you choose the <RadioMode> and <OnOffMode>. If set to TRUE (-1) the button will be On (selected), otherwise the button will be Off. Also, you will read the <ButtonState> property and you will write this property.

VBPicBtn Events

<None>

VBPicBtn Methods

Refresh
will repaint the VBPicBtn button.

VBPicBtn VC++ Extensions

CVBPicBtn is the class you must use with VBPicBtn controls.

Properties wrappers

CBool AlwaysRepaint()
CLong BackColor()
CInt BitmapHeight()
CInt BitmapWidth()
CBool ButtonState()
CLong CornersColor()
CHsz CtlName()
CInt DisplayMode()
CBool Enabled()
CBool HandCursor()
CLong Height()
CLong Left()
CBool OnOffMode()
CPicture PictureDisabled()
CPicture PictureDown()
CPicture PictureFocus()
CPicture PictureUp()
CBool RadioMode()
CBool SetState()
CBool ShowBorder()
CBool ShowFocus()
CBool TabStop()
CInt TabIndex()
CHsz Tag()
CLong Top()
CBool UseSystemColors()
CBool UseTranspBitmap()
CBool Visible()
CLong Width()
CInt XMargin()
CInt YMargin()

Events wrappers

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)

used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*

returns a CVBControl pointer for the current CVBPicBtn VBX.

VBPicBtn Tips

Drawing the picture...

You must give a picture for all <Picture...> properties. Also, if you use bitmaps or icons with different sizes you will probably set the <AlwaysRepaint> property to TRUE (-1).

Using the <RadioMode>...

Both VB sample and VC sample will give you some help to do a customised radio button with the VBPicBtn VBX. This can be used to choose colors or to create controls like small buttons in a toolbar. To do this, set <OnOffMode> and <RadioMode> to TRUE and then you must write your own handler for all your controls: When a VBPicBtn is clicked, check the <ButtonState>, if this one is TRUE (-1) unselect the other with the <SetState> property.

VBText

VBText is the VBX Studio static ('label') control.

Required file(s): *VBTEXT.VBX*
Control name: *VBText*
MFC Class name: *CVBText (Base class is CStatic)*

See also...

VBEdit

VBText Properties

Only non standard properties are described below in alphabetic order:

Frame3D

Type: Short

Specify the border mode for the text. This 3D border is not the same than other <Border3D> properties (this border is drawn on VBText and not outside of the control).

Style3D

Type: Short

Specify the 3D effect for the text .

Autosizing

Type: Bool

This property is not available with VB. With VC++ AppStudio, if set to TRUE (-1) the control will be sized with the text.

BackColor

Type: Short (COLOR40_...)

Specify the background color.

Help

Will display this Windows HLP file.

HorizAlignment

Type: Short

Specify the horizontal alignment for the text.

Multiline

Type: Bool

If set to TRUE (-1) the text can be multiline. With this mode, the vertical alignment will be set to 'top' in all cases.

vbxAbout

will display the VBX Studio 'About...' box.

VerticalAlignment

Type: Short

Specify the vertical alignment for the text. This property is not available with multiline text controls.

VBText Special Properties

<None>

VBText Events

<None>

VBText Methods

Refresh
will repaint the text control.

VBText VC++ Extensions

CVBText is the class you must use with VBText controls.

Properties wrappers

CInt Frame3D()
CInt Style3D()
CBool Autosizing()
CInt BackColor()
CHsz Caption()
CHsz CtlName()
CBool Enabled()
CBool FontBold()
CBool FontItalic()
CHsz FontName()
CFloat FontSize()
CBool FontUnderline()
CLong ForeColor()
CLong Height()
CInt HorizAlignment()
CLong Left()
CInt MousePointer()
CBool Multiline()
CBool TabStop()
CHsz Tag()
CLong Top()
CInt VerticalAlignment()
CBool Visible()
CLong Width()

Events wrappers

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBText VBX.

VBText Tips

Style properties

Some properties like <Sort>, <ListHeight>, <MultipleSel> and so on.. change the control style. That' s why you can' t change them dynamically with your application. Also, in some cases, under at design time you will have to reload your forms to reflect the changes.

Using String Studio...

If you build MFC applications, you will find a big advantage in using String Studio CStr class with the VBText control: This library can help you to format string, to use numbers and dates and to manage your control from a string!

VBAnim

VBAnim is the VBX Studio animated picture control.

Required file(s): *VBANIM.VBX*
Control name: *VBAnim*
MFC Class name: *CVBAnim (Base class is CStatic)*

See also...

VBPicBtn

VBAnim Properties

Only non standard properties are described below in alphabetic order:

Border3D

Type: Bool

Set this to TRUE (-1) if you wish the edit control has a 3D border at run time.

AlwaysRepaint

Type: Bool

If this property is set to TRUE (-1), a background rectangle will be displayed each time the picture change. This is required when the picture has a different size from <PictureUp> to <PictureDown>.

BackColor

Type: Short (COLOR40_...)

Specify the background color.

DisplayMode

Type: Short

Set this property to AUTO, CENTRE, NORMAL or USE BITMAP SIZE. AUTO will stretch the bitmap. With the NORMAL mode you can specify margins and bitmap size. You can' t use the stretch mode with transparent bitmaps.

Help

Will display this Windows HLP file.

Picture1 to Picture5

Type: Picture (Bitmap or Icon)

You can specify up to 5 different pictures for your VBAnim animated picture. The order these pictures will be displayed depends on the <ProcessMode> property. You don' t need to give a picture for all these 5 properties!

PictureHeight

Type: Short

Specifies the picture height in pixels in the 'normal' display mode.

PictureWidth

Type: Short

Specifies the picture width in pixels in the 'normal' display mode.

ProcessMode

Type: Short

You can choose CONTINUALLY (picture 1 to picture 5), RETURNS TO THE FIRST PICTURE (picture 1 , picture 2, picture 1 ...) and BLINK (same as CONTINUALLY but will refresh the control each time).

Speed

Type: Short

This property specifies the VBAnim timer speed: It can be different from VBAnim1 to VBAnim2.. Normal range is 5 to 100, the higher the <Speed> is, the slower the pictures will change (because <Speed> is the timer speed).

UseTransparentBitmap

Type: Bool

If set to TRUE (-1), the light fuchsia will be used as the 'transparent' color (it won' t be displayed). This mode is not available for stretched mode and Icons.

vbxAbout

will display the VBX Studio 'About...' box.

XMargin
Type: Short
Specifies the picture horizontal margin in pixels in the 'normal' display mode.

YMargin
Type: Short
Specifies the picture vertical margin in pixels in the 'normal' display mode.

VBAnim Special Properties

<None>

VBAnim Events

<None>

VBAnim Methods

Refresh
will repaint the control.

VBAnim VC++ Extensions

CVBAnim is the class you must use with VBAnim controls.

Properties wrappers

CBool Border3D()
CBool AlwaysRepaint()
CInt BackColor()
CHsz CtlName()
CInt DisplayMode()
CBool Enabled()
CLong Height()
CLong Left()
CInt MousePointer()
CPicture Picture1()
CPicture Picture2()
CPicture Picture3()
CPicture Picture4()
CPicture Picture5()
CInt PictureHeight()
CInt PictureWidth()
CInt ProcessMode()
CInt Speed()
CInt TabIndex()
CBool TabStop()
CHsz Tag()
CLong Top()
CBool UseTransparentBitmap()
CBool Visible()
CLong Width()
CInt XMargin()
CInt YMargin()

Events wrappers

EventKeyCode(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key code>.

EventKeyShift(LPVOID lp)
used for VBN_KEYUP and VBN_KEYDOWN and returns the <key shift>.

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBAnim VBX.

VBAnim Tips

Small is beautiful...
Try to make your VBAnim controls as little as possible to prevent them from flashing.

Timers are limited resources...
So, if you use many timers it' s a good idea to check if some timers resources are still available. To do this you can use SDK functions (SetTimer and KillTimer) to test it, then if there are no more timers, set the VBAnim <Visible> property to FALSE (0) for instance...

Using transparent bitmap(s)...
If you set <UseTransparentBitmap> to TRUE (-1), the light fuchsia color will be used as the transparent color, so this color won' t be displayed. This is not available when you use the stretch mode ('auto') and this has no effect for icons.

VB3D

VB3D is the VBX Studio automatic 3D border control.

Required file(s): *VB3D.VBX*
Control name: *VB3D*
MFC Class name: *CVB3D (Base class is CStatic)*

VB3D Properties

Only non standard properties are described below in alphabetic order:

Mode3D

Type: Short

This is the main VB3D property to select the 3D border mode: You can disable all VB3D 3D borders, use the built-in classes (for VBX Studio VBXs) and/or controls IDs (associated Windows numbers for all controls), Hwnd (Windows control handle you can use with most of VB controls) or classes (The name used by Windows to register controls style). Classes are like 'Thunder...' for VBX and, for 'normal' controls you need the SDK to know them ('Edit', 'ListBox'...). For instance, the VBListBox controls class is 'ThunderVBListBox', its Hwnd is VBListBox.Hwnd and its ID can be something like IDC_MYLIST when used with AppStudio.

Help

Will display this Windows HLP file.

UseSystemColors

Type: Bool

If set to TRUE, all VB3D 3D borders will use the user's color configuration choice instead of colors you define at design time. These colors are in the Button section.

vbxAbout

will display the VBX Studio 'About...' box.

VB3D Special Properties

<None>

VB3D use standard methods instead of special properties.

VB3D Events

<None>

VB3D Methods

AddItem

Parameters: <Item string> ,<index>

You will use this method to add a class name, a control ID or a control HWND for the 3D process. Also, this depends on the chosen <Mode3D>. Obviously you must translate all value into a string, for instance:

```
VB3D1.AddItem Str$(UnknownControl1.Hwnd), 0
```

You can put up to 30 different values (<index> 0 to 29) and you must give a value to the <index> parameter. If you choose 'Auto and class names' or 'class names' <Mode3D>, you will add class names like this:

```
VB3D1.AddItem "ThunderEdit", 0
```

If you choose 'Auto and IDs' or 'IDs' <Mode3D>, you will a control ID like this:

```
VB3D1.AddItem Str$( GetDlgCtrlID( VBEdit1.Hwnd) ), 0
```

And, if you choose 'Auto and HWNDs' or 'HWNDs' <Mode3D>, you will a control HWND like this:

```
VB3D1.AddItem Str$( VBEdit1.Hwnd ), 0
```

Of course, when you choose the 'automatic' <Mode3D> you don' t need to use this method if you use standard VBX Studio controls.

RemoveItem

Parameters: [<index>]

Will remove the specify 0 based <index> value or all values if you don' t give any <index>.

Refresh

will repaint all 3D borders.

VB3D VC++ Extensions

CVB3D is the class you must use with VB3D controls.

Properties wrappers

CHsz CtlName()
CLong Height()
CLong Left()
CInt Mode3D()
CBool TabStop()
CHsz Tag()
CLong Top()
CBool UseSystemColors()
CLong Width()

Events wrappers

<None>

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVB3D VBX.

VB3D Tips

Using VB3D...

It' s a bad idea to use the VB3D VBX and to set the other VBXs <Border3D> to TRUE. You must use VB3D when your dialog box is resizable. When you change VB3D datas or <Mode3D> at run time, you must refresh the main window (form). Also, 0 for IDs or Hwnd and "" for classes names disable the data. You can give up to 30 characters to the AddItem method when used with class names.

Using VB3D with VC++...

With VC++ you must manually call the VB3D <Refresh> method in your OnPaint (WM_PAINT message handler). If you don' t do this, VB3D won' t draw any 3D border. This is not required when used with VB.

System colors...

VB3D can use system colors instead of static colors. (<Border3D> properties that many other VBX Studio VBXs can use, doesn' t support this). These colors are the Button HighLight and Button Shadow colors.

VBLine

VBLine is the VBX Studio line control.

Required file(s): *VBLINE.VBX*
Control name: *VBLine*
MFC Class name: *CVBLine (Base class is CStatic)*

See also...

VBTab

VBLine Properties

Only non standard properties are described below in alphabetic order:

BackColor
Type: Short (COLOR40_..)
Specifies the background color. To use the user system colors instead of colors you choose at design time, set this property to 0. (Currently background system color is light grey).

Help
Will display this Windows HLP file.

LineColor
Type: Short (COLOR40_..)
Specifies the foreground (line) color. To use the user system colors instead of colors you choose at design time, set this property to 0.

LineStyle
Type: Short
Basically there are 2 kinds of lines: The first one is the 'normal' style, you can use <LineThickness> and <Style3D> and other properties. You can choose the line position too. The second one is the 'VBTab' mode (choice 4 to 7), you will use it to draw your tabbed dialog boxes. With this second kind of lines, other properties are disable and you will use special lines for your VBTabs controls. You can have a look at provided samples to get tips for these controls.

LineThickness
Type: Short
Specifies the line thickness in pixels.

Style3D
Type: Short
Specifies the line 3D border style.

vbxAbout
will display the VBX Studio 'About...' box.

VBLine Special Properties

<None>

VBLine Events

<None>

VBLine Methods

Refresh
will repaint the line.

VBLine VC++ Extensions

CVBLine is the class you must use with VBLine controls.

Properties wrappers

CInt BackColor()
CHsz CtlName()
CLong Height()
CLong Left()
CInt LineColor()
CInt LineStyle()
CInt LineThickness()
CInt MousePointer()
CInt Style3D()
CBool TabStop()
CHsz Tag()
CLong Top()
CBool Visible()
CLong Width()

Events wrappers

EventMouseBtn(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse button code (see MFC Tech notes for more informations).

EventMouseShift(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the shift key state (see MFC Tech notes for more informations).

EventMouseX(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

EventMouseY(LPVOID lp)
used for VBN_MOUSEUP and VBN_MOUSEDOWN and returns the mouse X position in pixels.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBLine VBX.

VBLine Tips

Using VBLine with VBTab

Basically when you use VBLine with the VBTab mode (a tabbed dialog box), you will draw at least 3 VBLine controls: The first one will be the left line, the second one, a rectangle and the last one a right line. Of course you must choose the 'VBTab' mode and you will reduce these lines as little as possible.

Drawing a rectangle for VBTab

The other way to use the VBLine control with a tabbed dialog box is to draw an unique VBTab rectangle: It' s quite more easy. This method has a big disadvantage: this rectangle will hide the Border3D for VBX Studio VBX, so if you choose to use the 3D borders you can' t use it.

VBDDrop

VBDDrop is the VBX Studio Winfile drag 'n drop control.

Required file(s): *VBDDROP.VBX*
Control name: *VBDDrop*
MFC Class name: *CVBDDrop (Base class is CStatic)*

VBDDrop Properties

Only non standard properties are described below in alphabetic order:

Help
Will display this Windows HLP file.

vbxAbout
will display the VBX Studio 'About...' box.

 *VBDDrop Special Properties*

DroppedFiles
Type: Short
This run-time read-only property specify the number of dropped files. This property is the same as the <FileDropped> 'Files' parameter.

Filename
Type: String (indexed)
This run-time read-only property specify the full name (with drive and directories) of specified file. To retrieve the name of the first dropped file you will read 'VBDDrop1.Filename(0)'.

VBDDrop Events

FileDropped (VBN_FILEDROPPED)

Parameters: Files as integer

The VBDDrop control send out this notification message each time a file or a group of files is dropped. The <Files> parameter is the number of files currently dropped (this is the same as <DroppedFiles> property).

VBDDrop Methods

Refresh
will repaint the VBDDrop area, so you will call this method when your dialog box is resized.

VBDDrop VC++ Extensions

CVBDDrop is the class you must use with VBDDrop controls.

Properties wrappers

CLong BackColor()
CHsz CtlName()
CInt DroppedFiles()
CHsz Filename()
CLong Height()
CLong Left()
CBool TabStop()
CLong Top()
CBool Visible()
CLong Width()

Events wrappers

EventDroppedFiles(LPVOID lp)
used for the VBN_FILESDROPPED event and returns <Files>.

Other

CVBControl GetVBX()*
returns a CVBControl pointer for the current CVBDDrop VBX.

VBDDrop Tips

How to use the VBDDrop VBX?

As other controls, click on the VBDDrop palette button and 'put' it on your form. Now, the only one property you maybe will change is the <BackColor> property. This color must be the same as your main form <BackColor> because VBDDrop will fill the entire dialog box with this color. Now, at run time, use the <FilesDropped> event and set the <Visible> property to FALSE if you wish to disable the VBX.

Window painting

VBDDrop handles Winfile drag'n drop for the dialog client zone (and not for its other controls). So, when you use a resizable dialog box, you must call the VBDDrop <Refresh> method to compute its new client zone. Some controls, specially controls without any background brush, aren' t compatible with VBDDrop because this VBX will hide these.

VBNote

VBNote is the VBX Studio design time note control.

Required file(s): VBNOTE.VBX
Control name: VBNote
MFC Class name: CVBNote (Base class is CStatic)

There is no special informations for this control: Just click with the mouse and type your text. If you use this control with your dialog boxes, you must (like all VBX) distribute the VBX with your applications. This control is specially useful if you develop Windows applications in a team.

See also...

VBText

Tips

How to install the VBXs...
VC++ Issues

VBList
VBFList
VBCombo
VBFCombo
VBEdit
VBSpin
VBTab
VBGauge
VBText
VBPicBtn
VBAnim
VB3D
VBDDrop
VBLine

How to install the VBX Studio VBXs...

VBX Installation (Design time)

If you use Visual Basic, choose "Add Files..." and select the VBX you wish to install. If you prefer VC++ AppStudio, use "Install controls..", double click on the VBXs and close the dialog box with the "Ok" button (do not use the "Close" button). If you use AppStudio and you want to use the VBX Studio DDX routines, have a look at DDX.TXT text file and, if you want to use classes wrappers, put VBXSTD12.H and ASVBDO12.H into your INCLUDE path and put the '#include <VBXSTD12.H>' at the end of your STDAFX.H precompiled header.

VBX Installation (Run time)

When you use the VBX Studio VBXs you must not forget to distribute the VBX(s) with your software. Also, these VBXs must be put into your customer WINDOWS\SYSTEM directory. To do this you can find a big help in Setup Studio 2.2 (Another 'Studio' shareware, see Author for more informations). With Setup Studio you can even use VBX Studio VBXs with your Setup program! To install the VBX(s), the best way is too check the VBX last modification date: If you don' t find the VBX you must install it, if the VBX is found but is older you don' t install it (but you maybe need to display a warning), and if your VBX is newer, you backup the old one and you install the new one.