

## **VBApi Help**

### **Api Reference Contents**

[Functions](#)

[Messages](#)

[Properties](#)

[Events](#)

[Data Structures](#)

[Types](#)

[Flags](#)

## VBApi Types

### *Type*

```
Bool=WordBool;  
Enum=Byte;  
Npnt=Word;  
Lpnt=Pointer;  
LPStr=PChar;  
Hsz= LPHandle;  
PHSZ=^HSZ;  
HLSTR =LPHandle;  
HCtl=LPHandle;  
HFormFile=PHandle;  
ERR=Word;  
HPIC=Word;  
Float=Single;
```

## VBApi Messages

VBAPI.TPW defines a number of messages that originate with Visual Basic. These messages are listed below, according to category:

### Messages

### Description

VB_M_CANCELMODE	Reset internal state.
VB_M_CHECKPROPERTY	Check property value.
VB_M_COPY	Copy data from Clipboard.
VB_M_CREATED	Creation and loading of a control completed.
VB_M_DATA_AVAILABLE [3.0]	Notification of data available from the data control.
VB_M_DATA_REQUEST	Notification of data requested from the data control.
VB_M_DATA_GET [3.0]	Get data from the data control.
VB_M_DATA_METHOD [3.0]	Notification of data requested from the data control.
VB_M_DATA_SET [3.0]	Send data to the data control.
VB_M_DRAGDROP	Item dropped on control.
VB_M_DRAGOVER	Item dragged over control.
VB_M_FIREREVENT	Implement delayed event.
VB_M_GETDEFSIZE [2.0]	Get default size.
VB_M_GETPALETTE [2.0]	Request for palette.
VB_M_GETPROPERTY	Get property value.
VB_M_GETPROPERTYHSZ	Get string to display in Properties bar.
VB_M_HELP	Process help request.
VB_M_HITTEST [2.0]	Return mouse hit status.
VB_M_INITIALIZE	HCTL allocated.
VB_M_INITPROPPUP	Determine how value is set in Properties bar.
VB_M_ISMNEMONIC [2.0]	Mnemonic entered.
VB_M_LINKENUMFORMATS [2.0]	Enumerate DDE data formats.
VB_M_LINKGETDATA [2.0]	Receive DDE data.
VB_M_LINKGETITEMNAME [2.0]	Get DDE item name.
VB_M_LINKSETDATA [2.0]	Send DDE data.
VB_M_LOADED	All controls on form loaded, or control dynamically loaded.
VB_M_LOADPROPERTY	Load property from disk.
VB_M_LOADTEXTPROPERTY [2.0]	Load properties as text.
VB_M_METHOD	One of control's methods used in a statement.
VB_M_MNEMONIC	Respond to access key.
VB_M_PAINT [2.0]	Repaint graphical control.
VB_M_PAINTMULTISEL [2.0]	Respond to multiple selection.
VB_M_PAINTOUTLINE [2.0]	Respond to moving.
VB_M_PALETTECHANGED [2.0]	Request to select a palette.
VB_M_PASTE	Paste data into Clipboard.
VB_M_QPASTEOK	Determine if Paste or Paste Link should proceed.
VB_M_SAVEPROPERTY	Save property to disk.
VB_M_SAVETEXTPROPERTY [2.0]	Save properties as text.
VB_M_SELECTED [2.0]	Control selected at design time.
VB_M_SETPROPERTY	Set property value.
VB_M_WANTSPECIALKEY [2.0]	Virtual key entered.

### VB Notification messages:

#### VB\_N\_{Message}:

VB_N_CHARTOITEM [2.0]	WM_CHARTOITEM
VB_N_COMMAND	WM_COMMAND
VB_N_COMPAREITEM	WM_COMPAREITEM
VB_N_CTLCOLOR	WM_CTLCOLOR
VB_N_DELETEITEM	WM_DELETEITEM
VB_N_DRAWITEM	WM_DRAWITEM
VB_N_HSCROLL	WM_HSCROLL
VB_N_MEASUREITEM	WM_MEASUREITEM
VB_N_PARENTNOTIFY	WM_PARENTNOTIFY
VB_N_VKEYTOITEM [2.0]	WM_VKEYTOITEM
VB_N_VSCROLL	WM_VSCROLL

## VBApi Data Structures

### ***TModel Record***

Type

```
LPMODEL=^TMODEL;
TModel=Record
    usVersion:Word;           { VB version used by control }
    fl:LongInt;               { Bitfield structure }
    ctlproc:TFARPROC;         { The control proc. }
    fsClassStyle:Word;         { Window class style }
    flWndStyle:LongInt;        { Default window style }
    cbCtlExtra:Word;           { # bytes alloc'd for HCTL structure }
    idBmpPalette:Word;         { BITMAP id for tool palette }
    DefCtlName: NPnt;          { Default control name prefix }
    ClassName:NPnt;            { Visual Basic class name }
    ParentClassName:NPnt;      { Parent window class if subclassed }
    proplist:NPPROPLIST;       { Property list }
    eventlist:NPEVENTLIST;     { Event list }
    nDefProp:Byte;             { Index of default property }
    nDefEvent:Byte;            { Index of default event }
    nValueProp:Byte;           { Index of control value property }
    usCtlVersion:Word;         { Identifies the current version of }
                                { the custom control. The values }
                                { 1 and 2 are reserved for custom }
                                { controls created with VB 1.0 and }
                                { VB 2.0. }
end;
```

### ***TPROPINFO Record***

Type

```
PPROPINFO=NPnt;
TPROPINFO=record
    npszName:NPnt;
    fl:LongInt;               { PF_ flags }
    offsetData:Byte;          { Offset into static structure }
    infoData:Byte;            { 0 or _INFO value for bitfield }
    dataDefault:LongInt;      { 0 or _INFO value for bitfield }
    npszEnumList:NPnt;        { For type = DT_ENUM, this is }
                                { a near ptr to a string containing }
                                { all the values to be displayed }
                                { in the popup enumeration listbox. }
                                { Each value is an sz, with an }
                                { empty sz indicated the end of list. }
    enumMax:Byte;             { Maximum legal value for enum.}
end;
```

### ***TEVENTINFO Record***

Type

```
TEVENTINFO=record
    npszName:NPnt;
    cParms:Word;
    cwParms:Word;             { # words of parameters }
    npParmTypes:NPnt;         { list of parameter types }
    npszParmProf:NPnt;        { event parameter profile string }
    fl:LongInt;               { EF_ flags }
end;
```

### ***TPIC Record***

Type

```
TPicData=record
    case Byte of
        PICType_BitMap:(BitMap:HBitMap;hpal:HPalette);
        PICType_MetaFile:(Meta:THandle;xExt,yExt:Integer);
        PICType_Icon:(Icon:HIcon);
    end;
```

```

PPIC=^TPIC;
TPIC=record
    picType:Byte;
    picData:TPicData;
    picReserved:array [0..3] of Byte;
end;

```

The use of each field is described below:

Data path to field	Description
picType	Enumerated type specifying the format of picture. Can be set to any of the following: PICTYPE_NONE PICTYPE_BITMAP PICTYPE_METAFILE PICTYPE_ICON
picData.bmp.hbitmap	Handle to a bitmap.
picData.bmp.hpal [2.0]	Handle to a palette.
picData.wmf.hmeta	Handle to a metafile.
picData.wmf.xExt	Width of the area to contain the picture.
picData.wmf.yExt	Height of the area to contain the picture.
picData.icon.hicon	Handle to an icon.
picData.icon.hcursor	Handle to a cursor; can be NULL if you don't wish to specify a drag cursor for the icon.

## ***TDATASTRUCT Record (PropertyArrays)***

```

Type
    TIndex=record
        datatype:Word; { Type of nth index (Currently always DT_SHORT) }
        data:LongInt; { Value of nth index }
    end;
    TDataStruct=record
        data:LongInt; { Data for Get/Set }
        cindex:Word; { Number of indices }
        index:array [0..0] of TIndex;
    end;

```

## ***TDRAGINFO Record***

```

Type
    TDRAGINFO=record
        Control:HCTL;
        pt:TPOINT;
        state:Word; { Enter, Over, Exit; only used for VBM_DRAGOVER }
    end;

```

## ***TVBLinkData Record***

Data xfer structure for VBM\_LINKGETDATA or VBM\_LINKSETDATA

```

Type
    LPLinkData=^TVBLinkData;
    TVBLinkData=record
        wReserved:Word;
        cb:LongInt;
        HData:THandle;
        dwReserved:LongInt;
    end;

```

## ***TModelInfo Record***

```

Type
    LPModelInfo=^TModelInfo;
    TModelInfo=record
        usVersion:Word;
        lpmodel:^LPMODEL;
    end;

```

## ***TDataAccess Record***

DA structure used for conversations with the data control

Type

```
LPDataAccess=^TDataAccess;
TDataAccess=record
    usVersion:Word;           { VB version of structure filled in }
                                { when structure is created }
    sAction:Integer;         { on VBM_DATA_GET/SET }
                                { specifies what to get/set on }
                                { VBM_DATA_AVAILABLE/REQUEST }
                                { tells why }
    hctlData:Hctl;           { The data control providing data }
    hctlBound:Hctl;          { The bound control receiving data }
    hszDataField:HSZ;        { The name of the field to get value of }
    sDataFieldIndex:Integer; { the field index used when FieldName is null }
    usDataType:Word;         { the property datatype to convert data to }
    hlstrBookMark:HLStr;     { used when getting multirow data }
    fs:Word;                 { Bitfield structure }
    lData:LongInt;           { The data }
    ulChunkOffset:LongInt;   { The offset to start at for GetChunk }
    ulChunkNumBytes:LongInt; { The number of bytes for GetChunk/SetChunk }
end;
```

## VBApi Functions

The VBAPI.TPW file provides access to a subset of the functions residing in VB.EXE and VBRUN100.DLL. These functions include many of the same functions used internally to support standard controls. The functions are listed below alphabetically:

<b>Function</b>	<b>Description</b>
VBAAllocPic	Allocate HPIC structure.
<b>function VBAAllocPic(PntPic:PPIC):HPIC;</b>	
VBAAllocPicEx [2.0]	Extended VBAAllocPic.
<b>function VBAAllocPicEx(PntPic:PPIC;usVersion:Word):HPIC;</b>	
VBAArrayBounds [2.0]	Returns upper and lower bounds.
<b>function VBAArrayBounds(VBArray:HAD;index:Integer):LongInt;</b>	
VBAArrayElement [2.0]	Returns a pointer to the value of a Basic array element.
<b>function VBAArrayElement(VBArray:HAD;clIndex:Integer;var lpi:Integer):Pointer;</b>	
VBAArrayElemSize [2.0]	Returns the size of a single Basic array element.
<b>function VBAArrayElemSize(VBArray:HAD):Word;</b>	
VBAArrayFirstElem [2.0]	Returns a pointer to the first element of a Basic array.
<b>function VBAArrayFirstElem(VBArray:HAD):Pointer;</b>	
VBAArrayIndexCount [2.0]	Returns the number of indexes for a Basic array.
<b>function VBAArrayIndexCount(VBArray:HAD):Integer;</b>	
VBCbSaveFPState [2.0]	Saves the current floating-point state.
<b>function VBCbSaveFPState(pb:Pointer;cb:Word):Word;</b>	
VBCClientToScreen [2.0]	Converts the given point to screen coordinates.
<b>procedure VBCClientToScreen(Control:HCTl;var Point:Point);</b>	
VBCoerceVariant [2.0]	Copies the value of a variant to a memory location
<b>function VBCoerceVariant(Variant:PVariant;vtype:Integer;lpData:Pointer):Word;</b>	
VBCreateHlstr	Allocate language string.
<b>function VBCreateHlstr(pb:Pointer;cbLen:Word):HLStr;</b>	
VBCreateHsz	Allocate HSZ string.
<b>function VBCreateHsz(Control:HCTl;Str:LPStr):HSZ;</b>	
VBCreateTempHlstr [2.0]	Creates temp Basic string.
<b>function VBCreateTempHlstr(pb:Pointer;cbLen:Word):HLStr;</b>	
VBDDefControlProc	Default message processing.
<b>function VBDDefControlProc(Control:HCTl;Wnd:HWND;Msg:Word;WParam:Word;LParam:LongInt):LongInt;</b>	
VBDerefControl	Get pointer to programmer-defined structure.
<b>function VBDerefControl(Control:HCTl):LongInt;</b>	
VBDerefHlstr	Get pointer to string data.
<b>function VBDerefHlstr(HStr:HLStr):LPStr;</b>	
VBDerefHlstrLen [2.0]	
<b>function VBDerefHlstrLen(HStr:HLStr;var pCbLen:Word):PChar;</b>	
VBDerefHsz	Get pointer to string data.
<b>function VBDerefHsz(HSZStr:HSZ):LPStr;</b>	
VBDerefZeroTermHlstr [2.0]	Returns the pointer to the string data.
<b>function VBDerefZeroTermHlstr(HStr:HLStr):PChar;</b>	
VBDestroyHlstr	Remove language string.
<b>procedure VBDestroyHlstr(HStr:HLStr);</b>	
VBDestroyHsz	Remove string.
<b>function VBDestroyHsz(HSZStr:HSZ):HSZ;</b>	
VBDialogBoxParam	Create a pop-up dialog box.
<b>function VBDialogBoxParam(Instance:THandle;TemplateName:LPStr;DialogFunc:TFARPROC;lp:LongInt):Integer;</b>	
VBDirtyForm [2.0]	Indicate property change.
<b>procedure VBDirtyForm(Control:HCTl);</b>	
VBFireEvent	Execute event procedure.
<b>function VBFireEvent(Control:HCTl;IdEvent:Word;LPParams:Pointer):ERR;</b>	
VBFormat [2.0]	converts a string to a format.
<b>function VBFormat(vtype:Integer;lpData:Pointer;lpszFmt:PChar;pb:Pointer;cb:Word):Integer;</b>	
VBFreePic	Decrement reference count and delete HPIC if count is zero.
<b>procedure VBFreePic(Pic:HPIC);</b>	
VBGetAppTitle	Get application title.
<b>procedure VBGetAppTitle(Str:LPStr;cbMax:Word);</b>	
VBGetCapture [2.0]	Retrieves a handle to the control that has mouse capture.
<b>function VBGetCapture:HCTl;</b>	
VBCGetClientRect [2.0]	Copies the dimensions of the client area to a TRect.
<b>procedure VBCGetClientRect(Control:HCTl;var Rect:TRect);</b>	
VBGetControl [2.0]	Searches for a control.

```

function VBGetControl(Control:HCTl;gc:WORD):HCTl;
VBGetControlHwnd      Get handle to window.
function VBGetControlHwnd(Control:HCTl):Hwnd;
VBGetControlModel     Get model structure.
function VBGetControlModel(Control:HCTl):LPModel;
VBGetControlProperty  Get property value.
function VBGetControlProperty(Control:HCTl;IdProp:Word;PData:Pointer):ERR;
VBGetControlName      Get property name.
function VBGetControlName(Control:HCTl;lpszName:LPStr):LPStr;
VBGetControlRect [2.0] Copies dimensions of bounding rect of the control.
procedure VBGetControlRect(Control:HCTl;var Rect:TRect);
VBGetDataSourceControl [3.0] Retrieve data control currently bound to.
function VBGetDataSourceControl(Control:HCTl;bIsRegistered:Bool):HCTl;
VBGetHInstance        Get handle to current instance of Visual Basic.
function VBGetHInstance:THandle;
VBGetHlstr [2.0]      Copies the string data to the dest buffer.
function VBGetHlstr(HStr:HLStr;pb:Pointer;cbLen:Word):Word;
VBGetHlstrLen         Get string length.
function VBGetHlstrLen(HStr:HLStr):Word;
VBGetHwndControl      Get handle to control.
function VBGetHwndControl(Wnd:HWND):HCTl;
VBGetMode             Determine whether in design, run, or break mode.
function VBGetMode:Word;
VBGetPic              Dereference picture data.
function VBGetPic(Pic:HPic;PntPic:PPic):HPic;
VBGetPicEx [2.0]      Extended VBGetPic function.
function VBGetPicEx(Pic:HPic;PntPic:PPic;usVersion:Word):HPic;
VBGetRectInContainer [2.0]
procedure VBGetRectInContainer(Control:HCTl;var Rect:TRect);
VBGetVariantType [2.0] Returns the Variant data type.
function VBGetVariantType(Variant:PVariant):Integer;
VBGetVariantValue [2.0] Gets the value of the given variant.
function VBGetVariantValue(Variant:PVariant;Value:PValue):Integer;
VBGetVersion [2.0]    Returns the version of the host environment.
function VBGetVersion:Word;
VBInvalidRect [2.0]   Adds a rectangle to the controls update region.
procedure VBInvalidRect(Control:HCTl;var Rect:TRect;fEraseBkGnd:Boolean);
VBIsControlEnabled [2.0] Determines if control is Enabled.
function VBIsControlEnabled(Control:HCTl):BOOL;
VBIsControlVisible [2.0] Determines if control is Visible.
function VBIsControlVisible(Control:HCTl):BOOL;
VBLinkMakeltemName [2.0] Create item name that contains control array information.
procedure VBLinkMakeltemName(Control:HCTl;lpszBuf:PChar);
VBLinkPostAdvise [2.0] Sends notification to the DDE client that the link data has been changed.
function VBLinkPostAdvise(Control:HCTl):Word;
VBLockHsz             Get pointer to data and prevent string from moving.
function VBLockHsz(HSZStr:HSZ):LPStr;
VBMoveControl [2.0]   Changes position and dimensions of a control.
procedure VBMoveControl(Control:HCTl;var Rect:TRect;fRepaint:BOOL);
VBPaletteChanged [2.0] Notifies Visual Basic that the palette has changed.
procedure VBPaletteChanged(Control:HCTl);
VBPasteLinkOk [2.0]
function VBPasteLinkOk(var phTriplet:THandle;Control:HCTl):BOOL;
VBPicFromCF           Get picture from Clipboard.
function VBPicFromCF(PntHPic:Pointer;HData:THandle;WFormat:Word):ERR;
VBReadBasicFile       Read data file.
function VBReadBasicFile(UsFileNo:Word;pb:Pointer;cb:Word):ERR;
VBReadFormFile        Read property value from disk during a load.
function VBReadFormFile(FormFile:HFormFile;pb:Pointer;cb:Word):ERR;
VBRecreateControlHwnd Destroy and recreate window, to enable new window styles.
function VBRecreateControlHwnd(Control:HCTl):ERR;
VBRefPic              Increment reference count.
function VBRefPic(Pic:HPic):HPic;
VBRegisterModel       Register a control class.
function VBRegisterModel(HMod:THandle;var Model:TModel):Boolean;
VBRelSeekBasicFile    Move position in data file a relative distance.

```



```

function VBRelSeekBasicFile(usFileNo:Word;offset:LongInt):LongInt;
VBRelSeekFormFile
function VBRelSeekFormFile(FormFile:HFormFile;OffSet:LongInt):LongInt;
VBResizeHlstr [2.0]      Reallocates the size of a given string.
function VBResizeHlstr(HStr:HLStr;newCbLen:Word):Word;
VBRestoreFPState [2.0]   Restores the floating point state.
procedure VBRestoreFPState(pb:Pointer);
VBRuntimeError [2.0]     Generates a runtime error.
procedure VBRuntimeError(err:Word);
VBScreenToClient [2.0]   Converts points.
procedure VBScreenToClient(Control:HCtl;var Point:TPoint);
VBSeekBasicFile [3.0]    Move position in data file.
function VBSeekBasicFile(usFileNo:Word;offset:LongInt):LongInt;
VBSeekFormFile           Moves the current position within a form file.
function VBSeekFormFile(FormFile:HFormFile;OffSet:LongInt):LongInt;
VBSendControlMsg          Send message to a control.
function VBSendControlMsg(Control:HCtl;Msg,WParam:Word;LParam:LongInt):LongInt;
VBSetControlFlags [2.0]  Set and return the characteristics of a control.
function VBSetControlFlags(Control:HCtl;mask:LongInt;value:LongInt):LongInt;
VBSetControlProperty      Set property value.
function VBSetControlProperty(Control:HCtl;IdProp:Word;Data:LongInt):Err;
VBSetErrorMessage        Set text of next error message.
function VBSetErrorMessage(error:ERR;Str:LPStr):ERR;
VBSetHlstr                Assign new string data.
function VBSetHlstr(PHStr:Pointer;pb:Pointer;cbLen:Word):ERR;
VBSetVariantValue [2.0]  Sets a variant to a given value.
function VBSetVariantValue(Variant:PVariant;vtype:Integer;lpData:Pointer):Word;
VBSuperControlProc        Call parent class directly.
function VBSuperControlProc(Control:HCtl;Msg,WParam:Word;LParam:LongInt):LongInt;
VBTranslateColor [2.0]   Converts a VB color to an RGB color.
function VBTranslateColor(Control:HCtl;Color:LongInt):LongInt;
VBUnlockHsz               Unlock string address.
function VBUnlockHsz(HSZStr:HSZ):LPStr;
VBUpdateControl [2.0]    Updates a control.
procedure VBUpdateControl(Control:HCtl);
VBWriteBasicFile          Write to data file.
function VBWriteBasicFile(UsFileNo:Word;pb:Pointer;cb:Word):ERR;
VBWriteFormFile           Write property value to disk during a save.
function VBWriteFormFile(FormFile:HFormFile;pb:Pointer;cb:Word):ERR;
VBXPixelsToTwips          Convert X units to twips.
function VBYPixelsToTwips(Pixels:Integer):LongInt;
VBXTwipsToPixels          Convert X units to pixels.
function VBXTwipsToPixels(Twips:LongInt):Integer;
VBYPixelsToTwips          Convert Y units to twips.
function VBXPixelsToTwips(Pixels:Integer):LongInt;
VBYPixelsToTwips          Convert Y units to pixels.
function VBYPixelsToTwips(Twips:LongInt):Integer;
VBZOrder [2.0]           Alters the drawing order of the control.
procedure VBZOrder(Control:HCtl;zorder:WORD);

```

## VBApi Properties

### Standard property constant

PPROPINFO\_STD\_ALIGN [2.0]  
PPROPINFO\_STD\_BACKCOLOR

PPROPINFO\_STD\_BORDERSTYLEON  
PPROPINFO\_STD\_BORDERSTYLEOFF  
PPROPINFO\_STD\_CAPTION

PPROPINFO\_STD\_CLIPCONTROLS [2.0]  
PPROPINFO\_STD\_CTLNAME

PPROPINFO\_STD\_DATACHANGED [3.0]

PPROPINFO\_STD\_DATAFIELD [3.0]  
PPROPINFO\_STD\_DATACHANGED [3.0]

PPROPINFO\_STD\_DRAGICON  
PPROPINFO\_STD\_DRAGMODE  
PPROPINFO\_STD\_ENABLED  
PPROPINFO\_STD\_FONTNAME

PPROPINFO\_STD\_FONTBOLD  
PPROPINFO\_STD\_FONTITALIC  
PPROPINFO\_STD\_FONTSIZE  
PPROPINFO\_STD\_FONTSTRIKE  
PPROPINFO\_STD\_FONTUNDER  
PPROPINFO\_STD\_FORECOLOR

property.

PPROPINFO\_STD\_HEIGHT  
PPROPINFO\_STD\_HELPCONTEXTID [2.0]  
PPROPINFO\_STD\_HWND  
PPROPINFO\_STD\_IMEMODE [2.0]  
PPROPINFO\_STD\_INDEX

PPROPINFO\_STD\_LAST [2.0]  
PPROPINFO\_STD\_LEFT

PPROPINFO\_STD\_LEFTNORUN [2.0]  
PPROPINFO\_STD\_LINKITEM [2.0]  
PPROPINFO\_STD\_LINKMODE [2.0]  
PPROPINFO\_STD\_LINKTIMEOUT [2.0]  
PPROPINFO\_STD\_LINKTOPIC [2.0]  
PPROPINFO\_STD\_MOUSEPOINTER  
PPROPINFO\_STD\_NAME [2.0]  
PPROPINFO\_STD\_NONE [2.0]  
PPROPINFO\_STD\_PARENT  
PPROPINFO\_STD\_TABINDEX

PPROPINFO\_STD\_TABSTOP

PPROPINFO\_STD\_TAG  
PPROPINFO\_STD\_TEXT

PPROPINFO\_STD\_TOP

### Comments

Forces control to align with form.

BackColor. You must send a  
WM\_CTLCOLOR message to get a  
brush that contains the background color.

BorderStyle: set to True by default.

BorderStyle: set to False by default.

Caption. Requires that you respond to  
WM\_SETTEXT, WM\_GETTEXT, and  
WM\_GETTEXTLENGTH messages.

Sets or clears the windows child flag.

CtlName. This property is required, and  
should be placed first in the list..

Indicates the data in the bound control  
has changed by something other than  
getting the current record .

Allows the control to bind to a database field.

Allows the control to bind to a specific data  
control.

DragIcon

DragMode

Enabled

FontName. Requires that you respond to  
WM\_SETFONT and WM\_GETFONT  
messages. Responding to these messages  
provides the support necessary for all the  
font properties. All font properties are saved  
and loaded together with FontName.

FontBold

FontItalic

FontSize

FontStrike

FontUnder

ForeColor. You must also send a  
WM\_CTLCOLOR message to use this

Height

User defined help context id.

The controls runtime window handle.

See VB control Ref.

Index. This property is required, and should be  
placed second in the list.

Indicates the max value of a standard property.

Left, Top, Width, and Height are all saved and  
loaded together with Left.

Use this prop instead of left for an invisible control.

DDE property.

DDE property.

DDE property.

DDE property.

Mousepointer.

Replaces CTLNAME.

Place holder for removed properties.

Parent

TabIndex. Should be supported if fFocusOk  
or fMnemonic flag is set.

TabStop. Should be supported if fFocusOk  
or fMnemonic flag is set.

Tag

Text. This property is the same as Caption,  
except for the property name.

Top

PPROPINFO\_STD\_TOPNORUN [2.0]  
PPROPINFO\_STD\_WIDTH  
PPROPINFO\_STD\_VISIBLE

Use this prop instead of top for an invisible control.  
Width  
Visible

## VBApi Events

### Standard event constant

PEVENTINFO\_STD\_CLICK

message is

over the

standard

PEVENTINFO\_STD\_DBLCLICK

received.

PEVENTINFO\_STD\_DRAGDROP

PEVENTINFO\_STD\_DRAGOVER

PEVENTINFO\_STD\_GOTFOCUS

PEVENTINFO\_STD\_KEYDOWN

PEVENTINFO\_STD\_KEYPRESS

PEVENTINFO\_STD\_KEYUP

PEVENTINFO\_STD\_LAST [2.0]

PEVENTINFO\_STD\_LINKCLOSE [2.0]

PEVENTINFO\_STD\_LINKERROR [2.0]

PEVENTINFO\_STD\_LINKNOTIFY [2.0]

PEVENTINFO\_STD\_LINKOPEN [2.0]

PEVENTINFO\_STD\_LOSTFOCUS

PEVENTINFO\_STD\_MOUSEDOWN

PEVENTINFO\_STD\_MOUSEMOVE

PEVENTINFO\_STD\_MOUSEUP

PEVENTINFO\_STD\_NONE [2.0]

### Comments

Click. Fired when the control has captured the mouse, a WM\_LBUTTONDOWN

received, and the button is released

control. This event is fired after the

MouseDown and MouseUp events.

DbClick. Similar to Click, but fired when a WM\_LBUTTONDBLCLK message is

DragDrop. Fired after a VBM\_DRAGDROP message is received.

DragOver. Fired after a VBM\_DRAGOVER message is received.

GotFous. After receiving a WM\_GOTFOCUS message, the default message processor posts a VBM\_FIREEVENT message. This is a mechanism that results in delayed firing of the event, which is processed after other pending messages are.

KeyDown. Fired when either a WM\_KEYDOWN or a WM\_SYSKEYDOWN message is received.

KeyPress. Fired when a WM\_CHAR message is received.

KeyUp. Fired when either a WM\_KEYUP or a WM\_SYSKEYUP message is received

Place holder for the last standard event.

DDE Event.

DDE Event.

DDE Event.

DDE Event.

LostFocus. Fire when a WM\_LOSTFOCUS message is received. The same delayed-processing mechanism is used as for GotFocus.

MouseDown. Fired if any BUTTONDOWN (right, left, or middle button) message is received. The mouse is captured as a result of this event.

MouseMove. Fired when a WM\_MOUSEMOVE message is received.

MouseUp. Fired if any BUTTONUP (right, left, or middle button) message is received. The mouse capture is released as a result of this event.

Place holder for removed event.

## VBApi Flags

This chapter provides a summary of each data structure used in custom control code. Understanding these structures can be useful in writing general DLL routines as well as in writing custom controls. Features that are specific to a version level of Visual Basic are denoted by the version number inside brackets. For example, version 3.0 are denoted by [3.0].

### Model Flags

Flag Value	Tells Visual Basic To:
MODEL_fArrows	Send to the control keyboard messages for arrow keys. If this flag is off, arrow keys are used to move between controls.
MODEL_fFocusOk	Enable the control to get the focus at run time.
MODEL_fMnemonic	Respond to the controls mnemonic key (access key) by moving the focus to the control and sending a VBM_MNEMONIC message. The control must support the standard Caption property to have a mnemonic key.
MODEL_fChildrenOk	Enable the Visual Basic developer to draw other controls on top of this control at design time. This lets the control function as a container of other controls, as picture boxes and frames can.
MODEL_fLoadMsg	Enable the control to get VBM_CREATED and VBM_LOADED messages.
MODEL_fInitMsg	Enable the control to get VBM_INITIALIZE messages.
MODEL_fDesInteract	Enable the control to get right-mouse-button messages at design time: WM_RBUTTONDOWN, WM_RBUTTONDBLCLK, WM_RBUTTONUP. By processing these messages at design time, the control procedure can provide special mechanisms for manipulating the control and its properties.
MODEL_fGraphical [2.0]	Indicate control is a graphical control.
MODEL_fInvisAtRun [2.0]	Specify that the control is invisible at run time. Visual Basic automatically handles the representation of the control on the form at design time it uses the same icon that appears in the Toolbox

### Property Data Type Flags

Only one of the following flags can be selected for each property. The selected flag is then logically OR'ed with flags listed in the next section.

Flag Value	Resulting data type of:
DT_BOOL	Boolean. This is a short integer, though it can be packed into as little as 1 bit. All incoming and outgoing nonzero values are converted to -1 during a set or get operation.
DT_COLOR	A long integer that holds a color value. This value is an RGB value, or (if the most significant bit is set) the lowest 3 bytes contain a system-color number defined in WINDOWS.H. Visual Basic automatically validates numbers input for validity as color values. The value is displayed in the Properties window using

DT_ENUM	<p>hexadecimal radix.</p> <p>A short integer holding an enumerated value (0 255). Selection of this data type (as opposed to <b>DT_SHORT</b>) enables the npszEnumList and enumMax fields, described in the previous section. The data can be packed into a bit field from 1 to 4 bits wide.</p>
DT_HLSTR [2.0]	<p>String. The string data should be stored as an <b>HLSTR</b> type. <b>DT_HLSTR</b> property values are string descriptors representing strings of bytes that can contain embedded NULL characters. This is the principle advantage over <b>DT_HSZ</b> strings which are terminated by a NULL character. <b>HLSTR</b> strings that you allocate must be freed in response to <b>WM_NCDESTROY</b>. Refer to Chapter 9 of the Control Development Guide, "Handling Strings and Fonts," for more details on <b>HLSTR</b> strings.</p>
DT_HSZ	<p>String. The string data should be stored as an <b>HSZ</b> type. If the <b>PF_fSetData</b> flag is set, allocation and freeing of the <b>HSZ</b> string are handled by Visual Basic. <b>HSZ</b> strings that you allocate must be freed in response to <b>WM_NCDESTROY</b>. Refer to Chapter 9 of the Control Development Guide, "Handling Strings and Fonts," for more details on <b>HSZ</b> strings.</p>
DT_LONG	32-bit signed integer.
DT_OBJECT	<p>Points to an iDispatch interface. Refer to Technical Note 1: Support for <b>DT_OBJECT</b> Properties, TN001.TXT, in the CDK directory.</p>
DT_PICTURE	<p>Picture structure. The picture should be stored as an <b>HPIC</b> handle. If the <b>PF_fSetData</b> flag is set, Visual Basic allocates <b>HPIC</b> handles as needed. The <b>HPIC</b> must be freed in response to <b>WM_NCDESTROY</b>.</p>
DT_REAL	4-byte real.
DT_SHORT	16-bit signed integer. The data can be packed into a bit field from 1 to 4 bits wide.
DT_XPOS	Long integer expressing an X coordinate. The data for a get or set operation is expressed in terms of the container's scale. The data gets converted, if necessary, so that the custom control code uses units expressed in twips, with the origin (0,0) at the control's upper-left corner. Similar conversions are performed for the remaining types.
DT_XSIZE	Long integer expressing an X size in twips. Origin does not affect size conversions.
DT_YPOS	Long integer expressing a Y

DT\_YSIZE

coordinate in twips.  
Long integer expressing a Y size  
in twips.

## Property Flags

You can use any number of the following flags in the fl field of the TPROPINFO structure.

**Important:** When the PF\_fPropArray flag is used, the following flags must also be used: PF\_fSetMsg, PF\_fGetMsg, and PF\_fNoShow. The PF\_fSaveData flag must not be used.

### Flag Value

### Description:

PF\_fGetData

Indicates that Visual Basic gets the property value by directly copying data from the programmer-defined structure. The offsetData and infoData fields of the TPROPINFO table specify exactly where the data is located within the structure.

PF\_fGetMsg

Causes Visual Basic to send a VBM\_GETPROPERTY message when the property value is requested. Either this flag or the PF\_fGetData flag should be used.

PF\_fSetCheck

Causes Visual Basic to send a VBM\_CHECKPROPERTY message before it sets the value of the property. This gives the custom control code a chance to check the validity of the property value. If the control procedure returns a nonzero error code in response to this message, Visual Basic reports an error to the user and will not proceed with setting the property to the new value.

PF\_fSetData

Indicates that Visual Basic sets the value of the property by placing data directly in the programmer-defined structure.

PF\_fSetMsg

Causes Visual Basic to send a VBM\_SETPROPERTY message when setting of the property is attempted. Either this flag or the PF\_fSetData flag should be used, unless the property is read-only. If both are used, then the data is transferred before the message is sent.

PF_fSaveData	When the form is being saved to a file, Visual Basic saves the property value by getting its value and then writing the data to disk directly. Similarly, Visual Basic loads the property by reading its value directly from the disk and then setting it.
PF_fSaveMsg	Causes Visual Basic to send a <b>VBM_SAVEPROPERTY</b> message to the control whenever the form is being saved to a file, and a <b>VBM_LOADPROPERTY</b> message when the form is loaded from a file. These messages pass a handle to file location. It is then the responsibility of the control procedure to read or write from disk (using <b>VBReadFormFile</b> and <b>VBWriteFormFile</b> ).
PF_fNoShow	Prevents the property from appearing in the Properties bar.
PF_fNoRuntimeW	Indicates that the property is read-only at run time.
PF_fGetHszMsg	Causes Visual Basic to send a <b>VBM_GETPROPERTYHSZ</b> message to the control whenever the property value is to be displayed in the Properties bar. If this flag is not used, then Visual Basic simply gets the property value and displays it in the Settings box.
PF_fUpdateOnEdit	Causes the property value to be set every time a character is typed in the Settings box of the Properties bar. If this flag is not used, the property is not set until change is committed.
PF_fEditable	Enables the Visual Basic developer to edit text in the Settings box directly, even if a list box or pop-up window is used. If neither is used, this flag has no effect.
PF_fPreHwnd	Causes the property to be loaded before the controls window structure is created. A property should be pre-hwnd if it affects a window style. The actual



	<p>setting of the window style (that is, the style altered by the pre-hwnd property) should be made in response to a <b>WM_NCCREATE</b> message.</p>
<b>PF_fDefVal</b>	<p>Causes Visual Basic to avoid saving and loading the property to disk when the value is equal to the dataDefault value in the <b>TPROPINFO</b> structure. This flag affects only loading and saving, not default values of newly created controls. During saving, the property value is not written to disk if equal to the dataDefault value. Then, during loading, if the property was not saved to disk, it is set to the dataDefault value. (But <b>PF_fNoInitDef</b>, described next, alters loading behavior.)</p>
<b>PF_fNoInitDef</b>	<p>Prevents Visual Basic from setting the property to the dataDefault value during loading of the control (though you can still initialize the property by responding to messages). If <b>PF_fDefVal</b> is not used, <b>PF_fNoInitDef</b> has no effect.</p>
<b>PF_fPropArray</b>	<p>Specifies that the property is an array. When this flag is used, you must use a <b>TDATASTRUCT</b> structure to implement all getting and setting of the property. Any type supported as a simple (scalar) property can also be used as the base type of a property array. Important: when this flag is used, the following flags must also be used: <b>PF_fSetMsg</b>, <b>PF_fGetMsg</b>, and <b>PF_fNoShow</b>. The <b>PF_fSaveData</b> flag must not be used.</p>
<b>PF_fLoadMsgOnly[2.0]</b>	<p>Causes Visual Basic to send <b>VBM_LOADPROPERTY</b> messages when the form is loaded from a</p>

file. These messages pass a handle to file location. It is then the responsibility of the control procedure to read from disk (using the VBRadFormFile function).

This property is similar to `PF_fSaveMsg`, except that a `VBM_SAVEPROPERTY` messages are not generated. This allows custom controls to be backward compatible by loading properties in previous versions that are no longer supported in newer ones.

`PF_fLoadDataOnly[2.0]` Causes Visual Basic to load the property from a form file, but prevents the property from being written back out to the form file. This property is similar to `PF_fSaveData`, except that properties are not written out to the form file. This allows custom controls to be compatible with Visual Basic version 1.0.

`PF_fNoMultiSelect[2.0]` Specifies that any given custom property NOT be displayed in the Properties window when that control is part of a selected group. Currently, no properties of type `DT_PICTURE` are included in a multiple selection. However, if `PF_fNoMultiSelect` is not present, a future version of Visual Basic may include a given property of type `DT_PICTURE` in a multiple selection. If you must guarantee that a property is never placed in a multiple selection (even if it is `DT_PICTURE`), then `PF_fNoMultiSelect` must be set.

`PF_fNoRuntimeR[2.0]` Indicates that the property is write-only at run time. Any attempt to read the property generates a runtime error. When

combined with `PF_fNoRuntimeW`, it indicates that the property is not available at all at run time (a design-time-only property), and references to it are flagged as Visual Basic compilation errors.



