VB Voice, Win32 Edition

Index

Control Summary

 VBVFrame The container control for all the other VBVoice controls

Conference An optional control that provides conferencing abilities between channels

 Count Maintains an internal counter, which can be reset or incremented. The call is transferred to a new control depending on the results of a comparison between the counter value and a preset limit. Useful for simple loops.

 DataChange Changes data in a database record, in one or more fields. The database and record are selected by a previous DataFind control.

 DataFind Searches for the first or the next database field depending on whether the control is entered via the main input or the "Next" input. A wide range of data match conditions is available.

 DataSwitch Reads a database record from a previous DataFind control, stores the result in the property 'Result', and routes according to the value obtained.

 Delay Implements a wait period. The caller can be put on hold, or a short voice clip can be played during the wait.

 Dial Dials some digits either during a call or to start a call. Can perform call supervision (answer, no answer, busy etc.) and has built-in support for PBX call transfer.

 GetDigits Plays a greeting, and waits for digits from the caller. Configurable exit conditions using wild characters, numeric characters and ranges. Built-in invalid digit and no digit handling.

 InConn Used for graphical connection of one control to another on a different form. Only InConn and OutConn controls can be connected across forms.

 IniSwitch Searches for a setting in a Windows initialization file. Branches to another control depending on the value found.

 Language An optional control that allows VBVoice to support multiple languages in a system

 LineGroup This control will wait for incoming ringing on a channel, or it can start a call automatically or under control of your code. After a preset number of rings, the channel is taken off hook and control is passed to the next control.

 Onhook . Plays a greeting, and hangs up the phone.

 OutConn Used for connection of one control to another when a graphical connection is not appropriate. Included mainly for backwards compatibility

 PlayGreeting Plays a greeting and exit. Provides capability for fast forward, rewind, and volume control

 PlayMsgs A high level control that accesses a database table containing a list of messages. Configurable options after play. Supports new, old and deleted messages. Updates database with new message status.

 ReceiveFax This control is part of the VBFax option. Although included in the VBV32.OCX, it is inactive unless the VBFax option is purchased

 Record Plays a greeting, records a file, and adds a new record to a database. At the end of the recording, allows the caller to choose options to re-record, delete, append to end of recording, or to save the message.

 SendFax This control is part of the VBFax option. Although included in the VBV32.OCX, it

is inactive unless the VBFax option is purchased.

 TimeSwitch Transfers the call to another control according to time of day and day of week.

 User This control allows you to use VB code to create your own control. You have access to all the high-level VBVoice functions and the low-level hardware-specific driver functions.

Methods List

Properties List

Events

- Condition
- DelayStarted
- Disconnect
- Enter
- EnterB
- Exit
- PhraseError
- PlayRequest
- PreDisc
- Ring
- VoiceError

VBVFrame

Description

Initial Setup Properties:

- DateFormat
- LanguageID
- LogDirectory
- ScriptSize
- TimeFormat
- VoiceDirectory

Runtime Properties:

- Transfer Object
- CurrentDirectory

Methods:

- SetLineStatGrid (GridCtl as Object)
- ShowLineStatus (Show as BOOL)
- ShowLogWindow (Channel as Integer, show as BOOL)
- StartSystem (ShowStatus as Integer)
- StopSystem (KillChannels as Integer)

Events:

- ShutDown()

VBVFrame property pages:

- General
- PBX Settings
- Transfer Properties
- Test Mode setup

This control primarily serves as a container for the other VBVoice controls, providing an environment where VBVoice can draw the control connections. It also provides project-wide properties and methods that control system operation. The VBVFrame also has a Transfer object property which can be used to transfer data from Visual Basic code into the underlying VBVoice controls, for example to provide data to the Greetings and other controls requiring variables.

StartSystem Method

Applies to VBVFrame

StartSystem (ShowStatus as BOOL)
StartSystem performs the same function as the StartSystem option in the run mode context menu. It checks all the controls on all loaded forms, loads the voice card driver and starts voice operations. When executed from the context menu, the line status window is automatically displayed. The method provides a parameter to toggle display of the status window. Once the system is started, you can stop voice operations with Stop System.
If an error occurs while starting the system, the method will generate a trappable error. See System Startup Errors for possible error values.

StopSystem Method

Applies to VBVFrame

StopSystem (KillChannels as BOOL) As Integer
Once the system is started, you can stop voice operations with the StopSystem method. If KillChannels is False, VBVoice will wait for all lines to clear before terminating. If KillChannels is True, VBVoice will attempt to halt all the lines immediately, terminating any calls currently on the line. The method returns True if successfully shutdown, False if system is not yet halted (there are still channels active)
If VBVoice is able to shut down all channels immediately (i.e. the lines are already idle), then a ShutDown event will occur while processing the method, the LineStatus window will be closed, and the method will return True.
If some lines are active when the method is called, the following events will occur:

- The Line Status window will appear and will mark all idle lines as 'waiting for termination'.
- The ShutDown event will occur when all lines are idle, and the system stopped.
- The LineStatus window will be closed.

ShowLineStatus Method

Applies to VBVFrame

ShowLineStatus (Show as BOOL)
Shows or hides the linestatus window. Note you can also use the LineStatus control which can be embedded in your own controls. If using the control, you will not be able to show the system provided window. See LineStatus window.

ShowLogwindow Method

Applies to VBVFrame

ShowLogwindow (Channel as Integer, show as BOOL)
Shows or hides the log windows. There is a log window for each channel.

SetLineStatGrid Method

SetLineStatGrid (GridCtl as Object)
You can embed the VBVoice status window into your own forms with this method. Add a MS Grid control to your form, and ensure it has at least 5 columns and at least 1 more row than there are channels. Assign the grid to VBVoice with the code

VBVFrame1.SetLineStatGrid Grid1

to have VBVoice update the grid with the linestatus. When finished, use

VBVFrame1.SetLineStatGrid vbNothing

to reset use of the grid by VBVoice before the grid is destroyed. In the CustomStatus example, this is done in the VBVFrame ShutDown event.
You will also have to resize the columns of the grid appropriately, and provide the column and row header text.
See example in CustomStatus.FRM

ShutDown Method

Applies to VBVFrame

ShutDown ( )
This event is fired when a shutdown has been completed. It only occurs in the first frame to be loaded in the application. See StopSystem above.

CurrentDirectory

Applies to VBVFrame

This is an read-only, per-channel property that returns the current voice directory for a channel. If language controls are not in use, it will always return the value of VoiceDirectory. If language controls are in use, it will return the directory as set by the language control in use on the specified channel.

VoiceDirectory

Applies to VBVFrame

The voice directory can be set on a per-project basis. On new projects, the default directory is used as set in the VBV32.INI file.
If there is more than one frame in the project, each frame will return the same value for this property, and setting this property will affect all frames.

LogDirectory

Applies to VBVFrame

The log directory can be set on a per-project basis. On new projects, the default directory (as set in the VBV32.INI file) is used.
If there is more than one frame in the project, each frame will return the same value for this property, and setting this property will affect all frames.

Transfer Object

Applies to VBVFrame

The transfer object provides a configurable list of string properties that can be used to pass data into VBVoice controls at runtime. For example, if the properties MyProp1 and MyProp2 are defined at design time using the Transfer property page, then these properties will be available in the control list window to be copied into property page fields that accept control properties. These properties can be written to by Visual Basic code using this syntax:

VBVFrame1.Transfer.MyProp1(channel) = "12345"

If a phrase is configured in a greeting such as
SayNumber %VBVFrame1.MyProp1%
then the system will say the number 12345.
Each frame control can have a different set of properties. Controls can access frame transfer properties regardless of which frame they are contained by.

See also Transfer property page

Language properties (VBVFrame)

Applies to VBVFrame

- TimeFormat
- DateOrder
- LanguageID

All these properties define the characteristics of a language. One language can be specified in the system for the Standard edition. The professional edition includes the language control which provides simultaneous multi-lingual capability, including different rules for saying numbers, dates and times.
If there is more than one frame in the project, each frame will return the same value for these properties, and setting these properties will affect all frames.

TimeFormat (vbvTimeFormatConstants)
TimeFormat is a project wide setting that specifies whether times are said in 12 hour or 24 hour time, e.g. “Eleven twenty two P.M.” or “Twenty three twenty two”.
Possible values:
Time12
Time24

DateOrder (vbvDateOrderConstants)
DateOrder is a project wide setting that defines the order in which the component parts of dates are spoken. Note that the components of the date to be spoken (day, month, year, weekday), and the format (numeric month, month name, etc.) are defined by each individual phrase.
Possible values:
ddmmyy
yymmdd
yyddmm
mmddyy

LanguageID (vbvLanguageConstants)
LanguageID is a project wide setting that specifies the concatenation rules for saying numbers. VBVoice has built-in rules for several languages. Support for dynamic language setting, multiple simultaneous languages, and other languages can be added using the Language control.

Possible values:
vbvEnglish
vbvFrench
vbvItalian

ScriptSize

This property will normally be used in conjunction with the language control to manage vap files in multiple languages. It restricts the number of characters that VBVoice will use when searching for phrases in vap files. See the Language control.

VBVFrame General property page

Specifies the default directories that VBVoice will use for phrase files, wave files and log files. These can be changed at run time using the LogDirectory and VoiceDirectory properties. If either directory is set to a specific directory different from the directory set in VBV32.INI, this directory will be used in the resulting .EXE. If the directory is left unchanged, then the .EXE will use the setting in VBV32.INI even if they are different than those used at design time. In other words, the directories are only saved in the .EXE if they are different from the defaults set in VBV32.INI. At runtime, if these properties are not set, the VBV32.INI defaults are used.

Default Greeting format:
Specifies the default play format used by phrases. This setting is used by all greetings that do not override the default by setting a specific format in the Greeting property page. Each greetings can set the play format for phrases contained within it.

See also Language Properties

See also VBVFrame

VBVFrame Transfer Properties property page

Use this property page to create custom properties for your project. See the Transfer property description.

See also VBVFrame

VBVFrame PBX Settings property page

Used for PBX integration. This page sets up dial strings to be used to instruct the PBX or telephone system to perform operations like transferring a call, or putting a call on hold. The defaults supplied will work with most telephone systems that have these capabilities. If you experience difficulties, check the PBX documentation, or experiment by dialing from an analog phone to find out which strings work for your system.

Start transfer
Complete blind transfer
Reconnect

These strings are used by Dial control when performing transfer operations.
Put on hold
Reconnect from on-hold
These strings are used by the Delay control with the Put-On-Hold option.

Flash Time
Pause Time
Duration (in milliseconds) of the time for a pause character (comma) and the hookflash character (exclamation mark).

See also VBVFrame

VBVFrame Test Mode property page

This property page is used to setup which device VBVoice uses for Test mode and for recording and playing prompts, and also how it uses the voice card in Test Mode, if selected. If your voice card is connected to a PBX or the external telephone system, you should configure it so you can dial in when starting a test, or alternatively have VBVoice call you on another line. Also use this property page to select which voice card channel to use while testing.

Test Mode options

Use Voice card

Use Sound card
These commands allow you to choose which device to use for listening to greetings, and for testing the system. If voice card is selected, the Voicecard options are enabled as described below. This option will be disabled if no voice card driver was detected.

Start Test Mode options

Go offhook and dial
This option is useful if you are using a Telephone Line Simulator or other device that gives you instant connection to the voice card without having to dial or if you want VBVoice to call you through the telephone system to start up in test mode. When you press the start button in test mode, VBVoice will take the channel specified offhook. If a dial string is specified, it will dial the number and wait for a connection. If the number is blank, it will assume you are using a direct connection using a line simulator or equivalent, and will establish a connection immediately.

Wait for ring
If you are using a PBX phone line and you want to call into the voice card to start testing, select this option. When you start a test, VBVoice will wait for ringing on the channel. When ringing is detected, it will go offhook.

Stay offhook after test
If you had to dial in to make the connection, this option allows you to keep the connection alive, removing the need to dial in again (or have VBVoice call you) for the next test or greeting.

If you lose the connection...
If you do hang up during a call, you can make the voice card go on and offhook using the offhook / onhook buttons in the Test window. You can then dial in again.

See also VBVFrame

Conference control

Properties Events
Default Property: None

Conference property page

Conference custom control

Conferencing Example

Description

The Conference control is used for merging individual voice lines together into a conference. This control is available as an option with VBVoice. Callers in a conference have the ability to verbally communicate to all other callers in the conference. Each instance of this control represents one unique conference. The control can:

- provide independent detection of dtmf digits for each channel (depending on hardware support). Digits detected by the Conference control can be passed to a GetDigits control for processing.
- remove a caller from a conference using Visual Basic code.
- send a notification message to one caller in a conference.
- send a warning message to a caller that their maximum time limit is about to be reached, and for a caller to be automatically removed from the conference when the time expires.
- add callers to a conference in listen-only and talk-listen modes.

Entering the conference

When a call enters this control, the caller is played a greeting, and the caller is then connected to the conference. The caller can be connected either in listen-only mode or in full talk and listen mode, depending on which entry point of the control is used.

Exiting the conference

The caller can exit out of the conference by either pressing one of the pre-defined exit digits, hanging up, reaching the maximum time allowance, or being removed from the conference by Visual Basic code. A maximum in-conference time limit can be set globally for each caller, or on a per-channel basis by setting a custom property for a specific conferenced channel.

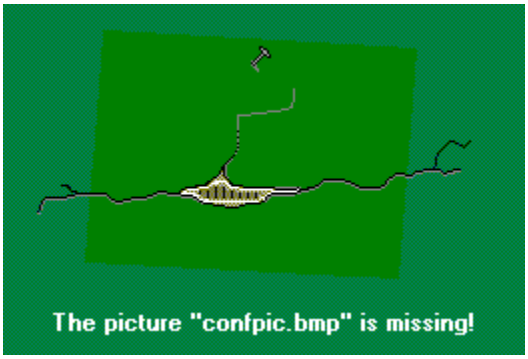
Hardware Requirements:

The Conference control requires a voice card that supports conferencing. Different levels of conferencing are supported by the various voice card manufacturers as follows:

- **Dialogic** - The Dialogic AMX card supports conferencing to a limited extent. It does not support DTMF detection on a per-channel basis, so that if a digit is pressed by any caller in a conference, it will be detected on all channels (and all callers will hear it). In addition, the AMX card cannot support large conferences without introducing loss of volume on all channels, and does not support listen-only mode.
- **Pika** - The Inline-4-GT and the V-12 cards support all the capability provided by this control.

Conference custom control

See also Conference



Input Nodes

The Conference control supports two inputs:
The main input plays the entry greeting, and then connects the caller into the conference in full talk-listen mode.
The second input (Listen) also plays the entry greeting, and then connects the caller into the conference in listen-only mode. Listen-only mode is available only on the Pika Inline GT and V12 cards, and the NewVoice cards. If a given voice card does not support this feature, then the Listen input node will function in an identical manner to the greeting input.
For both inputs, if the maximum number of lines has been reached, then the greeting will not be played and the call will exit the control out of the MaxLines output without being conferenced.
If a DTMF digit entered during a conference is to be serviced by a subsequent GetDigits control, then the caller's input alternatives should be played in the conference control greeting.

Output Nodes

The Conference control supports three distinct outputs. The output nodes are defined as follows:

Done
The caller will exit via the Done output node upon the occurrence of a standard termination event. This includes receiving an anticipated DTMF digit, or the caller being forced off the line. This output must always be connected to another control.

Max Lines
This node is used when a caller cannot enter a conference because the maximum number of conferenced lines has already been reached. The introduction greeting is not played in this case. If the maximum lines option is set to a positive value, then this output node must be connected.

Max Time
When the maximum time limit has been exceeded on a specific channel, the caller is transferred to the control that is connected to this output node. If a maximum time limit has been set, then this output node must be connected.
Note that the file "timewarn.vox" should reside in the directory that VBVoice is installed to. This file contains a simple beep, which can be used to send a time remaining warning. This file is user configurable.

LinesActive

Applies to Conference

The LinesActive property defines the total number of lines currently participating in a specific conference. This is a read only property.

Example

```
totalLines=Conference1.LinesActive
```

This line of code sets a Visual Basic variable (totalLines) to be equal to the number of callers currently conferenced on the Conference1 control.

ForceExit

Applies to Conference

Set this property to a channel number to force the caller on that channel out of the conference. If the channel specified is invalid, or is not currently in this conference, a runtime error (Err_BadChannel) occurs.
Otherwise, the user is removed from the conference. This property is write-only.

Example

Conference1.ForceExit=4.

This line of Visual Basic code forces the caller on line 4 to exit the Conference1 control.

NotifyVoxFile

Applies to Conference

This property is a array of strings, and is write-only. It is used to play a voice file to a conferenced caller, independently of the other conferees. The targeted conferee is temporarily removed from the conference while the file is played.
The property should be set to the file to be played. If the channel specified does not exist, or is not currently in conference, a runtime error (Err_CallNotHere) occurs. A runtime error (Err_BadNotify) occurs if the voice file is not found.

Example

Conference2.NotifyVoxFile(7)="CONFNTFY.VOX"

This code plays the voice file "confnty.vox" to the caller on channel 7. The file must be in the VBVoice directory, or a full or relative path name must be provided.

MaxTime

Applies to Conference

This property is used to set the maximum time allowable for a specific caller in a conference. It will override any time set in the property page at design time. See Setting Maximum Time below.

Setting Maximum Time

The time in conference for each conferee is checked every minute. If the maximum time field in the property page is non-zero, each conferee will be allowed the specified number of minutes before being forced out of the conference. The amount of time for each caller can also be set individually using the MaxTime property.

Example

Conference1.MaxTime(3) = 5

This line of code sets the maximum time on channel 3 to 5 minutes. Once five minutes has elapsed, the caller on line 3 exits the conference via the MaxTime output and is routed to the control connected to that output.

Conference Property page

See also Conference

Time Limit

This field is used to set the maximum time that a caller can spend in a conference. This field defaults to none (no time limit) and will accept a value from 1 to 255 minutes. This field applies to all conferenced lines. This value can be overridden on a channel by channel basis by the MaxTime(channel) property. When the time limit has been reached, the caller is removed from the conference and the call exits via the Max Time output node.

Max Lines in Conference

This field sets the maximum number of lines that can be concurrently conferenced. Legitimate input values are from 0 to the number of channels in the system. Setting this value to 0 disallows anyone from entering a conference. If a caller cannot access a conference because the conference already contains the maximum allotted number of conferees, the introduction greeting is not played and the caller is immediately transferred to the control connected to the Max Lines output node.

Exit Conference On

This field defines which digits will cause a caller to exit the conference. Valid input values include: all 12 standard DTMF phone touch-tone inputs, none, or any. If none is specified, the caller will not exit a conference due to a DTMF event. The only way a caller can exit a conference in this mode is to hang up. or to be forced off the conference by a timeout or by Visual Basic code.
If a user exits due to a DTMF event, then the DTMF digit is automatically passed on to the next control if it is a GetDigits control. The GetDigits control then suppresses the normal greeting and routes the call based on the received digit.

Send time remaining warning

If this field is checked, and a maximum time limit has been defined, then a warning voice file will be played to the caller at *TimeToMaxTime* minutes from the maximum time. *TimeToMaxTime* is defined in the VBVOICE.INI file under the [Conference] section, and defaults to one minute. The voice file played will be the file "TIMEWARN.VOX", found in the VBVOICE directory. The file supplied contains a user defined warning beep. This warning is sent to a conferee only once. The default for this field is enabled (checked).

Initialization Settings

The VBVOICE.INI file contains two optional entries in the [Conference] section.

MuxInterrupt
This field is used by the AMX card to specify the hardware interrupt set on the card. It should be set to the hardware interrupt level being used by the AMX card. The default is 0 (no hardware interrupt).
TimeToMaxTime
Specifies the time in minutes that a caller is allowed to stay in a conference after having received a Maxtime warning notification. The default is 1 minute.
This field is used if a maximum time limit has been set on a given channel, and the "Send time remaining warning" option is turned on. If this field value is greater than or equal to the Max Time value, then it will be disregarded.

Conferencing Example

See also Conference

This example is found in the CONF subdirectory. It sets up two independent conferences.
In order to run this example, you must have a Pika Inline GT, Pika V12 or NewVoice NV800/128 card.

Controls used:

- phone controls, one for each of the eight incoming lines.
- two distinct Conferences controls.
- A GetDigits control that handles the DTMF event from both conferences.
- An OnHook control.

When a caller rings any of the 8 channels, the LineGroup control will answer the call and the caller will hear the greeting defined in the greeting setup for the Conference1 control. After the greeting, the caller will be connected to all of the callers already in that conference.
Line 6 is treated slightly differently, in that it will enter the Conference1 control through the Listen input. This caller will be able to listen to the other conferees, but will not be able to participate.

Conference1 control definition

The implications of these definitions are:

- there is no maximum time
- only 4 conferees are allowed at any one time
- if a caller presses any digit, they will exit the conference

In this example only four lines can be in the first conference. Additional callers will automatically be transferred to the second conference since the Max Line output node is connected to the second conference control.

Conference2 control definition

This conference allows up to 8 conferees (which is the maximum lines supported in this example), a maximum time limit has been set at 10 minutes, and a time remaining warning is to be sent. This message is to be sent to each user once they have spent 9 minutes logged into this second conference (assuming that the TimeToMaxTime entry in VBVOICE.INI is not changed from the default of one minute).

Once the conferee's time limit has expired, they are automatically removed from the conference and the call is disconnected by the OnHook control, which is connected to the Max Time output.

A conferee can exit either of the two conferences by pressing any touch-tone key. In this example, the keys that are significant are:

- # - this key will disconnect the call
- 0 - this key will send the caller to the first conference
- 1 - this key will send the caller to the second conference

In this example, any invalid key will be handled in GetDigits as an invalid key input. The system will remain in GetDigits until a legitimate input is recognized, or the maximum number of retries has been exceeded.

These keys are defined in the GetDigits control, which processes the digit from conferees who exit from either of the two conference controls by pressing a digit.

Conference2 control definition

In addition to setting the expected digits as described above, the "Clear digits on entry" field has been cleared. This must be unchecked in order for the GetDigits control to process digits pressed by a conferee while in the conference. If this field is checked, then any digits pressed while in conference will be discarded, and the control will wait for a new digit from the caller after playing the entry greeting.

Properties

- LinesActive
- ForceExit
- NotifyVoxFile
- MaxTime

Events

- Disconnect
- Enter
- EnterB
- Exit
- PhraseError
- PlayRequest
- VoiceError

Runtime Properties

Counter (channel as integer)
This property contains the current value of the counter for each channel.
Example:

mycount% = Counter1.Counter(channel)

GotoNode

Count Initial Setup Properties

IMaxCount
The maximum value for Counter property until the call exits from the Limit output.

Delta
The amount (positive and negative) by which the counter value is changed whenever the call enters via the second input

MaxCount
This property contains the limit value for each channel and can be set for each channel individually.
Example:

mymaxcount% = Counter1.MaxCount

StartAt
The initial value of the counter when a call enters via the main control input

Count Events

- Enter
- Exit
- VoiceError

Count control

Runtime Properties Events Initial Setup Properties Methods.

Default Property: Count

Description:

This control maintains an internal counter which can be reset and incremented by call flow. It is useful for simple loops similar to the Visual Basic *for .. to .. step ..* construct.
When a call enters the Count control via the main input, an internal counter is reset to the start-from value and the call is passed directly to the control connected to the top output node. When a call enters via the second input, the Counter value is tested against the limit value. If Counter equals or has gone past the limit, the call is passed to node 2 (Limit), otherwise the Counter is incremented and the call is routed to node 1 (OK).
For example a limit of 5 means that on the fifth pass through the control, the call will be routed to the "Limit" node.

The 'Limit' node is treated in the same way as the 'Invalid' node in the GetDigits control. That is, if the node is not connected, the invalid digit termination procedure is invoked. See Error Handling

Methods

TakeCall

TakeCall

This method performs the same function as GotoControl but is more efficient and less error-prone. To move a call to a new control using the TakeCall method, use this syntax:

Example

Go to the User1 control from the GetDigits1 control:

```
GetDigits1_Enter(...)
User1.TakeCall(channel)
```

The destination control (User1) does not need to be connected to the current control, and does not need to be on the same form. The transfer takes place as soon as the event code is completed. The LineGroup control does not support the TakeCall method.

Conditions Object

The Conditions object is a list of Condition objects. Each condition object contains a condition name and a value, and represents one node on the control. The use of the name and value items are dependent on the control, but in general the name is the text printed on the control node, and the value is used for data matching. The Conditions object should be initialized using the property pages.
Conditions Object
This object represents a list of conditions used by several of the VBVoice controls. This object is normally created by the property pages, and there will not normally be a need for programmatic access.

Conditions Object Properties

NumConditions sets or returns the number of conditions in this object
Name (Array of String) sets or gets the name of one condition in the string. The name is normally the text printed in the output node corresponding to that condition.
Data (Array of String) sets or gets the data of one condition in the string. The use of the data string is control-specific.
UseVB (Array of BOOL) equivalent to the Use VB Condition flag in the condition property page.

DataChange control

Methods TakeCall

Events: Enter

Exit

VoiceError

Initial Setup Properties: AddRecord

Conditions Object

DataFind String

Runtime Properties:

GotoNode

Default Property:None

DataChange property page

Description:

Changes one or more fields in a database record. The database and record to be changed are selected by a previous DataFind control. The field values of the record in the database selected by the DataFind control will be changed to the new values. If Add New Record is set, a new record is added and the fields in that record updated instead. The data to be inserted can contain control names.

DataChange Property Page

See Also DataChange

Change data in this DataFind control (DataFind property)

This field shows the DataFind control that will be used to locate the record to be changed. The DataFind control specifies which database to use, how to open the database and the fields that can be updated for each record.

DataFind String

This property is the name of the DataFind control to be used as the source of data for comparison.

Change current record (AddRecord property)

Add new record

This selection boxes specify whether to change fields in the current record, or to add the new data to fields in a new record.

Field name / New data List (Conditions property)

This box contains a list of field names in the selected database, and the new values that will be assigned to them. Use the Edit, Delete and New buttons to modify this list. The available fields in this list are the fields selected for access by the DataFind control selected by this DataChange control.

Show available fields

Adds all available fields into the field list box. The available fields are set by the DataFind control selected in the Change data... dropdown list. When using a data control or an ODBC database, the control will show all available fields. When creating recordsets in code, only fields specified in the DataFind database property page will be available. To add other fields in the database, go to the DataFind setup and use the New, Edit and Delete buttons to modify the field list.

Reordering conditions

Conditions are matched in top to bottom order. In circumstances where more than one condition may match, you can order the conditions by dragging and dropping the items in the condition list

Edit command
Use the Edit command, or double-click on an entry to show the Edit Conditions dialog
Delete command
Deletes an entry
New command
Use this command to create a new entry.

Runtime Properties

- Fields Object
- GotoNode
- RecordCount

Initial Setup Properties

AlphaMatch Integer

DataSource

DBType

Filter

FindData String

GetRecordCount BOOL

MatchAllChars Integer

MatchType Integer

MatchField String

MatchData String

ODBCConnect String

ODBCTable

ReadOnly

RecordsetType Integer

UseGlobalBookmark BOOL

Filter

The Filter statement to be applied to an ODBC database. This statement can contain references to other control properties and thus allows a separate recordset to be created for each call.

Fields Object Array

This property contains an array of strings, one for each field in the database. Each field object provides access to the value of one of the fields in the database. This example reads the value of the DialNumber field in a database and sets the NumToDial property in the Dial1 control, prefixed with an ‘S’.

Example 1: using the Fields object

```
Dial1.NumToDial(channel) = “S” + DataFind1.Fields.DialNumber(channel)
```

FindData

Used to track down specific data.

GetRecordCount

This field can be checked if you wish to have the DataFind control count the number of records as soon as a call arrives in the control. This may be useful if you wish to announce the number of items in a database before allowing the caller to choose. This may have a severe performance impact on large recordsets. Record counting terminates automatically if the count exceeds 500.

MatchAllChars

This will match the entire text field against the database contents. In this case the database field and the contents of the selected field must be exactly the same.

MatchType

If this button is checked, DataFind will select the next record on each entry. The data matching fields will not be visible when this option is checked.

MatchField, MatchData

If this button is checked, DataFind will search the database for the next record that matches the specifications in the *matches this data* text field and Match type specifications.
The text you enter may contain control names. If using a Jet database, you will have to enter the fields manually since the control cannot find out which database will be used at runtime. When using bound data controls or ODBC databases, a drop-down list of fields will be provided.

ODBCConnect

The connect string used to open the ODBC database.

ODBCTable

The table to be opened when using an ODBC database. The table name is used in a SQL statement generated by the control which opens the table and accesses the fields specified by the Fields object (the list of fields set by the Change Fields.. button in the property page).

ReadOnly

Used in ODBC databases to indicate that the database should be opened in ReadOnly mode. This can provide an improvement in performance and should be used if no database updates are required.

Recordset

See RecordsetType

UseGlobalBookmark

If set, the DataFind control will use the same recordset position pointer for all channels.

DataFind control

Runtime Properties Events DataFind Topics
Initial Setup Properties Methods

Default Property: Count

Property pages

Database (using Data control)

Database (using recordset from code))

Database (ODBC)

Matching

Description:

See also
Using Databases
Choosing the database type

This control can select a record from a database for input validation and for subsequent data selection and update operations by DataSwitch, DataChange and Dial controls. The DataFind control can access databases using 3 methods:

1. ODBC drivers.
1. the Microsoft Jet Engine in conjunction with a Data control.
2. the Microsoft Jet Engine using recordsets created in VB code.

DataFind can perform searching using data from other controls including collected digits, and can search using several different techniques. The search can start from the current position in the database, or from the beginning. Recordsets can be created for each call based on input from the caller.

If the control is entered via the top input, the DataFind control performs a MoveFirst, and then searches for the first database record that matches the search criteria.
If the control is entered via the Next input, it will perform a MoveNext on the recordset, and if matching is enabled, will search for a record that matches the specifications supplied in the Matching property page. If this is the first search since the recordset was assigned, or it has been reset with the Reset() method, the database is searched from the beginning (the control will not do a MoveNext).

The position in the recordset can be set back to the beginning using the Reset() method. See the DialDB example form.
Normally each channel will either have it's own recordset, or will maintain it's own position in a shared recordset, dependent on the database type. This allows each channel to run independently. A flag can be set so that each channel shares the same recordset and position.

Exit to...

If the control is entered via the main input, and record is not found, the call is routed to the "None Found" node.
If the control is entered via the 'Next' input, and the record is not found, the call is routed to the "Not Found" node.
If the record is found, the call is routed to the "Found" node.

Searching for data
There are 2 basic ways of selecting the records of interest in a database:

DataFind data matching
VBVoice provides some basic data matching capability whereby it will search for a record using a simple

matching criteria against one field in the database. This matching criteria will compare a field in the database against some provided data, which can include caller's input or other database data, and will select each matching record in turn. Thus each caller can access a different set of records based on their selections.

Using custom recordsets

Another way of creating a recordset based on the caller's requirements is to use the ODBC capability or to create recordsets in code. This allows you to create more complex queries based on more than one field using SQL statements. See Filters below.

Data matching setup

If *All records* is checked, the DataFind control will iterate through all records one at a time, moving forward one record each time the control is entered by a call.

If *Records Containing* is checked, the DataFind control will search the database looking in the specified field for the data, using the Match specifications.

Data can be matched either using a digit to alphanumeric translation, (i.e. translating the letters on the telephone keypad to the corresponding numbers) or exact matching. Data can be matched completely or using a 'findfirst' method, where the database field is allowed to be longer that the supplied value. For example, the supplied digits 12 will match database records containing 123, 1245 and 12, but not 1 or 13. Using digit translation, the data field '234' will match 'ADH' and 'BEG'.

If the database is empty, call will exit out of the None found output. This is the case even if the DataFind is only being used to load a database for update using a DataChange or Dial control. So we recommend you always start with at least one record in your database to avoid problems.

Note that it is preferable to use a filter to select just the data you want, and then select All Records in the Data Matching property page, than to use the Data matching to select the records. This will cause the database to do the work of finding the required records, rather than the control, thus reducing the performance overhead. See Filters below, p **Error! Bookmark not defined.**.

Sharing a recordset between channels

The DataFind control will normally keep track of the position in the database for each channel. Sometimes it is desirable to maintain a position independent of channel, for instance in an out-dialing application where each number in a database should be dialed once, regardless of which channel performs the operation. In this case the recordset should be set to Shared. In this case the position will be global: each channel that enters the control will use and move a common position pointer.

Sharing the recordset:

1. Using a recordset created by code:, use the SetRecSource method with the channel parameter set to -1. All channels will then share the same recordset.
1. ODBC databases: check the Share Recordset checkbox in the property page: all channels will share the same. *Note* this requires an ODBC driver that supports bookmarks. The Filter statement should not contain any references to control properties (since these are channel specific)
2. Using a data control: check the Share Recordset checkbox in the property page. Although the data control is always shared between channels, the DataFind control will normally keep a bookmark for the position of each channel. If Share Recordset is checked, it will use the same bookmark for each channel.

Performing search by name

To use the DataFind control for directory lookup, use *Find record using... Alpha digits* to perform numeric to alphabetic translation, and use *Match On: First* so it finds all the records that start with the digits that the caller enters. See the Directory page on the VOICMAIL example.

Accessing database data

Once a record has been found in the database, the DataFind control will retrieve the data values for the fields specified in the Database property page. These data values can then be used by subsequent controls as necessary, for instance in the example below the database is providing a DialNumber field. To use this data in a Dial control, set the number to dial in the Dial control to %DataFind1.DialNumber%. Additional fields could be used elsewhere as required. The data can also be accessed from code using the Fields object.

See also
DataChange

Shortcuts to:

- Searching for data
- DataFind data matching
- Using custom recordsets
- Data matching setup
- Sharing a recordset between channels
- Performing search by name
- Accessing database data

Using a recordset created in VB code

Creating recordsets
Recordsets can be created in code or the recordset in a data control can be assigned to the DataFind control using code. When using code, each channel can be assigned a different recordset. If you wish to share the same recordset between channels, use SetRecSource with a channel parameter of -1.

Assigning the recordset created in code

form declarations: Dim MyDB As Database, MyData As Recordset

in Form.Load:

```
Set MyDB = Workspaces(0).OpenDatabase("c:\vbv\invdemo\inventry.mdb")
```

in Enter or EnterB:

```
Set MyData = MyDB.OpenRecordset("Inventory", dbOpenDynaset)
DataFind1.SetRecSource 1, MyData
```

Assigning a recordset derived from a data control

in Enter or EnterB:

```
DataFind1.SetRecSource 1, Data1.Recordset
```

The recordset can be manipulated using the underlying recordset object or data control. For instance, If the MoveFirst method is executed on the recordset, the next access by the DataFind will start searching from the first record.

Cautions

- You should not assign the same recordset to more than one channel by multiple calls to SetRecSource since this may result in faulty behavior. Each channel should have it's own recordset, or the value -1 should be used for channel, which will share the recordset between all channels. When using a recordset shared between all channels, VBVoice will use a common position in each recordset for each channel. Otherwise VBVoice will assume that the each recordset is different and has a separate current record.
- Each recordset assigned to a DataFind control must be derived from the same table or set of tables, since the DataFind maintains a global list of fields and field positions in the recordset. The list of fields is used to access the data from the current record when a call enters the control. It must be set up using the Database property page using the Edit, New and Delete buttons

Filters

You can generate custom queries for each channel when using recordsets created in code by adding a WHERE clause to the query of the RecordSet.
For example, use Find [*tablename*] WHERE *field* = 'value' to get a subset of records.

Using an ODBC database

To open a database using ODBC, you must first install the appropriate driver. Visual Basic comes with ODBC drivers for most common databases.

Creating an ODBC data source from an existing database

1) Open the Database property page and click on the .... button.
2) In the SQL Data Sources dialog, press New.
3) Select the appropriate driver (Microsoft Access for . MDB databases), press OK
4) In the Microsoft Access database setup dialog, press Select and choose the database file you want to use, and press OK.
5) Type in a name for the database in the Data Source name field and press OK

You will now be able to select the table name and fields from the table.

You can use the **Filter string** to add any SQL statement to the default query generated by the DataFind control. The control will normally generate a query of the form *'SELECT field1, field 2 etc. FROM table'* using the field names and table name provided in the Database property page.
The **Filter string** will be concatenated on the end of the query, allowing you to add WHERE and ORDERBY or other clauses to further qualify the data you want in the recordset. Each call will have it's own recordset (unless Share Recordset is checked), so the Filter statement can contain references to other controls, allowing each call to have it's own recordset. Note however that each recordset must have the same set of fields.

Filters

See also Filter

You can use the **Filter string** to add any SQL statement to the default query generated by the DataFind control. The control will normally generate a query of the form *'SELECT field1, field 2 etc. FROM table'* using the field names and table name provided in the Database property page.
The **Filter string** will be concatenated on the end of the query, allowing you to add WHERE and ORDERBY or other clauses to further qualify the data you want in the recordset. Each call will have it's own recordset (unless Share Recordset is checked), so the Filter statement can contain references to other controls, allowing each call to have it's own recordset. Note however that each recordset must have the same set of fields.

Using a VB data control

At first sight the Visual Basic data control appears to have a daunting array of properties, however it requires only 2 properties to be set in order to connect it to a database. These are DatabaseName and RecordSource. In addition, the DataSource property in the DataFind control must be set to the name of the data control. This setting is not accessible from the property page, but must be made using the Visual Basic properties window.
The DatabaseName property defines the file containing the database. If using Access databases, this file should be the Microsoft Access database file, created with Access or with the Data Manager Add-in..
The RecordSource property defines the recordset within that database that the data control will access. Although it is possible to use SQL statements in the RecordSource property, for most applications a table name will suffice.
If you are not writing to your database, you can improve performance by setting the ReadOnly property to TRUE.
To gain more experience with the data control, look at the Visual Basic sample program in the BIBLIO directory.

Filters

When using a Data control directly, any filter applied using the RecordSource property of the Data control will be applied to all channels. Since the recordset in the Data control is shared by all channels, the RecordSource property should not be changed once the system has started. If each call requires a recordset to be created dynamically based on caller input, either an ODBC database or a recordset created by code should be used.

A sample SQL statement for the RecordSource property

(taken from the Biblio Sample in your VB directory)
SELECT Titles.Title.Dept, Author FROM Titles, Authors WHERE Titles.AU_ID = Authors.AU_ID

Choosing the database type

The easiest way to access a database is to use the VB data control. When using the Jet engine, either via a recordset created in code, or using the data control, all accesses to the database are serialized (see Multi-threading below). Better performance may be achieved by running each channel in it's own copy of the application, or the data access for each channel in it's own OLE server, or by using the VBVoice ODBC support with multi-threaded ODBC drivers. The data control can be used to access both local databases and networked databases. Networked databases can be accessed either directly over the network or by using Access to set up an 'attached' database, which can provide a performance increase.
ODBC can also be used for both local and remote databases.
Using ODBC provides a capability of creating a different recordset for each channel using criteria based on caller's input. This is more efficient and more flexible than performing data matching, which can be a slow operation on a large database. Creating recordsets in code also provides this capability, at the expense of more coding work. ODBC can be more efficient when accessing databases over a network.

Multi-threading
Multi-threading is important when accessing large databases in multi-channel applications. Microsoft Jet database drivers are not multi-threaded so that each channel must serialize it's database access through one thread. One channel doing a large database access may delay other channels. In this situation, ODBC will provide better performance.
See also

- Using Databases
- Using an ODBC database
- Using a data control
- Using a recordset created in code

Result

Applies to DataSwitch, IniSwitch

This Result property is a string containing the value of the field found by the last search. It can be read and set by VB code in the Enter or Error event before branching takes place.

Example 1
To access the result:

mystring = DataSwitch1.Value(channel)

Example 2
To override the result:

DataSwitch1.Value(channel) = "this is the result I really wanted"

DataSource

(data control name)

Applies to DataFind

This field contains the name of the data control that defines the data to be acted upon.The data control that defines the database containing the message tables. using the DataBaseName and RecordSource properties in the data control.
This property, nor the Recordsource or DatabaseName properties of the data control should be changed while the voice system is running.

RecordCount

Array Integer(1, MAXCHANNELS)

Applies to DataFind

This field contains the number of fields found in the database that match the given specifications. The property is set to 0 when a call enters the control via the main input, and increments by 1 every time a record is found. If the 'Get record count' box is checked in the property page, the record count is calculated as soon as a control enters via the main input. Counting fields can be time consuming in a large database, and should be avoided if possible.

Example - reading the RecordCount

You may want to execute some code based on the value of RecordCount:

```
if (DataFind1.RecordCount(channel) = 0 then
        User.GotoNode(channel) = 0;
Endif
```

Database property page (ODBC)

See Also
DataFind
Choosing the database type
Using an ODBC database

If Use ODBC database is set, this property page is used to define the recordset on which to perform data searching.

ODBC Connect String (ODBCConnect property)
Press the ... button to open the ODBC selection dialog. See *Using an ODBC database* above for a description of how to create an ODBC data source from an existing database file or directory.

Use this table… (ODBCTable property)
This drop-down list box will show the list of tables available once the ODBC Connect string has been set.

RecordSet Types (RecordsetType property)
- Snapshot
- Dynaset
- Forward-only

For a description of database types see Using Databases.

Share recordset between all channels (UseGlobalBookmark property)

Get record count (GetRecordCount property)
This field can be checked if you wish to have the DataFind control count the number of records as soon as a call arrives in the control. This may be useful if you wish to announce the number of items in a database before allowing the caller to choose. This may have a severe performance impact on large recordsets. Record counting terminates automatically if the count exceeds 500.

Control List 

This button shows the Control List window. Any field that will accept control names or properties has a Control List button next to it.

Matching property page

See Also DataFind

Find:

All records in recordset (MatchType property)
If this button is checked, DataFind will select the next record on each entry. The data matching fields will not be visible when this option is checked.

Records where ... (MatchField, MatchData properties)
If this button is checked, DataFind will search the database for the next record that matches the specifications in the *matches this data* text field and Match type specifications.
The text you enter may contain control names. If using a Jet database, you will have to enter the fields manually since the control cannot find out which database will be used at runtime. When using bound data controls or ODBC databases, a drop-down list of fields will be provided.

Matching (AlphaMatch, MatchAllChars properties)

Match fields exactly
This search type will do a normal comparison, character by character, between the database contents and the data in the *For...* field.

Use digit translation
This search type will translate the text in the database field to the telephone keypad equivalent before performing the comparison. This is useful for directory lookup, etc., where the caller is limited to using telephone keys to represent an alphanumeric name.

Match on first characters
This search algorithm will match the *Records containing* text field against the database contents. It will stop matching and report a successful match if it reaches the end of the *Records containing* text field without a mismatch. See Data search specification, p **Error! Bookmark not defined.**.

Match on all characters
This will match the entire text field against the database contents. In this case the database field and the contents of the selected field must be exactly the same.

NOTE: if doing a Match on characters exactly, and Match on all characters, it is more efficient to setup your recordset to include only the records you want by using a WHERE statement or using a Filter (not available if using a bound data control).

Database property page (using Data control)

See Also
DataFind
Choosing the database type
Using a data control

DataSource property
When using a bound Data control, the Data control must be set using the DataSource property in the VB properties window. Microsoft provides this property, but has not given us the ability to embed this selection into a property page, although they tell us we must use them for all of our properties. In addition, the DatabaseName and RecordSource properties of the Data control should be set before using this property page.
See also Using a data control

Get these fields…
Use the Change button to select a list of fields from the selected table. The DataFind control will get the actual values of these properties when a call enters the control after moving to the record. If a field is not in this list, it will not be accessible from other VBVoice controls, and will not be available for update from the DataChange or Dial controls.

Share recordset between all channels
If this check box is not checked, the DataFind control will maintain a bookmark for each channel. This bookmark will be used to restore the position for a channel prior to searching for the next record. If this flag is checked, the DataFind will maintain one global bookmark, so each channel will use the same position.

Get record count
If the ‘Get Record Count’ box is checked on entry, the record count is calculated as soon as a control enters via the main input. Counting records can be time consuming in a large recordset, and should be avoided if possible.

Database property page (using recordset from code)

See Also
DataFind
Choosing the database type
Using a recordset created in code

Creating the list of fields
If Use recordset defined by code is set, the recordset is created in Visual Basic from code or from a VB Data control. You will have to provide a list of fields to be retrieved by the DataFind control, since VBVoice does not have access to your database at design time. The list of field names is used to access data from the database, and also to provide a list of fields that can be updated using the DataChange control. Use the New, Edit and Delete buttons to modify the list of fields to be accessed or updated.

Example code using a data control

```
DataFind1_Enter (....)
‘ using a data control
DataFind1.SetRecSource( channel, Data1.RecordSet
End Sub
```

Example code using a recordset created in code

```
DataFind1_Enter (....)
‘ using a data control
DataFind1.SetRecSource( channel, MyRecordSet)
End Sub
```

Runtime Properties

- GotoNode
- Result

Initial Setup Properties

AlphaMatch Integer

Conditions Object

DataFind String

FieldName String

MatchAllChars Integer

Events

- Condition
- Enter
- Exit
- VoiceError

DataSwitch control

 Properties Events Initial Setup Properties Methods

Property pages

Setup

Data Condition dialog

Default Property: Result

Description:

When a call enters the DataSwitch control, it reads a field from the database record found by a previous DataFind control and stores the result in the property 'Result'. The call is then passed to the control connected to one of the output nodes, based on a comparison operation performed on the field value obtained.
The data to be compared may contain property names. The Result property can be used by other controls, and VB code.

Pattern Matching

The DataSwitch control can test the data against as many different patterns as required. Pattern matching can be performed exactly, or using a digit to alphabetic character translation. Translation is useful when matching collected digits entered by the letters on the telephone keypad against names. In addition, the DataSwitch control can perform complete matching, or it can use a find-first method which will match any field that begins with the characters given. For instance, a condition "JO" will match fields containing both "JOHN" and "JONAS" in the database.
If the record is found, but there is no match between the data field and any of the conditions, the call is transferred to the "No Match" node. If a match is found between the value obtained and one of the result fields the call is routed to the appropriate node.
You can also bypass or add to the DataSwitch condition matching by checking the ‘Let VB decide’ check box in the Condition dialog. This will generate a Condition event allowing your code to determine the result of the comparison.

Using with DataFind

The DataSwitch must be connected to a DataFind control either directly or through other controls.

Result

Applies to DataSwitch , IniSwitch

Array String(1, MAXCHANNELS)
The Result property is a string containing the value of the field found by the last data access or ini file read operation. It can be read and set by VB code in the Enter or Error event before branching takes place.
Example 1
To access the result:
mystring = DataSwitch1.Value(channel)

Example 2
To override the result:
DataSwitch1.Value(channel) = “this is the result I really wanted”

Condition Event

Applies to DataSwitch , GetDigits, IniSwitch

This event is generated when you specify ‘Let VB code decide’ in the condition dialog.
Sub xx_Condition (channel as Integer, decided as Integer, node as Integer);

If your code determines that there is a match for this output node (as specified by the node parameter), it should set the decided parameter to TRUE and exit. Otherwise pattern matching will continue on the next node. If your code changes the node parameter in addition to setting *decided* TRUE, the call will be transferred to the node number you set. Nodes are counted from 0, starting from the top node.

DataSwitch control:
You can access the node name and the result using the Nodename and Result properties.

GetDigits control:
You can access the digits using Digits property.
Digit collection will terminate when the maximum digits or a a terminating digit have been received, or digit collection has timed out. The condition event is defined as

Data Condition dialog
Digit Condition dialog

DataSwitch property page

See Also DataSwitch

Exit is this field matches ... (Conditions property)

This is a list of the available fields in the database to which the DataSwitch is linked. The database is selected by the DataFind shown in the field below. This list is only available when using ODBC data sources. When using Jet, you will have to enter the field names yourself.

Matching selections:

Specify here the kind of comparison that you want to perform between the database contents and the condition fields.

Match characters exactly using AlphaMatch property
This search type will do a normal comparison, character by character, between the database contents and the data in each condition field.

Match using digit translation
This search type will translate the text in the database field to the telephone keypad equivalent before performing the comparison.

Match on all characters ((MatchAllChars property)
This will match the entire database field against the condition field. The database field and the condition field must be exactly the same.

Match on first characters
This search algorithm will stop matching when it reaches the end of the characters in the condition field.

Reordering conditions

Conditions are matched in top to bottom order. In circumstances where more than one condition may match, you can order the conditions by dragging and dropping the items in the condition list

Edit command
Use the Edit command, or double-click on an entry to show the Conditions dialog.
Delete command
Deletes an entry
New command
Use this command to create a new condition.

DataFind control
Data Condition dialog

Data Condition dialog

See also DataSwitch

Let VB code decide..

Check this box if the pattern matching capablility in DataSwitch does not meet your requirements. For every node that has this button checked, the DataSwitch control will generate a Condition event in the control, allowing your code to decide if the digits meet your criteria for acceptance. See Condition event above.

Database Field Value

Sets the value required in the database field to make the call exit vai this node.

Condition Name

The name of the condition that is drawn in the node which owns this condition.

Control List 

This button shows the Control List window. Any field that will accept control names or properties has a Control List button next to it.

Properties

- DelayTime
- GotoNode
- StopDelay

Initial Setup Properties

ClearDigits BOOL

HelpdigitControl String

HoldType Integer

IDelayTime Integer

TermDtmf Integer

Events

- DelayStarted
- Disconnect
- Enter
- Exit
- PhraseError
- PlayRequest
- VoiceError

ClearDigits property

Use this property if you want to clear all previously collected digits from the VBVoice digit buffer. You may want to allow experienced callers to save time by ‘typing ahead’ since they know which order the prompts are played, and what digits are required.

Delay control

Properties Events Inital Setup Properties Methods

Default Property: None

Property pages

Delay

Error Handlers

Greeting Properties:

EntryGreeting

MusicGreeting

Description:

Implements a wait period. A Delay control can be used if your VB code has to perform a lengthy activity, and you want the caller to wait until it is complete, for instance to wait for a busy line to be free before re-attempting a transfer. The Delay control can operate in three modes:

HoldType

Simulated Hold: VBVoice plays a greeting (hold music) to the caller during the wait period.

Hold: VBVoice puts the caller on hold using the PBX hold feature, as defined by the *Put on hold for onhook delay* and *Reconnect from onhook delay* dialing strings, and hangs up the line while waiting. See VBVFrame property page for PBX Settings
NOTE: This mode should only be used if there is no possibility of another incoming call on this line.

Silent: VBVoice remains silent until the delay is complete.
When playing hold music, VBVoice continually plays a short file containing music or a short announcement. This can be replaced with your own phrase if required.

StopDelay

Applies to Delay

Array Integer(1, MAXCHANNELS)
Set the StopDelay property to zero at runtime during a wait period to terminate the delay and allow the call to continue.

Example
DataSwitch1.StopDelay(channel) = 0

DelayStarted

Applies to Delay

This event occurs after the caller has been put on hold. Use this event to initiate any lengthy activity required, rather than the Enter event. Yielding is still required to allow other channels to run. (See Properties and Events). If you start a lengthy activity in the Enter event, the caller will not be put on hold , since the Enter event will not return to VBVoice for that channel, even if your code yields to Windows.

IDelayTime

Applies to Delay

Delay time in seconds. Affects all channels. Set this property to 0 for an infinite delay time.

DelayTime

Applies to Delay

This property can be set in the Enter event to override the default delay time. The new value is valid for the current call only.

Delay property page

See also Delay

Put on hold during wait

Music during wait

Silence during wait

HoldType**property**
These three buttons select the action to be taken during the wait period. Set the music greeting using the Greetings tab. This can be used to play a prerecorded piece of on-hold music or any other recording that you want your caller to listen to.

Delay time (lDelayTime property)
Sets the period of the delay, in seconds.

Always terminate on (TermDtmf property)
If this property is set to a digit, the music greeting will terminate on the specified digit, regardless of the Terminate on Digit setting in the greeting itself.

Runtime Properties

CallResult Array String

CallTime Array Integer

GotoNode

NumToDial

Initial Setup Properties:

DetectAnsMachine BOOL

DisconnectControl String

DoTransfer BOOL

HelpdigitControl String

INumToDial String

UseDatabase BOOL

WaitForDialtone BOOL

See Also Dial property page

Events

- Disconnect
- Enter
- Exit
- PhraseError
- PlayRequest
- VoiceError

CallResult

This property contains the result of the last dial operation on the specified channel. It uses the vbvCallResultConstants:

vbvNoDialTone	(27)
vbvFastBusy	(28)
vbvBusy	(29)
vbvNoAnswer	(30)
vbvNoRing	(31)
vbvConnected	(32)
vbvIntercept	(33)

CallTime

This property contains a formatted string containing the start time of the last call. The string is formatted as mm/dd/yy hh:mm using a 24 hr clock. Useful when creating call logs using the Dial Log option.

Dial control

Properties Events Initial Setup Properties Methods

Default Property: None

Property pages

Setup

Error Handlers

Greeting Properties

EntryGreeting

MusicGreeting

Description:

The Dial control can be used to go offhook and start a call, or on an existing call it can perform a transfer to another extension or dial some digits to an automated system at the far end. If the number to dial is prefixed with the letter 'S', call supervision is enabled to monitor the progress of the call until the result is determined - one of the outcomes listed in the Supervised Dial box above. It does this by listening and analyzing the tones being returned from the telephone system after dialing. A built-in call logging facility using a database is also provided.

Transfer

If Use Transfer is set, the Start Transfer digits will be dialed before the supplied string. The Start Transfer digits can be set in the VBVFrame PBX Settings properties page. The Dial control will then wait for dialtone if Wait For Dial Tone is enabled. If no dialtone is found, the control will dial the *Reconnect from busy* digits and exit via the NoDialTone output. If dialtone is found or not enabled, the control will dial the digits and proceed as follows:

Supervised transfer

If supervision is enabled, the control waits until a time-out or a known telephone condition is detected on the line, such as answer, busy or fast busy.

Answer: The call proceeds while connected to the new transferee, and the original caller is kept on hold. The original can be reconnected using the *Complete transfer after answer* string in a subsequent Dial control. See also Detect Answer Machine below.

Any other result: The original caller is reconnected using the *Reconnect from hold...* string, and the call continues with the original caller, in the control connected to the output node as defined by the call progress type.

Unsupervised transfer

If supervision is not enabled, a blind transfer will be performed by dialing the transfer digits and the number. After dialing, the call will be terminated.

No Transfer

Supervised dial

The line is taken off-hook, if not already off hook, and the control waits for dialtone before dialing the digits. Once the digits are dialed, the control waits for a recognized call progress condition (answer, busy etc.) and exits from an output node depending on the call progress result. This mode can be used to start a call. See also Detect Answer Machine below.

If the control is being used to initiate a call, note that it is possible for a call to arrive just as the call control is going off-hook. In this case, the call will continue via the NoDialTone output.

Unsupervised (blind) dial
This mode can be used to dial some digits during a call, for instance to send some digits to another voice response system.
The digits are dialed and the call is transferred to the control connected to the Dialed output.

Configuring the dial string

The number to dial can be fixed at design time or can be changed on each call, either by adding a control name to the dial string, or by using the NumToDial property in the Enter event. If using a control name, all call progress outputs will be shown in the control, since VBVoice cannot determine until the call starts whether the dial string starts with the letter 'S', indicating call supervision is required. Call progress supervision will only occur if the 'S' is present in the final string.
The number string can contain hyphens and spaces, which will be ignored. The maximum length of this string is 64 characters. The character '^' will be replaced by the escape character (0x1b) for use by the D42NS and D42SX special dial strings.

Detect answering machine

Note: not implemented in current release - use Record control method below

If the detect answer machine feature is enabled, and the call is answered with speech, VBVoice will wait until after the speech has ended, or up to 7 seconds, before continuing. If the speech terminates before 6 seconds, it is assumed that a real person has answered the call. If the speech continues for at least 6 seconds, it will assume that an answer machine has answered the call. The time used to differentiate between the two types can be set in VBV32.INI, section PBX, field DetectAnswerTime (default 6000 ms). (TODO)
An alternative method of answering machine detection is to use a Record control after answer has been detected, with a maximum time of 7 seconds, a minimum of 5 seconds, and a retry count of 0. If the call exists out of the NoMsg output, the answer was a human.

An alternative method of answering machine detection is to use a Record control after answer has been detected, with a maximum time of 7 seconds, a minimum of 5 seconds, and a retry count of 0. If the call exists out of the NoMsg output, the answer was a human.

Note that answering machine detection is based completely on the duration of the answer. Long pauses in the answer, or a short message on an answering machine can both result in failure to detect answering machines.

Database options - call logging

If the Update Database option is set, the control will add a record in a database for the call. The Dial control must be connected to a DataFind control, either directly or through some other controls. The Dial Logs property page can be used to set up a list of fields to be updated. This property page works in the same way as the DataChange control. In the DIALDB sample, a database record containing the dialed number, the call result and the time of the call is added to the database on each call. The CallResult property contains one of the vbvCallResultConstants.

NumToDial

Applies to Dial

Array String(1, MAXCHANNELS)
The dialed number is recalculated for each call if it contains control names. The NumToDial property makes the calcualted number available to VB code in the Enter event, and code can also set a new number. This does not change the number for subsequent calls..

Example 1
To access the number:
Dim actualnumber as String
actualnumber = Dial1.NumToDial(channel)

Example 2
To override the predefined number with your own:
Dial1.NumToDial(channel) = ‘‘613-839 0033’’ ‘spaces, hyphens and brackets are allowed
‘max string length is 64 characters

CPLFile

Applies to Dial

Array String(1, MAXCHANNELS)

The CPLFile property defines the set of parameters to be used when determining call progress conditions - busy, ringback, no answer etc. You may need to specify a different file for in-house PBX calls and for outside lines, due to differences between the ringing and busy tones supplied by the PBX, and those received from the public network. Normally this can be done using a database entry to specify the file, however it can also be set by code in the Enter event. When the CPLFile property is changed at runtime, the change only applies to the call that caused the event.

Look in the Getting Started booklet for your voice card for how to create CPL files and to configure call progress analysis for your card. The CustomCPL check box must be set in order for this property to take effect. The structure of the CPL file is described in the .BAS definitions file for your voice card. The CPL file can also be created using the MAKECPL utility. Some voice cards have their own utilities for defining call progess analysis.

Dial property page

See also Dial

Number to call (INumToDial property)
This field is used to enter the number that this control should dial. The field can contain control names, for instance it can refer to a field name in a DataFind control that will cycle through a list of numbers to dial. If the number begins with 'S', or if the dial string starts with a control name, call progress analysis is enabled and the related fields are made visible.
See *Configuring the dial string* above.

Wait for dial tone (WaitForDialtone property)

Use Transfer feature (DoTransfer property)
Check this button if you want the Dial control to transfer the current call to the number entered. See *Transfer* above.

Detect answering machine (DetectAnsMachine property)
TODO: this option not supported yet

Update database (UseDatabase property)
If this check box is set, the data fields specified in the Dial Logs page will be updated after the dial operation is completed. See Dial Logs property page below.

Control List 

This button shows the Control List window. Any field that will accept control names or properties has a Control List button next to it.

Dial Logs property page

This property page is used to setup a list of database fields which are to be updated with the result of this dial operation. In the example above, taken from the DialDB example form, the Dial control adds a dial log entry to this record, consisting of the number dialed, the time taken, the call result, and the channel which was used for this dial operation.
Other fields can be added as required.

Show available fields

Adds all available fields into the field list box. The available fields are set by the DataFind control selected in the Change data... dropdown list. When using a data control or an ODBC database, the control will show all available fields. When creating recordsets in code, only fields specified in the DataFind database property page will be available. To add other fields in the database, go to the DataFind setup and use the New, Edit and Delete buttons to modify the field list.

Properties

- Digits
- GotoNode
- MaxKeys
- MaxSil
- TermDtmf

Initial Setup Properties

ClearDigits BOOL

Conditions Object

DisconnectControl String

DisableHelp BOOL

HelpdigitControl String

IMaxKeys Integer

InvalidErrorControl String

ITermDtmf Integer

IMaxSil Integer

NoDigitsErrorControl String

RetryOnSilence Integer

UseDefaultError BOOL

Greeting Properties

EntryGreeting

InvalidGreeting

NoDigitsGreeting

Events

- Condition
- Disconnect
- Enter
- Exit
- PhraseError
- PlayRequest
- VoiceError

GetDigits control

Properties Events Initial Setup Properties Methods

Default Property: Digits

Property Pages

Terminations property page

Routing property page

Error Handlers

Digit Condition dialog

The GetDigits control plays a greeting and waits for digits from the caller for a predetermined amount of time. Pattern matching is performed on the collected digits using as many exit conditions as required. GetDigits provides built-in invalid digit and time-out handling.

Digit Pattern Matching

The termination conditions can be as simple as a single digit per exit branch (the default configuration) or complex sequences consisting of the following characters:

$	any char or series of characters. Use a terminating digit, maximum chars or silence to terminate input.
n	any single numeric char 0 - 9
- (hyphen)	specifies a numeric range - i.e. 222-333. The start and end of the range must have the same number of digits. For example, to cover 00-999, use 2 separate entries, 00-99 and 100-999.
?	specifies any one char
0-9 *#	each character specifies an actual digit to be received.
Let VB code decide	In addition, you can specify that a VB event handler should decide if the digit pattern is correct. Used in the Voicemail example to check the caller's new password before setting it in the database. See Condition Event.
Control Names	Conditions can also contain control names. This is sometimes useful when comparing digits received against database values. This method is used in the Voicemail example to check the subscriber's password.
Wild Chars	If using a wild character as a termination in conjunction with other termination conditions, put the wild character in the last digitmask in the list. The digitmask list is processed from top to bottom after the digits are collected, so the wild character may hide other valid digitmasks that should take priority.

Play Termination

Normally GetDigits controls will be configured to caller who know your system to bypass the Entry greeting by pressing one or more digits. However in some circumstances you may have important information that you want the caller to hear. You can do this by clearing the Terminate on digit' checkbox in the property page for the Entry Greeting. If this flag is set, the greeting will always play to completion and any collected digits will be cleared before starting to wait for digits. (The assumption is that if they are pressing digits, they are not listening to the prompt, which you want them to do).

Wait for Digits Termination Conditions

The GetDigits control automatically calculates the required termination conditions for collecting digits from the digit masks provided in the list. It can select termination based on maximum number of digits, termination characters, or number of seconds without any digits. A warning will occur on startup if it is unable to calculate a termination condition other than silence.
It is preferable to avoid using a silence timeout as the only terminating condition since this can slow down progress through the system. We recommend setting a terminating digit if a variable number of digits is being requested, i.e. if any condition contains a wild character $.

In these cases VBVoice has no other way of detecting whether the caller has entered all the digits.
If silence is the only possible termination, the response time for the caller will be longer, and type-ahead will not be possible. The system has to wait for the silence time-out period in order to determine that the caller has stopped entering digits.
You can avoid using a silence timeout with the Max Digits or Always Terminate On digit setting. In most systems, # can be used as a termination digit to avoid using time-outs. If a termination digit is set using this field, the digit will be stripped from the received digits before continuing, i.e. the Digits property will not contain the terminating digit even if received. If a terminating key has been set, and it is required to set a valid condition where only the termination digit is entered with no other digits, use a blank digit mask in the condition field.

Type-ahead and silence timeout terminations

Note that if a GetDigits control terminates with a silence timeout, all digits collected will be consumed by that control. Even if the GetDigits matches on a field with 2 digits, and 3 digits have been entered, the extra digit will not be available to subsequent controls for type-ahead operations.

Invalid digits, No digits handling

If an invalid digit sequence is entered by the caller, i.e. a sequence that does not match any of the digit masks, or if no digits are entered, the control allows the user to try again up to a preset number of attempts. A separate prompt can be set for either or both of these events. If the error count exceeds the number of retries set when invalid digits or a no digits event occurs, the GetDigits control will perform error handling as described below.
If the error count has not been exceeded, and an invalid digit sequence has been received, VBVoice will play the InvalidGreeting, if set, followed by the **EntryGreeting**, and increments the error count. The default Invalid Digits Greeting is “that was not a valid entry”. If no digits have been entered, VBVoice will play the NoDigitsGreeting, followed by the EntryGreeting, and increments the error count. The default NoDigitsGreeting is empty. When the retry count is exceeded, VBVoice will transfer to the control connected to the Invalid output, if this has been enabled using the Use default error handler checkbox. If not, it will attempt to invoke one of the default error handler in the usual way:

- The Invalid Control handler set in the Error Handlers dialog
- LineGroup Invalid exit
- Play the file ERROR.WAV, and hang up.

See also Error Handling

Err input

The GetDigits control can also be entered from a second input "Err". This input causes the control to act as if invalid digits had been received: it increments the error count, and if it exceeds the maximum retry count, the call is transferred to the 'Invalid' exit node. This input can be used when digits have been collected and are then checked against a database. If the digits are incorrect, they can be treated identically to digits checked by the control itself.

MaxSil

Applies to GetDigits

This is the maximum duration of silence (in seconds) while waiting for digits before issuing a time-out. This property is set by the control to the value of *Maximum Silence* in the Terminations property page. It can be changed in the Enter event to another value if required. The new value will only affect the current call. A value of 0 means that the control will not wait for digits after the greeting has finished playing, but will check the buffer immediately.

Example

To override the maximum silence setting:

```
If CallerName = “Fred” Then
    GetDigits1.MaxSil(channel) = 5        ‘stop after 5 seconds silence
Endif                                     ‘Fred is a slow talker
```

MaxKeys

Applies to GetDigits

The maximum number of keys to accept. This value is set by the control to the default Max Keys set in the setup dialog, or a value based on the digit masks entered. It can be changed in the Enter event to another value if required. The new value only affects the current call.

TermDtmf

Applies to GetDigits

This is a bit-mapped array indicating the keys that will terminate the digit collection. This property is calculated at startup based on the digit conditions in the control. It can be changed in the Enter event to another value if required. The new value only affects the current call Use the predefined vbvDigitMaskConstants vbvDigit_0, vbvDigit_1 etc..

Example

To override the TermDtmf property:

```
GetDigits1.TermDtmf(channel) = vbvDigit_0 OR vbvDigit_S
        'terminate on 0 and * only
```

Digits

Applies to GetDigits

This property contains the digits collected by the control. This is the default property.
It is sometimes useful to override the result of the GetDigits control: if you have two paths that both collect the same number - i.e you may be asking for a mailbox number, and provide a method of direct input, or a method using database lookup. Subsequent code must reference one control only for the result, so you can use the exit event in say the database loopup path to set the Digits property for the direct path as if the caller had entered the digits directly via than by database lookup. This method is used in the VMDEMO program, in the DIRECTRY form.

Example 1
To access the result:
mystring = GetDigits1.Digits(channel)

Example 2
To override the result:
GetDigits1.Digits(channel) = ‘‘this is the result I really wanted’’

GetDigits Terminations property page

See also GetDigits

Disable global help digit (DisableHelp property)

Set this check box to disable the help digit handler provided by the LineGroup control for this call. If this box is not checked, then if the help digit specified by the LineGroup control is detected in the digits entered by the caller, the call will exit this control and enter the control connected to the LineGroup HelpDigit output. The help digit handler specified by the Error Handlers property page will also be disabled by this field.

Use default error handler (UseDefaultError property)

This check box is set by default. When the maximum retries for invalid digits or retries have been exceeded, the system will check for an error handling control or a connection on the LineGroup error output. If these conditions are not true, the ERROR.WAV file is played and the system hangs up. If this check box is not set, 2 new outputs appear on the control, 'Invalid' and 'Timeout'. These outputs can be connected to other controls to override the default error handler. See Error Handling.

Retry on silence (RetryOnSilence property)

If this box is checked (the default), both silence timeout and invalid digit events use the retry count. If this box is unchecked, then a silence timeout will cause the call to exit via the silence output immediately, regardless of the number of retries set. Invalid digits increment the retry count as usual. This is useful in initial menus where you want to give the caller several attempts to enter the correct digit, but want to allow callers without touch-tone phones be transferred to an operator without undue delay.

Clear digits on entry (ClearDigits property)

Set this button if you want to clear all previously collected digits from the VBVoice digit buffer. Normally you will want to allow experienced callers to save time by ‘typing ahead’ since they know which order the prompts are played, and what digits are required.

Termination conditions

Max digits (IMaxKeys property)
This field sets the maximum number of digits that can be received before GetDigits terminates digit collection.

Maximum silence (IMaxSil property)
This field specifies the number of seconds that GetDigits will wait for a digit. If a digit is not received in the time, digit collection will be terminated. A value of 0 means that the control will not wait for digits to arrive after the greeting has played.

Number of retries on error (RetryOnSilence property)
This field specifies the number of invalid digits, and no digits errors that can occur before GetDigits passes the call to the NoDigits or Invalid nodes, or invokes the global error handler.

Always terminate on (ITermDtmf property)
This field specifies a digit that can be used to terminate digit collection. This can speed up menu selection by allowing the caller to enter say the ‘#’ key to end a variable length sequence of digits, rather than having to wait for a silence time-out.

GetDigits Routing property page

See also GetDigits

Reordering conditions

Conditions are matched in top to bottom order. In circumstances where more than one condition may match, you can order the conditions by dragging and dropping the items in the condition list

Edit command
Use the Edit command, or double-click on an entry to change an exit condition using the Digit Match Condition dialog (see below).

Delete command
Deletes an exit condition.

New command
Use this command to create a new exit condition.

Digit Condition dialog

Digit Condition dialog

See also GetDigits

Let VB code decide..

Check this box if the pattern matching capablility in GetDigits does not meet your requirements. For every node that has this button checked, the GetDigits control will generate a Condition event in the control, allowing your code to decide if the digits meet your criteria for acceptance. See Condition event above.

Digit Mask

Sets the acceptable digits to allow the call to exit this node. See Digit Matching.

Condition Name

The name of the condition that is drawn in the node which owns this condition.

Find Control

This command creates or shows the Control List window. You can choose a control name from this window and copy it into the clipboard with the Copy button. The control name can then be pasted into any text field as a parameter using Shift+Insert. Only controls on loaded forms that have default properties are listed in the window. If you have a large number of controls, you can quickly find the control you are looking for by selecting the form name first to show only the controls on that form.

Properties

Digit Array Integer

GotoNode

Position Array Integer

Events

- Disconnect
- Enter
- Exit
- PhraseError
- PlayRequest
- VoiceError

Events

- Enter
- Exit
- VoiceError

Initial Setup Properties

DisconnectControl String

FastForwardDigit

FastFwdSecs

DisconnectControl String

HelpdigitControl String

LouderDigit

PauseDigit

PauseTimeout

RewindDigit

RewindSecs

SofterDigit

PlayGreeting control

Properties Events Initial Setup Properties Methods

Default Property: None

Property pages

Setup

Error Handlers

Greeting Properties:

EntryGreeting

Description:

The PlayGreeting control plays a greeting and then passes the call to the next control. If the greeting plays to completion without any digits, the call will exit out of the Done node. The PlayGreeting control can also accept digits from the caller to pause, skip forward and backward in the greeting, or to change the volume and speed of the playback. The volume and speed adjustments are reset to the default on entry and exit of the control. Any digits entered which are not configured in the property page will generate a Digit event.
If the digit is not handled in the event, the call will exit out of the Digits output. If the digit is handled by the event, playback will continue. The play can be started at any position in the greeting using the StartPosition property.
If the Terminate On Digit property is set in the EntryGreeting, no digits will be received.

Volume control operation

The PlayGreeting control provides option digits for adjusting the volume of the playback. These digits affect the volume only during the play. The volume is reset to the default when the call leaves the control

Digit

This property is set to the value of the option digit when a digit is received which does not match any of the preset option digits.

Position

Set this value to be the position in the file where the play should start. This property should be set in the Enter event. Can also be read after a play terminates to query the position in the file where the play stopped. The play may terminate before the end of the file if an invalid digit is received.

PlayGreeting Setup property page

See Also PlayGreeting

Clear digits on entry (ClearDigits property)

Set this button if you want to clear all previously collected digits from the VBVoice digit buffer. You may want to allow experienced callers to save time by ‘typing ahead’ since they know which order the prompts are played, and what digits are required.

Play Option Digits

These digits allow the caller to control the playback of a greeting by jumping ahead or moving back to replay a section.

Fast Forward Secs	(FastFwdSecs property)
Rewind Secs	(RewindSecs property)
Pause Secs	(PauseSecs property)

.

Interaction with Play Termination on Digit option

Note that the option digits will work regardless of the *Terminate on Digit* setting in the EntryGreeting. If T*erminate on digit* is set, and an option digit is received, the option is performed. If any other digit is received, the play terminates. If play termination is not set, and a digit other than an option digit is received, the OptionDigit event is fired.

Fast Forward (FastForwardDigit property)

This digit will skip the playback forward when received. When the end of the file is reached, the play will stop in the normal way, and the call will continue from the next control

Rewind (RewindDigit property)

This digit will rewind the playback backwards when received.

Pause (PauseDigit property)

This digit will terminate the playback. The playback will restart at the same position when another digit is received, or after the pause timeout time.

Louder (LouderDigit property)

This digit will increase the volume of the playback by one increment. There are 20 volume increments.

Softer (SofterDigit property)

This digit will decrease the volume of the playback by one increment..

InCon

 No Properties No Events

Description:

Used for connection from an output on one control to the input of another on a different form. Other controls cannot be connected across forms.

Properties

When creating your own system phrases, you can use most of the standard system phrases as well as phrases from VAP files and VOX files. System phrases created while in an event will not create another event - the default number implementation will always be used.

- AddType
- Filename
- PhraseName
- PhraseIndex
- PhraseData

Events

DateToday

DateV

Digits

File_Date

File_Size

File_Time

FileDateTime

InitialGreeting

Money

Number

Number_Ordinal

NumberShort

TimeNow

TimeV

Language control

Properties Events

Language Setup property page

Language Events property page

Example

The language control is an optional control that performs several functions:

- setting of language for each call using the graphical controls
- capability to override default phrase concatenation rules for the built-in system phrases with your own rules
- A user-definable system phrase type

Each language control specifies the rules for system phrase generation for one language. Up to 24 language controls are allowed on a system. The default language as specified in the VBVFrame language property page is used when the caller first makes a connection until the caller makes a language selection.

The language for a caller can be changed by routing the call through one of the language controls. This usually occurs when the caller makes a numerical selection matching the desired language. This selection is picked up by the GetDigits control and the appropriate language is then selected and used for further communication.

Each language can have a separate set of rules for saying numbers, dates and times. In addition, the built-in concatenation rules can be overridden using code as required. After a call passes through a language control, the default language settings provided by the VBVFrame control are overridden for the duration of the call.

Numbers

Support for saying English, Italian, and French number formats is built-in. New number formats can be supported by overriding the system rules with code in the SayNumber event (and turning on the System event SayNumber).

Some numbers also are gender-specific. The PhraseData field can be used to set the gender of each phrase. The PhraseData field may contain the letters M, F or N to indicate gender. To support new system phrases formats, your code must analyze the data passed in from the application , usually the string provided in the phrase definition. If the concatenation rules for the new language are the same as the chosen default, then nothing needs to be done. If different, your code must create phrases to say the data as provided. The data string is in the format specified for the appropriate system phrase type described in System Phrases. ????

Date Order

The language control specifies the order in which the date components are said. Possible values are:

- mmddyy (month, day, year)
- ddmmyy (day, month, year)
- yyddmm
- yymmdd

The date format defined by each phrase specifies the components of each date (i.e day, month, weekday, month-name, year). This is defined for each phrase. The format of the string to say is defined by the system phrase type.

Times can be said using 12 hour or 24 hour time.

Managing Phrase files

The phrase files for additional languages must contain equivalent phrases in the same order and position as for the default language. VBVoice relies on all phrase files having the phrases in the same position in the file, with the same name, or unpredictable results will occur

To simplify maintaining multiple vap files in different languages, we suggest use of the ScriptSize property. This property of the VBVFrame control allows you to set the maximum number of characters which VBVoice will use for phrase matching. This allows you to use a numeric or other prefix for each phrase name which VBVoice will use to find the phrase. The remaining characters in the name can be the actual script in the correct language.

The voice files for the other languages must be in the directory specified in the control property page, normally a subdirectory of the VBVOICE directory.

Note: when VBVoice is saying phrases that require a file path such as SayFileDate, it will not use the language subdirectory by default. It will only use the language directory when looking for VAP phrase files, or files referenced by the VAPPhrase or VapPhraseIndex phrase types. If it is necessary to reference the language directory, it is available as the CurrentDirectory property of the VBVFrame control. This property can be used to prefix a filename in order to access a language-dependent file - e.g. %VBVFrame1.CurrentDirectory%msg1.vox

Important:

- Each directory must contain a complete set of phrase files required for your application.
- Each language phrase file must have the phrases in the same order and with the same script as the equivalent phrase file in the VBV directory unless the ScriptSize property is set.
- Only phrase files referenced without a path name will be substituted by the correct language file. If you reference a file using a full path e.g. “C:\VBV\VBVOICE.VAP”, then this phrase file will be used regardless of the language.

Custom Phrases

The language control introduces a new system phrase, type Custom.

This can be used to replace the previous custom greeting mechanisms using DLL functions for greetings and phrases with a more user-friendly approach. When a system phrase of type custom is selected, an event will be generated in the language control. The code in the custom phrase event can use the phrase properties to create new greetings based on concatenated phrases as required, in the same way as greetings are created when overriding the other system phrases.

Language Directories

The subdirectory containing the phrase files for that language must be specified in the property page. This directory must be a subdirectory of the main VBVoice directory, and must contain a copy of all the system VAP files, and any other voice files used by the system.

Properties

When creating your own system phrases, you can use most of the standard system phrases as well as phrases from VAP files and WAV files. System phrases created while in an event will not create another event - the default number implementation will always be used.

Language properties

- TimeFormat
- DateOrder
- LanguageID

All these properties define the characteristics of a language. One language can be specified in the system for the Standard edition. The professional edition includes the language control which provides simultaneous multi-lingual capability, including different rules for saying numbers, dates and times.

If there is more than one frame in the project, each frame will return the same value for these properties, and setting these properties will affect all frames.

TimeFormat (vbvTimeFormatConstants)
TimeFormat is a project wide setting that specifies whether times are said in 12 hour or 24 hour time, e.g. ‘‘Eleven twenty two P.M.’’ or ‘‘Twenty three twenty two’’.
Possible values:
Time12
Time24

DateOrder (vbvDateOrderConstants)
DateOrder is a project wide setting that defines the order in which the component parts of dates are spoken. Note that the components of the date to be spoken (day, month, year, weekday), and the format (numeric month, month name, etc.) are defined by each individual phrase.
Possible values:
ddmmyy
yymmdd
yyddmm
mmddyy

LanguageID (vbvLanguageConstants)
LanguageID is a project wide setting that specifies the concatenation rules for saying numbers. VBVoice has built-in rules for several common languages. Support for dynamic language setting, multiple simultaneous languages, and other languages can be added using the Language control.
Possible values:
English
French
Italian

Language Property Page

See also Language control

Default Number Format

These check boxes allow you to specify the default number system used by this language. At present, English, Italian and French are built-in to VBVoice. Other languages can be implemented by overriding the rules for one of these languages by checking the System Phrase Event for SayNumber, together with code in the SayNumber event.

Time Format

This selection allows the time to be said either in 12 hour (e.g. eleven fifty nine PM) or 24 hour format (e.g. twenty three fifty nine).

Setting new number formats:

The Number format fields allow you to select the default number format for each language. For instance, your language may be similar to French, with a few differences. In this case, select French as the default format, and use the Number phrase event to provide the new rules for the set of numbers that differ from French. (See Language Example)

System Phrase events can also be used to set gender for languages that have gender-specific words for numbers.

Setting new date and time

The date and time fields allow you to specify the format of phrases for dates and times used in each language. For instance, some languages may say dates as Month-day-year whereas others may use Year-Month-Day.

Language Events property page

This property page is used to select which phrases will have events fired so that VB code can modify the default system behavior for those phrase types.

See also Language control

You can override most of the VBVoice system phrases with your own implementation of concatenation rules. Your code only need implement the phrases where the rules differ from those provided by VBVoice.

Filename

See also Language Example

This property is used to specify the VAP file or WAV file containing the required phrase. Required for VAPPhraseIndex, WAVFile phrase types.

PhraseName

See also Language Example

This property is used to specify a VAP phrase using the script. When this property is set, the PhraseIndex property will contain the index of this phrase in the VAP file. Setting the PhraseIndex property directly is more efficient than using PhraseName.

PhraseIndex

See also Language Example

This property is used to specify a VAP phrase using the position of the phrase in the file. The first phrase is position 0. Setting the PhraseIndex property directly is more efficient than using the PhraseName property.

PhraseData1, PhraseData2

See also Language Example

These properties are used to specify the data to be used when saying a System phrase that requires data to say. See the example for SayNumber below, and the chapter on System Phrases for details of the data required by each phrase.

AddType

See also Language Example

AddType
Setting this property adds the phrase to the list of phrases to be played, A trappable error may be generated if the required properties for the selected phrase type are invalid. This property uses the predefined vbvSysPhraseConstants.

Phrase types	***Required properties:***
Num_Files	PhraseData1
FileCustom	PhraseData1 PhraseData2
File_Size	PhraseData1
File_Date	PhraseData1, PhraseData2
File_Time	PhraseData1, PhraseData2
SayNumber	PhraseData1, PhraseData2 optional
SayNumberShort	PhraseData1, PhraseData2 optional
Digits	PhraseData1
Money	PhraseData1
File_Date_Time	PhraseData1 PhraseData2 optional
InitialGreeting	None
Time_HrsMins	PhraseData1, PhraseData2 optional
Say_Date	PhraseData1
Time_Now	None
Date_Today	None
VAPPhrase	FileName and PhraseName
VAPPhraseIndex	FileName and PhraseIndex
Number_Ordinal	PhraseData1

Language Example

```
Private Sub French_Number(ByVal PhrsValue As String, ByVal Flags As String, UseDefault As
Integer)
'this code says '2,3,4' using a different method for each
‘digit
If PhrsValue = 234 Then
        ‘say 2 using SayNumber
        French.PhraseData1 = "2"
        French.PhraseData2 = "M" 'number flags, optional
        French.AddType = vbvSayNumber

        ‘say 3 using a named phrase
        French.filename = "vbvoice.vap"
        French.PhraseName = "three"
        French.AddType = vbvVAPPhrase

        ‘say 4 using an phrase by index
        French.filename = "vbvoice.vap"
        French.PhraseIndex = 4
        French.AddType = vbvVAPPhraseIndex
        UseDefault = False
End If
End Sub
errornumber:
debug.print "error in create phrase"
Resume Next
End Sub
```

Error Codes

ERR_CallNotHere	32005	attempt to set AddType property outside of event
Err_PhraseNotAllowed	32010	illegal phrase type
Err_AddPhrs	32011	
Err_PhraseNotFound	32012	could not find phrase in current file
Err_FileOpenError	32013	could not open file
Err_BadNotify	32014	

Properties

- GotoNode
- Result

Events

- Enter
- Exit
- VoiceError

IniSwitch

Properties Events Initial Setup Properties Methods

Default Property: Result

Property Pages

Field Setup

Conditions

Description:

Searches for a entry in a Windows initialization (.INI) file, and routes the call depending on the result found. You can set the filename, section name and field name of the entry you wish to test. The result property is set from the data found. The filename, section name, entry name and tested conditions can all contain control names. Pattern matching will be performed on the collected digits using up to 8 exit conditions in the same way as in the DataSwitch control, using find first or exact matching, and optional numeric to alphabetic translation.

Call Routing

- If the INI file entry is not found, the call is routed via the "Not Found" node.
- If the entry is found, but there are no conditions set, the call is transferred to the "Found" node. If conditions are set, but there is no match between the data field and any of the conditions, the call is transferred to the "No Match" node. If a match is found between the value obtained and one of the conditions, the call is routed to the appropriate node.
- Otherwise the call is routed to the "Default" node.

IniSwitch Field Setup property page

See also IniSwitch

Branch on

Field

Section

File
These three fields specify the Windows .INI file, the section within the file (this is the field contained within the [..]) and the field name. The INI file must be in the Windows directory or have a full pathname. The square braces should be omitted from the Section definition

Control List 

This button shows the Control List window. Any field that will accept control names or properties has a Control List button next to it.

IniSwitch Conditions property page

See also IniSwitch

Reordering conditions

Conditions are matched starting at the top and working down the bottom of the list. In circumstances where more than one condition may match, you can order the conditions by dragging and dropping the items in the condition list

Edit command
Use the Edit command, or double-click on an entry to change an exit condition using the Edit IniSwitch Conditions dialog below.

Delete command
Deletes a exit condition.

New command
Use this command to create a new exit condition.

Events

- Enter
- PhraseError
- PlayRequest
- VoiceError

OnHook control

(No Properties) Events

Description:

Plays a greeting, and hangs up the phone. Control of the line is returned to the LineGroup control that owns this channel, where a Disconnect event will occur. This event will allow termination code to run for this call.

OutCon control

(**No Properties)** **No Events**

Default Property: None

Description:

Used for connection from an output on one control to the input of another on a different form. Other controls cannot be connected across forms.

Properties

- CallerID Array String
- CallerName Array String
- CurrentControl Array Object
  HelpDigit
- LineStatus Array Integer
- MaxCallTime Array Integer
- PreviousControl Array Object
- RingsToAnswer

Operating Modes

Events

- Disconnect
- Exit
- Ring
- VoiceError
- PreDisconnect

LineGroup control

Properties Events Methods

Initial Setup Properties

Default Property: CallerId (with CallerID option)

LineGroup property page

This control 'owns' a group of telephone lines, and can receive calls and initiate calls on these lines. The channels can be set using the property page at design time, or channels can be allocated at run time via the AddChannel and RemoveChannel methods. LineGroup also contains handlers for global conditions on a channel: invalid and silence time-outs, call termination due to maximum call time or by the StopCall function, and global digit handlers.

Operating Modes

This control can operate in 3 modes. All lines must start in the mode specified by the Mode property, but the lines can be changed at runtime using the StartCall, and WaitForRings methods.

Idle
The control does not respond to Ring events or perform any action until the mode is changed.

Wait for ring
In this mode, the control waits for the number of rings set by RingsToAnswer to be received. It then takes the line off-hook and passes the call to the next control. A Ring event is generated in the control before the line goes offhook to allow your code to perform call initialization and/or to override normal behavior. If CallerId and/or CallerName is available from the hardware, it is accessible from the CallerId and CallerName properties. If RingsToAnswer is set to zero (blank) the control will not detect ringing.

Start call after delay
In this mode a call is started on each channel allocated to this control as soon as the system starts, or when the previous call on the channel completes. A delay is used after the termination of the previous call to ensure that the line is detected as on-hook by the telephone system before starting the next call. If ringing is detected during the delay time, the LineGroup control will wait 20 seconds for the line to clear before starting the next call. The next voice control in the chain will normally be a Dial control to take the line off-hook and dial a number.

LinesInGroup

A string containing the lines in this group. The lines in a group can be changed before the system is started by setting this string. After the system is started, the line configuration can be changed using the AddChannel and RemoveChannel methods.

Methods

AddChannel (channel as integer)

RemoveChannel (channel as integer)

MaxCallTime (Channel as Integer)

StartCall (channel as integer)

StopCall (channel as integer)

TakeCall

WaitForRing (channel as integer)

AddChannel (channel as integer)

Adds another channel to the channels allocated to this control. If the channel is already allocated to this control, the command is ignored. Channel is initially in the state defined by the Mode property.

RemoveChannel (channel as integer)

Remove a channel from the list of channels allocated to this control. Generates a trappable error if the channel is not allocated to this channel, or if the channel is currently active.

MaxCallTime (Channel as Integer)

This property can be used to set a maximum time for a specific channel. The new time will affect only the current call on the channel. If a call is not in progress at the time this property is set, it has no effect. See IMaxCallTime.

StartCall (channel as integer)

Starts a call by transferring the call to the control connected to the StartCall output. This will normally be a Dial control which will take the line offhook and start dialing.

StopCall (channel as integer)

When this method is called, the channel behaves in the same way as described in the MaxTime property. This method can be invoked at any time during a call. See IMaxCallTime above for a description of system behavior when a call is stopped.

WaitForRing (channel as integer)

Causes the channel to wait for the number of rings as defined by RingsToAnswer. When the requested number of rings occurs, Ring event will fire, and then the line will normally be taken off hook and call started with the control connected to the Ring output.

IdleChannel Method

This method sets a channel idle, if it is currently waiting for rings. If the channel has a call active, no action will be taken - but the channel will go idle after the call completes.

CurrentControl (channel as integer)

Provides the control that currently is processing the call on the channel. This property is read-only. This property can be used to access properties within the control - for instance CurrentControl(channel).Digits(channel) if the control is a GetDigits control. Visual Basic will generate a trappable error (method or property not supported) if the control does not support the requested property.

LineStatus (channel as integer)

Returns the status of the channel. Possible values are:

"idle"	0	"stopped - call complete"	12
"playing greeting"	1	"pre-call delay"	13
"waiting for digits"	2	"waiting for ring"	14
"recording"	3	"inter call delay"	15
"playing delay recording"	4	"waiting for shutdown"	16
"dialing"	5	"waiting for dialtone"	17
"unexpected event"	6	"waiting for channel stop"	18
"delay"	7	"in conference"	19
"stopped - error"	8	"getting conference digit"	20
"stopped"	9	"stopped - max errors"	21
"caller hung up"	10	"stopped - no connection"	22
"stopped - caller hung up"	11	"paused - waiting for digit"	23

NOTE!!!: this list is subject to change prior to full release

PreviousControl (channel as integer)

Provides the control that was processing the call on the channel, prior to the current control. Primarily useful when the call has been stopped by a MaxCallTime timeout, and the status prior to the timeout is required. This property is read-only. This property can be used to access properties in the same way as the CurrentControl property.

RingsToAnswer Property

Applies to LineGroup

Defines the number of rings required before answering a call. Defines the number of rings required before answering a call. May be set to 0 (blank) to inhibit call answering.

Example
To set the RingsToAnswer:
LineGroup1.RingsToAnswer = 5

CallerId, CallerName Properties

Applies to LineGroup

Provides the calling number and name as received by the CallerId hardware or the voicecard, if available. The property can also be set by VB code if the caller ID has been provided by an external device.

Example
To access the caller id:
mystring = LineGroup(channel).CallerId(channel)

Disconnect Event

Applies to All Controls

When a call that was initiated by this control is terminated, for whatever reason, a Disconnect event occurs in this control. The Disconnect event occurs just before the system hangs up the phone. This allows call termination code to run.

```
Sub LineGroup1_Disconnect (Channel As Integer, Reason As Integer)
```

The reason for disconnect can be one of the following constants:

```
Global Const CONTROLHANGUP = 0     ‘ the onhook control hung up the line
Global Const SYSERRORHANGUP = 1  ‘a system error occurred
Global Const CALLERHANGUP = 2                ‘the caller hung up: a loop current drop or
                                                        'other disconnect indication was received
Global Const INVALIDHANGUP = 3        ‘invalid digit or silence timeout handler was invoked
```

Ring Event

Applies to LineGroup

The Ring event occurs when the number of rings specifed by RingsToAnswer have been received, but before the LineGroup control takes the line offhook. A Ring event can occur regardless of which mode the control is in, but if RingsToAnswer is set to 0, no ring events will occur. If *NoOffhook* is set to TRUE, the phone will not be taken offhook. The default is FALSE - the line is taken off hook. If the line is taken off hook, the call exits out of the Ring output. Otherwise the call is ignored.

Sub ControlName_Ring(Channel as Integer, NoOffhook as Integer)

LineGroup property page

See also LineGroup

Lines in Group (LinesInGroup property)
Specifies the channels to be allocated to this control. This field can be left blank if you want to allocate lines at run-time, or set to a list of numbers delimited by commas, or a set of numbers separated by hyphens. To add channels at run time, either use the LinesInGroup property before the system has been started, using the same syntax as described here, or the AddChannel and RemoveChannel methods after the system has been started.
Example:
1-9
1,2,6-8,11

Channel Mode (mode property)

Answer Call
The LineGroup control will wait for the number of incoming rings specified in 'Rings before Answer', then go offhook and transfer the call to the control connected to the 'Ring' node.

Setting the mode at run time from WaitForRings to Idle will not stop the channel if it is currently waiting for rings. To re-initialize the channel, call the StopCall function. When a call completes, the mode is checked for the new status, and the channel will revert to Idle.

Start call after delay
The LineGroup control will start a call and transfer it to the control connected to the 'StartCall' node. After the call completes, the control will wait for the period defined in Delay Time ?? and then start another call. This mode is useful for out-dialing applications that iterate through a list of numbers. When dialing is complete, the mode can be changed to Idle.

Line Type (LineType property

Loopstart
This is the normal analog telephone line supplied to single line telephones.

Digital
Set this when using a digital interface card. Each supported digital card will have its own setup procedure.

Loopstart DID
Use this line type to have VBVoice automatically collect incoming digits from DID lines when using an external DID trunk connection device such as those from Exacom and VoicePRO Systems. The digits are collected before the line is taken offhook. The digits will be available in the CallerID property. For systems that provide the digits after the line has been taken offhook, use a GetDigits control connected to the Ring output, with a short 1 or 2 second timeout.

Rings before answer (RingsBeforeAnswer property

Specifies the number of rings to be received before the LineGroup control will answer the call.

Delay between calls (DelayTime property)

When in *Start call after delay* mode, specifies the number of seconds to wait from the end of one call to starting the next. Normally 4 seconds should be used to ensure that the call is cleared and available for use before starting the next call.

Maximum call time (IMaxCallTime property)

When set to a non-zero value, interrupts a call at the time specified and reroutes it to the StopCall output.

Help Digit (HelpDigit property)

(HelpDigit property).
This field can be set to a DTMF digit. If this digit is detected in any control, the call will exit the current control and will enter the control connected to the HelpDigit output of this control. This provides a global handler for one specified digit, useful for providing a help digit. The help digit handler provided here can be overridden using the Error Handler property page in individual controls, and the Disable Help Digit field in the GetDigits control.

Help Digit

(HelpDigit property).
This field can be set to a DTMF digit. If this digit is detected in any control, the call will exit the current control and will enter the control connected to the HelpDigit output of this control. This provides a global handler for one specified digit, useful for providing a help digit. The help digit handler provided here can be overridden using the Error Handler property page in individual controls, and the Disable Help Digit field in the GetDigits control.

HelpDigitControl property

(HelpDigit property).
This field can be set to a DTMF digit. If this digit is detected in any control, the call will exit the current control and will enter the control connected to the HelpDigit output of this control. This provides a global handler for one specified digit, useful for providing a help digit. The help digit handler provided here can be overridden using the Error Handler property page in individual controls, and the Disable Help Digit field in the GetDigits control.

DisconnectControl property

This property can be set to the name of a VBVoice control. When a caller hangs up while the call is in a control, the control specified by this property will continue processing the call. If this property is not set, then the normal caller hangup processing events will occur.

See Error Handling

InvalidErrorControl property

This property can be set to the name of a VBVoice control. When a caller enters an invalid digit or digits which increments the retry count up to the maximum number of retries, the control specified by this property will continue processing the call. If this property is not set, then the normal caller hangup processing events will occur.

See Error Handling

NoDigitsErrorControl property

This property can be set to the name of a VBVoice control. When a caller fails to enter a digit, which increments the retry count up to the maximum number of retries, the control specified by this property will continue processing the call. If this property is not set, then the normal caller hangup processing events will occur.

See Error Handling

Runtime Properties

Action Array Integer

GotoNode

MsgData String

MsgFile Array String

MsgIndex Array Integer

MsgType Array Integer

NumNewMessages Array Integer

NumOldMessages Array Integer

OptionDigit Array Integer

Initial Setup Properties

ClearDigits BOOL

DeleteDigit

DisconnectControl String

HelpdigitControl String

ExitDigit

FastForwardDigit

FastFwdSecs

ForwardDigit

IMaxSil

InfoDigit

InvalidErrorControl String

LouderDigit

MailboxDirectory

NoDigitsErrorControl String

PauseDigit

PreviousMsgDigit

ReviewDigit

RewindDigit

RewindSecs

SaveAsNewDigit

SkipDigit

SofterDigit

WaitForOption

For more information see Option Digit Handling and PlayMsgs Options property page

MsgFile property

The filename of the current message. This is the path provided by the Mailbox directory property, with the current message file name concatenated on the end.

MsgIndex property

The position of the current message in the list. This property is used by the MsgGreeting to say the message number

MsgType property

The type of the current message - uses the vbvMsgTypeConstants. It is used in the MsgGreeting to play either 'New', or 'Old'

NumNewMessages property

Returns the number of new messages in current mailbox when the control was entered. It does not reflect changes made by the caller while accessing the mailbox. This property is used by the default EntryGreeting to play the number of new messages available.

NumOldMessages property

Returns the number of old messages found in current mailbox when the control was entered. It does not reflect changes made by the caller while accessing the mailbox.

OptionDigit property

Used in conjunction with the custom digit handler output to access the last option digit pressed.

Greeting Properties

EntryGreeting

InfoGreeting

InvalidGreeting

MsgGreeting

NoDigitsGreeting

OptionsGreeting

There are three special greetings used in this control. The VAP file PLAYMSGS.VAP contain some special phrases used by these greetings. The phrases are indexed by position, rather than name, so the order of the phrases should not be changed. These phrases are:
“old”
“new”
“deleted”
“you have no messages
“please enter the box number to forward to”
“please record an introduction for this message”
“message forwarded”
“message”
“saved as new”
“you have no more messages”

EntryGreeting.
The default entry greeting uses PlayMsgs property NumNewMessages to inform the caller of the number of new messages. The special control name %this% is used to refer to the control (allowing the default prompt to work regardless of the actual control name).

InfoGreeting
This greeting is played when the caller presses the ‘Info’ digit, if configured.

InvalidGreeting.
This prompt will inform the caller that they have pressed an incorrect digit. The InvalidGreeting and the OptionsGreeting will then be played again, until the error count is exceeded. The default greeting is “that was not a valid entry”.

MsgGreeting
This greeting is used to play the message for the first time. On replay, fast forward, rewind operations, just the message file itself is played. Default greeting says ‘Message one, new <message file>’ where the ‘one’ refers to the index of this message in the list, and ‘new’ refers to it’s status. The ‘old’ and ‘new’ phrases are selected from the PLAYMSGS.VAP file using the MsgType property to index into the file to select the correct phrase.

NoDigitsGreeting
This greeting is played if no digit is pressed during or after OptionsGreeting. The NoDigitsGreeting and the OptionsGreeting will then be played again, until the error count is exceeded. The default NoDigitsGreeting greeting is empty.

OptionsGreeting:
This will normally say "Press 1 to save your message, 2 to save your message as new, 3 to delete your message, 4 to hear the next message, or 8 to exit". This greeting should be changed to match the digit options you have set.

Events

- Disconnect
- Enter
- Exit
- MsgUpdate
- OptionDigit
- PhraseError
- PlayRequest
- VoiceError

PlayMsgs control

Properties Events Greeting Properties. Initial Setup Properties Methods
Default Property: None

Property pages

Mailbox, Entry conditions

Digit Options

Error Handlers

A high level control that plays messages from a voice mailbox. It is the main building block for implementing voice mail systems using VBVoice.

Operation
Two types of messages are supported, New and Old. New messages are those that have not yet been heard, or have been saved as new. New messages are converted to Old once they have been heard. When a call enters a PlayMsgs control, it first plays the number of each type of message. (configurable in the EntryGreeting). It then plays the messages in sequence, starting with the new messages (oldest messages first), using the configurable MsgGreeting. The action after each message is configurable to either start playing the next message immediately, or to play the OptionsGreeting, allowing the caller to save, delete, forward or skip the message, or to exit. After both new and old messages have been played, the call exits the control. Once a new message has started playing, it is changed to an old message unless the caller selects one of the option digits for save-as-new, delete, skip message, or hangs up before the end of the message.

& Entry node
this input can be used to re-enter the control without changing the state of the control. For instance, if you provide a custom digit handler using the OptionDigit event, you can provide a special menu and actions to the caller, after which they can re-enter the PlayMsgs control in the same state as it was then the call left the control. In this case, the EntryGreeting is not played, but the MsgGreeting for the next message is played.

Exit node 0 ‘Exit’
This output is used when the PlayMsgs operation terminates normally, either all messages have ;been played, or the caller pressed the exit digit

Exit node 1 ‘Custom digit handler’
When a digit is pressed for which the PlayMsgs control does not have a valid action, it fires the OptionDigit event. If VB code sets the Accepted parameter to True, the call exits out of this output for special handling. The PlayMsgs control can be re-entered via the & input for resumption of message play at the original message, or the file specified by the MsgIndex property if set.

Message Database

The PlayMsgs control uses a directory-based system to identify message files to play. Each box number has a directory in which voice files are stored. Message types (old, new and deleted) are identified by the extension of the file.

Message Data

Each file recorded by the Record control can store a Data field within the file. This message data can be retrieved from the file before being played and can be used to modify the greeting, or put to other uses. The data is accessible from the MsgData property.

Directory Checking

The mailbox directory should be validated before the PlayMsgs control tries to access the message file. Validation can be achieved using a separate mailbox database of valid mailboxes, which can be used to check the mailbox number before entering the PlayMsgs control. This table can also contain the

password and other mailbox parameters. This is the method used by the sample voice mail form in the VOICMAIL directory. See the voicemail example code and description (page ?) for more details of how to use the PlayMsgs control.

Option Digit Handling

Delete (Delete Digit property)
Marks the current message as deleted. The file is not erased but the file extension is changed to ‘‘.dlt’’. An off-line utility (not provided) is responsible for deleting files. This allows for the possibility of undeleting ‘deleted’ messages.

Exit (Exit Digit property)
Exits the PlayMsgs control.

Fast Forward (FastForwardDigit property)
Jumps ahead in the message by a configurable amount.

Forward (Forward Digit property)
Forwards the message to another box.

Info (InfoDigit property)
Plays the InfoGreeting (by default, the time and date of the message)

Louder (LouderDigit property)
This digit will increase the volume of the playback by one increment. There are 20 volume increments, each one approximating to a doubling of the volume.

Softer (SofterDigit property)
This digit will decrease the volume of the playback by one increment..

Pause (PauseDigit property)
Pauses playing of the message for up to 20 seconds.

Review (ReviewDigit property)
Replays the current message from the beginning.

Rewind (RewindDigit property)
Jumps back in the message by a configurable amount.

Save as New (SaveAsNewDigit property)
Marks the current message as a new, unheard message, by changing the extension of the file to ‘New’.

Skip (SkipDigit property)
Skips to the next message. The status of the current message is not changed.

Invalid digit, No digit handling

If an invalid option digit is received, VBVoice will first fire the OptionDigit event. In this event, you can use code to exit to a new control using the TakeCall
method. This new control can provide custom menus for your caller, and when completed, you can re-enter the PlayMsgs control using the & input to resume message processing.
If no action is taken, the error count will be incremented, and the control will play the InvalidGreeting followed by the OptionsGreeting. If no digits are received, VBVoice will increment the error count, and play the NoDigitsGreeting followed by the OptionsGreeting.
When the error count exceeds the maximum error count, VBVoice will continue with default error handling.

EntryGreeting

See also Customizing Greetings

The default entry greeting uses PlayMsgs property NumNewMessages to inform the caller of the number of new messages. The special control name %this% is used to refer to the control (allowing the default prompt to work regardless of the actual control name).

PlayMsgs Mailbox, Entry Conditions property page

See also PlayMsgs

Mailbox directory (MailboxDirectory property)
This specifies the directory containing the messages to play. This will normally be the same path as used in the record control to take messages, but without the appended filename. It will normally contain the name of a GetDigits control which has collected the mailbox to use, in addition to a partial directory name.

After playing message: (WaitForOption property)
These options specify whether the PlayMsgs control will play the options greeting, and wait for an option digit after each message, or if it will start playing the next message immediately. A digit pressed during the prefix to the message, as set up in the MsgGreeting, will be applied to the previous message, not the new one.
For instance, if the default MsgGreeting is used, the first prompt might be:
'message'
'one' (message number)
'new' (message type)
'hello this is John. Please call' (the actual message)
Then it will start playing 'message two, new'. If the digit 2 (replay) is pressed during the 'message two, new', the system will replay 'hello this is John. Please call' - the previous message.

PlayMsgs Digit Options property page

See also PlayMsgs

Play Option Digits

These selection boxes allow you to specify which options should be available to the caller at the end of playing each message. To disable the option, set to digit to ‘none’. The OptionsGreeting should be set to match the options specified. See *Option Digit Handling*.

Forward option:

System prompts the caller for a mailbox to forward to (a fixed phrase in PLAYMSGS.VAP). The number of digits collected will be the same number as was collected in the mailbox field in the Directory field. The caller will then be prompted to record a prefix to the message. If recorded, this is prepended to the new message, which is put into the destination mailbox.

OptionDigit

This property can be used after a call has exited the Record control, to access the last option digit entered by the caller.

Invalid

This event occurs when the caller enters a digit which is not a valid option digit. VB code can choose to transfer the call to another control based on the digit selected (using the GotoNode, Gotocontrol properties) or allow the Record control to treat it as an invalid digit in the normal way. The digit is passed to the procedure allowing your code to decide what to do.

Sub Record_Invalid (Channel As Integer, Digit As Integer)

Properties

- Filename
- FileSize Array String
- GotoNode
- Maxsil
- MaxTime
- MsgData Array String
- OptionDigit
- TermDigitMask Integer

Initial Setup Properties:

AppendDigit

AppendFile

BeepBefore

CancelDigit

ClearDigits BOOL

CompressSil

DisconnectControl String

HelpdigitControl String

IgnoreDtmf

IMaxSil

IMaxTime Integer

IMinTime

InvalidErrorControl String

NoDigitsErrorControl String

RecordFormat

RerecordDigit

ReviewDigit

TermDigit

See Record Setup Property Page for more information

Events

- Disconnect
- Enter
- Exit
- Invalid
- PhraseError
- PlayRequest
- VoiceError

Record control

Properties Events Methods
Greetings

Default Property: None

Property Pages

Setup

Digit Options

Error Handlers

Description:

Records a wave file into the filename specified by either creating a new file or appending an existing one. Each message can be stored in a unique file name generated by the control. An options menu is provided to provide message review, re-record and deletion.

Record Filename format

The filename used for each message can be a unique name created by the Record control, and can also contain values from previous results in other controls. A wild char '*' can be used to indicate that a unique file name should be created. The Record control does this by replacing the * with 8 random characters when a call arrives at the control. If a * is not specified, the same file will be overwritten on every record operation. If the AppendFile option is specified, the file will be appended rather than overwritten.
To create a filename that is dependent on the channel, use the %channel% variable. (see control property substitution). This will be replaced by the current channel number when the recording starts.
If a full path for the file is not specified, the VoiceDirectory will be used. Any language-dependent directories will be ignored. However you can override this by adding the VBVFrame.CurrentDirectory property into the file string, e.g. %VBVFrame.CurrentDirectory%msg*.wav

Digit Options

After making a recording, the control will play the MessageRecordedGreeting, followed by the OptionsGreeting if there are any option digits set, and will then wait for a digit. If no option digits are set, the control will exit via the Done output after playing the MessageRecordedGreeting.

MsgData

The Record control can associate a data string with each message recorded. See MsgData property. This data is accessible from the PlayMsgs control for use in greetings etc.

Error handling

If an invalid option digit is received while waiting for a option digit, and code in the Invalid event does not process the digit, VBVoice will increment the error count, and play the InvalidGreeting and the OptionsGreeting again.
If no speech was detected during the record operation, VBVoice will increment the error count, play the SilenceGreeting and EntryGreeting, and restart the record operation.
If no digits are detected while waiting for an option digit, VBVoice will treat the condition as an invalid digit.
If the error count exceeds the maximum error count on an invalid digit or timeout, the standard error processing sequence is followed. See Error Handling.

MsgData

The Record control can associate a data string with each message recorded.
It does this by filling in the INFO section of the Wave file. The info section has a maximum size of 256 bytes. The default data string is set using the property page item ‘File Data’. This data can be changed in the Enter event.
This data is accessible from the PlayMsgs control for use in greetings etc

FileSize

This property contains the size (in bytes) of the file recorded by the Record control. If no file has been recorded, the value will be 0.

MaxTime

This is the maximum duration (in seconds) allowed for the recorded message. This property is set by the control to the value *Max time for Record* (IMaxTime property)in the setup dialog. It can be changed in the Enter event to another value if required using this property. The new value only affects the current call.

Example

To override the maximum time setting:

```
If CallerName = “Fred” Then
Record1.MaxTime(channel) = 5      ‘stop after 5 seconds silence
Endif
```

OptionDigit

This property can be used after a call has exited the Record control, to access the last option digit entered by the caller.

TermDigitMask

This property can be set in the Enter event to disable certain Record options for different callers.
DigitMask is an integer, made up of the vbvDigitMaskConstants.
Examples:

The value vbvMaskAll will enable all options as configured in the setup dialog.
The value 0 will disable all options.
The value vbvDigit_7 will enable the digit 7 for whatever option is set for it in the setup dialog

Different vbvDigit_ values can be or'ed together to enable and disable different option digits. This property can be set in the Enter event to assign different options to different callers

Example

To override the digit mask setting:

```
If CallerName = “Fred” Then
Record1. TermDigitMask(channel) = vbvDigit_1 OR vbvDigit_3
Endif
```

Greetings

The Record control uses prompts stored in the file RECORD.VAP by default. All phrases can be reconfigured.

EntryGreeting
This will normally be used to prompt the caller to leave a message, for instance: "Please leave a message at the tone. Press the # key or wait for further options, or simply hang up".

OptionsGreeting
This prompt is played after a valid recording has been made and the control has played the MessageRecordedGreeting. The prompt will provide a list of which match the digit options that you have set. If no options have been set, it can be left blank. If no options have been set, this greeting is not played, and the control exits out of the Done output immediately.

InvalidGreeting
This prompt will inform the caller that they have pressed an incorrect digit, or if they do not press a digit during the Options greeting. The OptionsGreeting will then be played again. The default invalid greeting is "that was not a valid entry".

SilenceGreeting
If no voice is recorded after the Entry greeting, this prompt is played. It might say "I'm sorry, I did not hear your message". The EntryGreeting is then played again and another attempt at recording is made, until the error count is exceeded. The default silence greeting is empty.

MessageCanceledGreeting
Played after the cancel digit has been pressed. The call then exits out of the Cancel output.

MessageSentGreeting
Played after the terminate and save digit has been pressed. The call then exits out of the Done output. This is played when the *Terminate and Save* option digit is received.

MessageRecordedGreeting
Played after a valid recording has been made, and before the OptionsGreeting.

Filename

Applies to Record

Array String (1, MAXCHANNELS)

The filename created is available from the Filename property. Your VB code can override the filename by setting the property in the Enter event. Changing the filename at any other time will have no effect. The filename can also be used as part of a greeting in order to replay the message. If the message is canceled or not recorded, the filename property will be empty.

Example 1

To access the filename:

mystring = Record1.Filename(channel)

Example 2

To override the filename:

Record1.Filename(channel) = “myfile.tmp”

MaxSil

Applies to Record

Array Integer(1, MAXCHANNELS)
This is the duration (in seconds) of silence before VBVoice decides that the caller has stopped talking. This property is set by the control to the value Maximum Silence in the Setup property page. It can be changed in the Enter event to another value if required. The new value only affects the current call.

MaxTime

Applies to Record

Array Integer(1, MAXCHANNELS)

This is the maximum duration (in seconds) allowed for the recorded message. This value is set by the control to the value Maximum Silence set in the setup dialog. It can be changed in the Enter event to another value if required. The new value only affects the current call. a

Example

To override the maximum time setting:

```
If CallerName = “Fred” Then
Record1.MaxTime(channel) = 5      ‘stop after 5 seconds silence
                                  ‘Fred is a slow talker
Endif
```

Record Setup property page

See also Record

Record Start

Filename (IFileName property)

Enter a file specification here. Record will create a file based on this specification to record the message. See Record filename format.

File Data (FileData property)

Default message data to be associated with this file. This data can be used in the PlayMsgs control to associate a message with a particular caller, or any other information you wish to store with the message. For instance, in a system with caller-id available, you could add %LineGroup1.CallerID% to this field. In a PlayMsgs control, add to the InfoGreeting a system phrase SayDigits %this.MsgData% to say the data to the caller.

Don’t record silence (CompressSilence property)

If checked, the voice card will remove all silence periods greater than a predefined size. This is set to a default size of 200ms, and can be changed using a setting in the CPL file.

Control List 

This button shows the Control List window. Any field that will accept control names or properties has a Control List button next to it.

Maximum time for record (IMaxTime property)

This fields sets the maximum time for a recording, in seconds.

Max silence (IMaxSil property)

This field sets the maximum period of silence before terminating a record operation.

Minimum time for valid recording (IMinTime property)

A recording shorter than this time will not be regarded as valid. The file will be erased and will be treated as a silence timeout.

Record Digit Options property page

See also Record

These selection boxes allow you to specify which options should be available to the caller after recording the message. If set to ‘none’, that option will not be available. The OptionsGreeting should be set to match the options specified. The options available can be modified at run time using the TermDigitMask property.

Rerecord message (RerecordDigit property)
The record process is restarted, with no prompt. The beep is played before starting the record again.

Cancel (Cancel Digit property)
The MessageCanceledGreeting is played, and the call exits out of the ‘NoMsg' output.

Review (Review Digit property)
The recorded message is played back to the caller. The OptionsGreeting is then played again.

Append (Append Digit property)
The caller is prompted to start recording again, and the resulting recording is appended to the existing recording. The options prompt is played again.

Terminate and Save (TermDigit property)
This digit will terminate the record process, save the file and exit the control via the ‘Done’ node.

Properties

GotoNode

Events

- Condition
- Enter
- Exit
- VoiceError

TimeSwitch control

Properties Events Methods

Default Property: Decision

TimeSwitch Property page

TimeSet Dialog

Description:

Transfers the call to another control according to time of day and day of week. Several timezones can be set for each control. If the current time and day of week do not match any of the fields specified, the call continues via the default connection.

TimeSwitch property page

See also TimeSwitch

The Time Periods property page shows the current list of valid times for this control. Use the Delete, New and Change buttons to change this list, or double-click on an entry to change it.

Time Set dialog

See also TimeSwitch, TimeSwitch Property page

The Time Set dialog allows you to create new and modify existing time specifications. Each entry specifies a time range in a set of days. Times are specified using 24 hour time.

Time periods should not span over the time 12.00 midnight. For example,
03:00 to 23.59 is OK
00:00 to 02:59 is OK
23:00 to 00:10 is incorrect: split this period into 2 times, 23:00 to 23:59 and 00:00 to 00:10

DisconnectControl String

HelpdigitControl String

InvalidErrorControl String

NoDigitsErrorControl String

Properties

- GotoNode
- Value

Events

- Enter
- EnterB
- State01 - State10
- PhraseError
- PlayRequest
- VoiceEvent
- VoiceError

State01 - State10

Applies to User

When an event is received from the voice card driver, either a VoiceEvent or a State<xx>. Event is generated. If the state machine variable is 0, a VoiceEvent will occur. If the state variable is between 1 and 10, a State<xx> event will occur.

VoiceEvent

Applies to User

If there was no greeting set, a VoiceEvent event occurs immediately after the Enter event, with the eventcode set to EVT_EOF. If there was a greeting set, the VoiceEvent occurs after the play greeting terminates. It can terminate either due to a digit being received (EVT_MAXDT), or due to the end of file (EVT_EOF).
Since the User control itself does nothing except provide the events, our code must either call another voice card driver function to generate another event, or it must transfer the call to another control using the GotoNode or GotoControl properties.
Use the function vbv_setstate to set the state machine variable to a new value. The state machine is initially set to 0 when a call enters the User control. The state variable defines which event procedure will get called on the next voice card driver event.

WARNING!
If no voice card multi-tasking function is called, and the call is not transferred to another control, the channel will hang in the User control. This property of the User control can be useful when say conferencing a call when you want the caller to wait. When ready to restart the call, generate an event in the control by calling vbv_queue_event for the correct channel. This function can be called at any time if the channel is idle. In the event procedure, you can set the GotoNode or GotoControl properties to start processing of the call again.

User control

Properties Events

Default Property: Value

Description:

This control has no predetermined action. It allows Visual Basic to define all operations and provides complete control over the voice card. Examine the project in the USERDEMO directory and the description below to see an example of how to use this control.

Events

Enter and EnterB events are generated when control passes to the main input or input B. VB Code can call any driver function, and is responsible for transferring the call to another control when required using the GotoNode and GotoControl properties.

There is no Disconnect event in this control: the Voice event must detect this condition and hang up the call if required.

State Machine support

The User control has built-in support for state machines. Each User control has 10 built in event functions which can be used to create a state machine. To use the state machine, call the vbv_setstate function in the VoiceEvent event, with the state parameter set to 1, and then call a voice function. When the function completes, the State01 event will occur. You can now call another voice function, and use vbv_setstate to set the state to 2. When this is complete, the State02 event will occur - and so on. If you run out of states, you can transfer to another control, or alternatively create your own state variable, and use the Select statement to branch depending on the state. See Events below for more information.

UserDemo example

Error handling

There are no predefined errors in this control. VB code must handle all events, and can invoke the default error handler, silence handler or caller hangup handler using VBVoice functions.

Why a state machine?

State machines are often required in voice processing systems since most voice functions do not wait until the operation is complete before returning to the calling program. The program must wait until it has received an event indicating that the operation is complete before continuing. A state machine is a way for the program to remember what it was doing between events, and to handle multiple channels at the same time.

Value

Applies to User

Array String(1, MAXCHANNELS)

The Value property can be set by code, and used subsequently in other controls to configure greetings and in other fields that accept Control result values.

Example
To set the User value property:
User1.Value(channel) = “this is the result”

GotoNode

Applies to All controls

Array Integer(1, MAXCHANNELS)

Setting this property to a valid node number in the control where the event occurs will route the call to the control connected to that node. Node numbers start at 0, numbered from the top.
For instance, if GotoNode is set to 0, the call is passed to the control connected to the first (top) node. The transfer takes place as soon as the Enter event code is completed. GotoNode can also be set to one of the *vbvExitNodeConstants* constants to invoke the default invalid digit, no digits or caller hangup handlers respectively

An error will be generated if the node number is invalid. If the node is not connected, processing will continue, but a VoiceError event will be generated when VBVoice attempts to transfer the call, and default error handling (see p ?) will ensue.

Example

To transfer to the control connected to the fifth output from the top, node 4:

```
GetDigits1.GotoNode(channel) = 4
```

See also GotoControl

Example

To set the next node:

```
GetDigits1.GotoNode(channel) = 4
```

GotoControl

Applies to All controls

Array String(1, MAXCHANNELS)

Obsolete. Use the TakeCall method

Event handlers

VoiceError Event

Applies to All controls

This event occurs if a non-fatal error occurs in the control. A non-fatal error is one where an unexpected situation has occurred and the control does not know how to deal with it, but does not necessarily mean a hardware malfunction. Normally the call will be terminated if the error is not processed. This event allows VB code to intercept the error and decide whether to allow the call to continue.
When an error event occurs, the control generating the event is not normally able to continue normal processing. Your code can choose to ignore the error by setting the processed parameter = True, or by redirecting the call to a new control using the GotoNode or GotoControl properties. If none of these actions are taken, the call will be terminated.

Sub VoiceError(Channel as Integer, ErrorType as Integer, ErrorData as Integer, Processed as Integer)

Examples of errors include invalid filenames in greetings and record filenames.
TODO: check these error codes

Error types:	**Data**	**Description**
UNEXPECTED_DRIVER_EVENT	the event code	The control received an unexpected event from the voice card driver

PhraseError Event

Function prototype:

Sub PhraseError(ByVal Channel as Integer, Phrase as Object, ByVal ErrType as Integer)

ErrType is one of the predefined vbvPhraseErrorConstants

vbvFileOpenError = 0,
vbvPhraseNotFound = 1,
vbvControlNotFound = 2,
vbvBadTimeSpec = 3,
vbvBadNumber = 4,
vbvBadDate = 5,
vbvPhraseNameNotFound = 6,

Your code can do one of the following:

- replace the phrase in error with a new phrase
- ignore the phrase
- allow the default action to occur - which will terminate the call.

To ignore the phrase:

Set the Phrase parameter to zero.

To insert a different phrase in its place:

Modify the phrase using Phrase methods and properties. To avoid infinite loops, if an error occurs while processing the replacement phrase, an event is not generated.

PlayRequest Event

This event occurs when a greeting contains a VBPhrase. Each VB Phrase is named so that it can be identified in the situation where a control has more than one VB Phrase. In the PlayRequest event procedure, your VB code should create a phrase object using the Phrase methods and properties, and return it in the PhraseObject parameter, or return 0 to skip the phrase. This is used to create custom greetings which are not fixed at design time, but must be determined at run time,
Other ways of dynamically configuring greetings are by selecting a VAP phrase by index or name, which is generated from a VB variable (via a VBVFrame Transfer property) or from another VBVoice control such as a GetDigits or DataFind. See Customizing greetings at runtime.

Sub PlayRequest(ByVal Channel as Integer, ByVal PhraseName as String, Phrase as Object)

Enter, EnterB Event

Applies to All controls (except LineGroup)

One of these events occur when a call enters a control, depending on the input by which the call arrives. The EnterB event occurs in controls that have 2 input nodes, when the call enters via the second (lower) input. The Enter events give VB code an opportunity to change some of the characteristics of the control by setting its runtime properties on a call-by-call basis, or to bypass the control altogether using the GotoNode property or TakeCall method.

The footprint of the Enter event in voice processing controls that have an Entry greeting is different than other controls: they have an additional parameter, Greeting as Object. This parameter is the greeting object to be played on entry to the control. The Enter event can change this greeting as required. Note that changing this greeting will only affect the greeting for that call at that instant only. Subsequent calls, or reentry into the control by the same call, will use the greeting as defined at design time unless this is also changed. See Custom Greetings .

Voice processing controls:

Sub xx_Enter ByVal Channel as Integer, Greeting as Object)

Other controls:

Sub xx_Enter(ByVal Channel as Integer)

Sub xx_EnterB(ByVal Channel as Integer)

Exit Event

This event occurs when a call leaves a control, and before the Enter event occurs in the destination control. The node parameter is the node number which indicates the destination control, unless the exit event is due to a no digits error, invalid digit error, caller hangup, or help digit detected, in which case the node value is one of the vbvExitNodeConstants below. Nodes are numbered from 0.
The exit node can be changed in the event procedure to override the default behavior. If a caller hangup is detected, the Exit event occurs first, followed by the Disconnect event sequence.
If an error occurs while the call is being handled in the control, the VoiceError event occurs instead.

vbvExitNodeConstants
vbvNoDigits = -1,
vbvInvalidDigits = -2
vbvCallerHangup = -3
vbvHelpDigit = -4,

Sub xx_Exit (ByVal Channel As Integer, Node As Integer)

Disconnect Event

Applies to All controls
See also PreDisconnect Event

This event procedure is called when a caller hangs up. It can occur in any control that does voice processing. This event allows clean-up on a per-control basis. After this event occurs, a Disconnect event occurs in the LineGroup control that initiated this call. This hierarchy of events allows localized clean-up procedures and/or global clean-up as appropriate. The global Disconnect event in the LineGroup control occurs regardless of whether the caller hangs up first or the system hangs up the call. The local Disconnect event will occur in most controls when the caller hangs up - the only exceptions are controls that terminate a call - the Onhook control, and the Dial control after a successful blind transfer.

Notes

The Disconnect event does not occur in the LineGroup control when in test mode.

There are possible reasons for hangup are listed below:

Sub xx_Disconnect (ByVal Channel as Integer, ByVal Reason as Integer)

Possible disconnect reasons are defined by the vbvDisconnectConstants enumeration. These constants are defined in the OCX and can be used directly.

vbvControlHangup = 0 the call terminated at an Onhook control
vbvSysErrorHangup = 1 the call terminated due to a system error in a control
vbvCallerHangup = 2 the call terminated due to the caller hang-up indication from the telephone switch
vbvInvalidHangup = 3 the call terminated due to default invalid digits or no digits timeout handling
vbvSysStopping =4 the system is stopping due to the StopSystem(Fast) method in the VBVFrame control.

PreDisconnect Event

See also Disconnect Event

When a call that was initiated by this control is terminated, for whatever reason, a PreDisconnect event occurs in this control. The PreDisconnect event occurs just before the system hangs up the line. This allows for call termination code to run before the call is taken back on-hook.

Greeting property sheet

See also: System Phrases

This dialog is where you create your greetings. The Phrase List is a list of VBVoice phrases, which can be Wave files, VAP phrases, or System Phrases.

Play List button
This command will play the entire greeting. This allows you to check the concatenation of the different phrases. If your greeting contains VBPhrases, it is likely that references will be made to control properties. For each reference to a control property, VBVoice will show a dialog asking for the value for this property.

Play Phrase button
This command will play the currently selected phrase.

Remove button
This command will delete the currently selected phrase from the greeting.

Add System Phrase.
See System Phrases

Add VAP Phrase
See : Add VAP Phrase property page

Add File

VAP Phrase property page

The Add VAP Phrase page allows you to add a phrase from a VAP phrase file to the greeting. The phrase files must be located in the Voice directory as specified in the VBVFrame VoiceDirectory property. Once you have found the phrase you want, double-click on the phrase or press Add Phrase / Replace Phrase to insert the phrase into the list.

VAP File

This list box shows all the VAP files in the VBVoice directory. When you select a new file, the phrase list is updated to show all the phrases in the list.

Phrase list

This list box shows the phrases in the current VAP file.
When you select a new file, the phrase list updates to show all the phrases in the list.

Show phrases containing

The phrases shown may be filtered by the ‘Show phrases containing’ field. Whenever this field is changed, the phrase list is updated to only show the phrases containing the text shown. If the field is blank, all phrases are shown.

Announce

Starts up or switches to Announce and opens the selected phrase for enhanced editing capabilities.

Play phrase

Plays the currently selected phrase.

Create file

Opens a file selection dialog which will create a new VAP phrase file.

Edit phrase

Opens the built-in phrase editor. See Creating and Editing Phrases.

Delete phrase

Permanently deletes the phrase from the VAP file. To delete a phrase from a greeting, close this dialog and select the phrase in the greeting property page, then press delete.

New phrase

Use this command to create a new, empty phrase in the phrase file. Enter the required script for the phrase, which is then added to the phrase list. Use the Edit phrase button to add the voice for this phrase. Once in the voice editor, you can create more new phrases. These will be added the list once you save the file in Announce!, and return to Visual Basic.

Shortcut

Double-click on a phrase to select it and add it to your greeting

Creating and Editing Phrases

In the Add Phrase property sheet, select the VAP Phrase page and press Create File. Type in a name of the file you wish to create. This file will appear as the selected file in the VAP phrase list.
To add phrases to your new VAP file, select the Add Phrase button. This will show the Edit phrase dialog (below).

To record the phrase, press the record button. The phrase will be recorded using the device currently set for test mode. The recording will stop after 3 seconds of silence, or use the Stop button to terminate recording.
Pressing record again will overwrite the existing phrase with a new one.

System Phrases

System phrases are used to say standard items like money, times, numbers, etc. and also to select phrases to play using collected digits, database results or defined by code.
The standard phrases use the pre-recorded words and phrases in the VBVOICE.VAP phrase file to generate the complete phrase. Most system phrases have a parameter field, which is a text string set at design time. This string can contain control property names.
Example: if the parameter field is BOX%MailBox%.WAV, and the MailBox control has collected the digits 123, VBVoice will translate the string to BOX123.WAV. Use database validation to check the number first, to avoid phrase errors when the caller selects invalid digits.
A VB Phrase is a special system phrase that allows your code to select the phrase to play at run time. When the system encounters a VBPhrase, it calls the PlayRequest event procedure in the control and passes in the name of that VBPhrase. Your event code can then modify the supplied phrase as required using the Phrase object methods and properties (see Greeting Objects, andCustom Greetings
The control will then play the phrase.

The System Phrase property page is used to add a System phrase to a greeting. System phrases are defined to say money, times, dates, numbers and other system phrases. Most system phrases require a parameter. Use the Find Control button to select a control name and add it to the parameter field. At run time the actual value of the control's default property will be substituted for the control name in the parameter string.

System Phrase Types

Date

Data1: a 4, 6 or 8 character Date string.
Data2: DateFormat string

The first 4 characters of the Date string are mmdd. The last 2 or 4 characters are an optional year field. The DateFormat string specifies the components of date to say (any of day, numeric month, month-name, weekday and year). The current language specifies the order in which to say the various date components.

e.g. 011294 will be said as: 12 Dec 94

01122094 will be said as: 12 Dec 2094

0112 will be said as: 12 Dec (current year assumed)

Digits Data1: the digits and letters to say
VBVoice says a number or string as a series of digits and letters.

FileDate Data1: FilePath
Data2: DateFormat string
VBVoice says the last modified date of the file. . The filename can contain control names. If no path is specified for the file, the VBV main directory is used. Any subdirectory specified by language controls is ignored. . The DateFormat field specifies the components of date to say (any of day, numeric month, month-name, weekday and year). The current language specifies the order in which to say the various date components

FileSize Data1: FilePath
VBVoice says the duration in seconds of the specified Wave file using the normal Wave file sampling rate and compression type. See FileDate for details on how VBVoice finds and opens the file.

FileSpec Data1: FilePath
Plays the specified wave file in full. The file name must be the name of a wave file in

the VBVoice directory, or a full path to a wave file. Example: if the parameter field is BOX%MAILBOX%.WAV, and the MailBox control has collected the digits 123, VBVoice will attempt to play a file BOX123.WAV from the default VBVoice directory. See Playing WAV files, p ?.

FileTime Data1: FilePath
As above, but VBVoice says the last modified time of the file. See FileDate for details on how VBVoice finds and opens the file..

Initial Greeting This phrase is intended to be used as the first phrase in the main voice system prompt. It says Good Morning, Good Afternoon, or Good Evening based on time of day. Before noon, it *says 'good morning'*, from noon to 5 PM, it says *'good afternoon'*, and from 5 PM to midnight it says *'good evening'*.

Money Data1: Money string
VBVoice translates a number into an amount, e.g. '1234' is "twelve dollars and 34 cents". The phrases used by this phrase type, dollar, dollars, cent and cents, are in VBVOICE.VAP and can be replaced with other units as required. To say dollars only without cents, append a period: e.g. '12.' will be 'twelve dollars'.

Number Data1: Number string
Data2: Flags string
VBVoice says the numeric value of the parameter, i.e. the string ''1563'' would be 'one thousand, five hundred and sixty three'. This phrase type can also handle negative and decimal numbers, e.g. ''-123.45'' would say 'minus one hundred and twenty three point four five'..
Flags string: This can contain one of M,F or N to specify gender (some languages have different numbers for gender) and can also contain the digit 0 to say 'Zero' instead of the default 'No'0.
Both the Number field and the flags fields can contain control names.

Number (Ordinal) Data1: Number string
Data2: Flags string
As Number, but says 'first', 'second' etc. Decimal points are not allowed in this number. See Number above for optional flags.

NumberShort Data1: Number string
Data2: Flags string
As Number, but numbers greater than 9 are said as "more than nine". See Number above for optional flags

NumFiles Data1: File specification
The file specification is a file path that contains a drive, directory and wild cards e.g. MAILBOX*.wav. VBVoice will say the number of files found that match the file specification.

SayText Data1: Text to say
This phrase type uses Text to speech algorithms to translate written text into spoken words. Text to speech is an additional option for VBVoice, requiring additional software and/or hardware.

Time Data1: a time in the format hh:mm or hhmm.

VAP Phrase by Index
Data1:VAP file name
Data2: Phrase Index
The index selects the phrase to play. Phrases in the VAP file are numbered from 1. Both the file name and phrase index can contain control property names.

VAP Phrase by Name
Data1:VAP file name

Data2: Phrase name
The filename and phrase name can contain control names to allow phrases to be selected at runtime by control property names.

VBPhrase Data1: phrase identifier string
This is a phrase type that allows VB code to create a phrase at runtime. A name is associated with the phrase so that each phrase can be identified in the PlayRequest event, in the case of multiple VBPhrases in the same control. When a control processes a greeting containing a VBPhrase before starting to play it, it will generate a PlayRequest event for each VBPhrase present in the greeting. VB code can then set the file or phrase to play using the Phrase object passed in with the event. There can be as many VBPhrases as required in a greeting. See PlayReques event for an example.

A PhraseError event is generated if an error occurs during phrase processing, such as the specified file does not exist, or an invalid control name has been specified.

Date and number formats

Greeting Objects

See also Phrase Objects

Greeting properties

Name
The name of this greeting

Fileformat
The play format of this greeting - one of the vbvFileFormatConstants:
vbvULAW_8K = 0
vbvOKI_8K = 1
vbvULAW_6K = 2
vbvOKI_6K = 3

Phrase(Index as Integer)
Each phrase object in a greeting can be accessed through this property. The phrase objects can then be manipulated using the Phrase methods and properties below.

TermOnDTMF (BOOL)
This property signifies whether playing the greeting will terminate when a DTMF digit is detected.

Greeting methods

Copy(Greeting as Object)
Deletes all existing phrases in a greeting and makes a copy of the greeting object provided by copying all phrases in it.

Count() As Integer
Returns the number of phrases in this greeting

Note:
The Insert.. methods all create a new, empty Phrase object, and then use one of the Phrase Create.. methods to create a valid phrase. See the Phrase methods below for more details on phrase creation..

InsertNamedPhrase(Position as Integer, FileName as String, PhraseName as String)
Creates a new phrase object, executes the CreateNamedPhrase method on it, and inserts it into the greeting at the specified position.

InsertIndexPhrase(Position as Integer, FileName as String, Index as String)
Creates a new phrase object, executes the CreateIndexPhrase method on it, and inserts it into the greeting at the specified position.

InsertFile(Position as Integer, Filename as String)
Creates a new phrase, executes the CreateWavePhrase method on it, and inserts it into the greeting at the specified position.

InsertSysPhrase (short Index, short Type, Data1 as String, Data2 as String)
Creates a new phrase, executes the CreateSysPhrase method on it, and inserts it into the greeting at the specified position.

RemoveAt(Index as Integer)
Removes the phrase at the given position and deletes the object

RemoveAll()
Removes all the phrases from the greeting and deletes them

Phrase Objects

Phrase properties

These properties can be used to query the contents of a phrase. We recommend that new phrases are created using the methods below.

Filename
the filename of the WAV file or VAP file for a VAP phrase. Ignored for system phrases

Phrasename
For phrases of VAP Phrase By Name, the script of the phrase to play

Type
the type of system phrase - one of the vbvSysPhraseConstants. Ignored if not a system phrase type

PhrsType
the type of phrase - one of the vbvPhraseTypeConstants,

Format
the play format (reserved for future use).

Phrase methods

CheckPhrase()
Performs a check on the phrase. If the check fails, Visual Basic will generate a trappable error with one of the following vbvCheckPhraseConstants:

vbvBadControlName	32200 the phrase contains a control name which does not exist, or is not loaded
vbvVapPhraseNotFound	32201 the VAP phrase was not found in the file
vbvVapPhraseSizeError	32202 the VAP phrase is empty
vbvFileOpenError	32202 the file could not be opened
vbvFileSizeError	32203 a WAV file is empty
vbvNoPhraseData	32204 the system phrase does not have enough data to be specified completely. System phrases require 0, 1 or 2 data fields depending on the type.
vbvAmbiguousName	32205 One of the fields contains a control name, however there are 2 controls of that name in the system and VBVoice cannot resolve which one to use.
vbvFileFormatEror	32306 Format of the file did not match the format set for the greeting

The Visual Basic Err object will contain the error code in the Number property, and the phrase or filename in error in the Description property.

CreateIndexPhrase(Filename as String, Index as Integer)
This method creates a phrase object that refers to a phrase in a VAP file. The phrase is referenced by position. Index 1 is the first phrase. TODO is this true?

CreateSysPhrase(PhraseType as Integer, Data1 as String, Data2 as String)
This methods creates a system phrase. PhraseType is one of the vbvSysPhraseConstants. The Data strings are dependent on the phrase type -see Types of system phrase and required data below.

CreateNamedPhrase(Filename as String, PhraseName as Integer)
This method creates a phrase that refers to a phrase in a VAP file. The phrase is referenced by the name

of the script. The phrase name is case-sensitive.

CreateWavePhrase(Filename as String)
This method creates a phrase that refers to a WAV file. The file will be played in its entirety.

Copy(Phrase as Object)
This method creates a phrase that is a copy of an existing phrase.

Types of system phrase and required data

vbvSysPhraseConstants	Data1	Data2	Constant Value
NumFiles	File Path	-	0
FileCustom	File name or path	-	1
FileSize	File name or path	-	2
FileDate	File name or path	DateFormat	3
FileTime	File name or path	-	4
SayNumber	Number to say	Number Flags	5
SayNumberShort	Number to say	Number Flags	6
Digits	Digits to say	-	7
Money	Money value		8
File Date and Time	File name or path	DateFormat	9
InitialGreeting	-	-	10
SayTime	Time to say	-	11
SayDate	Date to say	DateFormat	12
Time Now	-	-	13
Date_Today	DateFormat	-	14
VAPPhrase	Filename	PhraseName	15
VAPPhraseIndex	Filename	Index	16
NumberOrdinal	Number to say	Number Flags	17
Custom	Field1	Field2	18
SayText	Text to say	-	19

vbvPhraseTypeConstants;
vbvWavFile = 1
vbvSysPhrase = 2
vbvVapPhrase = 4

DateFormat strings:
can be one of
“MonthName"
“WeekDay"
“Day,Month"
“Day,Month,Year"
“Day,MonthName,Year"
“WeekDay,MonthName,Year"
“Day,MonthName"
These strings are not case sensitive.

Number Flags
This string can contain one of M,F or N (gender flags) and optionally 0 to say ‘Zero’ instead of ‘No’ for the number 0. For example, you may want to say ‘You have no messages’ in one phrase, but ‘Zero’ in another. TODO give me a good example of using zero, please

Possible Errors
The phrase creation functions do no error checking. Use the CheckPhrase method to check the phrases for validity.

Customizing greetings at runtime

There are 3 methods that can be used to customize greetings at run time. The most common method is to use a System Phrase in conjunction with control property substitution. This allows VBVoice to say data from the caller and from databases, or from properties set by VB code without having to manipulate phrase objects directly.
The 2 other methods that can be used to modify greetings using VB code are:

1. the System Phrase type VBPhrase. See System Phrases
1. direct manipulation of phrases in the EntryGreeting by code in the Enter and EnterB events.

The VBPhrase phrase type is a special system phrase that generates a VB event when encountered by VBVoice. The event code for this event can create any type of phrase to play. A greeting can have as many VBPhrases as required.
The EntryGreeting for each control can also be modified directly using the Greeting and Phrase methods - see Greeting Objects and Phrase Objects. These methods can add, remove or modify phrases in the greeting as required.

Event Log Options dialog

The Event Log Options dialog is used to control which events are shown in each channel log window, and also to control which events are logged to the log file.
Channel log windows will display events as they occur, and also can extract events from the log file and show them.

Note that having many log windows open may slow down your system.

Event types to monitor:

errors	
calls	the start and stop time for each call,and the system start and stop times
call flow	a call flow message is logged whenever a call leaves or enters a control
information messages	these are general information messages about the current action of each control
invalid digits, time-outs	these log the events that occur when a control detects a timeout or invalid digits from the caller.
driver events	low level voice driver events
driver functions	low level voice driver function calls
control states	These messages reflect the internal state machines used by each control

The VBVoice Log File

A new log file is created every day to avoid the log file becoming unnecessarily large. The file name for the log files are of the form vbvdd-mm.vlg, where dd is the day, and mm the month. Log file size may still become a problem if all events are monitored to the log file. Normally calls, call flow and error events should only be logged to the log file. You can choose which event types to log using the LogFiles item in the Setup menu.

See also Event log windows

Event log windows

There are 2 type of log window, the test log window that is used to display logs when in test mode, and the log windows used at runtime. The test log window is also used to show the results of control and system checks.
The test log window is shown automatically when a test starts.

Showing the log window:
Run-time log windows can be shown in two ways:
1. using a method in the VBVFrame control.
2. by double-clicking on the channel number in the linestatus window

The log window is a hierarchical list of items. The top level items represent calls - either incoming or outgoing, or in the case of the test window, a test call.
Below this are individual controls that are executed by the call. These are the only levels normally visible unless errors have occurred.
Individual log entries for each control are contained within each control item. Click on the + to open up the listing for a control.

Getting help on log entries

Most log entries have help items associated with them. You can access help by selecting a line and pressing the F1 key, or right clicking on the window to get the context menu, and selecting Help.

Finding a control referenced by a log entry

Often you will want to find the control associated with an entry in the log (usually an error!). Double click on the error line to show the form and page containing the control, which will become the highlighted control.

See also LineStatus Window

LineStatus Window

The linestatus window shows the status for all the lines in an application. It can be toggled on and off using the context menu in a VBVFrame control, or by VB code using the ShowLineStatus method. It is also possible to embed this window into your own form - see the SetLineStatGrid method in the VBVFrame control.
Double-click on the channel number in the linestatus window to show the channel log for that channel.

See also Event log windows

Test Mode

General Considerations using Test Mode

Test mode allows you to start a test of your system at any point in the call flow diagram. Test mode can use a sound card or a voice card. When using the sound card, it provides an on-screen dial pad to use for entering digits. (You can also use the keyboard).
Show the Test Mode dialog by selecting the Test menu item in the context menu. The control that will start the test is highlighted (white text on black) and is shown in the test dialog caption. You can change the start control by clicking on the required control, or it can be selected via the Start Control**...** button.

Testing Subsystems and Individual Controls

Test mode is useful for testing a subsystem without having to worry about how the call is normally routed to the subsystem. In complex multi-level menus, it is convenient to be able to start at any point in the system, however, consideration must be given to the interactions between controls in your system. Consider the scenario where a Record control uses control name substitution from a previous GetDigits control to create a filename based on the entered digits. If test mode is started at the Record control, the GetDigits control will not have collected any digits. VBVoice detects this condition, and will present a dialog requesting that you fill in the value that the GetDigits control would have supplied. You can enter an invalid value to check all error conditions.

Starting a test

Start a test by selecting the control where you want to start the test, either by clicking on it, or choosing Start In... and selecting a control name. Then select the Start command to start the test. If you are using a voice card, which is onhook, you may be prompted to dial in to start the test. If there is a validation error in the control, the test will not start and an error message will appear in the test log.
When the test starts, the test log window will be shown. This window monitors events as they occur in the system. If an error occurs, it will be logged in this window, and the test will stop. To get more information on a log entry, select the line and press F1. To select the control related to an error, double click on the error line.

Control Checks

Each control will be checked for errors before a call is passed to it, so it is not necessary to have the complete system error-free before starting a test. However you may find it useful to run the Check Control command before starting test mode.

Call Flow

You can see you how your call is progressing during the test. The call always starts in the control with the highlighted name. If the call moves to another control, the connection is highlighted. If an error occurs, or the call is terminated with the Stop button, the control name of the last control is highlighted in dark red, allowing you to zero in on error locations quickly.

Timeouts and caller hangup

The test dialog provides Timeout and Hangup buttons which allow you to simulate a time-out while waiting for digits, or a caller hangup at any time.

Outdialing and call progress

When in test mode, VBVoice will not dial but instead will speak the digits that it would have dialed. In addition it will not perform transfers or call progress detection. You can select the call progress condition that you want to test from a dialog, so you can test all control paths without having to recreate (and wait for) ring-no-answer, busy, etc..

Special control behavior in test mode

If a test is started in a LineGroup control, VBVoice uses a dialog to ask if the call should continue via the ‘Ring’ or ‘StartCall’ connections.

Similarly, if a TimeSwitch control is encountered, a dialog is used to allow you to enter the time you wish to test.

Design mode - limitations

Test mode can be invoked while Visual Basic is in design mode, and is often useful as a quick test of part of your design, however it does have limitations.

Database access:
Database access is not available in design mode when using recordsets based on code. If the call arrives at a DataFind control which uses VB data recordsets, and Visual Basic is in design mode, the test will terminate. ODBC database access is available in design mode and run mode.

Visual Basic code:
If your design relies on Visual Basic code, in custom VB phrase events, Enter events or elsewhere, this code will not be run while in design mode. Since VBVoice cannot detect whether you have written code in these events, test mode will run but may not function correctly.
VBVoice features that rely on being in run mode, such as User controls, and the system phrase type VB Phrase, will generate an error if they are encountered during a test when in design mode.

Run mode limitations

Test mode can be invoked while Visual Basic is in run mode. In this case, database access is always available and Visual Basic event procedures will be called.
You must ensure that Visual Basic loads the required forms at startup. This is a requirement for normal operation as well as for test mode. If only one form is to be tested, it can be made the startup form in the project options dialog (in the Options menu). If more than one form is required, the form load procedure in the startup form or module must load all other required forms. You can load the form as an icon by setting the WindowState property to 1.

Visual Basic debugging:
Test mode does not interact with Visual Basic debug mode, as it does when running using Start System (see Running your Application). If Visual Basic switches to break mode due to an error in your code, Test mode will terminate.

Using the voice card in test mode

Using the voice card in test mode operates exactly like a real call, except for the start and end of each call, and for dialing.
The system will establish a call before the test starts using the configuration settings in the Test Mode property page in the VBVFrame control. When the test is completed, the system can keep the line offhook, ready for the next test if required.

VBVoice Test Window

When VBVoice test mode is started, access to the Visual Basic and VBVoice main menus is denied. Terminate test mode with the Cancel button to return to normal operation.

Start In ...

This command allows you to select the control at which you want to start the test. You can also do this by clicking in the control.

Start/ Stop

Use this command to start the test. Once the test is active, you will not be able to make any design changes until the test is stopped.

HungUp

The HungUp command simulates the caller hanging up the phone to terminate the call when using a sound card, and avoids the need to drop the line when using a voice card.

Pause/Resume (Sound card only)

When using a sound card, you can pause play, get digits and delay operations until you are ready to resume.

Cancel

This command will terminate test mode and will return the system to normal operation.

DialPad (Sound card only)

The dial pad is used to enter digits from the caller when testing using a sound card.

Onhook / Offhook (Voice card only)

The onhook/offhook buttons show the current status of the hookswitch when using a voice card. Normally when testing a voice system you will want to stay offhook and connected to the voice card for the duration of the test - this saves time dialing and waiting to be reconnected each time. However if you lose the connection, you can use the onhook button to clear the line and dial in to the voice card again to re-establish the connection.

Using databases

Creating a database

You do not need a special application to create a simple database. Visual Basic provides a tool, Data Manager, which can create and update Access format databases. While Data Manager is limited, it does provide a quick and simple way to view and create databases. Data Manager can be started from the VB Add-Ins menu.

Recordsets

A **Recordset** represents a set of records selected from a data source. **Recordset**s are available in two basic forms: dynasets and snapshots. A dynaset is a dynamic recordset that stays synchronized with updates by other users. A snapshot is a static recordset that reflects the state of the database at the time of the snapshot. A dynaset reflects changes subsequently made to records in the dynaset, either by other users or by other recordsets in your application when you move to a new record.

A dynaset

is a recordset with dynamic properties. During its lifetime, a recordset object in dynaset mode (usually called simply a ‘‘dynaset’’) stays synchronized with the data source in the following way. In a multi-user environment, other users may edit or delete records that are in your dynaset or add records to the table your dynaset represents. Records that your application adds to or deletes from the recordset are reflected in your dynaset. Records that other users add to the table will not be reflected in your dynaset until you rebuild the dynaset by calling its **Requery** member function. When other users delete records, the code skips over the deletions in your recordset. Other users’ editing changes to existing records are reflected in your dynaset as soon as you scroll to the affected record.
Similarly, edits you make to records in a dynaset are reflected in dynasets in use by other users. Records you add are not reflected in other users’ dynasets until they requery their dynasets. Records you delete are marked as ‘‘deleted’’ in other users’ recordsets. If you have multiple connections to the same database (multiple **DataFind** controls), recordsets associated with those connections have the same status as the recordsets of other users.
Dynasets are most valuable when data must be dynamic, as, for example, in an airline reservation system.

Note A dynaset is a dynamic *but fixed* set of records. New records that meet the selection criteria after the dynaset-type recordset has been created are not added to the recordset. This includes records that other users add. Similarly if a record is changed after the dynaset has been created, such that it does not meet the criteria, that record will not be removed from the dynaset.
DAO: Dynasets have bookmarks, snapshots don’t. Bookmarks are required if the recordset is not read-only and the same recordset is being shared between channels.
ODBC: a forward-only recordset is a read-only recordset that can scroll forwards only. Bookmarks (required for shared access) are not supported). Has less overhead than other types of recordset.

A snapshot

is a recordset that reflects a static view of the data as it existed at the time the snapshot was created. Once you open the snapshot and move to all the records, the set of records it contains and their values don’t change until you rebuild the snapshot by calling **Requery**.

You can create updatable or read-only snapshots. Unlike a dynaset, an updatable snapshot doesn’t reflect changes to record values made by other users but does reflect updates and deletions made by your program. Records added to a snapshot don’t become visible to the snapshot until you call **Requery**.

Snapshots are most valuable when you need the data to remain fixed during your operations, as when you’re generating a report or performing calculations. Even so, the data source can diverge considerably from your snapshot, so you may want to rebuild it from time to time.
Snapshots are available only if the ODBC driver you’re using supports static cursors.

Important For some ODBC drivers, snapshots may not be updatable. Check your driver

documentation for cursor types supported and the concurrency types they support.

VoiceError event: code 3197. When using a bound data control, this means could not update record because it had changed already by another process or another data control. Can choose to ignore, or retry or get/set field directly via Fields object, or refresh database, although this may cause other channels to lose their position.

Accessing the data

You can add some text boxes to your form, and link them to fields in the database. The text boxes will display the contents of the current record in the database, and will also allow you to change the data in the database.

To link a text box to a database, perform the following operations:

1. Add a text box to your form. You can use a label also, but this will not allow you to change the data.
2. Set the DataSource property to the name of the data control to which you want to link.
3. Set the DataField property to the field name in the table.

Now, when you start your program running, the text box will show the contents of the first record in the database.

To scan through the database, use the arrows on the data control.

Control Property Substitution

VBVoice has a powerful mechanism to control the system based upon data collected from the caller, using control property data values.

Default properties

Most VBVoice controls have a default property value. This value is available for use by other controls, in greetings and other setup fields. For instance, the GetDigits default property is the Digits property, containing the digits collected from the caller. This collected digit string could be used in a DataFind control to select the database record to be found, or to select the file to be played as a greeting in another GetDigits control.

Syntax - using the default property

To tell VBVoice that you want to substitute a control value, surround the name of the control with % marks - for example %GetDigits%. At run time, the %xxx% string will be replaced with the actual value of the default property. If you copy the name from the Controls List dialog, and paste it into the dialog, the % marks will be added for you.

Syntax - using a named property

Some controls have more than one property which can be accessed from other controls. You can specify which property by using the %control-name.property% syntax.

Notes

- if you drag the control name or property name from the control list dialog, it is automatically surrounded by % signs for you.
- VBVoice uses the control name to identify properties. Therefore, no two control names should have the same name, even if on different forms. VBVoice will check for this condition during the system check.

Edit boxes on dialogs that will accept drag-drop operations from the control list window have the control list button alongside.
Pressing this button shows the control property selection dialog:

Special property names %this%, %channel%

The special control name *this* which can used in the same way as an actual control name - e.g. %this.Digits%. The *this* keyword refers to the control that contains the reference.
The special control property name *channel* should be used on it's own to refer to the number of the active channel. This can be used to create names that are unique to a channel.

See also
Control property selection dialog
Ways to use Control Property Substitution

Control property selection dialog

To add a control name to a field, select the required control, and then drag the control name (for the default property) or the property name to the field into which you want to add the property name. VBVoice will automatically add the %% syntax.

See also
Control Property Substitution
Ways to use Control Property Substitution

Ways to use Control Property Substitution

See also
Control Property Substitution
Control property selection dialog

Say an number entered by the caller

If you have collected some digits from a caller, and with to play them for confirmation, use the SayNumber phrase.
Say you have used a GetDigits control called GetMailBox to collect the digits. To play back the numbers that the user has entered, use the system phrase SayDigits or SayNumber in another control, with the parameter %GetMailbox%.

Data verification

To verify data entered by the caller, create a database containing all the valid numbers or digits, and use the name of the GetDigits control as the data to match in the DataFind control.
To add verification to the example above, add a data control that is connected to your database via the DatabaseName and RecordSource properties. Add a DataFind control, and in the setup dialog, set the "Search in database" field to the name of your data control, set the "For..." to the %GetPartNum% and set the "Field" entry to the name of the data field containing your valid data. Then, if the digits collected are in the database, the call will proceed out of node "Found". If there is no match, the call will continue out of the node "Not Found". In this case, you may want to give the caller another try, just as if the digits had been found invalid by the GetDigits control itself. To do this, connect the NotFound output back to the Err input on the GetDigits. This will increment the error count in the GetDigits control, play the error greeting followed by the main greeting and give the caller another change to enter a valid number. After the maximum number of retries have been made, the call will exit via the Invalid output in GetDigits.

Look up numeric data and say it

VBVoice can also say numbers from the database. For instance, to add to the example above, your database containing valid numbers can have another field Quantity.
Use a DataFind control called GetQuantity to access the field by connecting it to the Found output of the DataFind control, and setting the 'Get Data from Field' entry to the name Quantity. (It should appear in the list provided). No conditions are required in this example. Now in a subsequent voice control, use a greeting containing a phrase:
System Phrase Say Number %Quantity%

Look up other data and say that too

Your database could also have another field "Description' containing a list of filenames, each of which is a spoken description of the item.
VBVoice can also say data from the database. For instance, to add the example above, your database containing valid numbers can have another field containing the filename of a prompt describing the record. To say the description, use a greeting containing a phrase:

VAP Phrase: "You have selected"
System Phrase FileSpec<% GetQuantity Description' %

to say a description of the item the caller has selected.

Another way of solving this problem is to use a System phrase of type VB Phrase. When VBVoice needs to play a VB Phrase, it will call your PlayRequest event procedure. Your code can create a phrase of any type using the Phrase properties and methods.

Look up data using letters on the keypad

You can search databases using letter substitution for the numerals on the keypad, and cycle through all

the entries found - useful for directories in voice mail applications. Set the Alpha digits field in the DataFind control to do a search based on phone key to alphabetic translation.

Transferring data from VB variables into Phrases

The VBVFrame Transfer property is useful for this. This property allows you to create your own properties which can be accessed at runtime. To create a property, open the VBVFrame property sheet and select the Transfer tab. Press the New button to create a new transfer property with a name of your choosing. For instance, you could create a property MyProp1 in the VBVFrame1 control.
This property can then be added to a phrase or any other VBV control dialog that accepts the %control.property% syntax by inserting %VBVFrame1.MyProp1%. If you drag the MyProp1 field from the Control List dialog directly into a phrase setup field, the syntax is supplied for you.
At runtime you can write to this variable using the syntax VBVFrame1.Transfer.MyProp1(channel) = “NewValue”

Error Handlers property page

This property page, which is available on all voice processing controls, is used to select which control, if any, will handle errors in the control. It is also used to select a control to override the help digit handler provided by the LineGroup control. When one of these events occurs, the call will exit this control and will enter the control as specified for that event. Each control can specify a different event handler for each event. If no control is entered i.e. the field is left as [Default], then the standard handling for that event takes place, as described below.

Error Handling

Invalid Digit, No Digits and Silence Timeouts

Voice systems that expect input from a human always have to deal with incorrect responses, such as unexpected digits, or no digits. VBVoice controls that expect caller input normally make several attempts to get the required response from the user, using a configurable retry count and error message. If valid input is still not obtained, the normal practice is to play a polite message and terminate the call, or possibly to attempt to transfer the call to an operator. VBVoice has 3 levels of error handling:

1) Automatic retries in the control. Most controls have a configurable number of retries together with special prompts in order to attempt to get correct input from the user.
1) In the GetDigits control only, an option to connect another control directly to the Invalid and NoDigits outputs. These outputs are only available if the Use Default Error box is turned off.
1) A event handler for the control corresponding to the error type. These event handlers, set in the Error Handlers property page, allows each control to designate another control that will handle any of 4 different types of event:
   Invalid Digit error
   No Digits / Silence error
   Global Help digit detected
   Caller Hangup detected
   When one of these events is detected, the call will exit the control, and then enter the new control. This new control then handles the call in the normal way.
1) If no event handler is set in the control (see Error Handlers property page below), the LineGroup control has outputs which provide global event handlers for the Invalid, NoDigits and HelpDigit conditions. Connect a control to the appropriate output - when the error occurs on a channel serviced by this LineGroup control, the call will be passed to that control and the call will continue from that control.
1) If all else fails, VBV will play the ERROR.WAV file and hang up the line, following the Disconnect procedure

Disconnect handling

Detecting caller hangup

Many telephone systems indicate that the caller has hung up by signaling a loop current drop. This is a temporary drop in the current from the telephone switch. This event can be detected by the voice hardware, and used to terminate the call.

When the voice card is connected to a PBX, caller hangup is often indicated with dial tone, or a busy signal. Most voice cards can detect this tone while playing prompts and will cause VBVoice to generate a Disconnect event in the same manner as a loop current drop. The voice card may have to be trained to learn the tones on your particular system. See the section or notes specific to your voice card for more details. Loop current drop detection can be disabled using a setting in VBV32.INI. See VBV32 INI settings.

Note that if a loop current drop occurs while VBVoice is transferring a call or dialing, it will treat it as a reorder and return to the original caller.

Configuring Caller Hangup Detection

Some systems do not provide loop current drop on caller hangup. In this case, the voice card must be trained to detect the tones provided by the telephone system upon hangup, or the system must be designed so that it cannot get into an infinite loop of waiting for input from the caller.

Responding to caller hangup

When VBVoice detects a loop current drop or disconnect tone, the following sequence of events will occur:

1. The control will generate an Exit event in the current control
1. The control will then check for a disconnect event handler for the control - see Error Handlers below. If a handler exists, the designated control will receive the call, and the following actions will not occur, including the Disconnect event. (It will occur later after the line is taken onhook by the OnHook control). The designated handler may be used to update a database with call information before termination.
1. The LineGroup fires a PreDisconnect event. At this point the line is still off-hook.
1. VBVoice takes the line on-hook and fires a Disconnect event in the control where the call was being handled when the disconnect occurred.
1. A Disconnect event is then fired in the LineGroup control that started the call.
1. The line is then taken on hook

If none of the events set a GotoNode property or calls the TakeCall method, the call is now terminated, otherwise the call is moved to another control and processing will continue from that control. The disconnect handling routine may restart if the call arrives at a voice processing control (i.e. a control that plays greetings), since this control may also detect the loss of loop current.

If VBVoice goes onhook due to a Onhook control or a completed transfer operation in a Dial control, the disconnect event is fired in the LineGroup control only.

VBV32.INI File Settings

SECTIONS

[Directories]

[PBX]

[LineGroup]

[Record]

[System]

[VoiceCard]

Settings

Announce
AnswerDeglitch
CallerID
CallerIDName
Connect
DetectAnswerTime
DetectDialTone
DisableLCDetect
Flash_character
Flashtime
HW_Interrupt
LoadDriver
Logs
MaxDialToneWait
MinDialTone
MinLCOFFTime
MsgBoxErrors
Offhook_delay
PauseTime
PutOnHold
RecChop
RecChopSil
ReconnectFromBusyNoans
ReconnectFromHold
Ring_cnt_reset
Type
Voice

[VoiceCard]

INI SETTING	DESCRIPTION	OPTIONS / UNITS	DEFAULT
DisableLCDetect	Controls loop current detection	1 = no detect 0 = detect	0
Flash_character	the character to be used in dial strings for a hook flash		!
Flashtime	duration of a hookflash	ms	500
LoadDriver	Controls loading of the voice driver	0 = dont load 1 = always load	1
MinLCOFFTime	minimum time for loop drop	ms	set by voice card

			driver
MsgBoxErrors	show message boxes from DLL on startup failure if = 1	1	
Offhook_delay	adjusts offhook delay default	ms	500
PauseTime	duration for a pause character (comma) during dialing	ms	200
Ring_cnt_reset	time after which the driver resets the ring count default set by voice card driver	ms	set by voice card driver
Type	Type of voice card in the system		NONE

[Record]

INI SETTING	DESCRIPTION	OPTIONS / UNITS	DEFAULT
RecChop	deletes bytes from end of recorded messages in order to remove the tone when the record has been terminated by a tone. This setting is for voice cards that do not do this automatically	bytes	0
RecChopSil	deletes silence from end of recorded messages when ending in silence. This setting is for voice cards that do not do this automatically	0 / 1	0

[LineGroup]

INI SETTING	DESCRIPTION	OPTIONS / UNITS	DEFAULT

[System]

INI SETTING	DESCRIPTION	OPTIONS / UNITS	DEFAULT
Logs	set to 1 to enable event logging in your .exe		0

[PBX]

INI SETTING	DESCRIPTION	OPTIONS / UNITS	DEFAULT
DetectDialTone	duration of nonsilence to detect as dialtone during play operations. 0 to disable.	seconds	10
MaxDialToneWait	secs to wait for dialtone before abandoning	ms	
MinDialTone	minimum period of dialtone required to start dialling	ms	
Putonhold			!,
ReconnectFromHold			!,
Xfer			!,
Connect			!,
ReconnectFromBusyNoans			!,
CallerID	Enables caller id option		False
DetectAnswerTime	Time used to detect answer machine, person in dial	ms	6000
AnswerDeglitch	silence in answer before stopping answer size detect	ms	600

[Directories]

INI SETTING	DESCRIPTION	OPTIONS / UNITS	DEFAULT
Voice	default directory for voice files		C:\VBV32\
Logs	default directory for log files		C:\VBV32\LOG
Announce	default directory for ANNOUNCE.EXE		

ambiguous control name

VBVoice controls must have unique names
It is possible in Visual Basic to have 2 controls of the same name on different forms. Since VBVoice does not use form names to qualify control names, all controls names have to be unique across all forms. Rename one of the controls. If you have other controls which reference this control, ensure these are also updated to refer to the correct control.

bad control name in %s %s", msg, filename

An invalid control name was specifed as a parameter in a property page. Either the control name is incorrect, the control does not exist, or the control is on a form that is not presently loaded. If the control exists, ensure that it is on a form that is loaded (if VB is running), or on a form that is visible (if in design mode).

Control names are case sensitive, and must contain the index of the control if part of an array: i.e. ControlName(7)

incoming call detected. Call abandoned

An incoming call was detected after a call had been dialled - i.e. after going offhook, dial tone was not detected. This normally happens when the system goes off hook to start dialling, just as an incoming call arrives, but before the voice card has detected ringing.

delay start

The current call has terminated on the line, and the LineGroup control has been set to start a call after a delay."
INFOMSG

dialling on hold digits

A call has entered a Delay control, which is now dialling the digits required to put the calle on hold during the delay.
INFOMSG

delay completed

The delay time in the Delay control has expired.
INFOMSG

reconnecting from hold

INFOMSG

inc count (value)

A Count control has incremented its internal counter.
INFOMSG

number to dial

A call has entered a Dial control, which has set this number to dial. Once the greeting has completed, the number will be dialled.
INFOMSG

starting dial

The greeting in a Dial control has now compteted, and is starting the dial operation, with no call progress analysis.
INFOMSG

starting call

The greeting in a Dial control has now compteted, and is starting the dial operation, with call progress analysis.
INFOMSG

number changed to (dial number)

A call entered a Dial control, and VB code changed the number to dial using the NumToDial property.
INFOMSG

transfer to (control name) via VB

VB code has set the GotoControl property in the Enter event, and the call has been transferred to a new control.
CALLFLOW

transfer to (control name) via node (number)(name)

The call has been transferred to a new control via the output node given.
CALLFLOW

value on exit is (string)

All controls that have default properties (for the purposes of Property Substitution) log this message when a call leaves the control. The value shown is the string that will be substituted for the string %ControlName% in other controls.

INFOMSG

ini (file, section, field, value)

The IniSwitch logs this message, showing the INI file accessed and used for branching
INFOMSG

empty record file

A Record control started a record, but did not receive enough voice data for a valid voice file. You can set the minimum duration of voice required to validate a recording in the Record setup property page. This condition will invoke the silence handler in the Record control.
INVALIDMSG

record file name (filename)

A call has entered a Record control, which has set this filename to record into. Once the greeting has completed, the file will be created, and the record operation started.
INFOMSG

record ok

The Record control made a valid recording, and will now play the options greeting (if there is one).
INFOMSG

silence, error count (secs)

An information message indicating that the control received silence while recording or waiting for digits. The error count is incremented, and if the error count is exceeded, error handling is invoked.
INVALIDMSG

VB set bad block (control name)

VB code attempted to transfer a call to a new control via the GotoControl property, but the specified control does not exist.
ERRORMSG

VB set new block (control name)

VB code has set a new control to transfer to using the GotoControl property. Once VB code leaves the event, the transfer will take place.
INFOMSG

VB set bad node (node number)

VB code attempted to transfer a call to a new control via the ControlNode property, but the specified control name does not exist.
ERRORMSG

VB set new node (node number)

VB code attempted to transfer a call to a new control via the GotoNode property, but the specified node does not exist or is not connected.
INFOMSG

VB set termdigit (mask)

VB code set a new terminating digit via the TermDigit property.
INFOMSG

VB set max time (secs)

VB code set a new maximum time via the MaxTime property.
INFOMSG

VB set max sil (secs)

VB code set a new maximum silence time via the MaxSilence property.
INFOMSG

VB set new file (secs)

VB code in the Enter event for the Record control set a new filename for the file to be recorded, overriding the file selected by the Record control.
INFOMSG

VB set max keys (secs)

VB code set a new maximum number of keys for the operation via the MaxDigits property.
INFOMSG

VB set CPL file (secs)

VB code set a new CPL file using the CPLFile property in the Record control.
INFOMSG

waiting for digits, max n mask 0x1234

This is an information message only for those of you who know the voice card driver functions.
INFOMSG

invalid count (count)

An information message indicating that the control received invalid digits while waiting for digits. The error count is incremented, and if the error count is exceeded, error handling is invoked.
INVALIDMSG

no connection on node (node number, node name)

An attempt was made to transfer to another control, but the node selected for transfer is not connected, or is connected but VBVoice cannot find the destination control. Since the system is checked for complete connection integrity before system start up, this message should not normally occur except in test mode. In systems with more than one VBVoice form, you must ensure that all forms are displayed, or loaded and minimized, before starting a test. VBVoice will not be able to find controls on forms that have not been opened.

When testing in design mode, you must load the forms by selecting each form in turn and clicking View Form in the project window. The forms may be minimized if you wish.

When testing in run mode, you must tell Visual Basic to load the forms upon startup. To do this, go to the startup form (as defined in Options menu, under Project) and use a Load statement, or the WindowState property to load each form in the Form.Load procedure for that form. See the CALLANS form in the VMDEMO example for an example. Normally you will want to display the forms while testing, but have them hidden when creating your .exe. Use a command line option to hide the forms in run environment, but not in the VB environment. You can set command line options in Visual Basic environment using the Options... Project menu. To hide a form without unloading it, use the form's Visible property, or set its position off the screen.

ERRORMSG

incomplete control name (string)

While processing a field, VBVoice encountered an incomplete tag - ie a % sign with no terminating % sign.
ERRORMSG

bad_number (dial string)

A Dial control attempted to dial a number that does not contain valid characters.
ERRORMSG

bad phrase, type --- name ---

VBVoice was unable to process the phrase as defined.
ERRORMSG

vap phrase not found, file (filename), phrase (phrasename)

VBVoice was unable to process the phrase as defined. Normally due to the phrase as named does not exist in the VAP phrase specified.
ERRORMSG

play already active

This message occurs when the system attempts to play a greeting when there is already a greeting playing on the channel. This signifies that the VBVoice state machine has got out of step with reality, and should not occur.
ERRORMSG

control not found (control name)

A control name was referenced in a property page field for substition ie using %Controlname% syntax, but the control does not exist. If in VB design mode, check that the form containing the control is showing or minimized. The controls are not loaded into the system until the form is loaded.
If VB is in run mode, ensure that the form containing the control is loaded using a Form.Load or a Form.Windowstate statement. this statement should appear in the Load event for the startup form, or in Sub Main in Global.bas, depending on what has been set in Options/Project for the startup form.
ERRORMSG

bad record filename (filename)

VBVoice was unable to open the specifed file for recording
ERRORMSG

unexpected event (event name)

VBVoice encountered an event from the voice driver for which it has no predefined response. This means either there is a bug in our voice engine (surely not), a situation on your phone line which has not been anticipated, or the voice driver itself is not working correctly. Please call Tech Support.
ERRORMSG

digits received (digit string)

The GetDigits control has terminated digit collection based on the digits received from the caller.
INFOMSG

VB set new result (string)

VB code has set a new result (default property) for the control. Applies to GetDigits, User, IniSwitch and DataSwitch controls.
INFOMSG

control not found in condition (condition name)

A GetDigits or DataSwitch control has a condition set using property substitution, but the specified control cannot be found. See CONTROLNOTFOUND.
ERRORMSG

data control not found

Occurs if VBVoice could not locate the data control linked to a DataFind control at design time. Every DataFind must reference a VB data control on the same form. This control can be selected in the property page for the control.
ERRORMSG

database error (error code)

An error occurred while accessing a data control. The error code refers to a VB databaseerror - check in the VB help file for details under database error codes.
ERRORMSG

database data (fieldname, data)

Logged by a DataSwitch control when accessng a database field.
INFOMSG

database record found (search field, search data)

A DataFind control has found a record that matches the requirements specified.
INFOMSG

silence count exceeded (count)

A silence timeout has occurred and the control has exceeded the maximum number of timeout or invalid digit conditions allowed.
ERRORMSG

error count exceeded (count)

An invalid digit has occurred and the control has exceeded the maximum number of timeout or invalid digit conditions allowed.
ERRORMSG

data not set in control (controlname)

This message occurs when an attempt is made to access a default property from a control that does not have one. Since this condition is checked at startup, this message should never happen, except when the system has been started and an attempt is made to acces a property that has not been set yet - like a forward reference., or at design time when a property page is used to provide the default property and the Cancel button is pressed. The data provided with the message is the name of the control in question.
ERRORMSG

cannot create unique file

Generated by the Record control when it is building a new record filename. It tries 30 times to create a unique filename based on the record file specification in the property page, and then gives up and logs this message.
ERRORMSG

database not accessible in design mode

VB data controls do not allow access to the underlying database when VB is in design mode. To allow access to your data during testing, start VB in run mode using the Run/Start menu command, and then restart your test. Note that if you have multiple forms, they must be expicitluy loaded by Visual Basic code in the Form Load event or Sub Main. See Running your application in the User's Guide.
ERRORMSG

end of database

The DataFind control came to the end of a database without finding any more records that match the criteria.
INFOMSG

bad time spec (string)

Generated by phrase play function when playing a System phraes of type Say Date, Say Time,.
ERRORMSG

vap phrase id not found (phrase number)

VBVoice is searching for a particular phrase in a VAP phrase file, but cannot find it.
ERRORMSG

error setting recordsource property

The Record control is updating a database with the filename recorded. An error occurred while setting the RecordSource in the data control.
ERRORMSG

restarting record

Occurs in the Record control when the caller presses the digit assigned to re-record the message.
ERRORMSG

record cancelled

Occurs in the Record control when the caller presses the digit assigned to cancel the message.
INFOMSG

playing back recording

Occurs in the Record control when the caller presses the digit assigned to play back the message.
INFOMSG

connected to xferee

The Dial control is calling a number with call progress and Usr Transfer enabled. An answer has been detected and the call continues connected to the transferee.
INFOMSG

digits dialled

The Dial control is calling a number with call progress. An answer has been detected.
INFOMSG

phrase object does not exist

The DLL function vbv_remove_userphrase was called with a bad phrase object
INFOMSG

transfer to control (control name)

Logged by the voice engine when resetting the current control for a channel back to the LineGroup control that owns this line.
CALLFLOW

control error

An error occurred in a control while processing a voice driver event.
ERRORMSG

search database for (data) in (fieldname)

Logged by the DataFind control when starting a database search
INFOMSG

control state (number)

Logged by all controls when changing their internal state. Refers to the state machines internal to each control which are used to manage the voice engine.
CONTROLSTATES

setting recordsource

Logged by the Record control when updating a database with a new filename.
INFOMSG

fire event failed

VBVoice received an error code when attempting to fire an event proceduer within Visual Basic. This may occur if Visual Basic is already in an event - VB does not allow nested events, which may occur if code in an event allows the system to run by calling DoEvents or displays a modal dialog or message box
ERRORMSG

no data in control (control name)

This message occurs when an attempt is made to access a default property from a control that does not have one.
In test mode, the system will prompt you to enter some data, rather than generating this error message. If you press Cancel, this message will be generated.
At runtime, you may get this message when the system has been started and an attempt is made to acces a property that has not been set yet - like a forward reference
The data provided with the message is the string that references the control.
ERRORMSG

bookmark set

VB code set the bookmark for a DataFind control.
ERRORMSG

using cpl file (cpl filename)

A Dial control has been set to use a non-default CPL file. When the call arrives at the control, the channel will be loaded with the new Call Progress Analysis parameters before dialling and performing call analysis.
INFOMSG

transfer to (controlname) via (node number, node name)

A call has been transferred to a new control. This message logs the tranfer, and the value of the default property of the original control on exit.
CALLFLOW

voice call

Voice driver message - the voice driver CALL function has been invoked
LOGFUNCMSG

voice dial

Voice driver message - the voice driver DIAL function has been invoked
LOGFUNCMSG

voice getdtmfstring

Voice driver message - the voice driver GETDTMFSTRING function has been invoked
LOGFUNCMSG

voice play

Voice driver message - the voice driver PLAY function has been invoked
LOGFUNCMSG

voice record

Voice driver message - the voice driver RECORD function has been invoked
LOGFUNCMSG

voice offhook

Voice driver message - the voice driver SETHOOK function has been invoked
LOGFUNCMSG

voice onhook

Voice driver message - the voice driver SETHOOK function has been invoked
LOGFUNCMSG

voice getdtmf

Voice driver message - the voice driver GETDTMF function has been invoked
LOGFUNCMSG

voice clrdtmf

Voice driver message - the voice driver CLRDTMF function has been invoked
LOGFUNCMSG

voice stop

Voice driver message - the voice driver CHSTOP function has been invoked
LOGFUNCMSG

voice seize

Voice driver message - the voice driver SEIZE function has been invoked
LOGFUNCMSG

voice release

Voice driver message - the voice driver RELEASE function has been invoked
LOGFUNCMSG

voice setch

Voice driver message - the voice driver SETCH function has been invoked
LOGFUNCMSG

voice call failed (error code)

Voice driver message - the voice driver CALL function failed
ERRORMSG

voice dial failed (error code)

Voice driver message - the voice driver DIAL function failed
ERRORMSG

voice getdtmfstring failed (error code)

Voice driver message - the voice driver GETDTMFSTRING function failed
ERRORMSG

voice play failed (error code)

Voice driver message - the voice driver PLAY function failed
ERRORMSG

voice record failed (error code)

Voice driver message - the voice driver RECORD function failed
ERRORMSG

voice offhook failed (error code)

Voice driver message - the voice driver SETHOOK function failed
ERRORMSG

voice onhook failed (error code)

Voice driver message - the voice driver SETHOOK function failed
ERRORMSG

voice stop failed (error code)

Voice driver message - the voice driver CHSTOP function failed
ERRORMSG

voice seize failed (error code)

Voice driver message - the voice driver SEIZE function failed. Announce may already have this line open.
ERRORMSG

voice release failed (error code)

voice driver message - the voice driver RELEASE function failed
ERRORMSG

voice setch failed (error code)

Voice driver message - the voice driver SETCH function failed
ERRORMSG

startup check failed in control (controlname)

A control or controls failed the startup checks. Refer to the error messages listed for more information. The system will not start until all control have been checked for setup errors, and the whole system checked for integrity.
ERRORMSG

starting test in control (controlname)

CALLFLOW

no connection on node (nodename): playing invalid entry greeting

A control has exceeded it's error count. The error output is not connected so VBVoice is playing the default error greeting prior to handing up.
CALLFLOW

inter call delay

the current call has completed and the LineGroup control that owns this line has been set to start another call after a delay. It has just started the delay timer.
INFOMSG

channel idle

A call has been completed and the LineGroup control that owns this line has gone back to idle mode.
INFOMSG

waiting for ring

A call has been completed and the LineGroup control that owns this line has gone back to waiting for incoming ring.
INFOMSG

playing onhold music

The Delay control has started the Delay and is playing on-hold music to the caller.
INFOMSG

VB stopped delay

VB code terminated the delay wait period in a Delay control by setting the DelayTime property
INFOMSG

loop current drop: call terminated

A loop current drop was reported by the voice card driver. This is usually due to the caller hanging up. If it occurs just after starting a call or answering a call, you may need to set the delay between going off-hook and checking for loop current. This can be changed using the INI setting OffHook_Delay - check appendix in User's Guide for more details.
CALLFLOW

bad date specification (date string)

Generated by phrase processor in System phrase - Say Date
ERRORMSG

property name not found in <control>

A control name was referenced in a property page field for substitution ie using %Controlname% syntax. The control was found but it does not contain a property of the name given.

stopped - VB in break mode

Visual Basic was stopped after hitting a breakpoint or error. The voice system was stopped

bad digits in phrase 'phrase name

The system was using SayDigits when it encountered an invalid character in the string for this phrase type

field not found (<field name>)

The requested field in a database operation was not found among the list of fields in the table selecged

dispatch error <error code> <description>'

An error occurred while using OLE Automation to access an OLE object.

phrase requires language control event: VB not in run mode

A phrase was used which has been enabled in a Language control for custom processing. VBVoice was unable to fire the event because VB was not in run mode

cannot open ODBC database '<database name>'

A DataFind control attempted to open an ODBC database, but the call to ODBC failed. Check the ODBC Connect string.

no recordset defined: use SetRecSource method

A DataFind control was set to open a Recordset using code, however no recordset was assigned using the SetRecSource method

cannot play VB Phrase in design mode

A System phrase of type VB Phrase was specified in a greeting. VBVoice was unable to play the phrase because VB was is design mode. Put VB into run mode and try again

trappable database error <code> '<description>'

An error occurred while accessing a recordset provided by code, using the Microsoft DAO interface.

control error type <type #> data <##>

An unexpected error occurred. If this message appears without any other explanatory messages, please send us a copy of the log, together with a description of what was occuring at the time.

odbc error <code> < description>

An error occurred while accessing a recordset provided by ODBC. The description should give a clue about the problem, if not check the ODBC error codes in Appendix 2 of the manual.

data control error <code> < description>

An error occurred while accessing a recordset provided by a Visual Basic data control. The description should give a clue about the problem, if not check the error code against the trappable databaseerror codes in the VB help file, and also the error codes provided in in Appendix 2 of the manual.

sys start

The voice engine has been started by the Start System menu command or by the DLL function vbv_start_system
CALLS

sys stop

The voice engine has been started by the Stop System menu command or by the DLL function vbv_stop_system
CALLS

call start

A call has been started, initiated eithe by incoming ring, startcall after delay, or by VB code starting a call.
CALLS

call stop

"CALL STOP"
A call has been terminated and the line is back on-hook
CALLS

waiting for dialtone

The Dial control is waiting for dialtone before continuing to dial.
INFOMSG

updating db action

Used in PlayMsgs control
INFOMSG

get next msg

Used in PlayMsgs control to log message handling
INFOMSG

update num msgs

Used in PlayMsgs control to log message handling
INFOMSG

cannot create backup file

Used in PlayMsgs control to log message handling
ERRORMSG

mailbox file open error (maibox file)

PlayMsgs control could not open the specified mailbox file.
ERRORMSG

mailbox file write error

PlayMsgs control opened the mailbox file for read but could not write to it.
ERRORMSG

mailbox file invalid entry

PlayMsgs control could not make sense of the data in the specified mailbox file. The mailbox file has become corrupted.
ERRORMSG

voice get car failed

Voice driver error.
ERRORMSG

invalid user phrase

VB code tried to set a new phrase in a greeting but passed in an invalid phase object. Either the parameter passed in was not created using one of the vbv_create_phrase functions, or it has already been deleted.
ERRORMSG

cannot start test in VB break mode

You can start a test when Visual Basic is in design mode (with some limitations, see User's Guide, chapter on Test Mode), and also in Visual Basic run mode. You cannot start a test when in break mode - either stop Visual Basic and return to design mode, or start Visual Basic and return to run mode
ERRORMSG

vap phrase not found

The phrase name could not be found in the VAP phrase. Check that the script specified matches exactly the contents of the VAP file. Scrip matching is case sensitive.

vap phrase size was zero

Phrases selected for playing must have voice data.

file open error

The specified file could not be opened. Check the path (if supplied) is correct. If no path is supplied, the file must exist in the VBV default directory.

file size error

Vox files must be larger than 1000 bytes (250 ms) or VBVoice will not play them

no phrase data

The phrase requires parameters which are not set.

call terminated

A call has been terminated by the control.

warning: cannot generate VB event if not in run mode

The operation cannot be performed because Visual Basic is in design mode. Controls that rely on Visual Basic, such as Condition events in GetDigits, the User control or database controls do not work in design mode. Note that other controls can be tested in design mode, but that any attached code will not run until Visual Basic is put into run mode.

phone control is in idle mode, ring event ignored

An incoming ring occurred on a line, but the LineGroupcontrol that owns this line was in Idle mode, and ignored the Ring event.
INFOMSG

INVALIDDESTNODE

VBVoice cannot evaluate the destination connected to this output node. Please call Tech Support.

file not found

The specified file could not be opened. Check the path (if supplied) is correct. If no path is supplied, the file must exist in the VBV default directory.

file error

The indicated file exists, but an error occurred while trying to access the file.

phrase error in phrase

A phrase has failed the internal checks. Check the parameters supplied with the phrase to ensure they are valid, that the phrase indicated is contained in the file, and that the file exists in the path specified.

BADBLOCKID

The destination control for an output node could not be found. If this is an OutConn control, then the control must be connected to an InConn control.

If this message occurs during system check at design time, ensure that both source and destination forms are displayed.

If the message occurs during system start, ensure that all VBVoice forms are loaded by Visual Basic before calling System Start. This can be done using Show statements in the Form.Load procedure for the main form, or Sub Main as set in Project | Options,

node not connected

As a Warning:

An output node of this control is not connected. Although it is not essential that this node be connected, ensure that this is what you want to do. Invalid digits, silence timeout and 'not found' nodes will invoke default error handlers if a call is transferred to one of these unconnected nodes. In the LineGroup control, the Start Call and Ring nodes may be left unconnected if the corresponding 'Wait for Ring' and 'Start Call' are not set. note, however that attempting to set these nodes at runtime will cause an error if they are not connected.

As an Error:

The system requires that this node be connected to another control.

no time period name

CHECKFAIL
"node %i, no time period name", get_node_name(node));
Each time period in a list of time periods must be named. Use the TimeSwitch setup property page to add a name to each time period in the list.

no field specified

In the DataFind control, the Match All check box is not checked, but a database field to do a data search in has not been specified. Use the setup dialog to choose a field to search.
In the DataSwitch control, you must specify the database field from which to access the data
CHECKFAIL

no datasearch specified

In the DataFind control, the Match All check box is not checked, and a field to do a data search in has been specified, but there is no data to compare.
CHECKFAIL
"no datasearch specified"

no database table specified

In the PlayMsgs control, you must select which table of messages to access. This is normally based on a mailbox number entered by the caller and referred to using Property Substitution.

In the DataFind controls, a data control has not been selected in which to search for data or in which to add a new record
CHECKFAIL

feedback warning

An output node has been connected back to the input on the same control. This can cause an infinite loop to occur, or a channel to hang. Please ensure this is what you really want to do.

CHECKWARN

no valid termination specified

The GetDigits control cannot calculate valid digit termination conditions from the digit masks supplied. Add a silence timeout, or terminating digit in the GetDigits Terminations property page.

CHECKWARN

digit collection will terminate on silence

The GetDigits control cannot calculate valid digit termination conditions from the digit masks supplied, and will be forced to use silence timeout only. If possible, add a terminating digit in the Terminations property page to improve response time.

max keys too small for node

The maximum number of keys specified is less than that required for the digitmask specified. For instance, if a digit mask contains nnn (3 numeric digits), and Maximum digits is 2, this error will occur. Change the digit mask or maximum digits as appropriate.
CHECKFAIL

cannot use wild chars in range

An invalid digit mask has been entered. The hyphen operator must be used only in conjuction with numeric digits i.e. 100-134.

You also cannot use control names in range. VBVoice does not support control names embedded into a range. Use a VB condition and add some code to do your digit matching
CHECKFAIL

range min and max must have the same number of digits

i.e 11 -23 is ok
234 - 567 is ok
23 - 567 is not ok: use 2 separate masks 23-99 and 100 - 567
CHECKFAIL

phrase error in phrase msg_recorded

the record.vap phrase does not contain the phrase your message has been recorded
CHECKFAIL,

no record filename set

The record control must have a filename set in the required format. This filename can either be a fixed file name or can contain a *, which will create a unique filename at runtime, and Control names, which will be replaced by the value of the control at run time
CHECKFAIL

no filename specified

The IniSwitch control must have a **fieldname** specified. A .INI settting consists of 3 items, the .INI filename, the section name (the part enclosed by []), and the field name:
i
filename VBVOICE.INI
sectionname VoiceCard
fieldname Dialogic
CHECKFAIL

no condition specified on node

A DataSwitch or GetDigits control has a condition with a name but an empty digitmask. Each condition must have a valid string of digits or special characters.
CHECKFAIL

no DataFind block found

A DataSwitch or DataChange control must be connected to a DataFind control, either directly or via another control..
CHECKFAIL

data control not found

A DataFind control must be linked with a Visual Basic data control using the setup dialog.
CHECKFAIL

2 controls of name

It is possible in Visual Basic to have 2 controls of the same name on different forms. VBVoice likes to have unique names for all it's controls. Rename the control indicated to a new name. If you have other controls which reference this control, ensure these are also updated.

CHECKFAIL

Line limit exceeded

The application is not running in the Visual Basic environment, and the line number set in the LineGroup control exceeds the number of lines that this system has been authorized to build. For instance, a 4 line license allows the use of channesl 1-4 only. If you wish to build bigger systems, contact PRONEXUS on 613 839 0033 for an upgrade or contact your authorized distributor.

no fielddata specified

In the DataChange control, a field name has been selected for update, but no new data is attached.

could not access Data Control fields

An error occurred while trying to retrieve the field names from the data control. Check that the Databasename and RecordSource properties are valid, and that the database is available.

no fieldname specified

The DataFind control was set to use data matching, but no field name was specified for the condition.
In the DataFind control, use the Matching property page to either Match on all records, or set a valid field name and search data

invalid match field name

The DataFind control was set to use data matching, but an invalid field name was specified for the condition. Ensure that the field exists in the database, and also appears in the Fields list in the Database property page.

no match data set

The DataFind control was set to use data matching, but no data was specified for the search. There must be a valid field name and a string to search for set in the Matching property page. The search string may contain control and property names.

cannot open database

An unknown error occurred while opening the ODBC database

ODBC open error:<error code>

An error occurred while opening the ODBC database. Check the list of ODBC error codes

no odbc table name set

The DataFind control is set to open an ODBC database, but no table has been specified. Both the database name and the table to open must be specified in the Database property page

no fields specified

In order for the DataFind control to access and update data, the fields on which it will operate must be specified in the Fields list in the Database property page. This list can be filled in using the Change Fields button.

ODBC connect string is empty

The DataFind control is set to open an ODBC database, but no connect string for the database has been specified. Both the database name and the table to open must be specified in the Database property page

invalid match field name

The DataFind control was set to use data matching, but the field name specified was not found in the list of available fields. Ensure that the field name is valid, and that it has been added to the list of available fields in the Database property page.

must be at least one phrase in music greeting

When the Delay control has been set to play the MusicGreeting during the delay, it will play the greeting until the delay times out or is terminated by StopDelay. It must have at least one phrase in the greeting in order to do this.

no dial number specified

The Dial control does not have a valid number specified. Even if you are going to override the number in code, put a dial number in the Number to dial field in the Dial setup property page

entry greeting is not empty in a dial control without transfer set

In the Dial control, the EntryGreeting can be used to warn a caller that a transfer is going to be attempted. When using the Dial control to make an outgoing call, it is usually an error to play a greeting.

help digit handler specified, but help digit is disabled

In the GetDigits control, it is possible to set a specific handler for Help digits using the Errors property page. It is also possible to disable the help digit while in this control (for instance when you are collecting a number that may incldue the help digit. These 2 options are mutually exclusive.

no mailbox directory set

The PlayMsgs control must have a directory set in which it will look for messages. This directory is normally constructed using a mailbox number entered by the caller, along with a partial path name

cannot find message phrase in message greeting: ff and rew will not work

When setting up a message greeting in the PlayMsgs control, the control expects to find a phrase of type WaveFile, with the path pointing to %this.MsgFile%. This is the phrase that the rewind and fast-forward keys will operate on. This warning is to indicate that VBVoice could not find a phrase of this type.

time format error in <name>

VBVoice could not parse the time specified in the named time. Check the time using the TimeSwitch property page.

ambiguous control name: <name>. VBVoice controls must have unique names

A field specified a control name, but VBVoice found 2 controls of that name in the system (Controls on different forms can have the same name). Rename one of the controls to avoid this problem.

cannot update read-only database

A DataFind control is set to open a database in Read_Only mode, but a DataChange control is attempting to update this database. Change the

no phrases in greeting

A greeting has not been specified. Although the program will work without this greeting, you should make sure this is what you want to do.

No Silence timeout greeting:

The control does not have a silence timeout greeting. this greeting is played when the control times out while waiting for user input (speech or digits). In most cases the control will play the silence timeout greeting, if configured, and then play the entry prompt again. refer to help on the particluar control for more information.

No invalid digits greeting:

The control does not have a invalid digits greeting. This greeting is played when the control receives incorrect digits from the caller. In most cases the control will play the invalid digits greeting, if configured, and then play the entry prompt again. Refer to help on the particluar control for more information.

No entry greeting:

The control does not have an entry greeting. This greeting is played when a call enters the control , and also after a silence or invalid digits greeting (see above). Normally a control will need a greeting to tell the caller what to do.

No options greeting:

The Record control does not have an options greeting. This greeting is played after a recording has been made. If you have not set any options digits, then you may not want to play an options greeting.

data control error <code> <description>

This message is generated when an error occurs while accessing a recordset provided by code. See list of possible error codes at the end of this chapter.

0x4e00	DB_E_BADBINDINFO
0x4e01	DB_E_BADBOOKMARK
0x4e02	DB_E_BADCOLUMNID
0x4e03	DB_E_BADCRITERIA
0x4e04	DB_E_BADENTRYID
0x4e05	DB_E_BADFRACTION
0x4e06	DB_E_BADINDEXID
0x4e07	DB_E_BADQUERYSPEC
0x4e08	DB_E_BADSORTORDER
0x4e09	DB_E_BADVALUES
0x4e0a	DB_E_CANTCOERCE
0x4e0b	DB_E_CANTLOCK
0x4e0c	DB_E_COLUMNUNAVAILABLE
0x4e0d	DB_E_DATACHANGED
0x4e0e	DB_E_INVALIDCOLUMNORDINAL
0x4e0f	DB_E_INVALIDINTERFACE
0x4e10	DB_E_LOCKFAILED
0x4e11	DB_E_ROWDELETED
0x4e12	DB_E_ROWTOOSHORT
0x4e13	DB_E_SCHEMAVIOLATION
0x4e14	DB_E_SEEKKINDNOTSUPPORTED
0x4e15	DB_E_UPDATEINPROGRESS
0x4e16	DB_E_USEENTRYID
0x4e17	DB_E_STATEERROR
0x4e18	DB_E_BADFETCHINFO
0x4e19	DB_E_NOASYNC
0x4e1a	DB_E_ENTRYIDOPEN
0x4e1b	DB_E_BUFFERTOOSMALL
0x4ec0	DB_S_BUFFERTOOSMALL
0x4ec1	DB_S_CANCEL
0x4ec2	DB_S_DATACHANGED
0x4ec3	DB_S_ENDOFCURSOR
0x4ec4	DB_S_ENDOFRESULTSET
0x4ec5	DB_S_OPERATIONCANCELLED
0x4ec6	DB_S_QUERYINTERFACE
0x4ec7	DB_S_WORKINGASYNC
0x4ec9	DB_S_MOVEDTOFIRST
0x4eca	DB_S_CURRENTROWUNCHANGED
0x4ecb	DB_S_ROWADDED
0x4ecc	DB_S_ROWUPDATED
0x4ecd	DB_S_ROWDELETED

ODBC database error: <code> <description>

An error occurred while accessing the ODBC driver. See below for a list of possible error codes.

1000	SQL_ERROR
1001	SQL_ERROR_CONNECT_FAIL
1002	SQL_ERROR_RECORDSET_FORWARD_ONLY
1003	SQL_ERROR_EMPTY_COLUMN_LIST
1004	SQL_ERROR_FIELD_SCHEMA_MISMATCH
1005	SQL_ERROR_ILLEGAL_MODE
1006	SQL_ERROR_MULTIPLE_ROWS_AFFECTED
1007	SQL_ERROR_NO_CURRENT_RECORD
1008	SQL_ERROR_NO_ROWS_AFFECTED
1009	SQL_ERROR_RECORDSET_READONLY
1010	SQL_ERROR_SQL_NO_TOTAL
1011	SQL_ERROR_ODBC_LOAD_FAILED
1012	SQL_ERROR_DYNASET_NOT_SUPPORTED
1013	SQL_ERROR_SNAPSHOT_NOT_SUPPORTED
1014	SQL_ERROR_API_CONFORMANCE
1015	SQL_ERROR_SQL_CONFORMANCE
1016	SQL_ERROR_NO_DATA_FOUND
1017	SQL_ERROR_ROW_UPDATE_NOT_SUPPORTED
1018	SQL_ERROR_ODBC_V2_REQUIRED
1019	SQL_ERROR_NO_POSITIONED_UPDATES
1020	SQL_ERROR_LOCK_MODE_NOT_SUPPORTED
1021	SQL_ERROR_DATA_TRUNCATED
1022	SQL_ERROR_ROW_FETCH
1023	SQL_ERROR_INCORRECT_ODBC
1024	SQL_ERROR_UPDATE_DELETE_FAILED
1025	SQL_ERROR_DYNAMIC_CURSOR_NOT_SUPPORTED
1026	SQL_ERROR_MAX

silence timeout = 0 but max keys > 1

In the GetDigits control, you have set the silence timeout to 0. This means that the caller is not given any time after the greeting to enter digits. However the greeting is set to terminate on the first digit, so the maximum possible number of digits that can be entered is 1. There is at least one digit mask that requires more than one digit or the max keys field is set to more than 1. Therefore there is a conflict: either increase the silence timeout, or remove the requirement for more than 1 digit as a termination.

Time Set Dialog

This dialog appears when in test mode and a TimeSwitch control is encountered. It allows you to enter the time to be used when evaluating the branch condition. Using this dialog you can test all possible branches regardless of the actual time. The default time is the actual time.

Get Value Dialog

This dialog appears in test mode, when a control attempts to get a value from another control which has not been entered. This will happen if you have started a test from a point which has bypassed some earlier controls. Since the current control requires a value from the control specified, you must enter the value yourself.

This can be useful if you want to test different results from a database without having to search for the data you want to test with. If you select cancel, a runtime error occurs and the test will terminate.

Set Page Name dialog

This dialog is used to set the name of a page in a frame control. These names are used for desciptive purposes only and are not used directly by VBVoice.

New VAP Filename dialog

Enter the name of the new VAP file name here. The file will be created in the VBVoice directory for this project.

New Transfer property name dialog

Enter the name of a new transfer property. This name can be any alphanumeric string. Each property name in a VBVFrame control should be unique.

New control name dialog

Use this dialog to rename a control. Valid control names are 40 characters or less.

Add Control without Frame dialog

VBVoice controls must be contained within a VBVFrame control. The VBVFrame control is used to draw the interconnecting lines between the controls, and also provides methods for starting and stopping the voice system.
To do this, follow these steps:

1) Remove the control that was added
2) Add a frame control (see icon below) or select an existing frame control
3) Select a page other than the outline (i.e page 1)
4) Select the new VBVoice control and draw it on the frame by clicking and dragging

The VBVFrame control icon

Field List dialog

This dialog provides a list of fields in the currently selected database, if available. The DataFind control will only retrieve the values for fields that are selected from this dialog.
To create the list of fields to retrieve, select fields in the 'Available' list and use the << key to move them to the 'Used' list.

IniSwitch condition dialog

This dialog is used to set a value against which the actual value of an Ini file setting will be compared. The value can contain control names and control properties using the %..% syntax

User Condition dialog

This dialog is used to set the name of an output of a User control. These names are used for graphical display purposes only. The outputs are referred by by their position by VB code ,

DataChange condition

This dialog is used to set the new value for a field in a DataFind control. The new value can contain control names and control properties using the %..% syntax

Wait for shutdown Dialog

Authorization dialog

Call Result dialog

When running in test mode, VBVoice does not do dialing (since a connection is already established). This dialog allows you to select which path the application should take during this test. This also allows you to test all paths in your application without having to re-create the actual dialing conditions for Busy, Intercept etc.

Ring or Call dialog

When starting a test from a LineGroup control, the system cannot tell if you want to run a test based on an incoming call, or based on a system-initiated call. This dialog is used to choose which route to take from the LineGroup control - the Ring output or the StartCall output

Connecting controls using the dialog

To make a named connection, click on the output node of the control and drag to a blank part of the form, and release. A dialog will pop up showing all the currently available controls on this form.
(You must have already done this if you got here from pressing F1)
If a connection is already made from this node, it will be highlighted. Note forms must be displayed (either normal size or minimized) before their controls become available in this dialog. Use the Connect button to make a named connection, or the Disconnect button to remove an existing connection without replacing it.
Only controls with input connections are shown (i.e. LineGroup controls are excluded)

In large projects you can zero in on the required control by first selecting the frame and **page**which contains the control.

Note:
Forms must be displayed (normal size or minimized) before their controls become available on the dialog.

Deleting connections:

To delete a connection from one control to another without replacing it, repeat the actions for creating a named connection to get the Connection dialog, and then click Disconnect.

Page

Part of the VBV Frame control. In projects, most of the time, you will place other VBVoice controls on pages.

see also Add Control without Frame dialog

Choosing a destination control

Use this dialog to choose a control which will handle the error or event selected. In large projects you can zero in on the required control by first selecting the frame and **page** which contains the control.
Linegroup controls are excluded from this list.
Note:
Forms must be displayed (normal size or minimized) before their controls become available on the dialog.

Start Test in dialog

Use this dialog to select a control at which to start a test. A test can be started in any control. . In large projects you can zero in on the required control by first selecting the frame and **page** which contains the control.

See also: Test mode

Note:
Forms must be displayed (normal size or minimized) before their controls become available on the dialog.

Vcomments

Use this property page to document any settings of the particular control.
The default text states: "Use this page to document the configuration for this control", which of course is only a reminder of what this property is for and you can change it as desired.

Maximum silence

(IMaxSil property)

This field specifies the number of seconds that GetDigits will wait for a digit. If a digit is not received in the time, digit collection will be terminated. A value of 0 means that the control will not wait for digits to arrive after the greeting has played.
or:
This field sets the maximum period of silence before terminating a record operation.

Trappable Errors

These errors occur when your code attempts to set a property at an invalid time, or uses invalid data. These errors should be intercepted with a Visual Basic On Error... statement, so your code can deal with the error. Trappable errors not handled in this way will cause Visual Basic to break into design mode if in the design environment, or if running in a compiled application, your application will terminate, with a message “User Defined Error”. The error code is accessible using the built-in Err variable.

Example

```
On Error Goto ErrorFound
GetDigits1.GotoNode(5) = 5
Exit Sub
ErrorFound:
Debug.Print “Error detected ! “ + Str$(Err)
End Sub
```

Possible error codes

Name	Value	Description
vbvErrBadNode	32000	Attempt to set the GotoNode property to an invalid number, or node not connected.
vbvErrBadControlName	32001	Attempt to set the GotoControl property to an unknown control name
vbvErrBadParm	32002	Attempt to set a property to an invalid value. See description of property in question
vbvErrBadDial	32003	Attempt to set the NumToDial property to an number containing invalid digits or number too long.
vbvErrCallNotHere	32005	Attempt to set a property outside of the Enter event, when the property can only be set from Enter event
vbvErrWrongState	32006	Control cannot accept property change at this time
vbvErrFuncFailed	32007	Voice driver function failed during a set property
vbvErrSysNotStarted	32008	Attempt to set a property that is only available after the voice engine has been started.
vbvErrNoConnection	32009	Attempt to set the mode property in the LineGroup control, when a valid connection for the output associated with that mode does not exist.
vbvErrPhraseNotAllowed	32010	Cannot use this phrase type in a phrase object
vbvErrAddPhrs	32011	
vbvErrPhraseNotFound	32012	
vbvErrNoNotify	32014	Conference control
vbvErrBadNotify	32015	
vbvErrSysStarted	32016	Operation cannot be competed while system is started
vbvErrCannotSetODBC	32017	ODBC error
vbvErrQueryIF_Failed	32018	
vbvErrLineInUse	32019	LineGroup - line is already in use by another control
vbvErrNoLine	32020	LineGroup - attempt to remove line that is not allocated
System Startup Errors		
vbvErrInTestMode	32100	Cannot start system if test is running
vbvErrSystemStarted	32101	System is already started
vbvErrNotInRunMode	32102	VB is not in run mode
vbvErrCannotLoadVoiceCard	32103	Voice card driver cannot be loaded
vbvErrSystemErrors	32104	Errors were detected during check system
vbvErrPhoneInitFail	32105	Errors were detected initializing the lines
vbvErrNoFrame	32106	This project has no frame control

Other errors: - standard VB trappable errors:

Invalid Property Index	returned when an invalid channel is referenced.

112 John Cavanagh Road.
RR#2 CARP, Ontario
K0A 1L0 CANADA

tel.: +1 (613) 839-0033
fax.: +1 (613) 839-0035
BBS: +1 (613) 839-0034

http://www.pronexus.com
ftp://ftp.pronexus.com
CIS: 71054.3225@compuserve.com
email: support@pronexus.com

Technical Support

There are several ways you can contact us for support.
First, please ensure that the information you are looking for is not in the manual or in the help file.
If you have a error message that you don't understand, select it and press F1. If none of those help you to solve the problem, please check the message on the Bbs for a problem similar to yours. It is likely that the problem you are having has been experienced by someone else, and the solution has already been posted. If this doesn't help then please:

1. leave message on the BBS
2. send an e-mail
3. send us a fax

Before contacting us, please be sure to give us the following information:

1. All events leading up to the problem
2. Voice or Fax card type
3. Computer type
4. Full and complete information about the problem: does the problem occur in test mode, with a sound card or voice card, in the design environment, or in a .EXE?
5. A copy of the log for the channel causing the error. Either the binary .vlg file, or a text file saved using the 'Save to Text File' option in the channel log window.