

# BROADCAST CLIENT SDK METHOD DETAILS

---

This document specifies the Status and Error Code Numbers and also describes the events and functions supported by VaxVoice Broadcast SDK client Components:

## CODE NUMBERS

- Error Codes
- Status Codes

## EVENTS

- OnStatusEvent
- OnServerTextEvent
- OnTextEvent
- OnAcceptConnectionEvent
- OnRejectConnectionEvent

## METHODS

1. Initialize()
2. UnInitialize()
3. Connect()
4. Disconnect()
5. EnableCryptography()
6. DisableCryptography()
7. SetVAD
8. GetVaxObjectError()
9. SetLicenceKey()
10. SetSpkVolume()
11. GetSpkVolume()
12. SetMicVolume()
13. GetMicVolume()
14. MuteMic()
15. MuteSpk()
16. StartRecording()
17. StopRecording()
18. ResetRecording()
19. SaveRecordingToWaveFile()
20. EnableMicBoost()
21. DisableMicBoost()
22. IsMicBoostEnable()
23. SendTextMsg()
24. SendTextToServer()
25. GetOutBoundTotalBytes()
26. GetInBoundTotalBytes()
27. GetOutBoundDataRate()
28. GetInBoundDataRate()

**ERROR CODES DETAIL:**

<b>ERROR CODES</b>	<b>DESCRIPTION</b>
10	VAXOBJECT is not initialized, To initialize, the VaxVoice Object <i>Initialize</i> method should be called.
11	Cannot access the input device (Microphone) OR input device is already in use.
12	Cannot access the output device.
13	Cannot open local communication port, port is invalid or is already in use.
14	Provided License Key is invalid.
15	Recording media initialization error or cannot open the recording tmp file.
16	Cannot open the channel to start recording.
17	Unable to open the wave file to save the recoded voice data
18	Voice Compression Manager initialization failed
19	Not connected to the Broadcast Server.
20	Task Manager initialization failed.
21	Cryptography key is not provided while initializing the VaxVoice Object. Please see the <i>Initialize</i> method for further details.
22	Fail to access the Mic/Input device Volume OR Sound device does not support Mic volume feature.
23	Fail to access the Speaker/Output device Volume OR Sound device does not support Speaker volume feature.

## STATUS CODES DETAIL:

STATUS CODES	DESCRIPTION
50	Fail to connect to the Broadcast Server.
51	Trying to connect to Broadcast Server.
52	Connection to the Broadcast Server is Lost.
53	Connection closed by the Broadcast server.
54	Server has exceeded the connections limit.
55	You are blocked on the server side.
56	This status code notifies that Encrypted data is being received from the remote end and VaxVoice object is unable to decrypt it because Cryptography key is not provided while initializing the VaxVoice Object. Please see the <i>Initialize</i> method for further details.

## **EVENTS:**

### **OnStatusEvent**

This Event is triggered by the VaxVoice control, to notify about the connection status.

#### ***Parameters Values:***

- Status Code Number

### **OnServerTextEvent**

This event is triggered, when text message is received from Broadcast server.

#### ***Parameters Values:***

- Text Message

### **OnTextEvent**

This Event is triggered by the VaxVoice control, when a BROADCASTED text message along with the any text data is received from the remote end.

#### ***Parameters Values:***

- Text Message
- User Data (any text based information.)

### **OnAcceptConnectionEvent**

It notifies that the connection request is accepted on broadcast server end.

#### **Parameters Values:**

- User Data (any text based information sent by the server)

### **OnRejectConnectionEvent**

It notifies that the connection request is rejected by the broadcast server.

#### **Parameters Values:**

- User Data (any text based information sent by the server)
- CustomErrorCode (error code, which is sent from the server end)

## **METHODS:**

### **Initialize()**

This method is called to initialize the VaxVoice control. It requires Port Number to listen for incoming connections and/or data. After initializing, VaxVoice control starts listening for incoming Connections.

It also requires the cryptography key or secret words to decrypt the incoming data and/or Encrypt the outgoing Voice/Text data. BLOW FISH cryptography is used for Encryption/Decryption.

Two methods *EnableCryptography* and *DisableCryptography* can be used to enable/disable the encryption on outbound Voice Stream and Text Messages.

Please see the sample source code for more details.

#### **Parameters:**

- ListenPort
- InputDeviceId (-1 = auto select)
- CryptographyKey or Secret words

#### **Return Value:**

Non-zero on success, otherwise 0, and a appropriate error code can be retrieved by calling *GetVaxObjectError()* method.

#### **Remarks:**

Value -1 can be provided, if your computer has single sound device OR you want VaxVoice control to select the first/default sound device capable of recording.

VaxVoice controls use '*waveInOpen*' windows API to initialize the Input device.

If you want to run multiple instances of your client software and your computer has multiple sound devices then WinMM (Windows Multimedia API) can be used to determine the device-Ids.

VaxVoice controls use the following PCM Format to initialize the input device:

FormatTag	=	1
Channels	=	1
SamplesPerSec	=	8000
AvgBytesPerSec	=	16000
BlockAlign	=	2
BitsPerSample	=	16
cbSize	=	0

The following WinMM APIs can be used to determine the input device-Id.

- waveInGetNumDevs()
- waveInGetID()
- waveInGetDevCaps()

### **UnInitialize()**

To uninitialized the VaxVoice control, this method can be used.

### **Connect()**

This method is called to send connection request to conferencing server.

#### ***Parameters:***

- Chat Room
- IP Address of Conferencing Server
- Port Number of Conferencing Server
- Login
- Password
- User Data
- TimeOut to connect in seconds

#### ***Return Value:***

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

#### ***Remarks:***

This method tries to establish the connection to the Broadcast server and the Status Codes triggers accordingly.

Login & password provided to this method is received on Server in *OnClientConnectionEvent* event.

User Data is any text data, which can be sent from client to server and is received on server in *OnClientConnectionEvent* event.

Chat room name can also be specified in this method.

### **Disconnect()**

Calling this method closes the connection to the Server

#### ***Return Value:***

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

### **EnableCryptography()**

To enable the Encryption on outgoing Voice Stream and Text data. When this method is called, VaxVoice component starts encrypting the outbound Voice Stream and Text messages before sending it to the Remote end over the internet.

Remote end receives the encrypted Voice Stream/Text messages and decrypt it using the provided Cryptography key. So on both ends the Cryptography keys must be the same.

BLOW FISH cryptography is used for Encryption/Decryption.

### **DisableCryptography()**

To disable the Cryptography feature on outgoing Voice Stream and Text data. When this method is called, VaxVoice component stops encrypting the Voice Stream and Text messages and starts sending the plain data to the remote end over the Internet.

### **SetVAD()**

VaxBroadcast client SDK works in VAD (Voice Activity Detection) mode. VAD values can be set using this method. In this feature, VaxVoice control sends voice digital data only when you speak.

#### **Parameter:**

- Mic Sensitivity, any value from 0 to 255.
- Silence Duration in seconds.

#### **Return Value:**

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

#### **Remarks:**

Normally, you can set value 170 for Mic Sensitivity and Silence Duration: 5 seconds.

### **GetVaxObjectError()**

Call this method, to get the error for the last operation that failed.

#### **Return Value:**

- Error Code Number

### **SetLicenceKey()**

Call this method, to set the License key.

#### **Parameter:**

- License Key Provided by VaxVoice

#### **Return Value:**

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

#### **Remarks:**

You must pay one-time License fee in order to get the License Key. After getting the License key, you will set it using this method and it will remove the evaluation message box & expiry.

### **SetSpkVolume()**

To set the Output volume, this method can be called. the value range should be between 0 to 255.

#### **Parameter:**

- Volume Value between [0 – 255] Range

#### **Return Value:**

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

### **GetSpkVolume()**

Call to this method returns the speaker volume value between Range [0 to 255], where

0 = Min Volume

255 = Max Volume

#### **Return Value:**

Speaker Volume value on success, otherwise -1 and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

### **SetMicVolume()**

Call the method to set the Microphone Volume, the value range should between 0 to 255, where

0 = Min Volume

255 = Max Volume

#### **Parameter:**

- Volume Value between [0 – 255] Range

#### **Return Value:**

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

### **GetMicVolume()**

Call the method to get the Microphone Volume. Volume value is returned, in the range of [0 to 255], where

0 = Min Volume

255 = Max Volume

#### **Return Value:**

Microphone Volume value on success, otherwise -1 and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

### **MuteSpk()**

To mute the speaker, this method can be called. Muting the speaker does not affect the Master Mute Control.

#### **Parameter:**

- Boolean Value 0 or 1

#### **Return Value:**

Non-Zero on success, otherwise 0

### **MuteMic()**

To mute the microphone, this method can be called. Muting the Microphone does not affect the Master Mute Control.

#### **Parameter:**

- Boolean Value 0 or 1

#### **Return Value:**

Non-Zero on success, otherwise 0

### **StartRecording()**

This method is used to start the conversation recording.

#### **Remarks:**

VaxVoice component creates tmp file for buffering purposes or to store the digital data. When this method is called, VaxVoice component starts storing data into that tmp file.

### **StopRecording()**

This method is used to stop the conversation recording.

#### **Remarks:**

Call this method to stop storing data into recording tmp file.

### **ResetRecording()**

To reset/clear the recording buffer, this method can be called.

#### **Remarks:**

Call to this method clears all the saved digital data from the recording tmp file.

### **SaveRecordingToWaveFile()**

To save the recording tmp file voice data into wave (.wav) file.

#### **Parameter:**

- File name (.wav)

#### **Remarks:**

Call to this method saves the tmp voice data into wave (.wav) file.

### **EnableMicBoost()**

Call the method to increase the Microphone sensitivity.

### **DisableMicBoost()**

To disable the Mic Boost.

### **IsMicBoostEnable()**

Call the method to get Mic Boost is enabled or disabled.

### **SendTextMsg()**

Call to this method, sends Text Message to all clients connected to the Server.

#### **Parameter:**

- Text Message
- User Data

#### **Return Value:**

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

#### **Remarks:**

User data can be any string/text data (Login, user name, phone number, email etc), which can be sent along with the Text message.

### **SendTextToServer()**

To send text message to Broadcast Server.

#### **Parameter:**

- Server IP
- Server Port
- Text Message

#### **Return Value:**

Non-zero on success, otherwise 0, and a specific error code can be retrieved by calling *GetVaxObjectError()* method.

### **GetOutBoundTotalBytes()**

Total Outbound bytes, since the VaxVoice Object is initialized.

### **GetInBoundTotalBytes()**

Total Inbound bytes, since the VaxVoice object is initialized.

### **GetOutBoundDataRate()**

Outbound Bytes per second.

### **GetInBoundDataRate()**

Inbound Bytes per second.