

# VaxVoip

## SIP SOFTPHONE SDK

**Apple iOS(iPhone, iPad, iPod)**  
**TECHNICAL DOCUMENTATION**

**VERSION 6.0.8.4**

## **CONTENTS**

### **INTRODUCTION AND QUICK START ..... 6**

### **EXPORTED FUNCTIONS ..... 7**

GetVaxErrorCode()	7
GetVaxErrorMsg()	8
Initialize()	9
UnInitialize()	10
AddAccountSIP()	11
RemoveAccountSIP()	13
RegisterAccountSIP()	14
UnRegisterAccountSIP()	15
AudioDeviceVU()	16
GetAudioInDevTotal()	17
GetAudioOutDevTotal()	18
GetAudioOutDevName()	19
GetAudioInDevName()	20
OpenLine()	21
CloseLine()	22
DialCall()	23
DisconnectCall()	25
AcceptCall()	26
RejectCall()	27
TransferCallBlind()	28
TransferCallConsult()	29
HoldLine()	30
UnHoldLine()	31
IsLineOpen()	32
IsLineConnected()	33
IsLineHold()	34
IsLineBusy()	35
IsLinePlaying()	36
IsLineRecording()	37
EnableKeepAlive()	38
DisableKeepAlive()	39
SelectAllAudioCodec()	40
SelectAudioCodec()	41
DeselectAllVideoCodec()	42
DeselectAudioCodec()	43
SelectAllVideoCodec()	44
SelectVideoCodec()	45
DeselectAllVideoCodec()	46
DeselectVideoCodec()	47
DigitDTMF()	48
SetVolumeDTMF()	49
GetVolumeDTMF()	50
MuteMic()	51
MuteSpk()	52
MuteLineSpk()	53

MuteLineMic()	54
AutoGainMic()	55
AutoGainSpk()	56
SetVolumeMic()	57
GetVolumeMic()	58
SetVolumeSpk()	59
GetVolumeSpk()	60
SetLineVolumeSpk()	61
GetLineVolumeSpk()	62
DonotDisturb()	63
StartRecording()	64
StopRecording()	65
PlayWaveOpen()	66
PlayWaveClose()	67
PlayWaveSkipTo	68
PlayWaveTotalTime()	69
PlayWavePause()	70
PlayWaveStart()	71
PlayWaveStop()	72
PlayWavePosition()	73
GetOutboundCodec()	74
GetInboundCodec()	75
AudioSessionLost()	76
SetUserAgentSIP()	77
GetUserAgentSIP()	78
SetSubjectSDP()	79
GetSubjectSDP()	80
ConfAllowLine()	81
ChatAddContact()	82
ChatRemoveContact()	83
ChatFindContact()	84
ChatSendMessageTyping()	85
ChatSendMessageText()	86
ChatSetMyStatus()	87
VoiceChanger()	88
ForwardCall()	89
PlayAddPCM()	90
PlayResetPCM()	91
AddCustomHeader()	92
RemoveCustomHeader()	93
RemoveCustomHeaderAll()	94
GetVideoDevTotal()	95
GetVideoDevName()	96
OpenVideoDev()	97
CloseVideoDev()	98
CryptoMediaNONE()	99
CryptoMediaSDP()	100
DialCallToREC()	101
OpenLineREC()	102
DialRingEnable()	103
DialRingDisable()	104
BusyRingEnable()	105
BusyRingDisable()	106

EnableVideo()	107
GetCallId()	108
IsNetworkAvailable()	109
NetworkReachability()	110
AutoRegistration()	111
VideoCodecBitRate()	112
BusyLampAddContact()	113
BusyLampRemoveContact()	114
BusyLampFindContact()	115
AudioVAD()	116
DiagnosticLogSIP()	117
SetPushSIP()	118

## **EXPORTED EVENTS ..... 119**

OnInitialized()	119
OnUnInitialized()	120
OnConnectingToRegister()	121
OnTryingToRegister()	122
OnFailToRegister()	123
OnSuccessToRegister()	124
OnConnectingToReRegister()	125
OnTryingToReRegister()	126
OnFailToReRegister()	127
OnSuccessToReRegister()	128
OnTryingToUnRegister()	129
OnFailToUnRegister()	130
OnSuccessToUnRegister()	131
OnDialCallStarted()	132
OnDialingCall()	133
OnDialCallFailed()	134
OnConnectedCall()	135
OnHungupCall()	136
OnIncomingCallStarted()	137
OnIncomingCallEnded()	138
OnRingToneStarted()	139
OnRingToneEnded()	140
OnTransferCallAccepted()	141
OnTransferCallFailed()	142
OnPlayWaveDone()	143
OnDigitDTMF()	144
OnVoiceMailMsg()	145
OnIncomingDiagnostic()	146
OnOutgoingDiagnostic()	147
OnAudioSessionLost()	148
OnSuccessToHold()	149
OnTryingToHold()	150
OnFailToHold()	151
OnSuccessToUnHold()	152
OnTryingToUnHold()	153
OnFailToUnHold()	154
OnChatContactStatus()	155
OnChatSendMsgTextSuccess()	156

OnChatSendMsgTextFailed()	157
OnChatSendMsgTypingSuccess()	158
OnChatSendMsgTypingFailed()	159
OnChatRecvMsgText()	160
OnChatRecvMsgTypingStart()	161
OnHoldCall()	163
OnUnHoldCall()	164
OnVideoRemoteStarted()	165
OnVideoRemoteEnded()	166
OnVideoRemoteFrameRGB()	167
OnVideoDeviceFrameRGB()	168
OnAddCallHistory()	169
OnNetworkReachability()	171
OnAudioDeviceMicVU()	172
OnAudioDeviceSpkVU()	173
OnVaxErrorMsg()	174

## **LIST OF ERROR CODES ..... 175**

## **INTRODUCTION AND QUICK START**

The VaxVoIP SIP softphone SDK is a software development kit which is used to quickly embed SIP (Session Initiation Protocol) based softphone features to web and software. It provides full support to tailor the softphones features as desired like having your own GUIs or incorporating your brand name.

## **EXPORTED FUNCTIONS**

### **GetVaxErrorCode()**

The GetVaxErrorCode() method returns the error code for the last operation which is failed to execute.

Please see [LIST OF ERROR CODES](#) for more details.

### **Syntax**

```
integer GetVaxErrorCode()
```

### **Parameters**

No parameters

### **Return Value**

The function returns error code.

### **Example**

```
Result = Initialize()  
If(Result == 0) GetVaxErrorCode()
```

### **See Also**

GetVaxErrorMsg()

**GetVaxErrorMsg()**

The GetVaxErrorMsg() method returns the error text message for the last operation which is failed to execute.

Please see [LIST OF ERROR CODES](#) for more details.

**Syntax**

```
string GetVaxErrorMsg()
```

**Parameters**

No parameters

**Return Value**

The function returns error message text.

**Example**

```
Result = Initialize()  
If(Result == 0) GetVaxErrorMsg()
```

**See Also**

GetVaxErrorCode()



## **Initialize()**

The Initialize() function initializes the VaxVoIP component and once the component is successfully initialized, the user will be able to dial and receive phone calls.

### **Syntax**

```
Boolean Initialize()
```

### **Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### **Example**

```
Result = Initialize()  
If(Result == 0) GetVaxErrorCode()
```

### **See Also**

UnInitialize(), GetVaxErrorCode(), GetVaxErrorMsg()

**UnInitialize()**

The UnInitialize() function releases all the memory/resources that were held during component initialization.

**Syntax**

```
UnInitialize()
```

**Parameters**

No parameters.

**Return Value**

No return value.

**Example**

```
UnInitialize()
```

**See Also**

Initialize()

## AddAccountSIP()

The AddAccountSIP() function adds SIP account settings provided by SIP SERVER or ITSP (IP-Telephony Service Provider).

### Syntax

```
boolean AddAccountSIP(  
    AccountName,  
    AccountType,  
    ListenIP,  
    ListenPort,  
    DisplayName,  
    UserName,  
    AuthLogin,  
    AuthPwd,  
    DomainRealm,  
    ServerAddr,  
    ServerPort  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

AccountType(integer)

This parameter value specifies the account communication type.

0 = Account Type UDP

1 = Account Type TCP

2 = Account Type TLS

ListenIP(string)

The ListenIP parameter value specifies the IP address of machine on which VaxVoIP is running.

ListenPort(integer)

The ListenPort parameter specifies the port number for SIP softphone to receive the requests. If the port is specified as -1, the VaxVoIP assigns a unique port to the application.

DisplayName(string)

This parameter value specifies the display name for user which is provided by IP-Telephony or VoIP service provider otherwise leave it blank.

UserName(string)

This parameter value specifies the user-name which is provided by IP-Telephony or VoIP service provider otherwise leave it blank.

AuthLogin(string)

This parameter value specifies the user Login which is provided by IP-Telephony or VoIP service provider.

AuthPwd(string)

This parameter value specifies the password which is provided by IP-Telephony or VoIP service provider.

DomainRealm(string)

This parameter value specifies the IP/Domain address of the computer on which SIP server is running or provided by IP-Telephony or VoIP service providers.

ServerAddr(string)

This parameter value specifies the IP/Domain address of the computer on which SIP server is running or provided by IP-Telephony or VoIP service providers.

ServerPort(integer)

This parameter value specifies the port of the computer on which SIP server is running or provided by IP-Telephony or VoIP service providers, otherwise default port 5060 can be used.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = AddAccountSIP("TestSIP", 0, "", -1, "", "", "4002", "4002", "",  
                      "demo.vaxvoip.com", 8060)
```

```
If(Result == 0) GetVaxErrorCode()
```

### See Also

RemoveAccountSIP(), GetVaxErrorCode(), GetVaxErrorMsg()

**RemoveAccountSIP()**

The RemoveAccountSIP() function removes the SIP account, which was previously added by AddAccountSIP() function.

**Syntax**

```
void RemoveAccountSIP(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Return Value**

No return value.

**Example**

```
RemoveAccountSIP("TestSIP")
```

**See Also**

AddAccountSIP(), RegisterAccountSIP(), GetVaxErrorCode()

## RegisterAccountSIP()

The RegisterAccountSIP() function registers the client to SIP server. The registration with server is mandatory to receive calls however calls can be dialed without registration.

### Syntax

```
boolean RegisterAccountSIP(AccountName, Expire)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

Expire(integer)

The Expire parameter specifies the time interval after which the registration with server will be refreshed consequently server will remain updated about the present client status.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
RegisterAccountSIP("TestSIP", 1800)
```

### See Also

UnRegisterAccountSIP(), GetVaxErrorCode()

**UnRegisterAccountSIP()**

The UnRegisterAccountSIP() function unregisters/disconnects the client from SIP server.

**Syntax**

```
boolean UnRegisterAccountSIP(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
UnRegisterAccountSIP("TestSIP")
```

**See Also**

RegisterAccountSIP(), GetVaxErrorCode()

## AudioDeviceVU()

The AudioDeviceVU() activates VU (Volume Unit) functionality on a specific audio device (mic or spk). Such method can be used to develop VU meter.

### Syntax

```
boolean AudioDeviceVU(Activate, MicVU, SpkVU)
```

### Parameters

Activate(boolean)

The Activate parameter value can be 0 or 1. Assign value 1 to this parameter if you want to activate VU otherwise zero.

MicVU(boolean)

The MicVU parameter value can be 0 or 1. Assign value 1 to this parameter if you want to activate VU on MICROPHONE device otherwise zero.

SpkVU(boolean)

The SpkVU parameter value can be 0 or 1. Assign value 1 to this parameter if you want to activate VU on SPEAKER device otherwise zero.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
AudioDeviceVU(1, 1, 0)
```

### See Also

OnAudioDeviceMicVU(), OnAudioDeviceSpkVU()



**GetAudioInDevTotal()**

The GetAudioInDevTotal() function provides the total count of input devices attached to device.

**Syntax**

```
integer GetAudioInDeviceTotal()
```

**Parameters**

No parameters.

**Return Value**

Total number of audio input devices.

**Example**

```
GetAudioInDeviceTotal()
```

**See Also**

GetAudioOutDevTotal()

**GetAudioOutDevTotal()**

The GetAudioOutDevTotal() function provides the total count of output devices attached to device .

**Syntax**

```
integer GetAudioOutDeviceTotal()
```

**Parameters**

No parameters.

**Return Value**

Total number of audio output devices.

**Example**

```
GetAudioOutDeviceTotal()
```

**See Also**

GetAudioInDevTotal()

**GetAudioOutDevName()**

The GetAudioOutDevName() function returns the name of output audio device for provided device id.

**Syntax**

```
string GetAudioOutDevName(DeviceId)
```

**Parameters**

DeviceId(integer)

This parameter value can be any number from zero to total number of input devices – 1. Each number corresponds to a particular audio output device.

**Return Value**

Device name for corresponding device id, otherwise empty string.

**Example**

```
GetAudioOutDevName(0)
```

**See Also**

GetAudioInDevName(), GetAudioOutDevTotal(), GetAudioInDevTotal()

**GetAudioInDevName()**

The GetAudioInDevName() function returns the name of input audio device for provided device id.

**Syntax**

```
string GetAudioInDevName(DeviceId)
```

**Parameters**

DeviceId(integer)

This parameter value can be any number from zero to total number of input devices – 1. Each number corresponds to a particular audio input device.

**Return Value**

Device name for corresponding device id, otherwise empty string.

**Example**

```
GetAudioInDevName(0)
```

**See Also**

GetAudioOutDevTotal(), GetAudioInDevTotal(), GetAudioOutDevName()

## OpenLine()

The OpenLine() function opens a specific line to dial/receive call. As VaxVoIP supports multiple calls simultaneously so this function should be called prior to establishing connection, allowing user to dial/receive new calls on available free line.

### Syntax

```
boolean OpenLine(LineNo, RTPRxIP, RxAudioPortRTP, RxVideoPortRTP)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

RTPRxIP(string)

The RTPRxIP parameter value specifies the IP address of computer on which VaxVoIP receives voice streams. The ListenIP and RTPRxIP can be different if a computer has multiple IP addresses.

RxAudioPortRTP(integer)

The RxAudioPortRTP parameter value specifies the port number to receive voice streams. The Listen ports should be in range of 1024 to 65535 for UDP based transmission and for RTP compliance port number should be even.

RxVideoPortRTP(integer)

The RxVideoPortRTP parameter value specifies the port number to receive video streams. The Listen ports should be in range of 1024 to 65535 for UDP based transmission and for RTP compliance port number should be even.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = OpenLine(2, "", -1, -1) // "" = auto, -1 = auto
if (Result==0) GetVaxErrorCode( )
```

### See Also

CloseLine(), GetVaxErrorCode()

## CloseLine()

The CloseLine() function closes the specific line which is no longer in use. This method can be called every time a call is disconnected to close the specific line or all open lines can be closed once at component uninitialization.

### Syntax

```
boolean CloseLine(LineNo)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
OpenLine(2, "", -1, -1)  "" = auto, -1 = auto  
CloseLine(2)
```

### See Also

OpenLine(), GetVaxErrorCode()

## DialCall()

The DialCall() function sends call request to SIP server.

### Syntax

```
boolean DialCall(  
    AccountName,  
    LineNo,  
    CallerName,  
    CallerId,  
    DialNo,  
    InputDeviceId,  
    OutputDeviceId  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

CallerName(string)

This parameter specifies the user-name to be dialed.

CallerId(string)

This parameter specifies the user id to be dialed.

DialNo(string)

This parameter specifies the user-name or phone number to be dialed.

InputDeviceId(integer)

This parameter specifies the id of specific input device to be connected upon dialing call however -1 value can be provided for default input device.

OutputDeviceId(integer)

This parameter specifies the id of specific output device to be used upon dialing call however -1 value can be provided for default output device.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = DialCall("TestSIP", 0, "John", "", "001914600518", -1, -1)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

Disconnect(), GetAudioOutDevName(), GetAudioInDevName(),  
GetVaxErrorCode()



**DisconnectCall()**

The DisconnectCall() function disconnects the specific call in progress.

**Syntax**

```
boolean DisconnectCall(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = DisconnectCall(2)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

DialCall(), Connect(), GetVaxErrorCode()

## AcceptCall()

The AcceptCall() function accepts the incoming call.

### Syntax

```
boolean AcceptCall(  
    LineNo,  
    CallId,  
    InputDeviceId,  
    OutputDeviceId  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

CallId(string)

The CallId parameter value is a unique identifier for each incoming call. The value of this parameter is generated internally by the system (Incoming call-Id, please see OnIncomingCallStarted() event details).

InputDeviceId(integer)

This parameter specifies the id of specific input device to be connected upon accepting call however -1 value can be provided for default input device.

OutputDeviceId(integer)

This parameter specifies the id of specific output device to be connected upon accepting call however -1 value can be provided for default output device.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = AcceptCall(0, "24c654c@192.168.0.119", -1, -1)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

GetAudioOutDevName(), GetAudioInDevName(), RejectCall(),  
GetVaxErrorCode()

## **RejectCall()**

The RejectCall() function cancels/rejects the incoming call.

### **Syntax**

```
boolean RejectCall(CallId)
```

### **Parameters**

CallId(string)

The CallId parameter value is a unique identifier for each incoming call. The value of this parameter is generated internally by the system (Incoming call-Id, please see OnIncomingCallStarted() event details).

### **Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### **Example**

```
Result = RejectCall("24c654c@192.168.0.119")  
if(Result == 0) GetVaxErrorCode()
```

### **See Also**

AcceptCall(), GetVaxErrorCode()

## TransferCallBlind()

The TransferCallBlind() function transfers the call from a specific line to a specific number or user. This function can be used to implement “unannounced/blind call transfer i-e transferring the call without notifying the desired party/extension of the impending call”.

### Syntax

```
boolean TransferCallBlind(  
    LineNo,  
    ToUserName  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

ToUserName(string)

This parameter specifies the to user-name or phone number to be dialed.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = TransferCallBlind(2, "00192600524")  
if(Result == 0) GetVaxErrorCode()
```

### See Also

AcceptCall(), GetVaxErrorCode()

## TransferCallConsult()

The TransferCallConsult() function sends transfer call consult request to SIP Server and SIP Server links both calls. This function can be used to implement the feature “announced/consult call transfer i-e notifying the desired party/extension of the impending call by putting the caller on hold and dialing the desired party/extension”.

### Syntax

```
boolean TransferCallConsult(  
                                LineNoA,  
                                LineNoB  
                                )
```

### Parameters

LineNoA(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

LineNoB(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = TransferCallConsult (1, 3)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

TransferCallBlind(), GetVaxErrorCode()

**HoldLine()**

The HoldLine() method puts a specific line on hold.

**Syntax**

```
HoldLine(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = HoldLine(3)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

HoldLine(), GetVaxErrorCode()

**UnHoldLine()**

The UnHoldLine() function unholds a specific line.

**Syntax**

```
boolean UnHoldLine(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = UnHoldLine(2)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

HoldLine(), GetVaxErrorCode()

**IsLineOpen()**

The IsLineOpen() function returns open/close status of a specific line.

**Syntax**

```
boolean IsLineOpen(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns value 1 (true) if line is open otherwise zero.

**Example**

```
IsLineOpen(4)
```

**See Also**

OpenLine(), IsLineBusy()



**IsLineConnected()**

The IsLineConnected() function returns the status of already opened line i-e line is connected or free.

**Syntax**

```
boolean IsLineConnected(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns value 1 (true) if line is connected otherwise zero.

**Example**

```
IsLineConnected(4)
```

**See Also**

OpenLine(), IsLineOpen(), IsLineBusy()

**IsLineHold()**

The IsLineHold() method returns the HOLD status of a specific line.

**Syntax**

```
boolean IsLineHold(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns value 1 (true) if line is on hold otherwise zero.

**Example**

```
Result = IsLineHold(3)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

HoldLine(), GetVaxErrorCode()

**IsLineBusy()**

The IsLineBusy() function checks the status of already opened line i-e line is busy or free.

**Syntax**

```
boolean IsLineBusy(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns value 1 (true) if line is busy otherwise zero.

**Example**

```
IsLineBusy(4)
```

**See Also**

OpenLine(), IsLineOpen()

**IsLinePlaying()**

The IsLinePlaying() function checks whether the wave file playing is in progress or not on provided line.

**Syntax**

```
boolean IsLinePlaying(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns value 1(true) if wave file is playing on provided line otherwise it returns 0(false).

**Example**

```
IsLinePlaying(2)
```

**See Also**

PlayWaveOpen(), PlayWaveStart(), PlayWaveStop(), PlayWaveSkipTo(),  
GetVaxErrorCode()

**IsLineRecording()**

The IsLineRecording() function checks if recording is enabled or not on a specific line.

**Syntax**

```
boolean IsLineRecording(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns value 1(true) if recording is enabled on provided line otherwise 0(false).

**Example**

```
IsLineRecording(6)
```

**See Also**

StartRecording(), StopRecording(), GetVaxErrorCode()

## **EnableKeepAlive()**

The EnableKeepAlive() function keeps the ports open by periodically sending "keep alive" packets. This helps maintain the connection by preventing the NAT/firewall from closing the ports.

### **Syntax**

```
boolean EnableKeepAlive(AccountName, Seconds)
```

### **Parameters**

AccountName(string)

The unique name to identify a SIP account.

Seconds(integer)

This Seconds parameter value specifies the time interval in seconds after which keep alive packets will be sent to keep the port open for connection.

### **Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### **Example**

```
OpenLine(2, "192.168.0.103", 7006, 7008)  
EnableKeepAlive("TestSIP", 10)
```

### **See Also**

DisableKeepAlive(), GetVaxErrorCode()

**DisableKeepAlive()**

The DisableKeepAlive() method stops sending keep-alive packets, thereby disabling the functionality of the EnableKeepAlive() method.

**Syntax**

```
void DisableKeepAlive(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Return Value**

No return value.

**Example**

```
DisableKeepAlive("TestSIP")
```

**See Also**

EnableKeepAlive(), GetVaxErrorCode()

**SelectAllAudioCodec()**

The SelectAllAudioCodec() function selects all the voice codec.

**Syntax**

```
void SelectAllAudioCodec()
```

**Parameters**

No parameters.

**Return Value**

No return value.

**Example**

```
SelectAllAudioCodec()
```

**See Also**

DeselectAllAudioCodec(), GetVaxErrorCode()



## SelectAudioCodec()

The SelectAudioCodec() function selects a voice codec for provided codec number. The function can be called multiple times to select more than one voice codec. Moreover the sequence of selection of voice codec decides the priority of codec i-e the voice codec selected first has higher priority than the codec selected afterward.

### Syntax

```
boolean SelectAudioCodec(CodecNo)
```

### Parameters

CodecNo(integer)

This parameter value ranges from 0-4 and each value corresponds to a particular voice codec.

VaxVoIP SIP SDK supports the following voice codecs:

- 0 = G711 U-Law
- 1 = G711 A-Law
- 2 = GSM 6.10
- 3 = iLBC
- 4 = G729

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
DeselectAllAudioCodec()
```

```
SelectAudioCodec(4)  
SelectAudioCodec(1)  
SelectAudioCodec(2)  
SelectAudioCodec(3)
```

In this example G729 has the highest priority where as iLBC has Lowest priority.

### See Also

DeselectAudioCodec(), GetVaxErrorCode()

**DeselectAllVideoCodec()**

The DeselectAllVideoCodec() function deselects all the video codec options.

**Syntax**

```
void DeselectAllVideoCodec()
```

**Parameters**

No parameters.

**Return Value**

No return value.

**Example**

```
DeselectAllVideoCodec()
```

**See Also**

SelectAllVideoCodec(), GetVaxErrorCode()

**DeselectAudioCodec()**

The DeselectAudioCodec() function deselects a voice codec for provided codec number.

**Syntax**

```
boolean DeselectAudioCodec(CodecNo)
```

**Parameters**

CodecNo(integer)

This parameter value ranges from 0-4 and each value corresponds to a particular voice codec.

VaxVoIP SIP SDK supports the following voice codecs:

- 0 = G711 U-Law
- 1 = G711 A-Law
- 2 = GSM 6.10
- 3 = iLBC
- 4 = G729

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = DeselectAudioCodec(1)  
if(Result == 0) GetVaxErrorCode()
```

**See Also**

SelectAudioCodec(), GetVaxErrorCode()

**SelectAllVideoCodec()**

The SelectAllVideoCodec() function selects all the VaxVoIP SDK's supported video codecs.

**Syntax**

```
void SelectAllVideoCodec()
```

**Parameters**

No parameters.

**Return Value**

No return value.

**Example**

```
SelectAllVideoCodec()
```

**See Also**

DeselectAllVideoCodec(), GetVaxErrorCode()

## SelectVideoCodec()

The SelectVideoCodec() function selects a video codec for provided codec number. The function can be called multiple times to select more than one video codec. Moreover the sequence of selection of video codec decides the priority of codec i-e the video codec selected first has higher priority than the codec selected afterward.

### Syntax

```
boolean SelectVideoCodec(CodecNo)
```

### Parameters

CodecNo(integer)

This parameter value ranges from 0-2 and each value corresponds to a particular video codec.

VaxVoIP SIP SDK supports the following video codecs:

0 = VP8  
1 = H263  
2 = H263+

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
DeselectAllVideoCodec()
```

```
SelectVideoCodec(0)  
SelectVideoCodec(2)
```

In this example VP8 has the highest priority where as H263+ has lowest priority

### See Also

DeselectVideoCodec(), GetVaxErrorCode()

**DeselectAllVideoCodec()**

The DeselectAllVideoCodec() function deselects all the video codec options.

**Syntax**

```
void DeselectAllVideoCodec()
```

**Parameters**

No parameters.

**Return Value**

No return value.

**Example**

```
DeselectAllVideoCodec()
```

**See Also**

SelectAllVideoCodec(), GetVaxErrorCode()

## DeselectVideoCodec()

The DeselectVideoCodec() function deselects a video codec for provided codec number.

### Syntax

```
boolean DeselectVideoCodec(CodecNo)
```

### Parameters

CodecNo(integer)

This parameter value ranges from 0-2 and each value corresponds to a particular video codec.

VaxVoIP SIP SDK supports the following video codecs:

0 = VP8

1 = H263

2 = H263+

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = DeselectVideoCodec(1)
if(Result == 0) GetVaxErrorCode()
```

### See Also

SelectVideoCodec(), GetVaxErrorCode()

## DigitDTMF()

The DigitDTMF() function sends DTMF digit to the remote end SIP server. This method can also be used to play DTMF tones.

### Syntax

```
boolean DigitDTMF(  
    LineNo,  
    Digit,  
    TypeDTMF  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Digit(string)

This parameter value specifies any digit that has been pressed.  
(1, 2, 3, 4, 5, ..... 0, \*, #).

TypeDTMF(integer)

This parameter value specifies the Type Id.

The supported types are:

- 0 = AUTO
- 1 = RFC2833 TYPE
- 2 = SIP INFO TYPE
- 3 = INBAND or VOICE TYPE

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
DigitDTMF(1,"3", 0)
```

### See Also

SetVolumeDTMF(), GetVolumeDTMF()



**SetVolumeDTMF()**

The SetVolumeDTMF() function adjusts the volume of DTMF tones.

**Syntax**

```
boolean SetVolumeDTMF(Volume)
```

**Parameters**

Volume(integer)

This parameter specifies the volume level for DTMF tones ranges between 0-250.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
SetVolumeDTMF(6)
```

**See Also**

DigitDTMF(), GetVolumeDTMF()

**GetVolumeDTMF()**

The GeVolumeDTMF() function returns the volume level of DTMF tones.

**Syntax**

```
integer GeVolumeDTMF()
```

**Parameters**

No parameters.

**Return Value**

The function returns the volume of DTMF tones ranges between 0-250.

**Example**

```
SeVolumeDTMF(6)  
GeVolumeDTMF()
```

**See Also**

DigitDTMF(), SeVolumeDTMF()

## MuteMic()

The MuteMic() function mutes the microphone. Call to MuteMic() method does not affect the Master Mute Control. It simply starts sending silence data.

### Syntax

```
boolean MuteMic(Mute)
```

### Parameters

Mute(boolean)

The Mute parameter value can be 0 or 1. Assign value 1 to This parameter to mute the microphone otherwise zero.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
MuteMic(0)  
MuteMic(1)
```

### See Also

MuteSpk(), GetVaxErrorCode()

**MuteSpk()**

The MuteSpk() function mutes the speaker. Call to MuteSpk() does not affect the Master Mute Control.

**Syntax**

```
boolean MuteSpk(Mute)
```

**Parameters**

Mute(boolean)

The Mute parameter value can be 0 or 1. Assign value 1 to This parameter to mute the speaker otherwise zero.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
MuteSpk(0)  
MuteSpk(1)
```

**See Also**

MuteMic(), GetVaxErrorCode()

**MuteLineSpk()**

The MuteLineSpk() method mutes output voice stream of specific line.

**Syntax**

```
boolean MuteLineSpk(  
    LineNo,  
    Mute,  
    )
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Mute(boolean)

The Enable parameter value can be 0 or 1. Assign value 1 to This parameter to mute output voice stream otherwise zero.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
MuteLineSpk(2, 0)  
MuteLineSpk(2, 1)
```

**See Also**

MuteLineMic(), GetVaxErrorCode()

**MuteLineMic()**

The MuteLineMic() method mutes input voice stream of specific line.

**Syntax**

```
boolean MuteLineMic(  
    LineNo,  
    Mute  
)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Mute(boolean)

The Enable parameter value can be 0 or 1. Assign value 1 to This parameter to mute input voice stream otherwise zero.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
MuteLineMic(2, 0)  
MuteLineMic(2, 1)
```

**See Also**

MuteLineSpk(), GetVaxError

**AutoGainMic()**

The AutoGainMic() method enables auto gain functionality on outgoing voice stream.

**Syntax**

```
boolean AutoGainMic(  
    Enable,  
    Volume  
)
```

**Parameters**

Enable(boolean)

This parameter value enables/disables AGC functionality.

Volume(integer)

This parameter value specifies volume level ranges between [0-255]

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
AutoGainMic(1, 150)
```

**See Also**

AutoGainSpk(), GetVaxError

## AutoGainSpk()

The AutoGainSpk() method enables auto gain functionality on incoming voice stream.

### Syntax

```
boolean AutoGainSpk(  
    Enable,  
    Volume  
)
```

### Parameters

Enable(boolean)

This parameter value enables/disables AGC functionality.

Volume(integer)

This parameter value specifies volume level ranges between [0-255]

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
AutoGainSpk(0, 100)
```

### See Also

AutoGainMic(), GetVaxErrorCode()



**SetVolumeMic()**

The SetVolumeMic() function sets the volume of input voice stream. The microphone volume ranges between 0-255(0 = Min Volume, 255 = Max Volume).

**Syntax**

```
boolean SetVolumeMic(Volume)
```

**Parameters**

Volume(integer)

This parameter value specifies volume level ranges between [0-255].

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = SetVolumeMic(200)  
if(Result == 0) GetVaxErrorCode()
```

**See Also**

GetVolumeSpk(), SetVolumeSpk(), GetVolumeMic()

**GetVolumeMic()**

The GetVolumeMic() function returns the microphone volume. The microphone volume ranges between 0-255 (0 = Min Volume, 255 = Max Volume).

**Syntax**

```
integer GetVolumeMic()
```

**Parameters**

No parameters.

**Return Value**

The function returns microphone volume on its successful execution otherwise -1.

**Example**

```
GetVolumeMic()
```

**See Also**

GetVolumeSpk(), SetVolumeSpk(), SetVolumeMic()

**SetVolumeSpk()**

The SetVolumeSpk() function sets the volume of output voice stream. The speaker volume ranges between 0-255(0 = Min Volume, 255 = Max Volume).

**Syntax**

```
boolean SetVolumeSpk(Volume)
```

**Parameters**

Volume(integer)

This parameter value specifies volume level ranges between [0-255].

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = SetVolumeSpk(150)  
if(Result == 0) GetVaxErrorCode()
```

**See Also**

GetVolumeSpk(), GetVaxErrorCode()

**GetVolumeSpk()**

The GetVolumeSpk() function returns the speaker volume. The speaker volume ranges between 0-255 (0 = Min Volume, 255 = Max Volume).

**Syntax**

```
integer GetVolumeSpk()
```

**Parameters**

No parameters.

**Return Value**

The function returns speaker volume on its successful execution otherwise -1.

**Example**

```
GetVolumeSpk()
```

**See Also**

MuteSpk(), SetVolumeSpk()

## SetLineVolumeSpk()

The SetLineVolumeSpk() function adjusts the output volume of a specific line without affecting the operating system master volume control.

### Syntax

```
boolean SetLineVolumeSpk(  
                                LineNo,  
                                Volume  
                                )
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Volume(integer)

This parameter value specifies volume level ranges between [0-255].

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = SetLineVolumeSpk(2, 200)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

GetLineVolumeSpk()

**GetLineVolumeSpk()**

The GetLineVolumeSpk() function returns the output volume of a specific line. The speaker volume ranges between 0-255 (0 = Min Volume, 255 = Max Volume).

**Syntax**

```
integer GetLineVolumeSpk(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns speaker volume on its successful execution otherwise -1.

**Example**

```
GetLineVolumeSpk(2)
```

**See Also**

SetLineVolumeSpk(), SetVolumeSpk()

**DonotDisturb()**

The DonotDisturb() enables don't disturb functionality.

**Syntax**

```
boolean DonotDisturb(Enable)
```

**Parameters**

Enable(boolean)

This parameter value can be 0 or 1.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
DonotDisturb(1)
```

**See Also**

EchoCancellation(), GetVaxErrorCode()

## StartRecording()

The StartRecording() function starts recording voice stream on specific line.

### Syntax

```
boolean StartRecording(  
    LineNo,  
    FileName,  
    RecordVoice  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

FileName(string)

This parameter value specifies wave file name to be saved.

RecordVoice(integer)

This parameter value specifies the recording mode. It can have three values and each value corresponds to a particular recording mode.

0=Record outgoing only

1=Record incoming only

2=Record both

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
StartRecording(3, , "test.wav", 2)
```

### See Also

IsLineRecording(), StopRecording(), GetVaxErrorCode()



**StopRecording()**

The StopRecording() function stops the recording of voice stream on specific line.

**Syntax**

```
boolean StopRecording(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
StopRecording(4)
```

**See Also**

StartRecording(), IsLineRecording(), GetVaxErrorCode()

## PlayWaveOpen()

The PlayWaveOpen() function makes the wave file ready/set to play int a call.

### Syntax

```
boolean PlayWaveOpen(  
    LineNo,  
    FileName  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

FileName(string)

This parameter value specifies wave file name to be played.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = PlayWaveOpen(6, "test.wav")  
if(Result == 0) GetVaxErrorCode()
```

### See Also

IsLinePlaying(), PlayWaveStart(), PlayWaveStop(), PlayWaveSkipTo(),  
GetVaxErrorCode()

**PlayWaveClose()**

The PlayWaveClose() function vacates all the resources that were held by PlayWaveOpen() function.

**Syntax**

```
boolean PlayWaveClose(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
PlayWaveOpen(6, "test.wav")  
Result = PlayWaveClose(6)  
if(Result == 0) GetVaxErrorCode()
```

**See Also**

PlayWaveOpen(), PlayWaveStart(), PlayWaveStop(), PlayWaveSkipTo(),  
GetVaxErrorCode()

## PlayWaveSkipTo()

The PlayWaveSkipTo() function changes the position of playing cursor to the new position.

### Syntax

```
boolean PlayWaveSkipTo(  
    LineNo,  
    MilliSeconds  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

MilliSeconds(integer)

This parameter value specifies the time in milli-seconds to be skipped of playing wave file.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = PlayWaveSkipTo(4, 1000)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

PlayWaveOpen(), PlayWaveClose(), PlayWaveStop(), PlayWaveStart(),  
GetVaxErrorCode()

**PlayWaveTotalTime()**

The PlayWaveTotalTime() function returns the total playing time of a wave file on provided line.

**Syntax**

```
integer PlayWaveTotalTime(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns total playing time in milli-seconds of wave file otherwise 0.

**Example**

```
Result = PlayWaveTotalTime(4)
```

**See Also**

PlayWaveOpen(), PlayWaveClose(), PlayWaveStop(), PlayWaveStart(),  
PlayWavePause(), GetVaxErrorCode()

**PlayWavePause()**

The PlayWavePause() method pauses the playing wave file on its current position.

**Syntax**

```
boolean PlayWavePause(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = PlayWavePause(1)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

PlayWaveOpen(), PlayWaveClose(), PlayWaveStop(), PlayWaveStart(),  
PlayWaveSkipTo(), GetVaxErrorCode()

## PlayWaveStart()

The PlayWaveStart() method starts playing the already set wave file on provided line. The following sequence of execution starts playing the wave file.

- PlayWaveOpen()
- PlayWaveStart()
- 

It starts sending wave file data to the remote end, value listen = 1 starts sending and playing (on sound card) wave file data at the same time.

### Syntax

```
boolean PlayWaveStart(  
                        LineNo,  
                        Listen  
                      )
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Listen(boolean)

This parameter value can be 0 or 1. To play wave file just to remote end set its value 0 or sets its value 1 to play wave file to both remote end and sound card.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = PlayWaveStart(0, 0)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

PlayWaveOpen(), PlayWaveClose(), PlayWaveStop(), PlayWaveSkipTo(),  
GetVaxErrorCode()

## PlayWaveStop()

The PlayWaveStop() function stops playing the wave file on provided line and change the position of playing cursor at the beginning of file.

### Syntax

```
boolean PalyWaveStop(LineNo)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = PlayWaveStop(2)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

PlayWaveOpen(), PlayWaveClose(), PlayWavePause(), PlayWaveStart(),  
PlayWaveSkipTo(), GetVaxErrorCode()



**PlayWavePosition()**

The PlayWavePosition() method gets the current position of playing cursor.

**Syntax**

```
integer PlayWavePosition(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns current position in milli-seconds of playing cursor otherwise -1.

**Example**

```
PlayWaveOpen(4, "test.wav")
PlayWaveStart(4, 0)

Result = PlayWavePosition(4)
if(Result == -1) GetVaxErrorCode()
```

**See Also**

PlayWaveOpen(), PlayWaveClose(), PlayWaveStop(), PlayWaveStart(),  
PlayWaveSkipTo(), GetVaxErrorCode()

**GetOutboundCodec()**

The GetOutboundCodec() gets the codec number for the outbound voice stream of provided line.

**Syntax**

```
integer GetOutboundCodec(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

VaxVoIP SIP SDK support the following voice codecs:

- 0 = G711 A-Law
- 1 = G711 U-Law
- 2 = GSM 6.10
- 3 = iLBC
- 4 = G729

**Return Value**

The function returns a codec number on its successful execution otherwise -1.

**Example**

```
Result = GetOutboundCodec(1)  
if(Result == -1) GetVaxErrorCode()
```

**See Also**

GetInboundCodec(), GetVaxErrorCode()

**GetInboundCodec()**

The GetInboundCodec() gets the codec number for the Inbound voice stream of provided line.

**Syntax**

```
integer GetInboundCodec(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

VaxVoIP SIP SDK support the following voice codecs:

- 0 = G711 A-Law
- 1 = G711 U-Law
- 2 = GSM 6.10
- 3 = iLBC
- 4 = G729

**Return Value**

The function returns a codec number on its successful execution otherwise -1.

**Example**

```
Result = GetInBoundCodec(5)  
if(Result == -1) GetVaxErrorCode()
```

**See Also**

GetOutboundCodec(), GetVaxErrorCode()

**AudioSessionLost()**

The AudioSessionLost() function sets the specific time interval to check whether voice session is still intact or lost.

*NOTE: Due to some reasons, if VaxVoIP does not receives audio stream for a specific interval of time then it triggers OnAudioSessionLost() event.*

**Syntax**

```
void AudioSessionLost(Second)
```

**Parameters**

Second(integer)

This parameter value specifies the session lost time in seconds.

**Return Value**

No return value.

**Example**

```
AudioSessionLost(2)
```

**See Also**

OnAudioSessionLost()

**SetUserAgentSIP()**

The SetUserAgentSIP() function sets the user agent field of SIP packet.

**Syntax**

```
boolean SetUserAgentSIP(UserAgentName)
```

**Parameters**

UserAgentName(string)

This parameter value specifies the User agent Name.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = SetUserAgentSIP("abc")  
if(Result == 0) GetVaxErrorCode()
```

**See Also**

GetUserAgentSIP(), GetVaxErrorCode()

**GetUserAgentSIP()**

The GetUserAgentSIP() function returns the user agent field of SIP packet.

**Syntax**

```
string GetUserAgentSIP()
```

**Parameters**

No parameters.

**Return Value**

The function returns the user agent name otherwise empty string.

**Example**

```
GetUserAgentSIP()
```

**See Also**

SetUserAgentSIP()

**SetSubjectSDP()**

The SetSubjectSDP() function sets the subject field of SIP packet.

**Syntax**

```
boolean SetSubjectSDP(SubjectSDP)
```

**Parameters**

SubjectSDP(string)

This parameter specifies the value that is to be set as subject of SIP packet.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
SetSubjectSDP("xyz")
```

**See Also**

GetSubjectSDP()

**GetSubjectSDP()**

The GetSubjectSDP() function returns the subject field previously set by SetSubjectSDP() method.

**Syntax**

```
string GetSubjectSDP()
```

**Parameters**

No parameters.

**Return Value**

The function returns the subject.

**Example**

```
GetSubjectSDP()
```

**See Also**

SetSubjectSDP()



## ConfAllowLine()

The ConfAllowLine() function allows multiple users to speak/listen in conference. This feature of VaxVoIP componnet can be used for supervision of operators at call centers in real time.

### Syntax

```
boolean ConfAllowLine(  
    LineNo,  
    AllowListen,  
    AllowSpeak  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

AllowListen(boolean)

This parameter value can be 0 or 1. To allow user on specific line to listen in conference set parameter value to 1 otherwise 0.

AllowSpeak(boolean)

This parameter value can be 0 or 1. To allow user on specific line to speak in conference set parameter value to 1 otherwise 0.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
ConfAllowLine(1,0,1)  
ConfAllowLine(3,1,0)
```

### See Also

LineVoiceChannelSpk()

## ChatAddContact()

The ChatAddContact() method adds a contact to receive their status, such as online, busy, idle, etc.

### Syntax

```
boolean ChatAddContact(AccountName, UserName)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name to be added to chat.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = ChatAddContact("TestSIP", "abc")  
if(Result == 0) GetVaxErrorCode()
```

### See Also

ChatRemoveContact(), OnChatContactStatus(), GetVaxErrorCode()

**ChatRemoveContact()**

The ChatRemoveContact() method removes a contact that was previously added using the ChatAddContact() method.

**Syntax**

```
boolean ChatRemoveContact(AccountName, UserName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name to be removed from chat.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
ChatAddContact("TestSIP", "abc")  
ChatRemoveContact("TestSIP", "abc")
```

**See Also**

ChatAddContact(), GetVaxErrorCode()

**ChatFindContact()**

The ChatFindContact() function determines whether a user-name has already been added via the ChatAddContact() method.

**Syntax**

```
boolean ChatFindContact(AccountName, UserName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

**Return Value**

The function returns a non-zero value if user-name is found otherwise 0.

**Example**

```
ChatFindContact("TestSIP", "1010")
```

**See Also**

ChatAddContact()

**ChatSendMessageTyping()**

The ChatSendMessageTyping() functions sends the typing status to remote end/user.

**Syntax**

```
boolean ChatSendMessagingTyping(  
    AccountName,  
    UserName,  
    UserValue  
)
```

**Parameters**

AccountName(string)  
The unique name to identify a SIP account.

UserName(string)  
This parameter value specifies the user-name.

UserValue(integer)  
This parameter value is a user specified value.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
ChatSendMessagingTyping("TestSIP", "xyz", 0)
```

**See Also**

ChatSendMessageText(), GetVaxErrorCode()

## ChatSendMessageText()

The ChatSendMessageText() function sends the chat message text.

### Syntax

```
boolean ChatSendMessageText(  
    AccountName,  
    UserName,  
    MsgText,  
    MsgType,  
    UserValue  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

MsgText(string)

This parameter value specifies the message text.

MsgType(integer)

This parameter value specifies the number 101 or 102 which corresponds to particular message type.

UserValue(integer)

This parameter value is a user specified value

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
ChatAddContact("TestSIP", "abc")  
ChatSendMessageTyping("TestSIP", "abc", 3)  
ChatSendMessageText("TestSIP", "abc", "xyz", 101, 3)
```

### See Also

ChatSendMessageTyping(), GetVaxErrorCode()

## ChatSetMyStatus()

The ChatSetMyStatus() function sets the status of user for chat i-e online, offline, away, on-phone or busy.

### Syntax

```
boolean ChatSetMyStatus(AccountName, StatusId)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

StatusId(integer)

This parameter value corresponds to particular user chat status.

0 = Online

1 = Offline

2 = Away

3 = On Phone

4 = Busy

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
ChatSetMyStatus("TestSIP", 0)  
ChatSetMyStatus("TestSIP", 3)
```

### See Also

ChatAddContact(), ChatRemoveContact(), ChatSendMessageText()

**VoiceChanger()**

The VoiceChanger() functions changes the pitch of outgoing voice.

**Syntax**

```
boolean VoiceChanger(Pitch)
```

**Parameters**

Pitch(integer)

This parameter value can be -1 to disables the voice change or its value can be the pitch of the voice ranges between 0-20.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = VoiceChanger(4)
if(Result == 0) GetVaxErrorCode()
```

**See Also**



## ForwardCall()

The ForwardCall() function enables call forwarding.

### Syntax

```
boolean ForwardCall(  
    Enable,  
    ToUserName  
)
```

### Parameters

Enable(boolean)

This parameter value can be 0 or 1. Assign value 1 to enable the call forwarding to particular user or 0 to disable call forwarding.

ToUserName(string)

This parameter value specifies the user-name/number to be forwarded.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = ForwardCall(1, "abc")  
if(Result == 0) GetVaxErrorCode()
```

### See Also

DialCall(), GetVaxErrorCode()

## PlayAddPCM()

The PlayAddPCM() adds the voice digital PCM data to internally created buffer of VaxVoIP component and VaxVoIP component plays it to the call and remote end listens it.

### Syntax

```
boolean PlayAddPCM(  
    LineNo,  
    DataPCM,  
    SizePCM  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

DataPCM(data)

This parameter value specifies PCM data. Supported PCM format (8000Hz, 16bit, Mono)

SizePCM(integer)

This parameter value specifies the size of PCM data.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = PlayAddPCM(1, DataPCM, 8026)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

PlayResetPCM()

**PlayResetPCM()**

The PlayResetPCM() method resets/clear VaxVoIP internally play PCM buffer.

**Syntax**

```
boolean PlayResetPCM(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = PlayResetPCM(1)
if(Result == 0) GetVaxErrorCode()
```

**See Also**

PlayAddPCM()

## AddCustomHeader()

The AddCustomHeader() function can be used to add custom header fields in the SIP packets of different SIP requests.

Some of the SIP requests; REGISTER, INVITE

### Syntax

```
boolean AddCustomHeader(  
                        LineNo,  
                        ReqId,  
                        Name,  
                        Value  
                        )
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

ReqId(integer)

This parameter specifies a unique identification of a SIP request. Supported ReqId values are;

0 = INVITE

1 = REFER

Name(string)

This parameter specifies the name of custom header field.

Value(string)

This parameter specifies the value of custom header field.

### Return Value

On successful execution this function returns non-zero value otherwise it returns 0 value and specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
AddCustomHeader(0, 1, "Call_Info", "WaitingTime = 0")
```

### See Also

RemoveCustomHeader(), RemoveCustomHeaderAll()

## RemoveCustomHeader()

The RemoveCustomHeader() function removes the custom header fields added by using AddCustomHeader() function.

### Syntax

```
boolean RemoveCustomHeader(  
    ReqId,  
    Name  
)
```

### Parameters

ReqId(integer)

This parameter specifies a unique identification of a SIP request.  
Supported ReqId values are;

0 = INVITE

1 = REFER

Name(string)

This parameter specifies the custom header field.

### Return Value

On successful execution this function returns non-zero value otherwise it returns 0 value and specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
RemoveCustomHeader(0, "Call_Info")
```

### See Also

AddCustomHeader(), RemoveCustomHeaderAll()

**RemoveCustomHeaderAll()**

The RemoveCustomHeaderAll() function removes all custom header fields added by using AddCustomHeader() function.

**Syntax**

```
boolean RemoveCustomHeaderAll(ReqId)
```

**Parameters**

ReqId(integer)

This parameter specifies a unique identification of a SIP request.  
Supported ReqId values are;

0 = INVITE

1 = REFER

**Return Value**

On successful execution this function returns non-zero value otherwise it returns 0 value and specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
RemoveCustomHeaderAll(0)
```

**See Also**

AddCustomHeader(), RemoveCustomHeader()

**GetVideoDevTotal()**

The GetVideoDevTotal() function provides the total number of attached video devices.

**Syntax**

```
integer GetVideoDevTotal()
```

**Parameters**

No parameters

**Return Value**

Total number of video devices.

**Example**

```
GetVideoDevTotal()
```

**See Also**

GetVideoDevName()

**GetVideoDevName()**

The GetVideoDevName() function provides the name of specific video device attached to computer.

**Syntax**

```
string GetVideoDevName(DeviceId)
```

**Parameters**

DeviceId(integer)

This parameter value can be any number from zero to total number of video devices – 1. Each number corresponds to a particular video device.

**Return Value**

Device name for corresponding device id, otherwise empty string.

**Example**

```
GetVideoDevName(-1)
```

**See Also**

GetVideoDevTotal()



## OpenVideoDev()

The OpenVideoDev() function opens a specific video device attached to computer.

### Syntax

```
boolean OpenVideoDev(  
    DeviceId,  
    Quality  
)
```

### Parameters

DeviceId(integer)

This parameter value can be any number from zero to total number of video devices – 1. Each number corresponds to a particular video device.

Quality(integer)

This parameter value specifies the quality.

0 = LOW  
1 = STANDARD  
2 = MEDIUM  
3 = HIGH  
4 = MAX

### Return Value

On successful execution this function returns non-zero value otherwise it returns 0 value and specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = OpenVideoDev(0, 3)  
if(Result == 0) GetVaxErrorCode()
```

### See Also

CloseVideoDev()

**CloseVideoDev()**

The CloseVideoDev() function closes a specific video device previously opened by using OpenVideoDev() function.

**Syntax**

```
void CloseVideoDev(DeviceId)
```

**Parameters**

DeviceId(integer)

This parameter value can be any number from zero to total number of video devices – 1. Each number corresponds to a particular video device.

**Return Value**

No return value

**Example**

```
CloseVideoDev(0)
```

**See Also**

OpenVideoDev()

**CryptoMediaNONE()**

The CryptoMediaNONE() function disables encrypted media (audio/video) streaming (SRTP).

**Syntax**

```
boolean CryptoMediaNONE(Forced)
```

**Parameters**

Forced(boolean)

This parameter value disables the encrypted media streaming.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = CryptoMediaNONE(1)  
if(Result == 0) GetVaxErrorCode()
```

**CryptoMediaSDP()**

The CryptoMediaSDP() function enables encrypted media (audio/video) streaming (SRTP).

**Syntax**

```
boolean CryptoMediaSDP(Forced)
```

**Parameters**

Forced(boolean)

This parameter value enables the encrypted media streaming.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = CryptoMediaSDP(1)
if(Result == 0) GetVaxErrorCode()
```

**DialCallToREC()**

The DialCallToREC() function sends call request to SIP REC server.

**Syntax**

```
boolean DialCallToREC(  
    AccountName,  
    LineNo,  
    DialNo  
)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

DialNo(string)

This parameter specifies the user-name or phone number to be dialed.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
Result = DialCallToREC("TestSIP", 2, "000")  
if(Result == 0) GetVaxErrorCode()
```

**See Also**

OpenLineREC()

## OpenLineREC()

The OpenLineREC() function opens a specific line to connect call to SIP REC server. As VaxVoIP supports multiple calls simultaneously so this function should be called prior to establishing connection, allowing user to dial/receive new calls on available free line.

### Syntax

```
boolean OpenLineREC(AccountName, LineNo, RTPRxIP, AudioPortRTP)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

RTPRxIP(string)

The RTPRxIP parameter value specifies the IP address of computer on which VaxVoIP receives voice streams. The ListenIP and RTPRxIP can be different if a computer has multiple IP addresses.

AudioPortRTP(integer)

The AudioPortRTP parameter value specifies the port number to receive voice streams. The Listen ports should be in range of 1024 to 65535 for UDP based transmission and for RTP compliance port number should be even.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
Result = OpenLineREC("TestSIP", 2, "", -1) ; //-1 = auto, "" = auto  
if (Result==0) GetVaxErrorCode( )
```

### See Also

CloseLine(), GetVaxErrorCode()

**DialRingEnable()**

The DialRingEnable() function enable the dial ring.

**Syntax**

```
boolean DialRingEnable(FileName)
```

**Parameters**

FileName(string)

This parameter value specifies wave file name to be played as dial ring.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
DialRingEnable("DialRing.wav")
```

**See Also**

DialRingDisable(), GetVaxErrorCode()

**DialRingDisable()**

The DialRingDisable() function disables the dial ring functionality.

**Syntax**

```
boolean DialRingDisable()
```

**Parameters**

No Parameter.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
DialRingDisable()
```

**See Also**

DialRingEnable(), GetVaxErrorCode()



**BusyRingEnable()**

The BusyRingEnable() function enables the busy tone functionality.

**Syntax**

```
boolean BusyRingEnable(FileName)
```

**Parameters**

FileName(string)

This parameter value specifies wave file name to be played as busy tone.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
BusyRingEnable("Busy.wav")
```

**See Also**

BusyRingDisable(), GetVaxErrorCode()

**BusyRingDisable()**

The BusyRingDisable() function disables the busy tone functionality.

**Syntax**

```
boolean BusyRingDisable()
```

**Parameters**

No Parameter.

**Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
BusyRingDisable()
```

**See Also**

BusyRingEnable(), GetVaxErrorCode()

## EnableVideo()

The EnableVideo() function can be used to start or stop video streaming during the call or prior to dialing or receiving the call.

### Syntax

```
boolean EnableVideo(  
    LineNo,  
    Outbound,  
    Inbound  
)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Outbound(boolean)

This parameter value enables/disables the outbound video streaming.

Inbound(boolean)

This parameter value enables/disables the inbound video streaming.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
EnableVideo(1, 1, 0)
```

### See Also

OnVideoDeviceFrameRGB(), OnVideoRemoteFrameRGB()

**GetCallId()**

The GetCallId() function returns Call-Id field value of a SIP packet.

**Syntax**

```
string GetCallId(LineNo)
```

**Parameters**

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Return Value**

The function returns a string value on its successful execution otherwise empty string, a specific error code can be retrieved by calling GetVaxErrorCode() method.

**Example**

```
GetCallId(0)
```

**See Also**

GetVaxErrorCode()

**IsNetworkAvailable()**

The IsNetworkAvailable() function returns the status of the network.

**Syntax**

```
boolean IsNetworkAvailable()
```

**Parameters**

No Parameter.

**Return Value**

The function returns a Non-zero if the network is available otherwise 0

**Example**

```
IsNetworkAvailable()
```

**See Also**

OnNetworkReachability(), NetworkReachability()

**NetworkReachability()**

The NetworkReachability() function activates/deactivates the notification about network availability.

**Syntax**

```
boolean NetworkReachability(Enable)
```

**Parameters**

Enable(boolean)

This parameter value enables/disables the notification.

**Return Value**

The function returns a Non-zero if the network is available otherwise 0

**Example**

```
NetworkReachability(1)
```

**See Also**

OnNetworkReachability(), IsNetworkAvailable()

## AutoRegistration()

The AutoRegistration() function activates/deactivates the auto-registration functionality.

### Syntax

```
boolean AutoRegistration(  
    AccountName,  
    Enable,  
    TryCount,  
    TryAfterSeconds  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

Enable(boolean)

This parameter value enables/disables the functionality.

TryCount(integer)

This parameter value specifies total number of auto-registration tries.

TryAfterSeconds(integer)

This parameter value specifies the time period of auto-registration.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
AutoRegistration("TestSIP", true, -1, 10) // (-1 = there is no limit)
```

### See Also

OnFailToRegister(), GetVaxErrorCode()

## VideoCodecBitRate()

The VideoCodecBitRate() function can be used to set the video codec compression rate.

### Syntax

```
boolean VideoCodecBitRate(CodecNo, Quality)
```

### Parameters

CodecNo(integer)

This parameter's value is fixed at 0, corresponding to a specific video codec.

VaxVoIP SIP SDK supports the following video codecs:

0 = VP8

Quality(integer)

This parameter value specifies the quality.

0 = LOW

1 = STANDARD

2 = MEDIUM

3 = HIGH

4 = MAX

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
VideoCodecBitRate(0, 1) // (VP8 codec bitrate to standrad)
```

### See Also

OpenVideoDev(), CloseVideoDev(), GetVaxErrorCode()



## **BusyLampAddContact()**

The BusyLampAddContact() method adds and subscribes to receive a user statuses e.g online, busy etc. Busylamp functionality must be supported and configured on SIP SERVER side.

### **Syntax**

```
boolean BusyLampAddContact(AccountName, UserName)
```

### **Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

### **Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### **Example**

```
Result = BusyLampAddContact("TestSIP", "4040")  
if(Result == 0) GetVaxErrorCode()
```

### **See Also**

BusyLampRemoveContact(), GetVaxErrorCode()

## **BusyLampRemoveContact()**

The BusyLampRemoveContact() method removes and unsubscribes a user-name that was already added using BusyLampAddContact() method.

### **Syntax**

```
boolean BusyLampRemoveContact(AccountName, UserName)
```

### **Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

### **Return Value**

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method.

### **Example**

```
BusyLampAddContact("TestSIP", "abc")  
BusyLampRemoveContact("TestSIP", "abc")
```

### **See Also**

BusyLampAddContact(), GetVaxErrorCode()

**BusyLampFindContact()**

The BusyLampFindContact() function determines whether a user-name has already been added via the BusyLampAddContact() method.

**Syntax**

```
boolean BusyLampFindContact(AccountName, UserName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

**Return Value**

The function returns a non-zero value if user-name is found otherwise 0.

**Example**

```
BusyLampFindContact("TestSIP", "1010")
```

**See Also**

BusyLampAddContact()

## AudioVAD()

The AudioVAD method activates Voice Activity Detection, which identifies when speech is present in an audio signal and sends audio RTP packets only during speech.

### Syntax

```
boolean AudioVAD(LineNo, Enable)
```

### Parameters

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Enable(boolean)

This parameter enables or disables the VAD (Voice Activity Detection) functionality.

### Return Value

The function returns a Non-zero value on its successful execution otherwise 0, a specific error code can be retrieved by calling GetVaxErrorCode() method..

### Example

```
AudioVAD(0, 1)
```

### See Also

AudioSessionLost()

## DiagnosticLogSIP()

The DiagnosticLogSIP() method provides the logging and monitoring of SIP messages.

The VaxVoIP component triggers the OnIncomingDiagnostic() or OnOutgoingDiagnostic() event when it receives or sends SIP messages..

### Syntax

```
boolean DiagnosticLogSIP(AccountName, Inbound, Outbound)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

Inbound(boolean)

The value of this parameter can be 0 or 1. To enable diagnostic of inbound voice stream assign value 1 to this parameter or 0 to disable it.

Outbound(boolean)

The value of this parameter can be 0 or 1. To enable diagnostic of outbound voice stream assign value 1 to this parameter or 0 to disable it.

### Return Value

On successful execution this function returns non-zero value otherwise it returns 0 value and specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
DignosticLogSIP("TestSIP", 1, 0)
```

### See Also

OnOutgoingDiagnostic(), OnIncomingDiagnostic(), GetVaxErrorCode()

## SetPushSIP()

The SetPushSIP() method likely sets up or configures SIP push notifications for a specified account using the provided parameters.

### Syntax

```
boolean SetPushSIP(AccountName, ProjectId, TokenReg)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

ProjectId(string)

This parameter may denote a project or bundle identifier associated with the push notification service.

TokenReg(string)

This parameter likely represents a device registration token or identifier needed for push notification services.

### Return Value

On successful execution this function returns non-zero value otherwise it returns 0 value and specific error code can be retrieved by calling GetVaxErrorCode() method.

### Example

```
SetPushSIP("TestSIP","com.VaxSoft.VaxPhoneSDK.iOS", "77df3234242")
```

### See Also

RegisterAccountSIP()

## **EXPORTED EVENTS**

### **OnInitialized()**

The OnInitialized() event notifies that VaxVoIP component has initialized successfully.

#### **Syntax**

```
void OnInitialized()
```

#### **Parameters**

No parameters

#### **Example**

```
OnInitialized()  
{  
}
```

#### **See Also**

Initialize(), OnUnInitialized()

**OnUnInitialized()**

The OnUnInitialized() event notifies that VaxVoIP component has uninitialized successfully.

**Syntax**

```
void OnUnInitialized()
```

**Parameters**

No parameters

**Example**

```
OnUnInitialized()  
{  
}  
}
```

**See Also**

UnInitialize(), OnInitialized()



**OnConnectingToRegister()**

VaxVoIP triggers OnConnectingToRegister() event when VaxVoIP starts connecting to the SIP server to start registration process.

**Syntax**

```
void OnConnectingToRegister(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Example**

```
OnConnectingToRegister(AccountName)
{
}
```

**See Also**

OnTryingToRegister()

## OnTryingToRegister()

VaxVoIP triggers OnTryingToRegister() event when client sends the register request to SIP server and request is in process on server end.

### Syntax

```
void OnTryingToRegister(AccountName)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

### Example

```
OnTryingToRegister(AccountName)
{
}
```

### See Also

OnTryingToUnRegister(), OnFailToRegister(), OnSuccessToRegister(),  
RegisterAccountSIP(), UnRegisterAccountSIP()

## OnFailToRegister()

The OnFailToRegister() event triggers when client failed to register with server or registration request has not completed successfully.

### Syntax

```
void OnFailToRegister(  
    AccountName,  
    StatusCode,  
    ReasonPhrase  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

StatusCode(integer)

This parameter specifies SIP response status code (486, 404 etc).

ReasonPhrase(string)

This parameter specifies SIP response reason phrase (Unauthorized, Not Found etc).

### Example

```
OnFailToRegister(AccountName , StatusCode, ReasonPhrase)  
{  
}
```

### See Also

OnFailToUnRegister(), OnFailToRegister(), OnSuccessToRegister(),  
RegisterAccountSIP(), UnRegisterAccountSIP()

**OnSuccessToRegister()**

The OnSuccessToRegister() event triggers when client successfully registered with SIP server.

**Syntax**

```
void OnSuccessToRegister(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Example**

```
OnSuccessToRegister(AccountName)
{
}
```

**See Also**

OnTryingToRegister(), OnFailToRegister(), OnTryingToUnRegister()  
RegisterAccountSIP(), UnRegisterAccountSIP()

**OnConnectingToReRegister()**

VaxVoIP triggers OnConnectingToReRegister() event when VaxVoIP starts connecting to the SIP server to start re-registration process.

**Syntax**

```
void OnConnectingToReRegister(AccountName)
```

**Parameters**

AccountName(string)  
The unique name to identify a SIP account.

**Example**

```
OnConnectingToReRegister(AccountName)  
{  
}
```

**See Also**

OnTryingToReRegister()

**OnTryingToReRegister()**

OnTryingToReRegister() event triggers when client sends re-register request to SIP server and request is in process at server end.

It notifies that sip server is processing the re-register request.

**Syntax**

```
void OnTryingToReRegister(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Example**

```
OnTryingToReRegister(AccountName)
{
}
```

**See Also**

OnSuccessToReRegister(), OnFailToReRegister(), RegisterAccountSIP(),  
UnRegisterAccountSIP()

## OnFailToReRegister()

The OnFailToReRegister() event triggers when client failed to re-register with server or re-registration request has not completed successfully.

### Syntax

```
void OnFailToReRegister(  
    AccountName,  
    StatusCode,  
    ReasonPhrase  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

StatusCode(integer)

This parameter specifies SIP response status code (486, 404 etc).

ReasonPhrase(string)

This parameter specifies SIP response reason phrase (Unauthorized, Not Found etc).

### Example

```
OnFailToReRegister(StatusCode, ReasonPhrase)  
{  
  
}
```

### See Also

OnTryingToReRegister(), OnSuccessToReRegister(), RegisterAccountSIP(),  
UnRegisterAccountSIP()

**OnSuccessToReRegister()**

The OnSuccessToReRegister() event triggers when client successfully re-registered with SIP server.

**Syntax**

```
void OnSuccessToReRegister(AccountName)
```

**Parameters**

AccountName(string)  
The unique name to identify a SIP account.

**Example**

```
OnSuccessToRegister(AccountName)  
{  
}
```

**See Also**

OnTryingToReRegister(), OnFailToReRegister(), RegisterAccountSIP(),  
UnRegisterAccountSIP()



## **OnTryingToUnRegister()**

The OnTryingToUnRegister() event triggers when client sends the unregister request to SIP server and request is in process at server end.

### **Syntax**

```
void OnTryingToUnRegister(AccountName)
```

### **Parameters**

AccountName(string)  
The unique name to identify a SIP account.

### **Example**

```
OnTryingToUnRegister(AccountName)
{
}
```

### **See Also**

OnTryingToRegister(), OnFailToRegister(), OnSuccessToRegister()  
RegisterAccountSIP(), UnRegisterAccountSIP()

**OnFailToUnRegister()**

The OnFailToUnRegister() event triggers when client failed to unregister with server or unregister request has not been completed successfully.

**Syntax**

```
void OnFailToUnRegister(AccountName)
```

**Parameters**

AccountName(string)  
The unique name to identify a SIP account.

**Example**

```
OnFailToUnRegister(AccountName)
{
}
```

**See Also**

OnSuccessToUnRegister(), OnSuccessToRegister(), OnTryingToUnRegister()  
RegisterAccountSIP(), UnRegisterAccountSIP()

**OnSuccessToUnRegister()**

The OnSuccessToUnRegister() events triggers when client request to unregister with server is successfully completed.

**Syntax**

```
void OnSuccessToUnRegister(AccountName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

**Example**

```
OnSuccessToUnRegister(AccountName)
{
}
```

**See Also**

OnFailToUnRegister(), OnSuccessToRegister(), OnTryingToUnRegister()  
RegisterAccountSIP(), UnRegisterAccountSIP()

## OnDialCallStarted()

The OnDialCallStarted() events triggers when DialCall() method executes.

### Syntax

```
void OnDialCallStarted(  
    AccountName,  
    LineNo,  
    CallerName,  
    CallerId,  
    DialNo  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

CallerName(string)

This parameter value specifies the caller name which is provided by IP-Telephony or VoIP service provider.

CallerId(string)

The CallerId parameter value specifies the CallerId

DialNo(string)

This parameter specifies the user-name or phone number to be dialed.

### Example

```
OnDialCallStarted(AccountName, LineNo, CallerName, CallerId, DialNo)  
{  
}
```

### See Also

OnDialingCall(), OnDialCallFailed()

## OnDialingCall()

The OnDialingCall() events triggers when client dial a call started with server

### Syntax

```
void OnDialCallStarted(  
    AccountName,  
    LineNo,  
    StatusCode,  
    ReasonPhrase  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

StatusCode(integer)

This parameter specifies SIP response status code (100, 180 etc).

ReasonPhrase(string)

This parameter specifies SIP response reason phrase (Trying, Ringing etc).

### Example

```
OnDialingCall(AccountName, LineNo, StatusCode, ReasonPhrase)  
{  
}  
}
```

### See Also

OnDialCallStarted(), OnDialCallFailed()

## OnDialCallFailed()

The OnDialCallFailed() events triggers when dialed call fails and SIP server sends a failure response.

### Syntax

```
void OnDialCallFailed(  
    AccountName,  
    LineNo,  
    StatusCode,  
    ReasonPhrase  
    Contact  
)
```

### Parameters

AccountName(string)  
The unique name to identify a SIP account.

LineNo(integer)  
This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

StatusCode(integer)  
This parameter specifies SIP response status code (486, 404 etc).

ReasonPhrase(string)  
This parameter specifies SIP response reason phrase (Unauthorized, Not Found etc).

Contact(string)  
This parameter value specifies the contact where SIP server will redirect the call in case of call-forwarding is enabled on SIP server side.

### Example

```
OnDialCallFailed(AccountName, LineNo, StatusCode, ReasonPhrase, Contact)  
{  
}
```

### See Also

OnDialCallStarted(), OnDialingCall()

**OnConnectedCall()**

The OnConnectedCall() events triggers when a call gets connected successfully.

**Syntax**

```
void OnConnectedCall(  
    AccountName,  
    LineNo,  
    ToAddrRTP,  
    ToPortRTP  
)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

ToAddrRTP(string)

This parameter specifies the RTP IP address of remote end.

ToPortRTP(integer)

This parameter specifies the RTP port number of remote end.

**Example**

```
OnConnectedCall(AccountName, LineNo, ToAddrRTP, ToPortRTP)  
{  
}
```

**See Also**

OnHungupCall()

**OnHungupCall()**

The OnHungupCall() events triggers when remote party hangup the call.

**Syntax**

```
void OnHungupCall(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnHungupCall(AccountName, LineNo)
{
}
```

**See Also**

OnConnectedCall()



## OnIncomingCallStarted()

The OnIncomingCallStarted() event triggers when component receives a call request.

### Syntax

```
void OnIncomingCallStarted(  
    AccountName,  
    CallId,  
    CallerName, CallerId,  
    DialNo,  
    FromURI, ToURI  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

CallId(string)

The CallId parameter value is a unique identifier for each incoming call.  
The value of this parameter is generated internally by the system.

CallerName(string)

This parameter value specifies the caller name.

CallerId(string)

This parameter value specifies the callerId.

DialNo(string)

This parameter specifies the user-name or phone number to be dialed.

FromURI(string)

This parameter specifies FromURI in incoming SIP call request.

ToURI(string)

This parameter specifies ToURI in incoming SIP call request.

### Example

```
OnIncomingCallStarted(AccountName, CallId, CallerName, CallerId, DialNo,  
    FromURI, ToURI)  
{  
}
```

### See Also

AcceptCall(), RejectCall(), HoldLine()

**OnIncomingCallEnded()**

The OnIncomingCallEnded() event triggers when remote end cancels the call request.

**Syntax**

```
void OnIncomingCallEnded(AccountName, CallId)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

CallId(string)

The CallId parameter value is a unique identifier for each incoming call. The value of this parameter is generated internally by the system.

**Example**

```
OnIncomingCallEnded(AccountName, CallId)
{
}
```

**See Also**

AcceptCall(), RejectCall(), HoldLine()

## OnRingToneStarted()

The OnRingToneStarted() event notifies VaxVoIP integrated application about to start playing ringtone wave file.

### Syntax

```
void OnRingToneStarted(AccountName, CallId)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

CallId(string)

The CallId parameter value is a unique identifier for each incoming call. The value of this parameter is generated internally by the system.

### Example

```
OnRingToneStarted(AccountName, CallId)
{
}
```

### See Also

OnRingToneEnded()

**OnRingToneEnded()**

The OnRingToneEnded() event notifies VaxVoIP integrated application to stop the playing of ringtone wave file.

**Syntax**

```
void OnRingToneEnded(AccountName, CallId)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

CallId(string)

The CallId parameter value is a unique identifier for each incoming call. The value of this parameter is generated internally by the system.

**Example**

```
OnRingToneEnded(AccountName, CallId)
{
}
```

**See Also**

OnRingToneStarted()

**OnTransferCallAccepted()**

The OnTransferCallAccepted() event triggers when SIP server acknowledge/accepts the call transfer request.

**Syntax**

```
void OnTransferCallAccepted(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnTransferCallAccepted(AccountName, LineNo)
{
}
```

**See Also**

OnTransferCallFailed()

## OnTransferCallFailed()

The OnTransferCallFailed() event triggers when call transfer process fails and SIP Server sends an error response.

### Syntax

```
void OnTransferCallFailed(  
    AccountName,  
    LineNo,  
    StatusCode,  
    ReasonPhrase  
)
```

### Parameters

AccountName(string)  
The unique name to identify a SIP account.

LineNo(integer)  
This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

StatusCode(integer)  
This parameter specifies SIP response status code (486, 404 etc).

ReasonPhrase(string)  
This parameter specifies SIP response reason phrase (Unauthorized, Not Found etc).

### Example

```
void OnTransferCallFailed(AccountName, LineNo, StatusCode, ReasonPhrase)  
{  
  
}
```

### See Also

OnTransferCallAccepted()

**OnPlayWaveDone()**

The OnPlayWaveDone() event triggers on the completion of playing of a wave file.

**Syntax**

```
void OnPlayWaveDone(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnPlayWaveDone(AccountName, LineNo)
{
}
```

**See Also**

PlayWaveOpen(), PlayWaveClose(), PlayWaveStart(), PlayWaveStop()

## OnDigitDTMF()

The OnDigitDTMF() event triggers when remote end pressed any key/DTMF.

### Syntax

```
void OnDigitDTMF(  
    AccountName,  
    LineNo,  
    Digit  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

Digit(string)

This parameter value specifies any digit that has been pressed.  
(1, 2, 3, 4, 5, ..... 0, \*, #)

### Example

```
OnDigitDTMF(AccountName, LineNo, Digit)  
{  
}
```

### See Also

DigitDTMF(), SetVolumeDTMF(), GetVolumeDTMF()



## OnVoiceMailMsg()

The OnVoiceMailMsg() event triggers when VaxVoIP component receives voice mail notification from SIP server. This event only works if voice mail message service is enabled on SIP server side.

### Syntax

```
void OnVoiceMailMsg(  
    AccountName,  
    MsgWaiting, NewMsgCount, OldMsgCount,  
    NewUrgentMsgCount, OldUrgentMsgCount,  
    MsgAccount  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

MsgWaiting(boolean)

This parameter value specifies whether some messages are in waiting state or not.

NewMsgCount(integer)

This parameter specifies total count for new messages.

OldMsgCount(integer)

This parameter specifies total count for old messages.

NewUrgentMsgCount(integer)

This parameter value specifies total count for new urgent messages.

OldUrgentMsgCount(integer)

This parameter value specifies total count for old urgent messages.

MsgAccount(string)

This parameter value specifies message account.

### Example

```
OnVoiceMailMsg(AccountName, MsgWaiting, NewMsgCount, OldMsgCount,  
    NewUrgentMsgCount, OldUrgentMsgCount, MsgAccount)  
{  
}
```

### See Also

OnIncomingDiagnostic()

**OnIncomingDiagnostic()**

The OnIncomingDiagnostic() event triggers when VaxVoIP receives a SIP packet. This event can be use for logging and monitoring of inbound SIP messages.

**Syntax**

```
void OnIncomingDiagnostic(  
    AccountName,  
    MsgSIP,  
    FromIP,  
    FromPort  
)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

MsgSIP(string)

This parameter value specifies the SIP packet message.

FromIP(string)

This parameter value specifies the from IP address.

FromPort(integer)

This parameter specifies the from port number.

**Example**

```
OnIncomingDiagnostic(AccountName, MsgSIP, FromIP, FromPort)  
{  
}
```

**See Also**

OnOutgoingDiagnostic()

## OnOutgoingDiagnostic()

The OnOutgoingDiagnostic() event triggers when VaxVoIP sends a SIP packet. This event can be use for logging and monitoring of outbound SIP messages.

### Syntax

```
void OnOutgoingDiagnostic(  
    AccountName,  
    MsgSIP,  
    ToIP,  
    ToPort  
)
```

### Parameters

AccountName(string)  
The unique name to identify a SIP account.

MsgSIP(string)  
This parameter value specifies the SIP packet message.

ToIP(string)  
This parameter value specifies the to IP address.

ToPort(string)  
This parameter specifies the to port number.

### Example

```
OnOutgoingDiagnostic(AccountName, MsgSIP, ToIP, ToPort)  
{  
}
```

### See Also

OnIncomingDiagnostic()

**OnAudioSessionLost()**

The OnAudioSessionLost() triggers only when client has already enabled session lost through AudioSessionLost() and has not received any voice data for specified interval of time.

**Syntax**

```
void OnAudioSessionLost(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnAudioSessionLost(AccountName, LineNo)
{
}
```

**See Also**

AudioSessionLost()

**OnSuccessToHold()**

The OnSuccessToHold() event triggers when a call is successfully placed on hold.

**Syntax**

```
void OnSuccessToHold(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnSuccessToHold(AccountName, LineNo)
{
}
```

**See Also**

OnTryingToHold(), OnFailToHold(), HoldLine(), UnHoldLine(), IsLineHold()

## OnTryingToHold()

The OnTryingToHold() event triggers when client sends the hold request for specific line to SIP server and request is in process on server end.

### Syntax

```
void OnTryingToHold(AccountName, LineNo)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### Example

```
OnTryingToHold(AccountName, LineNo)
{
}
```

### See Also

OnSuccessToHold(), OnFailToHold(), HoldLine(), UnHoldLine(), IsLineHold()

## OnFailToHold()

The OnFailToHold() event triggers when hold request to server has not been completed successfully.

### Syntax

```
void OnFailToHold(AccountName, LineNo)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### Example

```
OnFailToHold(AccountName, LineNo)
{
}
```

### See Also

OnSuccessToHold(), OnTryingToHold(), HoldLine(), UnHoldLine(),  
IsLineHold()

**OnSuccessToUnHold()**

The OnSuccessToUnHold() event triggers when request to unhold a specific line is completed successfully.

**Syntax**

```
void OnSuccessToUnHold(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnSuccessToUnHold(AccountName, LineNo)
{
}
```

**See Also**

OnTryingToUnHold(), OnFailToUnHold(), HoldLine(), UnHoldLine(),  
IsLineHold()



**OnTryingToUnHold()**

The OnTryingToUnHold() event triggers when client sends the unhold request for specific line to SIP server and request is in process at server end.

**Syntax**

```
void OnTryingToUnHold(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnTryingToUnHold(AccountName, LineNo)
{
}
```

**See Also**

OnSuccessToUnHold(), OnFailToUnHold(), HoldLine(), UnHoldLine(),  
IsLineHold()

**OnFailToUnHold()**

The OnFailToUnHold() event triggers when unhold request to server has not been completed successfully.

**Syntax**

```
void OnFailToUnHold(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
OnFailToUnHold(AccountName, LineNo)
{
}
```

**See Also**

OnTryingToUnHold(), OnSuccessToUnHold(), HoldLine(), UnHoldLine(),  
IsLineHold()

## OnChatContactStatus()

The OnChatContactStatus() event triggers when remote party/user changes the status e.g. busy, away etc.

### Syntax

```
void OnChatContactStatus(  
    AccountName,  
    UserName,  
    StatusId  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

StatusId(integer)

This parameter value corresponds to particular user chat status.

0 = Online

1 = Offline

2 = Away

3 = On Phone

4 = Busy

5 = Unknown

### Example

```
void OnChatContactStatus(AccountName, UserName, StatusId)  
{  
}  
}
```

### See Also

ChatSetMyStatus(), ChatAddContact()

## OnChatSendMsgTextSuccess()

The OnChatSendMsgTextSuccess() event triggers when chat text message is sent successfully.

### Syntax

```
void OnChatSendMsgTextSuccess(  
                                AccountName,  
                                UserName,  
                                MsgText,  
                                UserValue  
                                )
```

### Parameters

AccountName(string)  
The unique name to identify a SIP account.

UserName(string)  
This parameter value specifies the user-name.

MsgText(string)  
This parameter value specifies the message text.

UserValue(integer)  
This parameter value is a user specified value.

### Example

```
void OnChatSendMsgTextSuccess(AccountName, UserName, MsgText  
                                UserValue)  
{  
}
```

### See Also

OnChatSendMsgTextFailed(), ChatSendMessageText()

## OnChatSendMsgTextFailed()

The OnChatSendMsgTextFailed() event triggers when message sending to remote end failed.

### Syntax

```
void OnChatSendMsgTextFailed(  
    AccountName,  
    UserName,  
    StatusCode,  
    ReasonPhrase,  
    MsgText,  
    UserValue  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

StatusCode(integer)

This parameter specifies SIP response status code.

ReasonPhrase(string)

This parameter specifies SIP response reason phrase (Trying, Ringing etc).

MsgText(string)

This parameter value specifies the message text.

UserValue(integer)

This parameter value is a user specified value.

### Example

```
void OnChatSendMsgTextFailed(AccountName, UserName, StatusCode,  
    ReasonPhrase, MsgText, UserValue)  
{  
}
```

### See Also

OnChatSendMsgTextSuccess(), ChatSendMessageText()

## OnChatSendMsgTypingSuccess()

The OnChatSendMsgTypingSuccess() event triggers when typing status is sent successfully.

### Syntax

```
void OnChatSendMsgTypingSuccess(  
                                AccountName,  
                                UserName,  
                                UserValue  
                                )
```

### Parameters

AccountName(string)  
The unique name to identify a SIP account.

UserName(string)  
This parameter value specifies the user-name.

UserValue(integer)  
This parameter value is a user specified value.

### Example

```
void OnChatSendMsgTypingSuccess(AccountName, UserName, UserValue)  
{  
}  
}
```

### See Also

OnChatSendMsgTypingFailed(), ChatSendMessageTyping()

## OnChatSendMsgTypingFailed()

The OnChatSenMsgTypingFail() event triggers when typing status sending to remote end failed.

### Syntax

```
void OnChatSendMsgTypingFailed(  
                                AccountName,  
                                UserName,  
                                StatusCode,  
                                ReasonPhrase,  
                                UserValue  
                                )
```

### Parameters

AccountName(string)  
The unique name to identify a SIP account.

UserName(string)  
This parameter value specifies the user-name.

StatusCode(integer)  
This parameter specifies SIP response status code.

ReasonPhrase(string)  
This parameter specifies SIP response reason phrase (Trying, Ringing etc).

UserValue(integer)  
This parameter value is a user specified value.

### Example

```
void OnChatSendMsgTypingFailed(AccountName, UserName, StatusCode,  
                                ReasonPhrase, UserValue)  
{  
}
```

### See Also

OnChatSendMsgTypingSuccess(), ChatSendMessageTyping()

**OnChatRecvMsgText()**

The OnChatRecvMsgText() event triggers when VaxVoIP component receives a text message.

**Syntax**

```
void OnChatRecvMsgText(  
    AccountName,  
    UserName,  
    MsgText  
)
```

**Parameters**

AccountName(string)  
The unique name to identify a SIP account.

UserName(string)  
This parameter value specifies the user-name.

MsgText(string)  
This parameter value specifies the message text.

**Example**

```
OnChatRecvMsgText(AccountName, UserName, MsgText)  
{  
}
```

**See Also**

OnChatSendMsgTextSuccess(), ChatSendMessageText()



**OnChatRecvMsgTypingStart()**

The OnChatRecvMsgTypingStart() event triggers when a user at remote end starts typing a text message.

**Syntax**

```
void OnChatRecvMsgTypingStart(AccountName, UserName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

**Example**

```
OnChatRecvMsgTypingStart(AccountName, UserName)
{
}
```

**See Also**

OnChatSendMsgTypingFailed(), ChatSendMessageTyping(),  
OnChatSendMsgTypingSuccess(), ChatSendMessageTyping()

**OnChatRecvMsgTypingStop()**

The OnChatRecvMsgTypingStop() event triggers when a user at remote end stops typing a text message.

**Syntax**

```
void OnChatRecvMsgTypingStop(AccountName, UserName)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

UserName(string)

This parameter value specifies the user-name.

**Example**

```
OnChatRecvMsgTypingStop(AccountName, UserName)
{
}
```

**See Also**

OnChatSendMsgTypingSuccess(), ChatSendMessageTyping(),  
OnChatSendMsgTypingFailed(), ChatSendMessageTyping()

**OnHoldCall()**

The OnHoldCall() event triggers if VaxVoIP component receives hold request from the SIP Server.

**Syntax**

```
void OnHoldCall(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
void OnHoldCall(AccountName, LineNo)
{
}
}
```

**See Also**

OnUnHoldCall()

**OnUnHoldCall()**

The OnUnHoldCall() event triggers if VaxVoIP component receives unhold request from the SIP Server.

**Syntax**

```
void OnUnHoldCall(AccountName, LineNo)
```

**Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

**Example**

```
void OnUnHoldCall(AccountName, LineNo)
{
}
}
```

**See Also**

OnHoldCall()

## OnVideoRemoteStarted()

The OnVideoRemoteStarted() event triggers when VaxVoIP component starts receiving the video frames from remote end.

### Syntax

```
void OnVideoRemoteStarted(AccountName, LineNo)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### Example

```
void OnVideoRemoteStarted(AccountName, LineNo)
{
}
}
```

### See Also

OnVideoRemoteEnded(), OnVideoRemoteFrameRGB()

## **OnVideoRemoteEnded()**

The OnVideoRemoteEnded() event triggers when VaxVoIP component stops receiving the video frames.

### **Syntax**

```
void OnVideoRemoteEnded(AccountName, LineNo)
```

### **Parameters**

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

### **Example**

```
void OnVideoRemoteEnded(AccountName, LineNo)
{
}
}
```

### **See Also**

OnVideoRemoteStarted(), OnVideoRemoteFrameRGB()

## OnVideoRemoteFrameRGB()

The OVideoRemoteFrameRGB() event triggers when VaxVoIP component receives a video frame.

### Syntax

```
void OnVideoRemoteFrameRGB(  
    AccountName, LineNo,  
    FrameRGB, FrameSize,  
    FrameWidth, FrameHeight  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

LineNo(integer)

This parameter value specifies the specific line. The LineNo value is a unique number to identify a specific line.

FrameRGB(integer)

Frame data in the form of RGB.

FrameSize(integer)

Specifies the Frame data size.

FrameWidth(integer)

Specifies the Frame width.

FrameHeight(integer)

Specifies the Frame height.

### Example

```
void OnVideoRemoteFrameRGB(AccountName,LineNo, FrameRGB,  
    FrameSize, FrameWidth, FrameHeight)  
{  
}
```

### See Also

OnVideoDeviceFrameRGB(), OnVideoRemoteStarted()

## OnVideoDeviceFrameRGB()

The OVideoDeviceFrameRGB() event triggers when VaxVoIP component receives a video frame from camera device for preview purposes.

### Syntax

```
void OnVideoDeviceFrameRGB(  
    DeviceId,  
    FrameRGB,  
    FrameSize,  
    FrameWidth,  
    FrameHeight  
)
```

### Parameters

DeviceId(integer)  
This parameter value can be any number from zero to total number of video devices – 1. Each number corresponds to a particular video device.

FrameRGB(integer)  
Frame data in the form of RGB.

FrameSize(integer)  
Specifies the Frame data size.

FrameWidth(integer)  
Specifies the Frame width.

FrameHeight(integer)  
Specifies the Frame height.

### Example

```
void OnVideoDeviceFrameRGB(DeviceId, FrameRGB, FrameSize,  
    FrameWidth, FrameHeight)  
{  
}
```

### See Also

OnVideoRemoteFrameRGB(), OpenVideoDev()



## OnAddCallHistory()

The OnAddCallHistory() event triggers when component notifies about the call info for call history purposes.

### Syntax

```
void OnAddCallHistory(  
    AccountName,  
    OutboundCallType,  
    CallerName, CallerId,  
    DialNo,  
    StartTime, EndTime,  
    Duration,  
    HistoryTypeId  
)
```

### Parameters

AccountName(string)

The unique name to identify a SIP account.

Outbound(boolean)

This parameter value is call type (outbound or inbound).

CallerName(string)

This parameter specifies the Caller-Name.

CallerId(string)

This parameter specifies the Caller-Id.

DialNo(string)

This parameter specifies the dialer number.

StartTime(integer)

This parameter specifies the start time of the call.

EndTime(integer)

This parameter specifies the end time of the call.

Duration(integer)

This parameter specifies the total duration of the call.

HistoryTypeId(integer)

This parameter specifies the history type of the call.

0 = Outbound call type

1 = Inbound call type

2 = Missed call type

3 = Rejected call type

**Example**

```
OnAddCallHistory(AccountName, Outbound, CallerName, CallerId, DialNo,  
                  StartTime, EndTime, Duration, HistoryTypeId)  
{  
}
```

**See Also**

OnConnectedCall(), OnHungupCall()

**OnNetworkReachability()**

The OnNetworkReachability() event notifies the network availability.

**Syntax**

```
void OnNetworkReachability(Available)
```

**Parameters**

Available(boolean)

This parameter value can be 0 or 1.

**Example**

```
OnNetworkReachability(Available)
{
}
```

**See Also**

IsNetworkAvailable(), NetworkReachability()

**OnAudioDeviceMicVU()**

The OnAudioDeviceMicVU() event notifies the microphone's VU level.

**Syntax**

```
void OnAudioDeviceMicVU(LevelVU)
```

**Parameters**

LevelVU(boolean)

This parameter specifies the VU value (0 to 100).

**Example**

```
OnAudioDeviceMicVU(LevelVU)
{
}
```

**See Also**

OnAudioDeviceSpkVU(), AudioDeviceVU()

**OnAudioDeviceSpkVU()**

The OnAudioDeviceSpkVU() event notifies the speaker's VU level.

**Syntax**

```
void OnAudioDeviceSpkVU(LevelVU)
```

**Parameters**

LevelVU(boolean)

This parameter specifies the VU value (0 to 100).

**Example**

```
OnAudioDeviceSpkVU(LevelVU)
{
}
```

**See Also**

OnAudioDeviceMicVU(), AudioDeviceVU()

**OnVaxErrorMsg()**

The OnVaxErrorMsg() event triggers when a specific VaxVoIP exported method fails.

**Syntax**

```
void OnVaxErrorMsg(FuncName, ErrorMsg, ErrorCode)
```

**Parameters**

FuncName(string)

This parameter specifies VaxVoIP exported method name.

ErrorMsg(string)

This parameter specifies VaxVoIP method failure reason.

ErrorCode(integer)

This parameter specifies VaxVoIP method failure code.

**Example**

```
OnVaxErrorMsg(FuncName, ErrorMsg, ErrorCode)
{
}
```

**See Also**

GetVaxErrorCode(), GetVaxErrorMsg()

## **LIST OF ERROR CODES**

10	Make your SIP Account online first. Failed to initialize VaxVoIP Library.
11	Failed to open SIP listen port or provided SIP listen IP is incorrect.
12	License key is not valid or expired.
13	Failed to initialize task manager.
14	Unable to access sound input device.
15	Unable to access sound output device
16	Unable to capture sound input device.
17	Unable to capture sound output device.
18	Audio device does not support Mic volume.
19	Audio device does not support Speaker volume.
20	Recording media initialization failed.
21	Unable to access wave file.
22	Provided SIP URI is invalid.
23	Selected codec is not supported.
24	Error to create SDP Packet.
25	Error to create SIP CONNECTION Packet.
26	Error to create SIP REGISTER Packet.
27	Error to create SIP UN-REGISTER Packet.
28	Error to create SIP DISCONNECT Packet.
29	Provided LineNo is invalid.
30	Provided LineNo is busy.
31	Provided LineNo is not ready to use.
32	Invalid Call-ID.
33	Function parameter is invalid.
34	Provided LineNo is not in Voice Session.
35	Failed to read sound file.
36	Failed to write sound file.
37	Unsupported sound file format.
38	Error to create SIP CANCEL Packet.
39	License limit exceeded.
40	Chat contact does not exist.
41	Chat contact subscription does not exist.
42	Unable to create presence notify request.
43	Unable to create presence request.
44	Unable to capture video device.
45	Functionality is not supported.
46	Network is not available.
47	BusyLamp contact does not exist.
48	Account does not exist.
49	Account initialization failed.
50	Unable to create socket dispatcher.
51	Unable to post socket dispatcher.
52	Unable to process socket dispatcher.
53	Socket binding address failed.
54	Socket binding address already in use.

55	Socket general error.
56	Socket failed time-out.
57	Failed to assign/allocate RTP port.
58	Crypto audio or video media mismatched.
59	Connection to the destination address failed.
60	Unable to capture or bind to the destination address.
61	Unable to create thread dispatcher.
62	Unable to post message to thread dispatcher.
63	Unable to process to thread dispatcher.
64	Terminating abnormally thread dispatcher.