SIP RECORDING SDK v2.8

TECHNICAL DOCUMENTATION

VERSION 3.2

INTRODUCTION AND QUICK START 4

EXPORTED FUNCTIONS 5

SetLicenseKey() 5
GetVaxObjectError() 6
Initialize() 7
SetListenPortRangeRTP() 9
AddListenIP() 10
UnInitialize() 11
AddUser() 12
RemoveUser() 14
RegisterUserExpiry() 15
AcceptRegister() 16
RejectRegister() 17
AuthRegister() 18
AddLine() 19
RemoveLine() 21
RegisterLine() 22
UnRegisterLine() 23
AddDirectProxySIP() 24
RemoveDirectProxySIP() 26
AcceptCallSession() 27
RejectCallSession() 28
CloseCallSession() 29
RecordWaveStartToCallSession() 30
RecordWaveStopToCallSession() 31
RecordWavePauseToCallSession() 32
VoiceSessionLost() 33
DiagnosticLogSIP() 34
StartVaxTeleTick() 35
StopVaxTeleTick() 36
SetUserAgentName() 37
GetUserAgentName() 38
SetSessionNameSDP() 39
GetSessionNameSDP() 40
SetUserData() 41
GetUserData() 42
SetLineData() 43
GetLineData() 44
SetDirectProxyData() 45
GetDirectProxyData() 46
SetCallSessionData() 47
GetCallSessionData() 48
AddCustomHeader() 49
RemoveCustomHeader() 50
RemoveCustomHeaderAll() 51
CallSessionSendStatusResponse() 52
TrvsUserStart() 53
TrvsUserNext() 54
TrvsLineStart() 55
TrvsLineNext() 56
TrvsDirectProxyStart() 57

TrvsDirectProxyNext() 58
TrvsCallSessionStart() 59
TrvsCallSessionNext() 60
GetUserCount() 61
GetLineCount() 62
GetDirectProxyCount() 63
GetCallSessionCount() 64

EXPORTED EVENTS 65
OnVaxErrorLog() 65
OnLineRegisterTrying() 66
OnLineRegisterFailed() 67
OnLineRegisterSuccess() 68
OnLineUnRegisterTrying() 69
OnLineUnRegisterFailed() 70
OnLineUnRegisterSuccess() 71
OnUnRegisterUser() 72
OnRegisterUser() 73
OnRegisterUserSuccess() 74
OnRegisterUserFailed() 75
OnIncomingCall() 76
OnCallSessionConnected() 78
OnCallSessionLost() 79
OnCallSessionHangup() 80
OnCallSessionTimeout() 81
OnCallSessionCancelled() 82
OnOutgoingDiagnosticLog() 83
OnIncomingDiagnosticLog() 84
OnVaxTeleTick() 85
OnCallSessionCreated() 86
OnCallSessionClosed() 87
OnCallSessionRecordedPCM() 88

LIST OF ERROR CODES 89

LIST OF SIP RESPONSES (SIP RFC 3261) 91

SIP REC CLIENT REGISTRATION FLOW 92

SIP CALL RECORDING SESSION FLOW 92

INTRODUCTION AND QUICK START

The VaxVoIP SIP CALL RECORDING COM (Component Object Model) component VaxTeleServerREC.dll. It is a collection of functions and events that allows you to develop SIP REC Protocol based Call Recording Server.

The VaxVoIP SIP REC COM component's exported (VaxTeleServerREC.dll) functions and events enable the rapid development of SIP REC based Call Recording Servers and allows you to use it in your call centers, in your offices and other SIP based VoIP services.

EXPORTED FUNCTIONS

SetLicenseKey()

The trial version of VaxVoIP SDK has trial period limitation of 30 days, so a license key is required after 30 days to avoid evaluation message box. License keys are delivered to customers on order.

The SetLicenseKey() method is used to make the trial version working as registered version without expiry and trial period limitation.

Syntax

```
SetLicenseKey(LicenseKey)
```

Parameters

LicenseKey (string)
The value of this parameter is license key provided by the company.

Return Value

No return value.

Example

```
SetLicenseKey("LicenseKey")
Initialize("sipsdk.com", "192.168.0.3", 5060, "192.168.0.3", -1)
```

See Also

Initialize(), GetVaxObjectError()

GetVaxObjectError()

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

Please see **LIST OF ERROR CODES** for more details.

Syntax

```
integer GetVaxObjectError()
```

Parameters

No parameters.

Return Value

The GetVaxObjectError() returns the error code.

Example

```
SetLicenseKey("LicenseKey")

Result = Initialize("192.168.0.3", "192.168.0.3", 5060, "192.168.0.3", -1)
if(Result == 0) GetVaxObjectError()
```

See Also

Initialize(), SetLicenseKey()

Initialize()

The Initialize() function initializes VaxVoIP SIP REC COM component. It allocates SIP listen port and starts listening for incoming SIP requests and triggers the COM (Component Object Model) events accordingly. It also allocates RTP port(s) to receive voice streams.

Syntax

```
boolean Initialize(
                    DomainRealm,
                    SIPListenIP,
                    SIPListenPort,
                    RTPListenIP,
                    RTPListenPort
                    )
```

Parameters

DomainRealm (string)

This parameter value is internally used to create SIP URI(s). It is used as "realm" field in SIP packets during the authentication of SIP REC clients and SIP REC based call processing.

This parameter can be blank/empty. If its value is blank/empty string then SIPListenIP parameter value is use in SIP authentication process and to create SIP URI(s).

Note: It is not necessary that an IP-address should be assigned to DomainRealm.

SIPListenIP (string)

The value of this parameter specifies the IP on which VaxVoIP based SIP REC Server listen and receives incoming SIP REC requests.
It can be the IP address assigned to the computer on which VaxVoIP REC COM integrated SIP REC server is running.

SIPListenPort (integer)

The value of this parameter specifies the port number for SIP REC server to receive SIP REC requests. Standard SIP listen port is 5060

RTPListenIP (string)

The value of this parameter specifies the IP address for VaxVoIP REC SDK's integrated SIP server to receive voice streams. If multiple IP addresses are assigned to the computer on which SIP REC server is running then SIPListenIP and RTPListenIP can be different.

RTPListenPort (integer)

The RTPListenPort specifies the start port number for the range of RTP ports allocation. The allocation starts from the parameter value and

increments for even number as for RTP compliance, RTP port number must be an even number.

This parameter can also be assigned value -1 which let VaxVoIP REC server to randomly choose a RTP port and allocates to the REC call for voice streaming.

If RTPListenPort = –1 then

Call-1 3828
Call-2 4042
Call-3 8922

Value is –1 then random RTP port number assigns to each call.

If RTPListenPort = 2234

Call-1 2236
Call-2 2238
Call-3 2240

Note: According to the SIP RFC 2327, the value of listen port must be in the range of 1024 to 65535 inclusive, for RTP compliance it should be an even number.

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 GetVaxObjectError() method.

Example

```
Initialize("", "192.168.0.3", 5060, "192.168.0.3", -1)
Initialize("sip.vaxtele.com", "10.18.0.3", 5060, "10.18.0.3", 6066)
Initialize("vaxtele.com", "209.237.151.17", 5060, "209.237.151.18", -1)
```

See Also

UnInitialize(), GetVaxObjectError(), SetListenPortRangeRTP()

SetListenPortRangeRTP()

The SetListenPortRangeRTP() sets the given range, so that VaxVoIP REC server allocates/opens the listen RTP port within that range.

Note: According to the SIP RFC 2327, the value of listen port must be in the range of 1024 to 65535 inclusive, for RTP compliance it should be an even number.

Syntax

```
boolean SetListenPortRangeRTP(ListenStartPort, ListenEndPort)
```

Parameters

ListenStartPort (integer)
The value of this parameter specifies starting value of the range.

ListenEndPort (integer)
The value of this parameter specifies end value of the range.

Return Value

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

Example

```
Result = Initialize("sipsdk.com", "9.7.11.17", 5060, "9.7.11.17", -1)
if(Result == 0) GetVaxObjectError()

SetListenPortRangeRTP(8088, 32406)
```

See Also

Initialize(), GetVaxObjectError()

AddListenIP()

The AddListenIP() function is used to add additional listen IP addresses to VaxVoIP REC SDK integrated SIP REC server application.

When VaxVoIP REC SDK based application is behind the Firewall/router/NAT and port forwarding is enabled from the router's end. In this scenario, initialize VaxVoIP REC COM with the private IP Address assigned to the computer and add public IP address assigned to the router by using AddListenIP() function.

When multiple IP addresses are assigned to a computer and requires VaxVoIP REC SDK to work with those IP addresses.

Syntax

```
boolean AddListenIP(SIPListenIP, RTPListenIP)
```

Parameters

SIPListenIP (string)
: This parameter specifies the listen IP to be used for SIP packets.

RTPListenIP (string)
: This parameter specifies the listen IP to be used for voice streaming.

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 GetVaxObjectError() method.

Example

```
Result = Initialize("sipsdk.com", "", 5060, "", -1)
if(Result = 0) GetVaxObjectError()

AddListenIP("66.77.88.99", "66.77.88.99")
SetListenPortRangeRTP(8000, 32406)
```

Run REC Server behind Firewall/router, from the router settings;

- Forward inbound UDP ports 8000 to 32406
- Enable outbound UDP ports 1024 to 65535
- 192.168.0.25 is the IP address assigned to the computer behind router.
- 66.77.88.99 is the IP address assigned to the router.

See Also

Initialize(), SetListenPortRangeRTP(), GetVaxObjectError()

UnInitialize()

The UnInitialize() function simply un-allocates all the resources previously occupied by Initialize() function.

Syntax

```
UnInitialize()
```

Parameters

No parameters.

Return Value

No return value.

Example

```
UnInitialize()
```

See Also

Initialize()

AddUser()

The AddUser() function add users to the SIP server developed by VaxVoIP REC COM component. VaxVoIP REC COM (VaxTeleServerREC.dll) component internally creates a list of users to process the SIP client REC registration and call requests.

Add user by AddUser() function so that user login and password can be used in any SIP REC client to connect and register it with VaxVoIP REC SDK based call-recording server.

Syntax

```
boolean AddUser(
                    UserName,
                    Password,
                    CodecList
                    )
```

Parameters

UserName (string)
: This parameter value specifies the user's login.

Password (string)
: This parameter value specifies the password of user's login.

CodecList (string)
: This parameter value specifies the list of voice codecs to be added in the call request.

  Supported codecs:

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

  "03" specifies that only GSM & G711u are supported to the call voice stream and GSM priority is higher.

  "4213" specifies that supported codecs for the call-request are G729, G711a, iLBC and G711u.
  Highest priority codec is 4 (G729) and lowest priority codec is 3 (G711u).

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 GetVaxObjectError() method.

Example

```
SetLicenseKey("LicenseKey")
Initialize("sip.vaxtele.com", "192.168.0.3", 5060, "192.168.0.3", -1)

Result = AddUser("9090", "123", "01")
if(Result == 0) GetVaxObjectError()
```

See Also

RemoveUser(), GetVaxObjectError()

RemoveUser()

The RemoveUser() function removes the provided user previously added by AddUser() function.

Syntax

```
RemoveUser(UserName)
```

Parameters

UserName (string)
This parameter value specifies the user's login.

Return Value

No return value.

Example

```
RemoveUser("9092")
RemoveUser("Jason")
```

See Also

AddUser()

RegisterUserExpiry()

The RegisterUserExpiry() function configures that which value of Expiration Interval should be used by VaxVoIP SIP REC server during SIP client registration process.

If -1 value is set then during the registration process, VaxVoIP SIP REC server accepts the Expiration Interval requested by the SIP REC client. Otherwise SIP REC Server ignores the Expiration Interval requested by the SIP REC client and sends the time, which is set by RegisterUserExpiry() function.

If RegisterUserExpiry() function is not called then default value 30 seconds is sent to the SIP REC client(s).

Note: In SIP packet Expires header designates the Expiration Interval of the registration.

Syntax

```
boolean RegisterUserExpiry(Expiry)
```

Parameters

Expiry (integer)
The Expire parameter specifies the time interval in seconds.

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 GetVaxObjectError() method.

Example

```
RegisterUserExpiry(1800)
RegisterUserExpiry(-1)
```

See Also

AddUser(), OnRegisterUser()

AcceptRegister()

The AcceptRegister() function accepts the registration request which receives from SIP REC client.

During the SIP registration process, SIP REC clients send registration requests. SIP REC servers authorize login and password and then accept those registration requests.

VaxVoIP REC SDK triggers OnRegisterUser() event upon receiving a registration request. When AcceptRegister() function is called in this event, it simply accepts the registration request without authorizing the login and password.

AcceptRegister() function skips the login authorization process and accepts the registration request.

Please see **SIP REC CLIENT REGISTRATION FLOW** for more details.

Syntax

```
boolean AcceptRegister(UserName)
```

Parameters

UserName (string)
: This parameter value specifies the unique user name to identify a specific user.

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 GetVaxObjectError() method.

Example

```
OnRegisterUser(UserLogin, Domain, UserAgentName, FromIP, FromPort)
{
        AcceptRegister(UserLogin)
}
```

See Also

AuthRegister(), RejectRegister(), OnRegisterUser()

RejectRegister()

The RejectRegister() function rejects the registration request receives from SIP REC client.

VaxVoIP REC SDK triggers OnRegisterUser() event upon receiving registration request. When RejectRegister() function is called in this event it simply rejects the registration request

Please see **LIST OF SIP RESPONSES** for more details.

Syntax

```
boolean RejectRegister(
                        UserName,
                        StatusCode,
                        ReasonPhrase
                        )
```

Parameters

UserName (string)
: This parameter value specifies the unique user name to identify a specific user.

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

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

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 GetVaxObjectError() method.

Example

```
OnRegisterUser(UserLogin, Domain, UserAgentName,  FromIP, FromPort)
{
        RejectRegister(UserLogin, 404, "Not Found")
}
```

See Also

AuthRegister(), AcceptRegister(), OnRegisterUser()

AuthRegister()

The AuthRegister() function starts the authentication process for a specified user before accepting registration request.

The SIP REC clients send SIP register request to connect and register to SIP REC based server. When VaxVoIP REC SDK receives register request then it triggers OnRegisterUser() event and starts the authentication process by calling AuthRegister() function.

Use AuthRegister() function and then VaxVoIP REC SDK;

1. Starts SIP authentication process.
2. Completes authentication process.
3. Accepts registration (register) request.

Please see **SIP REC CLIENT REGISTRATION FLOW** for more details.

Syntax

```
boolean AuthRegister(UserName)
```

Parameters

UserName (string)
: This parameter value specifies the unique user name to identify a specific user.

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 GetVaxObjectError() method.

Example

```
OnRegisterUser(UserLogin, Domain, UserAgentName, FromIP, FromPort)
{
        AuthRegister(UserLogin)
}
```

See Also

AcceptRegister(), RejectRegister(), OnRegisterUser()

AddLine()

The AddLine() function adds third party supported SIP/VoIP server provided account setting as line in VaxVoIP REC SDK thus allowing VaxVoIP REC SDK to connect with other SIP/VoIP servers and receive SIP REC requests from them.

Syntax

```
boolean AddLine(
                    LineName,
                    DisplayName,
                    UserName,
                    AuthUser
                    AuthPwd,
                    DomainRealm,
                    ProxySIP,
                    OutboundProxy,
                    CodecList
                    )
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

DisplayName (string)
: This Parameter value specifies the display name for user which is provided by IP-Telephony service provider or third party SIP server.

UserName (string)
: This Parameter value specifies the user name which is provided by IP-Telephony service provider or third party SIP server.

AuthUser (string)
: This Parameter value specifies the user Login which is provided by IP-Telephony service provider or third party SIP server.

AuthPwd (string)
: This Parameter value specifies the password which is provided by IP-Telephony service provider or third party SIP server.

DomainRealm (string)
: This Parameter value is provided by IP-Telephony service provider or third party SIP server.

ProxySIP (string)
: This Parameter value is provided by IP-Telephony service provider or third party SIP server.

OutboundProxy (string)

This Parameter value is provided by IP-Telephony service provider or third party SIP server.

CodecList (string)

This parameter value specifies the list of voice codecs to be added in the call request.
Supported codecs:

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

"4213" specifies that supported codec for the call request are G729, G711a, iLBC and G711u.
Highest priority codec is 4 (G729) and lowest priority codec is 3 (G711u).

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 GetVaxObjectError() method.

Example

```
AddLine("LineNameA", "8034", "8034", "8034", "9341", "sipsdk.com",
        "192.168.0.3", "", "0134")
```

See Also

RemoveLine(), RegisterLine(), UnRegisterLine(), GetVaxObjectError()

RemoveLine()

The RemoveLine() function removes the provided line which was previously added by AddLine() function.

Syntax

```
RemoveLine(LineName)
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

Return Value

No return value.

Example

```
RemoveLine("abc")
RemoveLine("LineName")
```

See Also

AddLine()

RegisterLine()

The RegisterLine() function registers the line to external third party. VaxVoIP REC SDK triggers registration related events during line registration process e.g. OnLineRegisterTrying(), OnLineRegisterSuccess() etc.

The line should be added by AddLine() function prior to registration.

Syntax

```
boolean RegisterLine(
                        LineName,
                        Expire
                        )
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

Expire (integer)
: The Expire parameter specifies the time interval (in seconds) after which the registration with server will be refreshed consequently server will remain updated about the present client status.

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 GetVaxObjectError() method.

Example

```
RegisterLine("abc", 1800)
RegisterLine("LineName", 3600)
```

See Also

UnRegisterLine(), OnLineRegisterSuccess(), OnLineRegisterTrying(), AddLine(), OnLineRegisterFailed(), GetVaxObjectError()

UnRegisterLine()

The UnRegisterLine() function unregisters the provided line which was registered by RegisterLine() function. VaxVoIP REC SDK triggers unregister process related events during line unregistration process e.g. OnLineUnRegisterTrying(), OnLineUnRegisterSuccess() etc.

Syntax

```
boolean UnRegisterLine(LineName)
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

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 GetVaxObjectError() method.

Example

```
UnRegisterLine("abc")
UnRegisterLine("2" )
```

See Also

RegisterLine(), AddLine(), OnLineUnRegisterSuccess(), GetVaxObjectError() OnLineUnRegisterTrying(), OnLineUnRegisterFailed()

AddDirectProxySIP()

The AddDirectProxySIP() function with function AcceptCallSession() provides functionality to receive SIP REC call requests directly on listen IP and port.

Syntax

```
boolean AddDirectProxySIP(
                          ProxyName,
                          AuthLogin,
                          AuthPwd,
                          DomainRealm,
                          ProxyIP,
                          ProxyPort,
                          CodecList
                          )
```

Parameters

ProxyName(string)
: This parameter specifies unique direct proxy name.

AuthLogin(string)
: This Parameter value specifies the user Login.

AuthPwd(string)
: This Parameter value specifies the user password.

DomainRealm(string)
: This parameter specifies the domain realm.

ProxyIP(string)
: This parameter specifies the remote IP.

ProxyPort(integer)
: This parameter specifies the remote port.

CodecList(string)
: This parameter value specifies the list of voice codecs to be added in the call request.

  Supported codecs:

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

"4213" specifies that supported codec for the call request are G729, G711a, iLBC and G711u. Highest priority codec is 4 (G729) and lowest priority codec is 3 (G711u).

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 GetVaxObjectError() method.

Example

```
AddDirectProxySIP("DirectCOMM", "", "", "sipsdk.com",
                        "192.168.0.20", 5060, "4213")
```

See Also

RemoveDirectProxySIP(), GetVaxObjectError()

RemoveDirectProxySIP()

The RemoveDirectProxySIP() function removes the provided direct proxy which was previously added by AddDirectProxySIP() function.

Syntax

```
RemoveDirectProxySIP(ProxyName)
```

Parameters

ProxyName(string)
This parameter specifies unique direct proxy name.

Return Value

No return value.

Example

```
RemoveDirectProxySIP("DirectCOMM")
```

See Also

AddDirectProxySIP(), GetVaxObjectError()

AcceptCallSession()

The AcceptCallSession() function accepts an incoming SIP REC call request and allows to start the recording by using RecordWaveStartToCallSession() function.

Syntax

```
boolean AcceptCallSession(
                        SessionId,
                        Timeout
                        )
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

Timeout (integer)
: This parameter specifies the time interval (in seconds) for which VaxVoIP REC SDK waits for call-recording session to be connected/established, if call-recording session is not established/connected within specified time interval then timeout occurs as a result OnCallSessionTimeout() event triggers.

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 GetVaxObjectError() method.

Example

```
OnIncomingCall(SessionId, CallerName, CallerId, CalleeName, CalleeId,
                RecXML, FromPeerType, FromPeerName, UserAgentName,
                FromIP, FromPort)
{
        AcceptCallSession(SessionId, 20)
        RecordWaveStartToCallSession(SessionId, "RecordCall.wav")
}
```

See Also

RejectCallSession(), CloseCallSession(), GetVaxObjectError(), OnIncomingCall()

RejectCallSession()

The RejectCallSession() function rejects an incoming SIP REC call request for a specific call-recording session.

Syntax

```
boolean RejectCallSession(
                        SessionId,
                        StatusCode,
                        ReasonPhrase
                        )
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

StatusCode (integer)
: This parameter specifies SIP response status.
LIST OF SIP RESPONSES

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

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 GetVaxObjectError() method.

Example

```
OnIncomingCall(SessionId, CallerName, CallerId, CalleeName, CalleeId,
                RecXML, FromPeerType, FromPeerName, UserAgentName,
                FromIP, FromPort)
{
        RejectCallSession(SessionId, 403, "Forbidden")
}
```

See Also

AcceptCallSession(), CloseCallSession(), OnIncomingCall()

CloseCallSession()

The CloseCallSession() function closes a call-recording session and REC call in that session gets disconnected.

Syntax

```
boolean CloseCallSession(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

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 GetVaxObjectError() method.

Example

```
CloseCallSession(SessionId)
```

See Also

RejectCallSession(), AcceptCallSession(), OnCallSessionClosed()

RecordWaveStartToCallSession()

The RecordWaveStartToCallSession() function starts recording the voice stream of a particular call-recording session to a wave file (.wav).

VaxVoIP REC SDK use (8000Hz, 16bit, mono) format and creates uncompressed wave file (.wav) to save CPU cycles.

Syntax

```
boolean RecordWaveStartToCallSession(
                                        SessionId,
                                        FileName
                                        )
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

FileName (string)
: The value of this parameter specifies the file name.

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 GetVaxObjectError() method.

Example

```
OnCallSessionConnected(SessionId)
{
        RecordWaveStartToCallSession(SessionId, "Record.wav")
}

OnCallSessionClosed(SessionId, ReasonCode)
{
        RecordWaveStopToCallSession(SessionId)
}
```

See Also

RecordWaveStopToCallSession(), RecordWavePauseToCallSession(), GetVaxObjectError()

RecordWaveStopToCallSession()

The RecordWaveStopToCallSession() function stops the recording of voice stream in a call-recording session.

Syntax

```
boolean RecordWaveStopToCallSession(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

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 GetVaxObjectError() method.

Example

```
OnCallSessionConnected(SessionId)
{
        RecordWaveStartToCallSession(SessionId, "RecordCall.wav")
}

OnCallSessionClosed(SessionId, ReasonCode)
{
        RecordWaveStopToCallSession(SessionId)
}
```

See Also

RecordWaveStartToCallSession(), RecordWavePauseToCallSession(), GetVaxObjectError()

RecordWavePauseToCallSession()

The RecordWavePauseToCallSession() function pauses the recording to wave file.

Syntax

```
boolean RecordWavePauseToCallSession(SessionId, Enable)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

Enable (Boolean)
This parameter value can be 0 or 1. Assign value 1 to pause the recording process or 0 to un-pause the recording process.

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 GetVaxObjectError() method.

Example

```
OnCallSessionConnected(SessionId)
{
	RecordWaveStartToCallSession(SessionId, "Record.wav")
}

OnVaxTeleTick(TickId)
{
	RecordWavePauseToCallSession(SessionId, 1)
}
```

See Also

RecordWaveStartToCallSession(), RecordWaveStopToCallSession(), GetVaxObjectError()

VoiceSessionLost()

The VoiceSessionLost() method enables the detection of voice data. VaxVoIP REC SDK waits for define interval of time for voice data, if it does not receive data within that interval then it triggers OnSessionLost() event.

Syntax

```
boolean VoiceSessionLost(Enable, Timeout)
```

Parameters

Enable (boolean)
: The value of this parameter can be 0 or 1. Assign value 1 to this parameter to enable voice session lost detection or 0 to disable it.

Timeout (integer)
: This parameter value specifies the time interval (in second) for detection of voice data.

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 GetVaxObjectError() method.

Example

```
VoiceSessionLost(1, 20)
```

See Also

OnCallSessionLost(), GetVaxObjectError()

DiagnosticLogSIP()

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

VaxVoIP REC SDK triggers OnIncomingDiagnosticLog()/OnOutgoingDiagnosticLog() event when it receives/sends SIP messages.

Syntax

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

Parameters

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 GetVaxObjectError() method.

Example

```
DignosticLogSIP(1, 0)
```

See Also

OnOutgoingDiagnosticLOg(), OnIncomingDiagnosticLog(), GetVaxObjectError()

StartVaxTeleTick()

The function StartVaxTeleTick() sets a time interval, certain task is performed on expiry of this set interval.

When StartVaxTeleTick() method is called, it sets a timer tick internally and trigger the tick event OnVaxTeleTick() after that specific time.

StartVaxTeleTick() function with event OnVaxTeleTick() can be used to perform a task after specific interval of time.

Syntax

```
boolean StartVaxTeleTick(TickId, Elapse)
```

Parameters

TickId (integer)
: This parameter specifies the unique tick identification.

Elapse (integer)
: The value of this parameter specifies the time (milliseconds).

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 GetVaxObjectError() method.

Example

```
/* Triggers OnVaxTeleTick() after every 8 seconds */

StartVaxTeleTick(2001, 8000)
```

See Also

StopVaxTeleTick(), OnVaxTeleTick(), GetVaxObjectError()

StopVaxTeleTick()

The StopVaxTeleTick() method is used to stop the time counter for specified tick. It works just like KillTimer() win32 API.

Syntax

```
boolean StopVaxTeleTick(TickId)
```

Parameters

TickId (integer)
This parameter specifies the unique tick identification

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 GetVaxObjectError() method.

Example

```
/* Triggers OnVaxTeleTick() after every 8 seconds */

StartVaxTeleTick(2001, 8000)

StopVaxTeleTick(2001)
```

See Also

StartVaxTeleTick(), OnVaxTeleTick(), GetVaxObjectError()

SetUserAgentName()

The SetUserAgentName() function sets the user-agent header field of SIP packet.

Syntax

```
boolean SetUserAgentName(Name)
```

Parameters

Name (string)
This parameter value specifies the User agent name.

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 GetVaxObjectError() method.

Example

```
Result = SetUserAgentName("abc")
if(Result == 0) GetVaxObjectError()
```

See Also

GetUserAgentName(), GetVaxObjectError()

GetUserAgentName()

The GetUserAgentName() function returns the user-agent value previously set by calling SetUserAgentName() function.

Syntax

```
string GetUserAgentName()
```

Parameters

No parameters.

Return Value

The function returns the user agent name otherwise empty string.

Example

```
GetUserAgentName()
```

See Also

SetUserAgentName()

SetSessionNameSDP()

The SetSessionNameSDP() function sets the session-name field of SDP part of SIP packet.

Syntax

```
boolean SetSessionNameSDP(Name)
```

Parameters

Name (string)
: This parameter specifies the value that is to be set as session name of SIP packet.

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 GetVaxObjectError() method.

Example

```
SetSessionNameSDP("xyz")
```

See Also

GetSessionNameSDP(), GetVaxObjectError()

GetSessionNameSDP()

The GetSessionNameSDP() function returns the session-name value previously set by SetSessionNameSDP() function.

Syntax

```
string GetSessionNameSDP()
```

Parameters

No parameters.

Return Value

The function returns the session name otherwise empty string.

Example

```
GetSessionNameSDP()
```

See Also

SetSessionNameSDP()

SetUserData()

The SetUserData() function attaches an instance/object of custom class as custom data to a specific user.

For further details, please see sample code after downloading from the website.

Syntax

```
boolean SetUserData(
                        UserName,
                        CustomData
                         )
```

Parameters

UserName(String)
This parameter specifies the User name.

CustomData(ClassObject)
This parameter is an instance of custom data class object.

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 GetVaxObjectError() method.

Example

```
AddUser("UserName", "Password", "03421")

CCustomUserData objData = new CCustomUserData()
SetUserData("UserName", objData)

OnRegisterUserSuccess(UserLogin)
{
        CCustomUserData objData = GetUserData(UserLogin)
        objData.SetRegistered(TRUE)
}
```

See Also

GetUserData(), GetVaxObjectError()

GetUserData()

The GetUserData() returns the instance/object of custom class attached to a specific user. Please have a look at sample code after downloading from the website.

Syntax

```
ClassObject GetUserData(UserName)
```

Parameters

UserName(String)
This parameter specifies the User name.

Return Value

If the function succeeds, the return value is instance/object of custom class.

If the function fails, the return value is nil, NULL or 0. To get extended error information, call GetVaxObjectError().

Example

```
AddUser("UserName", "Password", "03421")

CCustomUserData objData = new CCustomUserData()
SetUserData("UserName", objData)

OnUnRegisterUser(UserLogin)
{
        CCustomUserData objData = GetUserData(UserLogin)
        objData.SetRegistered(FALSE)
}
```

See Also

SetUserData(), GetVaxObjectError()

SetLineData()

The SetLineData() function allows to attach an instance/object of custom class as custom data to a specific line.

For further details, please see sample code after downloading from the website.

Syntax

```
boolean SetLineData(
                        LineName,
                        CustomData
                       )
```

Parameters

LineName(String)
This parameter specifies the Line name.

CustomData(ClassObject)
This parameter is an instance of custom data class object.

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 GetVaxObjectError() method.

Example

```
AddLine("LineName", "DisplayName", "UserName", "AuthUser", "AuthPwd"
          "DomainRealm", "ProxySIP", "", "02314")

CCustomLineData objData = new CCustomLineData()
SetLineData("LineName", objData)

OnLineRegisterSuccess(LineName)
{
        CCustomLineData objData = GetLineData(LineName)
        objData.bRegistered = TRUE
}
```

See Also

GetLineData(), AddLine(), GetVaxObjectError()

GetLineData()

The GetLineData() returns the instance/object of custom class attached to a specific line.

Please have a look at sample code after downloading from the website.

Syntax

```
ClassObject GetLineData(LineName)
```

Parameters

LineName(String)
This parameter specifies the Line name.

Return Value

If the function succeeds, the return value is instance/object of custom class.

If the function fails, the return value is nil, NULL or 0. To get extended error information, call GetVaxObjectError().

Example

```
AddLine("LineName", "DisplayName", "UserName", "AuthUser", "AuthPwd"
        "DomainRealm", "ProxySIP", "", "02314")

CCustomLineData objData = new CCustomLineData()
SetLineData("LineName", objData)

OnLineRegisterFailed(LineName)
{
    CCustomLineData objData = GetLineData(LineName)
    objData.bRegistered = FALSE
}
```

See Also

SetLineData(), GetVaxObjectError()

SetDirectProxyData()

The SetDirectProxyData() function attaches an instance/object of custom class as custom data to a specific proxy.

Syntax

```
boolean SetDirectProxyData(
                            ProxyName,
                            CustomData
                           )
```

Parameters

ProxyName(String)
This parameter specifies the proxy name.

CustomData(ClassObject)
This parameter is an instance of custom data class object.

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 GetVaxObjectError() method.

Example

```
AddDirectProxySIP(ProxyName, AuthLogin, AuthPwd, DomainRealm
                  ProxyIP, ProxyPort, CodecList)

CCustomProxyData objData = new CCustomProxyData()
SetDirectProxyData(ProxyName, objData)
```

See Also

GetDirectProxyData(), GetVaxObjectError()

GetDirectProxyData()

The GetDirectProxyData() returns the instance/object of custom class attached to a specific proxy.

Syntax

```
ClassObject GetDirectProxyData(ProxyName)
```

Parameters

ProxyName(String)
This parameter specifies the Proxy name.

Return Value

If the function succeeds, the return value is instance/object of custom class.

If the function fails, the return value is nil, NULL or 0. To get extended error information, call GetVaxObjectError().

Example

```
RemoveDirectProxySIP(ProxyName)

CCustomProxyData objData = GetDirectProxyData(ProxyName)
objData = nill
```

See Also

SetDirectProxyData(), GetVaxObjectError()

SetCallSessionData()

The SetCallSessionData() function allows to attach an instance/object of custom class as custom data to a specific call-recording session.

For further details, please see sample code after downloading from the website.

Syntax

```
boolean SetCallSessionData(
                            SessionId,
                            CustomData
                           )
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

CustomData(ClassObject)
This parameter is an instance of a custom data class object.

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 GetVaxObjectError() method.

Example

```
OnCallSessionCreated(SessionId, ReasonCode)
{
        CCustomCallSessionData objData = new CCustomCallSessionData()
        SetCallSessionData(SessionId, objData)
}
```

See Also

GetCallSessionData(), GetVaxObjectError()

GetCallSessionData()

The GetCallSessionData() returns the instance/object of custom class attached to a specific call-recording session.

Please have a look at sample code after downloading from the website.

Syntax

```
ClassObject GetCallSessionData(SessionId)
```

Parameters

SessionId(integer)
This parameter specifies an identification of a call-recording session.

Return Value

If the function succeeds, the return value is instance/object of custom class.

If the function fails, the return value is nil, NULL or 0. To get extended error information, call GetVaxObjectError().

Example

```
OnCallSessionClosed(SessionId, ReasonCode)
{
	CCustomCallSessionData objData = GetCallSessionData(SessionId)

	delete objCustomCallData
	objCustomCallData = nil
}
```

See Also

SetCallSessionData(), GetVaxObjectError()

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, ACK, CANCEL, BYE, OPTIONS

Syntax

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

Parameters

ReqId (integer)
: This parameter specifies a unique identification of a SIP request. Supported ReqId values are;

  0 = INVITE
  1 = REFER
  2 = CANCEL
  3 = BYE

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 GetVaxObjectError() method.

Example

```
Result = Initialize("sipsdk.com", "192.168.0.25", 5060, "192.168.0.25", -1)
if(Result = 0) GetVaxObjectError()

AddCustomHeader(0, "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
2 = CANCEL
3 = BYE

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 GetVaxObjectError() 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
2 = CANCEL
3 = BYE

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 GetVaxObjectError() method.

Example

```
RemoveCustomHeaderAll(0)
```

See Also

AddCustomHeader(), RemoveCustomHeader()

CallSessionSendStatusResponse()

The CallSessionSendStatusResponse() function is used to send SIP responses (Trying, Ringing, Not found etc.) to an incoming REC CALL request in a call-recording session.

Syntax

```
boolean CallSessionSendStatusResponse(
                                        SessionId,
                                        StatusCode,
                                        ReasonPhrase
                                        )
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

StatusCode (integer)
: This parameter specifies SIP response status.
LIST OF SIP RESPONSES

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

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 GetVaxObjectError() method.

Example

```
OnIncomingCall(SessionId, CallerName, CallerId, CalleeName, CalleeId,
                RecXML, FromPeerType, FromPeerName, UserAgentName,
                FromIP, FromPort)
{

        CallSessionSendStatusResponse(SessionId, 180, "Ringing")

}
```

See Also

StartVaxTeleTick(), AcceptCallSession()

TrvsUserStart()

The TrvsUserStart() starts iteration in the list of users added by using AddUser() function.

Syntax

```
boolean TrvsUserStart()
```

Parameters

No parameters.

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 GetVaxObjectError() method.

Example

```
TrvsUserStart()

Loop(GetUserCount())
{
	UserName = TrvsUserNext()
}
```

See Also

TrvsUserNext(), GetVaxObjectError()

TrvsUserNext()

The TrvsUserNext() iterates and returns user name.

Syntax

```
string TrvsUserNext()
```

Parameters

No parameters.

Return Value

On successful execution this function returns user-name value otherwise it returns empty string value.

Example

```
TrvsUserStart()

Loop()
{
        UserName = TrvsUserNext()
        if(UserName.length = 0) exit loop
}
```

See Also

TrvsUserStart(), AddUser()

TrvsLineStart()

The TrvsLineStart() starts iteration in the list of lines added by using AddLine() function.

Syntax

```
boolean TrvsLineStart()
```

Parameters

No parameters.

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 GetVaxObjectError() method.

Example

```
TrvsLineStart()

Loop(GetLineCount())
{
	LineName = TrvsLineNext()
}
```

See Also

TrvsLineNext(), GetVaxObjectError(), AddLine()

TrvsLineNext()

The TrvsLineNext() iterates and returns line name.

Syntax

```
string TrvsLineNext()
```

Parameters

No parameters.

Return Value

On successful execution this function returns line-name value otherwise it returns empty string value.

Example

```
TrvsLineStart()

Loop()
{
        LineName = TrvsLineNext()
        if(LineName.length = 0) exit loop
}
```

See Also

TrvsLineStart(), AddLine()

TrvsDirectProxyStart()

The TrvsDirectProxyStart() starts iteration in the list of direct proxies added by using AddDirectProxySIP() function.

Syntax

```
boolean TrvsDirectProxyStart()
```

Parameters

No parameters.

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 GetVaxObjectError() method.

Example

```
TrvsDirectProxyStart()

Loop(GetDirectProxyCount())
{
	ProxyName = TrvsDirectProxyNext()
}
```

See Also

TrvsDirectProxyNext(), GetVaxObjectError(), AddDirectProxySIP()

TrvsDirectProxyNext()

The TrvsDirectProxyNext() iterates and returns proxy name.

Syntax

```
string TrvsDirectProxyNext()
```

Parameters

No parameters.

Return Value

On successful execution this function returns proxy-name value otherwise it returns empty string value.

Example

```
TrvsDirectProxyStart()

Loop()
{
        ProxyName = TrvsDirectProxyNext()
        if(ProxyName.length = 0) exit loop
}
```

See Also

TrvsDirectProxyStart(), AddDirectProxySIP()

TrvsCallSessionStart()

The TrvsCallSessionStart() starts iteration in the list of call-recording session.

Syntax

```
boolean TrvsCallSessionStart()
```

Parameters

No parameters.

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 GetVaxObjectError() method.

Example

```
TrvsCallSessionStart()

Loop(GetCallSessionCount())
{
        SessionId = TrvsCallSessionNext()
}
```

See Also

TrvsCallSessionNext(), GetVaxObjectError(), OnIncomingCall()

TrvsCallSessionNext()

The TrvsCallSessionNext() iterates and returns SessionId.

Syntax

```
integer TrvsCallSessionNext()
```

Parameters

No parameters.

Return Value

On successful execution this function returns SessionId value otherwise it returns -1 value.

Example

```
TrvsCallSessionStart()

Loop()
{
        SessionId = TrvsCallSessionNext()
        if(SessionId = -1) exit loop
}
```

See Also

TrvsDirectProxyStart(), OnIncomingCall()

GetUserCount()

The GetUserCount() returns total number of users added by using AddUser() function.

Syntax

```
integer GetUserCount()
```

Parameters

No parameters.

Return Value

The function returns total users count on its successful execution otherwise it return -1 and specific error code can be retrieved by calling GetVaxObjectError() method.

Example

```
AddUser("UserA", "PwdA", "0213")
AddUser("UserB", "PwdB", "3124")

TrvsUserStart()

Loop(GetUserCount())
{
	UserName = TrvsUserNext()
}
```

See Also

AddUser(), GetVaxObjectError(), TrvsUserStart(), TrvsUserNext()

GetLineCount()

The GetLineCount() returns total number of lines added by using AddLine() function.

Syntax

```
integer GetLineCount()
```

Parameters

No parameters.

Return Value

The function returns total lines count on its successful execution otherwise it return -1 and specific error code can be retrieved by calling GetVaxObjectError() method.

Example

```
AddLine("LineA", "8034", "8034", "8034", "9341", "sipsdk.com",
          "192.168.0.3", "", "0134")

AddLine("LineB", "7034", "7034", "7034", "0000", "sipsdk.com",
          "192.168.0.13", "", "20134")

TrvsLineStart()

Loop(GetLineCount())
{
       LineName = TrvsLineNext()
}
```

See Also

AddLine(), GetVaxObjectError(), TrvsLineStart(), TrvsLineNext()

GetDirectProxyCount()

The GetDirectProxyCount() returns total number of direct proxies added by using AddDirectProxySIP() function.

Syntax

```
integer GetDirectProxyCount()
```

Parameters

No parameters.

Return Value

The function returns total direct proxies count on its successful execution otherwise it return -1 and specific error code can be retrieved by calling GetVaxObjectError() method.

Example

```
AddDirectProxySIP("DirectA", "8003", "8004", "sip.sdk.com",
                    "192.168.0.20", 5060, "4213")

AddDirextProxySIP("DirectB", "7003", "7004", "sdk.com",
                    "66.77.88.99", 5060, "4213")

TrvsDirectProxyStart()

Loop(GetDirectProxyCount())
{
        ProxyName = TrvsDirectProxyNext()
}
```

See Also

AddDirectProxySIP(), GetVaxObjectError(), TrvsDirectProxyStart(), TrvsDirectProxyNext()

GetCallSessionCount()

The GetCallSessionCount() returns total number of call-recording sessions.

Syntax

```
integer GetCallSessionCount()
```

Parameters

No parameters.

Return Value

The function returns total call-recording session count on its successful execution otherwise it return -1 and specific error code can be retrieved by calling GetVaxObjectError() method.

Example

```
TrvsCallSessionStart()

Loop(GetCallSessionCount())
{
        SessionId = TrvsCallSessionNext()
}
```

See Also

GetVaxObjectError(), TrvsCallSessionStart(), TrvsCallSessionNext(), OnIncomingCall()

EXPORTED EVENTS

OnVaxErrorLog()

VaxVoIP REC SDK triggers OnVaxErrorLog() when execution of any function fails.

Please see **LIST OF ERROR CODES** for more details.

Syntax

```
OnVaxErrorLog(FuncName, ErrorCode, ErrorMsg)
```

Parameters

FuncName (string)
This parameter value specifies name of the function.

ErrorCode (integer)
This parameter value specifies error code.

ErrorMsg (string)
This parameter value specifies error text message.

Example

```
Result = Initialize("sipsdk.com", "9.7.11.17", 5060, "9.7.11.17", -1)

// if Initialize() fails then OnVaxErrorLog() triggers

OnVaxErrorLog(FuncName, ErrorCode, ErrorMsg)
{

}
```

See Also

GetVaxObjectError()

OnLineRegisterTrying()

VaxVoIP REC SDK triggers OnLineRegisterTrying() event if during the LINE registration process VaxVoIP REC SDK receives SIP provisional responses.

Syntax

```
OnLineRegisterTrying(LineName)
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

Example

```
OnLineRegisterTrying(LineName)
{
}
```

See Also

RegisterLine(), AddLine(), OnLineRegisterFailed(), OnLineRegisterSuccess()

OnLineRegisterFailed()

VaxVoIP REC SDK triggers OnLineRegisterFailed() event when registration of a LINE (SIP account settings) to external SIP server.

Syntax

```
OnLineRegisterFailed(
                        LineName,
                        StatusCode,
                        ReasonPhrase
                        )
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

StatusCode (integer)
: This parameter specifies SIP response status code (408, 403 etc). **LIST OF SIP RESPONSES**

ReasonPhrase (string)
: This parameter specifies SIP response reason phrase (Request Timeout, Forbidden etc).

Example

```
OnLineRegisterFailed(LineName, StatusCode, ReasonPhrase)
{
}
```

See Also

AddLine(), RegisterLine(), OnLineRegisterTrying(), OnLineRegisterSuccess()

OnLineRegisterSuccess()

VaxVoIP REC SDK triggers OnLineRegisterSuccess() event when LINE (SIP account settings) registration request (by RegisterLine() function) to external SIP server successfully completes.

Syntax

```
OnLineRegisterSuccess(LineName)
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

Example

```
OnLineRegisterSuccess(LineName)
{
}
```

See Also

AddLine(), RegisterLine(), OnLineRegisterTrying(), OnLineRegisterFailed()

OnLineUnRegisterTrying()

VaxVoIP REC SDK triggers OnLineUnRegisterTrying() event when it receives SIP provisional response from other SIP server during unregister process.

To unregister or disconnect VaxVoIP REC SDKfrom external third party SIP Server the UnRegisterLine() is used.

Syntax

```
OnLineUnRegisterTrying(LineName)
```

Parameters

LineName (integer)
: This parameter value specifies the unique line name to identify a specific line.

Example

```
OnLineUnRegisterTrying(LineName)
{
}
```

See Also

RegisterLine(), UnRegisterLine(), AddLine(), OnLineUnRegisterFailed(), OnLineUnRegisterSuccess()

OnLineUnRegisterFailed()

VaxVoIP REC SDK triggers OnLineUnRegisterFailed() event if a provided LINE fails to un-register from external SIP server.

Syntax

```
OnLineUnRegisterFailed(
                        LineName,
                        StatusCode,
                        ReasonPhrase
                        )
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

StatusCode (integer)
: This parameter specifies SIP response status code (408, 403 etc). **LIST OF SIP RESPONSES**

ReasonPhrase (string)
: This parameter specifies SIP response reason phrase (Request Timeout, Forbidden etc).

Example

```
OnLineUnRegisterFailed(LineName)
{
}
```

See Also

RegisterLine(), UnRegisterLine(), AddLine(), OnLineUnRegisterTrying(), OnLineUnRegisterSuccess()

OnLineUnRegisterSuccess()

VaxVoIP REC SDK triggers OnLineUnRegisterSuccess() event when line unregisters successfully from external SIP server.

VaxVoIP REC SDK calls UnRegisterLine() function to un-register/disconnect a line (SIP account settings) from external SIP server and if unregister request is successfully executes then OnLineUnRegisterSuccess() event triggers.

Syntax

```
OnLineUnRegisterSuccess(LineName)
```

Parameters

LineName (string)
: This parameter value specifies the unique line name to identify a specific line.

Example

```
OnLineUnRegisterSuccess(LineName)
{
}
```

See Also

RegisterLine(), UnRegisterLine(), AddLine(), OnLineUnRegisterTrying(), OnLineUnRegisterFailed()

OnUnRegisterUser()

VaxVoIP REC SDK triggers OnUnRegisterUser() event when it receives unregister request from any SIP client.

Please see **SIP REC CLIENT REGISTRATION FLOW** for more details.

Syntax

```
OnUnRegisterUser(UserLogin)
```

Parameters

UserLogin (string)
: This parameter specifies the user's login of SIP client.

Example

```
OnUnRegisterUser(UserLogin)
{
        RemoveUser(UserLogin)
}
```

See Also

OnRegisterUser(), RemoveUser(), AddUser()

OnRegisterUser()

The OnRegisterUser() event triggers when VaxVoIP REC SDK receives register request from any SIP REC supported client.

Please see **SIP REC CLIENT REGISTRATION FLOW** for more details.

Syntax

```
OnRegisterUser(
                UserLogin,
                Domain,
                UserAgentName,
                FromIP
                FromPort
                )
```

Parameters

UserLogin (string)
: This parameter specifies the user's login of SIP client.

Domain (string)
: This parameter specifies the domain.

UserAgentName (string)
: This parameter specifies the UserAgentName of SIP client.

FromIP (string)
: This parameter value specifies the from IP address.

FromPort (integer)
: This parameter specifies the from port number.

Example

```
OnRegisterUser(UserLogin, Domain, UserAgentName, FromIP, FromPort)
{
        AddUser()
}
```

See Also

OnUnRegisterUser(), AddUser(), RemoveUser()

OnRegisterUserSuccess()

The OnRegisterUserSuccess() event triggers when SIP REC client successfully registers to VaxVoIP REC SDK server.

Please see **SIP REC CLIENT REGISTRATION FLOW** for more details.

Syntax

```
OnRegisterUserSuccess(UserLogin)
```

Parameters

UserLogin (string)
This parameter specifies the user's login of SIP client.

Example

```
OnRegisterUserSuccess(UserLogin)
{
}
```

See Also

OnRegisterUser(), OnRegisterUserFailed()

OnRegisterUserFailed()

The OnRegisterUserFailed() event triggers when SIP REC client fails to register with VaxVoIP REC SDK server.

Syntax

```
OnRegisterUserFailed(UserLogin)
```

Parameters

UserLogin (string)
: This parameter specifies the user's login of SIP client.

Example

```
OnRegisterUserFailed(UserLogin)
{
        RemoveUser(UserLogin)
}
```

See Also

OnRegisterUser(), OnRegisterUserSuccess(), RemoveUser()

OnIncomingCall()

The OnIncomingCall() event triggers when VaxVoIP REC SDK receives a SIP REC protocol based call request.

Syntax

```
OnIncomingCall(
        SessionId,
        CallerName,
        CallerId,
        CalleeName,
        CalleeId,
        RecXML,
        FromPeerType,
        FromPeerName,
        UserAgentName,
        FromIP,
        FromPort
        )
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

CallerName (string)
: This parameter specifies the caller name.

CallerId (string)
: This parameter specifies a unique identification of caller.

CalleeName (string)
: This parameter specifies the callee name.

CalleeId (string)
: This parameter specifies a unique identification of callee.

RecXML (string)
: This parameter specifies a XML part of SIP/SDP packet.

FromPeerType (integer)
: This parameter value specifies the type of From-Peer.

  3001 = User
  3002 = Line
  3003 = DirectProxy

FromPeerName (string)
: This parameter value specifies the name of From-Peer.

UserAgentName (string)
This parameter value specifies the name of UserAgent.

FromIP (string)
This parameter value specifies the From IP address.

FromPort (integer)
This parameter specifies the From port number.

Example

```
OnIncomingCall(SessionId, CallerName, CallerId, CalleeName, CalleeId,
                RecXML, FromPeerType, FromPeerName, UserAgentName,
                FromIP, FromPort)
{
    AcceptCallSession(SessionId, 10)
}
```

See Also

AcceptCallSession(), OnCallSessionConnected, OnCallSessionFailed()

OnCallSessionConnected()

The OnCallSessionConnected() event triggers when VaxVoIP REC SDK successfully established a call-recording session with SIP REC client/third party server.

Syntax

```
OnCallSessionConnected(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

Example

```
OnCallSessionConnected(SessionId)
{
	RecordWaveStartToCallSession(SessionId, "RecordCall.wav")
}
```

See Also

AcceptCallSession(), OnIncomingCall()

OnCallSessionLost()

The OnCallSessionLost() event triggers when VaxVoIP REC SDK does not receive voice data for define interval of time.

Syntax

```
OnCallSessionLost(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

Example

```
OnCallSessionLost(SessionId)
{
}
```

See Also

VoiceSessionLost()

OnCallSessionHangup()

The OnCallSessionHangup() event triggers when remote party hangup the disconnects the SIP REC call.

Syntax

```
OnCallSessionHangup(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

Example

```
OnCallSessionHangup(SessionId)
{
}
```

See Also

AcceptCallSession(), OnCallSessionConnected(), OnIncomingCall()

OnCallSessionTimeout()

The OnCallSessionTimeout() event triggers when VaxVoIP REC SDK fails to establish a call-recording session and does not receive the response from SIP REC client within time period specified in AcceptCallSession() method.

Syntax

```
OnCallSessionTimeout(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

Example

```
OnCallSessionTimeout(SessionId)
{
}
```

See Also

AcceptCallSession()

OnCallSessionCancelled()

The OnCallSessionCancelled() event triggers when caller cancels the incoming REC call prior to its acceptance by AcceptCallSession() function.

Syntax

```
OnCallSessionCancelled(SessionId)
```

Parameters

SessionId (integer)
This parameter specifies an identification of a call-recording session.

Example

```
OnCallSessionCancelled(SessionId)
{
}
```

See Also

OnOutgoingDiagnosticLog()

The OnOutgoingDiagnosticLog() event triggers when VaxVoIP REC SDK sends a SIP packet. This event can be used for logging and monitoring of outbound SIP messages.

Syntax

```
OnOutgoingDiagnosticLog(
                        MsgSIP,
                        ToIP,
                        ToPort
                        )
```

Parameters

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

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

ToPort (integer)
: This parameter value specifies the To port number.

Example

```
OnOutgoingDiagnosticLog(MsgSIP, ToIP, ToPort)
{

}
```

See Also

DiagnosticLogSIP(), OnIncomingDiagnosticLog()

OnIncomingDiagnosticLog()

The OnIncomingDiagnosticLog() event triggers when VaxVoIP REC SDK receives a SIP packet. This event can be used for logging and monitoring of inbound SIP messages.

Syntax

```
OnIncomingDiagnosticLog(
                        MsgSIP,
                        FromIP,
                        FromPort
                        )
```

Parameters

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

```
OnIncomingDiagnosticLog(MsgSIP, FromIP, FromPort)
{

}
```

See Also

DiagnosticLogSIP(), OnOutgoingDiagnosticLog()

OnVaxTeleTick()

The OnVaxTeleTick() event triggers after a specified time interval set by StartVaxTeleTick() function.

Syntax

```
OnVaxTeleTick(TickId)
```

Parameters

TickId (integer)
This parameter specifies the unique tick identification.

Example

```
OnVaxTeleTick(TickId)
{
}
```

See Also

StartVaxTeleTick(), StopVaxTeleTick()

OnCallSessionCreated()

The OnCallSessionCreated() event triggers when VaxVoIP REC SDK creates/allocates a call-recording session internally.

Syntax

```
OnCallSessionCreated(
                        SessionId,
                        ReasonCode
                        )
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

ReasonCode(integer)
: This parameter specifies the reason due to which the call-recording session is created.

- INCOMING_CALL 1001
- OUTGOING_CALL 1002
- MERGED 1003
- TRANSFERED 1004

Example

```
OnCallSessionCreated(SessionId, ReasonCode)
{
        CCustomCallSessionData objData = new CCustomCallSessionData()
        SetCallSessionData(SessionId, objData)
}
```

See Also

OnCallSessionClosed(), SetCallSessionData()

OnCallSessionClosed()

The OnCallSessionClosed() event triggers when VaxVoIP REC SDK closes/destroys a call-recording session internally.

Syntax

```
OnCallSessionClosed(SessionId, ReasonCode)
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

ReasonCode (integer)
: This parameter specifies the reason due to which the call-recording session is closed.

- HANGUP 2001
- SESSION_LOST 2002
- MERGED 2003
- TRANSFERED 2004
- REJECTED 2005
- FAILED 2006
- CANCELLED 2007
- TIMEOUT 2008
- CLOSED 2009

Example

```
OnCallSessionClosed(SessionId, ReasonCode)
{
	CCustomCallSessionData objData = GetCallSessionData(SessionId)
	objData = nil
}
```

See Also

OnCallSessionCreated(), SetCallSessionData(), GetCallSessionData()

OnCallSessionRecordedPCM()

The OnCallSessionRecordedPCM() event starts triggering when recording on a specific call-recording session starts by using RecordWaveStartToCallSession() with value empty string to the second parameter.

e.g. RecordWaveStartToCallSession(SessionId, "")

OnCallSessionRecordedPCM() can also be used to record or save conversation into .mp3 or other media files by using third-party libraries. In this event, application receives voice data PCM (8000Hz, 16bit, Mono), pass it to third-party library and library stores it in the required media file format. Please contact the third-party vendor for further details.

Syntax

```
OnCallSessionRecordedPCM(SessionId, DataPCM, SizePCM, TypePCM)
```

Parameters

SessionId (integer)
: This parameter specifies an identification of a call-recording session.

DataPCM (array)
: The value of this parameter specifies the voice PCM data.

SizePCM (integer)
: The value of this parameter specifies the size of voice PCM data.

TypePCM (integer)
: The value of this parameter specifies the type of voice PCM data.

  0 = 8000Hz, 16Bit, Mono

Example

```
OnCallSessionRecordedPCM(SessionId, DataPCM, SizePCM, TypePCM)
{
	// Pass PCM data to third party MP3 library engine.
}
```

See Also

RecordWaveStartToCallSession()

LIST OF ERROR CODES

200	VaxVoIP REC SDK failed to initialize RTP Socket.
201	VaxVoIP REC SDK failed to allocate RTP port or provided value of RTP listen IP is incorrect.
202	VaxVoIP REC SDK failed to create RTP task manager.
203	Failed to start media thread.
204	Unable to create RTP communication manager.
205	Unable to use RTP communication manager.
206	Unable/failed to open file.
207	Unable to read file.
208	Incorrect wave file format, please use 8000Hz, 16bit, mono uncompressed wave file.
209	Invalid play wave ID.
210	Unable to start recording manager.
211	Selected call is not in the voice session.
212	Failed to initialize VaxVoIP REC SDK object.
213	Unable to create SDP body.
214	Unable to create INVITE request.
215	Error to initialize SIP communication layer.
216	VaxVoIP REC SDK failed to open SIP listen port or provided value of SIP listen IP is incorrect.
217	Failed to create SIP task manager.
218	Unable to create REGISTER request.
219	Failed to create UN-REGISTER request.
220	Failed to create BYE request.
221	Unable to create CANCEL request.
222	Invalid Call-Id or call does not exist.
223	Invalid session-Id or session does not exist.
224	Provided login does not exist.
225	Login length is incorrect.
226	Provided line does not exist.
227	Unable to send SIP response.
228	Invalid SIP response code, please check SIP RFC 3261.
229	Error to create SIP message.
230	Provided line already exist.
231	Incorrect Reg-Id or Reg-Id does not exist.
232	Error to create SIP response.
233	User is not registered.
234	Invalid codec OR there is no codec found for voice streaming.
235	Invalid DTMF type.
236	Other party does not support RFC2833 DTMF digits.
237	Error to create REFER request to transfer the call.
238	Error to create event handler.
239	Invalid license key.
240	Invalid digit for DTMF.
241	Invalid proxy URI.

242	Line is not registered.
243	Invalid to URI.
244	Unable to perform desired operation, call is not connected.
245	Unable to perform desired operation on connected call.
246	Direct communication is not supported
247	Invalid parameter(s) value.
248	Invalid chat message Id.
249	Invalid subscribe Id.
250	Failed to allocate a unique Id.
251	Invalid Id.
252	Provided Id or value already exist.
253	Provided Id or value does not exist.
254	Failed to create call-recording session.
255	Failed to enable DTMF detection.
256	Failed to process transfer request.
257	Listen IP already exist.
258	Listen IP does not exist in the Server Key.
259	Provided Channel of a call-recording session is not free/available.

LIST OF SIP RESPONSES (SIP RFC 3261)

Provisional responses 1xx			
100	Trying	180	Ringing
181	Call Is Being Forwarded	182	Queued
183	Session Progress		
Redirection 3xx			
300	Multiple Choices	301	Moved Permanently
302	Moved Temporarily	305	Use Proxy
380	Alternative Service		
Request Failure 4xx			
400	Bad Request	401	Unauthorized
402	Payment Required	403	Forbidden
404	Not Found	405	Method Not Allowed
406	Not Acceptable	407	Proxy Authentication Required
408	Request Timeout	410	Gone
413	Request Entity Too Large	414	Request-URI Too Long
415	Unsupported Media Type	416	Unsupported URI Scheme
420	Bad Extension	421	Extension Required
423	Interval Too Brief	480	Temporarily Unavailable
481	Call/Transaction Does Not Exist	482	Loop Detected
483	Too Many Hops	484	Address Incomplete
485	Ambiguous	486	Busy Here
487	Request Terminated	488	Not Acceptable Here
491	Request Pending	493	Undecipherable
Server Failure 5xx			
500	Server Internal Error	501	Not Implemented
502	Bad Gateway	503	Service Unavailable
504	Server Time-out	505	Version Not Supported
513	Message Too Large		
Global Failures 6xx			
600	Busy Everywhere	603	Decline
604	Does Not Exist Anywhere	606	Not Acceptable

SIP REC CLIENT REGISTRATION FLOW

Please see SIP REC CLIENT REGISTRATION FLOW document for further details.

SIP CALL RECORDING SESSION FLOW

Please see SIP CALL RECORDING SESSION FLOW document for further details.